Electronic Communications of the EASST
Volume 30 (2010)

International Colloquium on Graph and Model Transformation On the occasion of the 65th birthday of Hartmut Ehrig (GraMoT 2010)

Expressiveness of graph conditions with variables

Annegret Habel and Hendrik Radke

18 pages

Guest Editors: Claudia Ermel, Hartmut Ehrig, Fernando Orejas, Gabriele Taentzer
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Expressiveness of graph conditions with variables

Annegret Habel[1] **and Hendrik Radke**[2*]

[1] habel@informatik.uni-oldenburg.de
[2] radke@informatik.uni-oldenburg.de
Carl von Ossietzky Universität Oldenburg, Germany

Abstract: Graph conditions are very important for graph transformation systems and graph programs in a large variety of application areas. Nevertheless, non-local graph properties like "there exists a path", "the graph is connected", and "the graph is cycle-free" are not expressible by finite graph conditions. In this paper, we generalize the notion of finite graph conditions, expressively equivalent to first-order formulas on graphs, to finite HR^+ graph conditions, i.e., finite graph conditions with variables where the variables are place-holders for graphs generated by a hyperedge replacement system. We show that graphs with variables and replacement morphisms form a weak adhesive HLR category. We investigate the expressive power of HR^+ graph conditions and show that finite HR^+ graph conditions are more expressive than monadic second-order graph formulas.

Keywords: Graph conditions, graphs with variables, hyperedge replacement systems, monadic-second order formulas, weak adhesive HLR categories.

1 Introduction

Graph transformation systems have been studied extensively and applied to several areas of computer science [Roz97, EEKR99, EKMR99]. Graph conditions, i.e., graph constraints and application conditions, studied, e.g., in [EH86, HHT96, HW95, KMP05, EEHP06, HP09], are very important for graph transformation systems in a large variety of application areas. Graph conditions are an intuitive, graphical, yet precise formalism, well-suited for describing structural properties. Moreover, finite graph conditions and first-order graph formulas are expressively equivalent [HP09]. Unfortunately, non-local graph properties like "there exists a path", "the graph is connected", and "the graph is cycle-free" are not expressible by first-order graph formulas [Cou90a, Cou97] and finite graph conditions. They only can be expressed by infinite graph conditions.

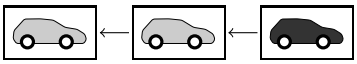
In this paper, we generalize the concept of graph conditions to graph conditions with variables where the variables are place-holders for graphs generated by a hyperedge replacement (HR) system. By the HR system, we obtain a finite description of a set of graphs that is infinite in general, like the set of all paths. We state that graphs with variables and replacement morphisms form a weak adhesive HLR category. We introduce HR conditions as conditions over graphs with variables (together with a HR system) and HR^+ conditions as extensions of HR conditions by

* This work is supported by the German Research Foundation (DFG) under grant GRK 1076/1 (Graduate School on Trustworthy Software Systems).

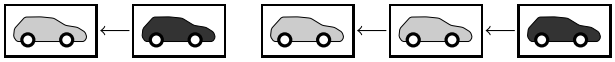
conditions of the form $x^\circ \sqsubseteq \boxed{X}$, related to monadic second order (MSO) formulas $x \in X$. It turns out that the validation problem for finite HR^+ conditions and graphs is decidable. We show that MSO graph formulas can be expressed by equivalent HR^+ conditions and that HR conditions can express second-order (SO) graph formulas.

The usefulness of HR conditions is illustrated by an example of a car platooning maneuver protocol.

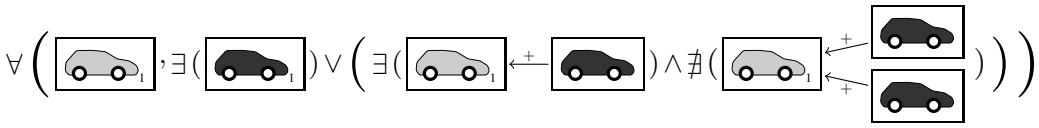
Example 1 (Car platooning) *We study "a prototypical instance of a dynamic communication system", originally taken from the California Path project [HESV91]. It represents a protocol for cars on a highway that can organize themselves into platoons, by driving close together, with the aim to conserve space and fuel. A car platoon is modeled as a directed graph where the nodes represent the cars and the direct edges the direct follower relation. Additionally the leader of a car platoon is marked by the "dark grey" color (resp. by a loop) where the color is used in figures, representing the loop in the formal definition.*



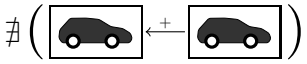
A car platooning state graph consists of zero or more car platoons.



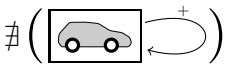
Car platooning operations like splitting a car platoon in two, or joining two car platoons into a single one can be described by graph replacement rules. When performing these operations, certain car platooning properties have to be satisfied.
(1) Every follower has a unique leader:



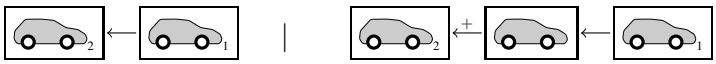
(2) Leaders are not connected:



(3) The car platooning state is circle-free:



where the HR *system* $+$ *::=*



generates the set of all non-empty paths from 1 to 2. The car platooning properties are described by HR *conditions. Bauer [Bau06] and Pennemann [Pen09] model the follower relation with respect to the leader, but not the order of the followers.* HR *conditions allow to express path conditions as in the car platooning example.*

The paper is organized as follows: In Section 2, we introduce graphs with variables and state that graphs with variables and replacement morphisms form a category. In Section 3, we generalize graph conditions to HR and HR^+ conditions, i.e., graph conditions with variables equipped

with a hyperedge replacement (HR) system, and present a number of examples for HR conditions. In Section 4, we show that the validity problem for HR^+ graph conditions is decidable. In Section 5, we investigate the expressiveness of HR^+ conditions. In Section 6, we present some related concepts. A conclusion including further work is given in Section 7. $\boxed{X}$

2 Graphs with variables

Graphs with variables consist of nodes, edges, and hyperedges. Edges have one source and one target and are labeled by a symbol of an alphabet; hyperedges have an arbitrary sequence of attachment nodes and are labeled by variables.

Definition 1 (Graphs with variables) Let $\mathrm{C} = \langle \mathrm{C_V}, \mathrm{C_E}, \mathrm{X} \rangle$ be a fixed, finite label alphabet where X is a set of variables with a mapping $\mathrm{rank}\colon \mathrm{X} \to \mathbb{N}_0$ defining the rank of each variable. A *graph with variables*, short X-*graph*, over C is a system $G = (\mathrm{V}_G, \mathrm{E}_G, \mathrm{Y}_G, \mathrm{s}_G, \mathrm{t}_G, \mathrm{att}_G, \mathrm{lv}_G, \mathrm{le}_G, \mathrm{ly}_G)$ consisting of finite sets V_G, E_G, and Y_G of *nodes* (or *vertices*), *edges*, and *hyperedges*, *source* and *target functions* $\mathrm{s}_G, \mathrm{t}_G\colon \mathrm{E}_G \to \mathrm{V}_G$, an *attachment function* $\mathrm{att}_G\colon \mathrm{Y}_G \to \mathrm{V}_G^*$, and *labeling functions* $\mathrm{lv}_G\colon \mathrm{V}_G \to \mathrm{C_V}$, $\mathrm{le}_G\colon \mathrm{E}_G \to \mathrm{C_E}$, $\mathrm{ly}\colon \mathrm{Y}_G \to \mathrm{X}$ such that, for all $y \in \mathrm{Y}_G$, $|\mathrm{att}_G(y)| = \mathrm{rank}(\mathrm{ly}_G(y))$. $\mathscr{G}_\mathrm{X}$ denotes the set of all X-graphs. For $\mathrm{Y}_G = \emptyset$, G is a *graph*; $\mathscr{G}$ denotes the set of all graphs.

Remark 1 *The definition extends the well-known definition of graphs [Ehr79] by the concept of hyperedges in the sense of [Hab92]. X-graphs also may be seen as special hypergraphs where the set of hyperedges is divided into a set of edges labelled with terminal symbols and a set of hyperedges labelled by nonterminal symbols.*

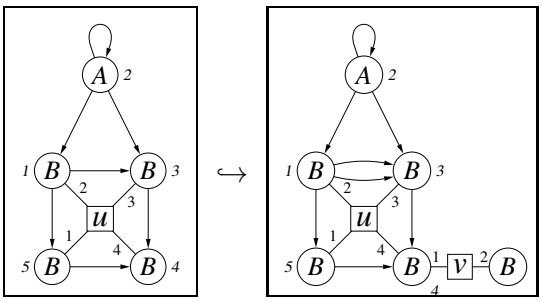
We extend the definition of graph morphisms to the case of graphs with variables.

