

# Graph Homomorphisms and Universal Algebra

## Course Notes

Manuel Bodirsky,  
Institut für Algebra, TU Dresden,  
Manuel.Bodirsky@tu-dresden.de

November 13, 2021

**Disclaimer:** these are course notes in draft state, and they probably contain many mistakes; please report them to *manuel.bodirsky@tu-dresden.de*.

## Contents

<b>1</b>	<b>The Basics</b>	<b>3</b>
1.1	Graphs and Digraphs . . . . .	3
1.2	Graph Homomorphisms . . . . .	5
1.3	The $H$ -colouring Problem and Variants . . . . .	8
1.4	Cores . . . . .	10
1.5	Polymorphisms . . . . .	13
<b>2</b>	<b>The Arc-consistency Procedure</b>	<b>13</b>
2.1	The Power Graph . . . . .	16
2.2	Tree Duality . . . . .	17
2.3	Totally Symmetric Polymorphisms . . . . .	19
2.4	Semilattice Polymorphisms . . . . .	20
<b>3</b>	<b>The Path-consistency Procedure</b>	<b>22</b>
3.1	Majority Polymorphisms . . . . .	23
3.2	Testing for Majority Polymorphisms . . . . .	26
3.3	Digraphs with a Maltsev Polymorphism . . . . .	27
<b>4</b>	<b>Logic</b>	<b>30</b>
4.1	Primitive positive formulas . . . . .	31
4.2	From Structures to Formulas . . . . .	32
4.3	From Formulas to Structures . . . . .	32
4.4	Primitive Positive Definability . . . . .	33
4.5	Cores and Constants . . . . .	34
4.6	Primitive Positive Interpretations . . . . .	35
4.7	Reduction to Binary Signatures . . . . .	39
4.8	The Structure-Building Operators $H$ , $C$ , and $I$ . . . . .	40

<b>5</b>	<b>Relations and Operations</b>	<b>43</b>
5.1	Operation Clones . . . . .	43
5.2	Inv-Pol . . . . .	43
5.3	Essentially Unary Clones . . . . .	44
5.4	Minimal Clones . . . . .	45
5.5	Schaefer's Theorem . . . . .	49
<b>6</b>	<b>Maltsev Polymorphisms</b>	<b>52</b>
6.1	Examples . . . . .	52
6.2	Compact Representations of Relations . . . . .	53
6.3	The Bulatov-Dalmau Algorithm . . . . .	54
<b>7</b>	<b>Universal Algebra</b>	<b>58</b>
7.1	Algebras and Clones . . . . .	58
7.2	Subalgebras, Products, Homomorphic Images . . . . .	60
7.3	Pseudovarieties and Varieties . . . . .	62
7.4	Birkhoff's Theorem . . . . .	63
7.5	(Abstract) Clones . . . . .	65
7.6	Taylor Terms . . . . .	66
7.7	The Tractability Conjecture . . . . .	71
<b>8</b>	<b>Undirected Graphs</b>	<b>73</b>
8.1	The Hell-Nešetřil Theorem . . . . .	73
8.2	Siggers Operations of Arity 6 . . . . .	76
<b>9</b>	<b>Congruence Lattices</b>	<b>77</b>
9.1	Congruence Permutability . . . . .	78
9.2	Congruence Distributivity . . . . .	78
9.3	Congruence Modularity . . . . .	82
9.4	Congruence Semidistributivity . . . . .	84
9.5	Abelian Algebras . . . . .	84
<b>10</b>	<b>Absorption</b>	<b>86</b>
10.1	Absorption Transfer . . . . .	86
10.2	The Absorption Theorem . . . . .	87
10.3	Hereditarily Absorption-Free Algebras . . . . .	87
10.4	$n$ -Absorption . . . . .	88
10.5	Near Unanimity Polymorphisms . . . . .	88
<b>11</b>	<b>Cyclic Polymorphisms</b>	<b>88</b>
11.1	Cyclic Terms . . . . .	88
11.2	Cyclic Relations . . . . .	89
11.3	Digraphs without Sources and Sinks . . . . .	91
11.4	The Cyclic Term Theorem . . . . .	91
11.5	Siggers Operations of Arity 4 . . . . .	92
<b>12</b>	<b>Open Problems</b>	<b>92</b>

**Prerequisites.** This course is designed for students of mathematics or computer science that already had an introduction to discrete structures. Almost all notions that we use in this text will be formally introduced, with the notable exception of basic concepts from complexity theory. For example, we do not formally introduce the class of polynomial-time computable functions and NP-completeness, even though these concepts are used when we discuss computational aspects of graph homomorphisms. Here we refer to an introduction to the theory of computation as for instance the book of Papadimitriou [63]. Many thanks to Mark Siggers and the participants of the course in the Corona spring semester 2020 for their bug reports.

The text contains 114 exercises; the ones with a star are harder.

## 1 The Basics

### 1.1 Graphs and Digraphs

The concepts in this section are probably known to most students, and can safely be skipped; the section fixes standard terminology and conventions from graph theory and can be consulted later if needed. Almost all definitions in this section have generalisations to *relational structures*, which will be introduced in Section 4; however, we focus exclusively on graphs in this section since they allow to reach the key ideas of the underlying theory with a minimum of notation.

A *directed graph* (also *digraph*)  $G$  is a pair  $(V, E)$  of a set  $V = V(G)$  of *vertices* and a binary relation  $E = E(G)$  on  $V$ . Note that in general we allow that  $V$  is an infinite set. For some definitions and results, we require that  $V$  is finite, in which case we say that  $G$  is a *finite digraph*. However, since this course deals exclusively with finite digraphs, we will omit this most of the time. The elements  $(u, v)$  of  $E$  are called the *arcs* (or *directed edges*) of  $G$ . Note that we allow *loops*, i.e., arcs of the form  $(u, u)$ ; a digraph without loops is called *loopless*. If  $(u, v) \in E(G)$  is an arc, and  $w \in V(G)$  is a vertex such that  $w = u$  or  $w = v$ , then we say that  $(u, v)$  and  $w$  are *incident*.

An (*undirected*) *graph* is a pair  $(V, E)$  of a set  $V = V(G)$  of *vertices* and a set  $E = E(G)$  of *edges*, each of which is an unordered pair of (not necessarily distinct) elements of  $V$ . In other words, we explicitly allow *loops*, which are edges that link a vertex with itself. Undirected graphs can be viewed as *symmetric digraphs*: a digraph  $G = (V, E)$  is called *symmetric* if  $(u, v) \in E$  if and only if  $(v, u) \in E$ . For a digraph  $G$ , we say that  $G'$  is the *undirected graph of  $G$*  if  $G'$  is the undirected graph with  $V(G') = V(G)$  and where  $\{u, v\} \in E(G')$  if  $(u, v) \in E(G)$  or  $(v, u) \in E(G)$ . For an undirected graph  $G$ , we say that  $G'$  is an *orientation of  $G$*  if  $G'$  is a directed graph such that  $V(G') = V(G)$  and  $E(G')$  contains for each edge  $\{u, v\} \in E(G)$  either the arc  $(u, v)$  or the arc  $(v, u)$ , and no other arcs.

For some notions for digraphs  $G$  one can just use the corresponding notions for undirected graphs applied to the undirected graph of  $G$ ; conversely, most notions for directed graphs, specialised to symmetric graphs, translate to notions for the respective undirected graphs.

### 1.1.1 Examples of graphs, and corresponding notation

- The *complete graph* on  $n$  vertices  $\{1, \dots, n\}$ , denoted by  $K_n$ . This is an undirected graph on  $n$  vertices in which every vertex is joined with any other distinct vertex (so  $K_n$  contains no loops).
- The *cyclic graph* on  $n$  vertices, denoted by  $C_n$ ; this is the undirected graph with the vertex set  $\{0, \dots, n-1\}$  and edge set

$$\{\{0, 1\}, \dots, \{n-2, n-1\}, \{n-1, 0\}\} = \{\{u, v\} : |u - v| = 1 \pmod n\}.$$

- The *directed cycle* on  $n$  vertices, denoted by  $\vec{C}_n$ ; this is the digraph with the vertex set  $\{0, \dots, n-1\}$  and the arcs  $\{(0, 1), \dots, (n-2, n-1), (n-1, 0)\}$ .
- The *path* with  $n+1$  vertices and  $n$  edges, denoted by  $P_n$ ; this is an undirected graph with the vertex set  $\{0, \dots, n\}$  and edge set  $\{\{0, 1\}, \dots, \{n-1, n\}\}$ .
- The *directed path* with  $n+1$  vertices and  $n$  edges, denoted by  $\vec{P}_n$ ; this is a digraph with the vertex set  $\{0, \dots, n\}$  and edge set  $\{(0, 1), \dots, (n-1, n)\}$ .
- A *tournament* is a directed loopless graph  $G$  with the property that for all distinct vertices  $x, y$  either  $(x, y)$  or  $(y, x)$  is an edge of  $G$ , but not both.
- The *transitive tournament* on  $n \geq 2$  vertices, denoted by  $T_n$ ; this is a directed graph with the vertex set  $\{1, \dots, n\}$  where  $(i, j)$  is an arc if and only if  $i < j$ .

Let  $G$  and  $H$  be graphs (we define the following notions both for directed and for undirected graphs). Then  $G \uplus H$  denotes the *disjoint union of  $G$  and  $H$* , which is the graph with vertex set  $V(G) \cup V(H)$  (we assume that the two vertex sets are disjoint; if they are not, we take a copy of  $H$  on a disjoint set of vertices and form the disjoint union of  $G$  with the copy of  $H$ ) and edge set  $E(G) \cup E(H)$ . A graph  $G'$  is a *subgraph* of  $G$  if  $V(G') \subseteq V(G)$  and  $E(G') \subseteq E(G)$ . A graph  $G'$  is an *induced subgraph* of  $G$  if  $V' = V(G') \subseteq V(G)$  and  $(u, v) \in E(G')$  if and only if  $(u, v) \in E(G)$  for all  $u, v \in V'$ . We also say that  $G'$  is *induced by  $V'$  in  $G$* , and write  $G[V']$  for  $G'$ . We write  $G - u$  for  $G[V(G) \setminus \{u\}]$ , i.e., for the subgraph of  $G$  where the vertex  $u$  and all incident arcs are removed.

We call  $|V(G)| + |E(G)|$  the *size* of a graph  $G$ . This quantity will be important when we analyse the efficiency of algorithms on graphs.

### 1.1.2 Paths and Cycles

We start with definitions for directed paths; the corresponding terminology is then also used for undirected graphs as explained in the beginning of this section.

A *path*  $P$  (from  $u_1$  to  $u_k$  in  $G$ ) is a sequence  $(u_1, \dots, u_k)$  of vertices of  $G$  and a sequence  $(e_1, \dots, e_{k-1})$  of edges of  $G$  such that  $e_i = (u_i, u_{i+1})$  or  $e_i = (u_{i+1}, u_i) \in E(G)$ , for every  $1 \leq i < k$ . The vertex  $u_1$  is called the *start vertex* and the vertex  $u_k$  is called the *terminal vertex* of  $P$ , and we say that  $P$  is a *path from  $u_1$  to  $u_k$* . Edges  $(u_i, u_{i+1})$  are called *forward edges* and edges  $(u_{i+1}, u_i)$  are called *backward edges*. If all edges are forward edges then the path is called *directed*. If  $u_1, \dots, u_k$  are pairwise distinct then the path is called *simple*. We write  $|P| := k - 1$  for the *length* of  $P$  (i.e., we count the number of edges of  $P$ ). The *net*

*length* of  $P$  is the difference between the number of forward and the number of backward edges. Hence, a path is directed if and only if its length equals its net length.

A sequence  $(u_0, \dots, u_{k-1})$  of vertices and a sequence of edges  $(e_0, \dots, e_{k-1})$  is called a *cycle* (of  $G$ ) if  $(u_0, \dots, u_{k-1}, u_0)$  and  $(e_0, \dots, e_{k-1})$  form a path. If all the vertices of the cycle are pairwise distinct then the cycle is called *simple*. We write  $|C| := k$  for the *length* of the cycle  $C = (u_0, \dots, u_{k-1})$ . The *net length* of  $C$  is the net length of the corresponding path  $(u_0, \dots, u_{k-1}, u_0)$ . The cycle  $C$  is called *directed* if the corresponding path is a directed path.

A digraph  $G$  is called (*weakly*) *connected* if there is a path in  $G$  from any vertex to any other vertex in  $G$ . Equivalently,  $G$  is connected if and only if it cannot be written as  $H_1 \uplus H_2$  for digraphs  $H_1, H_2$  with at least one vertex each. A *connected component* of  $G$  is a maximal (with respect to inclusion of the vertex sets) connected induced subgraph of  $G$ . A digraph  $G$  is called *strongly connected* if for all vertices  $x, y \in V(G)$  there is a directed path from  $x$  to  $y$  in  $G$ . Two vertices  $u, v \in V(G)$  are *at distance  $k$*  in  $G$  if the shortest path from  $u$  to  $v$  in  $G$  has length  $k$ .

Some particular notions for undirected graphs  $G$ . A (*simple*) *cycle* of  $G$  is a sequence  $(v_1, \dots, v_k)$  of  $k \geq 3$  pairwise distinct vertices of  $G$  such that  $\{v_1, v_k\} \in E(G)$  and  $\{v_i, v_{i+1}\} \in E(G)$  for all  $1 \leq i \leq k-1$ . An undirected graph is called *acyclic* if it does not contain a cycle. A sequence  $u_1, \dots, u_k \in V(G)$  is called a (*simple*) *path* from  $u_1$  to  $u_k$  in  $G$  if  $\{u_i, u_{i+1}\} \in E(G)$  for all  $1 \leq i < k$  and if all vertices  $u_1, \dots, u_k$  are pairwise distinct. We allow the case that  $k = 1$ , in which case the path consists of a single vertex and no edges. Two vertices  $u, v \in G$  are *at distance  $k$*  in  $G$  if the shortest path in  $G$  from  $u$  to  $v$  has length  $k$ . We say that an undirected graph  $G$  is *connected* if for all vertices  $u, v \in V(G)$  there is a path from  $u$  to  $v$ . The *connected components* of  $G$  are the maximal connected induced subgraphs of  $G$ . A *forest* is an undirected acyclic graph, a *tree* is a connected forest.

## 1.2 Graph Homomorphisms

Let  $G$  and  $H$  be directed graphs. A *homomorphism* from  $G$  to  $H$  is a mapping  $h: V(G) \rightarrow V(H)$  such that  $(h(u), h(v)) \in E(H)$  whenever  $(u, v) \in E(G)$ . If such a homomorphism exists between  $G$  and  $H$  we say that  $G$  *homomorphically maps* to  $H$ , and write  $G \rightarrow H$ . Two directed graphs  $G$  and  $H$  are *homomorphically equivalent* if  $G \rightarrow H$  and  $H \rightarrow G$ .

A homomorphism from  $G$  to  $H$  is sometimes also called an  $H$ -*colouring* of  $G$ . This terminology originates from the observation that  $H$ -colourings generalise classical colourings in the sense that a graph is  $n$ -colourable if and only if it has a  $K_n$ -colouring. Graph  $n$ -colorability is not the only natural graph property that can be described in terms of homomorphisms:

- a digraph is called *balanced* (in some articles: *layered*) if it homomorphically maps to a directed path  $\vec{P}_n$ ;
- a digraph is called *acyclic* if it homomorphically maps to a transitive tournament  $T_n$ .

The equivalence classes of finite digraphs with respect to homomorphic equivalence will be denoted by  $\mathcal{D}$ . Let  $\leq$  be a binary relation defined on  $\mathcal{D}$  as follows: we set  $C_1 \leq C_2$  if there exists a digraph  $H_1 \in C_1$  and a digraph  $H_2 \in C_2$  such that  $H_1 \rightarrow H_2$  (note that this definition does not depend on the choice of the representatives  $H_1$  of  $C_1$  and  $H_2$  of  $C_2$ ). If  $f$  is a homomorphism from  $H_1$  to  $H_2$ , and  $g$  is a homomorphism from  $H_2$  to  $H_3$ , then the composition  $f \circ g$  of these functions is a homomorphism from  $H_1$  to  $H_3$ , and therefore the relation  $\leq$  is transitive. Since every graph  $H$  homomorphically maps to  $H$ , the order  $\leq$  is also

reflexive. Finally,  $\leq$  is antisymmetric since its elements are equivalence classes of directed graphs with respect to homomorphic equivalence. Define  $C_1 < C_2$  if  $C_1 \leq C_2$  and  $C_1 \neq C_2$ . We call  $(\mathcal{D}, \leq)$  the *homomorphism order* of finite digraphs.

The homomorphism order on digraphs turns out to be a *lattice* where every two elements have a supremum (also called *join*) and an infimum (also called *meet*; see Example 7.5). In the proof of this result, we need the notion of *direct products* of graphs. This notion of graph product<sup>1</sup> can be seen as a special case of the notion of direct product as it is used in model theory [51]. The class of all graphs with respect to homomorphisms forms an interesting category in the sense of category theory [47] where the product introduced above is the product in the sense of category theory, which is why this product is sometimes also called the *categorical graph product*.

**Definition 1.1** (direct product). Let  $H_1$  and  $H_2$  be two graphs. Then the (*direct-, cross-, categorical-*) product  $H_1 \times H_2$  of  $H_1$  and  $H_2$  is the graph with vertex set  $V(H_1) \times V(H_2)$ ; the pair  $((u_1, u_2), (v_1, v_2))$  is in  $E(H_1 \times H_2)$  if  $(u_1, v_1) \in E(H_1)$  and  $(u_2, v_2) \in E(H_2)$ .

Note that the product is symmetric and associative in the sense that  $H_1 \times H_2$  is isomorphic to  $H_2 \times H_1$  and  $H_1 \times (H_2 \times H_3)$  is isomorphic to  $(H_1 \times H_2) \times H_3$ , and we therefore do not specify the order of multiplication when multiplying more than two graphs. The *n-th power*  $H^n$  of a graph  $H$  is inductively defined as follows.  $H^1$  is by definition  $H$ . If  $H^i$  is already defined, then  $H^{i+1}$  is  $H^i \times H$ .

**Proposition 1.2.** *The homomorphism order  $(\mathcal{D}, \leq)$  is a lattice; i.e., for all  $A_1, A_2 \in \mathcal{D}$*

- *there exists an element  $A_1 \vee A_2 \in \mathcal{D}$ , the join of  $A_1$  and  $A_2$ , such that  $A_1 \leq (A_1 \vee A_2)$  and  $A_2 \leq (A_1 \vee A_2)$ , and such that for every  $U \in \mathcal{D}$  with  $A_1 \leq U$  and  $A_2 \leq U$  we have  $A_1 \vee A_2 \leq U$ .*
- *there exists an element  $A_1 \wedge A_2 \in \mathcal{D}$ , the meet of  $A_1$  and  $A_2$ , such that  $(A_1 \wedge A_2) \leq A_1$  and  $(A_1 \wedge A_2) \leq A_2$ , and such that for every  $U \in \mathcal{D}$  with  $U \leq A_1$  and  $U \leq A_2$  we have  $U \leq A_1 \wedge A_2$ .*

*Proof.* Let  $H_1 \in A_1$  and  $H_2 \in A_2$ . For the join, the equivalence class of the disjoint union  $H_1 \uplus H_2$  has the desired properties. For the meet, the equivalence class of  $H_1 \times H_2$  has the desired properties.  $\square$

With the seemingly simple definitions of graph homomorphisms and direct products we can already formulate very difficult combinatorial questions.

**Conjecture 1** (Hedetniemi). *Let  $G$  and  $H$  be finite graphs, and suppose that  $G \times H \rightarrow K_n$ . Then  $G \rightarrow K_n$  or  $H \rightarrow K_n$ .*

The smallest  $n \in \mathbb{N}$  such that  $G \rightarrow K_n$  is also called the *chromatic number* of  $G$ , and denoted by  $\chi(G)$ . Clearly,  $\chi(G \times H) \leq \min(\chi(G), \chi(H))$ . Hedetniemi's conjecture can be rephrased as

$$\chi(G \times H) = \min(\chi(G), \chi(H)).$$

This conjecture is easy for  $n = 1$  and  $n = 2$  (Exercise 4), and has been solved for  $n = 3$  by El Zahar and Sauer [37]. The conjecture has been refuted in 2019 by Yaroslav Shitov [68].

---

<sup>1</sup>Warning: there are several other notions of graph products that have been studied; see e.g. [47].

Clearly,  $(\mathcal{D}, \leq)$  has infinite *ascending chains*, that is, sequences  $E_1, E_2, \dots$  such that  $E_i < E_{i+1}$  for all  $i \in \mathbb{N}$ . Take for instance the equivalence class of  $\vec{P}_i$  for  $E_i$ . More interestingly,  $(\mathcal{D}, \leq)$  also has infinite descending chains.

