Electronic Communications of the EASST
Volume 61 (2013)

Selected Revised Papers from the 4th International Workshop on Graph Computation Models (GCM 2012)

HR* Graph Conditions Between Counting Monadic Second-Order and Second-Order Graph Formulas

Hendrik Radke

21 pages

Guest Editors: Rachid Echahed, Annegret Habel, Mohamed Mosbah
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

HR* Graph Conditions Between Counting Monadic Second-Order and Second-Order Graph Formulas

Hendrik Radke *

Universität Oldenburg
hendrik.radke@uni-oldenburg.de

Abstract: Graph conditions are a means to express structural properties for graph transformation systems and graph programs in a large variety of application areas. With HR* graph conditions, non-local graph properties like "there exists a path of arbitrary length" or "the graph is circle-free" can be expressed. We show, by induction over the structure of formulas and conditions, that (1) any node-counting monadic second-order formula can be expressed by an HR* condition and (2) any HR* condition can be expressed by a second-order graph formula.

Keywords: graph transformation, graph conditions, hyperedge replacement, monadic second-order logic, second-order logic

1 Introduction

Formal methods play an important role for the development of trustworthy systems. Visual modeling techniques help to understand complex systems. It is therefore desirable to combine visual modeling with formal verification. The approach taken here is to use graphs and graph transformation rules [EEPT06] to model states and state changes, respectively. Structural properties of the system are described by graph conditions.

In [HP09, Pen09], nested graph conditions have been discussed as a formalism to describe structural properties in a visual and intuitive way. Nested graph conditions are expressively equivalent to first-order graph formulas and can express local properties in the sense of Gaifman [Gai82]. However, there are many interesting non-local graph properties like the existence of an arbitrary-length path between two nodes, connectedness or circle-freeness of the graph. Several logics and languages have been developed to express such non-local properties. In [BCKL06], a modal logic is described which uses monadic second-order formulas to describe state properties and temporal modalities to describe behavioural properties. A linear temporal logic is used in [Ren08], including the monadic second-order quantification over sets of nodes. In [BK10, BBFK13], a logic is presented that can quantify over subobjects of a categorical object. For the category of graphs, this logic is as expressive as monadic second-order logic on graphs. The idea of enhancing nested conditions with variables which are later substituted is also used for the E-conditions in [PP12]. However, the variables in [PP12] are placeholders for string or integer attributes, while in this work, variables are placeholders for graphs.

* This work is supported by the German Research Foundation (DFG), Grant HA 2936/4-1 (Meta Modeling and Graph Grammars: Integration of two Paragdigms for the Definition of visual Modeling Languages)

In [HR10], we introduced graph conditions with variables and showed that they are more expressive than monadic second-order graph formulas. However, an upper bound on the expressiveness of these conditions remained an open question. This paper gives both a tighter lower and an upper bound for so-called HR* conditions. We will show that HR* conditions are at least as expressive as node-counting monadic second order formulas and at most as strong as formulas in second-order logic on graphs. Figure 1 gives an overview on different languages that can be used to describe graph properties and compares their respective expressive power.

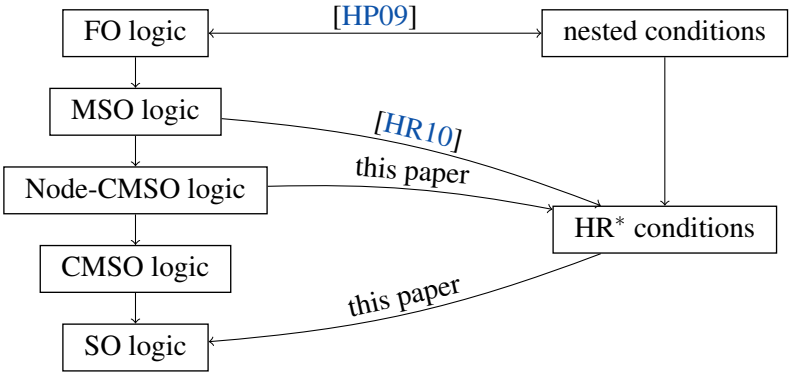


Figure 1: Expressiveness of languages for graph properties: State of the art and contributions.

The paper is organized as follows: In Section 2, we will give the necessary definitions for HR* graph conditions, along with some examples. In Section 4, we briefly recall second-order graph formulas. In Section 5, we show that HR* conditions can express every node-counting monadic second-order formula. In Section 6, the construction of a second-order graph formula from an HR* graph condition is given, step-by-step, along with some examples. We discuss the results in the concluding Section 7.

2 HR* conditions

HR* conditions combine the first-order logical framework and graph morphisms from nested conditions [HP09] with hyperedge replacement to represent context-free structures of arbitrary size.

Hyperedges relate an arbitrary, fixed number of nodes and are used in HR* conditions as variables, which are later substituted by graphs. We extend the concept of directed, labeled graphs with hyperedge variables, which can be seen as placeholders for graphs to be added later.

Definition 1 (graph with variables) Let C be a fixed, finite alphabet of set and edge labels and $\mathcal{X}$ a set of variables with a mapping rank: $\mathcal{X} \to \mathbb{N}$[1] defining the rank of each variable.
A *graph (with variables)* is a system $G = (V_G, E_G, Y_G, s_G, t_G, att_G, l_G, ly_G)$ consisting of finite sets V_G, E_G, and Y_G of *nodes* (or *vertices*), *edges*, and *hyperedges*, *source* and *target functions* $s_G, t_G \colon E_G \to V_G$, an *attachment function* $att_G \colon Y_G \to V_G^*$[2], and *labeling functions* $l_G \colon V_G \cup$

[1] $\mathbb{N}$ denotes the set of natural numbers, including 0.
[2] V_G^* denotes sequences of nodes from V_G. This also includes hyperedges with zero tentacles.

$E_G \to C$, $ly_G \colon Y_G \to \mathcal{X}$ such that, for all $y \in Y_G$, $|att_G(y)| = rank(ly_G(y))$. We call the set of all graphs with variables $\mathcal{G}_\mathcal{X}$. A graph G is empty, denoted $\emptyset$, iff $V_G = \emptyset$ and $Y_G = \emptyset$. Let $G_{\overline{H}}$ be the graph G with subgraph H removed (if possible, i.e. if the resulting graph has no dangling edges).

Example 1 *Consider the graphs G, H in Figure 2 over the label alphabet $C = \{A, B, \square\}$ where the symbol $\square$ stands for the invisible edge label and is not drawn and $\mathcal{X} = \{u, v\}$ is a set of variables that have rank 4 and 2, respectively. The graph G contains five nodes with the labels A and B, respectively, seven edges with (invisible) label $\square$, and one hyperedge of rank 4 with label u. Additionally, the graph H contains a node, an edge, and a hyperedge of rank 2 with label v.*

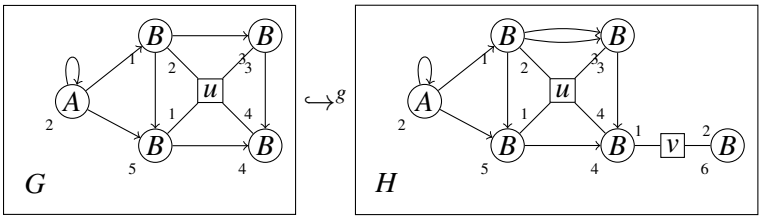


Figure 2: Graph morphisms with variables

Nodes are drawn as 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 with attachment nodes where the i-th tentacle has its number i written next to it and is attached to the i^{th} attachment node and the label of the hyperedge is inscribed in the box. Nodes with the invisible $\square$ label are drawn as points (•). For visibility reasons, we may abbreviate hyperedges of rank 2 by writing $\bullet \xrightarrow{x} \bullet$ instead of $\bullet \overset{1}{—} \boxed{x} \overset{2}{—} \bullet$.

Remark 1 *Edges are a special case of hyperedges, as they can be expressed as a hyperedge with two tentacles. However, our definition of graphs uses both, in order to have an easy distinction between "'terminal"' edges which cannot be changed and "'non-terminal"' hyperedges which are variables and can be substituted.*

Graph morphisms consist of structure-preserving mappings between the sets of nodes, edges and hyperedges of graphs.

Definition 2 (graph morphism with variables) A *(graph) morphism (with variables)* $g \colon G \to H$ consists of mappings $g_V \colon V_G \to V_H$, $g_E \colon E_G \to E_H$, and an injective mapping $g_Y \colon Y_G \hookrightarrow Y_H$ that preserve sources, targets, attachment nodes and labels, i.e. $s_H \circ g_E = g_V \circ s_G$, $t_H \circ g_E = g_V \circ t_G$, $att_H = g_V^* \circ att_G$, $l_H \circ g_V = l_G$, $l_H \circ g_E = l_G$, and $ly_H \circ g_Y = ly_G$, where $g^* \colon A^* \to B^*$ is the free symbolwise extension of g, defined by $g^*(a_1 \dots a_k) = g(a_1) \dots g(a_k)$ for $k \in \mathbb{N}$ and $a_i \in A$ $(i = 1, \dots, k)$.

The *composition* $h \circ g$ of g with a graph morphism $h \colon H \to M$ consists of the composed functions $h_V \circ g_V$, $h_E \circ g_E$, and $h_Y \circ g_Y$. 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 $G \cong H$. An injective graph morphism $m\colon G \hookrightarrow H$ is an *inclusion*, written $G \subseteq H$, if $\mathrm{V}_G \subseteq \mathrm{V}_H$, $\mathrm{E}_G \subseteq \mathrm{E}_H$ and $\mathrm{Y}_G \subseteq \mathrm{Y}_H$. For a 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.

Arbitrary graph morphisms are drawn by the usual arrows "$\to$"; the use of "$\hookrightarrow$" indicates an injective graph morphism. The actual mapping of elements is conveyed by indices, if necessary. For an example, see the graph morphism g in Figure 2.

The hyperedges are replaced by graphs according to a hyperedge replacement system. To describe how the original graph and the graph which replaces a hyperedge are connected, we need to map each tentacle of the hyperedge to a node in the latter graph.

Definition 3 (pointed graph with variables) A *pointed* graph with variables $\langle G, \mathrm{pin}_G\rangle$ is a graph with variables G together with a sequence $\mathrm{pin}_G = v_1 \dots v_n$ of pairwise distinct nodes from G. We write $\mathrm{rank}(\langle G, \mathrm{pin}_G\rangle)$ for the number n of *pinpoints* in pin_G. For $x \in \mathscr{X}$ with $\mathrm{rank}(x) = n$, $x^\bullet$ denotes the pointed graph with the nodes $v_1, \dots, v_n$, one hyperedge attached to $v_1 \dots v_n$, and pinpoints $v_1 \dots v_n$. $\mathrm{Pin}(G)$ denotes the set $\{v_1, \dots, v_n\}$ of pinpoints of $\langle G, \mathrm{pin}_G\rangle$.