Definition 2 (Graph morphisms with variables) A *morphism over graphs with variables*, short *(X-graph) morphism*, $g\colon G \to H$ consists of functions $g_\mathrm{V}\colon \mathrm{V}_G \to \mathrm{V}_H$, $g_\mathrm{E}\colon \mathrm{E}_G \to \mathrm{E}_H$, and an injective function $g_\mathrm{Y}\colon \mathrm{Y}_G \to \mathrm{Y}_H$ that preserve sources, targets, attachment nodes, and labels, that is, $\mathrm{s}_H \circ g_\mathrm{E} = g_\mathrm{V} \circ \mathrm{s}_G$, $\mathrm{t}_H \circ g_\mathrm{E} = g_\mathrm{V} \circ \mathrm{t}_G$, $\mathrm{att}_H = g_\mathrm{V}^* \circ \mathrm{att}_G$, $\mathrm{lv}_H \circ g_\mathrm{V} = \mathrm{lv}_G$, $\mathrm{le}_H \circ g_\mathrm{E} = \mathrm{le}_G$, and $\mathrm{ly}_H \circ g_\mathrm{Y} = \mathrm{ly}_G$.[1] For $\mathrm{Y}_H = \emptyset$, g is a *(graph) morphism*. A morphism g is *injective* (*surjective*) if g_V, g_E, and g_Y are injective (surjective), and an *isomorphism* if it is both injective and surjective. In the latter case G and H are *isomorphic*, which is denoted by $G \cong H$. The *composition* $h \circ g$ of g with a morphism $h\colon H \to M$ consists of the composed functions $h_\mathrm{V} \circ g_\mathrm{V}$, $h_\mathrm{E} \circ g_\mathrm{E}$, and $h_\mathrm{Y} \circ g_\mathrm{Y}$. For an X-graph G, the *identity* $\mathrm{id}_G\colon G \to G$ consists of the identities $\mathrm{id}_{G\mathrm{V}}$, $\mathrm{id}_{G\mathrm{E}}$, and $\mathrm{id}_{G\mathrm{Y}}$ on G_V, G_E, and G_Y, respectively.

Example 2 *Consider the X-graphs G and H over the label alphabet $\mathrm{C} = \langle \{A, B\}, \{\square\}, \mathrm{X} \rangle$ below where the symbol $\square$ stands for the invisible edge label and is not drawn and $\mathrm{X} = \{u, v\}$ is a set of variables that have rank 4 and 2, respectively. The X-graph G contains five nodes with the labels A and B, respectively, seven edges with label $\square$ which is not drawn, and one hyperedge of rank 4 with label u. Additionally, the X-graph H contains a node, an edge, and a hyperedge of rank 2*

[1] For a mapping $g\colon A \to B$, the free symbolwise extension $g^*\colon A^* \to B^*$ is defined by $g^*(a_1 \ldots a_k) = g(a_1) \ldots g(a_k)$ for all $k \in \mathbb{N}$ and $a_i \in A$ $(i = 1, \ldots, k)$.

with label v.



G *H*

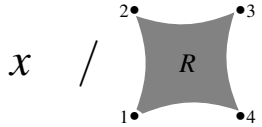
The drawing of X*-graphs combines the drawing of graphs in [Ehr79] and the drawing of hyperedges in [Hab92, DHK97]: Nodes are drawn by circles carrying the node label inside, edges are drawn by arrows pointing from the source to the target node and the edge label is placed next to the arrow, and hyperedges are drawn as boxes containing the label, which are connected by lines to their attachment nodes* $v_1,\ldots,v_k$ *so that the line to* v_i *has the number* i *written next to it. For simplicity, binary hyperedges also can be drawn like edges. Arbitrary morphisms are drawn by usual arrows "*$\to$*"; the use of "*$\hookrightarrow$*" indicates an injective morphism. The actual mapping of elements is conveyed by indices, if necessary.*

Hyperedges do not only play a static part as building blocks of X-graphs, but also a more dynamic part as place holders for graphs. Before an X-graph can take the place of a hyperedge, it needs some preparation. While a hyperedge is attached to a sequence of attachment nodes, an X-graph has to be equipped with a sequence of nodes.

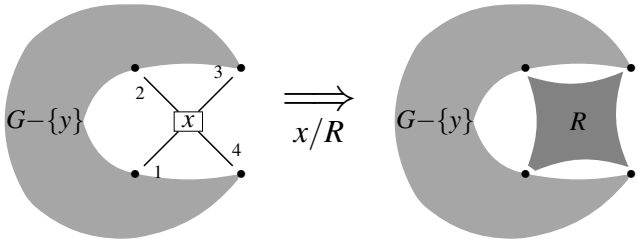
Definition 3 (Pointed graphs with variables) A *pointed* X-graph $\langle R, \text{points}_R \rangle$ is an X-graph R together with a sequence $\text{points}_R = v_1 \ldots v_n$ of pairwise distinct nodes from R. We write $\text{rank}(R)$ for the number n of nodes. For $x \in X$ with $\text{rank}(x) = n$, $x^\bullet$ denotes the pointed X-graph with the nodes $v_1,\ldots,v_n$, one hyperedge labeled by x and attached to $v_1 \ldots v_n$, and sequence $v_1 \ldots v_n$. $\mathscr{G}_X^\bullet$ denote the set of all pointed X-graphs.

Variables are replaced by graphs generated by a hyperedge replacement system.

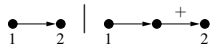
Definition 4 (HR systems) A *hyperedge replacement (HR) system* $\mathscr{R}$ over X is a finite set of replacement pairs of the form x/R where x is a variable in X and R a pointed X-graph with $\text{rank}(x) = \text{rank}(R)$.



Given an X-graph G, the application of the replacement pair x/R to a hyperedge y with label x proceeds in two steps: (1) Remove the hyperedge y from G, yielding the X-graph $G-\{y\}$. (2) Construct the disjoint union $(G-\{y\})+R$ and fuse the i^{th} node in $\text{att}_G(y)$ with the i^{th} attachment point of R, for $i = 1,\ldots,\text{rank}(y)$, yielding the X-graph H.



An X-graph G *directly derives* H by x/R applied to y, denoted by $G \Rightarrow_{x/R,y} H$ or $G \Rightarrow_{\mathscr{R}} H$ provided that $x/R \in \mathscr{R}$. A sequence of direct derivations $G \Rightarrow_{\mathscr{R}} \ldots \Rightarrow_{\mathscr{R}} H$ is called a *derivation* from G to H, denoted by $G \Rightarrow^*_{\mathscr{R}} H$. For every variable x in X, $\mathscr{R}(x) = \{G \in \mathscr{G} \mid x^\bullet \Rightarrow^*_{\mathscr{R}} G\}$ denotes the set of all graphs derivable from x° by $\mathscr{R}$.

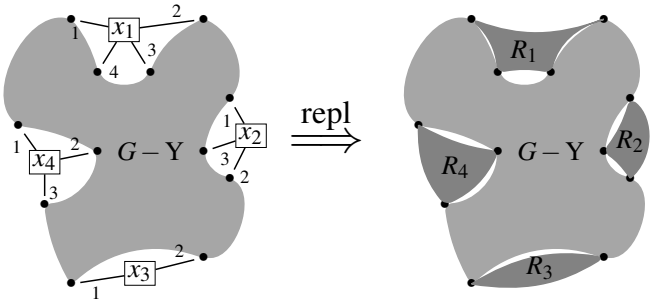
Example 3 *The* HR *system* $\mathscr{R}$ *with the rules* $+ ::=$  *given in Backus-Naur form (presented in the introduction) generates the set of all directed paths from node 1 to node 2.*

Assumption *In the following, let* $\mathscr{R}$ *be a fixed* HR *system.*

Given an X-graph, there are two ways to get rid of variables: (A) Apply a substitution according to a mapping from the variables, i.e., labels of the hyperedges, to graphs. Then same-labelled hyperedges are replaced by the same graph. (B) Apply a base for replacement according to mapping from the hyperedges to graphs. Then, same-labelled hyperedges may be replaced differently. In [PH96] and [Pra04], hyperedges are substituted by graphs. In the following, hyperedges are replaced by graphs. By our opinion, replacement is the more adequate.

HR systems are used to define the set of admissible replacements.

Definition 5 (Replacement) Let G be an X-graph and $\mathrm{Y} \subseteq \mathrm{Y}_G$ a set of hyperedges to be replaced. A mapping $\mathrm{repl}\colon \mathrm{Y} \to \mathscr{G}^\bullet$ is a *base for replacement* in G if, for all y in Y, $\mathrm{repl}(y)$ is in $\mathscr{R}(\mathrm{ly}(y))$. The replacement of Y in G by repl, denoted by $\mathrm{repl}(G)$, is obtained from G by simultaneously replacing all hyperedges y in Y by $\mathrm{repl}(y)$. If $\mathrm{repl}(G)$ is H, up to isomorphism, we write $G \overset{\mathrm{repl}}{\Longrightarrow} H$.