**Proposition 1.3.** *The lattice  $(\mathcal{D}, \leq)$  contains infinite descending chains  $E_1 > E_2 > \dots$ .*

*Proof.* For this we use the following directed graphs, called *zig-zags*, which are frequently used in the theory of graph homomorphisms. We may write an orientation of a path  $P$  as a sequence of 0's and 1's, where 0 represents a forward arc and 1 represents a backward arc. For two orientations of paths  $P$  and  $Q$  with the representation  $P = p_0, \dots, p_n \in \{0, 1\}^*$  and  $Q = q_0, \dots, q_m \in \{0, 1\}^*$ , respectively, the *concatenation*  $P \circ Q$  of  $P$  and  $Q$  is the oriented path represented by  $p_0, \dots, p_n, q_0, \dots, q_m$ . For  $k \geq 1$ , the *zig-zag of order  $k$* , denoted by  $Z_k$ , is the orientation of a path represented by  $11(01)^{k-1}1$ . We recommend the reader to draw pictures of  $Z_k$  where forward arcs point up and backward arcs point down. Now, the equivalence classes of the graphs  $Z_1, Z_2, \dots$  form an infinite descending chain.  $\square$

**Proposition 1.4.** *The lattice  $(\mathcal{D}, \leq)$  contains infinite antichains, that is, sets of pairwise incomparable elements of  $\mathcal{D}$  with respect to  $\leq$ .*

*Proof.* Again, it suffices to work with orientations of paths. For  $k, l \geq 1$ , the  *$k, l$  multi zig-zag*, denoted by  $Z_{k,l}$ , is the orientation of a path represented by  $1(1(01)^k)^l 1$ . Our infinite antichain now consists of the equivalence classes containing the graphs  $Z_{k,k}$  for  $k \geq 1$ .  $\square$

A *strong homomorphism* from a digraph  $G$  to a digraph  $H$  is a function from  $V(G)$  to  $V(H)$  such that  $(f(u), f(v)) \in E(H)$  if and only if  $(u, v) \in E(G)$  for all  $u, v \in V(G)$ . An *isomorphism* between two directed graphs  $G$  and  $H$  is an bijective strong homomorphism from  $G$  to  $H$ . Note that a homomorphism  $h: G \rightarrow H$  is an isomorphism if and only if it is bijective, and  $h^{-1}$  is a homomorphism from  $H$  to  $G$ .

### Exercises.

1. How many connected components do we have in  $(P_3)^3$ ?
2. How many weakly and strongly connected components do we have in  $(\vec{C}_3)^3$ ?
3. Let  $G$  and  $H$  be digraphs. Prove that  $G \times H$  has a directed cycle if and only if both  $G$  and  $H$  have a directed cycle.
4. Prove the Hedetniemi conjecture for  $n = 1$  and  $n = 2$ .
5. Show that the Hedetniemi conjecture is equivalent to each of the following two statements.
  - Let  $n$  be a positive integer. If for two graphs  $G$  and  $H$  we have  $G \not\rightarrow K_n$  and  $H \not\rightarrow K_n$ , then  $G \times H \not\rightarrow K_n$ .
  - Let  $G$  and  $H$  be graphs with  $\chi(G) = \chi(H) = m$ . Then there exists a graph  $K$  with  $\chi(K) = m$  such that  $K \rightarrow G$  and  $K \rightarrow H$ .
6. Show that Hedetniemi's conjecture is false for directed graphs.

**Hint:** there are counterexamples  $G, H$  with four vertices each.

7. Show that for every  $k \in \mathbb{N}$ , every pair of adjacent vertices of  $(K_3)^k$  has exactly one common neighbour (that is, every edge lies in a unique subgraph of  $(K_3)^k$  isomorphic to  $K_3$ ).
8. Show that for every  $k \in \mathbb{N}$ , every pair of non-adjacent vertices in  $(K_3)^k$  has at least two common neighbours.
9. Show that a digraph  $G$  homomorphically maps to  $\vec{P}_1 = T_2$  if and only if  $\vec{P}_2$  does not homomorphically map to  $G$ .
10. Construct an orientation of a tree that is not homomorphically equivalent to an orientation of a path.
11. Construct a balanced orientation of a cycle that is not homomorphically equivalent to an orientation of a path.
12. Show that for all digraphs  $G$  we have  $G \rightarrow T_3$  if and only if  $\vec{P}_3 \not\rightarrow G$ .
13. Show that  $G \rightarrow \vec{P}_n$ , for some  $n \geq 1$ , if and only if any two paths in  $G$  that start and end in the same vertex have the same net length.
14. Show that  $G \rightarrow \vec{C}_n$ , for some  $n \geq 1$ , if and only if any two paths in  $G$  that start and end in the same vertex have the same net length modulo  $n$ .
15. Let  $a$  be an automorphism of  $K_n^k$ . Show that there are permutations  $p_1, \dots, p_k$  of  $\{1, \dots, n\}$  and a permutation  $q$  of  $\{1, \dots, k\}$  such that

$$a(x_1, \dots, x_k) = (p_1(x_{q(1)}), \dots, p_k(x_{q(k)})).$$

### 1.3 The $H$ -colouring Problem and Variants

When does a given digraph  $G$  homomorphically map to a digraph  $H$ ? For every digraph  $H$ , this question defines a computational problem, called the  *$H$ -colouring problem*. The input of this problem consists of a finite digraph  $G$ , and the question is whether there exists a homomorphism from  $G$  to  $H$ .

There are many variants of this problem. In the *precoloured  $H$ -colouring problem*, the input consists of a finite digraph  $G$ , together with a mapping  $f$  from a subset of  $V(G)$  to  $V(H)$ . The question is whether there exists an extension of  $f$  to all of  $V(G)$  which is a homomorphism from  $G$  to  $H$ . In the *list  $H$ -colouring problem*, the input consists of a finite digraph  $G$ , together with a set  $S_x \subseteq V(H)$  for every vertex  $x \in V(G)$ . The question is whether there exists a homomorphism  $h$  from  $G$  to  $H$  such that  $h(x) \in S_x$  for all  $x \in V(G)$ . It is clear that the  $H$ -colouring problem reduces to the precoloured  $H$ -colouring problem (it is a special case: the partial map might have an empty domain), and that the precoloured  $H$ -colouring problem reduces to the list  $H$ -colouring problem (for vertices  $x$  in the domain of  $f$ , we set  $S_x := \{f(x)\}$ , and for vertices  $x$  outside the domain of  $f$ , we set  $S_x := V(H)$ ).

The *constraint satisfaction problem* is a common generalisation of all these problems, and many more. It is defined not only for digraphs  $H$ , but more generally for *relational structures*  $S$ . Relational structures are the generalisation of graphs that can have many relations of arbitrary arity instead of just one binary edge relation. The constraint satisfaction problem will be introduced formally in Section 4. If  $H$  is a digraph, then the constraint satisfaction



problem for  $H$ , also denoted  $\text{CSP}(H)$ , is precisely the  $H$ -colouring problem and we use the terminology interchangeably. Note that since graphs can be seen as a special case of digraphs,  $H$ -colouring is also defined for undirected graphs  $H$ . In this case we obtain essentially the same computational problem if we only allow undirected graphs in the input; this is made precise in Exercise 18.

For every finite graph  $H$ , the  $H$ -colouring problem is obviously in NP, because for every graph  $G$  it can be verified in polynomial time whether a given mapping from  $V(G)$  to  $V(H)$  is a homomorphism from  $G$  to  $H$  or not. Clearly, the same holds for the precoloured and the list  $H$ -colouring problem. We have also seen that the  $K_n$ -colouring problem is the classical  $n$ -colouring problem, which is NP-complete [42] for  $n \geq 3$ , and therefore, no polynomial-time algorithm exists for  $K_n$ -colouring with  $n \geq 3$ , unless  $\text{P}=\text{NP}$ . However, for many graphs and digraphs  $H$  (see Exercise 19 and 9) the  $H$ -colouring problem can be solved in polynomial time. Since the 1990s, researchers have studied the question: for which digraphs  $H$  can the  $H$ -colouring problem be solved in polynomial time? It has been conjectured by Feder and Vardi [40] that  $H$ -colouring is for any finite digraph  $H$  either NP-complete or can be solved in polynomial time. This is the so-called *dichotomy conjecture*. It was shown by Ladner that unless  $\text{P}=\text{NP}$  there are infinitely many complexity classes between P and NP; so the conjecture states that for  $H$ -colouring these intermediate complexities do not appear. Feder and Vardi also showed that if the dichotomy conjecture holds for  $H$ -colouring problems, then also the more general class of CSPs for finite relational structures exhibits a complexity dichotomy.

The list  $H$ -colouring problem, on the other hand, is quickly NP-hard, and therefore less difficult to classify. And indeed, a complete classification has been obtained by Bulatov [21] already in 2003. Alternative proofs can be found in [5, 23]. Also for finite *undirected* graphs, it is known since 1990 that the dichotomy conjecture holds; this course fully covers the proof of the following.

**Theorem 1.5** (of [45]). *Let  $H$  be a finite undirected graph. If  $H$  homomorphically maps to  $K_2$ , or contains a loop, then  $H$ -colouring can be solved in polynomial time. Otherwise,  $H$ -colouring is NP-complete.*

The case that  $H$  homomorphically maps to  $K_2$  will be the topic of Exercise 19. The entire proof of Theorem 1.5 can be found in Section 8.

The dichotomy conjecture has been confirmed in 2017, independently by Bulatov [24] and by Zhuk [71].

### Exercises.

16. Let  $H$  be a finite directed graph. Find an algorithm that decides whether there is a strong homomorphism from a given graph  $G$  to the fixed graph  $H$ . The running time of the algorithm should be polynomial in the size of  $G$  (note that we consider  $|V(H)|$  to be constant).
17. Let  $H$  be a finite digraph such that  $\text{CSP}(H)$  can be solved in polynomial time. Find a polynomial-time algorithm that constructs for a given finite digraph  $G$  a homomorphism to  $H$ , if such a homomorphism exists.
18. Let  $G$  and  $H$  be directed graphs, and suppose that  $H$  is symmetric. Show that  $f: V(G) \rightarrow V(H)$  is a homomorphism from  $G$  to  $H$  if and only if  $f$  is a homomorphism from the undirected graph of  $G$  to the undirected graph of  $H$ .

19. Show that for any graph  $H$  that homomorphically maps to  $K_2$  the constraint satisfaction problem for  $H$  can be solved in polynomial time.
20. Prove that  $\text{CSP}(T_3)$  can be solved in polynomial time.
21. Prove that  $\text{CSP}(\vec{C}_3)$  can be solved in polynomial time.
22. Let  $\mathcal{N}$  be the set  $\{Z_1, Z_2, Z_3, \dots\}$ . Show that a digraph  $G$  homomorphically maps to  $\vec{P}_2$  if and only if no digraph in  $\mathcal{N}$  homomorphically maps to  $G$ .
23. Suppose that  $\text{CSP}(G)$  and  $\text{CSP}(H)$ , for two digraphs  $G$  and  $H$ , can be solved in polynomial time. Show that  $\text{CSP}(G \times H)$  and  $\text{CSP}(G \uplus H)$  can be solved in polynomial time as well.
24. Suppose that  $G$  and  $H$  are homomorphically incomparable, and suppose that  $\text{CSP}(G \uplus H)$  can be solved in polynomial time. Show that  $\text{CSP}(G)$  and  $\text{CSP}(H)$  can be solved in polynomial time as well.
25. (\*) Find digraphs  $G$  and  $H$  such that  $\text{CSP}(G \times H)$  can be solved in polynomial time, but  $\text{CSP}(G)$  and  $\text{CSP}(H)$  are NP-hard.
26. Suppose that  $G$  and  $H$  are homomorphically incomparable. Give a polynomial-time reduction from  $\text{CSP}(G)$  to  $\text{CSP}(G) \cup \text{CSP}(H)$ .

## 1.4 Cores

An *endomorphism* of a digraph  $H$  is a homomorphism from  $H$  to  $H$ . An *automorphism* of a digraph  $H$  is an isomorphism from  $H$  to  $H$ . A finite digraph  $H$  is called a *core* if every endomorphism of  $H$  is an automorphism. A subgraph  $G$  of  $H$  is called a *core of  $H$*  if  $H$  is homomorphically equivalent to  $G$  and  $G$  is a core.

**Proposition 1.6.** *Every finite digraph  $H$  has a core, which is unique up to isomorphism.*

*Proof.* Any finite digraph  $H$  has a core, since we can select an endomorphism  $e$  of  $H$  such that the image of  $e$  has smallest cardinality; the subgraph of  $H$  induced by  $e(V(H))$  is a core of  $H$ . Let  $G_1$  and  $G_2$  be cores of  $H$ , and  $f_1: H \rightarrow G_1$  and  $f_2: H \rightarrow G_2$  be homomorphisms. Let  $e_1$  be the restriction of  $f_1$  to  $V(G_2)$ . We claim that  $e_1$  is the desired isomorphism. Suppose for contradiction that  $e_1$  is not injective, i.e., there are distinct  $x, y$  in  $V(G_2)$  such that  $e_1(x) = e_1(y)$ . It follows that  $f_2 \circ e_1$  cannot be injective, too. But  $f_2 \circ e_1$  is an endomorphism of  $G_2$ , contradicting the assumption that  $G_2$  is a core. Similarly, the restriction  $e_2$  of  $f_2$  to  $V(G_1)$  is an injective homomorphism from  $G_1$  to  $G_2$ , and it follows that  $|V(G_1)| = |V(G_2)|$  and both  $e_1$  and  $e_2$  are bijective.

Now, since  $|V(G_2)|$  is finite,  $e_2 \circ e_1 \circ \dots \circ e_2 \circ e_1 = (e_2 \circ e_1)^n = \text{id}$  for large enough  $n$ . Hence,  $e_2 \circ e_1 \circ \dots \circ e_2 = (e_1)^{-1}$ , so the inverse of  $e_1$  is a homomorphism, and hence an isomorphism between  $G_1$  and  $G_2$ .  $\square$

Since a core  $G$  of a finite digraph  $H$  is unique up to isomorphism, we call  $G$  *the core of  $H$* . We want to mention without proof that it is NP-complete to decide whether a given digraph  $H$  is not a core [46].

Cores can be characterised in many different ways; for some of them, see Exercise 28. There are examples of infinite digraphs that do not have a core in the sense defined above;

see Exercise 31. Since a digraph  $H$  and its core have the same CSP, it suffices to study  $\text{CSP}(H)$  for core digraphs  $H$  only. Working with cores has advantages; one of them is shown in Proposition 1.8 below. In the proof of this proposition, we need a concept that we will use again in later sections.

**Definition 1.7.** Let  $H$  be a digraph and let  $u, v \in V(H)$  be vertices of  $H$ . Then the digraph  $H/\{u, v\}$  obtained from  $H$  by *contracting*  $u, v$  is defined to be the digraph with vertex set  $V(H) \setminus \{u, v\} \cup \{\{u, v\}\}$  and the edge set obtained from  $E(H)$  by replacing each edge in  $E(H)$  of the form  $(x, u)$  or  $(x, v)$ , for  $x \in V(H)$ , by the edge  $(x, \{u, v\})$ , and each edge in  $E(H)$  of the form  $(u, x)$  or  $(v, x)$ , for  $x \in V(H)$ , by the edge  $(\{u, v\}, x)$ .

**Proposition 1.8.** *Let  $H$  be a core. Then  $\text{CSP}(H)$  and precoloured  $\text{CSP}(H)$  are linear-time equivalent.*

*Proof.* The reduction from  $\text{CSP}(H)$  to precoloured  $\text{CSP}(H)$  is trivial, because an instance  $G$  of  $\text{CSP}(H)$  is equivalent to the instance  $(G, c)$  of precoloured  $\text{CSP}(H)$  where  $c$  is everywhere undefined.

We show the converse reduction by induction on the size of the image of the partial mapping  $c$  in instances of precoloured  $\text{CSP}(H)$ . Let  $(G, c)$  be an instance of precoloured  $\text{CSP}(H)$  where  $c$  has an image of size  $k \geq 1$ . We show how to reduce the problem to one where the partial mapping has an image of size  $k - 1$ . If we compose all these reductions (note that the size of the image is bounded by  $|V(H)|$ ), we finally obtain a reduction to  $\text{CSP}(H)$ .

Let  $x \in V(G)$  and  $u \in V(H)$  be such that  $c(x) = u$ . We first contract all vertices  $y$  of  $G$  such that  $c(y) = u$  with  $x$ . Then we create a copy of  $H$ , and attach the copy to  $G$  by contracting  $x \in V(G)$  with  $u \in V(H)$ . Let  $G'$  be the resulting graph, and let  $c'$  be the partial map obtained from  $c$  by restricting it such that it is undefined on  $x$ , and then extending it so that  $c'(v) = v$  for all  $v \in V(H)$ ,  $v \neq u$ , that appear in the image of  $c$ . Note that the size of  $G'$  and the size of  $G$  only differ by a constant.

We claim that  $(G', c')$  has a solution if and only if  $(G, c)$  has a solution. If  $f$  is a homomorphism from  $G$  to  $H$  that extends  $c$ , we further extend  $f$  to the copy of  $H$  that is attached in  $G'$  by setting  $f(v')$  to  $v$  if vertex  $v'$  is a copy of a vertex  $v \in V(H)$ . This extension of  $f$  clearly is a homomorphism from  $G'$  to  $H$  and extends  $c'$ .

Now, suppose that  $f'$  is a homomorphism from  $G'$  to  $H$  that extends  $c'$ . The restriction of  $f'$  to the vertices from the copy of  $H$  that is attached to  $x$  in  $G'$  is an endomorphism of  $H$ , and because  $H$  is a core, it is an automorphism  $\alpha$  of  $H$ . Moreover,  $\alpha$  fixes  $v$  for all  $v \in V(H)$  in the image of  $c'$ . Let  $\beta$  be the inverse of  $\alpha$ , i.e., let  $\beta$  be the automorphism of  $H$  such that  $\beta(\alpha(v)) = v$  for all  $v \in V(H)$ . Let  $f$  be the mapping from  $V(G)$  to  $V(H)$  that maps vertices that were identified with  $x$  to  $\beta(f'(x))$ , and all other vertices  $y \in V(G)$  to  $\beta(f'(y))$ . Clearly,  $f$  is a homomorphism from  $G$  to  $H$ . Moreover,  $f$  maps vertices  $y \in V(G)$ ,  $y \neq x$ , where  $c$  is defined to  $c(y)$ , since the same is true for  $f'$  and for  $\alpha$ . Moreover, because  $x$  in  $G'$  is identified to  $u$  in the copy of  $H$ , we have that  $f(x) = \beta(f'(x)) = \beta(f'(u)) = u$ , and therefore  $f$  is an extension of  $c$ .  $\square$

**Corollary 1.9.** *If for every finite digraph  $H$ , the precoloured  $H$ -colouring problem is in  $P$  or  $NP$ -complete, then  $\text{CSP}(H)$  is in  $P$  or  $NP$ -complete for every finite digraph  $H$  as well.*

We have already seen in Exercise 17 that the computational problem to construct a homomorphism from  $G$  to  $H$ , for fixed  $H$  and given  $G$ , can be reduced in polynomial-time to the problem of deciding whether there exists a homomorphism from  $G$  to  $H$ . The intended

solution of Exercise 17 requires in the worst-case  $|V(G)|^2$  many executions of the decision procedure for  $\text{CSP}(H)$ . Using the concept of cores and the precoloured CSP (and its equivalence to the CSP) we can give a faster method to construct homomorphisms.

**Proposition 1.10.** *If there is an algorithm that decides  $\text{CSP}(H)$  in time  $T$ , then there is an algorithm that constructs a homomorphism from a given digraph  $G$  to  $H$  (if such a homomorphism exists) which runs in time  $O(|V(G)|T)$ .*

*Proof.* We may assume without loss of generality that  $H$  is a core (since  $H$  and its core have the same CSP). By Proposition 1.8, there is an algorithm  $B$  for precoloured  $\text{CSP}(H)$  with a running time in  $O(T)$ . For given  $G$ , we first apply  $B$  to  $(G, c)$  for the everywhere undefined function  $c$  to decide whether there exists a homomorphism from  $G$  to  $H$ . If no, there is nothing to show. If yes, we select some  $x \in V(G)$ , and extend  $c$  by defining  $c(x) = u$  for some  $u \in V(H)$ . Then we use algorithm  $B$  to decide whether there is a homomorphism from  $G$  to  $H$  that extends  $c$ . If no, we try another vertex  $u \in V(H)$ . Clearly, for some  $u$  the algorithm must give the answer “yes”. We proceed with the extension  $c$  where  $c(x) = u$ , and repeat the procedure with another vertex  $x$  from  $V(G)$ . At the end,  $c$  is defined for all vertices  $x$  of  $G$ , and  $c$  is a homomorphism from  $G$  to  $H$ . Clearly, since  $H$  is fixed, algorithm  $B$  is executed at most  $O(|V(G)|)$  many times.  $\square$

### Exercises.

27. Show that  $Z_{k,l}$  is a core for all  $k, l \geq 2$ .
28. Prove that for every finite digraph  $G$  the following is equivalent:
  - $G$  is a core.
  - Every endomorphism of  $G$  is injective.
  - Every endomorphism of  $G$  is surjective.
29. Show that the three properties in the previous exercise are no longer equivalent if  $G$  is infinite.
30. Prove that the core of a strongly connected digraph is strongly connected.
31. Show that the infinite tournament  $(\mathbb{Q}; <)$  has endomorphisms that are not automorphisms. Show that every digraph that is homomorphically equivalent to  $(\mathbb{Q}; <)$  also has endomorphisms that are not automorphisms.
32. Let  $H$  be a core of  $G$ . Show that there exists a *retraction* from  $G$  to  $H$ , i.e., a homomorphism  $e$  from  $G$  to  $H$  such that  $e(x) = x$  for all  $x \in V(H)$ .
33. The set of automorphisms of a digraph  $G$  forms a group; this group is called *transitive* if for all  $a, b \in V(G)$  there is an automorphism  $f$  of  $G$  such that  $f(a) = b$ . Show that if  $G$  has a transitive automorphism group, then the core of  $G$  also has a transitive automorphism group.
34. Show that the connected components of a core are cores that form an antichain in  $(\mathcal{D}, \leq)$ ; conversely, the disjoint union of an antichain of cores is a core.
35. Prove that the core of a digraph with a transitive automorphism group is connected.

