

# The complexity of the list homomorphism problem for graphs

László Egri, Andrei Krokhin, Benoit Larose, Pascal Tesson

April 2, 2009

## Abstract

We completely characterise the computational complexity of the list homomorphism problem for graphs in combinatorial and algebraic terms: for every graph  $\mathbf{H}$  the problem is either NP-complete, NL-complete, L-complete or is first-order definable; descriptive complexity equivalents are given as well via Datalog and its fragments. The central result relies on an inductive definition of graphs whose problem is solvable in Logspace. A characterisation by forbidden subgraphs is given as well, and as a consequence, the metaproblem can be decided in polynomial time.

## 1 Introduction

The complexity of graph homomorphism for a fixed graph template  $H$  (also known as  $H$ -coloring) can be viewed in the wider context of classifying the complexity of constraint satisfaction problems (CSP) or, equivalently, of the homomorphism problem for relational structures.

Let  $D$  be a finite domain: a *constraint language over  $D$*  is a set of finitary relations  $\mathbf{S} = \{R_1, \dots, R_k\}$  over the domain. An instance  $\mathcal{I}$  of the constraint satisfaction problem  $\text{CSP}(\mathbf{S})$  consists of a set of variables  $X = \{x_1, \dots, x_n\}$  and a list of constraints of the form  $(x_{i_1}, \dots, x_{i_j}) \in R_j$  and the objective is to determine whether the variables can be assigned values in  $D$  such that all constraints are satisfied. Alternatively, both  $\mathbf{S}$  and  $\mathcal{I}$  can be viewed as relational structures with a common signature  $\{R_1, \dots, R_k\}$  but over the domains  $D$  and  $X$  respectively. Now  $\mathcal{I}$  is a positive instance of  $\text{CSP}(\mathbf{S})$  iff there exists a homomorphism from  $\mathcal{I}$  to  $\mathbf{S}$ . When using this point of view, the relational structure  $\mathbf{S}$  is called a *template* (rather than a constraint language) and the corresponding computational problem is usually denoted  $\text{Hom}(\mathbf{S})$ .

The earliest results on the complexity of CSPs dates back to Schaefer [Sch78] who proved that, over the Boolean domain,  $\text{CSP}(\mathbf{S})$  is either NP-complete or solvable in polynomial time. A more systematic study of the non-Boolean cases was initiated by Feder and Vardi's seminal paper [FV98]. Their work established fundamental concepts, results and questions about the computational complexity of CSPs. In the last fifteen years, a series of results have established precise bounds on the computational complexity of  $\text{CSP}(\mathbf{S})$  for a number of important special cases. Algebra, logic and combinatorics provide three angles of attack which have fueled progress in this classification effort. Results relying on a combinatorial analysis of the templates include the work of Hell and Nešetřil showing that  $\text{Hom}(G)$  for an undirected graph  $G$  is tractable if  $G$  is bipartite and NP-complete otherwise [HN90], Feder and Vardi's analysis of CSPs with the "ability to count" and Feder, Hell and Huang's work on list  $H$ -coloring discussed below. The algebraic approach links the complexity of  $\text{CSP}(\mathbf{S})$  to the set of functions that preserve the relations in  $\mathbf{S}$  (see [CJ06] for a recent survey). In this framework, one associates to each  $\mathbf{S}$  an algebra  $\mathbb{A}_{\mathbf{S}}$  and exploits the fact that the properties

of  $\mathbb{A}_{\mathbf{S}}$  determine the complexity of  $\text{CSP}(\mathbf{S})$ . This angle of attack was crucial for establishing the dichotomy conjecture for the three-element domain [B02] and for conservative constraints [B03]. Finally, the logical approach seeks to understand the descriptive complexity of  $\text{CSP}(\mathbf{S})$ , i.e. to understand how powerful a logic is needed to define the set of relational structures  $\text{Hom}(\mathbf{S})$ ? This body of work has focused on the database-inspired fixpoint logic Datalog and its fragments. Feder and Vardi were first to note that a number of polynomial-time cases of  $\text{CSP}(\mathbf{S})$  correspond to structures for which  $\neg \text{CSP}(\mathbf{S})$  is definable in Datalog. More recently, close ties were found between the two fragments of Datalog known as linear Datalog and symmetric Datalog and the structures for which  $\neg \text{CSP}(\mathbf{S})$  is computable in NL and L respectively [FV98, D05, DK08, ELT07, ELT08].

Recent work has uncovered fundamental connections between these three points of view: Bulatov [B04] has given an algebraic proof of the dichotomy of Hell and Nešetřil [B04], Barto and Kozik [BK09] have obtained an algebraic characterization of the class of  $\mathbf{S}$  for which  $\neg \text{CSP}(\mathbf{S})$  is in Datalog, Larose, Valeriote and Zádori [LVZ09] an algebraic equivalent of the ability to count. Also known are necessary and sufficient conditions on the combinatorial structure of  $\mathbf{S}$  for expressibility of  $\neg \text{CSP}(\mathbf{S})$  in Datalog or its linear fragment (bounded width and bounded pathwidth duality respectively.)

To have a completely satisfactory classification of the complexity of  $\text{CSP}(\mathbf{S})$ , we ideally want not only to distinguish tractable cases from NP-complete cases but to also identify cases which are, say, P-complete, NL-complete, L-complete and so on. This can be done in the Boolean case:  $\text{CSP}(\mathbf{S})$  is either NP-complete, P-complete, NL-complete,  $\oplus$ L-complete, L-complete or in co-NLOGTIME [ABISV05]. In the same spirit, we also seek necessary and sufficient conditions on  $\mathbf{S}$  that ensure that  $\neg \text{CSP}(\mathbf{S})$  is expressible in the linear or symmetric fragments of Datalog. It was shown in [LT09] that, with some fine-tuning, these questions can also be approached through the aforementioned algebraic angle. In particular, one can obtain criteria on the variety  $\mathcal{V}(\mathbb{A}_{\mathbf{S}})$  generated by the algebra  $\mathbb{A}_{\mathbf{S}}$  which are sufficient for NL-hardness,  $\text{Mod}_p$ L-hardness, P-hardness and NP-hardness of  $\text{CSP}(\mathbf{S})$  as well as for inexpressibility in Datalog, linear Datalog and symmetric Datalog of  $\neg \text{CSP}(\mathbf{S})$  [LT09, CJ06, ELT07]. These rest on the notion of *types*, a central notion in the study of finite algebras which we describe in some more detail in Section 2. Roughly speaking, each variety  $\mathcal{V}$  either *admits* or *omits* types 1 through 5 and the typeset of  $\mathcal{V} = \mathcal{V}(\mathbb{A}_{\mathbf{S}})$  yields crucial information on the complexity of  $\text{CSP}(\mathbf{S})$ . In particular, if  $\mathcal{V}$  admits type 1 then  $\text{CSP}(\mathbf{S})$  is NP-complete and [BKJ00] conjectured that the problem is otherwise tractable.

While that conjecture is still open, the picture is increasingly clear when  $\mathcal{V}$  omits both types 1 and 2: the result of Barto and Kozik shows that  $\neg \text{CSP}(\mathbf{S})$  is expressible in Datalog iff  $\mathcal{V}$  omits types 1 and 2; if  $\mathcal{V}$  omits types 1 and 2 but admits type 5 then  $\text{CSP}(\mathbf{S})$  is P-complete and  $\neg \text{CSP}(\mathbf{S})$  cannot be defined in linear Datalog [LT09]. If  $\mathcal{V}$  omits types 1, 2 and 5 but admits type 4, then  $\text{CSP}(\mathbf{S})$  is NL-hard and  $\neg \text{CSP}(\mathbf{S})$  cannot be defined in symmetric Datalog [LT09]. It is tempting to conjecture that omitting types 1, 2 and 5 is sufficient for expressibility in linear Datalog and that omitting types 1, 2, 4 and 5 is sufficient for expressibility in symmetric Datalog. In fact, this does hold in the Boolean case and the aim of the present paper is to show that it also holds in the special case of list  $H$ -coloring for undirected graphs.

For a graph  $H$ , a graph with  $H$ -lists  $G$  is a graph in which each vertex  $v \in G$  is assigned a list  $L_v \subseteq H$ . The list  $H$ -coloring problem consists of determining whether there is a homomorphism  $h$  from  $G$  to  $H$  such that  $h(v) \in L_v$  for all  $v$ . We can view the problem as a CSP where the template is the structure  $\mathbf{H}^L$  consisting of the binary relation on  $H$  and all unary relations on  $H$  (i.e. every subset of  $H$ ). We study the complexity of  $\text{CSP}(\mathbf{H}^L)$  for undirected graphs  $H$ . Hell, Feder and Huang [FHH99] established that this problem is always either tractable or NP-complete, where the tractable cases are exactly those admitting a majority polymorphism which, in turn, implies that in these cases we in fact have  $\text{CSP}(\mathbf{H}^L)$  is in NL and, more precisely,  $\neg \text{CSP} \mathbf{H}^L$  definable in linear

Datalog [DK08]. We complete the picture by further classifying the tractable cases and giving explicit descriptions of the types admitted or omitted by the varieties  $\mathcal{V} = \mathcal{V}(\mathbb{A}_{\mathbf{H}^L})$ . We prove that  $\mathcal{V}$  either:

- admits type 1, in which case  $\text{CSP}(\mathbf{H}^L)$  is NP-complete and  $\neg \text{CSP}(\mathbf{H}^L)$  is not definable in Datalog;
- omits type 1, 2 and 5 but admits type 4, in which case  $\text{CSP}(\mathbf{H}^L)$  is NL-complete and  $\neg \text{CSP}(\mathbf{H}^L)$  is definable in linear Datalog but not in symmetric Datalog;
- admits only type 3, in which case  $\text{CSP}(\mathbf{H}^L)$  is either first-order definable or L-complete and  $\neg \text{CSP}(\mathbf{H}^L)$  is definable in symmetric Datalog.

We give two alternative descriptions of the class  $\mathcal{L}$  of graphs such that  $\mathcal{V}$  admits only type 3: the first is in terms of forbidden induced subgraphs while the second is an inductive definition. The latter description allows us to show that in this case the variety  $\mathcal{V}$  is in fact 4-permutable.

Because of space constraints, most of the proofs and the entire discussion about expressibility in Datalog and its fragments appear in the appendix.

## 2 Preliminaries

In the following we denote the underlying universe of a structure  $\mathbf{S}, \mathbf{T}, \dots$  by its roman equivalent  $S, T, \dots$ . Let  $\mathbf{T}$  be a relational structure of signature  $\sigma$ ; for each relational symbol  $R \in \sigma$  we denote the corresponding relation of  $\mathbf{T}$  by  $R(\mathbf{T})$ . Let  $\mathbf{S}$  be another structure of the same signature. A *homomorphism* from  $\mathbf{S}$  to  $\mathbf{T}$  is a map  $f$  from  $S$  to  $T$  such that  $f(R(\mathbf{S})) \subseteq R(\mathbf{T})$  for each  $R \in \sigma$ . We denote by  $\text{CSP}(\mathbf{T})$  the set of all structures  $\mathbf{S}$  that admit a homomorphism to  $\mathbf{T}$ .

For the purposes of this paper, a *graph* is a relational structure  $\mathbf{H} = \langle H; \theta \rangle$  where  $\theta$  is a symmetric, binary relation on  $H$ . The graph  $\mathbf{H}$  is *reflexive* (*irreflexive*) if  $(x, x) \in \theta$  ( $(x, x) \notin \theta$ ) for all  $x \in H$ .

Given a graph  $\mathbf{H}$ , let  $S_1, \dots, S_k$  denote all non-empty subsets of  $H$ ; let  $\mathbf{H}^L$  be the relational structure obtained from  $\mathbf{H}$  by adding all the  $S_i$  as unary relations; more precisely, let  $\sigma'$  be the signature that consists of one binary relational symbol  $\theta$  and unary symbols  $R_i, i = 1, \dots, k$ . The  $\sigma'$ -structure  $\mathbf{H}^L$  has universe  $H$ ,  $\theta(\mathbf{H}^L)$  is the edge relation of  $\mathbf{H}$ , and  $R_i(\mathbf{H}^L) = S_i$  for all  $i = 1, \dots, k$ . We call  $\text{CSP}(\mathbf{H}^L)$  the *list homomorphism problem for  $\mathbf{H}$* .

An  $n$ -ary operation on a set  $A$  is a map  $f : A^n \rightarrow A$ . Let  $I$  be a signature, i.e. a set of operation symbols  $f$  each of a fixed arity. An (indexed) *algebra* of signature  $I$  is a structure  $\mathbb{A} = \langle A; f^{\mathbb{A}} : f \in I \rangle$  where  $A$  is a non-empty set (the *universe of  $A$* ), and for each  $f \in I$ ,  $f^{\mathbb{A}}$  is an operation on  $A$  of the corresponding arity. A class of similar algebras which is closed under formation of homomorphic images, subalgebras and products is called a *variety*. The *variety generated* by an algebra  $\mathbb{A}$  is denoted by  $\mathcal{V}(\mathbb{A})$ , and is the smallest variety containing  $\mathbb{A}$ , i.e. the class of all homomorphic images of subalgebras of powers of  $\mathbb{A}$ . For a fixed signature, terms are defined in the usual way; the *term operations* of an algebra are all the operations obtained by interpreting the terms in that algebra (or equivalently, those operations built from the basic operations and projections of the algebra using composition.)

Given an  $h$ -ary relation  $\theta$  and an  $n$ -ary operation on the same set  $A$ , we say that  $f$  *preserves*  $\theta$  or that  $\theta$  is *invariant* under  $f$  if the following holds: given any matrix  $M$  of size  $h \times n$  whose columns are in  $\theta$ , applying  $f$  to the rows of  $M$  will produce an  $h$ -tuple in  $\theta$ .

Given a graph  $\mathbf{H}$ , we let  $\mathbb{H}$  denote the algebra associated with  $\text{CSP}(\mathbf{H}^L)$ , i.e.  $\mathbb{H}$  has universe  $H$  and its basic (term) operations are all operations on  $H$  that preserve the edge relation of  $\mathbf{H}$  and all subsets of  $H$ . An operation on a set  $H$  that preserves all subsets of  $H$  is said to be *conservative*. We say that the graph  $\mathbf{H}$  *admits* the conservative operation  $f$  if  $f$  is a term operation of the associated algebra. We say the terms  $s, t$  satisfy the identity  $s \approx t$  in  $\mathcal{V}$  if  $s(\bar{x}) = t(\bar{x})$  for all  $\bar{x}$  when interpreted

in every algebra of  $\mathcal{V}$ .

Tame Congruence Theory, as developed in [HM88], is a powerful tool for the analysis of finite algebras. Every finite algebra has a *typeset*, which describes the local behaviour of the algebra. It contains one or more of the following 5 *types*: (1) the *unary* type, (2) the *affine* type, (3) the *Boolean* type, (4) the *lattice* type and (5) the *semilattice* type. Simple algebras, i.e. algebras with no non-trivial proper homomorphic images, admit a unique type; the prototypical examples are: the 2-element algebra with no basic operations  $\langle\{0, 1\}; \emptyset\rangle$  has type 1. The 2 element semilattices are the 2-element algebras with a single binary operation  $\langle\{0, 1\}; \wedge\rangle$  and  $\langle\{0, 1\}; \vee\rangle$ : they have type 5. The 2 element lattice is the 2 element algebra with two binary operations  $\langle\{0, 1\}; \vee, \wedge\rangle$ : it has type 4. An algebra is *affine* if there is an abelian group structure on its base set such that (i)  $m(x, y, z) = x - y + z$  is a term of the algebra and (ii) every term of the algebra is affine, i.e. commutes with the operation  $m$ . Equivalently, an idempotent algebra is affine iff it is the full idempotent reduct of a module. An affine algebra has type 2. The 2-element Boolean algebra has type 3.

Roughly speaking, the typeset of a (locally finite) variety indicates the presence of algebras in the variety which behave locally as the above examples. For our purposes here, it will not be necessary to delve further in the technical aspects of typesets. Indeed, there is a very tight connection between the kind of equations that are satisfied by the algebras in a variety and the types that are admitted or omitted by a variety, i.e. those types that do or do not appear in the typeset of some algebra in the variety.

The following result gathers some of the algebraic machinery we will require later on. These results can be found in [HM88]. A 3-ary operation  $m$  is a *majority* operation if it satisfies the identities  $m(x, x, y) \approx m(x, y, x) \approx m(y, x, x) \approx x$ . Let  $n \geq 1$ . A variety is  $n + 1$ -*permutable* if and only if it has terms  $f_1, \dots, f_n$  satisfying the following identities:

$$x \approx f_1(x, y, y) \tag{1}$$

$$f_i(x, x, y) \approx f_{i+1}(x, y, y) \text{ for all } i \tag{2}$$

$$f_n(x, x, y) \approx y. \tag{3}$$

**Lemma 1** [HM88] *A variety  $\mathcal{V}$  is  $n$ -permutable for some  $n \geq 2$  if and only if  $\mathcal{V}$  omits types 1, 4 and 5; if a variety admits a majority term then it omits types 1, 2 and 5.*

In [FHH99], a dichotomy result was proved, identifying the class of bi-arc graphs as those whose list homomorphism problem is tractable, and others as being NP-complete. Let  $C$  be a circle with two specified points  $p$  and  $q$ . A bi-arc is a pair of arcs  $(N, S)$  such that  $N$  contains  $p$  but not  $q$  and  $S$  contains  $q$  but not  $p$ . A graph  $\mathbf{H}$  is a *bi-arc graph* if there is a family of bi-arcs  $\{(N_x, S_x) : x \in H\}$  such that, for every  $x, y \in H$ , the following hold: (i) if  $x$  and  $y$  are adjacent, then neither  $N_x$  intersects  $S_y$  nor  $N_y$  intersects  $S_x$ , and (ii) if  $x$  is not adjacent to  $y$  then both  $N_x$  intersects  $S_y$  and  $N_y$  intersects  $S_x$ . The following sharpens the dichotomy result of [FHH99].