Fact 1 (Commutativity)
(1) Given $G' \overset{\mathrm{repl}}{\Longleftarrow} G \overset{g'}{\Rightarrow} H'$, *there exists a pushout* $G' \overset{g}{\Rightarrow} H \overset{\mathrm{repl}'}{\Longleftarrow} H'$.
(2) Given $G' \overset{g}{\Rightarrow} H \overset{\mathrm{repl}'}{\Longleftarrow} H'$, *there exists a pullback* $G' \overset{\mathrm{repl}}{\Longleftarrow} G \overset{g'}{\Rightarrow} H'$.

(3) Given $G \xrightarrow{g'} H' \overset{\text{repl}'}{\Longrightarrow} H$*, there exists a pushout complement* $G \overset{\text{repl}}{\Longrightarrow} G' \xrightarrow{g} H$*.*

$$\begin{array}{ccc} G & \xrightarrow{g'} & H' \\ {\scriptstyle \text{repl}}\Downarrow & (1) & \Downarrow{\scriptstyle \text{repl}'} \\ G' & \xrightarrow[g]{} & H \end{array}$$

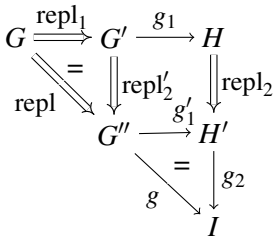
Construction (1) We make use of the fact that every item in an X-graph G' is in $G-\mathrm{Y}$ or in $\text{repl}(Y^\bullet)^-$ obtained from $\text{repl}(Y^\bullet)$ by removing all attachment nodes of hyperedges in Y. Let $\text{repl}'\colon g'_{\mathrm{Y}}(\mathrm{Y}) \to \mathscr{G}^\bullet$ be a base for replacement in H' with $\text{repl}' = \text{repl} \circ {g'_{\mathrm{Y}}}^{-1}$ and $g\colon G' \to H$ be the morphism with $g = g'[G-\mathrm{Y}]$ and $g = \text{id}[\text{repl}(Y^\bullet)^-]$[2]. (2) Let $\text{repl}\colon {g_{\mathrm{Y}}}^{-1}(\mathrm{Y}') \to \mathscr{G}^\bullet$ be a base for replacement in G with $\text{repl} = \text{repl}' \circ g$ and $g'\colon G \to H'$ be the morphism with $g'{=}g[G{-}\mathrm{Y}]$ and $g'{=}\text{id}[\mathrm{Y}]$. (3) Let $\text{repl}\colon {g'_{\mathrm{Y}}}^{-1}(Y') \to \mathscr{G}^\bullet$ be a base for replacement in G with $\text{repl} = \text{repl}' \circ g'$ and $g\colon G' \to H$ be the morphism with $g{=}g'[G{-}\mathrm{Y}]$, and $g = \text{id}[\text{repl}(Y)^-]$.

Remark 2 *The injectivity of* g_{Y} *guarantees the well-definedness of the bases for replacement.*

Replacement morphisms consist of a base for replacement and a graph morphism.

Definition 6 (Replacement morphisms) A replacement morphism $\langle \text{repl}, g\rangle\colon G \twoheadrightarrow H$ consists of a base for replacement repl in G and a graph morphism $g\colon \text{repl}(G) \to H$. It is *injective up to replacement* if the restriction $g|_{G-\mathrm{Y}_G}$[3] is injective and *injective* if g is injective.

Definition 7 (Identity and Composition) The replacement morphism $\langle \emptyset, \text{id}_G\rangle$ with empty base for replacement and identity $\text{id}_G\colon G \hookrightarrow G$ is called *identity*. Given replacement morphisms $\langle \text{repl}_1, g_1\rangle\colon G \twoheadrightarrow H$ and $\langle \text{repl}_2, g_2\rangle\colon H \twoheadrightarrow I$, the *composition* $\langle \text{repl}, g\rangle\colon G \twoheadrightarrow I$ is given by $\text{repl} = \text{repl}'_2 \circ \text{repl}_1$ and $g = g_2 \circ g'_1$.



Fact 2 (Composition) *The composition of replacement morphisms is a replacement morphism.*

X-graphs and replacement morphisms form a category; this category with the class of all injective graph morphisms is weak adhesive HLR.

[2] For graph morphisms $g_i\colon G_i \to H$ with $G \subseteq G_i$ $(i = 1,2)$, $g_1{=}g_2[G]$ abbreviates $g_1|_G = g_2|_G$.
[3] For a graph morphism $g\colon G \to H$, $g|_{G-\mathrm{Y}_G}$ denotes the restriction of g to $G-\mathrm{Y}_G$.

Theorem 1 (XGraphs is weak adhesive HLR) *X-graphs and replacement morphisms form the category* XGraphs. *The category* $\langle \mathrm{XGraphs}, \mathscr{M} \rangle$ *of X-graphs with the class* $\mathscr{M}$ *of all injective graph morphisms is a weak adhesive HLR and has an* $\mathscr{M}$*-initial X-graph and epi-*$\mathscr{M}$ *factorization.*

Proof. By associativity and identity of replacements and graph morphisms and the definition of weak adhesive HLR. □

3 HR and HR$^+$ conditions

Graph conditions are a well-known concept for describing graph properties in a graphical way by graphs and graph morphisms (see e.g. [EH86, HHT96, HW95, KMP05, EEHP06, HP09]). In the following, we generalize the concept to HR and HR$^+$ conditions. HR conditions are conditions in the category of X-graphs where the variables may be replaced by graphs generated by a hyperedge replacement (HR) system. HR$^+$ conditions are extensions of HR conditions by conditions of the form $x^\circ \sqsubseteq \boxed{X}$ with the meaning "is included in", a counterpart to the MSO formulas of the form $x \in X$ with the meaning "is element in".

Definition 8 (HR and HR$^+$ conditions) HR-*conditions*, short *conditions*, are inductively defined as follows: For an X-graph P, true is a condition over P. For every morphism $a\colon P \to C$ and every condition c over C, $\exists(a,c)$ is a condition over P. For conditions c, c_i over P with $i \in I$ (for all index sets I), $\neg\, c$ and $\wedge_{i\in I} c_i$ are conditions over P. HR$^+$ *conditions* are defined by adding: For X-graphs $x^\circ, \boxed{X} \subseteq P$, the expression $x^\circ \sqsubseteq \boxed{X}$ is a condition over P where, for an item $x \in \mathrm{V}_P + \mathrm{E}_P$, x° denotes the graph induced by x, i.e., the graph with the single node x and the edge x together with its source and target, respectively. A condition is *finite*, if every conjunction is finite.

Notation The expression false abbreviates the condition $\neg$true. For an index set I, the expression $\vee_{i\in I} c_i$ abbreviates the condition $\neg \wedge_{i\in I} \neg c_i$, $c_1 \Rightarrow c_2$ abbreviates $\neg c_1 \vee c_2$, $\exists a$ abbreviates $\exists(a, \mathrm{true})$, and $\forall(a,c)$ abbreviates $\neg\exists(a,\neg c)$. For a morphism $a\colon P \to C$ in a condition, we just depict C, if P can be unambiguously inferred, i.e. for constraints over the empty graph $\emptyset$ and for conditions over some left- or right-hand side of a rule. E.g., the condition $\exists(\emptyset \to$  $))$ is abridged $\exists($ $))$.

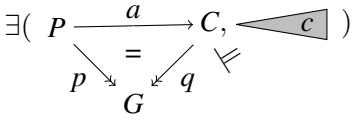
Remark 3 *The definition generalizes the definitions in [HHT96, HW95, KMP05, EEHP06].* HR*-conditions are conditions over* X*-graphs [HP09]. Variables in finite conditions may be replaced by graphs generated by a corresponding* HR *system. In this way, an infinite number of conditions is expressed by a finite* HR *condition. I*HR$^+$ *conditions extend* HR*-conditions by subconditions the form* $x^\circ \sqsubseteq \boxed{X}$. *Typical examples are* $\bullet_x \sqsubseteq \boxed{X}$ *with* $X ::= \emptyset \mid \boxed{X}\bullet$ *and* $\bullet\!\!\rightarrowtail\!\!\bullet \sqsubseteq \boxed{X}$ *with* $X ::= \emptyset \mid \boxed{X}\bullet\!\!\rightarrowtail\!\!\bullet$. *With this extension, there is a transformation of* MSO *formulas into equivalent* HR$^+$ *conditions (see Theorem 3).*

Remark 4 *For conditions, the underlying* HR *system* $\mathscr{R}$ *is important: For a condition* c*, this may be expressed explicitly by the pair* $\langle c, \mathscr{R} \rangle$ *meaning that the variables in* c *may be replaced*

by graphs derived by $\mathscr{R}$.

Let $\mathscr{B}'$ denote the class of all replacement morphisms, injective up to replacement, and $\mathscr{A}'$ be the class of all replacement morphisms.