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

Given a graph G, the *application* of the replacement pair $x/R \in \mathscr{R}$ to a hyperedge y with label x proceeds in two steps (see Figure 3): For a set X, let $G - X$ be the graph G with all elements in X removed, and for a graph H, let $G + H$ be the disjoint union of G and H.

1. Remove the hyperedge y from G, yielding the graph $G - \{y\}$.
2. Construct the disjoint union $(G - \{y\}) + R$ and fuse the i^{th} node in $\mathrm{att}_G(y)$ with the i^{th} attachment point of R, for $i = 1, \dots, \mathrm{rank}(y)$, yielding the graph H.

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

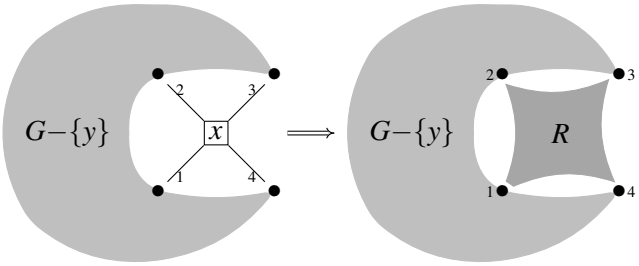
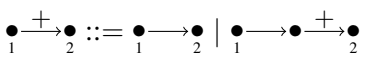


Figure 3: Application of replacement pair x/R.

Example 2 *The HR system $\mathscr{R}$ with the rules given in Backus-Naur form*



generates the set of all directed paths from node 1 to node 2.

In HR* conditions, we simultaneously substitute all hyperedges by graphs, which are generated according to a HR system.

Definition 5 (substitution) A *substitution* induced by a hyperedge replacement system $\mathscr{R}$ is a mapping $\sigma\colon \mathscr{X} \to \mathscr{G}_{\mathscr{X}}$ with $\sigma(x) \in \mathscr{R}(x)$ for all x in the domain of σ. The set of all substitutions induced by $\mathscr{R}$ is denoted by $\Sigma_{\mathscr{R}}$. Application of σ to a graph G, denoted $G \Rightarrow G^{\sigma}$, is obtained by simultaneous substitution of all hyperedges in $y \in Y_G$ by $\sigma(\mathrm{ly}_G(y))$ (see Figure 4).

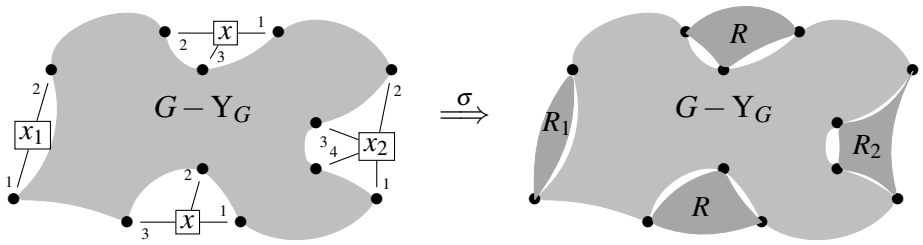


Figure 4: Substitution of hyperedges.

Now, we can define HR* conditions. They allow one to use variables for structures of arbitrary size, and to "peek into" such variables and formulate properties of the graphs that the variable is substituted by.

Definition 6 (HR* graph condition) A HR* *(graph) conditions (over $\mathscr{R}$)*, short *condition*, consists of a condition with variables and a HR system $\mathscr{R}$.
Conditions with variables are inductively defined as follows.

(1) For a graph P, `true` is a condition over P.

(2) For an injective morphism $a\colon P \hookrightarrow C$ and a condition c over C, $\exists(a,c)$ is a condition over P.

(2') For graphs P,C and a condition c over C, $\exists(P \sqsupseteq C, c)$ is a condition over P.

(3) For an index set J and conditions $(c_j)_{j\in J}$ over P, $\neg c_1$ and $\wedge_{j\in J} c_j$ are conditions over P [3].

HR* conditions c over $\mathscr{R}$ are denoted by $\langle c, \mathscr{R}\rangle$, or c if $\mathscr{R}$ is clear from the context. A HR* condition is *finite* if every index set J in the condition is finite; we will assume finite conditions in the following if not explicitly stated otherwise. In particular, the constructions and transformations througout this paper assume that all input and output HR* conditions and logical formulas are finite.

[3] Usually, J is a set of natural numbers from 1 to some number k.

The following abbreviations are used: $\exists a$ abbreviates $\exists(a, \mathtt{true})$, $\forall(\_, c)$ abbreviates $\neg\exists(\_, \neg c)$, $\mathtt{false}$ abbreviates $\neg\mathtt{true}$, and $\vee_{j\in J} c_j$ abbreviates $\neg\wedge_{j\in J}\neg c_j$. The domain of a morphism may be omitted if no confusion arises: $\exists C$ can replace $\exists(P \hookrightarrow C)$ in this case.

Example 3 *The following* HR* *condition intuitively expresses "There exists a path from the image of node 1 to the image of node 2".*

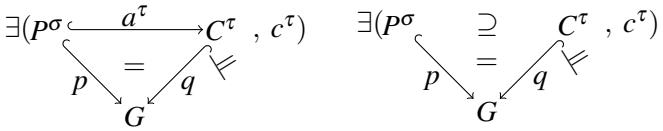
$$\exists(\bullet_1 \xrightarrow{+} \bullet_2) \quad \textit{with} \quad \bullet_1 \xrightarrow{+} \bullet_2 ::= \bullet_1 \longrightarrow \bullet_2 \mid \bullet_1 \longrightarrow \bullet \xrightarrow{+} \bullet_2$$

We now give the formal semantics for HR* conditions.

Definition 7 (satisfaction of HR* conditions) For an injective morphism $p\colon P^\sigma \hookrightarrow G$, the *satisfaction* of a condition $\langle c, \mathscr{R}\rangle$ by a substitution $\sigma \in \Sigma_{\mathscr{R}}$, written $p \models_\sigma c$, is inductively defined as follows.

(1) p satisfies $\mathtt{true}$.

(2) p satisfies $\exists(a,c)$ by σ for a morphism $a\colon P \hookrightarrow C$ if there is a (partial) substitution τ such that $P^\sigma = P^\tau$ and an injective morphism $q\colon C^\tau \hookrightarrow G$ such that $q \circ a^\tau = p$ and q satisfies c^τ (left diagram), where $a^\tau\colon P^\sigma \hookrightarrow C^\tau$ is the morphism with $a^\tau(o) = a(o)$ for all $o \in \mathrm{V}_P \uplus \mathrm{E}_P$ and $a^\tau(y) = y$ for all $y \in \mathrm{Y}_P$.



(2') p satisfies $\exists(P \sqsupseteq C, c)$ by σ if there are a substitution τ with $P^\sigma = P^\tau$, an inclusion $C^\tau \subseteq P^\sigma$ and an injective morphism $q\colon C^\tau \hookrightarrow G$ such that $p = q_{|P^\sigma}$ and q satisfies c^τ (right diagram), where $q_{|P^\sigma}$ is the morphism q restricted to P^σ: $q_{|P^\sigma}(x) = q(x)$ if $x \in P^\sigma$ and $\perp$ otherwise.

(3) p satisfies $\neg c$ by σ if p does not satisfy c by σ. p satisfies $\wedge_{i\in I} c_i$ by σ if p satisfies all c_i by σ $(i \in I)$.

A graph G *satisfies* a condition c over $\emptyset$ if the morphism $\emptyset \hookrightarrow G$ satisfies c. We write $G \models_\sigma c$ to denote that a graph G satisfies c by σ and $G \models c$ if there is a $\sigma \in \Sigma_{\mathscr{R}}$ such that $G \models_\sigma c$.

Example 4 *The following example shows a* HR* *condition expressing "There is a path from a node to another, and all nodes on this path have at least three outgoing edges to different nodes".*

$$\underbrace{\exists(\bullet_1 \xrightarrow{+} \bullet_2,}_{(1)} \quad \underbrace{\forall(\bullet_1 \xrightarrow{+} \bullet_2 \sqsupseteq \bullet_3,}_{(2)} \underbrace{\exists \begin{smallmatrix}\bullet & \bullet & \bullet \\ & \nwarrow\uparrow\nearrow & \\ & \bullet_3 & \end{smallmatrix}))}_{(3)} \quad \textit{with} \quad \bullet_1 \xrightarrow{+} \bullet_2 ::= \bullet_1 \longrightarrow \bullet_2 \mid \bullet_1 \longrightarrow \bullet \xrightarrow{+} \bullet_2$$

In subformula (1), the existence of the path is established. Part (2) quantifies over every node that is contained in the path, while part (3) ensures that each such node has three outgoing edges to different nodes.

Remark 2 *The idea of enhancing nested conditions with variables is also used in [PP12] for E-conditions. In contrast to* HR* *conditions, the variables in E-conditions are not substituted by graphs, but by labels or attributes, so E-conditions cannot express non-local conditions, but can work with infinite label alphabets (e.g. natural numbers) and perform calculations on attributes.*

3 Semantical variants of HR* conditions

HR* conditions are well-suited as a graphical formalism to express local and non-local properties of graphs. While HR* conditions borrow the quantors and Boolean operations from logical formulas, there are some important differences to the latter:

1. In a HR* condition, any node or edge o introduced by a condition $\exists(P \hookrightarrow P \oplus \{o\}, c)$ is disjoint with any node in P. In a logical formula, in contrast, every newly-quantified object may be identified with an already existing object o', as long as this is not explicitly forbidden by some sub-formula $o \neq o'$.

2. As we will see later, with logical formulas, it is easier to express the replacement of a single hyperedge by some graph, than the substitution of each occurrence of a variable with the same graph.

To this end, two varitions of the semantics are introduced. The first variation, called $\mathscr{A}$-*satisfiability*, makes it possible to identify nodes or edges in HR* conditions. The second variation uses replacement instead of substitution, dropping the restriction that every occurrence of a variable x in a HR* condition has to be substituted by the same (more exactly, an isomorphic) graph. We show that these variations do not increase the expressiveness of HR* conditions. This result will later be needed to show that HR* conditions can be converted into logical formulas.