**Lemma 2** *Let  $\mathbf{H}$  be a graph. Then the following conditions are equivalent:*

1. *the variety  $\mathcal{V}(\mathbf{H})$  omits type 1;*
2. *the graph  $\mathbf{H}$  admits a conservative majority operation;*
3. *the graph  $\mathbf{H}$  is a bi-arc graph.*

### 3 Irreflexive graphs

In this section all graphs considered are irreflexive, i.e. without loops. We introduce some terminology and notation. Let  $\mathbf{H}_1$  and  $\mathbf{H}_2$  be bipartite graphs, with colour classes  $B_1, T_1$  and  $B_2$  and  $T_2$  respectively, with  $T_1$  and  $B_2$  non-empty. We define the *special sum*  $\mathbf{H}_1 \odot \mathbf{H}_2$  (which depends on the choice of the  $B_i$  and  $T_i$ ) as follows: it is the graph obtained from the disjoint union of  $\mathbf{H}_1$  and  $\mathbf{H}_2$  by adding all possible edges between the vertices in  $T_1$  and  $B_2$ . Notice that we can often decompose a bipartite graph several ways, and even choose  $B_1$  or  $T_2$  to be empty. We'll say that a graph  $\mathbf{H}$  is a *special sum* or *expressed as a special sum* if there exist two bipartite graphs and a choice of ordering on both such that  $\mathbf{H}$  is isomorphic to the special sum of these two graphs.

**Definition 3** *Let  $\mathcal{K}$  denote the smallest class of irreflexive graphs containing the one-element graph and closed under (i) special sum and (ii) disjoint union. We'll call the graphs in  $\mathcal{K}$  basic.*

The following result gives a characterisation of basic irreflexive graphs in terms of forbidden subgraphs; its proof can be found in the Appendix.

**Lemma 4** *Let  $\mathbf{H}$  be an irreflexive graph. Then the following conditions are equivalent:*

1.  $\mathbf{H}$  is basic;
2.  $\mathbf{H}$  is bipartite, contains no induced 6-cycle, nor any induced path of length 5.

**Theorem 5** *Let  $\mathbf{H}$  be an irreflexive graph. Then the following conditions are equivalent:*

1.  $\mathcal{V}(\mathbb{H})$  is 4-permutable;
2.  $\mathcal{V}(\mathbb{H})$  is  $n$ -permutable for some  $n$ ;
3.  $\text{typ}(\mathcal{V}(\mathbb{H})) = \{3\}$ ;
4.  $\mathbf{H}$  is basic.

*Proof [sketch]:* (1) implies (2) is trivial. If (2) holds then by Lemma 1  $\mathcal{V}(\mathbb{H})$  omits type 1, and by Lemma 2 it implies that  $\mathbf{H}$  admits a majority operation so  $\mathcal{V}(\mathbb{H})$  omits type 2 also by Lemma 1; hence (3) holds. (3) implies (4) is the content of Lemma 14 in the Appendix, and (4) implies (1) by Lemma 6 that follows.  $\blacksquare$

**Lemma 6** *If  $\mathbf{H}$  is basic then  $\mathcal{V}(\mathbb{H})$  is 4-permutable.*

*Proof:* We shall show by induction on the size of  $\mathbf{H}$  that there exist conservative operations  $f_1, f_2, f_3$  preserving the graph  $\mathbf{H}$ , obeying the identities (1), (2), (3), and furthermore that satisfy the following condition (D):

*For every  $x, y, z, n, m \in H$  such that  $n$  is adjacent to  $x$  and  $m$  is adjacent to  $z$ ,  $f_1(x, y, z)$  is adjacent to  $n$  and  $f_3(x, y, z)$  is adjacent to  $m$ .*

The result is trivial for a one-element graph. If  $\mathbf{H}$  is not connected, then  $\mathbf{H}$  is the disjoint union of proper subgraphs  $\mathbf{H}_1$  and  $\mathbf{H}_2$ . Let  $f_1, f_2, f_3$  and  $g_1, g_2, g_3$  be the desired operations on  $\mathbf{H}_1$  and  $\mathbf{H}_2$  respectively; we define operations  $h_1, h_2, h_3$  on  $\mathbf{H}$  as follows:

For every  $1 \leq s \leq 3$ , let  $h_s(x, y, z) = f_s(x, y, z)$  if  $(x, y, z) \in H_1^3$  and let  $h_s(x, y, z) = g_s(x, y, z)$  if  $(x, y, z) \in H_2^3$ ; if  $(x, y, z) \in H_i \times H_j \times H_k$  with  $i, j, k$  not all equal, then let  $h_1(x, y, z) = x$  and  $h_3(x, y, z) = z$ , and finally let  $h_2(x, y, z) = z$  if  $(i, j, k) \in \{(1, 1, 2), (2, 2, 1)\}$  and let  $h_2(x, y, z) = x$  otherwise.

It is immediate that identities (1) and (3) are satisfied and that each  $h_s$  is a conservative homomorphism. For (2): we may assume that  $x \neq y$ ; if  $x$  and  $y$  are in the same  $H_i$  then (2) follows from the fact that the  $f_i$  and  $g_i$  satisfy it; otherwise we have that  $h_1(x, x, y) = x = h_2(x, y, y)$  and  $h_2(x, x, y) = y = h_3(x, y, y)$ . It is easy to see that condition (D) is satisfied by  $h_1$  and  $h_3$ .

Now suppose that the basic graph  $\mathbf{H}$  is connected, and hence is the special sum of two smaller graphs. For the moment, it will be convenient to denote the colour classes of  $\mathbf{H}$  by  $C_1$  and  $C_2$ ; our first task is to show it suffices to define our operations on  $C_1^3 \cup C_2^3$ . Indeed, suppose that we have functions  $f_1, f_2, f_3 : C_1^3 \cup C_2^3 \rightarrow H$  that satisfy all the required identities, are edge-preserving and conservative. Then we may extend these to full operations  $F_1, F_2, F_3 : H^3 \rightarrow H$  as follows: let

$$F_1(x, y, z) = \begin{cases} f_1(x, y, z), & \text{if } (x, y, z) \in C_1^3 \cup C_2^3; \\ x, & \text{otherwise.} \end{cases}$$

$$F_3(x, y, z) = \begin{cases} f_3(x, y, z), & \text{if } (x, y, z) \in C_1^3 \cup C_2^3; \\ z, & \text{otherwise.} \end{cases}$$

$$F_2(x, y, z) = \begin{cases} f_2(x, y, z), & \text{if } (x, y, z) \in C_1^3 \cup C_2^3; \\ z, & \text{if } (x, y, z) \in C_i \times C_i \times C_j \text{ for some } i \neq j; \\ x, & \text{otherwise.} \end{cases}$$

Notice that distinct sets  $C_i \times C_j \times C_k$  and  $C_{i'} \times C_{j'} \times C_{k'}$  are in different connected components of  $\mathbf{H}^3$ , unless  $i \neq i', j \neq j'$  and  $k \neq k'$ ; it follows immediately that the  $F_i$  are edge-preserving; they are also clearly conservative. It is a simple matter to verify that all the required identities are satisfied. Hence, from now on, we assume without mention that in all triples  $(x, y, z)$  considered all the entries come from the same colour class of the graph under consideration.

So let  $\mathbf{H}$  be the special sum of two smaller graphs  $\mathbf{H}_i$  with colour classes  $B_i$  and  $T_i$ ,  $i = 1, 2$ ; by induction hypothesis  $\mathbf{H}_1$  admits the required operations  $f_1, f_2, f_3$  and  $\mathbf{H}_2$  admits operations  $g_1, g_2, g_3$  satisfying the necessary conditions. We define operations  $F_1, F_2, F_3$  on  $\mathbf{H}$  as follows. For convenience, let  $S = T_1 \cup B_2$ . Notice that by definition of special sum  $S$  is a complete bipartite graph.

$$F_1(x, y, z) = \begin{cases} f_1(x, y, z), & \text{if } x, y, z \in H_1, \text{ else} \\ g_1(x, y, z), & \text{if } x, y, z \in H_2, \text{ else} \\ x, & \text{if } y = z \text{ or } x \in S, \text{ else} \\ u, & \text{where } u \in \{y, z\} \cap S. \end{cases}$$

$$F_3(x, y, z) = \begin{cases} f_3(x, y, z), & \text{if } x, y, z \in H_1, \text{ else} \\ g_3(x, y, z), & \text{if } x, y, z \in H_2, \text{ else} \\ z, & \text{if } x = y \text{ or } z \in S, \text{ else} \\ v, & \text{where } v \in \{x, y\} \cap S. \end{cases}$$

$$F_2(x, y, z) = \begin{cases} F_1(x, x, z), & \text{if } y = z, \text{ else} \\ F_3(x, z, z), & \text{if } x = y, \text{ else} \\ f_2(x, y, z), & \text{if } x, y, z \in H_1, \text{ else} \\ g_2(x, y, z), & \text{if } x, y, z \in H_2, \text{ else} \\ w, & \text{where } w \in \{x, y, z\} \cap S. \end{cases}$$

Obviously all three operations are conservative, and by definition they obey all the required identities. Now we verify that  $F_1$  satisfies condition (D): let  $(x, n)$  be an edge of  $\mathbf{H}$ : we show that  $F_1(x, y, z)$  is adjacent to  $n$ . If  $x, y, z \in H_i$  for some  $i = 1, 2$  then this follows by induction hypothesis, and it is clearly true if  $F_1(x, y, z) = x$ . Otherwise,  $F_1(x, y, z) = u$  for some  $u \in S$ ; if  $x, y, z \in B$  then  $x \in B_1$  so  $n \in T_1$  is adjacent to  $u$ . Otherwise  $x, y, z \in T$ , so  $x \in T_2$  hence  $n \in B_2$  is adjacent to  $u$ . The proof that  $F_3$  satisfies (D) is identical. It remains to show that each  $F_i$  is edge-preserving. We start with  $F_1$ : let  $(x, y, z)$  be adjacent to  $(x', y', z')$  and suppose without loss of generality that  $x, y, z \in B$  and  $x', y', z' \in T$ . If  $x, y, z \in B_1$  then  $x', y', z' \in T_1$  and hence  $F_1$  coincides with  $f_1$  on both tuples and we are done by induction hypothesis. If  $F_1(x, y, z) = x$  then by (D) we have  $F_1(x', y', z')$  adjacent to  $x$ . Otherwise, we have that  $x \in B_1$  (and thus  $x' \in T_1$ ) and  $F_1(x, y, z) = u \in B_2$ ; in any case  $F_1(x', y', z') \in T_1$  so it is adjacent to  $u$ . The argument for  $F_3$  is identical. Now we consider  $F_2$ : let  $(x, y, z)$  be adjacent to  $(x', y', z')$ , where  $x, y, z \in B$  and  $x', y', z' \in T$ . Notice that by induction hypothesis and definition of the  $F_i$ , we have that  $F_2$  coincides with  $f_2$  ( $g_2$ ) on tuples whose coordinates all lie in  $H_1$  (respectively  $H_2$ ). If  $x, y, z \in B_1$ , then certainly  $x', y', z' \in T_1$ , and then the result follows by induction hypothesis and the last remark. Now we require the following claim:

**Claim.** Suppose that  $a, b, c$  do not all lie in the same  $H_i$ . If  $b = c$  or  $a = b$  then  $F_2(a, b, c) \in S$ .

*Proof of Claim.* Suppose that  $b = c$ , so that  $F_2(a, b, c) = F_1(a, a, c)$ . By hypothesis  $a$  and  $c$  do not lie in the same  $H_i$ , and in particular they are distinct, hence by definition of  $F_1$  we have that  $F_1(a, a, c) = a$  if  $a \in S$  or  $F_1(a, a, c) = u$  for some  $u \in S$ . The proof for the case  $a = b$  is identical.

Now we can finish the proof. Suppose first that  $x, y, z$  are not all in the same  $H_i$ ; by the claim  $F_2(x, y, z) \in S$ . If  $x', y', z'$  are not all in the same  $H_i$  then  $F_2(x', y', z') \in S$  also and we're done. Otherwise,  $x', y', z'$  all lie in  $T_1$  (since one of them is a neighbour of an element of  $B_1$ ) and hence  $F_2(x', y', z') = f_2(x', y', z') \in S$  and we're done. Now suppose that  $x, y, z$  are all in  $B_2$  (we dealt with the case  $B_1$  earlier.) Then  $F_2(x, y, z) = g_2(x, y, z) \in S$ , so if  $x', y', z'$  are not all in the same  $H_i$  we are done by the claim again. Otherwise either  $x', y', z' \in T_1$  so  $F_2(x', y', z') = f_2(x', y', z') \in S$ , or else  $x', y', z' \in T_2$ : then  $F_2(x', y', z') = g_2(x', y', z')$  and we are done by induction hypothesis. ■

## 4 The general case

We shall describe a family of graphs which is precisely the family of graphs whose list homomorphism problem is in symmetric Datalog. Our family of graphs is first defined by forbidden induced subgraphs:

**Definition 7** Define the class  $\mathcal{L}$  of graphs as follows: a graph  $\mathbf{H}$  belongs to  $\mathcal{L}$  if it contains none of the following as an induced subgraph:

1. the reflexive path of length 3 and the reflexive 4-cycle;
2. the irreflexive odd cycles, irreflexive path of length 5 and irreflexive 6-cycle;
3. **B1**, **B2**, **B3**, **B4**, **B5** and **B6** (see Figure 9.)

As in the irreflexive case, in order to prove our result we'll need to describe the graphs in  $\mathcal{L}$  in an inductive manner.

**Definition 8** A connected graph  $\mathbf{H}$  is basic if either (i)  $\mathbf{H}$  is a single loop, or (ii)  $\mathbf{H}$  is a basic irreflexive graph, or (iii)  $\mathbf{H}$  is obtained from a basic irreflexive graph  $\mathbf{H}_1$  with colour classes  $B$  and  $T$  by adding every edge (including loops) of the form  $\{t, t'\}$  where  $t, t' \in T$ .

**Definition 9** Given two vertex-disjoint graphs  $\mathbf{H}_1$  and  $\mathbf{H}_2$ , the adjunction of  $\mathbf{H}_1$  to  $\mathbf{H}_2$  is the graph  $\mathbf{H}_1 \odot \mathbf{H}_2$  obtained by taking the disjoint union of the two graphs, and adding every edge of the form  $\{x, y\}$  where  $x$  is a loop in  $\mathbf{H}_1$  and  $y$  is a vertex of  $\mathbf{H}_2$ .

**Theorem 10** The class  $\mathcal{L}$  is the smallest class  $\mathcal{C}$  of graphs such that:

1.  $\mathcal{C}$  contains the basic graphs;
2.  $\mathcal{C}$  is closed under disjoint union;
3. if  $\mathbf{H}_1$  is a basic graph and  $\mathbf{H}_2 \in \mathcal{C}$  then  $\mathbf{H}_1 \odot \mathbf{H}_2 \in \mathcal{C}$ .

*Proof:* We start by showing that every basic graph is in  $\mathcal{L}$ , i.e. that a basic graph does not contain any of the forbidden graphs. If  $\mathbf{H}$  is a single loop or a basic irreflexive graph, then this is immediate. Otherwise  $\mathbf{H}$  is obtained from a basic irreflexive graph  $\mathbf{H}_1$  with colour classes  $B$  and  $T$  by adding every edge of the form  $(t_1, t_2)$  where  $t_i \in T$ . In particular, the loops form a clique and no edge connects two non-loops; it is clear in that case that  $\mathbf{H}$  contains none of **B1**, **B2**, **B3**, **B4**. On the other hand if  $\mathbf{H}$  contains **B5** or **B6**, then  $\mathbf{H}_1$  contains the path of length 5 or the 6-cycle, contradicting the fact that  $\mathbf{H}_1$  is basic.

Next we show that  $\mathcal{L}$  is closed under disjoint union and adjunction of basic graphs. It is obvious that the disjoint union of graphs that avoid the forbidden graphs will also avoid these. So suppose that an adjunction  $\mathbf{H}_1 \odot \mathbf{H}_2$ , where  $\mathbf{H}_1$  is a basic graph, contains an induced forbidden graph  $\mathbf{B}$  whose vertices are neither all in  $H_1$  nor  $H_2$ ; without loss of generality  $H_1$  contains at least one loop, its loops form a clique and none of its edges connects two non-loops. It is then easy to verify that  $\mathbf{B}$  contains both loops and non-loops. Because the other cases are similar, we prove only that  $\mathbf{B}$  is not **B3**: since vertex  $d$  is not adjacent to  $a$  it must be in  $\mathbf{H}_2$ , and similarly for  $c$ . Since  $b$  is not adjacent to  $d$  it must also be in  $\mathbf{H}_2$ ; since non-loops of  $\mathbf{H}_1$  are not adjacent to elements of  $\mathbf{H}_2$  it follows that  $a$  is in  $\mathbf{H}_2$  also, a contradiction.

Now we must show that every graph in  $\mathcal{L}$  can be obtained from the basic graphs by disjoint union and adjunction of basic graphs. Suppose this is not the case. If  $\mathbf{H}$  is a counterexample of minimum size, then obviously it is connected, and it contains at least one loop for otherwise it is a basic irreflexive graph. By Lemma 15 in the Appendix  $\mathbf{H}$  also contains at least one non-loop.

For  $a \in H$  let  $N(a)$  denote its set of neighbours. Let  $\mathbf{R}(\mathbf{H})$  denote the subgraph of  $\mathbf{H}$  induced by its set  $R(H)$  of loops, and let  $\mathbf{J}(\mathbf{H})$  denote the subgraph induced by  $J(H)$ , the set of non-loops of  $\mathbf{H}$ . Since  $\mathbf{H}$  is connected and neither **B1** nor **B2** is an induced subgraph of  $\mathbf{H}$ , the graph  $\mathbf{R}(\mathbf{H})$  is also connected, and furthermore every vertex in  $J(H)$  is adjacent to some vertex in  $R(H)$ . By the above we know that  $\mathbf{R}(\mathbf{H})$  contains at least one universal vertex: let  $U$  denote the (non-empty) set of universal vertices of  $\mathbf{R}(\mathbf{H})$ . Let  $J$  denote the set of vertices  $a \in J(H)$  such that  $N(a) \cap R(H) \subseteq U$ , and let  $\mathbf{S}$  denote the subgraph of  $\mathbf{H}$  induced by  $U \cup J$ . The graph  $\mathbf{S}$  is connected. We claim that the following properties also hold:

1. if  $a$  and  $b$  are adjacent non-loops, then  $N(a) \cap U = N(b) \cap U$ ;
2. if  $a$  is in a connected component of the subgraph of  $\mathbf{S}$  induced by  $J$  with more than one vertex, then for any other  $b \in J$ , one of  $N(a) \cap U, N(b) \cap U$  contains the other.