36. Determine the computational complexity of  $\text{CSP}(H)$  for

$$H := (\mathbb{Z}; \{(x, y) : |x - y| \in \{1, 2\}\}) .$$

## 1.5 Polymorphisms

Polymorphisms are a powerful tool for analysing the computational complexity of constraint satisfaction problems; as we will see, they are useful both for NP-hardness proofs and for proving the correctness of polynomial-time algorithms for CSPs. Polymorphisms can be seen as multi-dimensional variants of endomorphisms.

**Definition 1.11.** Let  $H$  be a digraph and  $k \geq 1$ . Then a *polymorphism of  $H$  of arity  $k$*  is a homomorphism from  $H^k$  to  $H$ .

In other words, a mapping  $f: V(H)^k \rightarrow V(H)$  is a polymorphism of  $H$  if and only if  $(f(u_1, \dots, u_k), f(v_1, \dots, v_k)) \in E(H)$  whenever  $(u_1, v_1), \dots, (u_k, v_k)$  are arcs in  $E(H)$ . Note that any digraph  $H$  has all *projections* as polymorphisms, i.e., all mappings  $p: V(H)^k \rightarrow V(H)$  that satisfy for some  $i$  the equation  $p(x_1, \dots, x_k) = x_i$  for all  $x_1, \dots, x_k \in V(H)$ .

**Example 1.12.** The operation  $(x, y) \mapsto \min(x, y)$  is a polymorphism of the digraph  $\vec{T}_n = (\{1, \dots, n\}; <)$ . △

An operation  $f: V(H)^k \rightarrow V(H)$  is called

- *idempotent* if  $f(x, \dots, x) = x$  for all  $x \in V(H)$ .
- *conservative* if  $f(x_1, \dots, x_k) \in \{x_1, \dots, x_k\}$  for all  $x_1, \dots, x_k \in V(H)$ .

**Definition 1.13.** A digraph  $H$  is called *projective* if every idempotent polymorphism is a projection.

The following will be shown in Section 7.

**Proposition 1.14.** For all  $n \geq 3$ , the graph  $K_n$  is projective.

### Exercises.

37. Show that if  $f: H^k \rightarrow H$  is a polymorphism of a digraph  $H$ , then  $\hat{f}(x) := f(x, \dots, x)$  is an endomorphism of  $H$ .
38. Show that if  $H$  is a finite core digraph with a *symmetric* binary polymorphism  $f$ , that is,  $f(x, y) = f(y, x)$  for all  $x, y \in V(H)$ , then  $H$  also has an *idempotent* symmetric polymorphism.

## 2 The Arc-consistency Procedure

The *arc-consistency procedure* is one of the most fundamental and well-studied algorithms that are applied for CSPs. This procedure was first discovered for constraint satisfaction problems in artificial intelligence [60,62]; in the graph homomorphism literature, the algorithm is sometimes called the *consistency check algorithm*.

$AC_H(G)$ Input: a finite digraph $G$ . Data structure: a list $L(x) \subseteq V(H)$ for each vertex $x \in V(G)$ .  Set $L(x) := V(H)$ for all $x \in V(G)$ . Do For each $(x, y) \in E(G)$ : Remove $u$ from $L(x)$ if there is no $v \in L(y)$ with $(u, v) \in E(H)$ . Remove $v$ from $L(y)$ if there is no $u \in L(x)$ with $(u, v) \in E(H)$ . If $L(x)$ is empty for some vertex $x \in V(G)$ then <b>reject</b> Loop until no list changes
--

Figure 1: The arc-consistency procedure for  $CSP(H)$ .

Let  $H$  be a finite digraph, and let  $G$  be an instance of  $CSP(H)$ . The idea of the procedure is to maintain for each vertex in  $G$  a list of vertices of  $H$ , and each element in the list of  $x$  represents a candidate for an image of  $x$  under a homomorphism from  $G$  to  $H$ . The algorithm successively removes vertices from these lists; it only removes a vertex  $u \in V(H)$  from the list for  $x \in V(G)$ , if there is no homomorphism from  $G$  to  $H$  that maps  $x$  to  $u$ . To detect vertices  $x, u$  such that  $u$  can be removed from the list for  $x$ , the algorithm uses two rules (in fact, one rule and a symmetric version of the same rule): if  $(x, y)$  is an edge in  $G$ , then

- remove  $u$  from  $L(x)$  if there is no  $v \in L(y)$  with  $(u, v) \in E(H)$ ;
- remove  $v$  from  $L(y)$  if there is no  $u \in L(x)$  with  $(u, v) \in E(H)$ .

If eventually we cannot remove any vertex from any list with these rules any more, the digraph  $G$  together with the lists for each vertex is called *arc-consistent*. The pseudo-code of the entire arc-consistency procedure is displayed in Figure 1.

Clearly, if the algorithm removes all vertices from one of the lists, then there is no homomorphism from  $G$  to  $H$ . It follows that if  $AC_H$  rejects an instance of  $CSP(H)$ , it has no solution. The converse implication does not hold in general. For instance, let  $H$  be  $K_2$ , and let  $G$  be  $K_3$ . In this case,  $AC_H$  does not remove any vertex from any list, but obviously there is no homomorphism from  $K_3$  to  $K_2$ .

However, there are digraphs  $H$  where the  $AC_H$  is a complete decision procedure for  $CSP(H)$  in the sense that it rejects an instance  $G$  of  $CSP(H)$  if and only if  $G$  does not homomorphically map to  $H$ . In this case we say that  $AC$  *solves*  $CSP(H)$ .

**Implementation.** The running time of  $AC_H$  is for any fixed digraph  $H$  polynomial in the size of  $G$ . In a naive implementation of the procedure, the inner loop of the algorithm would go over all edges of the digraph, in which case the running time of the algorithm is quadratic in the size of  $G$ . In the following we describe an implementation of the arc-consistency procedure, called  $AC-3$ , which is due to Mackworth [60], and has a worst-case running time that is linear in the size of  $G$ . Several other implementations of the arc-consistency procedure have been proposed in the Artificial Intelligence literature, aiming at reducing the costs of the algorithm in terms of the number of vertices of both  $G$  and  $H$ . But here we consider the

size of  $H$  to be fixed, and therefore we do not follow this line of research. With AC-3, we rather present one of the simplest implementations of the arc-consistency procedure with a linear running time.

```

AC-3H(G)
Input: a finite digraph G.
Data structure: a list L(x) of vertices of H for each x ∈ V(G).
                the worklist W: a list of arcs of G.

Subroutine Revise((x0, x1), i)
Input: an arc (x0, x1) ∈ E(G), an index i ∈ {0, 1}.
    change = false
    for each ui in L(xi)
        If there is no u1-i ∈ L(x1-i) such that (u0, u1) ∈ E(H) then
            remove ui from L(xi)
            change = true
        end if
    end for
    If change = true then
        If L(xi) = ∅ then reject
    else
        For all arcs (z0, z1) ∈ E(G) with z0 = xi or z1 = xi add (z0, z1) to W
    end if

W := E(G)
Do
    remove an arc (x0, x1) from W
    Revise((x0, x1), 0)
    Revise((x0, x1), 1)
while W ≠ ∅

```

Figure 2: The AC-3 implementation of the arc-consistency procedure for CSP( $H$ ).

The idea of AC-3 is to maintain a *worklist*, which contains a list of arcs  $(x_0, x_1)$  of  $G$  that might help to remove a value from  $L(x_0)$  or  $L(x_1)$ . Whenever we remove a value from a list  $L(x)$ , we add all arcs that are in  $G$  incident to  $x$ . Note that then any arc in  $G$  might be added at most  $2|V(H)|$  many times to the worklist, which is a constant in the size of  $G$ . Hence, the while loop of the implementation is iterated for at most a linear number of times. Altogether, the running time is linear in the size of  $G$  as well.

**Arc-consistency for pruning search.** Suppose that  $H$  is such that AC does not solve CSP( $H$ ). Even in this situation the arc-consistency procedure might be useful for *pruning the search space* in exhaustive approaches to solve CSP( $H$ ). In such an approach we might use the arc-consistency procedure as a subroutine as follows. Initially, we run AC<sub>H</sub> on the input instance  $G$ . If it computes an empty list, we reject. Otherwise, we select some vertex  $x \in V(G)$ , and set  $L(x)$  to  $\{u\}$  for some  $u \in L(x)$ . Then we proceed recursively with the resulting lists. If AC<sub>H</sub> now detects an empty list, we backtrack, but remove  $u$  from  $L(x)$ . Finally, if the algorithm does not detect an empty list at the first level of the recursion, we end up with singleton lists for each vertex  $x \in V(G)$ , which gives rise to a homomorphism

from  $G$  to  $H$ .

## 2.1 The Power Graph

For which  $H$  does the Arc-Consistency procedure solve  $\text{CSP}(H)$ ? In this section we present an elegant and effective characterisation of those finite digraphs  $H$  where AC solves  $\text{CSP}(H)$ , found by Feder and Vardi [40].

**Definition 2.1.** For a digraph  $H$ , the *power graph*  $P(H)$  is the digraph whose vertices are non-empty subsets of  $V(H)$  and where two subsets  $U$  and  $V$  are joined by an arc if the following holds:

- for every vertex  $u \in U$ , there exists a vertex  $v \in V$  such that  $(u, v) \in E(H)$ , and
- for every vertex  $v \in V$ , there exists a vertex  $u \in U$  such that  $(u, v) \in E(H)$ .

The definition of the power graph resembles the arc-consistency algorithm, and indeed, we have the following lemma which describes the correspondence.

**Lemma 2.2.**  $\text{AC}_H$  rejects  $G$  if and only if  $G \not\rightarrow P(H)$ .

*Proof.* Suppose first that  $\text{AC}_H$  does *not* reject  $G$ . For  $u \in V(G)$ , let  $L(u)$  be the list derived at the final stage of the algorithm. Then by definition of  $E(P(H))$ , the map  $x \mapsto L(x)$  is a homomorphism from  $G$  to  $P(H)$ .

Conversely, suppose that  $f: G \rightarrow P(H)$  is a homomorphism. We prove by induction over the execution of  $\text{AC}_H$  that for all  $x \in V(G)$  the elements of  $f(x)$  are never removed from  $L(x)$ . To see that, let  $(a, b) \in E(G)$  be arbitrary. Then  $((f(a), f(b)) \in E(P(H))$ , and hence for every  $u \in f(a)$  there exists a  $v \in f(b)$  such that  $(u, v) \in E(H)$ . By inductive assumption,  $v \in L(b)$ , and hence  $u$  will not be removed from  $L(a)$ . This concludes the inductive step.  $\square$

**Theorem 2.3.** Let  $H$  be a finite digraph. Then AC solves  $\text{CSP}(H)$  if and only if  $P(H)$  homomorphically maps to  $H$ .

*Proof.* Suppose first that AC solves  $\text{CSP}(H)$ . Apply  $\text{AC}_H$  to  $P(H)$ . Since  $P(H) \rightarrow P(H)$ , the previous lemma shows that  $\text{AC}_H$  does not reject  $P(H)$ . Hence,  $P(H) \rightarrow H$  by assumption.

Conversely, suppose that  $P(H) \rightarrow H$ . If  $\text{AC}_H$  rejects a digraph  $G$  then  $G \not\rightarrow H$ . If  $\text{AC}_H$  does accept  $G$ , then the lemma asserts that  $G \rightarrow P(H)$ . Composing homomorphisms, we obtain that  $G \rightarrow H$ .  $\square$

**Observation 2.4.** Let  $H$  be a core digraph. Note that if  $P(H)$  homomorphically maps to  $H$ , then there also exists a homomorphism that maps  $\{x\}$  to  $x$  for all  $x \in V(H)$  (here we use the assumption that  $H$  is a core!). We claim that in this case the precoloured CSP for  $H$  can be solved by the modification of  $\text{AC}_H$  which starts with  $L(x) := \{c(x)\}$  for all  $x \in V(G)$  in the range of the precolouring function  $c$ , instead of  $L(x) := V(H)$ . This is a direct consequence of the proof of Theorem 2.3. If the modified version of  $\text{AC}_H$  solves the precoloured CSP for  $H$ , then the classical version of  $\text{AC}_H$  solves  $\text{CSP}(H)$ . Hence, it follows that the following are equivalent:

- AC solves  $\text{CSP}(H)$ ;
- the above modification of  $\text{AC}_H$  solves the precoloured CSP for  $H$ ;



- $P(H) \rightarrow H$ .

Note that the condition given in Theorem 2.3 can be used to decide algorithmically whether AC solves  $\text{CSP}(H)$ , because it suffices to test whether  $P(H)$  homomorphically maps to  $H$ . Such problems about deciding properties of  $\text{CSP}(H)$  for given  $H$  are often called algorithmic *meta-problems*. A naive algorithm for the above test would be to first construct  $P(H)$ , and then to search non-deterministically for a homomorphism from  $P(H)$  to  $H$ , which puts the meta-problem for solvability of  $\text{CSP}(H)$  by AC into the complexity class NExpTime (*Non-deterministic Exponential Time*). This can be improved.

**Proposition 2.5.** *There exists a deterministic exponential time algorithm that tests for a given finite core digraph  $H$  whether  $P(H)$  homomorphically maps to  $H$ .*

*Proof.* We first explicitly construct  $P(H)$ , and then apply  $\text{AC}_H$  to  $P(H)$ . If  $\text{AC}_H$  rejects, then there is certainly no homomorphism from  $P(H) \rightarrow H$  by the properties of  $\text{AC}_H$ , and we return ‘false’. If  $\text{AC}_H$  accepts, then we cannot argue right away that  $P(H)$  homomorphically maps to  $H$ , since we do not know yet whether  $\text{AC}_H$  is correct for  $\text{CSP}(H)$ .

But here is the trick. What we do in this case is to pick an arbitrary  $x \in V(P(H))$ , and remove all but one value  $u$  from  $L(x)$ , and continue with the execution of  $\text{AC}_H$ . If  $\text{AC}_H$  then derives the empty list, we try the same with another value  $u'$  from  $L(x)$ . If we obtain failure for all values of  $L(x)$ , then clearly there is no homomorphism from  $P(H)$  to  $H$ , and we return ‘false’. Otherwise, if  $\text{AC}_H$  does not derive the empty list after removing all values but  $u$  from  $L(x)$ , we continue with another element  $y$  of  $V(P(H))$ , setting  $L(y)$  to  $\{v\}$  for some  $v \in L(y)$ . We repeat this procedure until at the end we have constructed a homomorphism from  $P(H)$  to  $H$ . In this case we return ‘true’.

If  $\text{AC}_H$  rejects for some  $x \in V(P(H))$  when  $L(x) = \{u\}$  for all possible  $u \in V(H)$ , then the adaptation of  $\text{AC}_H$  for the precoloured CSP would have given an incorrect answer for the previously selected variable (it said yes while it should have said no). By Observation 2.4, this means that  $P(H)$  does *not* homomorphically map to  $H$ . Again, we return ‘false’.  $\square$

The precise computational complexity to decide for a given digraph  $H$  whether  $P(H) \rightarrow H$  is not known; we refer to [31] for related questions and results.

**Question 1.** *What is the computational complexity to decide for a given core digraph  $H$  whether  $P(H) \rightarrow H$ ? Is this problem in P?*

## 2.2 Tree Duality

Another mathematical notion that is closely related to the arc-consistency procedure is *tree duality*. The idea of this concept is that when a digraph  $H$  has tree duality, then we can show that there is no homomorphism from a digraph  $G$  to  $H$  by exhibiting a *tree obstruction* in  $G$ . This is formalized in the following definition.

**Definition 2.6.** A digraph  $H$  has *tree duality* if there exists a (not necessarily finite) set  $\mathcal{N}$  of orientations of finite trees such that for all digraphs  $G$  there is a homomorphism from  $G$  to  $H$  if and only if no digraph in  $\mathcal{N}$  homomorphically maps to  $G$ .

We refer to the set  $\mathcal{N}$  in Definition 2.6 as an *obstruction set* for  $\text{CSP}(H)$ . Note that no  $T \in \mathcal{N}$  homomorphically maps to  $H$ . The pair  $(\mathcal{N}, H)$  is called a *duality pair*. We have already encountered such an obstruction set in Exercise 9, where  $H = T_2$ , and  $\mathcal{N} = \{\bar{P}_2\}$ . In

other words,  $(\{\vec{P}_2\}, T_2)$  is a duality pair. Other duality pairs are  $(\{\vec{P}_3\}, T_3)$  (Exercise 12), and  $(\{Z_1, Z_2, \dots\}, \vec{P}_2)$  (Exercise 22).

Theorem 2.7 is a surprising link between the completeness of the arc-consistency procedure, tree duality, and the power graph, and was discovered by Feder and Vardi [39] in the more general context of constraint satisfaction problems.

**Theorem 2.7.** *Let  $H$  be a finite digraph. Then the following are equivalent.*

1.  $H$  has tree duality;
2.  $P(H)$  homomorphically maps to  $H$ ;
3. AC solves  $\text{CSP}(H)$ .
4. If every orientation of a tree that homomorphically maps to  $G$  also homomorphically maps to  $H$ , then  $G$  homomorphically maps to  $H$ ;

*Proof.* The equivalence  $2 \Leftrightarrow 3$  has been shown in the previous section. We show  $3 \Rightarrow 1$ ,  $1 \Rightarrow 4$ , and  $4 \Rightarrow 2$ .

$3 \Rightarrow 1$ : Suppose that AC solves  $\text{CSP}(H)$ . We have to show that  $H$  has tree duality. Let  $\mathcal{N}$  be the set of all orientations of trees that do not homomorphically map to  $H$ . We claim that if a digraph  $G$  does not homomorphically map to  $H$ , then there is  $T \in \mathcal{N}$  that homomorphically maps to  $G$ .

By assumption, the arc-consistency procedure applied to  $G$  eventually derives the empty list for some vertex of  $G$ . We use the computation of the procedure to construct an orientation  $T$  of a tree, following the exposition in [57]. When deleting a vertex  $u \in V(H)$  from the list of a vertex  $x \in V(G)$ , we define an orientation of a rooted tree  $T_{x,u}$  with root  $r_{x,u}$  such that

1. there is a homomorphism from  $T_{x,u}$  to  $G$  mapping  $r_{x,u}$  to  $x$ ;
2. there is no homomorphism from  $T_{x,u}$  to  $H$  mapping  $r_{x,u}$  to  $u$ .

Assume that the vertex  $u$  is deleted from the list of  $x$  because we found an arc  $(x, y) \in E(H)$  such that there is no arc  $(u, v) \in E(H)$  with  $v \in L(y)$ ; if it was deleted because of an arc  $(y, x) \in E(H)$  the proof follows with the obvious changes.

If  $E(H)$  does not contain any arc  $(u, v)$ , we define  $T_{x,u}$  to be the tree that just contains an arc  $(p, q)$  with root  $r_{x,u} = p$ ; clearly,  $T_{x,u}$  satisfies property (1) and (2). Otherwise, for every arc  $(u, v) \in E(H)$  the vertex  $v$  has already been removed from the list  $L(y)$ , and hence by induction  $T_{y,v}$  having properties (1) and (2) is already defined. We then add a copy of  $T_{y,v}$  to  $T_{x,u}$  and add an edge between  $r_{x,u}$  and the root  $r_{y,v}$  of  $T_{y,v}$ .