In logical formulas, distinct variables do not necessarily mean distinct objects: one has to explicitly state that two variables x, y stand for distinct objects with a formula $\neg x \doteq y$. Nodes and edges in HR* conditions, on the other hand, are distinct by default. This can also be done with a variant on the semantics of HR* conditions, $\mathscr{A}$-satisfaction.

The definition of $\mathscr{A}$-satisfaction is similar to Definition 7, except that all of the morphisms are allowed to be non-injective. Note that the inclusion in (2') is not touched by this.

Definition 8 ($\mathscr{A}$-satisfaction of HR* conditions) For cases (1), (2), (2') and (3), $\mathscr{A}$-*satisfaction* is defined as for satisfaction with all injective morphisms replaced by arbitrary ones. A graph G $\mathscr{A}$-satisfies c if morphism $\emptyset \to G$ satisfies $\langle c, \mathscr{R} \rangle$, denoted $G \models_{\mathscr{A},\sigma} c$; and $G \models_{\mathscr{A}} c$ if there is a $\sigma \in \Sigma_{\mathscr{R}}$ such that $G \models_{\mathscr{A},\sigma} c$.

The consequence of this definition is that nodes and edges in $\mathscr{A}$-satisfiable HR* conditions no longer have a disjoint image in graph G by default, but may be identified. However, one can still forbid this identification to express any satisfiable HR* condition with a $\mathscr{A}$-satisfiable HR* condition.

Lemma 1 (from satisfaction to $\mathscr{A}$-satisfaction) *For every* HR* *condition* c*, there is a* HR*

condition $\mathrm{Cond}_{\mathscr{A}}(c)$ *such that for every graph* G,

$$G \models c \iff G \models_{\mathscr{A}} \mathrm{Cond}_{\mathscr{A}}(c).$$

For the construction, proof and an example, see Appendix A.

For $\mathscr{A}$-satisfiability, substitution of hyperedges (i.e. all edges with the same label are replaced by isomorphic graphs) is equivalent to replacement of hyperedges (i.e. edges with the same label may be replaced by different graphs). It is easy to see that, for every HR* condition using replacement, an equivalent HR* condition can be constructed using substitution, simply by giving each hyperedge a unique label and cloning the rules. To distinguish between satisfaction by substitution and by replacement, we will use $p \models_{\sigma} c$ and $p \models_{\mathscr{R}} c$, respectively. Furthermore, $c^{\mathscr{R}}$ denotes replacement of all variables in c according to $\mathscr{R}$, analogous to c^{σ} for $\sigma \in \Sigma_{\mathscr{R}}$.

Lemma 2 (from substitution to replacement) *For every* HR* *condition* $\langle c, \mathscr{R} \rangle$, *there is a* HR* *condition* $\langle c', \mathscr{R}' \rangle$ *such that for all graphs* G,

$$\exists \sigma \in \Sigma_{\mathscr{R}}. G \models_{\mathscr{A},\sigma} \iff \exists r \in \Sigma_{\mathscr{R}'}. G \models_{\mathscr{A},\mathscr{R}}$$

For the construction, proof and an example, see Appendix B.

4 Graph formulas

A classic approach to express properties of a graph is to use logical formulas over graphs. The expressiveness of such formulas depends on the underlying logic. We begin with the definition of second-order graph formulas, following [vD04]. Second-order formulas can quantify over individual objects in the underlying universe, as well as over arbitrary relations over the underlying universe, allowing one to formulate many interesting graph properties. For a comparative overview on the power of several graph logics, see [Cou96, CW05, CE12]; our definition of second-order logic is equivalent to that in [Cou96], except that we also consider node and edge labels.

Definition 9 (second-order graph formulas) Let C be a set of labels, $\mathscr{V}_1$ be a (denumerable) set of *individual* (or *first-order*) variables $x_0, x_1, \ldots$ and $\mathscr{V}_2$ a (denumerable) set of *relational* (or *second-order*) variables $X_0, X_1, \ldots$, together with a function $\mathrm{rank}\colon \mathscr{V}_2 \to \mathbb{N} - \{0\}$ that maps to each variable in $\mathscr{V}_2$ a positive natural number, its *rank*. We let $\mathscr{V} = \mathscr{V}_1 \cup \mathscr{V}_2$ be the set of all variables.

Second-order graph formulas, short *SO formulas*, are defined inductively:
$\mathrm{inc}(x,y,z)$, $\mathrm{lab}_b(x)$ and $x \doteq y$ are SO graph formulas for individual variables $x, y, z \in \mathscr{V}_1$ and labels $b \in C$. For any variable $x \in \mathscr{V}_1$ and SO formula F, $\exists x.F$ is an SO formula and also a term. Also, for any variable $X \in \mathscr{V}_2$ with $\mathrm{rank}(X) = k$ and terms $t_1, \ldots, t_k$, $X(t_1, \ldots, t_k)$ is both an SO formula and a term. Finally, Boolean expressions over SO formulas c, d are SO formulas: `true`, $\neg F$, $F_1 \wedge F_2$.

For a non-empty graph G, let $\mathrm{D}_G^{\times}$ be the set of all relations over $\mathrm{D}_G = \mathrm{V}_G \cup \mathrm{E}_G$. The semantics $G[\![F]\!](\theta)$ of a SO formula F under assignment $\theta\colon \mathscr{V} \to \mathrm{D}_G \cup \mathrm{D}_G^{\times}$ is inductively defined as follows:

1. $G[\![\mathrm{inc}(e,x,y)]\!](\theta) \iff \theta(e) \in \mathrm{E}_G$, $\mathrm{s}_G(\theta(e)) = \theta(x)$, and $\mathrm{t}_G(\theta(e)) = \theta(y)$,
   $G[\![\mathrm{lab}_b(x)]\!](\theta) \iff \theta(x) = \mathrm{lv}_G(b)$ or $\theta(x) = \mathrm{le}_G(b)$, and
   $G[\![x \doteq y]\!](\theta) \iff \theta(x) = \theta(y)$.

2. $G[\![\mathtt{true}]\!](\theta) \iff \mathtt{true}$, $G[\![\neg F]\!](\theta) \iff \neg G[\![F]\!](\theta)$,
   $G[\![F \wedge F']\!](\theta) \iff G[\![F]\!](\theta) \wedge G[\![F']\!](\theta)$, and
   $G[\![\exists x.F]\!](\theta) \iff G[\![F]\!](\theta\{x/d\})$ for some $d \in \mathrm{D}_G$,
   where $\theta\{x/d\}(y) = d$ if $x = y$ and $\theta\{x/d\}(y) = \theta(y)$ otherwise.

3. $G[\![\exists X.F]\!](\theta) \iff G[\![F]\!](\theta\{X/D\})$ for some $d \in \mathrm{D}_G^{\times}$.
   $G[\![X(t_1,\ldots,t_k)]\!](\theta) \iff (G[\![t_1]\!](\theta),\ldots,G[\![t_k]\!](\theta)) \in \theta(X)$.

4. $G[\![\neg F]\!](\theta) \iff \neg G[\![F]\!](\theta)$ and $G[\![F \wedge F']\!](\theta) \iff G[\![F]\!](\theta) \wedge G[\![F']\!](\theta)$.

A non-empty graph G *satisfies* a SO formula F, denoted by $G \models F$, iff, for all assignments $\theta\colon \mathscr{V} \to \mathrm{D}_G \cup \mathrm{D}_G^{\times}$, $G[\![F]\!](\theta) = \mathtt{true}$.

Example 5 *The SO formula below is true for every graph which has a non-trivial automorphism, i.e. an automorphism which is not the identity:*

$$\exists X.[\beta_{\mathrm{inj}}(X) \wedge \beta_{\mathrm{total}}(X) \wedge \beta_{\mathrm{surj}}(X) \wedge \beta_{\mathrm{ntriv}}(X) \wedge \beta_{\mathrm{predg}}(X)]$$

where the subformulas are defined as follows:

- $\beta_{\mathrm{inj}}(X) = \forall x,y,z.(X(x,y) \wedge X(x,z)) \Rightarrow y \doteq z \wedge (X(x,z) \wedge X(y,z)) \Rightarrow x \doteq y$
  expresses that relation X is injective,
- $\beta_{\mathrm{total}}(X) = \forall x \exists y.X(x,y)$ *expresses that X is total,*
- $\beta_{\mathrm{surj}}(X) = \forall x \exists y.X(y,x)$ *expresses that X is surjective,*
- $\beta_{\mathrm{ntriv}}(X) = \exists x,y.x \neq y \wedge X(x,y)$ *expresses that X is non-trivial,*
- $\beta_{\mathrm{predg}}(X) = \forall e,x,y,e',x',y'.(\mathrm{inc}(e,x,y) \wedge (X(e,e') \wedge X(x,x') \wedge X(y,y')) \Rightarrow \mathrm{inc}(e',x',y')$ *expresses that X preserves edges, i.e. for every pair of nodes x,y connected by an edge and related to nodes x',y' by relation X, x' and y' are connected by an edge.*

Counting monadic second-order graph formulas are a subclass of second-order graph formulas and an extension of monadic second-order graph formulas [Cou96]. Like monadic second-order graph formulas, they allow quantification over individual nodes and edges as well as quantification over unary relations, i.e. sets of nodes and edges. Furthermore, they have a special quantifier that allows one to count modulo natural numbers.

Definition 10 (counting monadic second-order graph formulas) A *counting monadic second-order graph formula*, short *CMSO formula*, is defined as follows. Every SO formula where every relational variable X has $\mathrm{rank}(X) = 1$ is a CMSO formula, and for every natural number $m \in \mathbb{N}$ and every CMSO formula F, $\exists^{(m)}x.F(x)$ is a CMSO formula.

For a non-empty graph G,

$$G[\![\exists^{(m)}x.F(x)]\!](\theta) \iff |\{u \in V_G \cup E_G : G[\![F(u)]\!](\theta)\}| \equiv 0 \pmod{m}.$$

A CMSO formula is a *node-CMSO formula* if counting is only allowed over nodes, i.e. every subformula $\exists^{(m)}x.F$ is equivalent to $\exists^{(m)}x.\,\mathrm{node}(x) \wedge F'$, where $\mathrm{node}(x) = \nexists y,z.\,\mathrm{inc}(x,y,z)$ states that x is a node. A CMSO formula is a *monadic second-order formula*, short *MSO formula*, if it contains no subformulas of the form $\exists^{(m)}x.F$.