The first statement holds because **B1** is forbidden, and the second follows from the first because **B4** is also forbidden. Let  $J_1, \dots, J_k$  denote the different connected components of  $J$  in  $\mathbf{S}$ . By (1) we may let  $N(J_i)$  denote the set of common neighbours of members of  $J_i$  in  $U$ . By (2), there exist indices, which we choose to be  $1, \dots, m \leq s$  without loss of generality, such that  $N(J_i) \subseteq N(J_j)$  for all  $i \leq m$  and all  $j > m$ . Let  $\mathbf{B}$  denote the subgraph of  $\mathbf{S}$  induced by  $B = \bigcup_{i=1}^m J_i \cup N(J_i)$ , and let  $\mathbf{C}$  be the subgraph of  $\mathbf{H}$  induced by  $H \setminus B$ . We claim that  $\mathbf{H} = \mathbf{B} \circledast \mathbf{C}$ . For this, it suffices to show that every element in  $\bigcup_{i=1}^m N(J_i)$  is adjacent to every non-loop  $c \in C$ . By construction this holds if  $c \in J \cap C$ . Now suppose this does not hold: then some  $x \in J(H) \setminus J$  is not adjacent to some  $y \in N(J_i)$  for some  $i \leq m$ . Since  $x \notin J$  we may find some  $z \in R(H) \setminus U$  adjacent to  $x$ ; it is of course also adjacent to  $y$ . Since  $z \notin U$  there exists some  $z' \in R(H) \setminus U$  that is not adjacent to  $z$ , but it is of course adjacent to  $y$ . If  $x$  is adjacent to  $z'$ , then  $\{x, z, z'\}$  induces a subgraph isomorphic to **B2**, a contradiction. Otherwise,  $\{x, z, y, z'\}$  induces a subgraph isomorphic to **B3**, also a contradiction.

By (2), either  $m = 1$  or every  $J_i$  with  $i \leq m$  contains a single element. In the second case, notice that  $\mathbf{B}$  is a basic graph: indeed, removing all edges between its loops yields a bipartite irreflexive graph which contains neither the path of length 5 nor the 6-cycle, since  $\mathbf{B}$  contains neither **B5** nor **B6**. Since this contradicts our hypothesis on  $\mathbf{H}$ , we conclude that  $m = 1$ . But this means that  $N(J_1)$  is a set of universal vertices in  $\mathbf{H}$ . Let  $u$  be such a vertex and let  $D$  denote its complement in  $\mathbf{H}$ : clearly  $\mathbf{H}$  is obtained as the adjunction of the single loop  $u$  to  $D$ , contradicting our hypothesis. This concludes the proof. ■

We can now state our main result:

**Theorem 11** *Let  $\mathbf{H}$  be a graph. Then the following conditions are equivalent:*

1.  $\mathcal{V}(\mathbb{H})$  is 4-permutable;
2.  $\mathcal{V}(\mathbb{H})$  is  $n$ -permutable for some  $n$ ;
3.  $\text{typ}(\mathcal{V}(\mathbb{H})) = \{3\}$ ;
4.  $\mathbf{H} \in \mathcal{L}$ ;
5.  $\neg\text{CSP}(\mathbf{H}^L)$  is in symmetric Datalog.

*If the above holds then the list homomorphism problem for  $\mathbf{H}$  is in Logspace. Otherwise, either:*

- *$\text{typ}(\mathcal{V}(\mathbb{H}))$  admits type 1, and then  $\neg\text{CSP}(\mathbf{H}^L)$  is not expressible in Datalog and  $\text{CSP}(\mathbf{H}^L)$  is NP-complete (under FO reductions); or*
- *$\text{typ}(\mathcal{V}(\mathbb{H}))$  omits type 1 but admits type 4; in that case  $\neg\text{CSP}(\mathbf{H}^L)$  is not expressible in symmetric Datalog but is expressible in linear Datalog, and  $\text{CSP}(\mathbf{H}^L)$  is NL-complete (under FO reductions.)*

*Proof [sketch]:* The arguments for (1)  $\Rightarrow$  (2)  $\Rightarrow$  (3) are identical to those in Theorem 5. (3) implies (4) is the content of Lemma 16 in the Appendix; the proof of (4) implies (1) is very similar to the proof of Lemma 6 and its proof can be found in Lemma 18 in the Appendix. (4) implies (5) is the content of Lemma 19 in the Appendix, and (5) implies (3) is a result of [LT09]. Now suppose that the variety admits type 1; the statement follows from results in [BKJ00] and [FV98]. If the variety

omits type 1, then the structure admits a majority operation by Lemma 2 and then  $\neg CSP(\mathbf{H}^L)$  is expressible in linear Datalog by results in [DK08]; in particular the problem is in NL. if furthermore the variety admits type 4, then  $\neg CSP(\mathbf{H}^L)$  is not expressible in symmetric Datalog and is NL-hard by results in [LT09]. ■

Given a graph  $\mathbf{H}$ , it can be decided in polynomial time which of the different cases delineated in Theorem 11 the list homomorphism problem for  $\mathbf{H}$  satisfies. Furthermore, in the case  $CSP(\mathbf{H}^L)$  is in Logspace, we can further determine in polynomial time whether it is first-order definable or not; we discuss this in the next section and in the Appendix.

## 5 First-order definable Cases

For completeness' sake, in this section we describe which graphs have a list homomorphism problem which is first-order definable (equivalently, are in  $AC^0$ , see [bkl08].) By results in [LT09], if  $\mathbf{H} \in \mathcal{L}$  either the list homomorphism problem for  $\mathbf{H}$  is first-order definable or it is Logspace complete under FO reductions. We'll need the following characterisation of structures whose CSP is first-order definable [LLT07]. Let  $\mathbf{G}$  be a relational structure and let  $a, b \in G$ . We say that  $b$  *dominates*  $a$  in  $\mathbf{G}$  if for any basic relation  $R$  of  $\mathbf{G}$ , and any tuple  $t \in R$ , replacement of any occurrence of  $a$  by  $b$  in  $t$  will yield a tuple of  $R$ . If  $\mathbf{G}$  is a relational structure, we say that the structure  $\mathbf{G}^2$  *dismantles to the diagonal* if there exists a sequence of elements  $\{x_0, \dots, x_n\} = G^2 \setminus \{(g, g) : g \in G\}$  such that, for all  $0 \leq i \leq n$ ,  $x_i$  is dominated in  $\mathbf{G}_i$ , where  $\mathbf{G}_0 = \mathbf{G}^2$  and  $\mathbf{G}_i = G^2 \setminus \{x_0, \dots, x_{i-1}\}$  for  $i > 0$ .

**Lemma 12 ([LLT07])** *Let  $\mathbf{G}$  be a core relational structure. Then  $CSP(\mathbf{G})$  is first-order definable if and only if  $\mathbf{G}^2$  dismantles to the diagonal.*

**Theorem 13** *Let  $\mathbf{H}$  be a graph. Then  $CSP(\mathbf{H}^L)$  is first-order definable if and only if  $\mathbf{H}$  has the following form:  $H$  is the disjoint union of two sets  $L$  and  $N$  such that (i)  $L$  is the set of loops of  $\mathbf{H}$  and induces a complete graph, (ii)  $N$  is the set of non-loops of  $\mathbf{H}$  and induces a graph with no edges, and (iii)  $N = \{x_1, \dots, x_m\}$  such that the neighbourhood of  $x_i$  is contained in the neighbourhood of  $x_{i+1}$  for all  $1 \leq i \leq m - 1$ .*

*Proof:* We first prove that conditions (i) and (ii) are necessary. Let  $\{x, y\}$  be an (irreflexive) edge of  $\mathbf{H}$  or a pair of non-adjacent loops; then the substructure it induces isn't first-order definable. Indeed, in the case of an edge, odd cycles of arbitrary size are obstructions. If  $x$  and  $y$  are non-adjacent loops, arbitrary long paths with endpoints coloured by  $x$  and  $y$  are obstructions.

Now we prove (iii) is necessary. Suppose for a contradiction that there exist distinct elements  $x$  and  $y$  of  $N$  and elements  $n$  and  $m$  of  $L$  such that  $m$  is adjacent to  $x$  but not to  $y$ , and  $n$  is adjacent to  $y$  but not to  $x$ . Then  $CSP(\mathbf{G}')$  is first-order definable, where  $\mathbf{G}'$  is the substructure of  $\mathbf{H}^L$  induced by  $\{x, y, m, n\}$ . By Lemma 12,  $(\mathbf{G}')^2$  dismantles to the diagonal. Then  $(x, y)$  must be dominated by one of  $(x, x)$ ,  $(y, x)$  or  $(y, y)$  (because domination respects the basic relation  $\{x, y\}$ .) But  $(m, n)$  is a neighbour of  $(x, y)$  and none of the other three, a contradiction.

For the converse: we show that we can dismantle  $(\mathbf{H}^L)^2$  to the diagonal. Let  $x \in H$ : then  $(x_1, x)$  and  $(x, x_1)$  are dominated by  $(x, x)$ . Suppose that we have dismantled every element containing a coordinate equal to  $x_i$  with  $i \leq j - 1$ : if  $x$  is any element of  $H$  such that the elements  $(x_j, x)$  and  $(x, x_j)$  remain, then either  $x$  is a loop or  $x = x_k$  with  $k \geq j$ ; in any case the elements  $(x_j, x_k)$  and  $(x_k, x_j)$  are dominated by  $(x, x)$ . In this way we can remove all pairs  $(x, y)$  with one of  $x$  or  $y$  a non-loop. For the remaining pairs, notice that if  $u$  and  $v$  are any loops then  $(u, v)$  is dominated (in what remains of  $(\mathbf{H}^L)^2$ ) by  $(u, u)$ . ■

## References

- [ABISV05] E. Allender, M. Bauland, N. Immerman, H. Schnoor, and H. Vollmer. The complexity of satisfiability problems: Refining Schaefer’s theorem. In *MFCS’05*, pages 71–82, 2005.
- [A05] A. Atserias. On digraph coloring problems and treewidth duality. *European Journal of Combinatorics*, 29(4), 796-820, 2008.
- [BK09] L. Barto, M. Kozik, Constraint satisfaction problems of bounded width, manuscript, 2009.
- [B04] A. Bulatov. A graph of a relational structure and constraint satisfaction problems. In *LICS’04*, pages 448–457, 2004.
- [B03] A. Bulatov, Tractable conservative constraint satisfaction problems. In *LICS’03*, 321–330, 2003.
- [B02] A. Bulatov. A dichotomy theorem for constraint satisfaction problems on a three-element set. *Journal of the ACM*, 53(1), 66-120, 2006.
- [BKJ00] A. Bulatov, P. Jeavons, A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM Journal of Computing*, 34(3), 720-742, 2005.
- [bk108] A. Bulatov, A. Krokhin, B. Larose, Dualities for Constraint Satisfaction Problems, In: *Complexity of Constraints*, LNCS 5250, 93-124, 2008.
- [CJ06] D. Cohen and P. G. Jeavons. *The complexity of constraint languages*. In *Handbook of Constraint Programming*, chapter 8. Elsevier, 2006.
- [D05] V. Dalmau. Linear Datalog and bounded path duality for relational structures. *Logical Methods in Computer Science*, 1(1), 2005. (electronic).
- [DK08] V. Dalmau and A. Krokhin. Majority constraints have bounded pathwidth duality. *European Journal of Combinatorics*, 29(4), 821-837, 2008.
- [DL08] V. Dalmau, B. Larose, Maltsev + Datalog  $\Rightarrow$  Symmetric Datalog. In *LICS’08*, 297-306, 2008.
- [ELT08] L. Egri, B. Larose, P. Tesson, Directed st-Connectivity is not definable in symmetric Datalog. In *ICALP’08*, 172-183, 2008.
- [ELT07] L. Egri, B. Larose, P. Tesson. Symmetric Datalog and constraint satisfaction problems in Logspace. In *LICS’07*, 193-202, 2007.
- [F01] T. Feder, Classification of homomorphisms to oriented cycles and k-partite satisfiability. *SIAM Journal on Discrete Mathematics*, 14(4), 471-480, 2001.
- [FHH99] T. Feder, P. Hell, J. Huang, Bi-arc graphs and the complexity of list homomorphisms. *Journal of Graph Theory*, 42, 61-80, 1999.
- [FHH99a] T. Feder, P. Hell, J. Huang, List homomorphisms and circular arc graphs. *Combinatorica*, 19, 487-505, 1999.

- [FV98] T. Feder, M.Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through datalog and group theory. *SIAM Journal of Computing*, 28, 57-104, 1998.
- [G06] F. Gurski, Characterizations for co-graphs defined by restricted NLC-width or clique-width operations. *Discrete Mathematics*, 306, 271–277, 2006.
- [HN90] P. Hell, J. Nešetřil, On the complexity of  $H$ -coloring. *Journal of Combinatorial Theory Ser. B*, 48, 92–110, 1990.
- [HM88] D. Hobby, R.N. McKenzie. *The Structure of Finite Algebras*, volume 76 of *Contemporary Mathematics*. AMS, Providence, R.I., 1988.
- [L06] B. Larose, Taylor operations on finite reflexive structures. *International Journal of Mathematics and Computer Science*, 1(1), 1-26, 2006.
- [LLT07] B. Larose, C. Loten, C. Tardif, A characterisation of first-order Constraint Satisfaction Problems. *Logical Methods in Computer Science*, 3(4), 2007.
- [LH08] B. Larose, L. Haddad, Colourings of hypergraphs, permutation groups and CSP's, *Logic Colloquium '04*, 93-108, Lect. Notes Log., Vol. 27, *Assoc. Symbol. Logic*, La Jolla, CA, 2008.
- [LT09] B. Larose, P. Tesson, Universal algebra and hardness results for constraint satisfaction problems. *Theoretical Computer Science*, 410, 1629-1647, 2009.
- [LVZ09] B. Larose, M. Valeriote, L. Zádori, An algebraic characterisation of the ability to count. Submitted, 2008.
- [LZ07] B. Larose, L. Zádori, Bounded width problems and algebras. *Algebra Universalis*, 56(3-4), 439-466, 2007.
- [L04] L. Libkin, *Elements of Finite Model Theory*. Springer, 2004.
- [M03] R. McConnell, Linear-time recognition of circular-arc graphs. *Algorithmica*, 37(2), 93-147, 2003.
- [Sch78] T.J. Schaefer. The complexity of satisfiability problems. In *STOC'78*, pages 216–226, 1978.
- [V07] M. Valeriote, A subalgebra intersection property for congruence distributive varieties, *Canadian Journal of Mathematics*, in press.

## 6 Appendix (including all Figures)

### 6.1 Refining the Duality

We refine the dichotomy result of [FHH99] by showing that the tractable cases are precisely those whose associated variety omits type 1. The proof uses the notion of *Taylor operations*; for our purposes it is not necessary to state the exact nature of the special identities these operations satisfy, it suffices to know that an idempotent variety (which are the only ones we consider here) admits a Taylor operation if and only if it omits type 1 [HM88].

**Lemma 2** *Let  $\mathbf{H}$  be a graph. Then the following conditions are equivalent:*

1. *the variety  $\mathcal{V}(\mathbb{H})$  omits type 1;*
2. *the graph  $\mathbf{H}$  admits a conservative majority operation;*
3. *the graph  $\mathbf{H}$  is a bi-arc graph.*

*Proof:* The equivalence of (2) and (3) is from [FHH99], and (2) implies (1) trivially, hence it remains to show that (1) implies (3). We shall use the following construction from [FHH99]. Given a graph  $\mathbf{H}$ , let  $\mathbf{H}^*$  denote the irreflexive graph obtained from  $\mathbf{H}$  as follows: its vertices consist of two copies of the vertex set of  $\mathbf{H}$ , say  $H' = \{x' : x \in H\}$  and  $H'' = \{x'' : x \in H\}$ , with edges  $(x', y'')$  iff  $(x, y)$  is an edge of  $\mathbf{H}$ . In other words,  $\mathbf{H}^* = \mathbf{H} \times \mathbf{K}_2$  where  $\mathbf{K}_2$  is the irreflexive edge.

Step 1: if the variety  $\mathcal{V}(\mathbb{H})$  omits type 1, then the graph  $\mathbf{H}^*$  admits a Taylor operation.

Proof of Step 1.<sup>1</sup>: we know that the graph  $\mathbf{H}$  admits a Taylor operation  $f$ , say of arity  $k$ . We define an operation  $g$  of arity  $2k - 1$  on  $\mathbf{H}^*$  as follows: let  $x = (x_1^{e_1}, \dots, x_{2k-1}^{e_{2k-1}})$  be an element of  $(\mathbf{H}^*)^{2k-1}$ , where  $x_1, \dots, x_{2k-1} \in H$  and  $e_1, \dots, e_{2k-1} \in \{', ''\}$ . Then obviously exactly one of ' or '' appears at least  $k$  times; let  $\epsilon$  denote this symbol; let  $i_1, \dots, i_k$  denote the first  $k$  positions where it appears in the tuple  $x$ ; then define

$$g(x) = f(x_{i_1}, \dots, x_{i_k})^\epsilon.$$

It is clear that this is a well-defined operation on the graph  $\mathbf{H}^*$ , and it is easy to see that it preserves edges; since  $f$  is conservative, so is  $g$ . Although it is not very difficult to show that  $g$  is itself a Taylor operation, we provide a simpler argument. Let  $\{x^u, y^v\}$  be a 2-element subgraph of  $\mathbf{H}^*$ . Suppose first that  $x^u$  and  $y^v$  belong to different colour classes of  $\mathbf{H}^*$ : then the restriction of  $g$  to this subgraph is a near-unanimity operation: indeed

$$g(x^u, \dots, x^u, y^v, x^u, \dots, x^u) = f(x, \dots, x)^u = x^u$$

and similarly for  $g(y^v, \dots, y^v, x^u, y^v, \dots, y^v)$ . On the other hand if  $x^u$  and  $y^v$  are in the same colour class, then the restriction of  $g$  to  $\{x^u, y^v\}$  is a Taylor operation (it is the same as  $f$ ). Consequently, if  $\mathbb{A}$  denotes the algebra whose terms are the conservative operations preserving the graph  $\mathbf{H}^*$ , then every of its 2-element subalgebras admits a Taylor operation. If the variety generated by  $\mathbb{A}$  admitted type 1,  $\mathbb{A}$  would have a trivial 2-element divisor, but since every 2-element subset is a subalgebra it would mean  $\mathbb{A}$  has a trivial 2-element subalgebra, a contradiction. Consequently the graph  $\mathbf{H}^*$  admits a conservative Taylor operation.

