Spatial Existential Positive Logics for Hyperedge Replacement Grammars

Yoshiki Nakamura
Tokyo Institute of Technology, Tokyo, Japan

Abstract

We study a (first-order) spatial logic based on graphs of conjunctive queries for expressing (hyper-)graph languages. In this logic, each primitive positive (resp. existential positive) formula plays a role of an expression of a graph (resp. a finite language of graphs) modulo graph isomorphism. First, this paper presents a sound- and complete axiomatization for the equational theory of primitive/existential positive formulas under this spatial semantics. Second, we show Kleene theorems between this logic and hyperedge replacement grammars (HRGs), namely that over graphs, the class of existential positive first-order (resp. least fixpoint, transitive closure) formulas has the same expressive power as that of non-recursive (resp. all, linear) HRGs.

1 Introduction

Existential positive (EP) formulas are first-order formulas that are built up from atomic predicates, equality (=), top (tt), bottom (ff), conjunction (\(\land\)), disjunction (\(\lor\)), and existential quantifier (\(\exists\)). In particular, primitive positive (PP) formulas are EP formulas without \(ff\) nor \(\lor\). PP formulas are semantically equivalent to conjunctive queries [1], which are at the core of query languages in database theory. In this paper, we focus on the (hyper-)graphs of conjunctive queries (a.k.a. natural models of conjunctive queries) [11][12, Fig. 1], which were introduced to characterize the semantical equivalence of conjunctive queries [11, Lemma 13][28] as follows: two PP formulas are semantically equivalent if and only if their graphs are homomorphically equivalent. For example, the graph of the PP formula \(\exists z.a(x, z) \land a(z, y) \land b(x, z, y)\) is the following: \[
\begin{array}{c}
\bullet_x \rightarrow \bullet_1 \rightarrow \bullet_2 \rightarrow \bullet_y \\
\bullet_z \rightarrow \bullet_1 \rightarrow \bullet_2 \rightarrow \bullet_y
\end{array}
\]
This characterization can be generalized to EP formulas by using finite sets of graphs (see, e.g., [39, Sect. 2.6]).

In this paper, turning our attention to the correspondence between primitive positive logics and (hyper-)graphs, we study PP/EP formulas as graph/graph-language expressions. To this end, we introduce a spatial semantics (like that of graph logic [10] or separation logic [34, 37]), which is based on graphs of conjunctive queries, called GI-semantics. The semantics enables us to study graphs and graph languages through logical formulas in a natural way. The remarkable difference from classical semantics is the following (cf. the above): two PP formulas are equivalent under GI-semantics if and only if their graphs are (graph-)isomorphically equivalent. While the equational theory of PP/EP formulas under GI-semantics is subclassical, some formula transformations under classical semantics, in logic and database theory, still work under GI-semantics.

Our first contribution is to present a sound- and complete axiomatization of the equational theory of PP/EP formulas under GI-semantics. Furthermore, we extend EP with the least-fixpoint operator and the transitive closure operator (see, e.g., [20, Sect. 8]), denoted by
EP(LFP) and EP(TC), respectively. They can express possibly infinite graph languages. Our second contribution is to show that each of the logics above has the same expressive power as some class of hyperedge replacement grammar (HRG) [25, 35] (see also [19]), which is a generalization of context-free word grammar from words to graphs, as follows.

**Theorem 1.** Under GI-semantics, for every graph language \( \mathcal{G} \) (closed under isomorphism):

1. Some EP formula recognizes \( \mathcal{G} \) iff some non-recursive HRG recognizes \( \mathcal{G} \) (i.e., \( \mathcal{G} \) is finite up to isomorphism). In particular, some PP formula recognizes \( \mathcal{G} \) iff some deterministic and non-recursive HRG recognizes \( \mathcal{G} \) (i.e., \( \mathcal{G} \) is a singleton up to isomorphism).

2. Some EP(LFP) formula recognizes \( \mathcal{G} \) iff some HRG recognizes \( \mathcal{G} \).

3. Some EP(TC) formula recognizes \( \mathcal{G} \) iff some linear HRG recognizes \( \mathcal{G} \).

This theorem is an analogy of Kleene theorem [27], that over words, for every language \( L \): some regular word grammar (or equivalently, non-deterministic finite automaton) recognizes \( L \) if and only if some regular expression recognizes \( L \). Such an equivalence between expressions and grammars/automata like Kleene theorem has also been widely studied for many other language classes (e.g., context-free word languages [29], \( \omega \)-regular word languages [31], regular tree languages [13, Theorem 2.2.8], language classes over some specific graph classes [30, 6, 5]).

To our knowledge, the Kleene theorem for HRGs and linear HRGs (namely, some syntax having the same expressive power) has not yet been investigated, whereas logical or algebraic characterizations are known, e.g., [3, 15].

**Related work.** This paper uses PP formulas as graph expressions and uses EP(LFP) formulas as graph language expressions. There also are some expressions for (bounded treewidth) graphs (or relational structures), e.g., HR-algebra [3, 16], SP-terms [33], 2p-algebra [14, 18], graphical (string diagrammatic) conjunctive queries [4]. As for the completeness result of PP (Theorem 19), Bauderon and Courcelle [3] have already presented a syntax and a complete axiomatization for graphs modulo isomorphism. However, our completeness proof (essentially [3] also) would have a sufficiently simple strategy relying on the transformation for obtaining conjunctive-queries from primitive positive formulas (under classical semantics); this is a reason that our expressions are based on logical formulas.

As for characterizing language classes by classical logics, it dates back to Büchi-Elgot-Trakhtenbrot Theorem [8, 9, 21, 42] (see also [23]), which states that over words, monadic second-order logic has the same expressive power as the class of regular expressions. See [16, Theorem 7.51] [15] for a logical characterization of HRGs, by using monadic second-order logic as a graph transducer. However, the characterization presented in this paper uses logical formulas as graph-language expressions.

Also, the number of variables in formulas has a deep connection with the treewidth [38, 26] of (hyper-)graphs (or relational structures), which is a parameter indicating how much a graph is similar to a tree. It was mentioned in [28, Remark 5.3] that under the classical semantics, for every relational structure of treewidth \( k \), its conjunctive query is semantically equivalent to an PP\(^{(k+1)(0)}\) formula. Here, PP\(^{k(l)}\) denotes the set of PP formulas using at most \( k \) variables and at most \( l \) free variables. In particular, it is shown in [32] that under the classical semantics, PP\(^{3(2)}\) has the same expressive power as the primitive positive calculus of relations, which is a fragment of Tarski’s calculus of relations [40]. In [14, 18], a sound- and complete axiomatization is presented for 2p-algebra, which is intuitively the primitive positive calculus of relations under GI-semantics. In connection with them, it would be interesting to present a sound- and complete axiomatization of the equational theory of PP\(^{k(l)}\) formulas under GI-semantics, but it still remains open.
Outline. Section 2 presents preliminaries. Section 3 introduces GI-semantics. Section 4 presents an axiomatization of the equational theory under GI-semantics for PP/EP formulas. Section 5 (and 3) shows Kleene theorems between spatial existential positive logic and HRGs (Theorem 1(1)-(3)). Section 6 concludes this paper.

2 Preliminaries

We write \( \mathbb{N} \) (resp. \( \mathbb{N}_+ \)) for the set of all non-negative (resp. positive) integers. For \( l, r \in \mathbb{N} \), we write \([l, r] \) for the set \( \{ i \in \mathbb{N} \mid l \leq i \leq r \} \). In particular, we write \( [n] \) for \([1, n] \). The cardinality of a set \( A \) is denoted by \#(A). For an equivalence relation \( \sim \) on a set \( X \), the quotient set of \( X \) by \( \sim \) is denoted by \( X/\sim \) and the equivalence class of an element \( x \) w.r.t. \( \sim \) is denoted by \([x]_\sim \). For sets \( X_1 \) and \( X_2 \), the disjoint union \( X_1 \uplus X_2 \) is defined by \( \{ (i, a) \mid i \in [2], a \in X_i \} \).

We denote by \( \vec{a} = \langle a_1, \ldots, a_n \rangle \) (also denoted by \( a_1 \ldots a_n \) or \( \langle a_i \rangle_{i=1}^n \)) a finite sequence. The length \( |\vec{a}| \) of \( \vec{a} \) is \( n \). We denote by \( \text{Occ}(\vec{a}) \) the set \( \{ a_1, \ldots, a_n \} \). We say that a sequence \( \vec{a} \) is a permutation of a set \( A \) if \( \text{Occ}(\vec{a}) = A \) and the elements of \( \vec{a} \) are pairwise distinct. We denote by \( \text{Perm}(A) \) the set of all permutations of a set \( A \). We denote by \( A^* \) (resp. \( A^k \)) the set of all finite sequences (resp. sequences of length \( k \)) over a set \( A \). Also, we denote by \( \iota_n \) (or just by \( \iota \) if \( n \) is obvious) the sequence \( 1, 2, \ldots, n \). An alphabet \( A \) is a possibly infinite set. A (finite-set-)typed alphabet \( A \) is an alphabet with a function \( \text{ty}(A) \) (or written \( \text{ty} \)) for \( A \) to finite sets. In particular we say that a symbol \( a \) in \( A \) is ordinal-typed if \( \text{ty}(A)(a) = [k] \) for some \( k \in \mathbb{N} \). The arity of \( a \) in \( A \) is \( k \), denoted by \( \text{ar}^A(a) \) (or just by \( \text{ar}(a) \)).

Graphs In the following, we define graphs (with ports) and graph languages.

Definition 2 (graph). Given a typed alphabet \( A \) and a finite set \( \tau \), an \( A \)-labelled graph \( G \) of type \( \tau \) is a tuple \( (V^G, E^G, \text{lab}^G, \text{vert}^G, \text{port}^G) \), where \( V^G \) is a finite set of vertices, \( E^G \) is a finite set of (hyper-)edges, \( \text{lab}^G : E^G \rightarrow A \) is a function denoting the label of each edge, \( \text{vert}^G(c) : ty^G(c) \rightarrow V^G \) is a function denoting the vertices of each edge, and \( \text{port}^G : ty(G) \rightarrow V^G \) is a function denoting the ports of \( G \). Here, \( ty(G) \triangleq \tau \) and \( ty^G \triangleq \text{ty}^A \circ \text{lab}^G \).

Example 3. Let \( A = \{ a, b, c \} \) with type \( ty^A = \{ a \mapsto [2], b \mapsto [3], c \mapsto [2] \} \). Let \( G = \langle \{ v_1, v_2, v_3 \}, \{ e_1, e_2 \}, \{ e_1 \mapsto a, e_2 \mapsto b \}, \{ e_1 \mapsto \lambda i \in [2], v_1, e_2 \mapsto \{ 1 \mapsto v_1, 2 \mapsto v_1, 3 \mapsto v_3 \} \rangle, \lambda i \in [3], v_1 \rangle \) and let \( H = \langle \{ v_1, v_2 \}, \{ c \mapsto e \}, \{ e \mapsto \{ 1 \mapsto v_2, 2 \mapsto v_1 \} \rangle, \lambda i \in [2], v_1 \rangle \) be \( A \)-labelled graphs (of type \( [3] \) and of type \( [2] \), respectively), where \( v_1, v_2, v_3, e_1, e_2 \) are pairwise distinct. Their graphical representations are in Figure 1a and 1b, respectively.