Definition 9 ($\mathscr{B}'$- and $\mathscr{A}'$-satisfiability) *($\mathscr{B}'$-) satisfiability* of conditions are inductively defines as follows: Every replacement morphism *satisfies* true. A replacement morphism $p\colon P \twoheadrightarrow G$ *satisfies* $\exists(a,c)$ if there exists a replacement morphism q in $\mathscr{B}'$ such that $q \circ a = p$ and q satisfies c.



A replacement morphism $p\colon P \twoheadrightarrow G$ *satisfies* the condition $x^\circ \sqsubseteq \boxed{X}$ if $p(x^\circ) \subseteq p(\boxed{X})$. *$p$ satisfies* $\neg c$ iff p does not satisfy c, and *p satisfies* $\wedge_{i\in I} c_i$ iff p satisfies all c_i ($i \in I$). An X-graph G *satisfies* a condition c, if c is a condition over $\emptyset$ and the morphism $\emptyset \to G$ satisfies c. We write $p \models c$ to denote that p satisfies c and $G \models c$ to denote that G satisfies c. The notion of $\mathscr{A}'$-satisfiability is obtained from $\mathscr{B}'$-satisfiability by replacing all occurrences of $\mathscr{B}'$ by $\mathscr{A}'$. We write $p \models_{\mathscr{A}'} c$ to denote that p $\mathscr{A}'$-satisfies c and $G \models_{\mathscr{A}'} c$ to denote that G $\mathscr{A}'$-satisfies c.

The following monadic second-order graph properties can be expressed by HR conditions.

Example 4 (MSO graph properties) *For the HR system with the rules* $+ ::= \bullet_1 \to \bullet_2 \mid \bullet_1 \xrightarrow{+} \bullet_2$, *we have the following.*

Simple paths. *A* nonempty path *in G is here a sequence of nodes $(v_1, v_2, \ldots, v_n)$ with $n \geq 2$ such that there is an edge with source v_i and target v_{i+1} for all i and $v_i = v_j \Rightarrow \{i,j\} = \{1,n\}$; if $v_1 = v_n$, this path is a* cycle. *If in a path each node only appears once, then it is called a* simple path; *if each node appears once except that $v_1 = v_n$, then it is a* simple cycle. *The* HR *condition* $\mathtt{path}(1,2) = \exists(\ \bullet_1\ \bullet_2 \to \bullet_1 \xrightarrow{+} \bullet_2\ )$ *requires that there is a simple, nonempty path from the image of 1 to the image of 2.*

Connectedness. *The* HR *condition* $\mathtt{connected} = \forall(\bullet_1\ \bullet_2, \mathtt{path}(1,2))$ *requires that, for each pair of distinct nodes, there is a simple, nonempty path, i.e., the graph is strongly connected.*

Cycle-freeness. *The* HR *condition* $\mathtt{cyclefree} = \nexists(\bullet_1 \circlearrowleft +)$ *requires that the graph is cycle-free.*

Planarity. *By Kuratowski's Theorem (see e.g. [Eve79]) a graph is* planar *if and only if it has no subgraph homeomorphic to $K_{3,3}$ or K_5. For undirected graphs, two graphs are* homeomorphic *if both can be obtained from the same graph by insertion of new nodes of degree 2, in edges, i.e. an edge is replaced by a path whose intermediate nodes are all new. The* HR *condition* $\mathtt{planar} = \nexists(K_5^*) \wedge \nexists(K_{3,3}^*)$ *where K_5^* and $K_{3,3}^*$ are obtained from the graphs K_5 and $K_{3,3}$ by replacing all edges by hyperedges with label x, respectively, requires that the graph has no subgraph homeomorphic to K_5 or $K_{3,3}$, i.e. that the graph is planar.*

Coloring. *A* coloring *of a graph is an assignment of colors to its nodes so that no two adjacent nodes have the same color. A k-coloring of a graph G uses k colors. By König's characterization*

(see e.g. [Har69]), a graph is 2-colorable if and only if it has no odd cycles. For undirected graphs, i.e., graphs in which each undirected edge stands for two directed edges in opposite direction, the HR *condition* $2\texttt{color} = \nexists(\bullet_1 \circlearrowright^{+2})$ *with* $+2 ::= \bullet_1 \to \bullet_2 \mid \bullet_1 \to \bullet \to \bullet^{+2}_2$ *requires that there are no cycles of odd length, i.e., the graph is 2-colorable.*

Hamiltonicity. *A graph is* Hamiltonian, *if there exists a Hamiltonian circuit, i.e. a simple circuit on which every node of the graph appears exactly once (see e.g. [Eve79]). For undirected graphs, the* HR *condition* $\texttt{hamiltonian} = \exists(\bullet_1 \circlearrowright^{+}, \nexists(\bullet_1 \circlearrowright^{+} \bullet))$ *with* $+ ::= \bullet_1 \to \bullet_2 \mid \bullet_1 \to \bullet \overset{+}{\to} \bullet_2$ *, requires that there exists a simple circuit in the graph and there is no additional node in the graph, i.e. every node of the graph lies on the circuit, the graph is Hamiltonian.*

The following second-order graph properties can be expressed by HR conditions.

Example 5 (SO graph properties) *The following second-order properties (which are not MSO-expressible) can be expressed by* HR *graph conditions.*

Even number of nodes. *The* HR *condition* $\texttt{even} = \exists(\boxed{X}, \nexists(\boxed{X} \bullet))$ *with* $X ::= \emptyset \mid \bullet \bullet \boxed{X}$ *expresses the* SO *property "the graph has an even number of nodes".*

Equal number of *a***'s and** *b***'s.** *The* HR *condition* $\texttt{equal} = \exists(\boxed{X}, \nexists(\boxed{X}\,\textcircled{a}) \wedge \nexists(\boxed{X}\,\textcircled{b}))$ *with* $X ::= \emptyset \mid \textcircled{a}\,\textcircled{b}\,\boxed{X}$ *expresses the* SO *property "the graph has as many nodes labelled a as b".*

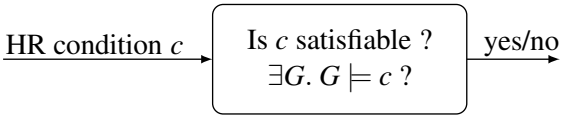
Paths of same length. *The* HR *condition* $2\texttt{paths}_{1,2} = \exists(\bullet_1 \quad \bullet_2 \to \bullet_1 \to \boxed{+}^{1,2}_{3,4} \to \bullet_1 \bullet_2)$ *with* $+ ::= \bullet_1 \to \bullet_2 \mid \bullet_1 \to \bullet \to \boxed{+}^{1,2}_{3,4} \to \bullet_2 \bullet_4$ *expresses the* SO *property "there exist two node-disjoint paths of same length from the image of 1 to the image of 2".*

Remark 5 (Counting MSO graph properties) *Counting* MSO *[Cou90b] extends* MSO *with first-order modulo-counting quantifiers such as "there exists an even number of elements such that ...". E.g., the SO graph property "even number of nodes" is counting* MSO.

4 Decidability of the validity problem

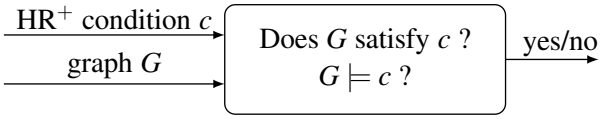
In this section, we show the undecidability of the satisfiability problem for HR conditions and the decidability of the validity problem for HR^+ conditions and graphs. By the undecidability of the satisfiability problem of first-order graph formulas [Tra50, Cou90a] and the equivalence of first-order graph formulas and finite graph conditions [HP09], we obtain the undecidability of the satisfiability problem of HR conditions.

Fact 3 (Undecidability of the satisfiability problem) *The satisfiability problem is undecidable, i.e., there is no algorithm for determining whether or not, given a finite* HR *condition c, there exists a replacement morphism p such that* $p \models c$ *[there exists a graph G such that* $G \models c$*].*



The validity problem for HR^+ conditions is decidable.