Step 2: if the irreflexive graph  $\mathbf{H}^*$  admits a conservative Taylor operation, then it is the complement of a circular arc graph (and hence  $\mathbf{H}$  is a bi-arc graph by Prop 3.1 of [FHH99].)

<sup>1</sup>This argument was obtained in collaboration with Gábor Kun.

Proof of Step 2: we invoke Corollary 4.6 of [FHH99a]: the bipartite graph  $\mathbf{H}^*$  is the complement of a circular arc graph provided it contains (i) no induced cycle of length greater than 4 nor (ii) a configuration called a *special edge-asteroid*. We show that if  $\mathbf{H}^*$  admits a conservative Taylor operation then it cannot contain these configurations.

(i) Suppose the graph contains an induced cycle  $\mathbf{C}$  of length greater than 4. Clearly the cycle itself admits a conservative Taylor operation. Suppose first that  $\mathbf{C}$  is of length 6;<sup>2</sup> without loss of generality the vertices of the cycle  $\mathbf{C}$  are  $\{0, 1, 2, 3, 4, 5\}$ . Consider the relation

$$\alpha = \{(f(x), f(y)) : \text{for some homomorphism } f : \mathbf{K} \rightarrow \mathbf{C}\}$$

where  $\mathbf{K}$  is the graph pictured in Figure 6: a vertex labelled  $i$  means that the value of  $f$  on this vertex is fixed to  $i$ . It is easy to verify that  $\alpha = \{(0, 3), (2, 5), (4, 1)\}$ . Now define the relation

$$\beta = \{(u, v) : \exists z (u, z) \in \alpha, (z, v) \in \theta\}$$

where  $\theta$  is the edge relation of  $\mathbf{C}$ . It is trivial to verify that

$$\beta = \{(u, v) \in \{0, 2, 4\} : u \neq v\},$$

and it is easy to verify (and well-known) that every idempotent operation preserving this relation is a projection; hence every idempotent operation on the 6-cycle must be a projection on the set  $\{0, 2, 4\}$  and in particular cannot be a Taylor operation.

Now suppose that  $\mathbf{C}$  has length  $n$  greater than 4 and  $n \neq 6$ : consider the relation

$$\gamma = \{(u, w) : \exists v (u, v), (v, w) \in \theta\}$$

where  $\theta$  is the edge relation of the cycle. Then  $\gamma$  is clearly a reflexive, symmetric relation which is also intransitive and contains a cycle: it follows from Theorem 3.6 of [L06] (see also Lemma 4.3 of [LH08]) that it is not invariant under a Taylor operation.

(ii) We need not know what a special edge-asteroid is, but rather we inspect the proof of Theorem 3.2 of [FHH99a]: if  $\mathbf{H}^*$  contains a special edge-asteroid, then there exist a 3-vertex subset  $T = \{a, b, c\}$  of the graph, together with structures  $\mathbf{C}_1, \dots, \mathbf{C}_6$  of the same type as the structure  $(\mathbf{H}^*)^L$  (recall this is the graph  $\mathbf{H}^*$  with all subsets added), each structure  $\mathbf{C}_i$  contains vertices  $s$  and  $t$  and

$$\theta_i = \{(f(s), f(t)) : \text{for some homomorphism } f : \mathbf{C}_i \rightarrow (\mathbf{H}^*)^L\}$$

where

$$\begin{aligned} \theta_1 &= \{a\} \times \{b\} \cup \{b\} \times \{b, c\} \cup \{c\} \times \{b, c\} \\ \theta_2 &= \{a\} \times \{c\} \cup \{b\} \times \{b, c\} \cup \{c\} \times \{b, c\} \\ \theta_3 &= \{a\} \times \{a, c\} \cup \{b\} \times \{c\} \cup \{c\} \times \{a, c\} \\ \theta_4 &= \{a\} \times \{a, c\} \cup \{b\} \times \{a\} \cup \{c\} \times \{a, c\} \\ \theta_5 &= \{a\} \times \{a, b\} \cup \{b\} \times \{a, b\} \cup \{c\} \times \{a\} \\ \theta_6 &= \{a\} \times \{a, b\} \cup \{b\} \times \{a, b\} \cup \{c\} \times \{b\}; \end{aligned}$$

---

<sup>2</sup>This construction is due to T. Feder, see [F01].

Now consider the following binary relation  $\beta$  on  $T$  (see Figure 3 of [FHH99a]): it consists of all pairs  $(u, v) \in T^3$  such that there exist vertices  $z_1, z_2, z_3 \in T$  with

$$\begin{aligned} (u, z_1) &\in \theta_1 \\ (u, z_2) &\in \theta_3 \\ (u, z_3) &\in \theta_5 \\ (v, z_1) &\in \theta_2 \\ (v, z_2) &\in \theta_4 \\ (v, z_3) &\in \theta_6 \end{aligned}$$

it is easy to verify that

$$\beta = \{(a, b), (b, a), (a, c), (c, a), (b, c), (c, a)\}.$$

It follows that any edge-preserving conservative operation on  $\mathbf{H}^*$  preserves this relation, and hence is a projection as mentioned earlier. Hence if the graph  $\mathbf{H}^*$  contains a special edge-asteroid, then  $\mathbf{H}^*$  cannot admit a conservative Taylor operation. ■

## 6.2 Irreflexive Graphs

**Lemma 4** *Let  $\mathbf{H}$  be an irreflexive graph. Then the following conditions are equivalent:*

1.  $\mathbf{H}$  is basic;
2.  $\mathbf{H}$  is bipartite, contains no induced 6-cycle, nor any induced path of length 5.

*Proof:* (1)  $\Rightarrow$  (2): Let  $\mathcal{K}'$  denote the class of all those bipartite irreflexive graphs containing no induced 6-cycle, nor any induced path of length 5. It obviously contains the one element graph. Suppose that  $\mathbf{H}_1 \odot \mathbf{H}_2$  contains an induced subgraph  $\mathbf{C}$  which is a 6-cycle or a path of length 5: we show that  $\mathbf{C}$  is contained in  $\mathbf{H}_1$  or  $\mathbf{H}_2$ . Suppose for a contradiction that it is not: by definition of special sum, it is clear that, since  $\mathbf{C}$  is connected, it must contain at least one vertex in  $T_1$  and at least one in  $B_2$ ; on the other hand, since it contains no 4-cycle,  $\mathbf{C}$  can have at most 2 vertices in  $T_1$  and at most 1 in  $B_2$ , without loss of generality. Suppose first that there is exactly one vertex of  $\mathbf{C}$  in  $T_1$ . Since every vertex of  $\mathbf{C}$  has degree at most 2, it follows that no more than 1 vertex of  $\mathbf{C}$  can be in  $B_1$ , and similarly no more than 1 vertex of  $\mathbf{C}$  can be in  $T_2$ , an obvious contradiction. On the other hand if  $\mathbf{C}$  has 2 vertices in  $T_1$ , then  $\mathbf{C}$  has no vertex in  $T_2$  and at most 2 in  $B_1$ , a contradiction again. Hence we conclude that  $\mathcal{K}'$  is closed under special sum. It is clear that it is closed under disjoint union, and hence (1) implies (2).

(2)  $\Rightarrow$  (1): Suppose for a contradiction that there exists a graph  $\mathbf{H}$  which is bipartite, contains no induced 6-cycle, nor any induced path of length 5, but is not basic: choose it so that its set of vertices is of minimal size. Obviously  $\mathbf{H}$  is connected. We denote the usual graph distance between vertices  $x$  and  $y$  by  $d(x, y)$ , i.e. the length of a shortest path in the graph between  $x$  and  $y$ . Let  $N(x)$  denote the set of neighbours of  $x$  in  $\mathbf{H}$ , and let

$$N_2(x) = \{t \in T_1 : d(x, t) = 2\}.$$

**Claim 1.** For every  $x \in H$  there exists  $y \in H$  such that  $d(x, y) = 3$ .

*Proof of Claim 1.* Otherwise, since  $\mathbf{H}$  is connected, we'd have some  $x \in H$  with  $d(x, y) \leq 2$  for all  $y \in H$ . Now let  $B_2$  denote the set of all vertices adjacent to  $x$ , and let  $T_2 = H \setminus (B_2 \cup \{x\})$ . Furthermore let  $B_1 = \emptyset$  and  $T_1 = \{x\}$ . Since  $\mathbf{H}$  is bipartite  $(B_2, T_2)$  is a bipartition, and hence  $\mathbf{H}$  is expressed as a special sum, a contradiction.

**Claim 2.** There exists  $x \in H$  such that the subgraph induced by  $H \setminus \{x\}$  is connected.

*Proof of Claim 2.* Notice first that if for some  $x$  the subgraph  $\mathbf{G}$  induced by  $H \setminus \{x\}$  is not connected, then it contains at most one connected component with 2 or more vertices. Indeed, by Claim 1 let  $y \in H$  such that  $d(x, y) = 3$ ; let  $y, w, z, x$  be an induced path of length 3 from  $y$  to  $x$ . Now choose some non-trivial component of  $\mathbf{G}$  not containing  $y$ : since  $\mathbf{H}$  is connected, this component clearly contains adjacent vertices  $u$  and  $v$  with  $u$  adjacent to  $x$ . But then the vertices  $y, w, z, x, u, v$  induce a path of length 5 in  $\mathbf{H}$ , a contradiction.

Now choose any vertex  $x$  in  $\mathbf{H}$ . If the subgraph induced by  $H \setminus \{x\}$  is connected we are done; otherwise, one of its components must be trivial, i.e.  $\mathbf{H}$  has a vertex  $x'$  dangling from  $x$ . Then the subgraph induced by  $H \setminus \{x'\}$  is connected.

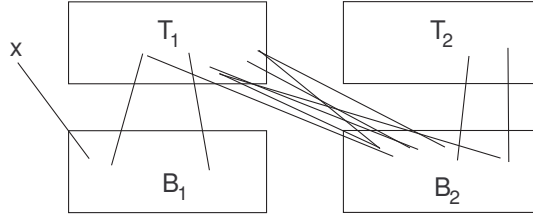


Figure 1: The graph  $\mathbf{H}$ .

So we may now suppose that  $\mathbf{H}$  has the following structure: there is some vertex  $x$  such that the subgraph  $\mathbf{G}$  induced by  $H \setminus \{x\}$  is connected; by induction hypothesis,  $\mathbf{G}$  is a special sum, with subsets  $B_i, T_i, (i = 1, 2)$  where  $T_1$  and  $B_2$  are non-empty. We suppose without loss of generality that  $x$  is adjacent to some vertex in  $B_1 \cup B_2$  (see Figure 1.)

Case 1.  $T_2$  is non-empty.

**Claim 3.** There exist an edge  $(u, v)$  with  $u \in B_2$  and  $v \in T_2$  such that  $u$  is not adjacent to  $x$ .

*Proof of Claim 3.* Suppose for a contradiction that this is not the case: then  $B_2 = B_2^0 \cup B_2^1$  where  $B_2^0$  consists of all elements of  $B_2$  not adjacent to any vertex in  $T_2$ , and since  $T_2$  is non-empty, the set  $B_2^1$  is non-empty, and contains by hypothesis only vertices adjacent to  $x$ . Then define a decomposition of  $H$  as follows: let  $B'_1 = B_1 \cup B_2^0, T'_1 = T_1 \cup \{x\}, B'_2 = B_2^1$  and  $T'_2 = T_2$ . But then  $\mathbf{H}$  is a special sum, a contradiction.

**Claim 4.** The subgraph induced by  $N(x) \cup N_2(x)$  is complete bipartite.

*Proof of Claim 4.* Otherwise, we may find elements  $t \in N_2(x)$  and  $z \in N(x)$  which are not adjacent. By definition there exists  $y \in N(x)$  adjacent to  $t$ . Let  $u$  and  $v$  be the elements whose existence is guaranteed by the last claim: then it is easy to see that the sequence  $z, x, y, t, u, v$  is an induced path of length 5 in  $\mathbf{H}$ , a contradiction.

Consider the following decomposition of  $H$ : let  $B'_1 = B_1 \setminus (N(x) \cap B_1), T'_1 = N_2(x), B'_2 = (N(x) \cap B_1) \cup B_2$  and  $T'_2 = (T_1 \setminus N_2(x)) \cup \{x\} \cup T_2$ . By Claim 4 this is a decomposition of  $\mathbf{H}$  as a special sum, unless there exists some edge  $(y, z)$  with  $y \in B'_1$  and  $z \in T'_2$ , i.e. with  $y \in B_1 \setminus N(x)$  and  $z \in T_1 \setminus N_2(x)$ . Suppose this occurs. Then we have the following:

**Claim 5.**  $(y, t)$  is an edge for every  $t \in N_2(x)$ .

*Proof of Claim 5.* If this is not the case, then choose some  $t \in N_2(x)$  not adjacent to  $y$ ; let  $n \in N(x)$  be adjacent to  $t$ . By Claim 3 we can find  $u \in B_2$  not adjacent to  $x$ . Then the sequence  $y, z, u, t, n, x$  is an induced path of length 5 in  $\mathbf{H}$ , a contradiction.

It follows from Claim 5 that we can modify our last decomposition as follows: simply remove from  $B'_1$  all the offending vertices such as  $y$ . More precisely, let  $Y$  be the set of all  $y \in B'_1$  that have some neighbour  $z \in T_1 \setminus N_2(x)$ , and let  $B''_1 = B'_1 \setminus Y$ ,  $T''_1 = T'_1$ ,  $B''_2 = Y \cup B'_2$  and  $T''_2 = T'_2$ . By Claim 5, this shows that  $\mathbf{H}$  is a special sum, a contradiction.

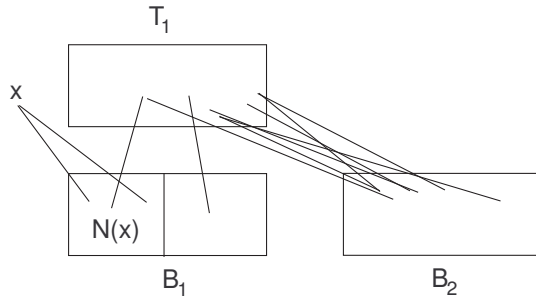


Figure 2: The graph  $\mathbf{H}$  with  $T_2$  empty.

Case 2.  $T_2$  is empty.

Notice that in this case we may assume that  $N(x) \subseteq B_1$ , by simply decomposing  $H \setminus \{x\}$  if necessary as  $B'_1 = B_1 \cup N(x)$ ,  $B'_2 = B_2 \setminus N(x)$ , and  $T'_i = T_i$  for  $i = 1, 2$ . (Of course, if  $\mathbf{H}$  is not a special sum then there is at least one vertex in  $B_2 \setminus N(x)$  for otherwise we could set  $T'_1 = T_1 \cup \{x\}$ .)

**Claim 6.** For every  $y, z \in N(x)$  we have  $N(y) \subseteq N(z)$  or  $N(z) \subseteq N(y)$ . Similarly, for every  $u, v \in T_1$ , either  $N(u) \cap N(x) \subseteq N(v) \cap N(x)$  or  $N(v) \cap N(x) \subseteq N(u) \cap N(x)$ .

*Proof of Claim 6.* Suppose this is not the case: then we may find  $y, z \in N(x)$  and  $u \in N(y)$  and  $v \in N(z)$  such that  $u$  is not adjacent to  $z$  and  $v$  is not adjacent to  $y$ . Let  $b \in B_2$ . Then clearly the subgraph of  $\mathbf{H}$  induced by  $\{x, y, z, u, v, b\}$  is a 6-cycle, a contradiction. The argument for the second statement is identical.

By Claim 6, there exists an ordering of  $N(x) = \{b_0, \dots, b_m\}$  such that  $N(b_i) \subseteq N(b_j)$  if  $i \leq j$ , and an ordering of  $T_1 = \{t_0, \dots, t_M\}$  such that  $N(t_i) \cap N(x) \subseteq N(t_j) \cap N(x)$  if  $i \leq j$ . Since  $\mathbf{H}$  is connected, it is easy to see that  $b_m$  must be adjacent to  $t_M$ ; and by Claim 1,  $b_m$  cannot be adjacent to  $t_0$ .

**Claim 7.** For every  $t \in T_1$ , either  $N(t) \cap N(x) = N(x)$  or  $N(t) \cap N(x) = \emptyset$ .

*Proof of Claim 7.* Suppose this is not the case. Then there exists some  $t \in T_1$  such that  $t$  is adjacent to  $b_m$  but not to  $b_0$ . Then for any  $b \in B_2$  the sequence  $b_0, x, b_m, t, b, t_0$  is an induced path of length 5, a contradiction.

Let  $F$  denote the set of vertices  $t \in T_1$  such that  $N(t) \cap N(x) = N(x)$  and let  $E$  denote the set of vertices  $t \in T_1$  such that  $N(t) \cap N(x) = \emptyset$ .

**Claim 8.** For every  $y \in B_1 \setminus N(x)$ , if  $y$  is adjacent to some vertex in  $E$  then it is adjacent to every vertex in  $F$ .

*Proof of Claim 8.* Otherwise we can find  $t \in E$  and  $t' \in F$  and  $y \in B_1 \setminus N(x)$  such that  $(y, t)$  is an edge but  $(y, t')$  is not. Then for any  $b \in B_2$  the sequence  $x, b_m, t', b, t, y$  is an induced path of length 5, a contradiction.