Example 6 *The node-CMSO formula* $\exists^{(2)}x.\,\mathrm{node}(x)$ *expresses "The graph has an even number of nodes".*

5 Expressing node-CMSO formulas with HR* conditions

In [HR10], a variant of HR* conditions was introduced and shown to be at least as strong as MSO formulas. We now go one step further and show that HR* formulas can also express arbitrary node-CMSO formulas.

Theorem 1 (node-CMSO formulas to HR* conditions) *For every node-CMSO formula F, there is a* HR* *graph condition* $\mathrm{Cond}(F)$ *such that for all graphs $G \in \mathscr{G}$, $G \models F$ iff $G \models \mathrm{Cond}(F)$.*

We use hyperedge replacement to count the nodes: It is easy to construct a grammar which generates all discrete graphs (i.e. with no edges) with $k*m$ nodes, where m is a fixed number and $k \in \mathbb{N}$ is variable. For all nodes inside the generated subgraph, the property F to be counted is checked. Also, F must not hold for any node outside of the generated subgraph.

Construction. For a graph P and a formula F, $\mathrm{Cond}(P,F)$ is defined as follows. For any MSO formula, Cond is defined as in [HR10]. Otherwise, i.e. for formulas of the form $\exists^{(m)}v.F$,

$$\mathrm{Cond}(P, \exists^{(m)}v.F) = \exists(\boxed{Y}, \forall(\boxed{Y} \sqsupseteq \bullet_v, \mathrm{Cond}(F(v))) \wedge \nexists(\boxed{Y}\ \bullet_v, \mathrm{Cond}(F(v))))$$
with $\boxed{Y} ::= \emptyset \mid \boxed{Y}\ D_m$, where D_m is a discrete graph with m nodes.

Example 7 *Take as an example the node-CMSO formula expressing "There is an even number of nodes":* $\exists^{(2)}x.\mathrm{node}(x)$. *Using the construction above, this yields* $\exists(\boxed{Y}, \forall(\boxed{Y} \sqsupseteq \bullet_1, \exists(\bullet_1)) \wedge \nexists(\boxed{Y}\ \bullet_2, \exists(\bullet_2)))$ *with* $\boxed{Y} ::= \emptyset \mid \boxed{Y}\ \bullet\bullet$. *Simplification of the* HR* *condition yields* $\exists(\boxed{Y}, \nexists(\boxed{Y}\ \bullet_2))$ *with* $\boxed{Y} ::= \emptyset \mid \boxed{Y}\ \bullet\bullet$.

Proof. For MSO formulas, see the proof in [HR10]. For formulas $\exists^{(m)}x.\phi(x)$ and every graph G, assume that $G \models \phi(x) \iff G \models \mathrm{Cond}(\phi(x))$ and let $p\colon \emptyset \hookrightarrow G$.
By the definition of HR* satisfaction (Def. 7) and construction 5,
$G \models \mathrm{Cond}(\exists^{(m)}x.\phi(x)) \Leftrightarrow p \models \mathrm{Cond}(\exists^{(m)}x.\phi(x))$
$\Leftrightarrow p \models \exists(\emptyset \hookrightarrow \boxed{Y}, \forall(\boxed{Y} \sqsupseteq \bullet_x, \mathrm{Cond}(\phi(x))) \wedge \nexists(\boxed{Y} \hookrightarrow \boxed{Y}\ \bullet_x, \mathrm{Cond}(\phi(x)))).$
Performing the substitution of hyperedge $\boxed{Y}$ yields
$\Leftrightarrow \exists n \in \mathbb{N}.p \models \exists(\emptyset \hookrightarrow D_{n*m}, \forall(D_{n*m} \sqsupseteq \bullet_x, \mathrm{Cond}(\phi(x))) \wedge \nexists(D_{n*m} \hookrightarrow D_{n*m}\bullet_x, \mathrm{Cond}(\phi(x)))).$
We use the semantics of HR* conditions again to get
$\Leftrightarrow \exists n \in \mathbb{N}.\exists D_{n*m} \overset{q_a}{\hookrightarrow} G.p = q_a \circ a \wedge$
$\forall \emptyset \overset{q_b}{\hookrightarrow} \bullet_x.q_b(\bullet_1) \subseteq q_a(D_{n*m}) \wedge q_b \models \mathrm{Cond}(\phi(x))^\sigma \wedge$
$\nexists(D_{n*m} + \bullet_x \overset{q_c}{\hookrightarrow} G.q_a = q_c \circ c \wedge q_c \models \mathrm{Cond}(\phi(x))^\sigma)$

and, by the injectivity of the morphisms,
$\exists n \in \mathbb{N}.\exists D_{n*m} \hookrightarrow G.\forall(\bullet_x \subseteq D_{n*m}.\bullet_x \models \mathrm{Cond}(\phi(x))^\sigma)$
$\wedge \nexists(D_{n*m} + \bullet_x \subseteq G.\bullet_x \models \mathrm{Cond}(\phi(x))^\sigma)$.
Using simple arithmetics and set theory, it is easy to see that
$\Leftrightarrow \exists n \in \mathbb{N}.|\{\bullet_x \subseteq V_G \mid \bullet_x \models \mathrm{Cond}(\phi(x))^\sigma\}| \geq n*m$
$\wedge \neg(|\{\bullet_x \subseteq V_G \mid \bullet_x \models \mathrm{Cond}(\phi(x))^\sigma\}| \geq n*m+1)$
$\Leftrightarrow \exists n \in \mathbb{N}.|\{v \in V_G : G \models \mathrm{Cond}(\phi(v))\}| = n*m$
$\Leftrightarrow \exists n \in \mathbb{N}.|\{v \in V_G : G \models \phi(v)\}| = n*m$.
Using the initial assumption and Definition 10, we get
$\Leftrightarrow |\{v \in V_G : G \models \phi(v)\}| \equiv 0 \pmod m$
$\Leftrightarrow G[\![\exists^{(m)}x.F(x)]\!](\theta) = \mathtt{true}$
$\Leftrightarrow G \models \exists^{(m)}x.\phi(x)$. □

Remark 3 *Using node-counting, it is possible to simulate edge-counting. A property $P(e)$ is valid for a number $n \equiv 0 \pmod m$ edges iff it is valid for n outgoing edges of k nodes. Thus, one can group the nodes by the number of outgoing edges which fulfill P, and count the nodes in each group.*

As an example, for $m = 2$, the edge-CMSO formula $\exists^{(2)}e.P(e)$ is valid iff there is an arbitrary number k_0 of nodes with a number $l_0 \equiv 0 \pmod 2$ of outgoing edges e which satisfy $P(e)$ and an even number $k_1 \equiv 0 \pmod 2$ of nodes with an odd number $l_1 \equiv 1 \pmod 2$ of outgoing edges e which satisfy $P(e)$. A similar scheme can be used for any m, although it gets quite complicated for greater values of m.

This scheme can be translated into a HR* *condition. We already showed that* HR* *condition can count nodes modulo m. (Outgoing) edges can also be counted modulo m using a HR system which generates "stars", i.e. graphs with a node v_0 in the middle, from which edges go out to otherwise isolated nodes, as seen in Figure 5. HR systems for generating stars are described in more detail in [Hab92]. In our case, however, the star graphs may be "collapsed", i.e. a node may have more than one outgoing edge to a single node. This can also be described with a HR system.*

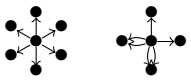


Figure 5: A "star" graph (left) and a "collapsed star" graph, both with 6 edges.

6 Expressing HR* conditions with SO formulas

With a lower bound for the expressiveness of HR* conditions established, we now turn to the upper bound and show that every HR* condition can be expressed as a SO formula. The main difficulty here lies in the representation of the replacement process within the formula. The transformation is made somewhat easier by using a slightly changed semantics for HR* conditions. We use replacement instead of substitution and $\mathscr{A}$-satisfiability.

Since every HR* condition can be transformed into an equivalent $\mathscr{A}$-satisfiable one, we can prove that every HR* condition can be expressed as a SO graph formula, which we will now do step by step.

Theorem 2 (from HR* conditions to SO formulas) *For every* HR* *graph condition c, there is a second-order graph formula* SO(c) *such that for all graphs* $G \in \mathscr{G}$,

$$G \models c \iff G \models \mathrm{SO}(c).$$

The construction of SO has to capture several things, which are done by appropriate sub-constructions:

1. The logical structure of the HR* condition has to be preserved. This is trivial, as the Boolean operators and quantors of HR* conditions can be represented by the same operators in SO formulas.
2. The graph morphisms and graphs in HR* conditions have to be translated into an SO formula. This is done by sub-construction SOgra.
3. The hyperedge replacement system, along with the process of hyperedge replacement, have to be encoded in SO formulas. Sub-construction SOsys fulfills this task.
4. For HR* conditions of the form $\exists(P \sqsupseteq C, c)$, we need to represent the sets P^σ and C^τ for some replacements σ, τ. This is done by sub-construction SOset.

We use several helper constructions that we will present and prove individually later. Construction $\mathrm{SOgra}(G,F)$ is used to represent a graph G as a formula, where F is some nested subformula. SOsys represents the replacement system of the HR* condition. Finally, $\mathrm{SOset}(G,X)$ is used to collect all nodes and edges in graph G^σ (i.e. after replacement) inside a set variable X. This is needed to check whether there is an inclusion $C^\tau \subseteq P^\sigma$ for a HR* condition $\exists(P \sqsupseteq C, c)$.

Construction. Without loss of generality, $P \hookrightarrow C$ is an inclusion. For a condition c with HR system $\mathscr{R}$, we let $\mathrm{SO}(\langle c, \mathscr{R}\rangle) = \mathrm{SOsys}(\mathscr{R}) \wedge \mathrm{SO}(c)$ and define

(1) $\mathrm{SO}(\texttt{true}) = \texttt{true}$.

(2) $\mathrm{SO}(\exists(P \hookrightarrow C, c)) = \mathrm{SOgra}(C - P, \mathrm{SO}(c))$.

(2') $\mathrm{SO}(\exists(P \sqsupseteq C, c)) = \mathrm{SOgra}(C, \exists X_P, X_C.\, \mathrm{SOset}(P, X_P) \wedge \mathrm{SOset}(C, X_C) \wedge X_C \subseteq X_P \wedge \mathrm{SO}(c))$, where X_P, X_C are fresh second-order variables of rank 1 (i.e. set variables) and the relation $\subseteq$ is constructed in SO logic as usual: $X_C \subseteq X_P = \forall x.x \in X_C \Rightarrow \exists y \in X_P.x \doteq y$.