Theorem 2 (Decidability of the validity problem) *The validity problem is decidable, i.e., there is an algorithm for determining whether or not, given a finite* HR^+ *condition* c *and a replacement morphism* p, $p \models c$ *[given a finite* HR^+ *condition* c *and a graph* G, $G \models c$*].*



Proof. Let c be a finite HR^+ condition with variables and corresponding HR system $\mathscr{R}$. Without loss of generality, we may assume that $\mathscr{R}$ is monotone, i.e., $\mathrm{size}(x^\circ) \leq \mathrm{size}(R)$ for each $x/R \in \mathscr{R}$. Otherwise, we transform it into an equivalent monotone one [Hab92]. Let $p\colon P \twoheadrightarrow G$ be a replacement morphism and $\mathrm{size}(G) = |\mathrm{V}_G| + |\mathrm{E}_G| = n$ for some $n \geq 0$. Let $\mathscr{G}^n$ denote the set of all graphs of size $\leq n$. By induction, it can be shown that, for all $x \in \mathrm{X}$, $n \in \mathbb{N}_0$, and $C \in \mathscr{G}_\mathrm{X}$, the sets

$$\begin{array}{lcl} \mathscr{R}^n(x) & = & \{G \in \mathscr{G}^n \mid x^\circ \Rightarrow^*_\mathscr{R} G\} \\ \mathrm{Repl}^n(C) & = & \{\mathrm{repl} \in \mathrm{Repl} \mid \mathrm{repl}(y) \in \mathscr{R}^n(\mathrm{ly}(y)) \text{ for all } y \in \mathrm{Y}_C\} \\ \mathscr{R}^n(C) & = & \{\mathrm{repl} \in \mathrm{Repl}^n(C) \mid \mathrm{repl}(C) \leq n\} \end{array}$$

can be constructed effectively. For HR conditions of the form $c = \exists(a,d)$, we have $p \models c \Leftrightarrow p \models \bigvee_{\mathrm{repl}' \in \mathscr{R}^n} c$. Since $\mathscr{R}^n(C)$ is finite, the condition $\bigvee_{\mathrm{repl} \in \mathscr{R}^n(C)} \mathrm{repl}(d)$ is finite. In this way, for existential subconditions, satisfiability can be tested and, for non-existential conditions, satisfiability can be checked. For a HR condition c over $\emptyset$ and a graph G, $G \models c \Leftrightarrow \emptyset \to G \models c$. □

Remark 6 *Theorem 2 can be extended to monotonic replacement systems, i.e., replacement systems consisting of a set of rules of the form* $\langle L \hookleftarrow K \hookrightarrow R \rangle$ *in the double-pushout approach [Ehr79] with* $\mathrm{size}(L) \leq \mathrm{size}(R)$.

5 Expressiveness of HR^+ conditions

For investigating the expressiveness of HR^+ conditions, we compare HR^+ conditions and monadic second-order (MSO) formulas on graphs [Cou90a, Cou97]. We show that there is a transformation from MSO formulas into "equivalent" HR^+ conditions. Vice versa, there is no such a transformation: HR^+ conditions can express counting MSO graph properties [Cou90b].

Let Rel be a finite set of relation symbols. Let Var contain individual variables and relation variables of arity one. Since a relation with one argument is nothing but a set, we call these variables *set variables*. A monadic second-order formula over Rel is a second-order formula written with Rel and Var: the quantified and free variables are individual or set variables; there is no restriction on the arity of symbols in Rel. In order to get more readable formulas, we shall write $x \doteq y$ instead of $\doteq(x,y)$ and $x \in X$ instead of $X(x)$ where X is a set variable. For simplicity, we consider graphs G with common label alphabet C and common labeling function l_G for nodes and edges.

Definition 10 (MSO graph formulas) Let Var be a countable set of individual and set variables. The set of all *monadic second-order (*MSO*) formulas* (over Var and C) is inductively defined as

follow: For $b \in \mathrm{C}$ and $x,y,z,X \in \mathrm{Var}$, $\mathrm{lab}_b(x)$, $\mathrm{inc}(x,y,z)$, $x \doteq y$, and $x \in X$ are formulas. For formulas F, F_i ($i \in I$) and variables $x,X \in \mathrm{Var}$, $\neg F$, $\wedge_{i\in I} F_i$, $\exists xF$, and $\exists XF$ are formulas. For a formula F, $\mathrm{Free}(F)$ denotes the set of all *free* variables of F.[4] A MSO formula is *closed*, if $\mathrm{Free}(F)$ is empty.

Notation For a (finite) index set I, the expression $\vee_{i\in I} F_i$ abbreviates the formula $\neg \wedge_{i\in I} \neg F_i$, $F \Rightarrow G$ abbreviates $\neg F \vee G$, $\forall xF$ abbreviates $\neg\exists x\neg F$, and $\forall XF$ abbreviates $\neg\exists X\neg F$. Moreover, the expression $\mathrm{edg}(y,z)$ abbreviates the formula $\exists x.\mathrm{inc}(x,y,z)$.

Definition 11 (Satisfiability) The *semantic* $G[\![F]\!](\sigma)$ of a formula F in a non-empty graph G with $\mathrm{D}_G = \mathrm{V}_G + \mathrm{E}_G + \mathscr{P}(\mathrm{V}_G) + \mathscr{P}(\mathrm{E}_G)$[5] under assignment $\sigma\colon \mathrm{Var} \to \mathrm{D}_G$ is inductively defined as follows: $G[\![\mathrm{lab}_b(x)]\!](\sigma) = \mathit{true}$ iff $\sigma(x) = \mathrm{l}_G(b)$, $G[\![\mathrm{inc}(x,y,z)]\!](\sigma) = \mathit{true}$ iff $\sigma(x) \in \mathrm{E}_G$, $\mathrm{s}_G(\sigma(x)) = \sigma(y)$, and $\mathrm{t}_G(\sigma(x)) = \sigma(z)$, $G[\![x \doteq y]\!](\sigma) = \mathit{true}$ iff $\sigma(x) = \sigma(y)$, $G[\![x \in X]\!](\sigma) = \mathit{true}$ iff $\sigma(x) \in \sigma(X)$, $G[\![\exists x\ F]\!](\sigma) = \mathit{true}$ iff $G[\![F]\!](\sigma\{x/d\}) = \mathit{true}$ for some $d \in \mathrm{D}_G$ where, for some $x \in \mathrm{Var}$ and some $d \in \mathrm{D}_G$, $\sigma\{x/d\}$ is the modified assignment with $\sigma\{x/d\}(x) = d$ and $\sigma\{x/d\}(y) = \sigma(y)$ otherwise. The semantics is extended to the operators in the usual way: $G[\![\mathrm{true}]\!](\sigma) = \mathit{true}$, $G[\![\neg F]\!](\sigma) = \neg G[\![F]\!](\sigma)$, and $G[\![\wedge_{i\in I} F_i]\!](\sigma) = \wedge_{i\in I} G[\![F_i]\!](\sigma)$. A graph G *satisfies* a formula F, denoted by $G \models F$, iff, for all assignments $\sigma\colon \mathrm{Var} \to D_G$, $G[\![F]\!](\sigma) = \mathit{true}$.

Example 6 *The* MSO *formula* $F_0(x_1,x_2) = \forall X((\forall y\forall z(y \in X \wedge \mathrm{edg}(y,z) \Rightarrow z \in X) \wedge \forall y(\mathrm{edg}(x_1,y) \Rightarrow y \in X)) \Rightarrow x_2 \in X)$ *with free variables* x_1 *and* x_2 *expresses the property "There is a nonempty path from* x_1 *to* x_2*" [Cou97]. The formula* $F_1(x_1,x_2) = x_1 \doteq x_2 \vee F_0(x_1,x_2)$ *expresses the property "*$x_1 = x_2$ *or there is a nonempty path from* x_1 *to* x_2*" and the formula* $F_2 = \forall x_1\ \forall y_2\ F_1(x_1,x_2)$ *expresses the property "the graph is strongly connected".*

MSO formulas and HR^+ conditions are closely related. More precisely, MSO formulas can be transformed into $\mathscr{A}'$-satisfiable HR^+ conditions.

Theorem 3 (From MSO formulas to $\mathscr{A}'$-satisfiable HR^+ conditions) *There is a transformation* Cond *from* MSO *formulas to* HR^+ *conditions such that for all* MSO *formulas* F *and all graphs* G*,*

$$G \models F \iff G \models_{\mathscr{A}'} \mathrm{Cond}(F).$$

Proof. Let F be a MSO formula. Without loss of generality, we may assume that F is closed and *rectified*, i.e. distinct quantifiers bind occurrences of distinct variables; otherwise, we build the universal closure of F and rename the variables. Since F is rectified, the variables of F can be represented by isolated nodes, edges, and hyperedges in the graphs of a constructed condition. Let P be an X-graph, Iso_P the set of all isolated nodes in P, $\mathrm{D}'_P = \mathrm{Iso}_P + \mathrm{E}_P + \mathrm{Y}_P$. If $\mathrm{D}'_P \subseteq \mathrm{Var}$, then every replacement morphism $p\colon P \twoheadrightarrow G$ into a non-empty graph G induces an assignment $\sigma\colon \mathrm{Var} \to \mathrm{D}_G$ such that $p = \sigma[\mathrm{D}'_P]$, i.e., $p(x) = \sigma(x)$ for each $x \in \mathrm{D}'_P$. Vice versa, every assignment $\sigma\colon \mathrm{Var} \to \mathrm{D}_G$ induces a replacement morphism $p\colon P \twoheadrightarrow G$ with $p = \sigma[\mathrm{D}'_P]$.