Let  $Y$  denote the set of vertices in  $B_1 \setminus N(x)$  that are adjacent to some vertex in  $E$ . By the last claim, the subgraph induced by  $(Y \cup B_2 \cup N(x)) \cup F$  is complete bipartite. Consider the following

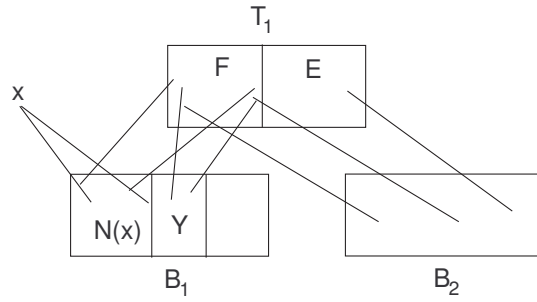


Figure 3: The graph  $\mathbf{H}$ .

decomposition: let  $B'_2 = Y \cup N(x) \cup B_2$ ,  $B'_1 = B_1 \setminus B'_2$ ,  $T'_1 = F$  and  $T'_2 = E \cup \{x\}$ . By the above argument, this shows that  $\mathbf{H}$  is basic, a contradiction. ■

**Lemma 14** *Let  $\mathbf{H}$  be a graph such that  $\mathcal{V}(\mathbb{H}) = \{3\}$ . Then  $\mathbf{H}$  is basic.*

*Proof:* Let  $\mathbf{H}$  be a graph such that  $\text{typ}(\mathcal{V}(\mathbb{H})) = \{3\}$ . By Lemma 4, and since all terms considered are conservative, it will suffice to show that if  $\mathbf{H}$  is an odd cycle, the 6-cycle or the path of length 5, then  $\mathcal{V}(\mathbb{H})$  admits some type other than 3.

It follows from the proof of Lemma 2 that if  $\mathbf{H}$  is a 6-cycle or an odd cycle of length at least 5 then it cannot admit a Taylor operation; consequently  $\mathcal{V}(\mathbb{H})$  admits type 1. The argument used for the 6-cycle included the fact that a 3-cycle is not invariant under a Taylor operation.

Now suppose that  $\mathbf{H}$  is the path of length 5 on vertices  $\{0, 1, 2, 3, 4, 5\}$ . Let

$$\epsilon = \{(f(x), f(y)) : \text{for some homomorphism } f : \mathbf{M} \rightarrow \mathbf{H}\}$$

where  $\mathbf{M}$  is the labelled graph in Figure 8. It is easy to verify that

$$\epsilon = \{(0, 0), (4, 0), (4, 4)\},$$

i.e. the set  $\{0, 4\}$  equipped with the relation  $\epsilon$  is isomorphic to the natural order on  $\{0, 1\}$ . Hence every term operation of the subalgebra  $\{0, 4\}$  of  $\mathbb{H}$  is order-preserving, and thus this subalgebra has type 1, 4 or 5 (it is in fact type 4 since the path admits a majority operation.) ■

### 6.3 Reflexive Graphs

**Lemma 15** *Let  $\mathcal{L}_R$  denote the class of reflexive graphs in  $\mathcal{L}$ . Then  $\mathcal{L}_R$  is the smallest class  $\mathcal{D}$  of reflexive graphs such that:*

1.  $\mathcal{D}$  contains the one-element graph;
2.  $\mathcal{D}$  is closed under disjoint union;
3. if  $\mathbf{H}_1$  is a loop and  $\mathbf{H}_2 \in \mathcal{D}$  then  $\mathbf{H}_1 \circlearrowleft \mathbf{H}_2 \in \mathcal{D}$ .

*Proof:* Clearly  $\mathcal{L}_R$  consists of all reflexive graphs that do not contain the reflexive path of length 3 nor the reflexive 4-cycle: it follows easily that  $\mathcal{D} \subseteq \mathcal{L}_R$ . Suppose that the other inclusion does not hold, i.e. let  $\mathbf{H}$  be a graph of smallest size that belongs to  $\mathcal{L}_R$  but cannot be obtained from the one-element graph using the operations of disjoint union and adjunction of a loop. By minimality,  $\mathbf{H}$  contains no universal vertex, it contains more than one vertex, and every of its proper induced connected subgraphs contains a universal vertex. Pick some edge  $(x, y)$  in  $\mathbf{H}$ ; since there is no universal vertex there exists some  $z$  not adjacent to  $y$ . Let  $\mathbf{G}$  be the subgraph induced by  $H \setminus \{x\}$ . Case 1:  $\mathbf{G}$  is connected. Let  $u$  be a universal vertex of  $\mathbf{G}$ ; we have edges  $(x, y), (y, u), (u, z)$ . Since  $\mathbf{H}$  has no universal vertex then  $x$  is not adjacent to  $u$ . Thus  $\{x, y, z, u\}$  is either a reflexive path of length 3 or a reflexive 4-cycle, a contradiction. Case 2:  $\mathbf{G}$  is not connected. Let  $C$  and  $D$  be distinct components of  $\mathbf{G}$ ; since  $x$  is not universal in  $\mathbf{H}$  there exists some  $z$  not adjacent to  $x$ , and without loss of generality suppose that  $z \in C$ . Since  $\mathbf{H}$  is connected there exists a path from  $z$  to some element in  $D$ , in particular we can find edges  $(z', u), (u, x), (x, v)$  where  $z', u \in C, v \in D$  and  $z'$  is not adjacent to  $x$ . It is easy to verify that  $\{z', u, x, v\}$  induces a reflexive path of length 3, a contradiction. ■

Our last result states that the reflexive graphs avoiding the path of length 3 and the 4-cycle are precisely those constructed from the one-element loop using disjoint union and adjunction of a universal vertex. These graphs can also be described by the following property: *every connected induced subgraph of size at most 4 has a universal vertex*. These graphs have been studied previously as those with *NLCT width 1*, which were proved to be exactly the trivially perfect graphs (Theorem 5, [G06]). Our result provides an alternate proof of the equivalence of these conditions.

## 6.4 The General Case

**Lemma 16** *Let  $\mathbf{H}$  be a graph such that  $\mathcal{V}(\mathbb{H}) = \{3\}$ . Then  $\mathbf{H} \in \mathcal{L}$ .*

To prove Lemma 16 we require the following:

**Lemma 17** *Let  $\mathbf{H}$  be a graph and let  $\mathbf{K} = \mathbf{H} \times \mathbf{K}_2$  be the product of  $\mathbf{H}$  with the edge. Let  $\mathbb{K}$  denote the algebra associated to the structure  $\mathbf{K}^L$ . If  $\mathcal{V}(\mathbb{K})$  admits type  $i$ , then  $\mathcal{V}(\mathbb{H})$  admits type  $j$  for some  $j \leq i$  (in the usual order of types.)*

*Proof:* Let  $i$  be a smallest type that  $\mathcal{V}(\mathbb{K})$  admits i.e. suppose it admits no type strictly smaller than  $i$ . We shall give a proof for  $i = 4$ , and outline how to modify it for the other types. By a result of Valeriote (Lemma 3.1 of [V07]) some divisor of  $\mathbb{K}$  is polynomially equivalent to the 2-element lattice; since every 2-element subset is a subalgebra it means we actually have a 2-element subalgebra of  $\mathbb{K}$  which is polynomially equivalent to a lattice. Without loss of generality let  $z$  and  $w$  denote the elements of this subset. In particular, the relation  $\theta = \{(z, z), (z, w), (w, w)\}$  is invariant under every term operation of  $\mathbb{K}$ . Hence there exists some structure  $\mathbf{L}$  similar to  $\mathbf{K}^L$  and vertices  $u$  and  $v$  in  $L$  such that

$$\theta = \{(f(u), f(v)) : f : \mathbf{L} \rightarrow \mathbf{K}^L\}.$$

For convenience, we view  $\mathbf{L}$  as a graph with lists assigned to each vertex. We claim that we may assume that the underlying graph of  $\mathbf{L}$  is symmetric, bipartite and connected. Indeed, it is clear that making every edge symmetric will not change anything, and since there exists a homomorphism from  $\mathbf{L}$  to  $\mathbf{K}$  (which is bipartite) the underlying graph is bipartite. Finally, suppose that  $\mathbf{L}$  is not connected: obviously we may suppose that there are exactly two components, one containing  $u$  and

the other  $v$  (other components may be removed with no change.) But then it is easy to see that if  $f$  and  $g$  are homomorphisms from  $\mathbf{L}$  to  $\mathbf{K}$ , and  $f_u$  denotes the restriction to the component of  $u$  and  $g_v$  the restriction to the component of  $v$ , then the map  $h$  defined by setting  $h(x) = f_u(x)$  if  $x$  is in the component of  $u$  and  $h(x) = g_v(x)$  otherwise is clearly a homomorphism. Hence if  $(a, b), (c, d) \in \theta$  then  $(a, d), (b, c) \in \theta$ , i.e.  $\theta$  is a product, a contradiction.

Since  $(z, z) \in \theta$ , there exists  $f : \mathbf{L} \rightarrow \mathbf{K}^L$  mapping  $(u, v)$  to  $(z, z)$  and hence  $u$  and  $v$  are in the same colour class of  $\mathbf{L}$ . Since there is a homomorphism mapping  $(u, v)$  to  $(z, w)$  it means that  $z$  and  $w$  are in the same colour class of  $\mathbf{K}$ : without loss of generality we may assume that  $z = (a, 0)$  and  $w = (b, 0)$  for some  $a, b \in H$ . Thus every homomorphism from  $\mathbf{L}$  to  $\mathbf{K}$  maps the colour class of  $u$  to  $H \times \{0\}$  and the other colour class to  $H \times \{1\}$ . In particular, for any  $x \in L$  in the colour class of  $u$  we can assume that the elements in its list are all from  $H \times \{0\}$  and if  $x$  is not in the class of  $u$  then its list is contained in  $H \times \{1\}$ . In other words, for every  $x \in L$ , the list of  $x$  is of the form  $L_x \times \{0\}$  or  $L_x \times \{1\}$  for some  $L_x \subseteq H$ . Finally, notice we may assume that the list for  $u$  and  $v$  is  $\{z, w\} = \{a, b\} \times \{0\}$ .

Now consider the structure  $\mathbf{M}$  of the same type as  $\mathbf{H}^L$  obtained from  $\mathbf{L}$  by replacing the list of each  $x$  by  $L_x$ . Let  $\theta' = \{(a, a), (a, b), (b, b)\}$ . We claim that

$$\theta' = \{(f(u), f(v)) : f : \mathbf{M} \rightarrow \mathbf{H}^L\}.$$

Given some  $f : \mathbf{L} \rightarrow \mathbf{K}^L$ , compose it with the projection onto  $H$ ; it is clear that this is a homomorphism from  $\mathbf{M}$  to  $\mathbf{H}^L$ , and hence  $\theta'$  is contained in the set on the right. Since the list for  $u$  and  $v$  in  $\mathbf{M}$  is  $\{a, b\}$ , it remains to show that  $(b, a)$  is not in the set. If it were, we'd have a homomorphism  $f : \mathbf{M} \rightarrow \mathbf{H}^L$  mapping  $(u, v)$  to  $(b, a)$ . But then define a map  $F : L \rightarrow K$  in the obvious way, by setting  $F(x) = (f(x), 0)$  if  $x$  is in the colour class of  $u$  and  $F(x) = (f(x), 1)$  otherwise. Clearly this is a homomorphism from  $\mathbf{L}$  to  $\mathbf{K}^L$  mapping  $(u, v)$  to  $(w, z)$ , a contradiction.

We conclude that  $\mathbb{H}$  has a 2-element subalgebra  $\{a, b\}$  whose terms preserve the ordering  $a \leq b$ ; hence this subalgebra has type 1, 4 or 5.

The argument for types 1, 2 and 5 is quite similar: for type 1 use the 3-ary relation “not-all-equal”  $\theta = \{0, 1\}^3 \setminus \{(0, 0, 0), (1, 1, 1)\}$ ; for type 2 use  $\theta = \{(x, y, z) : x + y = z\}$  and for type 5 use  $\theta = \{0, 1\}^3 \setminus \{(1, 1, 0)\}$ . ■

*Proof:* (of Lemma 16) We must show that if  $\mathcal{V}(\mathbb{H}) = \{3\}$  then  $\mathbf{H}$  cannot contain any of the graphs listed in the definition of the class  $\mathcal{L}$ .

(1) It is easy to verify that a reflexive cycle of length 4 is not a bi-arc graph (see [FHH99]). It follows from Lemma 2 that if  $\mathbf{H}$  contains an induced reflexive 4-cycle then  $\mathcal{V}(\mathbb{H})$  admits type 1.

Suppose that  $\mathbf{H}$  contains an induced reflexive path of length 3, say on vertices  $\{0, 1, 2, 3\}$ . Let

$$\delta = \{(f(x), f(y)) : \text{for some homomorphism } f : \mathbf{L} \rightarrow \mathbf{H}\}$$

where  $\mathbf{L}$  is the labelled graph in Figure 4: if a vertex  $v$  is labelled by a set  $S$  it means we must have  $f(v) \in S$ . It is easy to verify that

$$\delta = \{(0, 0), (2, 0), (2, 2)\},$$

i.e. the set  $\{0, 2\}$  equipped with the relation  $\delta$  is isomorphic to the natural order on  $\{0, 1\}$ . Hence every term operation of the subalgebra  $\{0, 2\}$  of  $\mathbb{H}$  is order-preserving, and thus this subalgebra cannot have type 3.

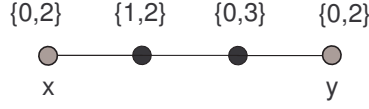


Figure 4: The graph  $\mathbf{L}$  in the construction of the order relation.

(2) It follows immediately from Lemma 14 that  $\mathbf{H}$  must avoid the irreflexive odd cycles and the irreflexive 5-path and 6-cycle.

(3) It is easy to verify that  $\mathbf{K} = \mathbf{H} \times \mathbf{K}_2$  contains either a path of length 5 or a 6-cycle when  $\mathbf{H}$  is one of  $\mathbf{B1}$ ,  $\mathbf{B2}$ ,  $\mathbf{B3}$ ,  $\mathbf{B4}$ . It follows from Lemma 14 and the previous lemma that in this case  $\mathcal{V}(\mathbb{H})$  admits some type other than 3.

Finally we prove that  $\mathbf{H}$  cannot contain an induced subgraph isomorphic to either  $\mathbf{B5}$  or  $\mathbf{B6}$  (the proof is a slight modification of two arguments we gave earlier for the irreflexive path of length 5 and the irreflexive 6-cycle.)

Suppose  $\mathbf{H}$  contains an induced copy of  $\mathbf{B6}$ . Notice that if we remove all edges in the reflexive part of  $\mathbf{B6}$  we obtain an irreflexive 6-cycle: let  $C = \{0, 1, 2, 3, 4, 5\}$  denote the vertices of this cycle. Consider the relation

$$\alpha = \{(f(x), f(y)) : \text{for some homomorphism } f : \mathbf{K} \rightarrow \mathbf{H}\}$$

where  $\mathbf{K}$  is the graph pictured in Figure 5: a vertex labelled  $i$  means that the value of  $f$  on this vertex is fixed to  $i$ . Furthermore, a vertex labelled  $D$  must be mapped to  $\{1, 3, 5\}$  and a vertex labelled  $E$  must be mapped to  $\{0, 2, 4\}$ . It is easy to verify that  $\alpha = \{(0, 3), (2, 5), (4, 1)\}$ . Now define the relation

$$\beta = \{(u, v) \in C^2 : \exists z (u, z) \in \alpha, (z, v) \in \theta\}$$

where  $\theta$  is the edge relation of  $\mathbf{H}$ . Then

$$\beta = \{(u, v) \in \{0, 2, 4\} : u \neq v\}.$$

As we argued in the proof of Lemma 2, it follows that  $\mathcal{V}(\mathbb{H})$  admits type 1.

Now consider the case where  $\mathbf{H}$  contains an induced copy of  $\mathbf{B5}$ . Notice that if we remove all edges in the reflexive part of  $\mathbf{B5}$  we obtain an irreflexive path of length 5: let  $P = \{0, 1, 2, 3, 4, 5\}$  denote the vertices of this path. We can proceed exactly as in Lemma 14 (see in particular Figure 8) to show that  $\mathcal{V}(\mathbb{H})$  admits some type other than 3. ■

**Lemma 18** *If  $\mathbf{H} \in \mathcal{L}$  then  $\mathcal{V}(\mathbb{H})$  is 4-permutable.*

*Proof:* We invoke the characterisation of  $\mathcal{L}$  from Theorem 10. We will prove that  $\mathcal{V}(\mathbb{H})$  is 4-permutable when  $\mathbf{H}$  is a basic graph, and show that this property is preserved under disjoint union and adjunction of basic graphs.

Let  $\mathbf{H}$  be a basic graph. The result is trivial if  $\mathbf{H}$  is a single loop, and if  $\mathbf{H}$  is a basic irreflexive graph then we invoke Lemma 6. So now assume that  $\mathbf{H}$  is obtained from some basic irreflexive graph  $\mathbf{H}_1$  with colour classes  $S$  and  $T$  by adding all edges  $(t, t')$  with  $t, t' \in T$ . By Lemma 6 there exist operations  $f_1, f_2, f_3$  on  $\mathbf{H}_1$  satisfying the required identities; furthermore recall that we can

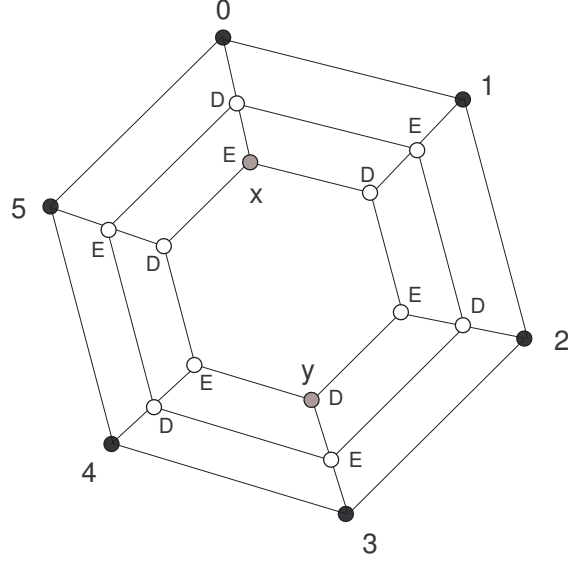


Figure 5: The graph  $\mathbf{K}$  in the construction of the relation  $\alpha$ .

assume that the  $f_i$  satisfy condition (D):

For every  $x, y, z, n, m \in H$  such that  $n$  is adjacent to  $x$  and  $m$  is adjacent to  $z$ ,  $f_1(x, y, z)$  is adjacent to  $n$  and  $f_3(x, y, z)$  is adjacent to  $m$ .