We verify that the resulting orientation of a tree  $T_{x,u}$  satisfies (1) and (2). Let  $f$  be the homomorphism from  $T_{y,v}$  mapping  $r_{y,v}$  to  $v$ , which exists due to (1). The extension of  $f$  to  $V(T_{x,u})$  that maps  $r_{x,u}$  to  $x$  is a homomorphism from  $T_{x,u}$  to  $G$ , and this shows that (1) holds for  $T_{x,u}$ . But any homomorphism from  $T_{x,u}$  to  $H$  that maps  $r_{x,u}$  to  $u$  would also map the root of  $T_{y,v}$  to  $v$ , which is impossible, and this shows that (2) holds for  $T_{x,u}$ . When the list  $L(x)$  of some vertex  $x \in V(G)$  becomes empty, we can construct an orientation of a tree  $T$  by contracting the roots of all  $T_{x,u}$  into a vertex  $r$ . We then find a homomorphism from  $T$  to  $G$  by mapping  $r$  to  $x$  and extending the homomorphism independently on each  $T_{x,u}$ . But any homomorphism from  $T$  to  $H$  must map  $r$  to some element  $u \in V(H)$ , and hence there is a homomorphism from  $T_{x,u}$  to  $H$  that maps  $x$  to  $u$ , a contradiction.

1  $\Rightarrow$  4: If  $H$  has an obstruction set  $\mathcal{N}$  consisting of orientations of trees, and if  $G$  does not homomorphically map to  $H$ , there exists an orientation of a tree  $T \in \mathcal{N}$  that maps to  $G$  but not to  $H$ .

4  $\Rightarrow$  2: To show that  $P(H)$  homomorphically maps to  $H$ , it suffices to prove that every orientation  $T$  of a tree that homomorphically maps to  $P(H)$  also homomorphically maps to  $H$ . Let  $f$  be a homomorphism from  $T$  to  $P(H)$ , and let  $x$  be any vertex of  $T$ . We construct a sequence  $f_0, \dots, f_n$ , for  $n = |V(T)|$ , where  $f_i$  is a homomorphism from the subgraph of  $T$  induced by the vertices at distance at most  $i$  to  $x$  in  $T$ , and  $f_{i+1}$  is an extension of  $f_i$  for all  $0 \leq i < n$ . The mapping  $f_0$  maps  $x$  to some vertex from  $f(x)$ . Suppose inductively that we have already defined  $f_i$ . Let  $y$  be a vertex at distance  $i+1$  from  $x$  in  $T$ . Since  $T$  is an orientation of a tree, there is a unique  $y' \in V(T)$  of distance  $i$  from  $x$  in  $T$  such that  $(y, y') \in E(T)$  or  $(y', y) \in E(T)$ . Note that  $u = f_i(y')$  is already defined. In case that  $(y', y) \in E(T)$ , there must be a vertex  $v$  in  $f(y)$  such that  $(u, v) \in E(H)$ , since  $(f(y'), f(y))$  must be an arc in  $P(H)$ , and by definition of  $P(H)$ . We then set  $f_{i+1}(y) = v$ . In case that  $(y, y') \in E(T)$  we can proceed analogously. By construction, the mapping  $f_n$  is a homomorphism from  $T$  to  $H$ .  $\square$

### 2.3 Totally Symmetric Polymorphisms

There is also a characterisation of the power of the arc-consistency procedure which is based on polymorphisms, due to [34].

**Definition 2.8.** A function  $f: D^k \rightarrow D$  is called *totally symmetric* if

$$f(x_1, \dots, x_k) = f(y_1, \dots, y_k) \text{ whenever } \{x_1, \dots, x_k\} = \{y_1, \dots, y_k\}.$$

**Theorem 2.9.** Let  $H$  be a finite digraph. Then the following are equivalent.

1.  $P(H)$  homomorphically maps to  $H$ ;
2.  $H$  has totally symmetric polymorphisms of all arities;
3.  $H$  has a totally symmetric polymorphism of arity  $2|V(H)|$ .

*Proof.* 1.  $\Rightarrow$  2.: Suppose that  $g$  is a homomorphism from  $P(H)$  to  $H$ , and let  $k \in \mathbb{N}$  be arbitrary. Let  $f$  be defined by  $f(x_1, \dots, x_k) = g(\{x_1, \dots, x_k\})$ . If  $(x_1, y_1), \dots, (x_k, y_k) \in E(H)$ , then  $\{x_1, \dots, x_k\}$  is adjacent to  $\{y_1, \dots, y_k\}$  in  $P(H)$ , and hence  $(f(x_1, \dots, x_k), f(y_1, \dots, y_k)) \in E(H)$ . Therefore,  $f$  is a polymorphism of  $H$ , and it is clearly totally symmetric.

The implication 2.  $\Rightarrow$  3. is trivial. To prove that 3.  $\Rightarrow$  1., suppose that  $f$  is a totally symmetric polymorphism of arity  $2|V(H)|$ . Let  $g: V(P(H)) \rightarrow V(H)$  be defined by

$$g(\{x_1, \dots, x_n\}) := f(x_1, \dots, x_{n-1}, x_n, x_n, \dots, x_n)$$

which is well-defined because  $f$  is totally symmetric. Let  $(U, W) \in E(P(H))$ , and let  $x_1, \dots, x_p$  be an enumeration of the elements of  $U$ , and  $y_1, \dots, y_q$  be an enumeration of the elements of  $W$ . The properties of  $P(H)$  imply that there are  $y'_1, \dots, y'_p \in W$  and  $x'_1, \dots, x'_q \in U$  such that  $(x_1, y'_1), \dots, (x_p, y'_p) \in E(H)$  and  $(x'_1, y_1), \dots, (x'_q, y_q) \in E(H)$ . Since  $f$  preserves  $E$ ,

$$g(U) = g(\{x_1, \dots, x_p\}) = f(x_1, \dots, x_p, x'_1, \dots, x'_q, x_1, \dots, x_1)$$

is adjacent to

$$g(W) = g(\{y_1, \dots, y_q\}) = f(y'_1, \dots, y'_p, y_1, \dots, y_q, y'_1, \dots, y'_1). \quad \square$$

Given Theorem 2.9, it is natural to ask whether there exists a  $k$  so that the existence of a totally symmetric polymorphism of arity  $k$  implies totally symmetric polymorphisms of all arities. The following example shows that this is not the case.

**Example 2.10.** For every prime  $p \geq 3$ , the digraph  $\vec{C}_p$  clearly does not have a totally symmetric polymorphism of arity  $p$ : if  $f: \{0, \dots, p-1\}^p \rightarrow \{0, \dots, p-1\}$  is a totally symmetric operation, then  $f(0, 1, \dots, p-1) = f(1, \dots, p-1, 0)$ , and hence  $f$  does not preserve the edge relation. On the other hand, if  $n < p$  then  $\vec{C}_p$  has the totally symmetric polymorphism

$$f(x_1, \dots, x_n) := |S|^{-1} \sum_{x \in S} x \pmod p$$

where  $S = \{x_1, \dots, x_n\}$ . (Note that  $|S| < p$  and hence has a multiplicative inverse.) The operation is clearly totally symmetric; the verification that it preserves the edge relation of  $\vec{C}_p$  is Exercise 47.  $\triangle$

## 2.4 Semilattice Polymorphisms

Some digraphs have a single binary polymorphism that generates operations satisfying the conditions in the previous theorem, as in the following statement. A binary operation  $f: D^2 \rightarrow D$  is called *commutative* if it satisfies

$$f(x, y) = f(y, x) \text{ for all } x, y \in D.$$

It is called *associative* if it satisfies

$$f(x, f(y, z)) = f(f(x, y), z) \text{ for all } x, y, z \in D.$$

**Definition 2.11.** A binary operation is called a *semilattice* operation  $f$  if it is associative, commutative, and idempotent.

Examples of semilattice operations are functions from  $D^2 \rightarrow D$  defined as  $(x, y) \mapsto \min(x, y)$ ; here the minimum is taken with respect to any fixed linear order of  $D$ .

**Theorem 2.12.** *Let  $H$  be a finite digraph. Then  $P(H) \rightarrow H$  if and only if  $H$  is homomorphically equivalent to a digraph with a semilattice polymorphism.*

*Proof.* Suppose first that  $P(H) \rightarrow H$ . Thus,  $H$  and  $P(H)$  are homomorphically equivalent, and it suffices to show that  $P(H)$  has a semilattice polymorphism. The mapping  $(X, Y) \mapsto X \cup Y$  is clearly a semilattice operation; we claim that it preserves the edges of  $P(H)$ . Let  $(U, V)$  and  $(A, B)$  be edges in  $P(H)$ . Then for every  $u \in U$  there is a  $v \in V$  such that  $(u, v) \in E(H)$ , and for every  $u \in A$  there is a  $v \in B$  such that  $(u, v) \in E(H)$ . Hence, for every  $u \in U \cup A$  there is a  $v \in V \cup B$  such that  $(u, v) \in E(H)$ . Similarly, we can verify that for every  $v \in V \cup B$  there is a  $u \in U \cup A$  such that  $(u, v) \in E(H)$ . This proves the claim.

For the converse, suppose that  $H$  is homomorphically equivalent to a digraph  $G$  with a semilattice polymorphism  $f$ . Let  $h$  be the homomorphism from  $H$  to  $G$ . The operation  $(x_1, \dots, x_n) \mapsto f(x_1, f(x_2, f(\dots, f(x_{n-1}, x_n) \dots)))$  is a totally symmetric polymorphism of  $G$ . Then Theorem 2.9 implies that  $P(G) \rightarrow G$ . The map  $S \mapsto \{h(u) \mid u \in S\}$  is a homomorphism from  $P(H)$  to  $P(G)$ . Therefore,  $P(H) \rightarrow P(G) \rightarrow G \rightarrow H$ , as desired.  $\square$

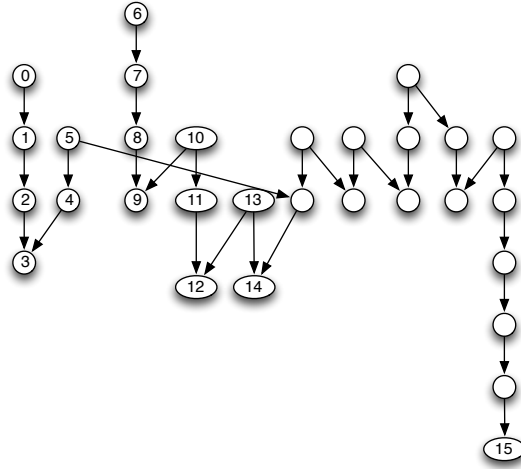


Figure 3: An orientation of a tree  $H$  with an NP-complete  $H$ -colouring problem [41].

By verifying the existence of semilattice polymorphisms for a concrete class of digraphs, we obtain the following consequence.

**Corollary 2.13.** *AC solves  $\text{CSP}(H)$  if  $H$  is an orientation of a path.*

*Proof.* Suppose that  $1, \dots, n$  are the vertices of  $H$  such that either  $(i, i + 1)$  or  $(i + 1, i)$  is an arc in  $E(H)$  for all  $i < n$ . It is straightforward to verify that the mapping  $(x, y) \mapsto \min(x, y)$  is a polymorphism of  $H$ . The statement now follows from Theorem 2.12.  $\square$

We want to remark that there are orientations of trees  $H$  with an NP-complete  $H$ -colouring problem (the smallest known example has 30 vertices, found by Jana Fischer [41]; see Figure 3). So, unless  $P=NP$ , this shows that there are orientations of trees  $H$  that do not have tree-duality.

### Exercises.

39. Recall that a digraph is called *balanced* if it homomorphically maps to a directed path. Let  $H$  be a digraph.
  - Prove: if  $H$  is balanced, then  $P(H)$  is balanced;
  - Disprove: if  $H$  is an orientation of a tree, then  $P(H)$  is an orientation of a forest.
40. Up to isomorphism, there is only one unbalanced cycle  $H$  on four vertices that is a core and not the directed cycle. Show that AC does not solve  $\text{CSP}(H)$ .
41. Does the digraph

$$(\{0, 1, 2, 3, 4, 5\}; \{(0, 1), (1, 2), (0, 2), (3, 2), (3, 4), (4, 5), (3, 5), (0, 5)\})$$

have tree duality?

42. Show that AC solves  $\text{CSP}(T_n)$ , for every  $n \geq 1$ .
43. Let  $H$  be a finite digraph. Show that  $P(H)$  contains a loop if and only if  $H$  contains a directed cycle.
44. Show that the previous statement is false for infinite digraphs  $H$ .
45. Let  $G$  and  $H$  be finite digraphs that are homomorphically incomparable and suppose that  $\text{CSP}(G)$  is NP-hard or  $\text{CSP}(H)$  is NP-hard. Show that  $\text{CSP}(G \uplus H)$  is also NP-hard.
46. Show that an orientation of a tree homomorphically maps to  $H$  if and only if it homomorphically maps to  $P(H)$ .
47. Prove the final statement in Example 2.10.
48. Let  $H$  be a finite digraph. Then  $\text{AC}_H$  rejects an orientation of a tree  $T$  if and only if there is no homomorphism from  $T$  to  $H$  (in other words, AC solves  $\text{CSP}(H)$  if the input is restricted to orientations of trees).
49. Show that there is a linear-time algorithm that tests whether a given orientation of a tree is a core (the author thanks Florian Starke and Lea Bänder for the idea for this exercise).
50. Show that the core of an orientation of a tree can be computed in polynomial time.

### 3 The Path-consistency Procedure

The path-consistency procedure is a well-studied generalization of the arc-consistency procedure from artificial intelligence. The path-consistency procedure is also known as the pair-consistency check algorithm in the graph theory literature.

Many CSPs that can not be solved by the arc-consistency procedure can still be solved in polynomial time by the path-consistency procedure. The simplest examples are  $H = K_2$  (see Exercise 19) and  $H = \tilde{C}_3$  (see Exercise 21). The idea is to maintain a list of pairs from  $V(H)^2$  for each pair of elements from  $V(G)$  (similarly to the arc-consistency procedure, where we maintained a list of vertices from  $V(H)$  for each vertex in  $V(G)$ ). We successively remove pairs from these lists when the pairs can be excluded *locally*. Some authors maintain a list only for each pair of *distinct* vertices of  $V(G)$ , and they refer to our (stronger) variant as the *strong path-consistency procedure*. Our procedure (where vertices need not be distinct) has the advantage that it is at least as strong as the arc-consistency procedure, because the lists  $L(x, x)$  and the rules of the path-consistency procedure for  $x = y$  simulate the rules of the arc-consistency procedure.

In Subsection 3.1 we will see many examples of digraphs  $H$  where the path-consistency procedure solves the  $H$ -colouring problem, but the arc-consistency procedure does not. The greater power of the path-consistency procedure comes at the price of a bigger worst-case running time: while the arc-consistency procedure has linear-time implementations, the best known implementations of the path-consistency procedure require cubic time in the size of the input (see Exercise 51).

```

PCH(G)
Input: a finite digraph G.
Data structure: for all  $x, y \in V(G)$  a list  $L(x, y)$  of elements of  $V(H)^2$ 

For each  $(x, y) \in V(G)^2$ 
  If  $(x, y) \in E(G)$  then  $L(x, y) := E(H)$ ,
  else  $L(x, y) := V(H)^2$ .
  If  $x = y$  then  $L(x, y) := L(x, y) \cap \{(u, u) \mid u \in V(H)\}$ .
Do
  For all vertices  $x, y, z \in V(G)$ :
    For each  $(u, w) \in L(x, z)$ :
      If there is no  $v \in V(H)$  such that  $(u, v) \in L(x, y)$  and  $(v, w) \in L(y, z)$  then
        Remove  $(u, w)$  from  $L(x, z)$ 
    If  $L(x, z)$  is empty then reject
Loop until no list changes

```

Figure 4: The strong path-consistency procedure for  $\text{CSP}(H)$ .

**The  $k$ -consistency procedure.** The path-consistency procedure can be generalised further to the  $k$ -consistency procedure. In fact, arc- and path-consistency procedure are just a special case of the  $k$ -consistency for  $k = 2$  and  $k = 3$ , respectively. In other words, for digraphs  $H$  the path-consistency procedure is the 3-consistency procedure and the arc-consistency procedure is the 2-consistency procedure.

The idea of  $k$ -consistency is to maintain sets of  $(k-1)$ -tuples from  $V(H)^{k-1}$  for each  $(k-1)$ -tuple from  $V(G)^{k-1}$ , and to successively remove tuples by local inference. It is straightforward to generalise also the details of the path-consistency procedure. For fixed  $H$  and fixed  $k$ , the running time of the  $k$ -consistency procedure is still polynomial in the size of  $G$ . But the dependency of the running time on  $k$  is clearly exponential.

However, we would like to point out that path consistency alias 3-consistency is of *particular* theoretical importance, due to the following recent result.

**Theorem 3.1** (Barto and Kozik [7]). *If  $\text{CSP}(H)$  can be solved by  $k$ -consistency for some  $k \geq 3$ , then  $\text{CSP}(H)$  can also be solved by 3-consistency.*

### Exercises

51. Show that the path-consistency procedure for  $\text{CSP}(H)$  can (for fixed  $H$ ) be implemented such that the worst-case running time is cubic in the size of the input digraph. (Hint: use a worklist as in AC-3.)
52. Show that if path consistency solves  $\text{CSP}(H_1)$  and path consistency solves  $\text{CSP}(H_2)$ , then path consistency solves  $\text{CSP}(H_1 \uplus H_2)$ .

### 3.1 Majority Polymorphisms

In this section, we present a powerful criterion that shows that for certain digraphs  $H$  the path-consistency procedure solves the  $H$ -colouring problem. Again, this condition was first

discovered in more general form by Feder and Vardi [40]; it subsumes many criteria that were studied in artificial intelligence and in graph theory before.

**Definition 3.2.** Let  $D$  be a set. A function  $f$  from  $D^3$  to  $D$  is called a *majority function* if  $f$  satisfies the following equations, for all  $x, y \in D$ :

$$f(x, x, y) = f(x, y, x) = f(y, x, x) = x$$

**Example 3.3.** As an example, let  $D$  be  $\{1, \dots, n\}$ , and consider the ternary *median* operation, which is defined as follows. Let  $x, y$  be three elements from  $D$ . We define  $\text{median}(x, x, y) = \text{median}(x, y, x) = \text{median}(y, x, x) := x$ . If  $x, y, z$  are pairwise distinct elements of  $D$ , suppose that  $\{x, y, z\} = \{a, b, c\}$ , where  $a < b < c$ . Then  $\text{median}(x, y, z)$  is defined to be  $b$ .  $\triangle$

If a digraph  $H$  has a polymorphism  $f$  that is a majority operation, then  $f$  is called a *majority polymorphism of  $H$* .

**Example 3.4.** Let  $H$  be the transitive tournament on  $n$  vertices,  $T_n$ . Suppose the vertices of  $T_n$  are the first natural numbers,  $\{1, \dots, n\}$ , and  $(u, v) \in E(T_n)$  if and only if  $u < v$ . Then the median operation is a polymorphism of  $T_n$ , because if  $u_1 < v_1$ ,  $u_2 < v_2$ , and  $u_3 < v_3$ , then clearly  $\text{median}(u_1, u_2, u_3) < \text{median}(v_1, v_2, v_3)$ . This yields a new proof that the  $H$ -colouring problem for  $H = T_n$  is tractable.  $\triangle$

**Theorem 3.5** (of [40]). *Let  $H$  be a finite digraph. If  $H$  has a majority polymorphism, then the  $H$ -colouring problem can be solved in polynomial time (by the path-consistency procedure).*

For the proof of Theorem 3.5 we need the following lemma.

**Lemma 3.6.** *Let  $G$  and  $H$  be finite digraphs. Let  $f$  be a polymorphism of  $H$  of arity  $k$  and let  $L := L(x, z)$  be the final list computed by the path-consistency procedure for  $x, z \in V(G)$ . Then  $f$  preserves  $L$ , i.e., if  $(u_1, w_1), \dots, (u_k, w_k) \in L$ , then  $(f(u_1, \dots, u_k), f(w_1, \dots, w_k)) \in L$ .*

*Proof.* Let  $(u_1, w_1), \dots, (u_k, w_k) \in L$ . We prove by induction over the execution of  $\text{PC}_H$  on  $G$  that at all times the pair  $(u, w) := (f(u_1, \dots, u_k), f(w_1, \dots, w_k))$  is contained in  $L$ . Initially, this is true because  $f$  is a polymorphism of  $H$ . For the inductive step, let  $y \in V(G)$ . By definition of the procedure, for each  $i \in \{1, \dots, k\}$  there exists  $v_i$  such that  $(u_i, v_i) \in L(x, y)$  and  $(v_i, w_i) \in L(y, z)$ . By the inductive assumption,  $(f(u_1, \dots, u_k), f(v_1, \dots, v_k)) \in L(x, y)$  and  $(f(v_1, \dots, v_k), f(w_1, \dots, w_k)) \in L(y, z)$ . Hence,  $(f(u_1, \dots, u_k), f(w_1, \dots, w_k))$  will not be removed in the next step of the algorithm.  $\square$

*Proof of Theorem 3.5.* Let  $f: V(H)^3 \rightarrow V(H)$  be a majority polymorphism of  $H$ . Clearly, if the path-consistency procedure derives the empty list for some pair  $(x, z)$  from  $V(G)^2$ , then there is no homomorphism from  $G$  to  $H$ .