[4] For a MSO formulas, the set of free variables is defined inductively as for first-order formulas.
[5] For a set A, $\mathscr{P}(A)$ denotes the power set of A.

$$\begin{array}{ccccccc} \mathrm{Free}(F) & \subseteq & \mathrm{D}'_P & \subseteq & \mathrm{D}_P & \cdots\cdots & P \\ & & \cap\downarrow & & \downarrow & = & \downarrow p \\ & & \mathrm{Var} & \xrightarrow{\sigma} & \mathrm{D}_G & \cdots\cdots & G \end{array}$$

Construction For a MSO formula F, the HR$^+$ condition is given by $\mathrm{Cond}(F) = \mathrm{Cond}(\emptyset, F)$. For an X-graph P and a MSO formula F with $\mathrm{Free}(F) \subseteq \mathrm{D}'_P \subseteq \mathrm{Var}$, the HR$^+$ condition $\mathrm{Cond}(P,F)$ is constructed as follows:

$\mathrm{Cond}(P, \mathrm{lab}_b(x))$ $=$ true if $\mathrm{l}_P(x) = b$; false otherwise.

$\mathrm{Cond}(P, \mathrm{inc}(x,y,z))$ $= \exists(P \to P_{\mathrm{s}_P(x)=y,\mathrm{t}_P(x)=z})$ if $x \in \mathrm{E}_P$, $y,z \in \mathrm{V}_P$ where $P_{x_1=y_1,x_2=y_2}$ is obtained from P by identifying the pairs $(x_1,y_1),(x_2,y_2)$; false otherwise.

$\mathrm{Cond}(P, x \doteq y)$ $= \exists(P \to P_{x=y})$ if x and y are identifiable in P, i.e., $x,y \in \mathrm{V}_P$ and $\mathrm{l}_P(x) = \mathrm{l}_P(y)$ or $x,y \in \mathrm{E}_P$, $\mathrm{s}_P(x) = \mathrm{s}_P(y)$, $\mathrm{t}_P(x) = \mathrm{t}_P(y)$, and $\mathrm{l}_P(x) = \mathrm{l}_P(y)$ or $x,y \in \mathscr{P}(\mathrm{V}_P) + \mathscr{P}(\mathrm{E}_P)$ and there exist morphisms $a\colon x^\circ \to P$ and $b\colon y^\circ \to P$ from the graphs induced by x and y, respectively, such that $a(x) = a(y)$.

$\mathrm{Cond}(P, x \in X)$ $= \bigvee_{i=1}^2 \langle x^\circ \sqsubseteq \boxed{X}, \mathscr{R}_i \rangle$; false otherwise.

$\mathrm{Cond}(P, \exists x F)$ $= \bigvee_{b \in \mathrm{C}} \exists(P \to P_b, \mathrm{Cond}(P_b, F)) \vee \bigvee_{b,d,d' \in \mathrm{C}} \exists(P \to P_{bdd'}, \mathrm{Cond}(P_{bdd'}, F)) \vee \bigvee_{i=1}^2 \langle \exists(P \to P + \boxed{X}, \mathrm{Cond}(P + \boxed{X}, F)), \mathscr{R}_i \rangle$ where P_b is obtained from P by adding a node x with label b, $P_{bdd'}$ by adding an edge x with label b together with a d-labeled source and a d'-labeled target, $\mathscr{R}_1 : X ::= \emptyset \mid \boxed{x}\ \textcircled{b}$ for $b \in \mathrm{C}$, and $\mathscr{R}_2 : x ::= \emptyset \mid \boxed{x}\ \textcircled{d} \xrightarrow{b} \textcircled{d'}$ for $b,d,d' \in \mathrm{C}$.

For Boolean formulas over formulas, the transformation is extended as usual: $\mathrm{Cond}(P, \mathrm{true}) = \mathrm{true}$, $\mathrm{Cond}(P, \neg F) = \neg \mathrm{Cond}(P,F)$, and $\mathrm{Cond}(P, \wedge_{i \in I} F_i) = \wedge_{i \in I} \mathrm{Cond}(X, F_i)$. For Boolean formulas over HR$^+$ conditions with variables $\langle c, \mathscr{R} \rangle$ and $\langle c_i, \mathscr{R}_i \rangle$ $(i \in I)$ with pairwise distinct sets free variables[6], $\mathrm{Cond}(P, \mathrm{true}) = \langle \mathrm{true}, \emptyset \rangle$, $\neg \langle c, \mathscr{R} \rangle = \langle \neg c, \mathscr{R} \rangle$ and $\wedge_{i \in I} \langle c_i, \mathscr{R}_i \rangle = \langle \wedge_{i \in I} c_i, \cup_{i \in I} \mathscr{R}_i \rangle$.

There is a nice relationship between the satisfiability of MSO formulas and the $\mathscr{A}'$-satisfiability of the corresponding HR$^+$ conditions.

Claim 1 *For all rectified* MSO *formulas F, all graphs G, all assignments $\sigma\colon \mathrm{Var} \to \mathrm{D}_G$, and all replacement morphisms $p\colon P \twoheadrightarrow G$ with $\sigma = p[\mathrm{D}'_P]$, $G[\![F]\!](\sigma) = \mathit{true} \iff p \models_{\mathscr{A}'} \mathrm{Cond}(P,F)$.*

Proof (of the claim) The proof makes use of the proof of the corresponding statement for rectified first-order formulas given in [HP09] and is done by structural induction.
Basis. For the atomic formulas $\mathrm{lab}_b(x)$, $\mathrm{inc}(x,y,z)$, and $x \doteq y$, the proof is as in [HP09]. For atomic formulas of the form $x \in X$, the statement follows by the semantics of $x \in X$, $p = \sigma[\mathrm{D}'_P]$, $x, X \in \mathrm{Free}(x \in X) \subseteq D'_P$, the semantics of $x^\circ \sqsubseteq \boxed{X}$, and the definition of Cond:

[6] Without loss of generality, we may assume that, for a collection of HR$^+$ conditions $\langle c_i, \mathscr{R}_i \rangle$ $(i \in I)$, the sets of free variables are pairwise disjoint. Otherwise, we rename the free variables in the conditions.

$$\begin{array}{ll} & G[\![x \in X]\!](\sigma) = \mathit{true} \\ \Leftrightarrow & \sigma(x) \in \sigma(X) \\ \Leftrightarrow & p(x^\circ) \subseteq p(\boxed{X}) \\ \Leftrightarrow & p \models_{\mathscr{A}'} \bigvee_{i=1}^{2} \langle x^\circ \sqsubseteq \boxed{X}, \mathscr{R}_i \rangle \\ \Leftrightarrow & p \models_{\mathscr{A}'} \mathrm{Cond}(P, x \in X) \end{array}$$

Hypothesis. Assume that the statement holds for rectified MSO formulas F.
Step. For rectified MSO formulas of the form $\neg F$ and $\wedge_{i\in I} F_i$, the proof is as in [HP09]. For formulas of the form $\exists x F$, graphs G, and assignments σ, the statement follows from the semantics of $\exists x F$, the induction hypothesis, $q = \sigma\{x/d\}[\mathrm{D}'_P]$, and the definitions of $\models_{\mathscr{A}'}$ and Cond:

$$\begin{array}{ll} & G[\![\exists x F]\!](\sigma) = \mathit{true} \\ \Leftrightarrow & \exists d \in \mathrm{D}_G . G[\![F]\!](\sigma\{x/d\}) = \mathit{true} \\ \Leftrightarrow & \exists b \in \mathrm{C} . \exists q \colon P_b \to G \in \mathscr{A}' . p = q \circ (P \to P_b) \wedge q \models_{\mathscr{A}'} \mathrm{Cond}(P_b, F) \\ & \vee \exists b, d, d' \in \mathrm{C} . \exists q \colon P_{bdd'} \to G \in \mathscr{A}' . p = q \circ (P \to P_{bdd'}) \wedge q \models_{\mathscr{A}'} \mathrm{Cond}(P_{bdd'}, F) \\ & \vee \exists q \colon P + \boxed{x} \to G \in \mathscr{A}' . p = q \circ (P \to P + \boxed{x}) \wedge q \models_{\mathscr{A}'} \mathrm{Cond}(P + \boxed{x}, F) \\ \Leftrightarrow & p \models_{\mathscr{A}'} \bigvee_{b\in \mathrm{C}} \exists (P \to P_b, \mathrm{Cond}(P_b, F)) \vee \bigvee_{b,d,d' \in \mathrm{C}} \exists (P \to P_{bdd'}, \mathrm{Cond}(P_{bdd'}, F)) \\ & \vee \bigvee_{i=1}^{2} \langle \exists (P \to P + \boxed{X}, \mathrm{Cond}(P + \boxed{X}, F)), \mathscr{R}_i \rangle \\ \Leftrightarrow & p \models_{\mathscr{A}'} \mathrm{Cond}(P, \exists x F) \end{array}$$

This completes the inductive proof of the claim.