For convenience of notation, define, on triples  $(x_1, x_2, x_3)$  such that  $\{x_1, x_2, x_3\}$  intersects the set  $T$ , two ternary operations  $\mu^L$  and  $\mu^R$  by  $\mu^L(x_1, x_2, x_3) = x_j$  where  $j = \min\{i : x_i \in T\}$  and  $\mu^R(x_1, x_2, x_3) = x_k$  where  $k = \max\{i : x_i \in T\}$ . Notice that both of these operations are trivially edge-preserving. We define operations  $F_1, F_2$  and  $F_3$  on  $\mathbf{H}$  as follows:

$$F_1(x_1, x_2, x_3) = \begin{cases} x_1, & \text{if } x_2 = x_3, \text{ else} \\ f_1(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \text{ intersects only one of } S \text{ and } T, \\ \mu^L(x_1, x_2, x_3), & \text{otherwise.} \end{cases}$$

$$F_3(x_1, x_2, x_3) = \begin{cases} x_3, & \text{if } x_1 = x_2, \text{ else} \\ f_3(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \text{ intersects only one of } S \text{ and } T, \\ \mu^R(x_1, x_2, x_3), & \text{otherwise.} \end{cases}$$

$$F_2(x_1, x_2, x_3) = \begin{cases} f_2(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \text{ intersects only one of } S \text{ and } T, \\ \mu^L(x_1, x_2, x_3), & \text{otherwise.} \end{cases}$$

It is clear that all three operations are conservative, and that identities (1) and (3) are satisfied. To prove (2), suppose without loss of generality that  $x \neq y$ : if  $\{x, y\}$  intersects only one of  $S$  and  $T$  then  $F_1(x, x, y) = f_1(x, x, y) = f_2(x, y, y) = F_2(x, y, y)$ ; on the other hand if  $\{x, y\}$  intersects both  $S$  and  $T$  then  $F_1(x, x, y) = F_2(x, y, y)$  is the unique element in  $\{x, y\}$  that belongs to  $T$ . The proof that  $F_2(x, x, y) = F_3(x, y, y)$  is similar.

Next we prove that property (D) holds for  $F_1$  (the proof for  $F_3$  is identical.) Let  $n$  be a neighbour of  $x_1$ . If  $F_1(x_1, x_2, x_3) = x_1$  the result is trivial, and if  $F_1(x_1, x_2, x_3) = f_1(x_1, x_2, x_3)$

we're done since  $f_1$  satisfies (D). If  $F_1(x_1, x_2, x_3) = \mu^L(x_1, x_2, x_3)$ , there are two cases: if  $x_1 \in T$  then  $F_1(x_1, x_2, x_3) = x_1$ , otherwise  $x_1 \in S$  forces  $n \in T$  so  $n$  is necessarily adjacent to  $\mu^L(x_1, x_2, x_3) \in T$ .

Finally we show that  $F_1$  is edge-preserving (the proof for  $F_2$  and  $F_3$  is identical.) Let  $(x_1, x_2, x_3)$  and  $(y_1, y_2, y_3)$  be adjacent. If  $x_2 = x_3$  then  $F_1(x_1, x_2, x_3) = x_1$  and  $F_1(y_1, y_2, y_3)$  is adjacent to  $x_1$  by property (D). Otherwise, we may assume without loss of generality that  $F_1(x_1, x_2, x_3) = f_1(x_1, x_2, x_3)$  and the  $x_i$  are all in  $S$  or all in  $T$ . If  $F_1(y_1, y_2, y_3) = f_1(y_1, y_2, y_3)$  we're done since  $f_1$  is edge-preserving; hence we may assume that  $\{y_1, y_2, y_3\}$  intersects both  $S$  and  $T$  and hence every  $x_i$  is in  $T$ . In that case  $F_1(x_1, x_2, x_3)$  is in  $T$  and it is adjacent to  $F_1(y_1, y_2, y_3)\mu_L(y_1, y_2, y_3) \in T$ . This completes the proof for all basic graphs.

The proof for disjoint union is identical to the one in the irreflexive case (Lemma 6).

Finally we show that 4-permutability is preserved under adjunction of a basic graph. Let  $\mathbf{H}_1$  be a basic graph, where  $L_1$  and  $N_1$  denote the set of loops and non-loops of  $\mathbf{H}_1$  respectively, and let  $\mathbf{H}_2$  satisfy our induction hypothesis, and let  $L_2$  and  $N_2$  denote the set of loops and non-loops of  $\mathbf{H}_2$  respectively. We may assume that  $L_1$  is non-empty, and hence it is a clique. Let  $g_1, g_2, g_3$  be operations on  $\mathbf{H}_2$  that satisfy all required identities and property (D). By our earlier analysis, we know there exist operations  $f_1, f_2, f_3$  on the basic graph  $\mathbf{H}_1$  that satisfy all required identities and property (D), and moreover satisfy the following condition (E):

$$\begin{aligned} & \text{If } \{x, y, z\} \text{ intersects } L_1 \text{ and } y \neq z \text{ then} \\ & f_1(x, y, z) \text{ and } f_3(y, z, x) \text{ belong to } L_1. \end{aligned}$$

For convenience of notation, define two ternary operations  $\lambda^L, \lambda^R$  on triples  $(x_1, x_2, x_3)$  such that  $\{x_1, x_2, x_3\}$  intersects the set  $L_1$  by  $\lambda^L(x_1, x_2, x_3) = x_j$  where  $j = \min\{i : x_i \in L_1\}$  and  $\lambda^R(x_1, x_2, x_3) = x_k$  where  $k = \max\{i : x_i \in L_1\}$ . Define two ternary operations  $\nu^L$  and  $\nu^R$  on triples  $(x_1, x_2, x_3)$  such that  $\{x_1, x_2, x_3\}$  intersects the set  $H_2$  by  $\nu^L(x_1, x_2, x_3) = x_j$  where  $j = \min\{i : x_i \in H_2\}$  and  $\nu^R(x_1, x_2, x_3) = x_k$  where  $k = \max\{i : x_i \in H_2\}$ . Notice that  $\lambda^L$  and  $\lambda^R$  are trivially edge-preserving, and so are  $\nu^L$  and  $\nu^R$  if we restrict them to triples  $(x_1, x_2, x_3)$  such that  $\{x_1, x_2, x_3\} \subseteq N_1 \cup H_2$ .

We define operations  $F_1, F_2$  and  $F_3$  on  $\mathbf{H}$  as follows:

$$\begin{aligned} F_1(x_1, x_2, x_3) &= \begin{cases} x_1, & \text{if } x_2 = x_3, \text{ else} \\ f_1(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \subseteq H_1, \text{ else} \\ g_1(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \subseteq H_2, \text{ else} \\ \lambda^L(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \text{ intersects both } L_1 \text{ and } H_2, \\ \nu^L(x_1, x_2, x_3), & \text{otherwise.} \end{cases} \\ F_3(x_1, x_2, x_3) &= \begin{cases} x_3, & \text{if } x_1 = x_2, \text{ else} \\ f_3(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \subseteq H_1, \text{ else} \\ g_3(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \subseteq H_2, \text{ else} \\ \lambda^R(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \text{ intersects both } L_1 \text{ and } H_2, \\ \nu^R(x_1, x_2, x_3), & \text{otherwise.} \end{cases} \\ F_2(x_1, x_2, x_3) &= \begin{cases} f_2(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \subseteq H_1, \text{ else} \\ g_2(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \subseteq H_2, \text{ else} \\ \lambda^L(x_1, x_2, x_3), & \text{if } \{x_1, x_2, x_3\} \text{ intersects both } L_1 \text{ and } H_2, \\ \nu^L(x_1, x_2, x_3), & \text{otherwise.} \end{cases} \end{aligned}$$

It is clear that each  $F_i$  is conservative and that identities (1) and (3) are satisfied. To prove (2), suppose without loss of generality that  $x \neq y$ : if  $\{x, y\}$  is contained in  $H_1$  or contained in  $H_2$  then the result follows from the fact that the  $f_i$  and  $g_i$  satisfy (2); if  $\{x, y\}$  intersects both  $L_1$  and  $H_2$  then  $F_1(x, x, y) = F_2(x, y, y)$  is the unique element in  $\{x, y\}$  that belongs to  $L_1$ ; if  $\{x, y\}$  intersects both  $N_1$  and  $H_2$  then  $F_1(x, x, y) = F_2(x, y, y)$  is the unique element in  $\{x, y\}$  that belongs to  $H_2$ . The proof that  $F_2(x, x, y) = F_3(x, y, y)$  is similar.

Next we prove that property (D) holds for  $F_1$  (the proof for  $F_3$  is identical.) Let  $n$  be a neighbour of  $x_1$ . If  $F_1(x_1, x_2, x_3) = x_1$  the result is trivial, and if  $\{x_1, x_2, x_3\}$  is contained in  $H_1$  or contained in  $H_2$  then the result follows from the fact that the  $f_i$  and  $g_i$  satisfy (D). Suppose now that  $\{x_1, x_2, x_3\}$  intersects both  $L_1$  and  $H_2$ . Then  $F_1(x_1, x_2, x_3) \in L_1$ ; in particular if  $n \in H_2 \cup L_1$  we are done. If on the other hand  $n \in N_1$  then  $x_1 \in L_1$  so  $F_1(x_1, x_2, x_3) = \lambda^L(x_1, x_2, x_3) = x_1$ .

Finally we show that  $F_1$  is edge-preserving (the proof for  $F_2$  and  $F_3$  is identical.) Let  $(x_1, x_2, x_3)$  and  $(y_1, y_2, y_3)$  be adjacent. We analyse the different cases. Without loss of generality we may assume throughout that  $y_2 \neq y_3$ .

(1) Suppose first that  $x_2 = x_3$  so that  $F_1(x_1, x_2, x_3) = x_1$ . (a) If  $\{y_1, y_2, y_3\} \subseteq H_1$ , then  $F_1(y_1, y_2, y_3) = f_1(y_1, y_2, y_3)$ ; either  $x_1 \in H_2$ , forcing  $y_1 \in L_1$  so by property (E) we have that  $F_1(y_1, y_2, y_3) \in L_1$  adjacent to  $x_1$ , or else  $x_1 \in H_1$  and so property (D) guarantees  $F_1(y_1, y_2, y_3)$  adjacent to  $x_1$ . (b) If  $\{y_1, y_2, y_3\} \subseteq H_2$ , then  $F_1(y_1, y_2, y_3) = g_1(y_1, y_2, y_3)$ ; if  $x_1 \in H_1$  then it is in  $L_1$  and is adjacent to  $F_1(y_1, y_2, y_3)$ ; otherwise  $x_1 \in H_2$  and property (D) applies. (c) If  $\{y_1, y_2, y_3\}$  intersects both  $L_1$  and  $H_2$ , then  $F_1(y_1, y_2, y_3)$  returns the leftmost entry which is in  $L_1$ ; hence if  $x_1$  is not in  $N_1$  then it is adjacent to  $F_1(y_1, y_2, y_3)$ . If  $x_1 \in N_1$  then  $y_1 \in L_1$  so  $F_1(y_1, y_2, y_3) = y_1$  and we are done. (d) Suppose that  $\{y_1, y_2, y_3\}$  intersects both  $H_1$  and  $H_2$  but not  $L_1$ : then  $F_1(y_1, y_2, y_3)$  returns the leftmost entry in  $H_2$ ; if  $x_1$  is in  $H_1$  then it must be in  $L_1$ ; otherwise  $x_1 \in H_2$  forces  $y_1 \in H_2$ ; in both cases  $x_1$  is adjacent to  $F_1(y_1, y_2, y_3)$ .

From now on we may assume that  $x_2 \neq x_3$ .

(2) Suppose  $\{x_1, x_2, x_3\} \subseteq H_1$ . (a) If  $\{y_1, y_2, y_3\} \subseteq H_2$ , then  $x_i \in L_1$  for all  $i$  so we're done. (b) If  $\{y_1, y_2, y_3\}$  intersects both  $L_1$  and  $H_2$ , then  $F_1(y_1, y_2, y_3)$  returns the leftmost entry which is in  $L_1$ . There is some  $i$  such that  $y_i \in H_2$ , hence  $x_i \in L_1$ ; by (E) it follows that  $F_1(x_1, x_2, x_3) \in L_1$  and is adjacent to  $F_1(y_1, y_2, y_3)$ . (c) Suppose that  $\{y_1, y_2, y_3\}$  intersects both  $H_1$  and  $H_2$  but not  $L_1$ ; then  $F_1(y_1, y_2, y_3)$  returns the leftmost entry in  $H_2$ . There is some  $i$  such that  $y_i \in H_2$ , hence  $x_i \in L_1$ ; by (E) again  $F_1(x_1, x_2, x_3) \in L_1$  and is adjacent to  $F_1(y_1, y_2, y_3)$ .

(3) Suppose  $\{x_1, x_2, x_3\} \subseteq H_2$ . By the previous cases we may assume that  $\{y_1, y_2, y_3\}$  intersects both  $H_1$  and  $H_2$ ; in this case it must also intersect  $L_1$ , hence  $F_1(y_1, y_2, y_3)$  returns the leftmost entry which is in  $L_1$ , which is adjacent to every vertex in  $H_2$ .

(4) Suppose  $\{x_1, x_2, x_3\}$  intersects both  $L_1$  and  $H_2$ . We may now assume that  $\{y_1, y_2, y_3\}$  intersects both  $H_1$  and  $H_2$  but not  $L_1$ . Then by definition  $F_1(x_1, x_2, x_3) \in L_1$  and  $F_1(y_1, y_2, y_3) \in H_2$ , and hence are adjacent. ■

## 6.5 Symmetric Datalog Constructions

The goal of this section is to provide a proof of

**Lemma 19** *If  $\mathbf{H} \in \mathcal{L}$  then  $\neg\text{CSP}(\mathbf{H}^L)$  is in symmetric Datalog.*

### 6.5.1 An Overview of Datalog and CSP

Datalog is a query and rule language for deductive databases. A Datalog program  $\mathcal{D}$  over a vocabulary  $\tau$  is a finite set of rules of the form  $h \leftarrow b_1 \wedge \dots \wedge b_m$  where  $h$  and each  $b_i$  are atomic formulas  $R_j(v_1, \dots, v_k)$ . We say that  $h$  is the *head* of the rule and that  $b_1 \wedge \dots \wedge b_m$  is its *body*. Relational predicates  $R_j$  which appear in the head of some rule of  $\mathcal{D}$  are called *intensional database predicates (IDBs)* and are not part of the vocabulary  $\tau$ . All other relational predicates are called *extensional database predicates (EDBs)* and are in  $\tau$ .

A rule of  $\mathcal{D}$  is *linear* if its body contains at most one IDB and is *non-recursive* if its body contains only EDBs. A linear but recursive rule is of the form  $I_1(\bar{x}) \leftarrow I_2(\bar{y}) \wedge E_1(\bar{z}_1) \wedge \dots \wedge E_k(\bar{z}_k)$  where  $I_1, I_2$  are IDBs and the  $E_i$  are EDBs<sup>3</sup>. Each such rule has a *symmetric*  $I_2(\bar{y}) \leftarrow I_1(\bar{x}) \wedge E_1(\bar{z}_1) \wedge \dots \wedge E_k(\bar{z}_k)$ . A Datalog program is *non-recursive* if all its rules are non-recursive, *linear* if all its rules are linear and *symmetric* if it is linear and if the symmetric of each recursive rule of  $\mathcal{D}$  is also a rule of  $\mathcal{D}$ . We further say that  $\mathcal{D}$  has *width*  $(j, k)$  if each rule of  $\mathcal{D}$  has at most  $k$  variables and at most  $j$  variables in the head.

A Datalog program  $\mathcal{D}$  takes a  $\tau$ -structure  $\mathbf{A}$  as input and returns a structure  $\mathcal{D}(\mathbf{A})$  over the vocabulary  $\tau' = \tau \cup \{I : I \text{ is an IDB in } \mathcal{D}\}$ . We also want to view a Datalog program as being able to accept or reject an input  $\tau$ -structure and this is achieved by choosing one of the IDB's of  $\mathcal{D}$  as the *goal predicate*: the  $\tau$ -structure  $\mathbf{A}$  is *accepted by*  $\mathcal{D}$  if the goal predicate  $G$  is non-empty in  $\mathcal{D}(\mathbf{A})$ .

The semantics of Datalog are very intuitive and we only illustrate them through an example. A formal definition can be found, for example, in [D05, L04]. Consider the problem of two-coloring. An undirected graph is two-colorable if and only if it contains no cycles of odd length. The following Datalog program  $\mathcal{D}$  defines two-coloring because the goal predicate becomes non-empty if and only if the input graph contains an odd cycle.

$$\begin{aligned} O(x, y) &\leftarrow E(x, y) \\ O(x, y) &\leftarrow O(x, w) \wedge E(w, z) \wedge E(z, y) \\ O(x, w) &\leftarrow O(x, y) \wedge E(w, z) \wedge E(z, y) \\ G &\leftarrow O(x, x) \end{aligned}$$

Here  $E$  is the binary EDB representing the adjacency relation in the input graph,  $O$  is a binary IDB whose intended meaning is “there exists an odd-length path from  $x$  to  $y$ ” and  $G$  is the 0-ary goal predicate. Intuitively, the program first finds a path of length one using the only non-recursive rule and then iteratively finds paths of higher odd lengths using the middle two rules. Whenever the path begins and ends at the same vertex  $x$ , the goal predicate becomes non-empty indicating the presence of a cycle of odd length.

Note that the two middle rules form a symmetric pair. In the above description, we have not included the symmetric of the last rule. In fact, the fairly counterintuitive rule  $O(x, x) \leftarrow G$  can be added to the program without changing the class of structures accepted by the program since the rule only becomes relevant if an odd cycle has already been detected in the graph.

Assume that a program  $\mathcal{D}$  accepts a structure  $\mathbf{A}$ . Intuitively, a *derivation tree* over  $\mathbf{A}$  is a representation of the “proof” that  $\mathcal{D}$  accepts  $\mathbf{A}$ . Consider, for example, the following (linear) Datalog program  $\mathcal{D}$  over the vocabulary consisting of a binary relation symbol  $E$  and two unary

<sup>3</sup>Note that the variables occurring in  $\bar{x}, \bar{y}, \bar{z}_i$  are not necessarily distinct.

relation symbols  $S$  and  $T$ :

$$\begin{aligned} I(y) &\leftarrow S(y) \\ I(y) &\leftarrow I(x) \wedge E(x, y) \\ G &\leftarrow I(y) \wedge T(y) \end{aligned}$$

We choose  $G$  as the goal predicate. One can verify that an input structure  $\mathbf{A}$  is accepted if and only if there is a path in the graph  $\mathbf{A}$  from a vertex in  $S$  to a vertex in  $T$ .