Now suppose that after running the path-consistency procedure on  $G$  the list  $L(x, z)$  is non-empty for all pairs  $(x, z)$  from  $V(G)^2$ . We have to show that there exists a homomorphism from  $G$  to  $H$ . A homomorphism  $h$  from an induced subgraph  $G'$  of  $G$  to  $H$  is said to *preserve the lists* if  $(h(x), h(z)) \in L(x, z)$  for all  $x, z \in V(G')$ . The proof shows by induction on  $i$  that every homomorphism from a subgraph of  $G$  with  $i$  vertices that preserves the lists can be extended to any other vertex in  $G$  such that the resulting mapping is a homomorphism to  $H$  that again preserves the lists.

For the base case of the induction, observe that for all vertices  $x, z \in V(G)$  every mapping  $h$  from  $\{x, z\}$  to  $V(H)$  such that  $(h(x), h(z)) \in L(x, z)$  can be extended to every  $y \in V(G)$  such



that  $(h(x), h(y)) \in L(x, y)$  and  $(h(y), h(z)) \in L(y, z)$  (and hence preserves the lists), because otherwise the path-consistency procedure would have removed  $(h(x), h(z))$  from  $L(x, z)$ .

For the inductive step, let  $h'$  be any homomorphism from a subgraph  $G'$  of  $G$  on  $i \geq 3$  vertices to  $H$  that preserves the lists, and let  $x$  be any vertex of  $G$  not in  $G'$ . Let  $x_1, x_2$ , and  $x_3$  be some vertices of  $G'$ , and  $h'_j$  be the restriction of  $h'$  to  $V(G') \setminus \{x_j\}$ , for  $1 \leq j \leq 3$ . By inductive assumption,  $h'_j$  can be extended to  $x$  such that the resulting mapping  $h_j$  is a homomorphism to  $H$  that preserves the lists. We claim that the extension  $h$  of  $h'$  that maps  $x$  to  $f(h_1(x), h_2(x), h_3(x))$  is a homomorphism to  $H$  that preserves the lists.

For all  $y \in V(G')$ , we have to show that  $(h(x), h(y)) \in L(x, y)$  (and that  $(h(y), h(x)) \in L(y, x)$ , which can be shown analogously). If  $y \notin \{x_1, x_2, x_3\}$ , then  $h(y) = h'(y) = f(h'(y), h'(y), h'(y)) = f(h_1(y), h_2(y), h_3(y))$ , by the properties of  $f$ . Since  $(h_i(x), h_i(y)) \in L(x, y)$  for all  $i \in \{1, 2, 3\}$ , and since  $f$  preserves  $L(x, y)$  by Lemma 3.6, we have  $(h(x), h(y)) \in L(x, y)$ , and are done in this case.

Clearly,  $y$  can be equal to at most one of  $\{x_1, x_2, x_3\}$ . Suppose that  $y = x_1$  (the other two cases are analogous). There must be a vertex  $v \in V(H)$  such that  $(h_1(x), v) \in L(x, y)$  (otherwise the path-consistency procedure would have removed  $(h_1(x), h_1(x_1))$  from  $L(x, x_1)$ ). By the properties of  $f$ , we have  $h(y) = h'(y) = f(v, h'(y), h'(y)) = f(v, h_2(y), h_3(y))$ . Because  $(h_1(x), v), (h_2(x), h_2(y)), (h_3(x), h_3(y))$  are in  $L(x, y)$ , Lemma 3.6 implies that  $(h(x), h(y)) = (f(h_1(x), h_2(x), h_3(x)), f(v, h_2(y), h_3(y)))$  is in  $L(x, y)$ , and we are done.

We conclude that  $G$  has a homomorphism to  $H$ . □

**Corollary 3.7.** *The path-consistency procedure solves the  $H$ -colouring problem for  $H = T_n$ .*

Another class of examples of digraphs having a majority polymorphism are *unbalanced cycles*, i.e., orientations of  $C_n$  that do not homomorphically map to a directed path [38]. We only prove a weaker result here.

**Proposition 3.8.** *Directed cycles have a majority polymorphism.*

*Proof.* Let  $\vec{C}_n$  be a directed cycle. Let  $f$  be the ternary operation on the vertices of  $\vec{C}_n$  that maps  $u, v, w$  to  $u$  if  $u, v, w$  are pairwise distinct, and otherwise acts as a majority operation. We claim that  $f$  is a polymorphism of  $\vec{C}_n$ . Let  $(u, u'), (v, v'), (w, w') \in E(\vec{C}_n)$  be arcs. If  $u, v, w$  are all distinct, then  $u', v', w'$  are clearly all distinct as well, and hence  $(f(u, v, w), f(u', v', w')) = (u, u') \in E(\vec{C}_n)$ . Otherwise, if two elements of  $u, v, w$  are equal, say  $u = v$ , then  $u'$  and  $v'$  must be equal as well, and hence  $(f(u, v, w), f(u', v', w')) = (u, u') \in E(\vec{C}_n)$ . □

### Exercises.

53. Show that every orientation of a path has a majority polymorphism.  
 54. A *quasi majority operation* is an operation from  $V^3$  to  $V$  satisfying

$$f(x, x, y) = f(x, y, x) = f(y, x, x) = f(x, x, x)$$

for all  $x, y \in V$ . Use Theorem 1.5 to show that a finite undirected graph  $H$  has an  $H$ -colouring problem that can be solved in polynomial time if  $H$  has a quasi majority polymorphism, and is NP-complete otherwise.

55. There is only one unbalanced cycle  $H$  on four vertices that is a core and not the directed cycle (we have seen this digraph already in Exercise 40). Show that for this digraph  $H$  the  $H$ -colouring problem can be solved by the path-consistency procedure.

56. (\*) Show that every unbalanced orientation of a cycle has a majority polymorphism.
57. Modify the path-consistency procedure such that it can deal with instances of the precoloured  $H$ -colouring problem. Show that if  $H$  has a majority polymorphism, then the modified path-consistency procedure solves the precoloured  $H$ -colouring problem.
58. Modify the path-consistency procedure such that it can deal with instances of the list  $H$ -colouring problem. Show that if  $H$  has a *conservative* majority polymorphism, then the modified path-consistency procedure solves the list  $H$ -colouring problem.
59. An *interval graph*  $H$  is an (undirected) graph  $H = (V; E)$  such that there is an interval  $I_x$  of the real numbers for each  $x \in V$ , and  $(x, y) \in E$  if and only if  $I_x$  and  $I_y$  have a non-empty intersection. Note that with this definition interval graphs are necessarily *reflexive*, i.e.,  $(x, x) \in E$ . Show that the precoloured  $H$ -colouring problem for interval graphs  $H$  can be solved in polynomial time. Hint: use the modified path-consistency procedure in Exercise 57.
60. (\*) Let  $H$  be an (irreflexive) graph. Show that  $H$  has a conservative majority polymorphism if and only if  $H$  is an interval graph.
61. (\*) Let  $H$  be a reflexive graph. Show that  $H$  has a conservative majority polymorphism if and only if  $H$  is a *circular arc graph*, i.e.,  $H$  can be represented by arcs on a circle so that two vertices are adjacent if and only if the corresponding arcs intersect.
62. Show that the digraph  $(\mathbb{Z}; \{(x, y) \mid x - y = 1\})$  has a majority polymorphism.
63. Prove that  $H$ -colouring can be solved in polynomial time when  $H$  is the digraph from the previous exercise.
64. Show that the digraph  $H = (\mathbb{Z}; \{(x, y) \mid x - y \in \{1, 3\}\})$  has a majority polymorphism, and give a polynomial time algorithm for its  $H$ -colouring problem.

### 3.2 Testing for Majority Polymorphisms

In this section we show that the question whether a given digraph has a majority polymorphism can be decided in polynomial time.

**Theorem 3.9.** *There is a polynomial-time algorithm to decide whether a given digraph  $H$  has a majority polymorphism.*

*Proof.* The pseudo-code of the procedure can be found in Figure 5. Given  $H$ , we construct a new digraph  $G$  as follows. We start from the third power  $H^3$ , and precolour all vertices of the form  $(u, u, v)$ ,  $(u, v, u)$ ,  $(v, u, u)$ , and  $(u, u, u)$  with  $u$ . Let  $G$  be the resulting precoloured digraph. Note that there exists a homomorphism from  $G$  to  $H$  that respects the colours if and only if  $H$  has a majority polymorphism. To decide whether  $G$  has a homomorphism to  $H$ , we run the modification of  $PC_H$  for the precoloured  $H$ -colouring problem on  $G$  (see Exercise 57). If this algorithm rejects, then we can be sure that there is no homomorphism from  $G$  to  $H$  that respects the colours, and hence  $H$  has no majority polymorphism. Otherwise, we use the same idea as in the proof of Proposition 2.5: pick  $x \in V(G)$  and remove all but one pair  $(u, u)$  from  $L(x, x)$ . Then we continue with the execution of  $PC_H$ . If  $PC_H$  derives the empty list, we try the same with another pair  $(v, v)$  from  $L(x, x)$ . If we obtain failure for all pairs in  $L(x, x)$ ,

```

Majority-Test( $H$ )
Input: a finite digraph  $H$ .

Let  $G := H^3$ .
For all  $u, v \in V(H)$ , precolour the vertices  $(u, u, v), (u, v, u), (v, u, u), (u, u, u)$  with  $u$ .
If  $\text{PC}_H(G)$  derives the empty list, reject.
For each  $x \in V(G)$ 
  For each  $(u, u) \in L(x, x)$ 
    Found := False.
    For all  $y, z \in V(G)$ , let  $L'(y, z)$  be a copy of  $L(y, z)$ .
     $L'(x, x) := \{(u, u)\}$ .
    Run  $\text{PC}_H(G)$  with the lists  $L'$ .
    If this run does not derive the empty list
      For all  $y, z \in V(G)$ , set  $L(y, z) := L'(y, z)$ .
      Found := True.
  End For.
  If Found = False then reject.
End For.
Accept.

```

Figure 5: A polynomial-time algorithm to find majority polymorphisms.

then clearly there is no homomorphism from  $G$  to  $H$ , and we return ‘false’. Otherwise, if  $\text{PC}_H$  does not derive the empty list after removing all pairs but  $(u, u)$  from  $L(x, x)$ , we continue with another vertex  $y \in V(G)$ , setting  $L(y, y)$  to  $\{(u, u)\}$  for some  $(u, u) \in L(y, y)$ . We repeat this procedure until eventually all lists for pairs of the form  $(x, x)$  are singleton sets  $\{(u, u)\}$ ; the map that sends  $x$  to  $u$  is a homomorphism from  $G$  to  $H$  that respects the colours. In this case we return ‘true’. If there exists  $u \in V(G)$  such that  $\text{PC}_H$  detects an empty list for all  $(u, u) \in L(x, x)$  then the adaptation of  $\text{PC}_H$  for the precoloured CSP would have given an incorrect answer for the previously selected variable:  $\text{PC}_H$  did not detect the empty list even though the input was unsatisfiable. Hence,  $H$  cannot have a majority polymorphism. It is easy to see that the procedure described above has polynomial running time.  $\square$

### Exercises.

65. Modify the algorithm ‘Majority-Test’ to obtain an algorithm that tests whether a given digraph  $H$  has a quasi majority polymorphism.

### 3.3 Digraphs with a Maltsev Polymorphism

If a digraph  $H$  has a *majority* polymorphism, then the path-consistency procedure solves  $\text{CSP}(H)$ . How about digraphs  $H$  with a *minority* polymorphisms of  $H$ ? It turns out that this is an even stronger restriction.

**Definition 3.10.** A ternary function  $f: D^3 \rightarrow D$  is called

- a *minority operation* if it satisfies

$$\forall x, y \in D. f(y, x, x) = f(x, y, x) = f(x, x, y) = y$$

- and a *Maltsev operation* if it satisfies

$$\forall x, y \in D. f(y, x, x) = f(x, x, y) = y.$$

**Example 3.11.** Let  $D := \{0, \dots, n-1\}$ . Then the function  $f: D^3 \rightarrow D$  given by  $(x, y, z) \mapsto x - y + z \pmod n$  is a Maltsev operation, since  $x - x + z = z$  and  $x - z + z = x$ . For  $n = 2$ , this is even a minority operation. If  $n > 2$ , this function is not a minority, since then  $1 - 2 + 1 = 0 \not\equiv 2 \pmod n$ . Note that  $f$  is a polymorphism of  $\vec{C}_n$ . To see this, suppose that  $u_1 - v_1 \equiv 1 \pmod n$ ,  $u_2 - v_2 \equiv 1 \pmod n$ , and  $u_3 - v_3 \equiv 1 \pmod n$ . Then

$$\begin{aligned} f(u_1, u_2, u_3) &\equiv u_1 - u_2 + u_3 \equiv (v_1 + 1) - (v_2 + 1) + (v_3 + 1) \\ &\equiv f(v_1, v_2, v_3) + 1 \pmod n. \end{aligned} \quad \triangle$$

The following result appeared in 2011.

**Theorem 3.12** (Kazda [54]). *If a digraph  $G$  has a Maltsev polymorphism then  $G$  also has a majority polymorphism.*

Hence, for digraphs  $H$  with a Maltsev polymorphism, the strong path-consistency procedure solves the  $H$ -colouring problem, and in fact even the precoloured  $H$ -colouring problem. Theorem 3.12 is an immediate consequence of Theorem 3.18 below; to state it, we need the following concepts.

**Definition 3.13.** A digraph  $G$  is called *rectangular* if  $(x, y), (x', y), (x', y') \in E(G)$  implies that  $(x, y') \in E(G)$ .

We start with the fundamental observation: digraphs with a Maltsev polymorphism  $m$  are rectangular. This follows immediately from the definition of polymorphisms: we must have  $(m(x, x', x'), m(y, y, y')) \in E(G)$ , but  $m(x, x', x') = x$  and  $m(y, y, y') = y'$ , so  $(x, y') \in E(G)$ . The converse does not hold, as the following example shows.

**Example 3.14.** The digraph  $(\{a, b, c\}; \{(a, a), (a, b), (b, c), (c, c)\})$  is rectangular (draw a picture!), but has no Maltsev polymorphism  $m$ . Indeed, such an  $m$  would have to satisfy  $m(a, a, c) = c$  and  $m(a, c, c) = a$ . Note that

$$\begin{aligned} (m(a, a, c), m(a, b, c)) &\in E(G) \\ \text{and } (m(a, b, c), m(a, c, c)) &\in E(G), \end{aligned}$$

but  $G$  has no vertex  $x$  such that  $(c, x) \in E(G)$  and  $(x, a) \in E(G)$ . △

We are therefore interested in stronger consequences of the existence of a Maltsev polymorphism.

**Definition 3.15.** A digraph  $G$  is called  *$k$ -rectangular* if whenever  $G$  contains directed paths of length  $k$  from  $x$  to  $y$ , from  $x'$  to  $y$ , and from  $x'$  to  $y'$ , then also from  $x$  to  $y'$ . A digraph  $G$  is called *totally rectangular* if it is  $k$ -rectangular for all  $k \geq 1$ .

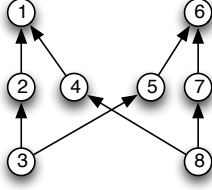


Figure 6: A totally rectangular digraph.

**Lemma 3.16.** *Every digraph with a Maltsev polymorphism  $m$  is totally rectangular.*

*Proof.* Let  $k \geq 1$ , and suppose that  $G$  is a digraph with directed paths  $(x_1, \dots, x_k)$ ,  $(y_1, \dots, y_k)$ , and  $(z_1, \dots, z_k)$  such that  $x_k = y_k$  and  $y_1 = z_1$ . We have to show that there exists a directed path  $(u_1, \dots, u_k)$  in  $G$  with  $u_1 = x_1$  and  $u_k = z_k$ . It can be verified that  $u_i := m(x_i, y_i, z_i)$  has the desired properties.  $\square$

An example of a totally rectangular digraph is given in Figure 6. The next lemma points out an important consequence of  $k$ -rectangularity.

**Lemma 3.17.** *Let  $G$  be a finite totally rectangular digraph with a cycle of net length  $d > 0$ . Then  $G$  contains a directed cycle of length  $d$ .*

*Proof.* Let  $C = (u_0, \dots, u_{k-1})$  be a cycle of  $G$  of net length  $d$ ; we prove the statement by induction on  $k$ . Clearly,  $C$  can be decomposed into maximal directed paths, that is, there is a minimal set  $\mathcal{D}$  of directed paths such that each pair  $(u_0, u_1), (u_1, u_2), \dots, (u_{k-1}, u_0)$  is contained in exactly one of the paths of  $\mathcal{D}$ . If the decomposition  $\mathcal{D}$  consists of a single directed path then we have found a directed cycle and are done. Let  $P$  be the shortest directed path of  $\mathcal{D}$ , leading from  $u$  to  $v$  in  $G$ . Then there are directed paths  $Q$  and  $Q'$  in  $\mathcal{D}$  such that  $Q$  starts in  $u$  and  $Q'$  ends in  $v$ , and  $P \neq Q$  or  $P \neq Q'$ . By assumption,  $|Q|, |Q'| \geq \ell := |P|$ . By  $\ell$ -rectangularity, there exists a directed path  $P'$  of length  $\ell$  from the vertex  $s$  of  $Q'$  at position  $|Q'| - \ell$  to the vertex  $t$  of  $Q$  at position  $\ell$ . Now we distinguish the following cases.

- $Q = Q'$ : the cycle that starts in  $s$ , follows the path  $Q$  until  $t$ , and then returns to  $s$  via the path  $P'$  is shorter than  $C$  but still has net length  $d$ .
- $Q \neq Q'$ : the cycle starting in  $s$ , following  $Q$  for the final  $|Q| - \ell$  vertices of  $Q$ , the cycle  $C$  until  $Q'$ , the first  $|Q'| - \ell$  vertices of  $Q'$  until  $t$ , and then  $P'$  back to  $s$  is a cycle which is shorter than  $C$  but still has net length  $d$ .

In both cases, the statement follows by induction.  $\square$

The following is a strengthening of Theorem 3.12; we only prove that 1 implies 2, and 2 implies 3, which suffices for the already mentioned consequence that for digraphs  $H$  with a Maltsev polymorphism, path consistency solves the  $H$ -colouring problem (cf. Exercise 52).

**Theorem 3.18** (Theorem 3.3 and Corollary 4.12 in [30]). *Let  $G$  be a finite digraph. Then the following are equivalent.*

1.  $G$  has a Maltsev polymorphism.

2.  $G$  is totally rectangular.
3. If  $G$  is acyclic, then the core of  $G$  is a directed path. Otherwise, the core of  $G$  is a disjoint union of directed cycles.
4.  $G$  has a minority and a majority polymorphism.

*Proof.* The implication from 4 to 1 is trivial since every minority operation is in particular a Maltsev operation. The implication from 1 to 2 is Lemma 3.16. For the implication from 2 to 3, let us assume that  $G$  is connected. The general case then follows by applying the following argument to each of its connected components, and the observation that directed paths homomorphically map to longer directed paths and to directed cycles.

We first consider the case that  $G$  is acyclic, and claim that in this case  $G$  is balanced, i.e., there exists a surjective homomorphism  $h$  from  $G$  to  $\vec{P}_n$  for some  $n \geq 1$ . Otherwise, there exist  $u, v \in V(G)$  and two paths  $P$  and  $Q$  from  $u$  to  $v$  of different net lengths  $\ell_1$  and  $\ell_2$  (see Exercise 13). Put these two paths together at  $u$  and  $v$  to form an unbalanced cycle  $C$ . Then Lemma 3.17 implies that  $G$  contains a directed cycle contrary to our assumptions.

Now, choose  $n$  with  $G \rightarrow \vec{P}_n$  minimal, and fix  $u \in h^{-1}(0)$  and  $v \in h^{-1}(n)$ . Then it is easy to see from total rectangularity that there must exist a path of length  $n$  in  $G$  from  $u$  to  $v$ , and hence the core of  $G$  is  $\vec{P}_n$ .

Now suppose that  $G$  contains a directed cycle; let  $C$  be the shortest directed cycle of  $G$ . We prove that  $G$  homomorphically maps to  $C$ . It is easy to see that it suffices to show that for any two vertices  $u, v$  of  $G$  and for any two paths  $P$  and  $Q$  from  $u$  to  $v$  we have that their net lengths are congruent modulo  $m := |C|$  (see Exercise 14). Suppose for contradiction that there are paths of net length  $\ell_1$  and  $\ell_2$  from  $u$  to  $v$  in  $G$  such that  $d := \ell_1 - \ell_2 \neq 0$  modulo  $m$ ; without loss of generality,  $\ell_2 < \ell_1$ , so  $d > 0$ . We can assume that  $u$  is an element of  $C$ , since otherwise we can choose a path  $S$  from a vertex of  $C$  to  $u$  by connectivity of  $G$ , and append  $S$  to both  $P$  and  $Q$ . We can also assume that  $d < m$  because if not, we can append  $C$  to  $Q$  to increase the length of  $Q$  by a multiple of  $m$ , until  $d = \ell_1 - \ell_2 < m$ . Lemma 3.17 then implies that  $G$  contains a directed cycle of length  $d$ , a contradiction to the choice of  $C$ .