By the claim above and the definitions $\models$, $\models_{\mathscr{A}'}$, and Cond, for all graphs G and all closed, rectified MSO formulas F, we have $G \models F \Leftrightarrow \forall \sigma \colon \mathrm{Var} \to \mathrm{D}_G . G[\![F]\!](\sigma) = \mathit{true} \Leftrightarrow \emptyset \to G \models_{\mathscr{A}'} \mathrm{Cond}(\emptyset, F) \Leftrightarrow G \models_{\mathscr{A}'} \mathrm{Cond}(F)$, i.e., the HR^+ condition $\mathrm{Cond}(F)$ has the wanted property. □

Example 7 *The closure of the* MSO *formula*

$$F(x_1, x_2) = \forall X (\underbrace{(\forall y \forall z (y \in X \wedge \mathrm{edg}(y,z) \Rightarrow z \in X)}_{G_1} \wedge \underbrace{\forall y' (\mathrm{edg}(x_1, y') \Rightarrow y' \in X)}_{G_2}) \Rightarrow \underbrace{x_2 \in X}_{G_3})$$

is transformed into the HR^+ *condition*

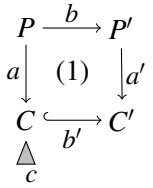
$$\begin{array}{ll} & \mathrm{Cond}(\forall x_1 \forall x_2 F(x_1, x_2)) \\ = & \mathrm{Cond}(\emptyset, \forall x_1 \forall x_2 F(x_1, x_2)) \\ \equiv & \forall (\bullet_{x_1} \bullet_{x_2}, \mathrm{Cond}(\emptyset, \forall X (G_1 \wedge G_2 \Rightarrow G_3)))) \\ \equiv & \forall (\bullet_{x_1} \bullet_{x_2} \boxed{X}, (\mathrm{Cond}(\boxed{X}, G_1 \wedge G_2 \Rightarrow G_3)) \\ = & \forall (\bullet_{x_1} \bullet_{x_2} \boxed{X}, (\mathrm{Cond}(\boxed{X}, G_1) \wedge \mathrm{Cond}(\boxed{X}, G_2) \Rightarrow \mathrm{Cond}(\boxed{X}, G_3))) \\ \equiv & \forall (\bullet_{x_1} \bullet_{x_2} \boxed{X}, (\forall (\bullet_{x_1} \bullet_{x_2} \boxed{X} \bullet_{y} \bullet_{z}, (\bullet_{y} \sqsubseteq \boxed{X} \wedge \exists (\bullet_{x_1} \bullet_{x_2} \boxed{X} \bullet_{y} \to \bullet_{z}) \Rightarrow \bullet_{z} \sqsubseteq \boxed{X}) \wedge \\ & \forall (\bullet_{x_1} \bullet_{x_2} \boxed{X} \bullet_{y'}, \exists (\bullet_{x_1} \to \bullet_{y'} \bullet_{x_2} \boxed{X} \bullet_{y'}) \Rightarrow \bullet_{y'} \sqsubseteq \boxed{X}) \Rightarrow \bullet_{x_2} \sqsubseteq \boxed{X})) \end{array}$$

with $X ::= \emptyset \mid \boxed{X} \bullet$ *using the equivalence* $\forall (x(\forall (y,c)) \equiv \forall (y \circ x, c))$ *in [HP05].*

$\mathscr{A}'$-satisfiable HR^+ conditions can be transformed into $\mathscr{B}'$-satisfiable HR^+ conditions.

Lemma 1 (From $\mathscr{A}'$- to $\mathscr{B}'$-satisfiability) *There is a transformation* Bsat *such that, for every* HR^+ *condition* c *over* $\emptyset$ *and every graph* G, $G \models_{\mathscr{A}'} c \iff G \models \mathrm{Bsat}(c)$.

Construction For a HR condition c over $\emptyset$, $\mathrm{Bsat}(c) = \mathrm{Shift}(\emptyset \to \emptyset, c)$, and, for a HR$^+$ condition of the form $x^\circ \sqsubseteq \boxed{X}$, $\mathrm{Bsat}(x^\circ \sqsubseteq \boxed{X}) = x^\circ \sqsubseteq \boxed{X}$.



For a morphism $b\colon P \to P'$, $\mathrm{Shift}(b, \_)$ is inductively defined as follows:
$\mathrm{Shift}(b, \mathrm{true}) = \mathrm{true}$.
$\mathrm{Shift}(b, \exists(a,c)) = \bigvee_{(a',b') \in \mathscr{F}} \exists(a', \mathrm{Shift}(b', c))$ if $\mathscr{F} = \{(a',b') \mid (a',b')$ jointly epimorphic, (1) commutes$\} \neq \emptyset$ and false, otherwise.
$\mathrm{Shift}(b, x^\circ \sqsubseteq \boxed{X}) = x^\circ \sqsubseteq \boxed{X}$.

For Boolean formulas over HR$^+$ conditions, $\mathrm{Bsat}(\_)$ and $\mathrm{Shift}(b, \_)$ are extended as usual.

Proof. For HR conditions c over $\emptyset$, the statement follows directly from the corresponding statement in [HP09]. For HR$^+$ conditions of the form $x^\circ \sqsubseteq \boxed{X}$, we have $p \models_{\mathscr{A}'} x^\circ \sqsubseteq \boxed{X} \Leftrightarrow p \models x^\circ \sqsubseteq \boxed{X}$. Consequently, for all graphs G, we have $G \models_{\mathscr{A}'} x^\circ \sqsubseteq \boxed{X}) \Leftrightarrow i_G \models_{\mathscr{A}'} x^\circ \sqsubseteq \boxed{X} \Leftrightarrow i_G \models x^\circ \sqsubseteq \boxed{X} \Leftrightarrow G \models x^\circ \sqsubseteq \boxed{X}$ where i_G denotes the injective morphism $\emptyset \to G$. This completes the proof. □

Example 8 *The* HR *condition* $c = \exists(\emptyset \to \bigcirc_1 \overset{+}{\to} \bigcirc_2)$ *with* HR *system* $\mathscr{R}$: $+ ::= \bullet_1 \to \bullet_2 \mid \bullet_1 \overset{+}{\to} \bullet_2$ *meaning "There exists some simple, nonempty path" is transformed into the* HR *condition* $\mathrm{Bsat}(c) = \exists(\emptyset \to \bigcirc_1 \overset{+}{\to} \bigcirc_2) \vee \exists(\emptyset \to \bigcirc_{1=2}^{+})$ *with the* HR *system* $\mathscr{R}$ *meaning "There exists some simple, nonempty path or cycle". The* HR *condition* $c = \exists(\emptyset \to \boxed{X}\ \bullet_x\ \bullet_y)$ *is transformed into the* HR *condition* $\mathrm{Bsat}(c) = \exists(\emptyset \to \boxed{X}\ \bullet_x\ \bullet_y) \vee \exists(\emptyset \to \boxed{X}\ \bullet_{x=y})$.

The standard semantics for HR$^+$ conditions is $\mathscr{B}'$-satisfiability. By Theorem 3 and Lemma 1, MSO formulas also can be transformed into $\mathscr{B}'$-satisfiable HR$^+$ conditions.

Corollary 1 (From MSO formulas to HR$^+$ conditions) *There is a transformation* $\mathrm{Cond}_{\mathscr{B}'}$ *from* MSO *formulas to* HR$^+$ *conditions, such that, for all* MSO *formulas* F *and all graphs* G,

$$G \models F \iff G \models \mathrm{Cond}_{\mathscr{B}'}(F).$$

Proof. Let $\mathrm{Cond}_{\mathscr{B}'} = \mathrm{Bsat} \circ \mathrm{Cond}$. By Theorem 3 and Lemma 1, we have $G \models F \iff G \models_{\mathscr{A}'} \mathrm{Cond}(F) \iff G \models \mathrm{Cond}_{\mathscr{B}'}(F)$. □

Remark 7 *For finite* MSO *formulas* c, *the corresponding* HR$^+$ *conditions* $\mathrm{Cond}_{\mathscr{B}'}(c)$ *are finite.*

6 Related concepts

In [Hab92, DHK97], hypergraphs and hyperedge replacement are introduced. In [PH96], the concept of graphs with variables is introduced using hyperedges as graph variables and hyperedge replacement as substitution mechanism. In [Hof01, HJG08], graph transformation — using graph variables in rules without/with application conditions — are considered and, in [HM10], an extension of hyperedge replacement grammars by context nodes and HR application conditions is introduced. In [KMP05], graphs with $*$-labelled edges are considered; the $*$-labelled edge stands for paths of arbitrary length. In [KK08], the authors describe regular expressions

for forbidden paths which should not occur in any reachable graph. In [BKK03], the authors introduce a monadic second-order logic over graphs expressively enough to characterize typical graph properties. They propose to describe state predicates, i.e. the graph properties of interest, by means of a monadic second-order logic on graphs, where quantification is allowed over (sets of) edges. In [CE95], a logical characterization of the sets of hypergraphs defined by HR grammars is given saying that a set of hypergraphs L can be generated by an HR grammar if and only if the set of structures $|L|_2$ is the image of a recognizable set of finite trees under a MSO-definable transduction. The comparison diagram in [Cou90a] shows that the families of HR- and MSO-definable sets of graphs are not comparable: In MSO, one cannot express whether the number of elements of a set is even; in HR, this is expressible. In HR, one cannot express that a graph is a square grid; in MSO, this is expressible. In [Pra04], it is sketched that graphs with variables and substitution morphisms (consisting of a graph morphism and a substitution) form an adhesive HLR category.