![Figure 1](image)

Figure 1 Examples of graphs and operations on graphs.

Later (e.g., in Example 12), for binary edges and ports, we often use \( a \) to denote \( \circ \) and \( a \) to denote \( \circ \). Also, for unlabelled non-hyper graphs, let \( A_E \triangleq \{ E \} \) with \( ty^A_E = \{ E \mapsto [2] \} \) and we use \( \circ \) to denote \( \circ \).
We denote by $\text{GR}_A^\tau$ the set of all $A$-labelled graphs of type $\tau$. An $(A$-labelled) graph language $\mathcal{G}$ (of type $\tau$) is a subset of $\text{GR}_A^\tau$. Given a system $\mathcal{S}$ (e.g., HRGs, EP formulas, …) over $A$ (that defines a graph language $\mathcal{G}(E)$ for every $E$ in $\mathcal{S}$), we say that $\mathcal{G}$ is recognized by $\mathcal{S}$ if there exists some element $E$ in $\mathcal{S}$ such that $\mathcal{G} = \mathcal{G}(E)$.

**Definition 4** (homomorphism, isomorphism). Let $G, H \in \text{GR}_A^\tau$ be graphs. A pair $h = (h^V, h^E)$ of $h^V: V^G \to V^H$ and $h^E: E^G \to E^H$ is a homomorphism from $G$ to $H$ if (1) $\text{lab}^G = \text{lab}^H \circ h^E$, (2) $\text{vert}^H(h^E(e))(x) = h^V(\text{vert}^G(e)(x))$, and (3) $\text{port}^H = h^V \circ \text{port}^G$. In particular, $h$ is an isomorphism if both $h^V$ and $h^E$ are bijective. We say that $G$ and $H$ are isomorphic, written $G \cong H$ if there exists an isomorphism between $G$ and $H$.

In this paper, we will only focus on $\cong$-closed (i.e., if $G \in \mathcal{G}$ and $G \cong H$, then $H \in \mathcal{G}$) graph languages. We denote by $\mathcal{G}^{\cong}$ the minimal $\cong$-closed graph language including $\mathcal{G}$.

**Some operations on graphs.** In the following, we present some primitive operations on graphs. See Figure 1c-1g for graphical examples of Definition 5-8. In GI-semantics, * uses gluing, $\exists$ uses forgetting, LFP uses hyperedge replacing, TC uses concatenating.

**Definition 5** (glueing). Let $G_1, G_2 \in \text{GR}_A^\tau$. Let $G_1 \otimes G_2 \in \text{GR}_A^\tau$. Let $G_1 \otimes G_2 \in \text{GR}_A^\tau$ be the graph such that $V^{G_1 \otimes G_2} = (V^{G_1} \sqcup V^{G_2}) / \sim$, $E^{G_1 \otimes G_2} = E^{G_1} \sqcup E^{G_2}$, $\text{lab}^{G_1 \otimes G_2}((k, e)) = \text{lab}^{G_k}(e)$, $\text{vert}^{G_1 \otimes G_2}((k, e))(x) = \text{vert}^{G_k}(e)(x)_z$, and $\text{port}^{G_1 \otimes G_2}(x) = [\text{port}^{G_k}(x)]_z$. Here, $\sim$ is the minimal equivalence relation such that for every $x \in \tau \cap v$, $\langle 1, \text{port}^{G_1}(x) \rangle \sim (2, \text{port}^{G_2}(x))$.

**Definition 6** (labelling/forgetting/renameing). Let $G \in \text{GR}_A^\tau$. For a vertex $v \in V^G$, a variable $z \not\in \tau$, and a variable $x \in \tau$, we define the graphs $G[z := v] \in \text{GR}_A^\tau$, $G[f/x] \in \text{GR}_A^\tau$, $G[z/x] \in \text{GR}_A^\tau$ by $G[z := v] \equiv (V^G, E^G, \text{lab}^G, \text{vert}^G, \text{port}^G \cup \{z \mapsto v\}), G[f/x] \equiv (V^G, E^G, \text{lab}^G, \text{vert}^G, \text{port}^G \setminus \{x \mapsto \text{port}^G(x)\}), G[z/x] \equiv G[f/x][z := \text{port}^G(x)]$.

We write $G[y_1/\ldots/y_n/x_1/\ldots/x_n]$ for $G[z_1/x_1/\ldots/z_n/x_n][y_1/z_1/\ldots/y_n/z_n]$, where $z_1/\ldots/z_n$ is a sequence of fresh variables. For a sequence $z_1/\ldots/z_n$ of pairwise distinct variables, we write $G[z_1/\ldots/z_n := v_1/\ldots/v_n]$ for $G[z_1 := v_1/\ldots/z_n := v_n]$.

**Definition 7** (hyperedge replacing). Let $G \in \text{GR}_A^\tau$. For an edge $e \in E^G$ and a graph $H \in \text{GR}_A^\tau$, let $G[H/e] \in \text{GR}_A^\tau$ be the graph $((G \setminus e)[z := \text{vert}^G(e)(x_1)\ldots\text{vert}^G(e)(x_n)] \sqcup H[z/x_1/\ldots/x_n])[f/\ldots/f/z]$, where $G \setminus e$ denotes the graph $G$ in which the edge $e$ has been removed. Here, $x_1/\ldots/x_n \in \text{Perm}(\text{ty}(H))$, and $z$ is a sequence of fresh variables.

We write $G[H_1/\ldots/H_n/e_1/\ldots/e_n]$ for $G[H_1/e_1][H_2/\ldots/H_n/(1/e_2)\ldots(1/e_n)]$ if $n \geq 1$, and if $n = 0$.

**Definition 8** (concatenating). Let $G \in \text{GR}_A^\tau$ and $H \in \text{GR}_A^\tau$. Let $\bar{x} \in \text{ty}(H)^k$ and $\bar{y} \in \text{ty}(H)^l$ be sequences of pairwise different elements, where $k \geq 1$. Then, let $G \circ_{\bar{x}\bar{y}} H \in \text{GR}_A^\tau$ be the graph $(G[z/\bar{x}] \otimes H[z/\bar{y}])[\bar{x}/\ldots/\bar{z}]$, where $\bar{z}$ is a sequence of fresh variables.

Finally, we list some basic equations in the following.

**Proposition 9.** (1) $G_1 \otimes (G_2 \otimes G_3) \cong (G_1 \otimes G_2) \otimes G_3$; (2) $G \otimes H \cong H \otimes G$; (3) $(H_1 \otimes H_2)[G/(1/e)] \cong H_1[G/e] \otimes H_2$; (4) $G[z/x][H/e] \equiv G[H/e][z/x]$; (5) $G[z/x] \otimes H \cong (G \otimes H)[z/x]$ if $x \not\in \text{ty}(H)$. 


Hyperedge Replacement Grammars. In the following, we present the definition of hyperedge replacement grammars (HRGs).

**Definition 10** (e.g., [19]). A hyperedge replacement grammar (HRG) \( \mathcal{H} \) over a typed alphabet \( A \) is a tuple \( \langle \mathcal{X}, \mathcal{R}, \mathcal{S} \rangle \), where \( \mathcal{X} \) is a finite typed alphabet disjoint with \( A \) for (non-terminal) labels, \( \mathcal{R} \) is a finite set of pairs \( r = (X, G) \) (written \( X \Leftarrow G \)) of \( X \in \mathcal{X} \) and \( G \in \text{GR}_A^{ty}(\mathcal{X}) \) for rewriting rules, and \( \mathcal{S} \in \mathcal{X} \) denotes the source label.

We also define the graph languages of HRGs as follows.

**Definition 11** (cf. [19, Sect. 2.3.2]). For an HRG \( \mathcal{H} = \langle \mathcal{X}, \mathcal{R}, \mathcal{S} \rangle \) over a typed alphabet \( A \), the binary relation \( \vdash_{\mathcal{H}} \subseteq \bigcup_{X \in \mathcal{X}} \text{GR}_A^{ty}(X) \times \{X\} \) is defined as the least \( \subseteq \)-closed (i.e., if \( G \subseteq H \) and \( G \vdash_{\mathcal{H}} X \), then \( H \vdash_{\mathcal{H}} X \)) relation closed under the following rule: If \( X \Leftarrow G \in \mathcal{R} \), then \( H_1 \vdash_{\mathcal{H}} \text{lab}^G(e_1) \ldots H_n \vdash_{\mathcal{H}} \text{lab}^G(e_n) \). The graph language is defined by: \( \mathcal{G}(\mathcal{H}) \triangleq \{ G \in \text{GR}_A^{ty}(\mathcal{S}) \mid G \vdash_{\mathcal{H}} \mathcal{S} \} \).

For an HRG \( \mathcal{H} \), we say that \( \mathcal{H} \) is linear [35, Definition 3] if for every rule \( X \Leftarrow G \in \mathcal{R} \), the number of non-terminal labels occurring in \( G \) is at most one. We say that \( \mathcal{H} \) is \((n-) \)recursive if there exist rules \( X_0 \Leftarrow G_0, \ldots, X_n \Leftarrow G_n \in \mathcal{R} \) such that \( X_i \) occurs in \( G_{i-1} \) for \( i \in [0, n] \) where \( n \in \mathbb{N} \) and \( G_{-1} \) denotes \( G_n \).