For the missing implication from 3 to 4, we refer to [30] (Corollary 4.11).  $\square$

## Exercises.

66. Let  $H$  be the digraph  $(\{0, 1, \dots, 6\}; \{(0, 1), (1, 2), (3, 2), (4, 3), (4, 5), (5, 6)\})$ . For which  $k$  is it  $k$ -rectangular?

## 4 Logic

A *signature* is a set of relation and function symbols. The relation symbols are typically denoted by  $R, S, T, \dots$  and the function symbols are typically denoted by  $f, g, h, \dots$ ; each relation and function symbol is equipped with an *arity* from  $\mathbb{N}$ . A  $\tau$ -*structure*  $\mathfrak{A}$  consists of a set  $A$  (the *domain* or *base set*; we typically use the same letter in a different font) together with a relation  $R^{\mathfrak{A}} \subseteq A^k$  for each relation symbol  $R$  of arity  $k$  from  $\tau$ , and an operation  $f^{\mathfrak{A}}: A^k \rightarrow A$  for each function symbol  $f$  of arity  $k$  from  $\tau$ . Functions of arity 0 are allowed; they are also called *constants*. In this text it causes no harm to allow structures whose domain is empty. A  $\tau$ -structure  $\mathfrak{A}$  is called *finite* if its domain  $A$  is finite.

A *homomorphism*  $h$  from a  $\tau$ -structure  $\mathfrak{A}$  to a  $\tau$ -structure  $\mathfrak{B}$  is a function from  $A$  to  $B$  that *preserves* each relation and each function: that is,

- if  $(a_1, \dots, a_k)$  is in  $R^{\mathfrak{A}}$ , then  $(h(a_1), \dots, h(a_k))$  must be in  $R^{\mathfrak{B}}$ ;
- for all  $a_1, \dots, a_k \in A$  we have  $h(f^{\mathfrak{A}}(a_1, \dots, a_k)) = f^{\mathfrak{B}}(h(a_1), \dots, h(a_k))$ .

An *isomorphism* is a bijective homomorphism  $h$  such that the inverse mapping  $h^{-1}: B \rightarrow A$  that sends  $h(x)$  to  $x$  is a homomorphism, too.

A *relational* structure is a  $\tau$ -structure where  $\tau$  only contains relation symbols, and an *algebra* (in the sense of universal algebra) is a  $\tau$ -structure where  $\tau$  only contains function symbols. In this section, we will work with relational structures only; algebras will appear in Section 7.

#### 4.1 Primitive positive formulas

A first-order  $\tau$ -formula  $\phi(x_1, \dots, x_n)$  is called *primitive positive* (in database theory also *conjunctive query*) if it is of the form

$$\exists x_{n+1}, \dots, x_\ell (\psi_1 \wedge \dots \wedge \psi_m)$$

where  $\psi_1, \dots, \psi_m$  are *atomic  $\tau$ -formulas*, i.e., formulas of the form  $R(y_1, \dots, y_k)$  with  $R \in \tau$  and  $y_i \in \{x_1, \dots, x_\ell\}$ , of the form  $y = y'$  for  $y, y' \in \{x_1, \dots, x_\ell\}$ , or  $\top$  and  $\perp$  (for *true* and *false*). As usual, formulas without free variables are called *sentences*. If  $\mathfrak{A}$  is a  $\tau$ -structure and  $\phi$  a  $\tau$ -sentence, then we write  $\mathfrak{A} \models \phi$  if  $\mathfrak{A}$  satisfies  $\phi$  (i.e.,  $\phi$  holds in  $\mathfrak{A}$ ).

Note that if we would require that all our structures have a non-empty domain, we would not need the symbol  $\top$  since we can use the primitive positive sentence  $\exists x. x = x$  to express it. It is possible to rephrase the  $H$ -colouring problem and its variants using primitive positive sentences.

**Definition 4.1.** Let  $\mathfrak{B}$  be a structure with a finite relational signature  $\tau$ . Then  $\text{CSP}(\mathfrak{B})$  is the computational problem of deciding whether a given primitive positive  $\tau$ -sentence  $\phi$  is true in  $\mathfrak{B}$ .

The given primitive positive  $\tau$ -sentence  $\phi$  is also called an *instance* of  $\text{CSP}(\mathfrak{B})$ . The conjuncts of an instance  $\phi$  are called the *constraints* of  $\phi$ . A mapping from the variables of  $\phi$  to the elements of  $B$  that is a satisfying assignment for the quantifier-free part of  $\phi$  is also called a *solution* to  $\phi$ .

**Example 4.2** (Disequality constraints). Consider the problem  $\text{CSP}(\{1, 2, \dots, n\}; \neq)$ . An instance of this problem can be viewed as an (existentially quantified) set of variables, some linked by disequality<sup>2</sup> constraints. Such an instance holds in  $(\{1, 2, \dots, n\}; \neq)$  if and only if the graph whose vertices are the variables, and whose edges are the disequality constraints, has a homomorphism to  $K_n$ .  $\triangle$

---

<sup>2</sup>We deliberately use the word *disequality* instead of *inequality*, since we reserve the word *inequality* for the relation  $x \leq y$ .

## Exercises.

67. Show that  $\text{CSP}(\{0, 1\}; \{(0, 1), (1, 0), (0, 0)\}, \{(0, 1), (1, 0), (1, 1)\}, \{(1, 1), (1, 0), (0, 0)\})$  can be solved in polynomial time.
68. Generalise the notion of *direct products* from digraphs (Definition 1.1) to general relational  $\tau$ -structures.
69. Generalise the arc-consistency procedure from digraphs to general relational structures.
70. Does the arc-consistency procedure solve  $\text{CSP}(\mathfrak{B})$  where  $\mathfrak{B}$  has domain  $B = \{0, 1, 2, 3\}$ , the unary relation  $U_i^{\mathfrak{B}}$  for every  $i \in B$ , and the binary relations  $B^4 \setminus \{(0, 0)\}$  and  $\{(1, 2), (2, 3), (3, 1), (0, 0)\}$ ?
71. Generalise the path-consistency procedure from digraphs to general relational structures.
72. Verify that the structure  $\mathfrak{B}$  from Exercise 70 has the binary idempotent commutative polymorphism  $*$  defined as  $1*2 = 2, 2*3 = 3, 3*1 = 1$ , and  $0*b = b$  for all  $b \in \{1, 2, 3\}$ . Verify that  $*$  satisfies ‘restricted associativity’, i.e.,  $x*(x*y) = (x*x)*y$  for all  $x, y \in B$  (and since it is additionally idempotent and commutative it is called a *2-semilattice*).
73. Does the structure  $\mathfrak{B}$  from Exercise 70 have a majority polymorphism?
74. (\*) Does the path-consistency procedure solve  $\text{CSP}(\mathfrak{B})$  for the structure  $\mathfrak{B}$  from Exercise 70?

## 4.2 From Structures to Formulas

To every finite relational  $\tau$ -structure  $\mathfrak{A}$  we can associate a  $\tau$ -sentence, called the *canonical conjunctive query* of  $\mathfrak{A}$ , and denoted by  $\phi(\mathfrak{A})$ . The variables of this sentence are the elements of  $\mathfrak{A}$ , all of which are existentially quantified in the quantifier prefix of the formula, which is followed by the conjunction of all formulas of the form  $R(a_1, \dots, a_k)$  for  $R \in \tau$  and tuples  $(a_1, \dots, a_k) \in R^{\mathfrak{A}}$ .

For example, the canonical conjunctive query  $\phi(K_3)$  of the complete graph on three vertices  $K_3$  is the formula

$$\exists u \exists v \exists w (E(u, v) \wedge E(v, u) \wedge E(v, w) \wedge E(w, v) \wedge E(u, w) \wedge E(w, u)) .$$

The proof of the following proposition is straightforward.

**Proposition 4.3.** *Let  $\mathfrak{B}$  be a structure with finite relational signature  $\tau$ , and let  $\mathfrak{A}$  be a finite  $\tau$ -structure. Then there is a homomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}$  if and only if  $\mathfrak{B} \models \phi(\mathfrak{A})$ .*

## 4.3 From Formulas to Structures

To present a converse of Proposition 4.3, we define the *canonical structure*  $\mathfrak{S}(\phi)$  (in database theory this structure is called the *canonical database*) of a primitive positive  $\tau$ -sentence, which is a relational  $\tau$ -structure defined as follows. We require that  $\phi$  does not contain  $\perp$ . If  $\phi$  contains an atomic formula of the form  $x = y$ , we remove it from  $\phi$ , and replace all



occurrences of  $x$  in  $\phi$  by  $y$ . Repeating this step if necessary, we may assume that  $\phi$  does not contain atomic formulas of the form  $x = y$ .

Then the domain of  $\mathfrak{S}(\phi)$  is the set of variables that occur in  $\phi$ . There is a tuple  $(v_1, \dots, v_k)$  in a relation  $R$  of  $\mathfrak{S}(\phi)$  if and only if  $\phi$  contains the conjunct  $R(v_1, \dots, v_k)$ . The following is similarly straightforward as Proposition 4.3.

**Proposition 4.4.** *Let  $\mathfrak{B}$  be a relational  $\tau$ -structure and let  $\phi$  be a primitive positive  $\tau$ -sentence that does not contain  $\perp$ . Then  $\mathfrak{B} \models \phi$  if and only if  $\mathfrak{S}(\phi)$  homomorphically maps to  $\mathfrak{B}$ .*

Due to Proposition 4.4 and Proposition 4.3, we may freely switch between the homomorphism and the logic perspective whenever this is convenient. In particular, instances of  $\text{CSP}(\mathfrak{B})$  can from now on be either finite structures  $\mathfrak{A}$  or primitive positive sentences  $\phi$ .

Note that the  $H$ -colouring problem, the precoloured  $H$ -colouring problem, and the list  $H$ -colouring problem can be viewed as constraint satisfaction problems for appropriately chosen relational structures.

#### 4.4 Primitive Positive Definability

Let  $\mathfrak{A}$  be a  $\tau$ -structure, and let  $\mathfrak{A}'$  be a  $\tau'$ -structure with  $\tau \subseteq \tau'$ . If  $\mathfrak{A}$  and  $\mathfrak{A}'$  have the same domain and  $R^{\mathfrak{A}} = R^{\mathfrak{A}'}$  for all  $R \in \tau$ , then  $\mathfrak{A}$  is called the  $\tau$ -*reduct* (or simply *reduct*) of  $\mathfrak{A}'$ , and  $\mathfrak{A}'$  is called a  $\tau'$ -*expansion* (or simply *expansion*) of  $\mathfrak{A}$ . If  $\mathfrak{A}$  is a structure, and  $R$  is a relation over the domain of  $\mathfrak{A}$ , then we denote the expansion of  $\mathfrak{A}$  by  $R$  by  $(\mathfrak{A}, R)$ .

If  $\mathfrak{A}$  is a  $\tau$ -structure, and  $\phi(x_1, \dots, x_k)$  is a formula with  $k$  free variables  $x_1, \dots, x_k$ , then the *relation defined by  $\phi$*  is the relation

$$\{(a_1, \dots, a_k) \mid \mathfrak{A} \models \phi(a_1, \dots, a_k)\}.$$

If the formula is primitive positive, then this relation is called *primitive positive definable*.

**Example 4.5.** The relation  $\{(a, b) \in \{0, 1, 2, 3, 4\}^2 \mid a \neq b\}$  is primitive positive definable in  $C_5$ : the primitive positive definition is

$$\exists p_1, p_2 (E(x_1, p_1) \wedge E(p_1, p_2) \wedge E(p_2, x_2)) \quad \triangle$$

**Example 4.6.** The non-negative integers are primitively positively definable in  $(\mathbb{Z}; 0, 1, +, *)$ , namely by the following formula  $\phi(x)$  which states that  $x$  is the sum of four squares:

$$\exists x_1, x_2, x_3, x_4 (x = x_1^2 + x_2^2 + x_3^2 + x_4^2).$$

Clearly, every integer that satisfies  $\phi(x)$  is non-negative; the converse is the famous four-square theorem of Lagrange [44].  $\triangle$

The following lemma says that we can expand structures by primitive positive definable relations without changing the complexity of the corresponding CSP. Hence, primitive positive definitions are an important tool to prove NP-hardness: to show that  $\text{CSP}(\mathfrak{B})$  is NP-hard, it suffices to show that there is a primitive positive definition of a relation  $R$  such that  $\text{CSP}(\mathfrak{B}, R)$  is already known to be NP-hard. Stronger tools to prove NP-hardness of CSPs will be introduced in Sections 4.6 and 4.8.

**Lemma 4.7** (Jeavons, Cohen, Gyssens [52]). *Let  $\mathfrak{B}$  be a structure with finite relational signature, and let  $R$  be a relation that has a primitive positive definition in  $\mathfrak{B}$ . Then  $\text{CSP}(\mathfrak{B})$  and  $\text{CSP}(\mathfrak{B}, R)$  are linear-time equivalent.*

*Proof.* It is clear that  $\text{CSP}(\mathfrak{B})$  reduces to the new problem. So suppose that  $\phi$  is an instance of  $\text{CSP}((\mathfrak{B}, R))$ . Replace each conjunct  $R(x_1, \dots, x_l)$  of  $\phi$  by its primitive positive definition  $\psi(x_1, \dots, x_l)$ . Move all quantifiers to the front, such that the resulting formula is in *prenex normal form* and hence primitive positive. Finally, equalities can be eliminated one by one: for equality  $x = y$ , remove  $y$  from the quantifier prefix, and replace all remaining occurrences of  $y$  by  $x$ . Let  $\phi'$  be the formula obtained in this way.

It is straightforward to verify that  $\phi$  is true in  $(\mathfrak{B}, R)$  if and only if  $\phi'$  is true in  $\mathfrak{B}$ , and it is also clear that  $\phi'$  can be constructed in linear time in the representation size of  $\phi$ .  $\square$

Recall from Section 1.3 that  $\text{CSP}(K_5)$  is NP-hard. Since the edge relation of  $K_5$  is primitively positively definable in  $C_5$  (Example 4.5), Lemma 4.7 implies that  $\text{CSP}(C_5)$  is NP-hard, too.

### Exercises.

75. Show that the relation  $R := \{(a, b, c) \in \{1, 2, 3\}^3 \mid a = b \text{ or } b = c \text{ or } a = c\}$  has a primitive positive definition over  $K_3$ .

76. Show that the relation  $\neq$  on  $\{1, 2, 3\}$  has a primitive positive definition in the structure  $(\{1, 2, 3\}; R, \{1\}, \{2\}, \{3\})$  where  $R$  is the relation from the previous exercise.

77. Let  $R_+, R_*$  be the relations defined as follows.

$$R_+ := \{(x, y, z) \in \mathbb{Q}^3 \mid x + y = z\}$$

$$R_* := \{(x, y, z) \in \mathbb{Q}^3 \mid x * y = z\}.$$

Show that  $R_*$  is primitive positive definable in the structure  $(\mathbb{Q}; R_+, \{(x, y) \mid y = x^2\})$ .

78. Let  $B$  be any set, and for  $n \in \mathbb{N}$  define the relation  $P_B^{2n}$  of arity  $2n$  as follows.

$$P_B^{2n} := \{(x_1, y_1, x_2, y_2, \dots, x_n, y_n) \in B^{2n} \mid \bigvee_{i \in \{1, \dots, n\}} x_i = y_i\}$$

Show that for every  $n$  the relation  $P_B^{2n}$  has a primitive positive definition in  $(B; P_B^4)$ .

79. (\*) Let  $n \geq 4$ . Is there a primitive positive definition of  $\neq$  over the structure

$$M_n := (\{1, \dots, n\}; R, \{1\}, \{2\}, \dots, \{n\})$$

where  $R := \{(1, \dots, 1), (2, \dots, 2), \dots, (n, \dots, n), (1, 2, \dots, n)\}$ ?

## 4.5 Cores and Constants

An *automorphism* of a structure  $\mathfrak{B}$  with domain  $B$  is an isomorphism between  $\mathfrak{B}$  and itself. The set of all automorphisms  $\alpha$  of  $\mathfrak{B}$  is denoted by  $\text{Aut}(\mathfrak{B})$ , and forms a permutation group. When  $G$  is a permutation group on a set  $B$ , and  $b \in B$ , then a set of the form

$$S = \{\alpha(b) \mid \alpha \in G\}$$

is called an *orbit* of  $G$  (the *orbit of  $b$* ). Let  $(b_1, \dots, b_k)$  be a  $k$ -tuple of elements of  $\mathfrak{B}$ . A set of the form

$$S = \{(\alpha b_1, \dots, \alpha b_k) \mid \alpha \in \text{Aut}(\mathfrak{B})\}$$

is called an *orbit of  $k$ -tuples* of  $\mathfrak{B}$ ; it is an orbit of the componentwise action of  $G$  on the set  $B^k$  of  $k$ -tuples from  $B$ .

**Lemma 4.8.** *Let  $\mathfrak{B}$  be a structure with a finite relational signature and domain  $B$ , and let  $R = \{(b_1, \dots, b_k)\}$  be a  $k$ -ary relation that only contains one tuple  $(b_1, \dots, b_k) \in B^k$ . If the orbit of  $(b_1, \dots, b_k)$  in  $\mathfrak{B}$  is primitive positive definable, then there is a polynomial-time reduction from  $\text{CSP}(\mathfrak{B}, R)$  to  $\text{CSP}(\mathfrak{B})$ .*

*Proof.* Let  $\phi$  be an instance of  $\text{CSP}(\mathfrak{B}, R)$  with variable set  $V$ . If  $\phi$  contains two constraints  $R(x_1, \dots, x_k)$  and  $R(y_1, \dots, y_k)$ , then replace each occurrence of  $y_1$  by  $x_1$ , then each occurrence of  $y_2$  by  $x_2$ , and so on, and finally each occurrence of  $y_k$  by  $x_k$ . We repeat this step until all constraints that involve  $R$  are imposed on the same tuple of variables  $(x_1, \dots, x_k)$ . Replace  $R(x_1, \dots, x_k)$  by the primitive positive definition  $\theta$  of its orbit in  $\mathfrak{B}$ . Finally, move all quantifiers to the front, such that the resulting formula  $\psi$  is in prenex normal form and thus an instance of  $\text{CSP}(\mathfrak{B})$ . Clearly,  $\psi$  can be computed from  $\phi$  in polynomial time. We claim that  $\phi$  is true in  $(\mathfrak{B}, R)$  if and only if  $\psi$  is true in  $\mathfrak{B}$ .

Suppose  $\phi$  has a solution  $s: V \rightarrow B$ . Since  $(b_1, \dots, b_k)$  satisfies  $\theta$ , we can extend  $s$  to the existentially quantified variables of  $\theta$  to obtain a solution for  $\psi$ . In the opposite direction, suppose that  $s'$  is a solution to  $\psi$  over  $\mathfrak{B}$ . Let  $s$  be the restriction of  $s'$  to  $V$ . Since  $(s(x_1), \dots, s(x_k))$  satisfies  $\theta$ , it lies in the same orbit as  $(b_1, \dots, b_k)$ . Thus, there exists an automorphism  $\alpha$  of  $\mathfrak{B}$  that maps  $(s(x_1), \dots, s(x_k))$  to  $(b_1, \dots, b_k)$ . Then the extension of the map  $x \mapsto \alpha s(x)$  that maps variables  $y_i$  of  $\phi$  that have been replaced by  $x_i$  in  $\psi$  to the value  $b_i$  is a solution to  $\phi$  over  $(\mathfrak{B}, R)$ .  $\square$

The definition of cores can be extended from finite digraphs to finite structures: as in the case of finite digraphs, we require that every endomorphism be an automorphism. All results we proved for cores of digraphs remain valid for cores of structures. In particular, every finite structure  $\mathfrak{C}$  is homomorphically equivalent to a core structure  $\mathfrak{B}$ , which is unique up to isomorphism (see Section 1.4). The following proposition can be shown as in the proof of Proposition 1.8.

**Proposition 4.9.** *Let  $\mathfrak{B}$  be a finite core structure. Then the orbits of  $k$ -tuples of  $\mathfrak{B}$  are primitive positive definable.*

Proposition 4.9 and Lemma 4.8 have the following consequence.

**Corollary 4.10.** *Let  $\mathfrak{B}$  be a finite core with a finite relational signature. Let  $b_1, \dots, b_n \in B$ . Then  $\text{CSP}(\mathfrak{B})$  and  $\text{CSP}(\mathfrak{B}, \{b_1\}, \dots, \{b_n\})$  are polynomial time equivalent.*

## Exercises.

80. Show that if  $m$  is the number of orbits of  $k$ -tuples of a finite structure  $\mathfrak{A}$ , and  $\mathfrak{C}$  is the core of  $\mathfrak{A}$ , then  $\mathfrak{C}$  has at most  $m$  orbits of  $k$ -tuples.
81. Show that if  $\mathfrak{A}$  is a finite structure, and  $\mathfrak{C}$  its core, and if  $\mathfrak{A}$  and  $\mathfrak{C}$  have the same number of orbits of pairs, then  $\mathfrak{A}$  and  $\mathfrak{C}$  are isomorphic.