### 6.5.2 Composing Symmetric Datalog Programs

The output of a Datalog program over  $\sigma$  with a set of IDBs  $I$  is a structure over the extended signature  $\sigma \cup I$ . Such a structure can naturally be fed as input to another Datalog program working over this extended signature and using a set  $J$  of IDBs disjoint from  $I$ . The result is a structure over the signature  $\sigma \cup I \cup J$ . Of course, the effect of this composition can be obtained by simply merging the list of rules of the two programs. However, this naive construction does not preserve the linearity or symmetricity of the programs. The following lemma shows that in fact symmetricity can be preserved at the cost of an increase in the arity of the IDBs.

**Lemma 20** *Let  $\mathcal{D}$  be a symmetric Datalog program over the signature  $\sigma = \{E_1, \dots, E_k\}$  and outputting IDBs  $I_1, \dots, I_t$  of respective arities  $a_1, \dots, a_t$ . Let  $\ulcorner I_{j_1} \wedge \dots \wedge I_{j_s} \urcorner$  denote the relation of arity  $a_{j_1} + \dots + a_{j_s}$  which is the cartesian product of the (not necessarily distinct) relations  $I_{j_1}, \dots, I_{j_s}$ . For any  $d$ , there exists a program  $\mathcal{D}_d$  over the signature  $\sigma$  which correctly derives all the  $\ulcorner I_{j_1} \wedge \dots \wedge I_{j_s} \urcorner$  with  $s \leq d$ .*

*Proof:* It suffices to show that this holds when  $d = 2$  since the more general statement can be obtained by iterating the construction detailed below. Note that we consider  $\ulcorner I_k \wedge I_\ell \urcorner$  as a single<sup>4</sup> IDB. We use  $\bar{x}, \bar{y}$  and so on to denote tuples of variables and say that  $\bar{x}, \bar{y}$  are disjoint if they share no variable. We also  $E(\bar{w})$  to denote some conjunction of EDBs. We construct  $\mathcal{D}_2$  with the following rules.

1. Original rules of  $\mathcal{D}$  are kept.
2. If  $I_j(\bar{x}) \leftarrow E(\bar{w})$  is a non-recursive rule in  $\mathcal{D}$ , we include for any  $1 \leq q \leq t$  the rule

$$\ulcorner I_j(\bar{x}) \wedge I_q(\bar{y}) \urcorner \leftarrow I_q(\bar{y}) \wedge E(\bar{w})$$

where  $\bar{y}$  is disjoint from  $\bar{x}$  and  $\bar{w}$ . We also include the symmetric rule

$$I_q(\bar{y}) \leftarrow \ulcorner I_j(\bar{x}) \wedge I_q(\bar{y}) \urcorner \wedge E(\bar{w}).$$

3. Finally, if  $I_j(\bar{x}) \leftarrow I_k(\bar{y}) \wedge E(\bar{w})$  is a recursive rule of  $\mathcal{D}$ , we include for any  $1 \leq q \leq t$  the rule

$$\ulcorner I_q(\bar{z}) \wedge I_j(\bar{x}) \urcorner \leftarrow \ulcorner I_q(\bar{z}) \wedge I_k(\bar{y}) \urcorner \wedge E(\bar{w})$$

where  $\bar{z}$  is disjoint from  $\bar{x}, \bar{y}$  and  $\bar{w}$ . Because  $\mathcal{D}$  is symmetric, we know that the symmetric of the above rule also appears in  $\mathcal{D}_2$ .

---

<sup>4</sup>As stated, the lemma distinguishes the IDBs  $\ulcorner I_j \wedge I_k \urcorner$  and  $\ulcorner I_k \wedge I_j \urcorner$ . However, the two are clearly equivalent from a computation perspective. To simplify our description of  $\mathcal{D}_2$ , we thus implicitly assume that any rule involving  $\ulcorner I_j \wedge I_k \urcorner$  is accompanied by the counterpart rule using  $\ulcorner I_k \wedge I_j \urcorner$ .

By construction  $\mathcal{D}_2$  is symmetric. We claim that it computes the product relations correctly. Note first that all rules above are sound, i.e. there is a derivation in  $\mathcal{D}_2$  for  $\lceil I_j(\bar{x}) \wedge I_k(\bar{y}) \rceil$  only if there are derivations for  $I_j(\bar{x})$  and  $I_k(\bar{y})$  in  $\mathcal{D}$ . Indeed, one can verify that if this property is true at some stage of the derivation then it still holds after the application of any of the above rules. The same argument shows that  $\mathcal{D}$  and  $\mathcal{D}_2$  derive the same IDBs  $I_1, \dots, I_t$ . In fact, it is convenient to view the execution of  $\mathcal{D}_2$  as a two-stage process where the original IDBs are derived first.

It remains to show that if there are derivations in  $\mathcal{D}$  for  $I_j(\bar{x})$  and  $I_k(\bar{y})$  then there is a derivation of  $\lceil I_j(\bar{x}) \wedge I_k(\bar{y}) \rceil$  in  $\mathcal{D}_2$ . Let  $\rightarrow I_{j_1}(\bar{x}_1) \rightarrow \dots \rightarrow I_{j_n}(\bar{x}_n) \rightarrow I_j(\bar{x})$  denote the sequence of IDBs used in the derivation of  $I_j(\bar{x})$  in  $\mathcal{D}$ . Since there is a derivation of  $I_k(\bar{y})$  in  $\mathcal{D}_2$ , we can use a rule of the second type above to derive  $\lceil I_{j_1}(\bar{x}_1) \wedge I_k(\bar{y}) \rceil$ . From there, the rules of the third type can successively derive  $\lceil I_{j_2}(\bar{x}_2) \wedge I_k(\bar{y}) \rceil$ , then  $\lceil I_{j_3}(\bar{x}_3) \wedge I_k(\bar{y}) \rceil$  and so on, thus yielding the desired derivation for  $\lceil I_j(\bar{x}) \wedge I_k(\bar{y}) \rceil$ .  $\blacksquare$

**Lemma 21** *Let  $\mathcal{D}$  be a symmetric Datalog program over the signature  $\sigma = \{E_1, \dots, E_q\}$  and with a set of IDBs  $I = \{I_1, \dots, I_s\}$  where each  $I_k$  has arity  $a_k$ . Further let  $\mathcal{E}$  be a symmetric Datalog program over the signature  $\sigma' = \sigma \cup I$  and with IDBs  $J_1, \dots, J_t$  with respective arities  $b_1, \dots, b_t$ . For a  $\sigma$ -structure  $H$ , let  $H'$  denote the  $\sigma'$ -structure defined by  $\mathcal{D}(H)$ . One can construct a symmetric Datalog program  $\mathcal{F}$  over the signature  $\sigma$  which includes the IDBs of  $I$  and  $J$  and such that  $I_k(\bar{x})$  is derived in  $\mathcal{F}(H)$  iff  $I_k(\bar{x})$  is derived in  $\mathcal{D}(H)$  and  $J_\ell(\bar{x})$  is derived in  $\mathcal{F}(H)$  iff  $J_\ell(\bar{x})$  is derived in  $\mathcal{E}(H')$ .*

*Proof:* Let  $d$  be such that each rule of  $\mathcal{E}$  uses at most  $d$  EDBs in  $\sigma' - \sigma$ , i.e. at most  $d$  of the IDBs of  $\mathcal{D}$ . By the previous lemma, we can assume that  $d = 1$  since we can otherwise construct  $\mathcal{D}_d$  and thus get relations that represent any conjunction of the  $I_j$ . Our symmetric program  $\mathcal{F}$  includes as IDBs those in  $I$ ,  $J$  and  $K$  as well as IDBs of the form  $\lceil J_\ell \wedge I_k \rceil$  for any  $1 \leq k \leq s$  and  $1 \leq \ell \leq t$ . The rules of  $\mathcal{F}$  are constructed as follows.

1. Every rule of  $\mathcal{D}$  is kept.
2. All rules of  $\mathcal{E}$  which use only EDBs of the original signature  $\sigma$  are kept.
3. For any non-recursive rule of  $\mathcal{D}$ , say  $I_k(\bar{x}) \leftarrow E(\bar{w})$ , we include for each  $1 \leq \ell \leq t$  the symmetric pair of rules

$$\begin{aligned} \lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil &\leftarrow J_\ell(\bar{y}) \wedge E(\bar{w}) \\ J_\ell(\bar{y}) &\leftarrow \lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil \wedge E(\bar{w}) \end{aligned}$$

where  $\bar{y}$  is disjoint from  $\bar{x}$  and  $\bar{w}$ .

4. For any recursive rule of  $\mathcal{D}$ , say  $I_{k_1}(\bar{x}_1) \leftarrow I_{k_2}(\bar{x}_2) \wedge E(\bar{w})$ , we include for each  $1 \leq \ell \leq t$  the rule

$$\lceil J_\ell(\bar{y}) \wedge I_{k_1}(\bar{x}_1) \rceil \leftarrow \lceil J_\ell(\bar{y}) \wedge I_{k_2}(\bar{x}_2) \rceil \wedge E(\bar{w})$$

where  $\bar{y}$  is disjoint from  $\bar{x}_1, \bar{x}_2$  and  $\bar{w}$ . The symmetricity of  $\mathcal{D}$  ensures that  $\mathcal{F}$  contains the symmetric rule:

$$\lceil J_\ell(\bar{y}) \wedge I_{k_2}(\bar{x}_2) \rceil \leftarrow \lceil J_\ell(\bar{y}) \wedge I_{k_1}(\bar{x}_1) \rceil \wedge E(\bar{w})$$

5. For any non-recursive rule of  $\mathcal{E}$  that uses one of the  $I_k$  as an EDB, say  $J_\ell(\bar{y}) \leftarrow I_k(\bar{x}) \wedge E(\bar{w})$ , we include the symmetric pair of (recursive) rules

$$\begin{aligned} \lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil &\leftarrow I_k(\bar{x}) \wedge E(\bar{w}) \\ I_k(\bar{x}) &\leftarrow \lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil \wedge E(\bar{w}) \end{aligned}$$

6. For any non-recursive rule of  $\mathcal{E}$  that does not use any of the  $I_k$  as an EDB, say  $J_\ell(\bar{y}) \leftarrow E(\bar{w})$ , we include for each  $1 \leq k \leq s$  the symmetric pair of (recursive) rules

$$\begin{aligned} \lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil &\leftarrow I_k(\bar{x}) \wedge E(\bar{w}) \\ I_k(\bar{x}) &\leftarrow \lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil \wedge E(\bar{w}) \end{aligned}$$

7. For any recursive rule of  $\mathcal{E}$  that use some  $I_k$  as an EDB, say  $J_{\ell_1}(\bar{y}_1) \leftarrow J_{\ell_2}(\bar{y}_2) \wedge I_k(\bar{x}) \wedge E(\bar{w})$ , we include the rule.

$$\lceil J_{\ell_1}(\bar{y}_1) \wedge I_k(\bar{x}) \rceil \leftarrow \lceil J_{\ell_2}(\bar{y}_2) \wedge I_k(\bar{x}) \rceil \wedge E(\bar{w})$$

Because  $\mathcal{E}$  is symmetric, we know that  $\mathcal{F}$  also includes the rule

$$\lceil J_{\ell_2}(\bar{y}_2) \wedge I_k(\bar{x}) \rceil \leftarrow \lceil J_{\ell_1}(\bar{y}_1) \wedge I_k(\bar{x}) \rceil \wedge E(\bar{w}).$$

8. For any recursive rule of  $\mathcal{E}$  that does not use an  $I_j$  as an EDB, say  $J_{\ell_1}(\bar{y}_1) \leftarrow J_{\ell_2}(\bar{y}_2) \wedge E(\bar{w})$ , we include for each  $1 \leq k \leq s$  the rule.

$$\lceil J_{\ell_1}(\bar{y}_1) \wedge I_k(\bar{x}) \rceil \leftarrow \lceil J_{\ell_2}(\bar{y}_2) \wedge I_k(\bar{x}) \rceil \wedge E(\bar{w})$$

where  $\bar{x}$  is disjoint from  $\bar{y}_1, \bar{y}_2$  and  $\bar{w}$ . Because  $\mathcal{E}$  is symmetric, we know that  $\mathcal{F}$  also includes the rule

$$\lceil J_{\ell_2}(\bar{y}_2) \wedge I_k(\bar{x}) \rceil \leftarrow \lceil J_{\ell_1}(\bar{y}_1) \wedge I_k(\bar{x}) \rceil \wedge E(\bar{w}).$$

We claim that  $\mathcal{F}$  has the desired properties. Again, we first note that all the rules are sound: if there is a derivation in  $\mathcal{F}(H)$  for  $I_k(\bar{x})$  (resp.  $J_\ell(\bar{y})$ ;  $\lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil$ ) then there is a derivation for  $I_k(\bar{x})$  in  $\mathcal{D}(H)$  (resp. a derivation for  $J_\ell(\bar{y})$  in  $\mathcal{E}(H')$ ; derivations for  $I_k(\bar{x})$  in  $\mathcal{D}(H)$  and for  $J_\ell(\bar{y})$  in  $\mathcal{E}(H')$ ). In other words, none of the above rules can produce unwanted tuples.

It remains to show that  $\mathcal{F}$  is complete, i.e. that if there exists a derivation for  $I_k(\bar{x})$  in  $\mathcal{D}(H)$  and a derivation for  $J_\ell(\bar{y})$  in  $\mathcal{E}(H')$  then there are derivations in  $\mathcal{F}(H)$  for  $I_k(\bar{x})$ ,  $J_\ell(\bar{y})$  and  $\lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil$ . This is obvious for  $I_k(\bar{x})$  since the original rules of  $\mathcal{D}$  are also rules of  $\mathcal{F}$ . Similarly, rules of type 2 yield the claim if the derivation of  $J_\ell(\bar{y})$  in  $\mathcal{E}(H')$  never uses one of the  $I_j$  as an EDB. The crux of the argument rests on the fundamental property of symmetric programs: if, say, in  $\mathcal{D}(H)$  we have a derivation from some  $I_{k_1}(\bar{x}_1)$  to some  $I_{k_2}(\bar{x}_2)$  then we can also derive  $I_{k_1}(\bar{x}_1)$  from  $I_{k_2}(\bar{x}_2)$ .

Notice that the rules of type 3 guarantee that if there is a one-step derivation of  $I_k(\bar{x})$  in  $\mathcal{D}(H)$  and a derivation in  $\mathcal{F}(H)$  of some  $J_\ell(\bar{y})$  then there exists a derivation for  $\lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil$  in  $\mathcal{F}(H)$ . The rules of type 4 are then sufficient to show that once we have found a derivation in  $\mathcal{F}(H)$  for some  $J_\ell(\bar{y})$  we can construct one for any  $\lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil$  as long as  $I_k(\bar{x})$  can be derived in  $\mathcal{D}(H)$ . But the rules of type 3 can be used symmetrically and so if  $I_k(\bar{x})$  can be derived in  $\mathcal{D}(H)$  then, in  $\mathcal{F}(H)$ , we can derive  $J_\ell(\bar{y})$  from  $\lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil$ . To complete the argument, suppose there is a derivation of  $J_\ell(\bar{y})$  in  $\mathcal{E}(H')$  of length  $n$ . If  $n = 1$ , i.e. is derived from a first-order rule. By using a rule of type 5 or 6 and the argument above, we know that there are derivations in  $\mathcal{F}(H)$  for all  $\lceil J_{\ell_1}(\bar{y}_1) \wedge I_k(\bar{x}) \rceil$ . We can now proceed inductively: if  $n + 1 \geq 2$ , and if the last step in the derivation of  $J_\ell(\bar{y})$  is given by  $J_\ell(\bar{y}) \leftarrow J_{\ell_n}(\bar{y}_n) \wedge I_{k_n}(\bar{x}_n) \wedge E(\bar{w}_n)$ , we know by the induction hypothesis that there is a derivation in  $\mathcal{F}(H)$  for  $\lceil J_{\ell_n}(\bar{y}_n) \wedge I_{k_n}(\bar{x}_n) \rceil$  so the rules of type 7 yield a derivation in  $\mathcal{F}(H)$  for  $\lceil J_\ell(\bar{y}) \wedge I_k(\bar{x}) \rceil$ .  $\blacksquare$

### 6.5.3 Symmetric Programs for List-Homomorphism of Graphs in $\mathcal{L}$

Our objective is to show that for any undirected graph  $H$  in  $\mathcal{L}$  the set  $\neg\text{CSP}(H^L)$  of graphs with  $H$ -lists that do not map homomorphically to  $H^L$  is definable in symmetric Datalog. We proceed by induction on the structure of  $H$ , i.e. using the inductive definition of  $\mathcal{L}$ . If  $H$  consists of a single loop or non-loop,  $\neg\text{CSP}(H^L)$  is trivially definable in symmetric Datalog and it remains to show that this property is preserved by the operators disjoint union, basic graph adjunction, formation of a basic graph by completion of a color class and special sum.

We begin with simple but useful observations that allow more concise and intuitive descriptions of our constructions. These remarks and basic tricks all rely on Lemma 21.

1. In a number of constructions below, we want to obtain from two symmetric Datalog programs  $\mathcal{D}_1, \mathcal{D}_2$  with goal predicates  $T_1, T_2$  a new symmetric program  $\mathcal{D}$  which accepts an input  $G$  if  $G$  is accepted by  $\mathcal{D}_1$  or if  $G$  is accepted by  $\mathcal{D}_2$ . This can be done effortlessly since we can simply take the union of the rules of  $\mathcal{D}_1$  and  $\mathcal{D}_2$ , create a new goal predicate  $T$  and include the rules  $T \leftarrow T_1$  and  $T \leftarrow T_2$ .

If instead we want  $\mathcal{D}$  to accept  $G$  if both  $\mathcal{D}_1$  and  $\mathcal{D}_2$  accept  $G$ , we can use Lemma 21 as follows. Consider the relational structure output by  $\mathcal{D}_1$ : this structure includes the relation  $T_1$  which we can now use as an EDB in  $\mathcal{D}$ . It now suffices to include in  $\mathcal{D}$  the rules of  $\mathcal{D}_2$  and including  $T_1$  in the body of each rule. In other words, these rules only start working if  $G$  is accepted by  $\mathcal{D}_1$ .