**Example 12.** Let \( \mathcal{H}_E \) be the HRG over \( A_E \), defined by \( ty^{X,E} = \{ \mathcal{S} \mapsto \{[0], X \mapsto [2]\} \} \), \( \mathcal{R}^{X,E} = \{ (\mathcal{S}), (E), (s), (p) \} \), and \( S^{X,E} = \mathcal{S} \), where each rule in \( \mathcal{R} \) is as follows:

\[
\begin{align*}
(S) & \quad \mathcal{S} \leftarrow \mathcal{S} \\
(E) & \quad X \leftarrow \mathcal{S} \quad \mathcal{S} \leftarrow \mathcal{S} \\
(s) & \quad X \leftarrow \mathcal{S} \quad \mathcal{S} \leftarrow \mathcal{S} \\
(p) & \quad X \leftarrow \mathcal{S} \quad \mathcal{S} \leftarrow \mathcal{S}
\end{align*}
\]

Then, \( \mathcal{G}(\mathcal{H}_E) \) is the set of all (directed) series-parallel graphs [24], e.g., \( \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \) is \( \mathcal{G}(\mathcal{H}_E) \) is shown by:

\[
\begin{align*}
\quad \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} & \quad \mathcal{S} \rightarrow \mathcal{S} \\
\mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} & \quad \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \\
\mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} & \quad \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \\
\mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} & \quad \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S} \rightarrow \mathcal{S}
\end{align*}
\]

\section{Existential Positive Logics under GI-Semantics}

In this section, we introduce the syntax and a spatial semantics of our existential positive logics. Let \( A \) be an ordinal-typed alphabet, \( \mathcal{Y}_1 \) be a countably infinite set of first-order variables, and \( \mathcal{Y}_2 \) be an ordinal-typed set of second-order variables, where for every \( k \in \mathbb{N}_+ \), the number of second-order variables of arity \( k \) is countably infinite. Here, \( A \), \( \mathcal{Y}_1 \), and \( \mathcal{Y}_2 \) are disjoint. For \( \tau \subseteq \mathcal{Y}_1 \) and \( \mathcal{X} \subseteq A \cup \mathcal{Y}_2 \), we define \( \text{Fml}_A^\tau \) as the least set closed under the rules as follows.\(^1\)

\[
\begin{align*}
\top & \in \text{Fml}_A^\emptyset \\
x = y & \in \text{Fml}_A^{\{x,y\}} \\
X \bar{x} & \in \text{Fml}_A^{\text{Occ}(\bar{x})} \\
\varphi \in \text{Fml}_A^\tau & \quad \psi \in \text{Fml}_A^\tau \quad \varphi \in \text{Fml}_A^{\tau \cup \{x\}} \\
\varphi \lor \psi & \in \text{Fml}_A^\tau \quad \varphi \in \text{Fml}_A^{\text{Occ}(\bar{x})} \\
\varphi \psi & \in \text{Fml}_A^{\tau \cup \{x\}} \\
\exists x. \varphi & \in \text{Fml}_A^{\tau \cup \{x\}} \\
\exists \bar{y}. \varphi & \in \text{Fml}_A^{\tau \cup \{\bar{y}\}} \\
\exists \bar{y} \bar{u}. \varphi & \in \text{Fml}_A^{\tau \cup \{\bar{y} \bar{u}\}} \\
\& x \in \mathcal{X} \text{ and } \text{ar}^\mathcal{Y}(X) = |\bar{x}| & \quad \tau_1: \quad x \not\in \tau. \\
\text{ar}(X) = |\bar{x}| = |\bar{y}| & \geq 1. \\
\bar{x} \text{ and } \bar{y} \text{ are sequences of pairwise distinct variables.} \\
|\bar{x}| = |\bar{y}| = |\bar{u}| & \geq 1. \\
\bar{x} \bar{y} \text{ and } \bar{u} \bar{u} \text{ are sequences of pairwise distinct variables.}
\end{align*}
\]

\(^1\) We adopt the spatial conjunction symbol \( \ast \) instead of \( \land \).
We often use parentheses in ambiguous situations. We say that $\varphi$ is a formula over $A$ of type $\tau$ if $\varphi \in \text{Fml}_A^\tau$. Note that, for a technical reason, $\text{ff}$ has any type $\tau$. We use $\text{FV}_1(\varphi)/\text{FV}_2(\varphi)$ (resp. $\text{BV}_1(\varphi)/\text{BV}_2(\varphi)$) to denote the set of first-/second-order free (resp. bound) variables of $\varphi$, and use $\text{V}_1(\varphi)$ to denote the set $\text{FV}_1(\varphi) \cup \text{BV}_1(\varphi)$ for $l = 1, 2$. The set $\text{PP}_A^\tau$ (resp. $\text{EP}_A^\tau$, $\text{EP}(\text{LFP})_A^\tau$, $\text{EP}(\text{TC})_A^\tau$) is defined as the set of all $\varphi \in \text{Fml}_A^\tau$ such that $\varphi$ is generated from the rules for $\top, =, X\bar{x}, \ast$, and $\exists$. (resp. the rules for PP with $\text{ff}$ and $\lor$, the rules for EP with LFP, the rules for EP with TC). Note that some syntax restrictions exist, e.g., $\top \lor X\bar{x} \notin \text{Fml}_A^\tau$ for any $X$ and $\tau$. They are for simplifying the definition of GI-semantics.

For notational simplicity, we denote by $\bigodot_{i=1}^n \varphi_i$ (similarly for $\bigodot_{i=1}^n \varphi_i$) the formula $\bigodot_{i=1}^n \varphi_i$ if $n \geq 1$ and the formula $\top$ if $n = 0$, by $x_1 \ldots x_n = y_1 \ldots y_n$ the formula $\bigodot^1 \varphi_i = y_i$, by $\exists x_1 \ldots x_n \varphi$ the formula $\bigodot^1 \exists x_1 \exists x_2 \ldots \exists x_n \varphi$, and by $\varphi[y_1 \ldots y_n / x_1 \ldots x_n]$ the formula $\varphi$ in which each free variable $x_i$ occurring in $\varphi$ has been replaced with $y_i$, where $i \in [n]$. A formula $\varphi$ is atomic if $\varphi$ forms $\top$, $x = y$, or $X\bar{x}$. Explicitly, we may use $\varphi$ to denote an atomic formula. We use atomic formulas to denote atomic graphs as follows.

**Definition 13.** For a finite set $\tau$, let $G^n_{\top} \triangleq \{\top, \emptyset, \emptyset, \lambda x \in \tau. x\}$.

For an atomic formula $\varphi$, we define the graph $G_{\varphi}$ by:

$$
G_{\top} = \{v\}, \text{G}_{\emptyset} = \{\emptyset\}, \text{G}_{\lambda x \in \tau. x} = \{x\}.
$$

In the following, we define a spatial semantics for graph languages, called GI-semantics.

Note that for every $\varphi$, if $G \models^\text{GI} \varphi$, then $\text{ty}(G)$ is determined to $\text{FV}_1(\varphi)$.

**Definition 14 (GI-semantics).** The binary relation $\models^\text{GI} \subseteq \bigcup_{\tau \subseteq \tau_1 \subseteq \tau_2} G_{\top}^\tau \times \text{Fml}_A^\tau$ is defined as the least $\models^\text{Esc}$-closed relation closed under the rules in Figure 2.

The graph language of $\varphi$ is defined by $G(\varphi) \triangleq \{G \mid G \models^\text{GI} \varphi\}$. We say that $\varphi$ and $\psi$ are graph-isomorphically equivalent (GI-equivalent), written $\varphi \equiv^\text{GI} \psi$ if $G(\varphi) = G(\psi)$.

**Example 15.** Let $G \triangleq \{x, y\}$ and $\varphi \triangleq x = y \ast \exists z. Exz \ast Ez\bar{y}$. Then, $G \models^\text{GI} \varphi$ is shown by:

```
\begin{array}{cl}
\frac{G \models^\text{GI} \varphi \quad H \models^\text{GI} \psi}{(\ast) \quad (\text{TC}) \quad (LFP)} & \frac{G \models^\text{GI} \varphi \quad H \models^\text{GI} \varphi \ast \psi}{(\ast)}
\end{array}
```

We will generalize this example in Definition 16, for expressing any graphs by PP formulas.

---

2 See Appendix A for an alternative definition. Here, we adopt this style for extending to Definition 27.
3.1 PP/EP formulas as graph/finite-graph-language expressions

In this subsection, we show that PP (resp. EP) formulas under GI-semantics play a role as graph expressions (resp. finite graph language expressions).

**Definition 16.** Let $G$ be a graph, $\vec{x} = x_1 \ldots x_k \in \text{Perm}(\text{ty}(G))$, $\vec{v} = v_1 \ldots v_n \in \text{Perm}(V^G)$, and $\vec{e} = e_1 \ldots e_n \in \text{Perm}(E^G)$. Let $\varphi_{\vec{x},\vec{v},\vec{e}}^G$ (or written $\varphi_G$ if they are not important) be the following PP formula, where $z_{v_1}, \ldots, z_{v_n}$ are fresh variables:

$$\exists z_{v_1} \ldots z_{v_n} (\bigotimes_{i=1}^{k} z_{\text{port}^G(e_i)} = x_i) \ast (\bigotimes_{i=1}^{m} \text{lab}^G(e_i) z_{\text{vert}^G(e_i)(1)} \ast \ldots \ast z_{\text{vert}^G(e_i)(\text{arity}(e_i))})$$

Also, for a finite sequence $\vec{G} = G_1 \ldots G_n$ of graphs, let $\varphi_{\vec{G}}$ be the EP formula $\bigvee_{i=1}^{n} \varphi_{G_i}$.

Then, $\mathcal{G}(\varphi_G) = \{G\}^\mathcal{G}$ and $\mathcal{G}(\varphi_{\vec{G}}) = \text{Occ}(\vec{G})^\mathcal{G}$. By using them, the following holds.

**Proposition 17 (Theorem 1(1)).** For every graph language $\mathcal{G}$ closed under isomorphism:

(1): $\mathcal{G}$ is singleton up to isomorphism iff some PP formula recognizes $\mathcal{G}$. (2): $\mathcal{G}$ is finite up to isomorphism iff some EP formula recognizes $\mathcal{G}$.

**Proof.** (1)(2)(⇒): By using $\varphi_G$ and $\varphi_{\vec{G}}$, respectively. (1)(2)(⇐): By a straightforward induction on the structure of PP (resp. EP) formulas. □

**Remark 18.** Indeed, GI-semantics characterizes the graphs of PP formulas [11] (see also [12, Figure 1]), namely, for every PP formula $\varphi$, $G \models^{\text{GI}} \varphi$ if $G$ is isomorphic to the graph of $\varphi$. Thus, two PP formulas are GI-equivalent iff their graphs are isomorphically equivalent.

4 An Axiomatization of the Equational Theory of PP/EP

This section presents an axiomatization of the equational theory under GI-semantics (i.e., the binary relation $\equiv^{\text{GI}}$) of PP/EP formulas. Given an ordinal-typed alphabet $A$, we define the binary relation $\simeq \subseteq \bigcup_{i \leq \gamma_i} \text{EP}_\gamma \times \text{EP}_\gamma$ as the minimal relation closed under the rules in Figure 3.3 Inference rules consist of the rules for equivalence relation and the rules for “α-equivalence” (see, e.g., [36, Sect. 4.1] for λ-calculus).