## 4.6 Primitive Positive Interpretations

Primitive positive interpretations are a powerful generalisation of primitive positive definability that can be used to also relate structures with *different* domains. They are a special case of (*first-order*) *interpretations* that play an important role in model theory (see, e.g., [50]).

If  $C$  and  $D$  are sets and  $g: C \rightarrow D$  is a map, then the *kernel* of  $g$  is the equivalence relation  $E$  on  $C$  where  $(c, c') \in E$  if  $g(c) = g(c')$ . For  $c \in C$ , we denote by  $c/E$  the equivalence class of  $c$  in  $E$ , and by  $C/E$  the set of all equivalence classes of elements of  $C$ . The *index* of  $E$  is defined to be  $|C/E|$ .

**Definition 4.11.** Let  $\sigma$  and  $\tau$  be relational signatures, let  $\mathfrak{A}$  be a  $\tau$ -structure, and let  $\mathfrak{B}$  be a  $\sigma$ -structure. A *primitive positive interpretation*  $I$  of  $\mathfrak{B}$  in  $\mathfrak{A}$  consists of

- a natural number  $d$ , called the *dimension* of  $I$ ,
- a primitive positive  $\tau$ -formula  $\delta_I(x_1, \dots, x_d)$ , called the *domain formula*,
- for each atomic  $\sigma$ -formula  $\phi(y_1, \dots, y_k)$  a primitive positive  $\tau$ -formula  $\phi_I(\bar{x}_1, \dots, \bar{x}_k)$ , called the *defining formulas*, and
- the *coordinate map*: a surjective map  $h: D \rightarrow B$  where

$$D := \{(a_1, \dots, a_d) \in A^d \mid \mathfrak{A} \models \delta_I(a_1, \dots, a_d)\}$$

such that for all atomic  $\sigma$ -formulas  $\phi$  and all tuples  $\bar{a}_i \in D$

$$\mathfrak{B} \models \phi(h(\bar{a}_1), \dots, h(\bar{a}_k)) \Leftrightarrow \mathfrak{A} \models \phi_I(\bar{a}_1, \dots, \bar{a}_k).$$

Sometimes, the same symbol is used for the interpretation  $I$  and the coordinate map. Note that the dimension  $d$ , the set  $D$ , and the coordinate map  $h$  determine the defining formulas up to logical equivalence; hence, we sometimes denote an interpretation by  $I = (d, D, h)$ . Note that the kernel of  $h$  coincides with the relation defined by  $(y_1 = y_2)_I$ , for which we also write  $=_I$ , the defining formula for equality. Also note that the structures  $\mathfrak{A}$  and  $\mathfrak{B}$  and the coordinate map determine the defining formulas of the interpretation up to logical equivalence.

**Example 4.12.** Let  $G$  be a digraph and let  $F$  be an equivalence relation on  $V(G)$ . Then  $G/F$  is the digraph whose vertices are the equivalence classes of  $F$ , and where  $S$  and  $T$  are adjacent if there are  $s \in S$  and  $t \in T$  such that  $\{s, t\} \in E(G)$ . If  $F$  has a primitive positive definition in  $G$ , then  $G/F$  has a primitive positive interpretation in  $G$ .  $\triangle$

**Example 4.13.** The field of rational numbers  $(\mathbb{Q}; 0, 1, +, *)$  has a primitive positive 2-dimensional interpretation  $I$  in  $(\mathbb{Z}; 0, 1, +, *)$ . Example 4.6 presented a primitive positive definition  $\phi(x)$  of the set of non-negative integers. The interpretation is now given as follows.

- The domain formula  $\delta_I(x, y)$  is  $y \geq 1$  (using  $\phi(x)$ , it is straightforward to express this with a primitive positive formula);
- The formula  $=_I(x_1, y_1, x_2, y_2)$  is  $x_1 y_2 = x_2 y_1$ ;
- The formula  $0_I(x, y)$  is  $x = 0$ , the formula  $1_I(x, y)$  is  $x = y$ ;
- The formula  $+_I(x_1, y_1, x_2, y_2, x_3, y_3)$  is  $y_3 * (x_1 * y_2 + x_2 * y_1) = x_3 * y_1 * y_2$ ;
- The formula  $*_I(x_1, y_1, x_2, y_2, x_3, y_3)$  is  $x_1 * x_2 * y_3 = x_3 * y_1 * y_2$ .  $\triangle$

**Theorem 4.14.** *Let  $\mathfrak{B}$  and  $\mathfrak{C}$  be structures with finite relational signatures. If there is a primitive positive interpretation of  $\mathfrak{B}$  in  $\mathfrak{C}$ , then there is a polynomial-time reduction from  $\text{CSP}(\mathfrak{B})$  to  $\text{CSP}(\mathfrak{C})$ .*

*Proof.* Let  $d$  be the dimension of the primitive positive interpretation  $I$  of the  $\tau$ -structure  $\mathfrak{B}$  in the  $\sigma$ -structure  $\mathfrak{C}$ , let  $\delta_I(x_1, \dots, x_d)$  be the domain formula, and let  $h: \delta_I(\mathfrak{C}^d) \rightarrow D(\mathfrak{B})$  be the coordinate map. Let  $\phi$  be an instance of  $\text{CSP}(\mathfrak{B})$  with variable set  $U = \{x_1, \dots, x_n\}$ . We construct an instance  $\psi$  of  $\text{CSP}(\mathfrak{C})$  as follows. For distinct variables  $V := \{y_1^1, \dots, y_n^d\}$ , we set  $\psi_1$  to be the formula

$$\bigwedge_{1 \leq i \leq n} \delta_I(y_i^1, \dots, y_i^d).$$

Let  $\psi_2$  be the conjunction of the formulas  $\theta_I(y_{i_1}^1, \dots, y_{i_1}^d, \dots, y_{i_k}^1, \dots, y_{i_k}^d)$  over all conjuncts  $\theta = R(x_{i_1}, \dots, x_{i_k})$  of  $\phi$ . By moving existential quantifiers to the front, the sentence

$$\exists y_1^1, \dots, y_n^d (\psi_1 \wedge \psi_2)$$

can be re-written to a primitive positive  $\sigma$ -sentence  $\psi$ , and clearly  $\psi$  can be constructed in polynomial time in the size of  $\phi$ .

We claim that  $\phi$  is true in  $\mathfrak{B}$  if and only if  $\psi$  is true in  $\mathfrak{C}$ . Suppose that  $f: V \rightarrow C$  satisfies all conjuncts of  $\psi$  in  $\mathfrak{C}$ . Hence, by construction of  $\psi$ , if  $\phi$  has a conjunct  $\theta = R(x_{i_1}, \dots, x_{i_k})$ , then

$$\mathfrak{C} \models \theta_I((f(y_{i_1}^1), \dots, f(y_{i_1}^d)), \dots, (f(y_{i_k}^1), \dots, f(y_{i_k}^d))).$$

By the definition of interpretations, this implies that

$$\mathfrak{B} \models R(h(f(y_{i_1}^1), \dots, f(y_{i_1}^d)), \dots, h(f(y_{i_k}^1), \dots, f(y_{i_k}^d))).$$

Hence, the mapping  $g: U \rightarrow B$  that sends  $x_i$  to  $h(f(y_i^1), \dots, f(y_i^d))$  satisfies all conjuncts of  $\phi$  in  $\mathfrak{B}$ .

Now, suppose that  $f: U \rightarrow B$  satisfies all conjuncts of  $\phi$  over  $\mathfrak{B}$ . Since  $h$  is a surjective mapping from  $\delta_I(\mathfrak{C}^d)$  to  $B$ , there are elements  $c_i^1, \dots, c_i^d$  in  $\mathfrak{C}$  such that  $h(c_i^1, \dots, c_i^d) = f(x_i)$ , for all  $i \in \{1, \dots, n\}$ . We claim that the mapping  $g: V \rightarrow C$  that maps  $y_i^j$  to  $c_i^j$  satisfies  $\psi$  in  $\mathfrak{C}$ . By construction, any constraint in  $\psi$  either comes from  $\psi_1$  or from  $\psi_2$ . If it comes from  $\psi_1$  then it must be of the form  $\delta_I(y_i^1, \dots, y_i^d)$ , and is satisfied since the pre-image of  $h$  is  $\delta_I(\mathfrak{C}^d)$ . If the constraint comes from  $\psi_2$ , then it must be a conjunct of a formula  $\theta_I(y_{i_1}^1, \dots, y_{i_1}^d, \dots, y_{i_k}^1, \dots, y_{i_k}^d)$  that was introduced for a constraint  $\theta = R(x_{i_1}, \dots, x_{i_k})$  in  $\phi$ . It therefore suffices to show that

$$\mathfrak{C} \models \theta_I(g(y_{i_1}^1), \dots, g(y_{i_1}^d), \dots, g(y_{i_k}^1), \dots, g(y_{i_k}^d)).$$

By assumption,  $R(f(x_{i_1}), \dots, f(x_{i_k}))$  holds in  $\mathfrak{B}$ . By the choice of  $c_1^1, \dots, c_n^d$ , this shows that  $R(h(c_{i_1}^1, \dots, c_{i_1}^d), \dots, h(c_{i_k}^1, \dots, c_{i_k}^d))$  holds in  $\mathfrak{C}$ . By the definition of interpretations, this is the case if and only if  $\theta_I(c_{i_1}^1, \dots, c_{i_1}^d, \dots, c_{i_k}^1, \dots, c_{i_k}^d)$  holds in  $\mathfrak{C}$ , which is what we had to show.  $\square$

In many hardness proofs we use Theorem 4.14 in the following way.

**Corollary 4.15.** *Let  $\mathfrak{B}$  be a finite relational structure. If there is a primitive positive interpretation of  $K_3$  in  $\mathfrak{B}$ , then  $\text{CSP}(\mathfrak{B})$  is NP-hard.*

*Proof.* This is a direct consequence of Theorem 4.14 and the fact that  $\text{CSP}(K_3)$  is NP-hard (see, e.g., [42]).  $\square$

Indeed,  $K_3$  is one of the most expressive finite structures, in the following sense.

**Theorem 4.16.** *If  $n \geq 3$  then every finite structure has a primitive positive interpretation in  $K_n$ .*

*Proof.* Let  $\mathfrak{A}$  be a finite  $\tau$ -structure with the domain  $A = \{1, \dots, k\}$ . Our interpretation  $I$  of  $\mathfrak{A}$  in  $K_n$  is  $2k$ -dimensional. The domain formula  $\delta_I(x_1, \dots, x_k, x'_1, \dots, x'_k)$  expresses that for exactly one  $i \leq k$  we have  $x_i = x'_i$ . Note that this formula is preserved by all permutations of  $\{1, \dots, k\}$ . We will see in Proposition 5.12 that every such formula is equivalent to a primitive positive formula over  $K_n$ . Equality is interpreted by the formula

$$=_I (x_1, \dots, x_k, x'_1, \dots, x'_k, y_1, \dots, y_k, y'_1, \dots, y'_k) := \bigwedge_{i=1}^k ((x_i = x'_i) \Leftrightarrow (y_i = y'_i))$$

Note that  $=_I$  defines an equivalence relation on the set of all  $2k$ -tuples  $(u_1, \dots, u_k, u'_1, \dots, u'_k)$  that satisfy  $\delta_I$ . The coordinate map sends this tuple to  $i$  if and only if  $u_i = u'_i$ . When  $R \in \tau$  is  $m$ -ary, then the formula  $R(x_1, \dots, x_m)_I$  is any primitive positive formula which is equivalent to the following disjunction of conjunctions with  $2mk$  variables  $x_{1,1}, \dots, x_{m,k}, x'_{1,1}, \dots, x'_{m,k}$ : for each tuple  $(t_1, \dots, t_m) \in R^{\mathfrak{A}}$  the disjunction contains the conjunct  $\bigwedge_{i \leq m} x_{i,t_i} = x'_{i,t_i}$ ; again, Proposition 5.12 implies that such a primitive positive formula exists.  $\square$

Primitive positive interpretations can be composed: if

- $\mathfrak{C}_1$  has a  $d_1$ -dimensional pp-interpretation  $I_1$  in  $\mathfrak{C}_2$ , and
- $\mathfrak{C}_2$  has an  $d_2$ -dimensional pp-interpretation  $I_2$  in  $\mathfrak{C}_3$ ,

then  $\mathfrak{C}_1$  has a natural  $(d_1 d_2)$ -dimensional pp-interpretation in  $\mathfrak{C}_3$ , which we denote by  $I_1 \circ I_2$ . To formally describe  $I_1 \circ I_2$ , suppose that the signature of  $\mathfrak{C}_i$  is  $\tau_i$  for  $i = 1, 2, 3$ , and that  $I_1 = (d_1, S_1, h_1)$  and  $I_2 = (d_2, S_2, h_2)$ . When  $\phi$  is a primitive positive  $\tau_2$ -formula, let  $\phi_{I_2}$  denote the  $\tau_3$ -formula obtained from  $\phi$  by replacing each atomic  $\tau_2$ -formula  $\psi$  in  $\phi$  by the  $\tau_3$ -formula  $\psi_{I_2}$ . Note that  $\phi_{I_2}$  is again primitive positive. The coordinate map of  $I_1 \circ I_2$  is defined by

$$(a_1^1, \dots, a_{d_2}^1, \dots, a_1^{d_1}, \dots, a_{d_2}^{d_1}) \mapsto h_1(h_2(a_1^1, \dots, a_{d_2}^1), \dots, h_2(a_1^{d_1}, \dots, a_{d_2}^{d_1})) .$$

Two pp-interpretations  $I_1$  and  $I_2$  of  $\mathfrak{B}$  in  $\mathfrak{A}$  are called *homotopic*<sup>3</sup> if the relation

$$\{(\bar{x}, \bar{y}) \mid I_1(\bar{x}) = I_2(\bar{y})\}$$

of arity  $d_1 + d_2$  is pp-definable in  $\mathfrak{A}$ . Note that  $\text{id}_{\mathfrak{C}}$  is a pp-interpretation of  $\mathfrak{C}$  in  $\mathfrak{C}$ , called the *identity interpretation* of  $\mathfrak{C}$  (in  $\mathfrak{C}$ ).

**Definition 4.17.** Two structures  $\mathfrak{A}$  and  $\mathfrak{B}$  with an interpretation  $I$  of  $\mathfrak{B}$  in  $\mathfrak{A}$  and an interpretation  $J$  of  $\mathfrak{A}$  in  $\mathfrak{B}$  are called *mutually pp-interpretable*. If both  $I \circ J$  and  $J \circ I$  are homotopic to the identity interpretation (of  $\mathfrak{A}$  and of  $\mathfrak{B}$ , respectively), then we say that  $\mathfrak{A}$  and  $\mathfrak{B}$  are *primitively positively bi-interpretable (via  $I$  and  $J$ )*.

<sup>3</sup>We follow the terminology from [2].

## 4.7 Reduction to Binary Signatures

In this section we prove that every structure  $\mathfrak{C}$  with a relational signature of maximal arity  $m \in \mathbb{N}$  is primitively positively bi-interpretable with a *binary structure*  $\mathfrak{B}$ , i.e., a relational structure where every relation symbol has arity at most two. Moreover, if  $\mathfrak{C}$  has a finite signature, then  $\mathfrak{B}$  can be chosen to have a finite signature, too. It follows from Theorem 4.14 that every CSP is polynomial-time equivalent to a binary CSP. This transformation is known under the name *dual encoding* [32, 35]. We want to stress that the transformation works for relational structures with domains of arbitrary cardinality.

A  $d$ -dimensional primitive positive interpretation  $I$  of  $\mathfrak{B}$  in  $\mathfrak{A}$  is called *full* if for every  $R \subseteq B^k$  we have that  $R$  is primitively positively definable in  $\mathfrak{B}$  if and only if the relation  $I^{-1}(R)$  of arity  $kd$  is primitively positively definable in  $\mathfrak{A}$ . Note that every structure with a primitive positive interpretation in  $\mathfrak{A}$  is a reduct of a structure with a full primitive positive interpretation in  $\mathfrak{A}$ .

**Definition 4.18.** Let  $\mathfrak{C}$  be a structure and  $d \in \mathbb{N}$ . Then a  *$d$ -th full power of  $\mathfrak{C}$*  is a structure  $\mathfrak{D}$  with domain  $C^d$  such that the identity map on  $C^d$  is a full  $d$ -dimensional primitive positive interpretation of  $\mathfrak{D}$  in  $\mathfrak{C}$ .

In particular, for all  $i, j \in \{1, \dots, d\}$  the relation

$$E_{i,j} := \{((x_1, \dots, x_d), (y_1, \dots, y_d)) \mid x_1, \dots, x_d, y_1, \dots, y_d \in C \text{ and } x_i = y_j\}$$

is primitively positively definable in  $\mathfrak{D}$ .

**Proposition 4.19.** *Let  $\mathfrak{C}$  be a structure and  $\mathfrak{D}$  a  $d$ -th full power of  $\mathfrak{C}$  for  $d \geq 1$ . Then  $\mathfrak{C}$  and  $\mathfrak{D}$  are primitively positively bi-interpretable.*

*Proof.* Let  $I$  be the identity map on  $C^d$  which is a full interpretation of  $\mathfrak{D}$  in  $\mathfrak{C}$ . Our interpretation  $J$  of  $\mathfrak{C}$  in  $\mathfrak{D}$  is one-dimensional and the coordinate map is the first projection. The domain formula is *true* and the pre-image of the equality relation in  $\mathfrak{C}$  under the coordinate map has the primitive positive definition  $E_{1,1}(x, y)$ . To define the pre-image of a  $k$ -ary relation  $R$  of  $\mathfrak{C}$  under the coordinate map it suffices to observe that the  $k$ -ary relation

$$S := \{((a_{1,1}, \dots, a_{1,d}), \dots, (a_{k,1}, \dots, a_{k,d})) \mid (a_{1,1}, \dots, a_{k,1}) \in R\}$$

is primitively positively definable in  $\mathfrak{D}$  and  $J(S) = R$ .

To show that  $\mathfrak{C}$  and  $\mathfrak{D}$  are primitively positive bi-interpretable we prove that  $I \circ J$  and  $J \circ I$  are pp-homotopic to the identity interpretation. The relation

$$\{(u_0, u_1, \dots, u_k) \mid u_0 = I(J(u_1), \dots, J(u_k)), u_1, \dots, u_k \in C^{k+1}\}$$

has the primitive positive definition  $\bigwedge_{i \in \{1, \dots, k\}} E_{i,1}(u_0, u_i)$  and the relation

$$\{(v_0, v_1, \dots, v_k) \mid v_0 = J(I(v_1, \dots, v_k)), v_1, \dots, v_k \in D^{k+1}\}$$

has the primitive positive definition  $v_0 = v_1$ . □

Note that for every relation  $R$  of arity  $k \leq d$  of  $\mathfrak{C}$ , in a  $d$ -th full power  $\mathfrak{D}$  of  $\mathfrak{C}$  the unary relation

$$R' := \{(a_1, \dots, a_d) \mid (a_1, \dots, a_k) \in R\}$$

must be primitively positively definable. We now define a particular full power.

**Definition 4.20.** Let  $\mathfrak{C}$  be a relational structure with maximal arity  $m$  and let  $d \geq m$ . Then the structure  $\mathfrak{B} := \mathfrak{C}^{[d]}$  with domain  $C^d$  is defined as follows:

- for every relation  $R \subseteq C^k$  of  $\mathfrak{C}$  the structure  $\mathfrak{B}$  has the unary relation  $R' \subseteq B = C^d$  defined above, and
- for all  $i, j \in \{1, \dots, d\}$  the structure  $\mathfrak{B}$  has the binary relation symbol  $E_{i,j}$ .

It is clear that the signature of  $\mathfrak{B}$  is finite if the signature of  $\mathfrak{C}$  is finite. Also note that the signature of  $\mathfrak{C}^{[d]}$  is always binary.