(3) $\mathrm{SO}(\neg c) = \neg \mathrm{SO}(c)$ and $\mathrm{SO}(\bigwedge_{i\in I} c_i) = \bigwedge_{i\in I} \mathrm{SO}(c_i)$.

The construction is straightforward for HR* conditions of the form (1) and (4), as these have equivalent constructs in SO formulas. For HR* conditions of form (2), it suffices to state the existence of the graph $C-P$ and to translate subcondition c into a SO formula, too. The construction gets a bit more complicated for case (2'). The SO formula has to state that graph C exists, that

the sets X_C of nodes and edges in C^τ is a subset of the set X_P of nodes and edges in P^σ for some substitutions τ and σ,

The transformation SOgra represents a graph with variables as a SO formula. These are needed to express HR* conditions of the form $\exists(P \hookrightarrow^a C, c)$. Such a HR* condition is equivalent to a SO formula stating that, given P, the graph $C-P$ is present. For example, HR* condition $\exists(\emptyset \hookrightarrow \underset{1}{\bullet}\!\!\supset)$ is equivalent to the SO formula $\exists v, e.v \neq e \,\mathrm{inc}(e,v,v)$, stating that the graph contains a node v and an edge e (which are, of course, not identical) and e links v to itself.

Lemma 3 *For graphs $R \in \mathscr{G}_\mathscr{X}$ and $G \in \mathscr{G}$,*

$$G \models_\mathscr{A} \exists(\emptyset \hookrightarrow R - \mathrm{Y}_R) \iff G \models \mathrm{SOgra}(R - \mathrm{Y}_R, \mathtt{true}).$$

The construction is split in three parts for nodes, edges and hyperedges, respectively. The construction for nodes and edges is quite straightforward: we state the existence of every node and edge and then specify the node and edge labels and the incidence relation for the edges. For the hyperedges, we state the existence of each hyperedge label x as a $(\mathrm{rank}(x))$-ary relation, where the elements represent the attachment points of the tentacles.

Construction. For a set A and SO formula F, let $\exists F$ be the existential closure of F and $\dot{\exists} F = \exists F \wedge \bigwedge_{a \neq b}^{a,b \in A} (\neg a \doteq b)$ be the existential closure of F with disjointness check. Define the universal closure $\forall F$ analogously. For a graph with variables G and a SO formula F, we define

$$\begin{aligned}
\mathrm{SOnod}(G,F) &= \exists \textstyle\bigwedge_{v \in \mathrm{V}_R} \mathrm{lab}_{\mathrm{l}_G(v)}(v) \wedge F \\
\mathrm{SOedg}(G) &= \exists \textstyle\bigwedge_{e \in \mathrm{E}_R} \mathrm{lab}_{\mathrm{l}_G(e)}(e) \wedge \mathrm{inc}(e, \mathrm{s}_G(e), \mathrm{t}_G(e)) \\
\mathrm{SOhyp}(G) &= \dot{\exists} \textstyle\bigwedge_{\mathrm{ly}_G(y) \mid y \in \mathrm{Y}_R} . \mathrm{ly}_G(y)(\mathrm{att}_G(y)_{1,\ldots,k}) \\
\mathrm{SOgra}(G,F) &= \mathrm{SOnod}(G, \mathrm{SOedg}(G) \wedge \mathrm{SOhyp}(G) \wedge F)
\end{aligned}$$

Remark 4 *Note that this representation of hyperedges with SO formulas only works if the hyperedge has at least one tentacle. This in no problem: it is easy to see that every hyperedge with zero tentacles in a* HR* *condition can be replaced by a hyperedge with one tentacle. Without loss of generality, we assume that every* HR* *condition is free of hyperedges with zero tentacles.*

Example 8 *The two graphs with variables from Example 1, can be encoded SO formulas as follows:*

$$\begin{aligned}
&\mathrm{SOgra}(G) = \exists v_1, v_2, v_3, v_4, v_5 . \mathrm{lab}_B(v_1) \wedge \mathrm{lab}_A(v_2) \wedge \mathrm{lab}_B(v_3) \wedge \mathrm{lab}_B(v_4) \wedge \mathrm{lab}_B(v_5) \\
&\quad \wedge \exists e_1, \ldots, e_7 . \mathrm{lab}_\square(e_1) \wedge \ldots \wedge \mathrm{lab}_\square(e_7) \\
&\qquad \wedge \mathrm{inc}(e_1, v_1, v_3) \wedge \mathrm{inc}(e_2, v_1, v_5) \wedge \ldots \wedge \mathrm{inc}(e_7, v_5, v_4) \\
&\quad \wedge \exists u . u(v_5, v_1, v_3, v_4)
\end{aligned}$$

$$\begin{aligned}
&\mathrm{SOgra}(H) = \exists v_1, \ldots, v_6 . \mathrm{lab}_B(v_1) \wedge \mathrm{lab}_A(v_2) \wedge \ldots \wedge \mathrm{lab}_B(v_6) \\
&\quad \wedge \exists e_1, \ldots, e_8 . \mathrm{lab}_\square(e_1) \wedge \ldots \wedge \mathrm{lab}_\square(e_8) \\
&\qquad \wedge \mathrm{inc}(e_1, v_1, v_3) \wedge \mathrm{inc}(e_2, v_1, v_3) \wedge \ldots \wedge \mathrm{inc}(e_8, v_5, v_4) \\
&\quad \wedge \exists u, v . u(v_5, v_1, v_3, v_4) \wedge v(v_4, v_6)
\end{aligned}$$

Proof. Assume $G \models_{\mathscr{A}} \exists(\emptyset \hookrightarrow^{a} R_{\overline{Y}_R})$.
By the semantics of HR* conditions, for $p\colon \emptyset \to G$, this is equivalent to
$\Leftrightarrow p \models_{\mathscr{A}} \exists(\emptyset \to^{a} R_{\overline{Y}_R})$
$\Leftrightarrow \exists q\colon R_{\overline{Y}_R} \to G.p = q \circ a \wedge q \models_{\mathscr{A}} \texttt{true}$.
By the definition of morphisms, this equals
$\Leftrightarrow \exists q\colon R_{\overline{Y}_R} \to G.\forall o \in \mathrm{D}_R.p(o) = q(a(o))$,
and because the domain of p and a is the empty graph,
$\Leftrightarrow \exists R' \in \mathscr{G}.R_{\overline{Y}_R} \cong R' \wedge R' \subseteq G$
which can be expressed as a SO formula
$\Leftrightarrow \exists R' \in \mathscr{G}.\exists_{v \in \mathrm{V}'_R}.(\bigwedge_{v \in \mathrm{V}'_R}(\mathrm{lab}_{\mathrm{l}(v)}(v)) \wedge \exists(\bigwedge_{e \in \mathrm{E}'_R}(\mathrm{lab}_{\mathrm{l}(e)}(e)) \wedge \mathrm{inc}(e,\mathrm{s}(e),\mathrm{t}(e))))$
$\Leftrightarrow \exists R' \in \mathscr{G}.\mathrm{SOnod}(R',\mathrm{SOedg}(R') \wedge \mathrm{SOhyp}(R'))$
which equals the definition of SOgra:
$\Leftrightarrow G \models \mathrm{SOgra}(R',\texttt{true})$.
Since $R' \cong R_{\overline{Y}_R}$,
$\Leftrightarrow G \models \mathrm{SOgra}(R_{\overline{Y}_R},\texttt{true})$. □

We now turn to the simulation of the hyperedge replacement process itself.

Lemma 4 *For a graph $S \in \mathscr{G}_{\mathscr{X}}$, hyperedge replacement system $\mathscr{R}$ and graph G,*

$$G \models_{\mathscr{A}} \langle \exists(\emptyset \to S), \mathscr{R} \rangle \iff G \models \mathrm{SOgra}(S) \wedge \mathrm{SOsys}(\mathscr{R}).$$

The main idea of the construction is to represent hyperedges as relations over nodes. A hyperedge with k nodes is represented as a k-ary relation, where the elements represent the nodes attached to the hyperedge by its k tentacles. In order to keep track of all nodes and edges that replace each hyperedge in a graph G^{σ}, we use sets $\mathrm{Set}(x(v_1,\ldots,v_k))$, i.e. sets which are dependent on a x-labeled hyperedge attached to points $v_1,\ldots,v_k$. Let $o \in \mathrm{Set}(x(v_1,\ldots,v_k))$ abbreviate the second-order $k+1$-ary relation $\mathrm{Set}_x(v_1,\ldots,v_k,o)$, which denotes that o is element of a set dependent on x and $v_1,\ldots,v_k$.

Construction. For any replacement pair x/R with $\mathrm{rank}(x) = k$ and HR system $\mathscr{R}$, let

$$\begin{aligned}\mathrm{SOsys}(\mathscr{R}) = \quad & \textstyle\bigwedge_{x \in \mathscr{X}} \exists x, \mathrm{Set}_x.\forall v_1,\ldots,v_k.x(v_1,\ldots,v_k) \Rightarrow \bigvee_{x/R \in \mathscr{R}}(\mathrm{SOgra}(R) \\ & \wedge \mathrm{SOset}(R,\mathrm{Set}(x(v_1,\ldots,v_k))))\end{aligned}$$

where SOset is needed to keep track of the elements in G^{σ} and will be explained in the following.

Example 9 *The HR system from Example 2*

$$\mathscr{R} = \bullet_1 \xrightarrow{+} \bullet_2 ::= \bullet_1 \longrightarrow \bullet_2 \mid \bullet_1 \longrightarrow \bullet_3 \xrightarrow{+} \bullet_2$$

is transformed into the SO formula

$$\begin{aligned}\mathrm{SOsys}(\mathscr{R}) = \forall v_1, v_2.+(v_1,v_2) \Rightarrow (\exists e.(\mathrm{inc}(e,v_1,v_2) \wedge \mathrm{SOset}(\bullet_1 \longrightarrow \bullet_2, \mathrm{Set}(+(v_1,v_2)))) \\ \vee (\exists v_3, e.\mathrm{inc}(e,v_1,v_3) \wedge +(v_3,v_2)) \wedge \mathrm{SOset}(\bullet_1 \longrightarrow \bullet_3 \xrightarrow{+} \bullet_2, \mathrm{Set}(+(v_1,v_2)))\end{aligned}$$

Proof. From the semantics of HR* conditions, it is clear that for $p\colon \emptyset \to G$,
$G \models_{\mathscr{A}} \langle \exists(\emptyset \to^a S), \mathscr{R}\rangle \iff p \models_{\mathscr{A}} \langle \exists(\emptyset \to^a S), \mathscr{R}\rangle \Leftrightarrow \exists \sigma, q\colon S^\sigma \hookrightarrow G.p = q \circ a$
$\Leftrightarrow \exists S^\sigma, q\colon S^\sigma \hookrightarrow G.S \Rightarrow^*_{\mathscr{R}} S^\sigma$.
We continue by induction over the length of derivations.