**Inference rules:**

$$\begin{array}{ll}
& \varphi \equiv \psi \quad \varphi \equiv \psi' \quad \psi \equiv \psi' \\
& \varphi \equiv \rho \quad \varphi \star \psi \equiv \psi' \quad \varphi[z/x] \equiv \psi[z/y] \\
& \exists x. \varphi \equiv \exists y. \psi \\
& \varphi \equiv \varphi' \quad \psi \equiv \psi' \\ \\
& \text{Axioms:} \\
& (=1) x = y \equiv y = x \quad (=2) x = x \star \varphi \equiv \varphi \quad (=3) x = y \star \varphi[x/z] \equiv x = y \star \varphi[y/z] \quad (=4) \exists x. x = y \equiv y = y \\
& (\ast1) \varphi \star (\psi \star \rho) \equiv (\varphi \star \psi) \star \rho \quad (\ast2) \varphi \star \psi \equiv \psi \star \varphi \quad (\ast3) \varphi \star T \equiv \varphi \quad (\ast4) \exists x. \exists y. \varphi \equiv \exists y. \exists x. \varphi \\
& (\ast5) \exists x. \varphi \equiv \exists x. \varphi \star \psi \quad (V1) \varphi \vee (\psi \vee \rho) \equiv (\varphi \vee \psi) \vee \rho \quad (V2) \varphi \vee \psi \equiv \psi \vee \varphi \quad (V3) \varphi \vee \psi \equiv \varphi \quad (\forall) \varphi \equiv \varphi \vee \varphi \\
& (\forall4) \varphi \equiv \varphi \vee \varphi \equiv (\forall5) \exists x. \varphi \equiv \exists x. \varphi \vee (\exists x. \psi) \equiv (\exists x. \varphi) \vee (\exists x. \psi) \equiv (\forall6) \varphi \star (\psi \vee \rho) \equiv (\varphi \star \psi) \vee (\varphi \star \rho) \\
& \text{1: } z \text{ is a fresh variable.}
\end{array}$$

**Figure 3** An axiomatization of the equational theory under GI-semantics of PP/EP formulas.

---

3 We assume that the left- and right-hand side formulas have an identical type. This restriction implicitly implies the following: when their graph languages are not empty, $x \notin \text{FV}_1(\psi)$ in (32), $x \in \text{FV}_1(\varphi)$ in (4=), and $y \neq x$ in (4=), respectively. Also, note that we can use (ff) even if ty(ψ) ≠ Ø, because ff has any type.
Theorem 19. The system in Figure 3 is sound and complete for the equational theory under GI-semantics of PP/EP formulas, that is, for every \( \varphi, \psi \in \text{EP}^*, \varphi \simeq \psi \text{ iff } \varphi \equiv_{\text{GI}} \psi. \)

In the next subsection, we prove this theorem. The following is a proof sketch.

Proof Sketch of Theorem 19. The soundness is straightforward. For completeness, we show by using the rules in Figure 3 that we can transform each formula into a normal form in two steps: (1) transform each EP formula into a disjunctive normal form of PP formulas; (2) transform each PP formula into a formula of the form \( \varphi_G \) in Definition 16.

4.1 Proof of Theorem 19

Proposition 20. (1): \( \varphi_G \mid \varphi_G' \). (2): If there is an isomorphism \( h \) from \( G \) to \( H \), then \( \varphi_G^x = \varphi_H^{y/x} \). (3): If \( G \equiv H \), then \( \varphi_G \equiv \varphi_H \).

Proof. (1): By permuting names using \((x_1)x_2, (x_3)\) for \( \varphi_1 \) and \( \varphi_2 \), \((x_1)(x_2)\) for \( \varphi_1 \) and \( \varphi_2 \), respectively. (2): Since they are the same up to variable names. (3): By (2)(1).

Hereafter in this section, relying on this proposition, we write \( \varphi_G^x \) as \( \varphi_G \) for simplicity.

Lemma 21. For every PP formula \( \varphi \): (1): Let \( x \in \text{FV}_1(\varphi) \) and \( y \neq x \). Then, \( \exists x,y \varphi \simeq_{\text{(32)}} \varphi[y/x] \). (2): Let \( z_1 \ldots z_n \in \text{Perm}(\text{FV}_1(\varphi)) \), \( k \in \mathbb{N} \), and \( f,g : [k] \to [n] \) be maps. Let \( \sim \) be the minimal equivalence relation on \([n]\) such that for every \( i \in [k] \), \( f(i) \sim g(i) \) and let \( I_1 \ldots I_m \) be a permutation of all the quotient classes of \([n]\) w.r.t. \( \sim \). Then, \( \exists z_1 \ldots z_n \left( k \prod_{i=1}^{2} \varphi[z_i] \right) \sim \varphi \). Here, \( z_1, \ldots, z_n \) are pairwise distinct variables.

Proof. (1): \( \exists x,y \varphi \simeq_{\text{(32)}} \exists x,y \varphi[y/x] \simeq_{\text{(32)}} \varphi[y/x] \). (2): By induction on \( k \). Case \( k = 0 \). \( \exists z_1 \ldots z_n \top \simeq_{\text{(32)}} \exists z_1 \ldots z_n \varphi \). Case \( k \geq 1 \). Then, \( \exists z_1 \ldots z_n \left( \prod_{i=1}^{k-1} \varphi[z_i] \right) \simeq \exists z_1 \ldots z_n \left( \prod_{i=1}^{k-1} \varphi[z_i] \right) \). (Apply \((=2)\) if \( f(k) \neq g(k) \) and \((1)\) if \( f(k) = g(k) \).)

Here, we assume without loss of generality by \((\exists)\) that \( z_{I_m'} = \varphi[z_{I_m'}] \).

Lemma 22. For every PP formula \( \varphi \), if \( G \models_{\text{GI}} \varphi \), then \( \varphi \simeq \varphi_G \).

Proof. By induction on the structure of PP formulas. Case \( \varphi \equiv \top \). By \( \varphi_G \equiv \top \simeq_{\text{(32)}} \top \).

Case \( \varphi \equiv x \). By \( \varphi_{G \equiv x} \equiv_{\text{(32)}} \exists z.z = x \simeq_{\text{(32)}} x = x \).

Case \( \varphi \equiv y \) where \( x \neq y \).
By $\varphi_{\ast} \simeq_{(e)} \exists z, z = x \ast z = y \simeq_{\text{Lemma 21}(1)} x = y$. Case $\varphi \equiv a(x_{f(1)}, \ldots, x_{f(n)})$ where $f: [n] \to [k]$ is a surjective map for some $k$. Then,

$$\varphi_{\ast} \simeq_{\text{Lemma 21}(1)} \exists z_{k} \ldots z_{1}, (\bigstar_{i=1}^{k} z_{i} = x_{i}) \ast a(z_{f(1)}, \ldots, z_{f(n)}) \simeq_{\text{Lemma 21}(1)} \cdots \simeq_{\text{Lemma 21}(1)} a(z_{f(1)}, \ldots, z_{f(n)})[x_{1} \ldots x_{k} / z_{1} \ldots z_{k}] \equiv \varphi.$$ 

Case $\varphi \equiv \varphi_{1} \ast \varphi_{2}$. Let $G_{1}$ and $G_{2}$ be such that $G \cong G_{1} \otimes G_{2}, G_{1} \models \varphi_{1}, G_{2} \models \varphi_{2}$. By I.H., $\varphi_{1} \simeq_{\varphi_{1}},$ and $\varphi_{2} \simeq_{\varphi_{2}}$. We denote them by $\varphi_{G_{1}} \equiv \exists z_{1}, \ldots, z_{n}, \bigstar_{i=1}^{k} z_{g_{i}(i)} = x_{i} \ast \bigstar_{i=1}^{m} \tilde{\varphi}_{i}$, and $\varphi_{G_{2}} \equiv \exists z_{n+1} \ldots z_{n}, \bigstar_{i=1}^{k} z_{g_{2}(i)} = x_{i} \ast \bigstar_{i=m+1}^{m} \tilde{\varphi}_{i}$, respectively. Here, $g_{1}: [k'] \to [n]$ and $g_{2}: [k] \to [n]$ are some maps. We assume, without loss of generality that $z_{1}, \ldots, z_{n}$ are pairwise distinct and $k' \leq k$ (by swapping $G_{1}$ and $G_{2}$ appropriately using ($*2$)). Then,

$$\varphi \simeq_{\text{I.H.}} \varphi_{G_{1}} \ast \varphi_{G_{2}} \simeq_{(31) \simeq_{(32)} (a1) \ast (e2)} \exists z_{1}, \ldots, z_{n}, \bigstar_{i=1}^{k} z_{g_{1}(i)} = x_{i} \ast \bigstar_{i=1}^{k} z_{g_{2}(i)} = x_{i} \ast (\bigstar_{i=1}^{m} \tilde{\varphi}_{i}) \simeq_{(a3) \ast (2) \simeq_{(33)} (a1) \ast (e2)} \exists z_{1}, \ldots, z_{n}, \bigstar_{i=1}^{k} z_{g_{1}(i)} = z_{g_{2}(i)} \ast \bigstar_{i=1}^{k} z_{g_{2}(i)} = x_{i} \ast (\bigstar_{i=1}^{m} \tilde{\varphi}_{i}) \simeq_{\text{Lemma 21}(2)} \exists z_{1}, \ldots, z_{m}, \bigstar_{i=1}^{k} z_{g(i)} = x_{i} \ast (\bigstar_{i=1}^{m} \tilde{\varphi}_{i}).$$

Here, $\sim$ and $I_{1} \ldots I_{m}$ are the ones obtained from Lemma 21(2). Case $\varphi \equiv \exists y, \varphi_{1}$. Let $G_{1}$ be such that $G \cong G_{1} / y$ and $G_{1} \models \varphi_{1}$. By I.H., $\varphi_{1} \simeq_{\varphi_{1}}$. We denote it by $\varphi_{G_{1}} \equiv \exists z_{1}, \ldots, z_{n}, \bigstar_{i=1}^{k} z_{g(i)} = x_{i} \ast \bigstar_{i=1}^{m} \tilde{\varphi}_{i}$. Here, $g: [k] \to [n]$ is a map, and we assume, without loss of generality that $y, z_{1}, \ldots, z_{n}$ are pairwise distinct. Then, $y = x_{i}$ for some $i \in [k]$ (note $y \notin FV(\varphi_{1})$). We assume, without loss of generality by ($*1$)($*2$) that $y = x_{k}$. Then, $\varphi \simeq_{\text{I.H.}} \exists y, \varphi_{G_{1}} \simeq_{(31) \simeq_{(32)} (a1) \ast (e2)} \exists z_{1}, \ldots, z_{n}, \bigstar_{i=1}^{k} z_{g(i)} = x_{i} \ast (\bigstar_{i=1}^{m} \tilde{\varphi}_{i}) \simeq \varphi_{G}$. △

**Proof of Theorem 19 for PP formulas.** Assume $\psi \equiv a_{i} \ast \psi$. By Proposition 17(1), $G(\psi) = \mathcal{G}(\psi) = \mathcal{G}^{\ast}(\psi) = \{G\}^{\ast}$ for some $G$. Then, $\psi \simeq_{\text{Lemma 22}} \varphi_{G} \simeq_{\text{Lemma 22}} \psi$. △

In the following, we consider EP formulas.

**Lemma 23.** If $\{G_{1}, \ldots, G_{m}\}^{\ast} = \{H_{1}, \ldots, H_{m}\}^{\ast}$, then $\varphi_{G_{1}}^{n_{1}} \ast \cdots \ast \varphi_{H_{1}}^{n_{1}}$.

**Proof.** By the assumption, let $f: [n] \to [m]$ be a map such that $G_{i} \models H_{f(i)}$ for every $i \in [n]$. Then, $\varphi_{G_{1}}^{n_{1}} \ast \cdots \ast \varphi_{H_{1}}^{n_{1}}$ for some $k$. △

**Lemma 24.** For all $\varphi \in \mathcal{E}_{1}^{p}$, there exists some $\langle \varphi_{i} \rangle_{i=1}^{n} \in (\mathcal{P}_{1}^{p})^{\ast}$ such that $\varphi \simeq \bigwedge_{i=1}^{n} \varphi_{i}$.