2. Consider Datalog programs over a signature  $\sigma$  that contains a binary relation  $E$  (this is the case in all our constructions). It is straightforward to construct in symmetric Datalog a  $k$ -ary relation  $C_E$  which contains the tuple  $(x_1, \dots, x_k)$  iff all  $x_i$  are in the same symmetric connected component of  $E$ . Moreover, Lemma 21 allows us to assume that any program  $\mathcal{D}$  has access to this relation as an EDB.

Suppose that the body of each rule in  $\mathcal{D}$  includes the EDB condition  $C_E(x_1, \dots, x_k)$  where the  $x_i$  are the variables occurring in the rule. If  $G$  is a graph given as input to  $\mathcal{D}$ , note that any derivation of  $\mathcal{D}(G)$  must take place within a single connected component of  $G$ . So the graphs accepted by  $\mathcal{D}$  are disjoint unions of *connected graphs*, one of which is accepted by  $\mathcal{D}$ . In our case, we construct Datalog programs which accept a graph  $G$  with  $H$ -lists iff there is no homomorphism from  $G^L$  to  $H^L$  and this of course holds iff there is some connected component of  $G$  that does not map to  $H^L$ . When we prove correctness of a given program, we can therefore assume without loss of generality that the input structure is connected.

3. Let  $G$  be a graph with  $H$ -lists. Any  $v \in G$  is potentially bound by more than one unary predicate but of course this amounts to imposing a list on  $v$  which is the intersection of all such unary predicates. Clearly, we can construct a simple symmetric Datalog program which returns a graph  $G'$  with  $H$ -lists over the same set of vertices but such that every  $v$  is bound by a unique list  $L_v$ . In the same vein, if  $T$  is a subset of vertices of  $H$  and if  $G$  is a graph with  $H$ -lists, we can construct in symmetric Datalog a graph  $G^{\cap T}$  in which every vertex  $v$  is bound by a list  $L'_v$  which is the intersection of the original  $L_v$  with  $T$ .

Furthermore, in the constructions below we typically assume that symmetric programs  $\mathcal{D}_1, \mathcal{D}_2$  exist for  $\neg\text{CSP}(H_1^L)$  and  $\neg\text{CSP}(H_2^L)$  and construct a symmetric program  $\mathcal{D}$  for  $\neg\text{CSP}(H^L)$  where  $H$  is a graph obtained by combining  $H_1$  and  $H_2$  through some operator. Strictly speaking, the inputs of  $\mathcal{D}$  are graphs with  $H$ -lists and thus cannot be fed as inputs to  $\mathcal{D}_1$  or  $\mathcal{D}_2$  since they only deal with lists contained in  $H_1$  and  $H_2$  respectively. Note however that

$G^{\cap H_1}$  can be used as an input to  $\mathcal{D}_1$  and we use this trick repeatedly. If need be, we can also use symmetric programs to construct graphs  $G_1, G_2$  with, respectively,  $H_1$ -lists and  $H_2$ -lists and new edge relations denoted  $E_1$  and  $E_2$  respectively. Since we can now modify the rules of  $\mathcal{D}_1$  and  $\mathcal{D}_2$  by replacing the occurrences of  $E$  by  $E_1$  and  $E_2$ . By our first remark, we can then construct a symmetric program  $\mathcal{D}$  that accepts  $G$  iff  $\mathcal{D}_1$  accepts  $G_1$  and  $\mathcal{D}_2$  accepts  $G_2$  or a program  $\mathcal{D}$  that accepts  $G$  iff  $\mathcal{D}_1$  accepts  $G_1$  or  $\mathcal{D}_2$  accepts  $G_2$ .

Let us start with the simplest inductive step.

**Lemma 22** *If  $\neg \text{CSP}(H_1^L)$  and  $\neg \text{CSP}(H_2^L)$  are definable in symmetric Datalog then for the disjoint union  $H$  of  $H_1$  and  $H_2$  we also have  $\neg \text{CSP}(H^L)$  definable in symmetric Datalog.*

*Proof:* Suppose  $\neg \text{CSP}(H_1^L)$  and  $\neg \text{CSP}(H_2^L)$  are recognized by symmetric Datalog programs  $\mathcal{D}_1, \mathcal{D}_2$  with respective goal predicates  $K_1, K_2$ . If  $H$  is the disjoint union of  $H_1$  and  $H_2$ , it is clear that a connected graph  $G$  with  $H$ -lists maps into  $H^L$  iff  $G$  maps to  $H_1^L$  or to  $H_2^L$ . Of course  $G$  maps to  $H_i^L$  iff  $G^{\cap H_i}$  does. In other words,  $G$  does not map into  $H^L$  iff  $\mathcal{D}_1$  accepts  $G^{\cap H_1}$  and  $\mathcal{D}_2$  accepts  $G^{\cap H_2}$ . As we noted in remark ??, this can be tested within a single symmetric program  $\mathcal{D}$ . ■

**Lemma 23** *Let  $H_1$  be an irreflexive basic graph with color classes  $B$  and  $T$  and assume that  $\neg \text{CSP}(H_1^L)$  is recognized by symmetric Datalog programs  $\mathcal{D}_1$ . If  $H$  is the graph obtained by transforming  $T$  into a reflexive clique then  $\neg \text{CSP}(H^L)$  is also definable in symmetric Datalog.*

*Proof:* Let  $G$  be a graph with  $H$ -lists. Suppose  $h$  is a homomorphism from  $G^L$  to  $H^L$  exists. Because  $H_1$  is bipartite, any  $g \in G$  mapped to some  $b \in B$  must have its neighbors mapped to  $T$ . So if  $g$  has some element  $t$  of  $T$  in its list we still have a homomorphism if we set  $h(g) = t$ . We can therefore assume that all lists of  $G$  are either contained in  $T$  or contained in  $B$  and, more precisely, we can use a symmetric Datalog program to trim the lists of  $G$  accordingly. Denote as  $G_T$  and  $G_B$  the resulting partition of  $G$ 's vertices. If  $G_B$  contains a loop there can be no homomorphism from  $G^L$  to  $H^L$  and this condition is trivial to check in symmetric Datalog. Let  $G'$  be the graph obtained by deleting from  $G$  all edges between vertices in  $G_T$ . One can verify that  $G^L$  maps to  $H^L$  iff  $G'$  maps to  $H_1^L$ . It remains to show that we can construct a symmetric program  $\mathcal{D}$  which, on input  $G^L$ , outputs  $G'$ . We have already shown that  $G^L$  can be assumed to have lists either contained in  $B$  or  $T$  and, accordingly, we can assume that  $\mathcal{D}$  is given EDBs  $G_T, G_B$ . The edge relation of  $G'$  can now be defined with the non-recursive rules  $E'(x, y) \leftarrow E(x, y) \wedge G_T(x) \wedge G_B(y)$  and  $E'(x, y) \leftarrow E(x, y) \wedge G_T(y) \wedge G_B(x)$ . ■

**Lemma 24** *Let  $H_1$  be a basic graph and assume that  $\neg \text{CSP}(H_1^L)$  and  $\neg \text{CSP}(H_2^L)$  are recognized by symmetric Datalog programs  $\mathcal{D}_1, \mathcal{D}_2$ . If  $H$  is the result of adjoining  $H_1$  to  $H_2$  then  $\neg \text{CSP}(H^L)$  is also definable in symmetric Datalog.*

*Proof:* Let  $R_i, I_i$  respectively denote the set of loops and non-loops of each  $H_i$ . Recall that the adjunction of  $H_1$  to  $H_2$  is the graph obtained by taking the union of the two graphs and adding every edge from  $R_1$  to  $H_2$ . Moreover, because  $H_1$  is basic, its loops  $R_1$  form a clique and there are no edges between  $I_1$  and  $H_2$ . Let  $G$  be a graph with  $H$ -lists and consider some vertex  $g$  whose list contains some element of  $r \in R_1$ . If there is any homomorphism from  $G$  to  $H^L$  then there is one such that  $h(g) \in R_1$ . Indeed, if  $u \in G$  is such that  $h(u) \in I_1$  then if  $(u, g)$  is an edge in  $G$  we must have  $h(g) \in R_1$  and if  $h(u) \notin I_1$  then  $h(u)$  has an edge to any  $r \in R_1$ . Similarly, if the list of  $g$

contains no element of  $R_1$  but contains elements of both  $I_1$  and  $H_2$ , then we can assume that  $h$  maps  $g$  to some element of  $H_2$  because mapping  $g$  to  $I_1$  forces its neighbors to be mapped in  $R_1$ . These observations allow us to assume that each list in  $G^L$  is either contained in  $R_1$ , contained in  $I_1$  or contained in  $H_2$ . We can use a symmetric program to trim the lists accordingly and obtain unary predicates  $r_1, \ell_1, h_2$  that represent this partition.

Let  $G_1, G_2$  be the subgraphs of  $G$  induced respectively by vertices in  $r_1 \cup \ell_1$  and vertices in  $h_2$ . It is now clear that  $G^L$  maps to  $H^L$  iff  $G_1^L$  maps to  $H_1^L$  and  $G_2^L$  maps to  $H_2^L$  and, as in the previous proofs,  $G_1$  and  $G_2$  can be constructed in symmetric Datalog.  $\blacksquare$

**Lemma 25** *Let  $H_1$  and  $H_2$  be irreflexive graphs such that  $\neg \text{CSP}(H_1^L)$  and  $\neg \text{CSP}(H_2^L)$  are recognized by symmetric Datalog programs  $\mathcal{D}_1, \mathcal{D}_2$ . If  $H$  is the special sum of  $H_1$  and  $H_2$  then  $\neg \text{CSP}(H^L)$  is also definable in symmetric Datalog.*

*Proof:* Recall that the special sum of bipartite irreflexive graphs  $H_1$  and  $H_2$  with color classes  $B_i, T_i$  consists of the disjoint union of the graphs in which all edges between  $B_1$  and  $T_2$  are added. Note first that  $G$  must be bipartite in order to map to  $H$  and since bipartiteness can be checked in symmetric Datalog, we can assume that any input  $G$  is indeed bipartite. More importantly, we can construct a symmetric Datalog program that outputs unary relation  $B_G, T_G$  giving the color classes of  $G$  and use them as EDBs in the sequel. Any homomorphism from  $G$  to  $H$  must either map all  $g \in B_G$  to  $B_1 \cup B_2$  and all  $g \in T_G$  to  $T_1 \cup T_2$  or map all  $g \in B_G$  to  $T_1 \cup T_2$  and all  $g \in T_G$  to  $B_1 \cup B_2$ . To check if  $G^L$  is in  $\neg \text{CSP}(H^L)$  it suffices to verify that neither of these options is viable. We show how to rule out the existence of a homomorphism  $h$  mapping  $B_G$  to  $B_1 \cup B_2$  and  $T_G$  to  $T_1 \cup T_2$ : the other case is handled symmetrically (no pun intended). First we construct a graph  $G'$  by trimming the lists of  $G^L$ , i.e. by intersecting the list of any  $g \in T_G$  with  $T_1 \cup T_2$  and the list of any  $g \in B_G$  with  $B_1 \cup B_2$ . We claim that if  $h$  is a homomorphism from  $G'^L$  to  $H^L$  and if the list of  $g \in T_G$  contains elements from both  $T_1$  and  $T_2$  then there exists a homomorphism  $h'$  which maps  $g$  to some element of  $T_1$ . Indeed, every  $v \in T_2$  only has neighbors in  $B_2$ , all of which are connected to every vertex in  $T_1$ . We can therefore trim our lists further so that every list of a  $g \in T_G$  is either contained in  $T_1$  or contained in  $T_2$ . Similarly, we trim the lists so that every list of a  $g \in B_G$  is either contained in  $B_1$  or contained in  $B_2$ . We can therefore construct in symmetric Datalog a graph  $G''$  with  $H$ -lists contained in one of the four  $B_i, T_i$ . Since there are edges in  $H$  between any vertex in  $T_1$  and any vertex in  $B_2$  so we can safely ignore the edges in  $G''$  that connect vertices whose lists are respectively contained in  $T_1$  and  $B_2$ . On the other hand, if  $G''$  contains an edge between vertices whose lists are respectively contained in  $B_1$  and  $T_2$  then there can be no homomorphism from  $G''$  to  $H^L$ . With this possibility ruled out, construct in symmetric Datalog the graphs  $G_1$  and  $G_2$  induced by the vertices whose lists are in  $T_1 \cup B_1$  and  $T_2 \cup B_2$  respectively. There is a homomorphism from  $G''$  to  $H$  iff  $G_1$  maps to  $H_1^L$  and  $G_2$  maps to  $H_2^L$  and this can be checked using  $\mathcal{D}_1$  and  $\mathcal{D}_2$  respectively.  $\blacksquare$

## 6.6 Remarks on the Metaproblem

**Theorem 26** *There is a polynomial-time algorithm to determine whether, for a given graph  $\mathbf{H}$ , the list homomorphism problem for  $\mathbf{H}$  is NP-complete (not definable in Datalog), NL-complete (in linear Datalog but not symmetric Datalog), L-complete or is first-order definable.*

*Proof:* It is known (see [FHH99]) that  $\mathbf{H}$  is a bi-arc graph if and only if the complement of the product of  $\mathbf{H}$  with the edge is a circular arc graph; such graphs can be recognised in linear time [M03]. Now assume that  $\mathbf{H}$  is a bi-arc graph. It is clear that the forbidden substructure definition of the class  $\mathcal{L}$  in gives us a logspace algorithm to recognise them; finally those graphs whose list homomorphism problem is first-order definable can be recognised in polynomial time by Lemma 12. ■

### 6.7 Figures for the body

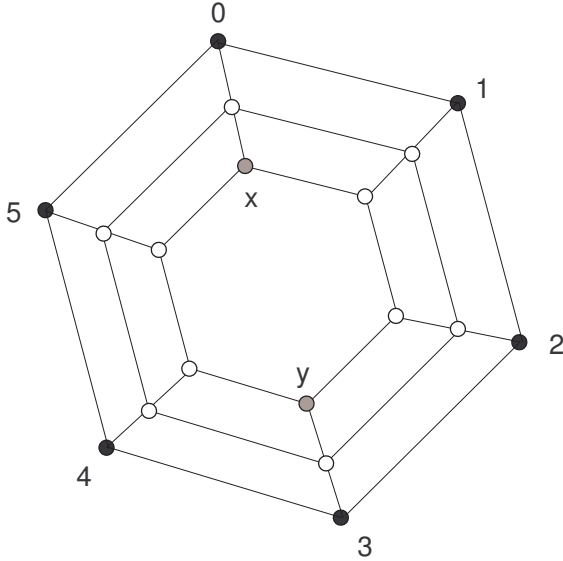


Figure 6: The graph  $\mathbf{K}$  in the construction of the relation  $\alpha$ .

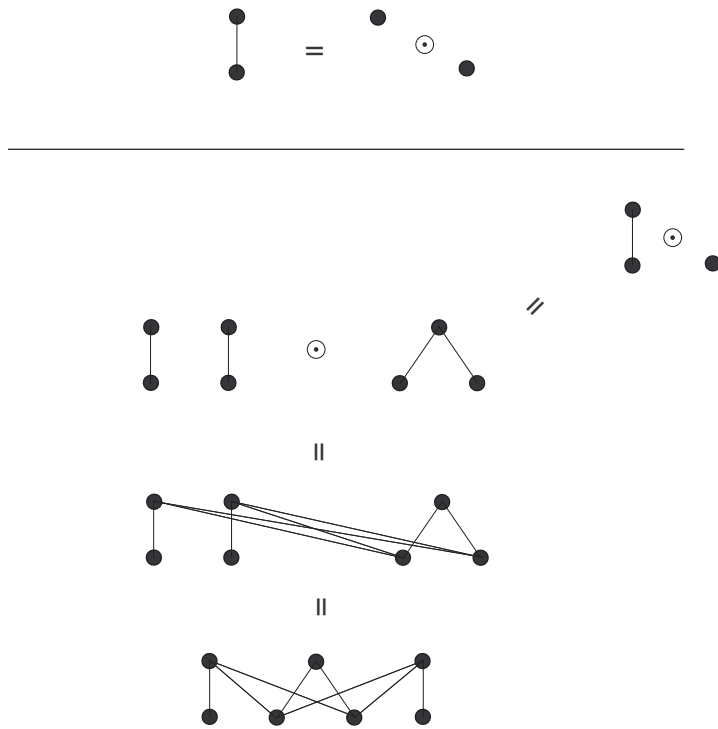


Figure 7: The edge as a special sum of two one-element graphs; and a basic graph on 7 vertices.

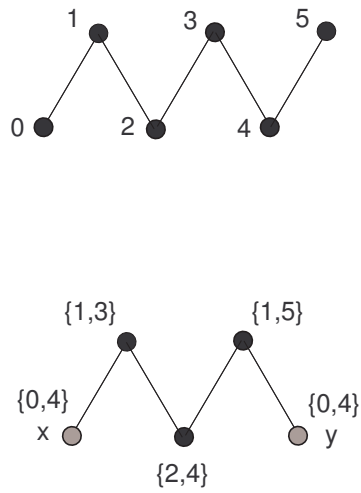


Figure 8: The irreflexive path of length 5 and the graph  $\mathbf{M}$  in the construction of the order relation.

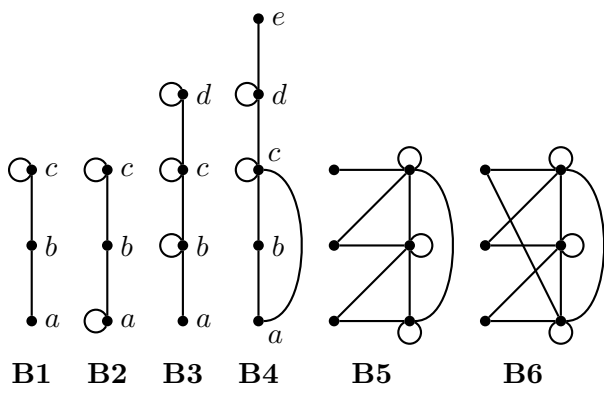


Figure 9: The forbidden mixed graphs.