Base case. By the definition of derivations,
$\exists S^\sigma \in \mathscr{G}, q\colon S^\sigma \to G.S \Rightarrow_{\mathscr{R}} S^\sigma \iff \exists S^\sigma, q\colon S^\sigma \to G.\exists x/R \in \mathscr{R}.S \Rightarrow_{x/R} S^\sigma$
$\Leftrightarrow \exists S^\sigma, q\colon S^\sigma \to G.\exists y \in \mathrm{Y}_S.\mathrm{ly}(y) = x \wedge S^\sigma \cong S_{\overline{y}} \cup R \wedge \forall i \in [k].\mathrm{pin}_{Ri} = \mathrm{att}_S(y)_i$
By Lemma 3, we can reduce this to
$\Leftrightarrow \exists y \in \mathrm{Y}_S.\mathrm{ly}(y) = x \wedge G \models \mathrm{SOgra}(S_{\overline{y}} \cup R_{\overline{\mathrm{Pin}(R)}}, \bigwedge_{i\in[k]} \mathrm{pin}_{Ri} \doteq \mathrm{att}_S(y)_i)$.
Since $k \geq 1$ and $v_i = \mathrm{pin}_{R_i}$ for $i \in [k]$, we include the formula for $\mathrm{SOgra}(y)$:
$\Leftrightarrow G \models \mathrm{SOgra}(S) \wedge \forall_{i\in[k]} v_i.x(v_1,\ldots,v_k) \Rightarrow \mathrm{SOgra}(R_{\overline{\mathrm{Pin}(R)}}, \bigwedge_{i\in[k]} v_i \doteq \mathrm{att}_S(y)_i))$.
and by the definition of SOsys, we get
$\Leftrightarrow G \models \mathrm{SOgra}(S) \wedge \forall_{i\in[k]} v_i.\mathrm{SOrule}(x/R)$.
Since S has only a single hyperedge, $x'(v_1,\ldots,v_{\mathrm{rank}\,x'})$ is false for every $x' \neq x$,
$\Leftrightarrow G \models \mathrm{SOgra}(S) \wedge \mathrm{SOsys}(\mathscr{R})$.

Induction hypothesis. For some $S' \in \mathscr{G}_{\mathscr{X}}$ with $S \Rightarrow_{\mathscr{R}} S'$, assume
$\exists S', q'\colon S' \to G.S' \Rightarrow^*_{\mathscr{R}} S^\sigma \iff G \models \mathrm{SOgra}(S') \wedge \mathrm{SOsys}(\mathscr{R})$.

Induction step. Then
$\exists S^\sigma, q\colon S^\sigma \to G.S \Rightarrow^*_{\mathscr{R}} S^\sigma \iff \exists S^\sigma, q\colon S^\sigma \to G.\exists S'.S \Rightarrow_{\mathscr{R}} S' \Rightarrow^*_{\mathscr{R}} S^\sigma$.
By Lemma 3, we can express S' as a SO formula
$\Leftrightarrow \exists S'.G \models \mathrm{SOgra}(S') \wedge \bigwedge_{x\in\mathscr{X}} \bigvee_{x/R\in\mathscr{R}} \forall v_i.\mathrm{SOrule}(x/R)$
$\Leftrightarrow \exists S'.G \models \mathrm{SOgra}(S') \wedge \mathrm{SOsys}(\mathscr{R}) \wedge S \Rightarrow_{\mathscr{R}} S' \Leftrightarrow G \models \mathrm{SOgra}(S) \wedge \mathrm{SOsys}(\mathscr{R})$.
This completes the inductive proof. □

In order to translate HR* conditions of the form $\exists(P \sqsupseteq C, c)$, we need sets of every object in P^σ and C^σ, i.e. *after* the replacement of the hyperedges. This is the role of transformation $\mathrm{SOset}(R,X)$, which ensures that every node and edge in R^σ (after replacement) is member of the set variable X in the SO formula.

The construction begins by stating that all nodes and edges of graph R are in set X. Then, it is stated that every node or edge that is part of the set $\mathrm{Set}(\mathrm{ly}_R(y)(u_1,\ldots,u_{\mathrm{rank}(y)}))$ of elements generated by a hyperedge y in R, is also element in set X. Iteratively, this ensures that set X contains every node and edge in R^σ.

Construction. For any graph R and unary variable X,

$$\mathrm{SOset}(R,X) = \quad \bigwedge_{o\in \mathrm{D}_R} o \in X \wedge \bigwedge_{y\in \mathrm{Y}_R} \forall v_1,\ldots,v_k.\forall o.o \in \mathrm{Set}(x(v_1,\ldots,v_k)) \Rightarrow o \in X$$

where $x = \mathrm{ly}_G(y)$ is the label and $k = \mathrm{rank}(y)$ the rank of hyperedge y.

Example 10 *For the left-hand graph R from Example 1, SOset yields the formula below. The v_i and u_i are nodes, while the e_i are edges, and u is the hyperedge.*

$$\begin{aligned}\mathrm{SOset}(R,X) = \quad & v_1,\ldots,v_5,e_1,\ldots,e_7 \in X \wedge u(v_5,v_1,v_3,v_4) \\ & \wedge \forall v'_1,\ldots,v'_4,o.o \in \mathrm{Set}(u(v'_1,\ldots,v'_4)) \Rightarrow o \in X\end{aligned}$$

We now have all parts ready to construct a full example of transforming a HR* condition into an SO formula with construction SO, and to prove that SO yields equivalent formulas for $\mathscr{A}$-satisfiable HR* conditions, as stated in Theorem 2: For every HR* condition c and for all graphs $G \in \mathscr{G}$, $G \models_{\mathscr{A}} c \iff G \models \mathrm{SO}(c)$.

Example 11 *We convert the* HR* *condition from Example 3 into an equivalent SO formula. To improve readability of the SO formula, the hyperedge label + is replaced by X.*

$$
\begin{aligned}
&\mathrm{SO}\left(\left\langle \exists(\bullet_1 \xrightarrow{X} \bullet_2), \bullet_1 \xrightarrow{X} \bullet_2 ::= \bullet_1 \longrightarrow \bullet_2 \mid \bullet_1 \longrightarrow \bullet \xrightarrow{X} \bullet_2 \right\rangle\right) \\
&\equiv \mathrm{SOsys}(\bullet_1 \xrightarrow{X} \bullet_2 ::= \bullet_1 \longrightarrow \bullet_2 \mid \bullet_1 \longrightarrow \bullet \xrightarrow{X} \bullet_2) \\
&\quad \wedge \mathrm{SO}(\exists(\bullet_1 \xrightarrow{X} \bullet_2)) \\
&\equiv \exists X.\forall v_1, v_2.(X(v_1,v_2) \Rightarrow \mathrm{SOgra}(\bullet_1 \longrightarrow \bullet_2) \vee \mathrm{SOgra}(\bullet_1 \longrightarrow \bullet \xrightarrow{X} \bullet_2)) \\
&\quad \wedge \mathrm{SOgra}(\bullet_1 \xrightarrow{X} \bullet_2) \\
&\equiv \exists X.\forall v_1, v_2.(X(v_1,v_2) \Rightarrow \exists e.\,\mathrm{inc}(e,v_1,v_2) \vee \exists v_3, e.\,\mathrm{inc}(e,v_1,v_3) \wedge X(v_3,v_2)) \\
&\quad \wedge \mathrm{SOgra}(\bullet_1 \xrightarrow{X} \bullet_2) \\
&\equiv \exists X.\forall v_1, v_2.X(v_1,v_2) \Rightarrow \exists e.\,\mathrm{inc}(e,v_1,v_2) \vee \exists v_3, e.\,\mathrm{inc}(e,v_1,v_3) \wedge X(v_3,v_2) \\
&\quad \wedge \exists v_1, v_2.X(v_1,v_2)
\end{aligned}
$$

The resulting formula expresses "There is a relation X such that for every pair v_1, v_2 in relation X, there is either an edge from v_1 to v_2 or an edge from v_1 to some node v_3, which is in turn in relation X with v_2 (i.e. there is a path of arbitrary length from v_1 to v_2); and the graph has two nodes v_1, v_2 in relation X." This is equivalent to "There is a path between two nodes v_1, v_2, as was expressed by the HR* *condition in Example 3.*

Proof of Theorem 2. We proceed by induction over the structure of HR* conditions. The proofs for conditions `true`, $\neg c$ and $\bigwedge_{i\in I} c_i$ are straightforward. For conditions $\exists(a,c)$, we use the Lemmata 3 and 4 to show that graph morphisms and substitution can be simulated by our construction. For conditions $\exists(P \sqsupseteq C, c)$, Lemma 3 is used to show that the inclusion of C^σ in P^σ is simulated by the constructed formula.

Base case. $c = \texttt{true}$. Then $\mathrm{SO}(c) = \texttt{true} \Rightarrow G \models_{\mathscr{A}} c \Leftrightarrow \texttt{true} \Leftrightarrow \mathrm{SO}(c) \models \texttt{true}$.

Induction hypothesis. Assume that for HR* conditions $c_i, i \in \mathbb{N}$, the theorem holds: $G \models_{\mathscr{A}} c_i \Leftrightarrow G \models \mathrm{SO}(c_i)$.

Induction step.

1. $c = \exists(a, c_1)$ for some $a = P \hookrightarrow C$.
   By the definition of HR* conditions and the induction hypothesis, we have
   $G \models_{\mathscr{A}} \exists(a,c_1) \Leftrightarrow \exists \sigma, p\colon P \hookrightarrow G, q\colon C^\sigma \to G.q \circ a^\sigma = p \wedge q \models_{\mathscr{A},\sigma} c_1$
   Using constructions SOgra and then SO yields
   $\Leftrightarrow G \models \mathrm{SOgra}(C-P) \wedge \mathrm{SOsys}(\mathscr{R}) \wedge \mathrm{SO}(c_1)$
   ($C-P$ denotes graph C without graph P and has no dangling edges, since C adds only a single object to P.)
   $\Leftrightarrow G \models \mathrm{SO}(\exists(a,c_1))$.