**Proof.** By induction on the structure of $\varphi$. Case $\varphi \equiv \emptyset$. By letting $n = 0$, Case $\varphi \equiv \emptyset$. By letting $n = 1$. Case $\varphi \equiv \varphi(1) \ast \varphi(2)$. For $l \in [2]$, let $\langle \varphi_{i} \rangle_{i=1}^{n}$ be the one obtained by I.H. w.r.t. $\varphi(i)$. If $n_{1} = 0$ or $n_{2} = 0$, then $\varphi \simeq_{(a2)} \emptyset$. Otherwise, $\varphi \simeq_{(v1) \ast (v2)} \bigwedge_{i=1}^{n_{1}} \bigwedge_{j=1}^{n_{2}} \varphi_{i} \ast \varphi_{j}$ (and apply ($v1$)($v2$)). Case $\varphi \equiv \varphi(1) \ast \varphi(2)$. Let $\langle \varphi_{i} \rangle_{i=1}^{n}$ and $\langle \varphi_{i} \rangle_{i=m+1}^{n}$ be the ones obtained by I.H. w.r.t. $\varphi(1)$ and $\varphi(2)$, respectively. Then, $\varphi \simeq_{(v1) \ast (v2)} \bigwedge_{i=1}^{n} \varphi_{i}$. Case $\varphi \equiv \exists x. \varphi(1)$. Let $\langle \varphi_{i} \rangle_{i=1}^{n}$ be the one obtained by I.H. w.r.t. $\varphi(1)$. If $n = 0$, then $\varphi \equiv \exists x. \varphi(1)$. △

**Lemma 25.** For every EP formula $\varphi$ and finite sequence $\tilde{G}$ s.t. $\mathcal{G}(\varphi) = \mathcal{O}(\tilde{G})^{\ast}$, $\varphi \simeq \varphi_{\tilde{G}}$. △
Proof. By \( \varphi \simeq_{\text{Lemma 24}} \bigvee_{i=1}^{n} \varphi_i \simeq_{\text{Lemma 22}} \bigvee_{i=1}^{n} \varphi G_i \simeq_{\text{Lemma 23}} \varphi_G \). Here, for each \( i \in [n] \), \( \varphi_i \) is a PP formula and \( G_i \) is a graph such that \( G_i \models \varphi_i \).

Proof of Theorem 19 for EP formulas. Assume \( \psi \models_{G} \rho \). Let \( G \) be a finite sequence such that \( G(\psi) = G(\rho) = \text{Occ}(G) \models_{G} \). Then, \( \psi \models_{\text{Lemma 25}} \varphi_G \simeq_{\text{Lemma 25}} \rho \).

5 Kleene Theorems Between EPs and HRGs

In this section, we show that EP(LFP) (resp. EP(TC)) has the same expressive power as the class of HRGs (resp. linear HRGs). To this end, we introduce term (formula) rewriting systems [2] (FRSs) and show the equivalence above via FRSs. Intuitively, FRSs play the same role as finite automata with transitions labelled by regular expressions [7] (so-called extended finite automata) in translating finite automata into regular expressions.\(^4\)

5.1 Formula Rewriting Systems (FRSs)

Definition 26. A formula rewriting system \((\text{FRS}\{\mathcal{C}\})\) \( F \) over an ordinal-typed alphabet \( A \) is a tuple \((\mathcal{X}^F, \mathcal{R}^F, \sigma^F)\), where \( \mathcal{X}^F \) is an ordinal-typed alphabet disjoint with \( A \) for denoting (non-terminal) labels, \( \mathcal{R}^F \) is a finite set of pairs \( r = (X \mathcal{X}, \varphi) \) (written \( X \mathcal{X} \leftarrow \varphi \)) of a strictly atomic \( \mathcal{X}^F \)-formula \( X \mathcal{X} \) and a \( \mathcal{C}^\text{Occ}(\mathcal{F}) \)-formula \( \varphi \) for denoting rewriting rules, and \( \sigma^F \) is a strictly atomic \( \mathcal{X}^F \)-formula for denoting the source formula. Here, for an ordinal-typed alphabet \( \mathcal{X} \), we say that \( \varphi \) is a strictly atomic \( \mathcal{X} \)-formula if \( \varphi \) is of the form \( X \mathcal{X} \), where \( X \in \mathcal{X} \) and the elements of \( \mathcal{X} \) are pairwise distinct.

Definition 27. For an FRS\{\mathcal{C}\} \( F = (\mathcal{X}, \mathcal{R}, \mathcal{S}) \) over an ordinal-typed alphabet \( A \), the binary relation \( \models_{\mathcal{F}} \subseteq \bigcup_{\mathcal{F} \subseteq \mathcal{X}_{\mathcal{C}^\mathcal{A}; \mathcal{R}_{\mathcal{F}}} \mathcal{G}_{\mathcal{X}^\mathcal{F}} \times \text{Fml}_{\mathcal{X}^\mathcal{F}} \) is defined as the least \( \simeq \)-closed relation closed under all the rules of \( \models_{\mathcal{F}} \) (in Definition 14) and the following rule: If \( X \mathcal{X} \leftarrow \varphi \in \mathcal{R} \), then \( G \models_{\mathcal{F}} \varphi [G] X \mathcal{Y} \). We write \( G \models_{\mathcal{F}} \mathcal{F} \) for \( G \models_{\mathcal{F}} \mathcal{S} \). The graph language of \( \mathcal{F} \) is defined by \( \mathcal{G}(\mathcal{F}) \triangleq \{ G \mid G \models_{\mathcal{F}} \mathcal{F} \} \).

Example 28 (cf. Example 12). Let \( \mathcal{F} \) be the FRS\{PP\} over \( A_E \), defined by \( \mathcal{F}^X = \{ [0], X \mapsto [2] \} \), \( \mathcal{R}^X = \{ ([S], (E), (S), (p)) \} \), where each rule in \( \mathcal{R}^X \) is as follows:

\[
\begin{align*}
(S) &\leftarrow \exists xy, Xxy \\
(E) &\leftarrow Xxy \\
(s) &\leftarrow \exists z, xz, xz, Xzy \\
p) &\leftarrow Xxy, Xxy
\end{align*}
\]

Then, \( \mathcal{G}(\mathcal{F}) \) is the set of all series-parallel graphs. For example, \( \alpha_{\text{E}} \models_{\mathcal{F}} \mathcal{F} \) is shown by:\(^5\)

In general, the following proposition is immediate from the translations between graphs and PP formulas in Proposition 17(1). Also, we use linear/(n-)recursive for FRS[PP]s in the same manner as for HRGs.

\(^4\) FRS\{\mathcal{C}\} is essentially the same as positive Datalog [20, Section 9] if \( \mathcal{C} \) is the class of conjunctive queries.

\(^5\) Double line denotes that 0 or more rules are applied in the place.
Proposition 29. For every $G$, some HRG (resp. linear HRG) recognizes $G$ iff some FRS[PP] (resp. linear FRS[PP]) recognizes $G$.

An FRS $F$ is deterministic if for every $X \in \mathcal{X}^F$, the number of rules of the form $X\bar{x} \leftarrow \varphi$ is at most one. In Example 28, we can put together the three rules for $X$ as follows in FRS[EP]:

\[
(S) \quad S \leftarrow \exists x y. X x y \quad (X) \quad X x y \leftarrow (E x y \ast X x y) \vee (\exists z. X x z \ast X z y).
\]


Proof. (i) $\Rightarrow$ (ii): By the same argument as above. (ii) $\Rightarrow$ (iii): Trivial. (iii) $\Rightarrow$ (i): By replacing each rule $X\bar{x} \leftarrow \varphi$ with $X\bar{x} \leftarrow \psi_1, \ldots, X\bar{x} \leftarrow \psi_n$. Here, $\psi_1, \ldots, \psi_n$ are PP formulas such that $\varphi \equiv \bigwedge_{n=1}^\infty \psi_i$ (Lemma 24).

The following are useful properties of hyperedge replacing and glueing.

Proposition 31. For every FRS[EP(LFP)] $F$: (1) If there is a derivation tree that shows $G \models \varphi$ from the assumptions $(H_1 \models \varphi_1), \ldots, (H_n \models \varphi_n)$, then there exist some $G'$ and $e_1, \ldots, e_n$ such that $G \equiv G'[H_1 \ldots H_n/e_1 \ldots e_n]$. (2) If there is a derivation tree that shows $G[H_1 \ldots H_n/e] \models \varphi$ from the assumptions $(H_1 \models \varphi_1), \ldots, (H_n \models \varphi_n)$, then there exist some $G'$ and $e_1, \ldots, e_n$ such that $G \equiv G'[H_1 \ldots H_n/e_1 \ldots e_n]$. For every FRS[EP(TC)] $F$: (3) If there is a derivation tree that shows $G \models \varphi$ from $H \models \psi$ and $\text{ty}(H) \cap \text{BV}_1(\varphi) = \emptyset$, then there exist some $G'$ such that $G \equiv G' \odot H$. (4) If there is a derivation tree that shows $G \odot H \models \varphi$ from $G' \odot H \models \psi$, $\text{ty}(H) \cap \text{BV}_1(\varphi) = \emptyset$, $\text{ty}(H') \cap \text{BV}_1(\varphi) = \emptyset$, and $\text{ty}(H) = \text{ty}(H')$, then there is a derivation tree that shows $G \odot H' \models \varphi$ from $G' \odot H' \models \varphi$.


5.2 Equivalence of EP(LFP) formulas and HRGs (Theorem 1(2))

In the following, by using Proposition 29 and 30, we show that EP(LFP) has the same expressive power as (deterministic) FRS[EP].

From EP(LFP) formulas to FRS[EP]. We say that an EP(LFP) formula $\varphi$ is simple if (a) all the second-order variables $X$ occurring in the form $[\text{LFP}_{F,X}(\varphi)]\vec{y}$ are pairwise distinct, (b) $\bar{x} = \vec{y} = \vec{\iota}$ for each subformula of the form $[\text{LFP}_{F,X}(\varphi)]\vec{y}$, and (c) $\bar{x} = \vec{\iota}$ for each subformula of the form $X\bar{x}$. This restriction simplifies the translation and the proof.

Lemma 32. Every EP(LFP) formula $\varphi$ has a GI-equivalent simple EP(LFP) formula.

Proof Sketch. For (a), rename variables appropriately. For (b)(c), use the following translations, respectively: $[\text{LFP}_{F,X}(\varphi)]\vec{y} \rightarrow \exists \vec{x}. \vec{z} = \vec{y} \ast \exists \iota. t \rightarrow \vec{z} \ast [\text{LFP}_{F,X}(\exists \vec{x}. \vec{z} = t \ast \exists \vec{x}. \vec{z} = \vec{z} \ast \varphi)]\vec{\iota}$ and $X\bar{x} \rightarrow \exists \vec{z}. \vec{z} = \vec{\iota} \ast [\text{LFP}_{F,X}(\exists \vec{z}. \vec{z} = \vec{\iota} \ast \exists \vec{x}. \vec{z} = \vec{z} \ast \varphi)]\vec{\iota}$. Here, $\vec{z}$ is a sequence of fresh variables.