7 Conclusion

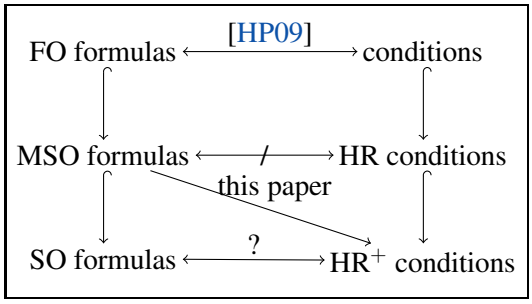
In this paper, we have investigated graph conditions with variables where the variables may be replaced by graphs generated by some assigned hyperedge replacement system. We have shown the following.

Graphs with variables. Graphs with variables and replacement morphisms form a category. Distinguishing the class of all injective graph morphisms, we obtain a weak adhesive HLR category $\langle \mathrm{XGraphs}, \mathscr{M} \rangle$. All results — known for weak adhesive HLR categories — hold for rules with variables: Local Church-Rosser, Parallelism, Concurrency, Amalgamation, Embedding/Extension, Restriction, and the Local Confluence Theorem.

HR **and** HR^+ **conditions.** HR conditions are conditions in $\langle \mathrm{XGraphs}, \mathscr{M} \rangle$. HR^+ conditions extend HR conditions by subconditions of the form $x^\circ \sqsubseteq \boxed{X}$, a counterpart to MSO subformulas of the form $x \in X$.

Validity for HR^+ **conditions.** By the monotony property of HR systems the validity problem for HR^+ conditions is decidable, i.e., there is an algorithm for determining whether or not, given a finite HR^+ condition and a graph, the graph satisfies the condition.

Expressiveness of HR^+ **conditions.** There is a transformation from MSO formulas to HR^+ conditions, but HR^+ conditions are more powerful: HR^+ can express certain counting MSO formulas. It remains the question whether or not there is a transformation from HR^+ conditions to SO formulas, and vice versa.



Expressiveness of OCL constraints. The concept of OCL constraints is well-known and there are translations from restricted OCL constraints into first-order graph formulas [BKS02] and graph constraints [EKT09], respectively. What about a translation of a larger class of OCL constraints into HR constraints?

Acknowledgements: We thank the reviewers for their constructive criticism that helped to improve our paper.

Bibliography

[Bau06] J. Bauer. *Analysis of Communication Topologies by Partner Abstraction*. PhD thesis, Universität Saarbrücken, 2006.

[BKK03] P. Baldan, B. König, B. König. A Logic for Analyzing Abstractions of graph transformation systems. In *Int. Static Analysis Symposium (SAS '03)*. LNCS 2694, pp. 255–272. 2003.

[BKS02] B. Beckert, U. Keller, P. H. Schmitt. Translating the Object Constraint Language into First-order Predicate Logic. In *Proc. VERIFY, Workshop at Federated Logic Conferences (FLoC)*. 2002.

[CE95] B. Courcelle, J. Engelfriet. A Logical Characterization of the Sets of Hypergraphs Defined by Hyperedge Replacement Grammars. *Mathematical Systems Theory* 28:515–552, 1995.

[Cou90a] B. Courcelle. Graph Rewriting: An Algebraic and Logical Approach. In *Handbook of Theoretical Computer Science*. Volume B, pp. 193–242. Elsevier, 1990.

[Cou90b] B. Courcelle. The Monadic Second-Order Logic of Graphs I: Recognizable Sets of Finite Graphs. *Information and Computation* 85:12–75, 1990.

[Cou97] B. Courcelle. The Expression of Graph Properties and Graph Transformations in Monadic Second- Order Logic. In *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 1, pp. 313–400. World Scientific, 1997.

[DHK97] F. Drewes, A. Habel, H.-J. Kreowski. Hyperedge Replacement Graph Grammars. In *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 1, pp. 95–162. World Scientific, 1997.

[EEHP06] H. Ehrig, K. Ehrig, A. Habel, K.-H. Pennemann. Theory of Constraints and Application Conditions: From Graphs to High-Level Structures. *Fundamenta Informaticae* 74(1):135–166, 2006.

[EEKR99] H. Ehrig, G. Engels, H.-J. Kreowski, G. Rozenberg (eds.). *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 2: Applications, Languages and Tools. World Scientific, 1999.

[EH86] H. Ehrig, A. Habel. Graph Grammars with Application Conditions. In Rozenberg and Salomaa (eds.), *The Book of L*. Pp. 87–100. Springer, 1986.

[Ehr79] H. Ehrig. Introduction to the Algebraic Theory of Graph Grammars. In *Graph-Grammars and Their Application to Computer Science and Biology*. LNCS 73, pp. 1–69. 1979.

[EKMR99] H. Ehrig, H.-J. Kreowski, U. Montanari, G. Rozenberg (eds.). *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 3: Concurrency, Parallelism, and Distribution. World Scientific, 1999.

[EKT09] K. Ehrig, J. M. Küster, G. Taentzer. Generating instance models from meta models. *Software and System Modeling* 8(4):479–500, 2009.

[Eve79] S. Even. *Graph Algorithms*. Computer Science Press, Rockville, Maryland, 1979.

[Hab92] A. Habel. *Hyperedge Replacement: Grammars and Languages*. LNCS 643. Springer, 1992.

[Har69] F. Harary. *Graph Theory*. Addison-Wesley, Reading, Mass., 1969.

[HESV91] A. Hsu, F. Eskafi, S. Sachs, P. Varaiya. The Design of Platoon Maneuver Protocols for IVHS. Technical report, Institute of Transportation Studies, University of California at Berkeley, 1991.

[HHT96] A. Habel, R. Heckel, G. Taentzer. Graph Grammars with Negative Application Conditions. *Fundamenta Informaticae* 26:287–313, 1996.

[HJG08] B. Hoffmann, E. Jakumeit, R. Geiß. Graph Rewrite Rules with Structural Recursion. In Mosbah and Habel (eds.), *Int. Workshop on Graph Computational Models (GCM 2008)*. Pp. 5–16. 2008.

[HM10] B. Hoffmann, M. Minas. Defining Models - Meta Models versus Graph Grammars. In *Graph Transformation and Visual Modeling Techniques (GT-VMT 2010)*. Electronic Communications of the EASST. 2010.

[Hof01] B. Hoffmann. Shapely Hierarchical Graph Transformation. In *Proc. IEEE Symposia on Human-Centric Computing Languages and Environments*. IEEE Computer Press, pp. 30–37. 2001.

[HP05] A. Habel, K.-H. Pennemann. Nested Constraints and Application Conditions for High-Level Structures. In *Formal Methods in Software and System Modeling*. LNCS 3393, pp. 293–308. 2005.

[HP09] A. Habel, K.-H. Pennemann. Correctness of High-Level Transformation Systems Relative to Nested Conditions. *Mathematical Structures in Computer Science* 19:245–296, 2009.

[HW95] R. Heckel, A. Wagner. Ensuring Consistency of Conditional Graph Grammars — A Constructive Approach. In *SEGRAGRA '95*. ENTCS 2, pp. 95–104. 1995.

[KK08] B. König, V. Kozioura. Augur 2—A New Version of a Tool for the Analysis of Graph Transformation Systems. In *Proc. Workshop on Graph Transformation and Visual Modeling Techniques (GT- VMT '06)*. ENTCS 211, pp. 201–210. 2008.

[KMP05] M. Koch, L. V. Mancini, F. Parisi-Presicce. Graph-based Specification of Access Control Policies. *Journal of Computer and System Sciences* 71:1–33, 2005.

[Pen09] K.-H. Pennemann. *Development of Correct Graph Transformation Systems*. PhD thesis, Universität Oldenburg, 2009.

[PH96] D. Plump, A. Habel. Graph Unification and Matching. In *Graph Grammars and Their Application to Computer Science*. LNCS 1073, pp. 75–89. 1996.

[Pra04] U. Prange. Graphs with Variables as an Adhesive HLR Category. 2004. Private Communication.

[Roz97] G. Rozenberg (ed.). *Handbook of Graph Grammars and Computing by Graph Transformation*. Volume 1: Foundations. World Scientific, 1997.

[Tra50] B. A. Trakhtenbrot. The impossibility of an algorithm for the decision problem on finite classes (In Russian). *Doklady Akademii Nauk SSSR* 70:569–572, 1950. English translation in: Nine Papers on Logic and Quantum Electrodynamics, *AMS Transl. Ser. 2*, 23:1–5, 1963.