2. $c = \exists(P \sqsupseteq C, c_1)$.
   By the definition of HR* conditions and the induction hypothesis, we have

$G \models_{\mathscr{A}} \exists(P \sqsupseteq C, c_1)$
$\Leftrightarrow \exists p\colon P \to G, \sigma, b\colon C^\sigma \to P^\sigma, q\colon C^\sigma \to G.p \circ b = q \wedge q \models_{\mathscr{A},\sigma} c_1$
$\Leftrightarrow \exists \sigma.P^\sigma \supseteq C^\sigma \wedge C \models_{\mathscr{A},\sigma} c_1$
$\Leftrightarrow \exists \sigma.P^\sigma \supseteq C^\sigma \wedge \mathrm{SO}(c_1)$
We can now use the constructions SOgra and then SO:
$\Leftrightarrow G \models \mathrm{SOgra}(C, \exists X_P, X_C.\bigwedge_{x \in \mathrm{D}_P}(x \in X_P) \wedge \bigwedge_{y \in \mathrm{D}_C}(y \in X_C) \wedge X_C \subseteq X_P \wedge \mathrm{SO}(c_1))$
$\Leftrightarrow G \models \mathrm{SOgra}(C, \exists X_P, X_C.\mathrm{SOset}(P, X_P) \wedge \mathrm{SOset}(C, X_C) \wedge X_C \subseteq X_P \wedge \mathrm{SO}(c_1))$
$\Leftrightarrow G \models \mathrm{SO}(\exists(P \sqsupseteq C, c_1))$

3. For $c = \neg c_1$, $\mathrm{SO}(c) = \neg \mathrm{SO}(c_1)$. By the induction hypothesis, we have
$G \models_{\mathscr{A}} c \Leftrightarrow G \not\models_{\mathscr{A}} c_1 \Leftrightarrow G \not\models \mathrm{SO}(c_1) \Leftrightarrow G \models \mathrm{SO}(c)$.
For $c = \bigwedge_{i \in J} c_j$, $\mathrm{SO}(c) = \mathrm{SO}(\bigwedge_{i \in J} c_j)$.
Using the induction hypothesis, we get:
$G \models_{\mathscr{A}} \bigwedge_{i \in J} c_j \Leftrightarrow G \models \bigwedge_{i \in J} \mathrm{SO}(c_j) \Leftrightarrow G \models \mathrm{SO}(\bigwedge_{i \in J} c_j)$.

This completes the inductive proof.

It follows that every ($\mathscr{A}$-satisfiable) HR* condition can be transformed into an equivalent second-order formula. Since, by Lemma 1, every HR* condition can be transformed into an $\mathscr{A}$-satisfiable one with replacement, this is also true for HR* conditions. This concludes the proof of Theorem 2. □

7 Conclusion

In this paper, we established a lower and an upper bound on the expressiveness of HR* conditions. The relation of HR* conditions to other formalisms is shown in Figure 6: HR* conditions extend nested conditions and are situated between node-counting monadic second-order logic and second-order logic. A rough idea how edge-counting monadic second-order formulas could be represented as HR* conditions is given.

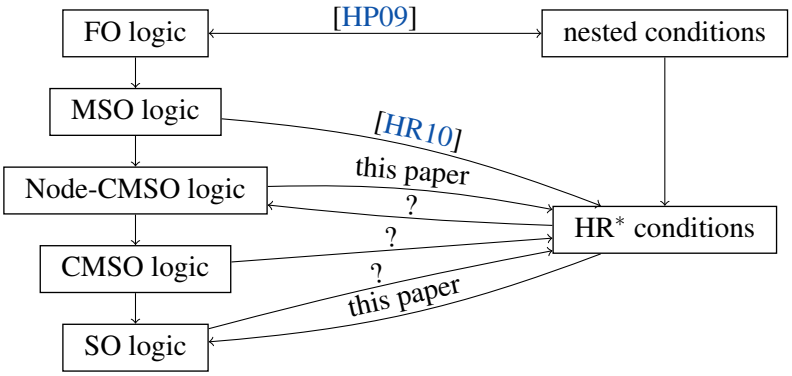


Figure 6: Comparison of the expressiveness of several types of logics and conditions.

As a side result, some variants of HR* conditions satisfaction definitions were compared. Instead of substitution of hyperedges, one may also define HR* conditions with replacement of

hyperedges, without loss or gain of expressive power. Also, ($\mathscr{A}$-)satisfaction with arbitrary morphisms is at least as powerful as satisfaction with injective morphisms (as in the initial definition).

Several questions regarding the expressiveness of HR* conditions remain open, as indicated by question marks in Figure 6. It is still not fully clear how HR* conditions relate to counting monadic second-order formulas, which may count over nodes *and* edges. Furthermore, the question remains open whether any second-order formula can be expressed as a HR* condition. The author suspects that this is not the case, as quantification over arbitrary relations seems to be more powerful than the hyperedge replacement used in HR* conditions.

Bibliography

[BBFK13] C. Blume, H. J. S. Bruggink, M. Friedrich, B. König. Treewidth, Pathwidth and Cospan Decompositions with Applications to Graph-Accepting Tree Automata. *Journal of Visual Languages & Computing*, 2013. To appear.

[BCKL06] P. Baldan, A. Corradini, B. König, A. Lluch-Lafuente. A Temporal Graph Logic for Verification of Graph Transformation Systems. In *WADT 2006*. Pp. 1–20. 2006.

[BK10] H. S. Bruggink, B. König. A Logic on Subobjects and Recognizability. In *IFIP Conference on Theoretical Computer Science*. Pp. 197–212. 2010.

[CE12] B. Courcelle, J. Engelfriet. *Graph structure and monadic second-order logic. A language theoretic approach*. Cambridge University Press, 2012.

[Cou96] B. Courcelle. On the Expression of Graph Properties in some Fragments of Monadic Second-Order Logic. In *Descriptive Complexity and Finite Models*. Pp. 33–62. 1996.

[CW05] B. Courcelle, P. Weil. The recognizability of sets of graphs is a robust property. *Theoretical Computer Science* 342(2):173–228, 2005.

[vD04] D. van Dalen. *Logic and Structure*. Springer-Verlag Berlin, 4th edition, 2004.

[EEPT06] H. Ehrig, K. Ehrig, U. Prange, G. Taentzer. *Fundamentals of Algebraic Graph Transformation*. EATCS Monographs of Theoretical Computer Science. Springer, 2006.

[Gai82] H. Gaifman. On Local and Non-Local Properties. In Stern (ed.), *Proceedings of the Herbrand Symposium: Logic Colloquium'81*. Pp. 105–135. North Holland Pub. Co., 1982.

[Hab92] A. Habel. *Hyperedge replacement: grammars and languages*. Lecture Notes in Computer Science 643. Springer, 1992.

[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.

[HR10] A. Habel, H. Radke. Expressiveness of Graph Conditions with Variables. *Electronic Communications of the EASST* 30, 2010.

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

[PP12] C. M. Poskitt, D. Plump. Hoare-Style Verification of Graph Programs. *Fundamenta Informaticae* 118(1-2):135–175, 2012.

[Ren08] A. Rensink. Explicit State Model Checking for Graph Grammars. In *Concurrency, Graphs and Models*. LNCS 5065, pp. 114–132. 2008.

A Proof: Expressiveness of $\mathscr{A}$-satisfiability

In this section, we prove Lemma 1 from Section 3:

For every HR* condition c, there is a HR* condition $\mathrm{Cond}_{\mathscr{A}}(c)$ such that for every graph G,

$$G \models c \iff G \models_{\mathscr{A}} \mathrm{Cond}_{\mathscr{A}}(c).$$

The construction of transformation $\mathrm{Cond}_{\mathscr{A}}$ is quite straightforward: To any HR* condition of the form $\exists(P \hookrightarrow C, c)$, a subcondition noId is added that forbids the identification of nodes and edges in C.

Construction. For a condition over P, $\mathrm{Cond}_{\mathscr{A}}$ is inductively defined:

(1) $\mathrm{Cond}_{\mathscr{A}}(\texttt{true}) = \texttt{true}$.

(2) $\mathrm{Cond}_{\mathscr{A}}(\exists(P \hookrightarrow C, c)) = \exists(P \to C, \mathrm{Cond}_{\mathscr{A}}(c) \wedge \mathrm{noId})$ where
$\mathrm{noId} := \forall(C \sqsupseteq \bullet, \nexists(\bullet \to \bullet)) \wedge \forall(C \sqsupseteq \circlearrowleft\!\bullet, \nexists(\circlearrowleft\!\bullet \to \downarrow\!\bullet))$.

(2') $\mathrm{Cond}_{\mathscr{A}}(\exists(P \sqsupseteq C, c)) = \exists(P \sqsupseteq C, \mathrm{Cond}_{\mathscr{A}}(c))$.

(3) $\mathrm{Cond}_{\mathscr{A}}(\neg c) = \neg \mathrm{Cond}_{\mathscr{A}}(c)$ and $\mathrm{Cond}_{\mathscr{A}}(\bigwedge_{i \in I} c_i) = \bigwedge_{i \in I} \mathrm{Cond}_{\mathscr{A}}(c_i)$.

Example 12 *The* HR* *condition* $\texttt{even} = \exists(\boxed{2}, \nexists(\boxed{2}\ \bullet))$ *with* $\boxed{2} ::= \emptyset \mid \boxed{2}\,\bullet\!\bullet$ *expresses the property "the graph has an even number of nodes" with satisfaction. With* $\mathscr{A}$*-satisfaction, the same condition would express "the graph has any number of nodes (including zero)", i.e. be equivalent to* `true`.

Using the construction of $\mathrm{Cond}_{\mathscr{A}}$*, we get*

$$\mathrm{Cond}_{\mathscr{A}}(\exists(\boxed{2}, \nexists(\boxed{2}\ \bullet))) = \exists(\boxed{2}, \forall(\boxed{2} \sqsupseteq \bullet, \nexists(\bullet \to \bullet)) \wedge \forall(\boxed{2} \sqsupseteq \circlearrowleft\!\bullet, \nexists(\circlearrowleft\!\bullet \to \downarrow\!\bullet))$$
$$\wedge \nexists(\boxed{2}\ \bullet, \forall(\boxed{2}\ \bullet \sqsupseteq \bullet, \nexists(\bullet \to \bullet)) \wedge \forall(\boxed{2}\ \bullet \sqsupseteq \circlearrowleft\!\bullet, \nexists(\circlearrowleft\!\bullet \to \downarrow\!\bullet))))$$

Proof. For conditions `true`, $\exists(P \sqsupseteq C, c)$, $\neg c$ and $\bigwedge_{i \in I} c_i$, the proof is trivial as $\mathrm{Cond}_{\mathscr{A}}$ does not change the condition and just "passes on" the construction. For a condition $\exists(P \hookrightarrow C, c)$ and graph G, we can directly transform the statement that two objects d, d' must be injective into a condition that fits our construction:

By Definition 7, we have $G \models \exists(P \hookrightarrow C, c) \iff \exists \sigma, q\colon C^\sigma \hookrightarrow G.p = q \circ a^\sigma \wedge q \models c^\sigma$.
$\Leftrightarrow \exists \sigma, q\colon C^\sigma \to G.p = q \circ a^\sigma \wedge q \models c^\sigma \wedge \nexists d, d' \in \mathrm{D}_C.d \neq d' \wedge q(d) = q(d')$.