Let $\vec{z}$ be a map from each EP(LFP) formula $\varphi$ to a permutation $\vec{z}_\varphi$ of $\text{BV}_1(\varphi)$. Figure 4 gives a translation from a simple EP(LFP) formula $\varphi$ into an FRS[EP] $F_\varphi = (\mathcal{X}_\varphi, \mathcal{R}_\varphi, s_\varphi)$.

---

6 This translation is essentially the same as the translation from existential fixpoint logic to Datalog, see, e.g., [20, Theorem 9.1.4]. The only difference is the semantics.
Spatial Existential Positive Logics for Hyperedge Replacement Grammars

\[ F_\varphi \triangleq \{ (s_\varphi, \{ s_\varphi \}) \mid \varphi \in \mathcal{LFP}(s_\varphi) \} \]

\[ F_{\exists \varphi, \psi} \triangleq \{ \{ s_\varphi \} \cup X_\psi, \{ s_\varphi \leftarrow \exists x.s_\psi \} \cup \mathbb{R}_\psi, S_\varphi^x \} \]

\[ F_{\psi \bullet} \varphi \triangleq \{ \{ s_\varphi \} \cup X_\psi \cup X_\varphi, \{ s_\varphi \leftarrow s_\psi \bullet s_\varphi \} \cup \mathbb{R}_\psi \cup \mathbb{R}_\varphi, S_\psi^x \} \quad (\bullet \in \{ \ast, \vee \}) \]

\[ F_{[\mathcal{LFP}, X(\psi)]} \triangleq \{ \{ s_\varphi \}, X \} \cup \mathbb{X}_\psi, \{ s_\varphi \leftarrow X, X \leftarrow s_\psi \} \cup \mathbb{R}_\psi, S_\varphi^x \}

**Lemma 33.** A translation from EP(LFP) formulas into (deterministic) FRS[EP].

**Proof.** \( G \models_{GL} \varphi \Rightarrow G \models_{\mathcal{LFP}} \varphi \): By induction on the size of the derivation tree of \( G \models_{GL} \varphi \).

**From FRS[EP] to EP(LFP) formulas.** This part is shown by folding non-terminal labels for a given deterministic FRS[EP] as follows: for non-0-recursive labels \( X \), replace each occurrence of \( X \) with the formula corresponding to \( X \) in the rule; for 0-recursive labels, use the LFP. Note that by Proposition 30, from an FRS[EP], we can obtain a deterministic one.

**Lemma 34.** Every deterministic FRS[EP(LFP)] has a GI-equivalent EP(LFP) formula.

**Proof.** Let \( F = (X, R, S) \). Let \( \#_n(F) \triangleq \#(X \setminus \{ S \}) \) and \( \#_i(F) \) be the number of 0-recursive labels in \( F \). We prove by induction on the pair \( (\#_n(F), \#_i(F)) \). Case \( \#_n(F) = 0 \): Let \( R = \{ Sx \leftarrow \psi \} \). Then, \( G(F) = G(\psi[z/x]) \). Case \( \#_n(F) > \#_i(F) \). Then, there exists a non-0-recursive label \( X_0 \in X \setminus \{ S \} \). Let \( X_0x_0 \leftarrow \psi_0 \in R \). Let \( F' \triangleq (X \setminus \{ X_0 \}, \{ X \leftarrow \psi \} \cup \mathbb{X}_\psi \cup X_0 \leftarrow \psi \} \) denotes the formula \( \psi \) in which each \( X_0 \) has been replaced with \( \psi_0[z/x_0] \). Then, \( G(F) = G(F') \) because there is a trivialization between derivation trees of \( F \) and those of \( F' \). Also by I.H., there exists an EP(LFP) formula \( \varphi \) such that \( G(F') \models_{GL} \varphi \). Therefore, \( G(F) \models_{GL} \varphi \). For the other case \( i.e., \#_i(F) \geq 1 \), there exists a 0-recursive label \( X_0 \in X \). Let
\[ X_0 \vec{z}_0 \leftarrow \psi_0 \in \mathcal{R}. \text{ Let } \mathcal{F}' := (X, \{X \vec{z} \leftarrow \psi \in \mathcal{R} | X \neq X_0\} \cup \{X_0 \vec{z}_0 \leftarrow [\text{LFP}_{\vec{z}_0, X_0}(\psi_0)]\vec{z}_0\}, \mathbb{S} \). Then, \( G(\mathcal{F}) = G(\mathcal{F}') \) because there exists a transformation between derivation trees of \( \mathcal{F}' \) and those of \( \mathcal{F} \) in the same manner as the proof of Lemma 33. Also by I.H., there exists an EP(LFP) formula \( \varphi \) such that \( G(\mathcal{F}') = G(\varphi) \). Hence, \( G(\mathcal{F}) = G(\varphi) \).

\[ \text{Proof of Theorem 1(2) } \implies \text{. By Lemma 34 (with Proposition 29 and 30).} \]

### 5.3 Equivalence of EP(TC) formulas and linear HRGs (Theorem 1(3))

In the following, by using Proposition 29 and 30, we show that EP(TC) has the same expressive power as the class of linear FRS[PP].

From EP(TC) formulas to linear FRS[PPs]. We say that an EP(TC) formula \( \varphi \) is simple if all the variables \( x \) occurring in the form \( \exists x.\psi \), the variables in \( \vec{z}\vec{u}\vec{w} \) occurring in the form \( [\varphi]\vec{z}\vec{u}\vec{w} \), and the free variables in \( \varphi \) are pairwise distinct. As with Lemma 32, from a given EP(TC) formula, we can obtain a GI-equivalent simple one by renaming variables and using the following translation: \( [\varphi]\vec{z}\vec{u}\vec{w} \simeq [\varphi]\vec{z}'\vec{u}'\vec{w}' \). Here, elements of \( \vec{z} \) and \( \vec{z}' \) are fresh variables. Furthermore, the following holds.

**Lemma 35.** Every EP(TC) formula \( \varphi \) has a GI-equivalent simple EP(TC) formula of the form \( \exists z_0.\varphi_0 \) or \( \top \lor \exists z_0.\varphi_0 \).

**Proof.** If \( \text{FV}_1(\varphi) \neq \emptyset \), then \( \varphi \simeq^\text{GI} \exists z_0.\\ z_0 = x \land \varphi \), where \( x \in \text{FV}_1(\varphi) \) and \( z_0 \) is a fresh variable. Otherwise, let \( \bigvee_{i=1}^{n} \varphi_i = \text{ a disjunctive normal form of } \varphi \), where each \( \varphi_i \) is a prenex normal form EP(TC) formula. Let \( \rho_i \equiv \exists z_i.\psi_i \) if \( \varphi_i \) is of the form \( \exists x.\psi_i \) and \( \rho_i \equiv \top \) otherwise (note that \( \varphi_i \equiv \top \) should because \( \text{FV}_1(\varphi_i) = \emptyset \)). Note that \( \varphi_i \simeq^\text{GI} \rho_i \). Let \( l_1 \ldots l_m \) be the subsequence of \( t_n \) such that for each \( i \in [n], i \in \{l_1, \ldots, l_m\} \) iff \( \rho_i \equiv \top \). Then \( m < n \), \( \varphi \simeq^\text{GI} \top \lor \bigvee_{j=1}^{m} \exists z_i.\psi_i \). Otherwise, \( \varphi \simeq^\text{GI} \bigvee_{i=1}^{n} \exists z_i.\psi_i \). Hence, it has been proven.

Let \( \vec{z} \) be a sequence of pairwise distinct variables. For a simple EP(TC) formula \( \varphi \) such that \( \text{V}_1(\varphi) \subseteq \text{Occ}(\vec{z}) \), we define the linear FRS[PP] \( \vec{\mathcal{F}}_\varphi = (\mathcal{X}_\varphi, \mathcal{R}_\varphi, \vec{\mathcal{S}}_\varphi) \) (we may explicitly write \( \vec{\mathcal{F}}_\varphi = (\mathcal{X}_\varphi, \mathcal{R}_\varphi, \vec{\mathcal{S}}_\varphi, \vec{\rho}_\varphi) \)) in Figure 5. Our construction is based on Thompson’s construction [41] and the product construction (in translating regular expressions into finite automata), but is generalized for first-order variables.

\[ \vec{\mathcal{F}}_\varphi \triangleq \{(\mathcal{S}_\varphi, \mathcal{T}_\varphi), \{\mathcal{S}_\varphi.\vec{z} \leftarrow \varphi \land \mathcal{T}_\varphi.\vec{z}\}, \mathcal{S}_\varphi.\vec{z}\} \]

\[ \vec{\mathcal{F}}_{\exists x.\varphi} \triangleq \{(\mathcal{S}_\varphi, \mathcal{T}_\varphi), \{\mathcal{S}_\varphi.\vec{z} \leftarrow x \land \exists x.\mathcal{S}_\varphi.\vec{z} \land \mathcal{T}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \]

\[ \vec{\mathcal{F}}_{\top \lor \varphi} \triangleq \{(\mathcal{S}_\varphi, \mathcal{T}_\varphi), \{\mathcal{S}_\varphi.\vec{z} \leftarrow \top \lor \mathcal{S}_\varphi.\vec{z} \land \mathcal{T}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \]

\[ \vec{\mathcal{F}}_{\top \lor \varphi} \triangleq \{(\mathcal{S}_\varphi, \mathcal{T}_\varphi), \{\mathcal{S}_\varphi.\vec{z} \leftarrow \mathcal{T}_\varphi.\vec{z} \land \mathcal{S}_\varphi.\vec{z} \land \mathcal{T}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \]

\[ \vec{\mathcal{F}}_{[\varphi_1]_\vec{z}_1 \vec{u}_1 \vec{w}_1 \vec{z}_2 \vec{u}_2 \vec{w}_2} \triangleq \{(\mathcal{S}_\varphi, \mathcal{T}_\varphi), \{\mathcal{S}_\varphi.\vec{z} \leftarrow \vec{u}_1 \lor \mathcal{S}_\varphi.\vec{z} \land \mathcal{T}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \]

\[ \vec{\mathcal{F}}_{\vec{z}_1 \vec{u}_1 \vec{w}_1} \triangleq \{(\mathcal{S}_\varphi, \mathcal{T}_\varphi), \{\mathcal{S}_\varphi.\vec{z} \leftarrow \vec{u}_1 \lor \mathcal{S}_\varphi.\vec{z} \land \mathcal{T}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\} \cup \mathcal{S}_\varphi.\vec{z}\}

\[ \uparrow \] r \langle 0, Y \rangle/ \langle r \rangle \text{ is the rule } r \text{ in which each } X \text{ (resp. } Y \text{) has been replaced with } (X, Y). \]

**Figure 5** Definition of linear FRS[PP] \( \vec{\mathcal{F}}_\varphi \).
Lemma 36. For every simple EP(TC) formula \( \varphi \) and every \( G \in \text{GR}_A^T \) (where \( \varphi \in \text{Fm}_A^T \)), \( G \models_{\text{GI}} \varphi \) iff there is a derivation tree that shows \( G \circ \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{S}_\varphi \bar{z} \) from \( \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{T}_\varphi \bar{z} \).

Proof. \( \Rightarrow \): By induction on the structure of \( \varphi \). The essential case is when \( \varphi = [\bar{v}]_{T^z} \bar{u} \bar{w} \). Let \( G \cong (G_1 \circ \bar{u}_2 \ldots \circ \bar{u}_n G_n) [\bar{u} \bar{w} / \bar{F}] \) be such that \( G_i \models_{\text{GI}} \psi \) for \( i \in [n] \). For notational simplicity, let \( G_{i,n} \cong G_i \circ \bar{u}_2 \ldots \circ \bar{u}_n G_n[i/n] \) for \( i \in [n] \). Note that \( G \cong G_{[1,n]}[\bar{u} / \bar{x}] \) and \( G_{i,n} \cong G_i \circ G_{i+1,n} [\bar{y} / \bar{x}] [\bar{f} \ldots \bar{f} / \bar{g}] \). For each \( i \in [n] \), by I.H., there is a derivation tree \( \bullet - i \) that shows \( G_i \circ G_{i+1,n} [\bar{y} / \bar{x}] \models_{\mathcal{F}_T^z} \mathcal{S}_\psi \bar{z} \) from \( \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{T}_\varphi \bar{z} \). Then, we obtain a derivation tree that shows \( G \circ \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{S}_\varphi \bar{z} \) from \( \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{T}_\varphi \bar{z} \) by concatenating \( \bullet - 1 \)-\( \bullet - n \) using Proposition 31(4) as follows.

\[
\begin{align*}
G_{[1,n]}[\bar{y} / \bar{x}] & \circ \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{S}_\varphi \bar{z} \\
G_1 \circ G_{[2,n]}[\bar{y} / \bar{x}] & \circ \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{S}_\varphi \bar{z} \\
\vdots \bullet - i & \quad \ldots \\
G_{1,n}[\bar{y} / \bar{x}] & \circ \mathcal{G}_T^\text{Occ}(\bar{z}) \models_{\mathcal{F}_T^z} \mathcal{S}_\varphi \bar{z}
\end{align*}
\]
Proof. For the condition (a), we introduce a fresh non-terminal label T and introduce
the rule T⃗ x ← ⃗ x = ⃗ x. For the condition (b), for each rule X⃗ x ← ⃗ x, if ⃗ x does not have non-terminal labels, then we replace the rule with X⃗ x ← ⃗ z (⃗ z = ⃗ x ⋁ ⃗ y), where ⃗ z is a sequence of fresh variables. Otherwise, let Y be the non-terminal label and transform
the PP formula ⃗ x into a GI-equivalent formula of the form ∃⃗ z.ϕ ⋁ ⃗ y by taking its
prenex normal form and reordering the inner formulas appropriately. Then, transform it into the
following formula: ∃⃗ y. (∃⃗ z.⃗ y = ⃗ x ⋁ ⃗ y) ⋁ ⃗ y, where ⃗ y is a sequence of fresh variables. Next,
for each pair of (Y, Y′), let ⟨X⃗ x, Y⃗ y⟩ ← ⃗ z, Y⃗ y⟩ = 1 be a permutation of all the rules of the
form X⃗ x ← ∃⃗ y.ϕ ⋁ Y⃗ y. Without loss of generality, we can assume that ⃗ x1⃗ y1 = ⋯ = ⃗ xn⃗ yn
(so we denote it by ⃗ x⃗ y) by renaming variables. Then, replace these rules with the single rule
X⃗ x ← ∃⃗ y. ⟨Y⃗ y⟩ ⋁ ⃗ y.

Finally, we present a translation from FA-linear FRS[EP]s into EP(TC) formulas.


Proof. By induction on #(X⃗ F). If X⃗ F = {S⃗ F, T⃗ F}, then ⃗ F is denoted by ⟨{S⃗ F}, T⃗ F⟩, S⃗ F ← ∃⃗ z.ϕ ⋁ T⃗ x, T⃗ x ← ⃗ x = ⃗ x, s⃗ F⟩. Thus, ⃗ F is GI-equivalent to the EP(TC) formula ∃⃗ z.ϕ ⋁ ⃗ x = ⃗ x (≤qν ∃⃗ z.ϕ). Otherwise, there exists Y0 ∈ X⃗ F \ {S⃗ F, T⃗ F}. We define F′ = (X⃗ F \ {Y0}, X⃗ F ← ∃⃗ z.ϕ ⋁ X⃗ F, X⃗ F ← ⃗ x = ⃗ x, s⃗ F′), where elements of ⃗ x⃗ y⃗ z are pairwise distinct.

<table>
<thead>
<tr>
<th>References</th>
</tr>
</thead>
</table>


Alternative Definition of GI-semantics

In this section, we present an alternative definition of GI-semantics in Definition 14.

Definition 40. The GI-semantics \([\phi] \subseteq \text{GR}_X^\tau\) of a formula \(\phi \in \text{Fml}_X^\tau\) is defined as follows.
Spatial Existential Positive Logics for Hyperedge Replacement Grammars

By I.H., following.

Proof of (1).

Proposition 41. $G(\varphi) = [\varphi]$.

Proof. $G \models^{\text{ct}} \varphi \Rightarrow G \models [\varphi]$. By induction on the size of the derivation tree of $G \models^{\text{ct}} \varphi$. The cases of (At)(*) and (Ψ) are immediate. For (LFP), let $G \models H[G_1 \ldots G_n/c^X_\varphi][\vec{y}/\vec{z}]$ be such that $H \models^{\text{ct}} \varphi$ and $G_i \models^{\text{ct}} [\text{LFP}_{X,Y,T} \psi]_i$ for $i \in [m]$. By I.H., $H \models [\psi]_i$ and $G_i \models [\text{LFP}_{X,Y,T} \psi]_i$. Then by the definition of [*], $H[G_1 \ldots G_n/c^X_\varphi] \models [\text{LFP}_{X,Y,T} \psi]_i$, and hence $H[G_1 \ldots G_n/c^X_\varphi][\vec{y}/\vec{z}] \models [\text{LFP}_{X,Y,T} \psi]_i$. Therefore, $G \models [\text{LFP}_{X,Y,T} \psi]_i$. $G \models [\varphi]_i \Rightarrow G \models^{\text{ct}} \varphi$: By induction on the structure of $\varphi$. Because other cases are immediate, we only write the case of LFP. Let $\varphi = [\text{LFP}_{X,Y,T} \psi]_i$. Then, let $f : 2^{G^{\text{ct}}_{X,Y,T}} \rightarrow 2^{G^{\text{ct}}_{X,Y,T}}$ be the function defined by $f(\vec{y}) = G \cup \{H[\vec{G}/c^X_{\varphi}]_i/\vec{z} \mid H \models [\psi]_i, \vec{G} \in G^{ct}_{X,Y,T} \}$, and hence $H[G_1 \ldots G_n/c^X_{\varphi}][\vec{y}/\vec{z}] \models [\text{LFP}_{X,Y,T} \psi]_i$. Therefore, by $G \models [\text{LFP}_{X,Y,T} \psi]_i$, it has been proved.

B Proof of Proposition 31

Proof of (1). By induction on the structure of the derivation tree. Case (At): Trivial since $n = 0$.

Case (*): Then by reordering the sequence $\langle H, \models^{\text{ct}} \varphi_i \rangle_{i=1}^{m}$, the derivation tree forms the following.

\[
\begin{array}{c}
(H, \models^{\text{ct}} \varphi_i)_{i \in [m]} \\
(G_1, \models^{\text{ct}} \varphi_1) \\
(G_2, \models^{\text{ct}} \varphi_2) \\
(G_1 \otimes G_2, \models^{\text{ct}} \varphi_1 \ast \varphi_2)
\end{array}
\]

By I.H., $G_1 \models G'_1[H_1 \ldots H_n/c_1]$ and $G_2 \models G'_2[H_{m+1} \ldots H_n/c_2]$. Then by Proposition 9, $G_1 \otimes G_2 \models G'_1[H_1 \ldots H_m/c_1] \otimes G'_2[H_{m+1} \ldots H_n/c_2] \equiv (G'_1 \otimes G'_2)[H_1 \ldots H_n/c]$. Case (3): Then, the derivation tree forms the following.

\[
\begin{array}{c}
(H, \models^{\text{ct}} \varphi_i)_{i=1}^{m} \\
(G_1, \models^{\text{ct}} \varphi_1) \\
(G_1[t/x], \models^{\text{ct}} \exists x \varphi_1)
\end{array}
\]
By I.H., $G_1 \equiv G_1'[H_1 \ldots H_n/\vec{e}]$. Then, $G_1'[H_1 \ldots H_n/\vec{e}][\vec{t}/\vec{x}] \equiv (G'_1[\vec{t}/\vec{x}])[H_1 \ldots H_n/\vec{e}]$ by Proposition 9.

Case $\langle \vee \rangle$: Then, the derivation tree forms the following.

$$
\frac{\langle H, \psi \rangle}{\bar{\psi} \circ \vec{t}}
\qquad \frac{\bar{\psi}}{\bar{\psi} \psi_1 \vee \psi_2}
\frac{\bar{\psi}}{\bar{\psi} \psi_1}
\frac{\bar{\psi}}{H \psi \varphi}
\frac{H \psi \varphi}{H \psi \varphi}

\frac{G \equiv \bar{\psi} \psi_1 \vee \psi_2}{G \equiv \bar{\psi} \psi_1 \vee \psi_2}
\frac{G \equiv \bar{\psi} \psi_1}{G \equiv \bar{\psi} \psi_1}
$$

By I.H., $G \equiv G'_1[H_1 \ldots H_n/\vec{e}]$, hence this part has been proved.

Case $LFP$: Then, by reordering the sequence $\langle H_i \models \varphi \rangle_{i=1}^n$, the derivation tree forms the following where $m_k = n$.

$$
\frac{\langle H_i \models \varphi \rangle_{i=1}^n}{\bar{\varphi} \circ \vec{t}}
\qquad \frac{\bar{\varphi}}{\bar{\varphi} \varphi_1 \vee \varphi_2}
\frac{\bar{\varphi}}{\bar{\varphi} \varphi_1}
\frac{\bar{\varphi}}{H \varphi \varphi_1}
\frac{H \varphi \varphi_1}{H \varphi \varphi_1}

\frac{G \equiv \bar{\varphi} \varphi_1 \vee \varphi_2}{G \equiv \bar{\varphi} \varphi_1 \vee \varphi_2}
\frac{G \equiv \bar{\varphi} \varphi_1}{G \equiv \bar{\varphi} \varphi_1}
$$

By I.H., $H \equiv H'[H_1 \ldots H_{m_n}/\vec{e}_0]$ and $G_j \equiv G'_j[H_{m_j+1} \ldots H_{m_k}/\vec{e}_j]$ for $j \in \{k\}$. Then, for some $\vec{c}_1, \ldots, \vec{c}_k, \vec{c}_1', \ldots, \vec{c}_k', \vec{c}_1, \vec{c}_1', \vec{c}_k, \vec{c}_1', \vec{c}_k', \vec{c}_k', H'[H_{m_0}, \ldots, H_{m_k}/\vec{e}_0]$, and $\vec{c}_k'$. Note that, since $H_1, \ldots, H_{m_0}$ don’t have $X$-labelled edge by the assumption, $|\vec{c}_k'|$ coincides with $|\vec{c}_k|$. By I.H., $G \equiv G'_1[H_1 \ldots H_n/\vec{e}]$, hence this part has been proved.