Since q is injective, this is equivalent to
$\Leftrightarrow \exists \sigma, q\colon C^\sigma \to G.p = q \circ a^\sigma \wedge q \models c^\sigma$
$\quad \wedge \nexists v, v' \in \mathrm{V}_C.v \neq v' \wedge q(v) = q(v') \wedge \nexists e, e' \in \mathrm{E}_C.e \neq e' \wedge q(e) = q(e')$
By the semantics of HR* conditions, this yields
$\Leftrightarrow \exists \sigma, q\colon C^\sigma \to G.p = q \circ a^\sigma \wedge q \models c^\sigma$
$\quad \wedge q \models \forall(C \sqsupseteq$  $, \nexists($  $)) \wedge \forall(C \sqsupseteq$  $, \nexists($  $))$,
which, by the definition of noId, equals
$\Leftrightarrow \exists \sigma, q\colon C^\sigma \to G.p = q \circ a^\sigma \wedge q \models c^\sigma \wedge q \models \mathrm{noId}$.
By the definitions of $\mathrm{Cond}_{\mathscr{A}}$ and $\mathscr{A}$-satisfiability, this yields
$\Leftrightarrow G \models_{\mathscr{A}} \mathrm{Cond}_{\mathscr{A}}(\exists(P \hookrightarrow C, c))$. □

B Proof: replacement vs. substitution

In this section, we prove Lemma 2 from Section 3:

For every HR* condition $\langle c, \mathscr{R} \rangle$, there is a HR* condition $\langle c', \mathscr{R}' \rangle$ such that for all graphs G,

$$\exists \sigma \in \Sigma_{\mathscr{R}}.G \models_{\mathscr{A},\sigma} \iff \exists r \in \Sigma_{\mathscr{R}'}.G \models_{\mathscr{A},\mathscr{R}}$$

To simulate substitution with replacement we use the following construction idea: For every HR* condition containing several hyperedges with the same label (e.g. $\exists(\boxed{X}\ \boxed{X})$), we add a subcondition with both hyperedges combined into a "big" hyperedge, attached to all attachment points the two seperate hyperedges were attached to. The replacement system is supplemented with rules for the "big" hyperedge performing the replacement steps of both seperate hyperedges simultaneously.

Construction. Without loss of generality, assume that in the HR* conditions, variables are introduced one at a time, i.e. for each subcondition $\exists(P \to C, c)$, C is obtained from P by adding exactly one node, edge or hyperedge. For a graph G, let $\mathrm{clone}(G,n)$ be an n-fold copy of G:

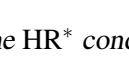
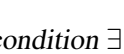
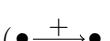
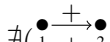
$$\begin{aligned}\mathrm{clone}(G,n) := \langle \{(k,v) \mid k \in [n], v \in \mathrm{V}_G\}, \{(k,e) \mid k \in [n], e \in \mathrm{E}_G\}, \{(k,y) \mid k \in [n], y \in \mathrm{Y}_G\},\\ \mathrm{s}_{G'}, \mathrm{t}_{G'}, \mathrm{att}_{G'}, \mathrm{lv}_{G'}, \mathrm{le}_{G'}, \mathrm{ly}_{G'} \rangle \text{ with } \mathrm{s}_{G'}((k,e)) = (k, \mathrm{s}_G(e))\end{aligned}$$

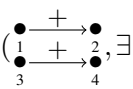
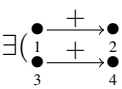
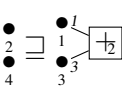
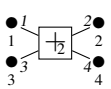
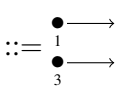
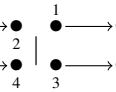
and analogous for the other mappings. For a variable $x \in \mathscr{X}$, let $x2$ be a variable with $\mathrm{rank}(x2) = n \cdot \mathrm{rank}(x)$. For improved readability, the tentacles and pinpoints of hyperedges are hidden in the following construction.
Let $\mathrm{Sub2Rep}(\exists(P + \boxed{x} \to P + \boxed{x}\ \boxed{x}, c)) =$
$\exists(P + \boxed{x} \to P + \boxed{x}\ \boxed{x}, \exists(P + \boxed{x}\ \boxed{x} \sqsupseteq P + \boxed{x2} \wedge c))$,
where $\mathrm{rank}(x2) = \mathrm{rank}(x)$ and $\mathrm{att}(\boxed{x2}) = \mathrm{att}(\boxed{x}) \circ \mathrm{att}(\boxed{x})$, and modify the replacement system $\mathscr{R}$ as follows:

$$\mathscr{R}' = \mathscr{R} + \{x2/\mathrm{clone}(R,2) \mid x/R \in \mathscr{R}\}$$

For every other form of HR* condition, Sub2Rep is straightforward: $\mathrm{Sub2Rep}(\mathtt{true}) = \mathtt{true}$, $\mathrm{Sub2Rep}(\exists(P \to C, c)) = \exists(P \to C, \mathrm{Sub2Rep}(c))$, $\mathrm{Sub2Rep}(\exists(P \sqsupseteq C, c)) = \exists(P \sqsupseteq C, \mathrm{Sub2Rep}(c))$, $\mathrm{Sub2Rep}(\bigwedge_{i \in I} c_i) = \bigwedge_{i \in I} \mathrm{Sub2Rep}(c_i)$, and $\mathrm{Sub2Rep}(\neg c) = \neg\, \mathrm{Sub2Rep}(c)$.

Example 13 (substitution simulated by replacement)

The HR* *condition* $\exists($$,\nexists($  $)$ *with*  $::=$  $\mid$  *) is satisfied by any graph which has a path between two nodes* 1 *and* 2*, but no second, disjoint path of the same length, since both "+" hyperedges are substituted by a path of the same length. Using replacement instead of substitution, this is equivalent to the* HR* *condition*

$\exists($$,\nexists($  $,\exists($  $\sqsupseteq$  $)))$ *with*  $::=$  $\mid$  .

With replacement, the substitution of the two hyperedges by isomorphic graphs is simulated by combining both hyperedges into a single one, where two isomorphic graphs are generated in parallel.

Proof. By structural induction over HR* conditions. Assume that the proposition holds for c, c_i.
Let $p\colon \emptyset \to G$. $p \models_{\mathscr{A},\mathscr{R}} \text{Sub2Rep}(\exists(P+\boxed{x} \to^a P+\boxed{x}\,\boxed{x}, c))$
By the definition of Sub2Rep and $\mathscr{A}$-satisfaction,
$\Leftrightarrow p \models_{\mathscr{A},\mathscr{R}} \exists(P+\boxed{x} \to^a P+\boxed{x}\,\boxed{x}, \exists(P+\boxed{x}\,\boxed{x} \sqsupseteq P+\boxed{x2} \wedge c))$
$\Leftrightarrow \exists q\colon (P+\boxed{x}\,\boxed{x})^{\mathscr{R}} \to G.p = q \circ a^{\mathscr{R}} \wedge q \models_{\mathscr{A},\mathscr{R}} \exists(P+\boxed{x}\,\boxed{x} \sqsupseteq P+\boxed{x2} \wedge c)$
$\Leftrightarrow \exists q\colon (P+\boxed{x}\,\boxed{x})^{\mathscr{R}} \to G.p = q \circ a^{\mathscr{R}} \wedge$
$\quad \exists q'\colon (P+\boxed{x2})^{\mathscr{R}} \to G.(P+\boxed{x2})^{\mathscr{R}} \subseteq (P+\boxed{x}\,\boxed{x})^{\mathscr{R}} \wedge q = q'_{|(P+\boxed{x}\,\boxed{x})^{\mathscr{R}}} \wedge q' \models_{\mathscr{A},\mathscr{R}} c$
Since $P+\boxed{x2}$ contains only one non-substituted variable, and by the hypothesis,
$\Leftrightarrow \exists q\colon (P+\boxed{x}\,\boxed{x})^{\mathscr{R}} \to G.p = q \circ a^{\mathscr{R}} \wedge$
$\quad \exists \tau, q'\colon (P+\boxed{x2})^{\tau} \to G.(\boxed{x2})^{\tau} \subseteq (P+\boxed{x}\,\boxed{x})^{\tau} \wedge q = q'_{|(P+\boxed{x}\,\boxed{x})^{\tau}} \wedge q' \models_{\mathscr{A},\tau} c.$
Since the pinpoints in $(\boxed{x2})^{\tau}$ and $(P+\boxed{x}\,\boxed{x})^{\tau}$ are identical,
$\Leftrightarrow (\boxed{x2})^{\tau} \subseteq (P+\boxed{x}\,\boxed{x})^{\tau} \Leftrightarrow (\boxed{x2})^{\tau} \cong (P+\boxed{x}\,\boxed{x})^{\tau}$
and because the rules for $\boxed{x2}$ generate two identical subgraphs,
$\Leftrightarrow \exists \sigma \in \Sigma_{\mathscr{R}}, q\colon (P+\boxed{x}\,\boxed{x})^{\sigma} \to G.p = q \circ a^{\sigma} \wedge$
$\quad \exists \tau, q'\colon (P+\boxed{x2})^{\tau} \to G.(\boxed{x2})^{\tau} \cong (P+\boxed{x}\,\boxed{x})^{\tau} \wedge q = q'_{|(P+\boxed{x}\,\boxed{x})^{\tau}} \wedge q' \models_{\mathscr{A},\tau} c.$
By the satisfaction of HR* conditions,
$\Leftrightarrow \exists \sigma, q\colon (P+\boxed{x}\,\boxed{x})^{\sigma} \to G.p = q \circ b \wedge q \models_{\mathscr{A},\tau} c$
$\Leftrightarrow p \models_{\mathscr{A},\sigma} \exists(P+\boxed{x} \to P+\boxed{x}\,\boxed{x}, c).$
For all other forms of HR* conditions, the proof is trivial. □