**Proof of (2).** By induction on the structure of the derivation tree. Case (At): Trivial since $n = 0$. Case ($*$): Then, by reordering the sequence $\langle H_i \models \varphi \rangle_{i=1}^n$ and (1), the derivation tree forms the following left-hand side (note that by $G_1[H_1 \ldots H_n/\vec{e}_1][\vec{t}/\vec{x}] \equiv G_1[\vec{t}/\vec{x}][H_1 \ldots H_n/\vec{e}_1]$ by using Proposition 9, $G \equiv G_1[\vec{t}/\vec{x}]$). Then, by I.H. and applying ($*$), this part has been proved by the following right-hand side tree.
20 Spatial Existential Positive Logics for Hyperedge Replacement Grammars

\[
\begin{align*}
(H \models \varphi_{i,1}) & \implies (H \models \varphi_{i,1}) \\
G[H_1 \ldots H_{m_n}/\vec{e}_0] & \models \varphi \quad \text{(LFP)}
\end{align*}
\]  

Case (\lor): Then by (1), the derivation tree forms the following left-hand side where \( l \in [2] \). Then, by I.H. and applying (\lor), this part has been proved by the following right-hand side tree.

\[
\begin{align*}
(H \models \varphi_{i,1}) & \implies (H \models \varphi_{i,1}) \\
G[H_1 \ldots H_{m_n}/\vec{e}_0] & \models \varphi \quad \text{(LFP)}
\end{align*}
\]

Case (LFP): Then, by reordering the sequence \((H \models \varphi_{i,1})\) and (1), the derivation tree forms the following left-hand side where \( m_k = n \). Here, for some \( \vec{e}', \vec{e}'' \), \( H[G[H_1 \ldots H_{m_n}/\vec{e}_0]] \), and \( e_X \).

\[
H[H_1 \ldots H_{m_o}/\vec{e}_0]H[H_1 \ldots H_{m_i}/\vec{e}_1] \ldots G[H_{m_k-1}+1 \ldots H_{m_k}/\vec{e}_k]H[H_{m_k-1} \ldots H_{m_k}/\vec{e}_n] \models H[H_1 \ldots H_{m_o}/\vec{e}_0][H[H_1 \ldots H_{m_i}/\vec{e}_1] \ldots G[H_{m_k-1}+1 \ldots H_{m_k}/\vec{e}_k]H[H_{m_k-1} \ldots H_{m_k}/\vec{e}_n]]\models H[H_1 \ldots H_{m_o}/\vec{e}_0][H[H_1 \ldots H_{m_i}/\vec{e}_1] \ldots G[H_{m_k-1}+1 \ldots H_{m_k}/\vec{e}_k]H[H_{m_k-1} \ldots H_{m_k}/\vec{e}_n]]\models H[H_1 \ldots H_{m_o}/\vec{e}_0][H[H_1 \ldots H_{m_i}/\vec{e}_1] \ldots G[H_{m_k-1}+1 \ldots H_{m_k}/\vec{e}_k]H[H_{m_k-1} \ldots H_{m_k}/\vec{e}_n]]
\]

Note that, since \( H_1, \ldots, H_{m_o} \) don’t have \( X \)-labelled edge by the assumption, \( e_X \) coincides with \( e_X'H[H_1 \ldots H_{m_o}/\vec{e}_0] \). From this, \( G \models H[H_1 \ldots H_{m_o}/\vec{e}_0][H[H_1 \ldots H_{m_i}/\vec{e}_1] \ldots G[H_{m_k-1}+1 \ldots H_{m_k}/\vec{e}_k]H[H_{m_k-1} \ldots H_{m_k}/\vec{e}_n]]\models H[H_1 \ldots H_{m_o}/\vec{e}_0][H[H_1 \ldots H_{m_i}/\vec{e}_1] \ldots G[H_{m_k-1}+1 \ldots H_{m_k}/\vec{e}_k]H[H_{m_k-1} \ldots H_{m_k}/\vec{e}_n]]\models H[H_1 \ldots H_{m_o}/\vec{e}_0][H[H_1 \ldots H_{m_i}/\vec{e}_1] \ldots G[H_{m_k-1}+1 \ldots H_{m_k}/\vec{e}_k]H[H_{m_k-1} \ldots H_{m_k}/\vec{e}_n]]
\]

Proof of (3). By induction on the structure of the derivation tree. Case (At): This case does not occur since the tree does not have any assumptions.

Case (+): Then, the derivation tree forms the following (or \( (\bullet) \) exists in the above of \( G_2 \), but this case is in the same way as below).
By I.H., \( G_1 \cong G'_1 \otimes H \). Then, \( G_1 \otimes G_2 \cong (G'_1 \otimes G_2) \otimes H \).

Case (3): Then, the derivation tree forms the following.

\[
\begin{array}{c}
H \models \phi' \\
\vdots \vdots \\
G_1 \models \phi' \quad G_2 \models \phi_2 \\
G_1 \otimes G_2 \models \phi' \otimes \phi_2
\end{array}
\]

By I.H., \( G_1 \cong G'_1 \otimes H \). Then, \((G'_1 \otimes H)[t/x] \cong G'_1[t/x] \otimes H\) by Proposition 9 and \( x \notin ty(H)\) (because \( BV(\exists x. \phi_1) \cap ty(H) = \emptyset\)).

Case (\( \forall \)): Then, the derivation tree forms the following.

\[
\begin{array}{c}
H_t \models \phi' \\
\vdots \vdots \\
G \models \phi' \quad G \models \phi_1 \\
G \models \phi' \otimes \phi_2
\end{array}
\]

By I.H., \( G \cong G' \otimes H \), hence this part has been proved.

Case (TC): Then, the derivation tree forms the following for some \( j \in [k] \).

\[
\begin{array}{c}
H \models \phi' \\
\vdots \vdots \\
\frac{G_1 \models \phi' \quad G_j \models \phi' \quad G_k \models \phi'}{(G_1 \otimes \ldots \otimes G_j \otimes \ldots \otimes G_k)[t/x][y/x]} \models \phi' \quad \text{(TC)}
\end{array}
\]

By I.H., \( G_j \cong G'_j \otimes H \). Then, \((G'_1 \otimes \ldots \otimes G'_j \otimes \ldots \otimes G'_k)[u/x][v/x] \cong (G_1 \otimes \ldots \otimes G_j \otimes \ldots \otimes G_k)[u/x][v/x] \otimes H\) by Proposition 9 and \( Oc(x)[y/x] \cap ty(H) = \emptyset\) (because \( BV([y/x][u/x]) \cap ty(H) = \emptyset\)).

Case \( X\bar{x} \leftarrow \varphi \in R^x\): Then, the derivation tree forms the following.

\[
\begin{array}{c}
H \models \phi' \\
\vdots \vdots \\
G \models \phi'[y/x] \\
G \models \phi' X\bar{y}
\end{array}
\]

By I.H., \( G \cong G' \otimes H \), hence this part has been proved.

**Proof of (4).** By induction on the structure of the derivation tree. Case (At): This case does not occur since the tree does not have any assumptions. Case (+): Then by using (3), the derivation tree forms the following left-hand side (note that by \((G_1 \otimes H) \otimes G_2 \cong (G_1 \otimes G_2) \otimes H\) (Proposition 9), \( G \cong G_1 \otimes G_2 \)). Then, by I.H. and applying (+), this part has been proved by the following right-hand side tree.

\[
\begin{array}{c}
H \models \phi' \\
\vdots \vdots \\
G \otimes H \models \phi' \otimes \phi_2
\end{array}
\]

\[
\begin{array}{c}
H' \models \phi' \\
\vdots \vdots \\
G \otimes H' \models \phi' \otimes \phi_2
\end{array}
\]
Case (3): Then by using (3), the derivation tree forms the following left-hand side (note that by \((G_1 \otimes H)[\mathbf{f}/x] \cong G_1[\mathbf{f}/x] \otimes H\) by Proposition 9 and \(x \notin \tau(H)\), \(G \cong G_1[\mathbf{f}/x]\)). Then, by I.H. and applying (3), this part has been proved by the following right-hand side tree.

\[
\frac{H \models y \psi}{H \models y \psi} \leftarrow \frac{G \otimes H \models y \varphi_1}{G \otimes H \models y \varphi_1 (3)}
\]

Case (\(\lor\)): Then by (3), the derivation tree forms the following left-hand side where \(l \in [2]\). Then, by I.H. and applying (\(\lor\)), this part has been proved by the following right-hand side tree.

\[
\frac{H \models y \psi}{H \models y \psi} \leftarrow \frac{G \otimes H \models y \varphi_1 \lor \varphi_2}{G \otimes H \models y \varphi_1 \lor \varphi_2 (\lor)}
\]

Case (TC): Then, by (3), the derivation tree forms the following left-hand side for some \(j\). Here, \((G_1 \odot_{G2} \cdots \odot_{G2} (G_j \odot_{G3} H) \odot_{G2} \cdots \odot_{G2} G_k)\)[\(\mathbf{u}/x\mathbf{f}/\mathbf{y}\)] \(\cong (G_1 \odot_{G2} \cdots \odot_{G2} G_j \odot_{G2} \cdots \odot_{G2} G_k)\)[\(\mathbf{u}/x\mathbf{f}/\mathbf{y}\)] \(\otimes H\) by Proposition 9 and \(\text{Occ}(x\mathbf{f}/y) \cap \tau(H) = \emptyset\). From this, \(G \cong (G_1 \odot_{G2} \cdots \odot_{G2} G_k)\)[\(\mathbf{u}/x\mathbf{f}/\mathbf{y}\)]. Then, by I.H. and applying (TC), this part has been proved by the following right-hand side tree.

\[
\frac{G_1 \models y \varphi}{G_1 \models y \varphi} \leftarrow \frac{G_j \otimes H \models y \varphi_1}{G_j \otimes H \models y \varphi_1 (\text{TC})} \quad \frac{G_k \models y \varphi}{G_k \models y \varphi} \leftarrow \frac{G \otimes H' \models y \varphi_1}{G \otimes H' \models y \varphi_1 (\text{TC})}
\]

Case \(X\mathbf{x} \leftarrow \varphi \in \mathcal{R}^F\): Then by (3), the derivation tree forms the following left-hand side. Then, by I.H. and applying the same rule, this part has been proved by the following right-hand side tree.

\[
\frac{H \models y \psi}{H \models y \psi} \leftarrow \frac{G \otimes H \models y \varphi[\mathbf{g}/\mathbf{x}]}{G \otimes H \models y \varphi[\mathbf{g}/\mathbf{x}]} \quad \frac{H' \models y \psi}{H' \models y \psi} \leftarrow \frac{G \otimes H' \models y \varphi[\mathbf{g}/\mathbf{x}]}{G \otimes H' \models y \varphi[\mathbf{g}/\mathbf{x}]}
\]