**Lemma 4.21.** *Let  $\mathfrak{C}$  be a relational structure with maximal arity  $m$  and let  $d \geq m$ . Then the binary structure  $\mathfrak{C}^{[d]}$  is a full power of  $\mathfrak{C}$ .*

*Proof.* The identity map is a  $d$ -dimensional primitive positive interpretation  $I$  of  $\mathfrak{B} := \mathfrak{C}^{[d]}$  in  $\mathfrak{C}$ . Our interpretation  $J$  of  $\mathfrak{C}$  in  $\mathfrak{B}$  is one-dimensional and the coordinate map is the first projection. The domain formula is *true* and the pre-image of the equality relation in  $\mathfrak{C}$  under the coordinate map has the primitive positive definition  $E_{1,1}(x, y)$ . The pre-image of the relation  $R$  of  $\mathfrak{C}$  under the coordinate map is defined by the primitive positive formula

$$\exists y \left( \bigwedge_{i \in \{1, \dots, k\}} E_{1,i}(x_i, y) \wedge R'(y) \right).$$

The proof that  $I \circ J$  and  $J \circ I$  are pp-homotopic to the identity interpretation is as in the proof of Proposition 4.19. □

**Corollary 4.22.** *For every structure  $\mathfrak{C}$  with maximal arity  $m$  there exists a structure  $\mathfrak{B}$  with maximal arity 2 such that  $\mathfrak{B}$  and  $\mathfrak{C}$  are primitively positively bi-interpretable. If the signature of  $\mathfrak{C}$  is finite, then the signature of  $\mathfrak{B}$  can be chosen to be finite, too.*

*Proof.* An immediate consequence of Lemma 4.21 and Proposition 4.19. □

## 4.8 The Structure-Building Operators H, C, and I

In the previous three sections we have seen several conditions on  $\mathfrak{A}$  and  $\mathfrak{B}$  that imply that  $\text{CSP}(\mathfrak{A})$  reduces to  $\text{CSP}(\mathfrak{B})$ ; in this section we compare them. Let  $\mathcal{C}$  be a class of structures. We write

1.  $\text{H}(\mathcal{C})$  for the class of structures homomorphically equivalent to structures in  $\mathcal{C}$ .
2.  $\text{C}(\mathcal{C})$  for the class of all structures obtained by expanding a core structure in  $\mathcal{C}$  by singleton relations  $\{a\}$ . In the setting of relational structures, they play the role of constants (which formally are operation symbols of arity 0).
3.  $\text{I}(\mathcal{C})$  for the class of all structures with a primitive positive interpretation in a structure from  $\mathcal{C}$ .

Let  $\mathcal{D}$  be the smallest class containing  $\mathcal{C}$  and closed under H, C, and I. Barto, Opršal, and Pinsker [12] showed that  $\mathcal{D} = \text{H}(\text{I}(\mathcal{C}))$ . In other words, if there is a chain of applications of the three operators H, C, and I to derive  $\mathfrak{A}$  from  $\mathfrak{B}$ , then there is also a two-step chain to derive  $\mathfrak{A}$  from  $\mathfrak{B}$ , namely by interpreting a structure  $\mathfrak{B}'$  that is homomorphically equivalent to  $\mathfrak{A}$ . This insight is conceptually important for the CSP since it leads to a better understanding of the power of the available tools.



**Proposition 4.23** (from [12]). *Suppose that  $\mathfrak{B}$  is a core, and that  $\mathfrak{C}$  is the expansion of  $\mathfrak{B}$  by a relation of the form  $\{c\}$  for  $c \in B$ . Then  $\mathfrak{C}$  is homomorphically equivalent to a structure with a primitive positive interpretation in  $\mathfrak{B}$ . In symbols,*

$$\mathbf{C}(\mathfrak{C}) \subseteq \mathbf{H}(\mathbf{I}(\mathfrak{C})).$$

*Proof.* By Proposition 4.9, the orbit  $O$  of  $c$  has a primitive positive definition  $\phi(x)$  in  $\mathfrak{B}$ . We give a 2-dimensional primitive positive interpretation of a structure  $\mathfrak{A}$  with the same signature  $\tau$  as  $\mathfrak{C}$ . The domain formula  $\delta_I(x_1, x_2)$  for  $\mathfrak{A}$  is  $\phi(x_2)$ . Let  $R \in \tau$ . If  $R$  is from the signature of  $\mathfrak{B}$  and has arity  $k$  then

$$R^{\mathfrak{A}} := \{((a_1, b_1), \dots, (a_k, b_k)) \in A^k \mid (a_1, \dots, a_k) \in R^{\mathfrak{B}} \text{ and } b_1 = \dots = b_k \in O\}.$$

Otherwise,  $R^{\mathfrak{C}}$  is of the form  $\{c\}$  and we define  $R^{\mathfrak{A}} := \{(a, a) \mid a \in O\}$ . It is clear that  $\mathfrak{A}$  has a primitive positive interpretation in  $\mathfrak{B}$ .

We claim that  $\mathfrak{A}$  and  $\mathfrak{C}$  are homomorphically equivalent. The homomorphism from  $\mathfrak{C}$  to  $\mathfrak{A}$  is given by  $a \mapsto (a, c)$ :

- if  $(a_1, \dots, a_k) \in R^{\mathfrak{C}} = R^{\mathfrak{B}}$  then  $((a_1, c), \dots, (a_k, c)) \in R^{\mathfrak{A}}$ ;
- the relation  $R^{\mathfrak{C}} = \{c\}$  is preserved since  $(c, c) \in R^{\mathfrak{A}}$ .

To define a homomorphism  $h$  from  $\mathfrak{A}$  to  $\mathfrak{C}$  we pick for each  $a \in O$  an automorphism  $\alpha_a \in \text{Aut}(\mathfrak{B})$  such that  $\alpha_a(a) = c$ . Note that  $b \in O$  since  $\mathfrak{B} \models \delta(a, b)$ , and we define  $h(a, b) := \alpha_b(a)$ . To check that this is indeed a homomorphism, let  $R \in \tau$  be  $k$ -ary, and let  $t = ((a_1, b_1), \dots, (a_k, b_k)) \in R^{\mathfrak{A}}$ . Then  $b_1 = \dots = b_k =: b \in O$  and we have that  $h(t) = (\alpha_b(a_1), \dots, \alpha_b(a_k))$  is in  $R^{\mathfrak{C}}$  since  $(a_1, \dots, a_k) \in R^{\mathfrak{B}} = R^{\mathfrak{C}}$  and  $\alpha_b$  preserves  $R^{\mathfrak{B}} = R^{\mathfrak{C}}$ . If  $R^{\mathfrak{A}} = \{(a, a) \mid a \in O\}$ , then  $R$  is preserved as well, because  $h((a, a)) = \alpha_a(a) = c \in \{c\} = R^{\mathfrak{C}}$ .  $\square$

**Theorem 4.24** (from [12]). *Suppose that  $\mathfrak{A}$  can be obtained from  $\mathfrak{C}$  by applying  $\mathbf{H}$ ,  $\mathbf{C}$ , and  $\mathbf{I}$ . Then  $\mathfrak{A} \in \mathbf{H}(\mathbf{I}(\mathfrak{C}))$ . That is,  $\mathfrak{A}$  is homomorphically equivalent to a structure with a primitive positive interpretation in a structure from  $\mathfrak{C}$ .*

*Proof.* We have to show that  $\mathbf{H}(\mathbf{I}(\mathfrak{C}))$  is closed under  $\mathbf{H}$ ,  $\mathbf{C}$ , and  $\mathbf{I}$ . Homomorphic equivalence is transitive so  $\mathbf{H}(\mathbf{H}(\mathfrak{C})) \subseteq \mathbf{H}(\mathfrak{C})$ .

We show that if  $\mathfrak{A}$  and  $\mathfrak{B}$  are homomorphically equivalent, and  $\mathfrak{C}$  has a  $d$ -dimensional primitive positive interpretation  $I_1$  in  $\mathfrak{B}$ , then  $\mathfrak{C}$  is homomorphically equivalent to a structure  $\mathfrak{D}$  with a  $d$ -dimensional primitive positive interpretation  $I_2$  in  $\mathfrak{A}$ . Let  $h_1: A \rightarrow B$  be the homomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}$ , and  $h_2$  the homomorphism from  $\mathfrak{B}$  to  $\mathfrak{A}$ . The interpreting formulas of  $I_2$  are the same as the interpreting formulas of  $I_1$ ; this describes the structure  $\mathfrak{D}$  up to isomorphism. We claim that the map  $g_1(I_2(a_1, \dots, a_d)) := I_1(h_1(a_1), \dots, h_1(a_d))$  is a homomorphism from  $\mathfrak{D}$  to  $\mathfrak{C}$ . Indeed, for a  $k$ -ary relation symbol from the signature of  $\mathfrak{C}$  and  $\mathfrak{D}$ , let  $((a_1^1, \dots, a_d^1), \dots, (a_1^k, \dots, a_d^k)) \in R^{\mathfrak{D}}$ ; hence, the  $dk$ -tuple  $(a_1^1, \dots, a_d^1, \dots, a_1^k, \dots, a_d^k)$  satisfies the primitive positive defining formula for  $R(x_1^1, \dots, x_d^k)$ , and

$$(h_1(a_1^1), \dots, h_1(a_d^1), \dots, h_1(a_1^k), \dots, h_1(a_d^k))$$

satisfies this formula, too. This in turn implies that

$$(I_1(h_1(a_1^1), \dots, h_1(a_d^1)), \dots, I_1(h_1(a_1^k), \dots, h_1(a_d^k))) \in R^{\mathfrak{C}}.$$

Similarly,  $g_2(I_1(b_1, \dots, b_d)) := I_2(h_2(b_1), \dots, h_2(b_d))$  is a homomorphism from  $\mathfrak{C}$  to  $\mathfrak{D}$ . So we conclude that

$$\begin{aligned} \text{I}(\text{H}(\text{I}(\mathfrak{C}))) &\subseteq \text{H}(\text{I}(\text{I}(\mathfrak{C}))) \\ &\subseteq \text{H}(\text{I}(\mathfrak{C})) \end{aligned}$$

because primitive positive interpretability is transitive, too. Finally, Proposition 4.23 shows that

$$\begin{aligned} \text{C}(\text{H}(\text{I}(\mathfrak{C}))) &\subseteq \text{H}(\text{I}(\text{H}(\text{I}(\mathfrak{C})))) \\ &\subseteq \text{H}(\text{I}(\mathfrak{C})) \end{aligned}$$

where the last inclusion again follows from the observations above.  $\square$

There are finite structures  $\mathfrak{B}$  all of whose polymorphisms are idempotent such that  $\text{H}(\text{I}(\mathfrak{B}))$  is strictly larger than  $\text{I}(\mathfrak{B})$ .

**Example 4.25.** Let  $\mathfrak{B}$  be the structure with domain  $(\mathbb{Z}_2)^2$  and signature  $\{R_{a,b} \mid a, b \in \mathbb{Z}_2\}$  such that

$$R_{a,b}^{\mathfrak{B}} := \{(x, y, z) \in ((\mathbb{Z}_2)^2)^3 \mid x + y + z = (a, b)\}.$$

Let  $\mathfrak{B}'$  be the reduct of  $\mathfrak{B}$  with the signature  $\tau := \{R_{0,0}, R_{1,0}\}$ . Let  $\mathfrak{A}$  be the  $\tau$ -structure with domain  $\mathbb{Z}_2$  such that for  $a = 0$  and  $a = 1$

$$R_{a,0}^{\mathfrak{A}} := \{(x, y, z) \in (\mathbb{Z}_2)^3 \mid x + y + z = a\}.$$

Now observe that

- $(x_1, x_2) \mapsto x_1$  is a homomorphism from  $\mathfrak{B}'$  to  $\mathfrak{A}$ , and  $x \mapsto (x, 0)$  is a homomorphism from  $\mathfrak{A}$  to  $\mathfrak{B}'$ . Therefore  $\mathfrak{A} \in \text{H}(\mathfrak{B}')$ .
- Trivially,  $\mathfrak{B}' \in \text{I}(\mathfrak{B})$  and consequently  $\mathfrak{A} \in \text{H}(\text{I}(\mathfrak{B}))$ .
- All polymorphisms of  $\mathfrak{B}$  are idempotent.

We finally show that  $\mathfrak{A} \notin \text{I}(\mathfrak{B})$ . Suppose for contradiction that there is a pp-interpretation of  $\mathfrak{A}$  in  $\mathfrak{B}$  with coordinate map  $c: C \rightarrow A$  where  $C \subseteq B^n$  is primitive positive definable in  $\mathfrak{B}$ . The kernel  $K$  of  $c$  has a primitive positive definition  $\phi$  in  $\mathfrak{B}$ . The two equivalence classes of  $K$  are pp-definable relations over  $\mathfrak{B}$ , too: the formula  $\exists x(\phi(x, y) \wedge R_{a,b}(x))$  defines the equivalence class of  $(a, b)$ . But the relations with a primitive positive definition in  $\mathfrak{B}$  are precisely affine linear subspaces of the vector space  $(\mathbb{Z}_2)^2$ , so their cardinality must be a power of 4. And two powers of 4 cannot add up to a power of 4.  $\triangle$

We will revisit primitive positive interpretations in Section 7 where we study them from a universal-algebraic perspective.

## 5 Relations and Operations

### 5.1 Operation Clones

For  $n \geq 1$  and a set  $D$  (the *domain*), denote by  $\mathcal{O}_D^{(n)}$  the set  $D^{D^n} := (D^n \rightarrow D)$  of  $n$ -ary functions on  $D$ . The elements of  $\mathcal{O}_D^{(n)}$  will typically be called the *operations* of arity  $n$  on  $D$ , and  $D$  will be called the *domain*. The set of all operations on  $D$  of finite arity will be denoted by  $\mathcal{O}_D := \bigcup_{n \geq 1} \mathcal{O}_D^{(n)}$ . An *operation clone* (over  $D$ ) is a subset  $\mathcal{C}$  of  $\mathcal{O}_D$  satisfying the following two properties:

- $\mathcal{C}$  contains all projections, that is, for all  $1 \leq k \leq n$  it contains the operation  $\pi_k^n \in \mathcal{O}_D^{(n)}$  defined by  $\pi_k^n(x_1, \dots, x_n) = x_k$ , and
- $\mathcal{C}$  is *closed under composition*, that is, for all  $f \in \mathcal{C} \cap \mathcal{O}_D^{(n)}$  and  $g_1, \dots, g_n \in \mathcal{C} \cap \mathcal{O}_D^{(m)}$  it contains the operation  $f(g_1, \dots, g_n) \in \mathcal{O}_D^{(m)}$  defined by

$$(x_1, \dots, x_m) \mapsto f(g_1(x_1, \dots, x_m), \dots, g_n(x_1, \dots, x_m)).$$

A *clone* is an abstraction of an operation clone that will be introduced later in the course. In the literature, operation clones are often called clones, or *concrete clones*; we prefer to use the terms ‘operation clone’ and ‘clone’ in analogy to ‘permutation group’ and ‘group’.

If  $\mathcal{C}$  is an operation clone, then  $\mathcal{C}'$  is called a *subclone* of  $\mathcal{C}$  if  $\mathcal{C}'$  is an operation clone and  $\mathcal{C}' \subseteq \mathcal{C}$ . If  $\mathcal{F}$  is a set of functions, we write  $\langle \mathcal{F} \rangle$  for the smallest operation clone  $\mathcal{C}$  which contains  $\mathcal{F}$ , and call  $\mathcal{C}$  the clone *generated* by  $\mathcal{F}$ . Note that the set of all clones over a set  $B$  forms a lattice: the meet of two operation clones  $\mathcal{C}$  and  $\mathcal{D}$  is their intersection  $\mathcal{C} \cap \mathcal{D}$  (which is again a clone!); the join of  $\mathcal{C}$  and  $\mathcal{D}$  is the clone generated by their union,  $\langle \mathcal{C} \cup \mathcal{D} \rangle$ .

### 5.2 Inv-Pol

The most important source of operation clones in this text are *polymorphism clones* of di-graphs and, more generally, structures. For simplicity, we only discuss relational structures; the step to structures that also involve function symbols is straightforward.

Let  $f$  be from  $\mathcal{O}_B^{(n)}$ , and let  $R \subseteq B^m$  be a relation. Then we say that  $f$  *preserves*  $R$  (and that  $R$  is *invariant under*  $f$ ) if  $f(r_1, \dots, r_n) \in R$  whenever  $r_1, \dots, r_n \in R$ , where  $f(r_1, \dots, r_n)$  is calculated componentwise. If  $\mathfrak{B}$  is a relational structure with domain  $B$  then  $\text{Pol}(\mathfrak{B})$  contains precisely those operations that preserve  $\mathfrak{B}$ . It is easy to verify that  $\text{Pol}(\mathfrak{B})$  is an operation clone. Conversely, if  $\mathcal{F}$  is a set of operations on  $B$ , then we write  $\text{Inv}(\mathcal{F})$  for the set of all relations on  $B$  that are invariant under all functions in  $\mathcal{F}$ . It will be convenient to define the operator  $\text{Pol}$  also for sets  $\mathcal{R}$  of relations over  $B$ , writing  $\text{Pol}(\mathcal{R})$  for the set of operations of  $\mathcal{O}_B$  that preserve all relations from  $\mathcal{R}$ .

**Proposition 5.1.** *Let  $\mathfrak{B}$  be any relational structure. Then  $\text{Inv}(\text{Pol}(\mathfrak{B}))$  contains the set of all relations that are primitive positive definable in  $\mathfrak{B}$ .*

*Proof.* Suppose that  $R$  is  $k$ -ary, has a primitive positive definition  $\psi(x_1, \dots, x_k)$ , and let  $f$  be an  $l$ -ary polymorphism of  $\mathfrak{B}$ . To show that  $f$  preserves  $R$ , let  $t_1, \dots, t_l$  be  $k$ -tuples from  $R$ . Let  $x_{k+1}, \dots, x_n$  be the existentially quantified variables of  $\psi$ . Write  $s_i$  for the  $n$ -tuple which extends the  $k$ -tuple  $t_i$  such that  $s_i$  satisfies the quantifier-free part  $\psi'(x_1, \dots, x_k, x_{k+1}, \dots, x_n)$  of  $\psi$ . Then the tuple  $f(s_1, \dots, s_l)$  satisfies  $\psi'$  since  $f$  is a polymorphism. This shows that  $\mathfrak{B} \models \psi(f(t_1, \dots, t_l))$  which is what we had to show.  $\square$

**Theorem 5.2** (of [20,43]). *Let  $\mathfrak{B}$  be a finite relational structure. A relation  $R$  has a primitive positive definition in  $\mathfrak{B}$  if and only if  $R$  is preserved by all polymorphisms of  $\mathfrak{B}$ .*

*Proof.* One direction has been shown in Proposition 5.1. For the other direction, let

$$(a_1^1, \dots, a_k^1), \dots, (a_1^w, \dots, a_k^w)$$

be an enumeration of  $R$ . Let  $b_1, b_2, \dots, b_\ell$  be an enumeration of  $B^w$ . Let  $\psi$  be the canonical query of  $\mathfrak{B}^w$  (see Exercise 68 for the definition of  $\mathfrak{B}^w$  and Section 4.2 for the definition of canonical queries). We claim that the primitive positive formula  $\psi'(x_1, \dots, x_k)$  obtained from  $\psi$  by adding the conjuncts  $x_i = b_j$  if  $(a_i^1, \dots, a_i^w) = b_j$  is a primitive positive definition of  $R$ . Indeed, for every  $(a_1, \dots, a_k) \in R$  there exists a  $j \in \{1, \dots, w\}$  such that  $(a_1, \dots, a_k) = (b_1^j, \dots, b_k^j)$ . Then the elements  $b_1^j, \dots, b_\ell^j$  provide witnesses for the existentially quantified variables showing that  $(a_1, \dots, a_k) = (b_1^j, \dots, b_k^j)$  satisfies the formula.

Conversely, suppose that  $(a_1, \dots, a_k)$  satisfies  $\psi'$ . Then  $(a_1, \dots, a_k) = (b_1^j, \dots, b_k^j)$  for some  $j \leq \ell$ , and  $\psi'$  contains the conjuncts  $x_i = b_j$ . The witnesses for the existentially quantified variables  $b_1, \dots, b_\ell$  define a homomorphism  $f$  from  $\mathfrak{B}^w$  to  $\mathfrak{B}$ . Then  $f$  is a polymorphism of  $\mathfrak{B}$  and by assumption preserves  $R$ . Since the tuples  $(a_1^1, \dots, a_k^1), \dots, (a_1^w, \dots, a_k^w)$  are from  $R$  and  $f((a_1^1, \dots, a_k^1), \dots, (a_1^w, \dots, a_k^w)) = (a_1, \dots, a_k)$ , we obtain that  $(a_1, \dots, a_k) \in R$ .  $\square$

**Corollary 5.3.** *The complexity of  $\text{CSP}(\mathfrak{B})$  only depends on  $\text{Pol}(\mathfrak{B})$ . If  $\mathfrak{C}$  is such that  $\text{Pol}(\mathfrak{B}) \subseteq \text{Pol}(\mathfrak{C})$ , then  $\text{CSP}(\mathfrak{C})$  reduces in linear time to  $\text{CSP}(\mathfrak{B})$ .*

*Proof.* Direct consequence of Theorem 5.2 and Lemma 4.7.  $\square$

### Exercises.

82. Let  $R_+, R_*$  be the relations as defined in Exercise 77. Show that  $R_*$  is not primitively positively definable in the structure  $(\mathbb{Q}; R_+, \{(x, y) \mid y \geq x^2\})$ .

### 5.3 Essentially Unary Clones

We say that an operation  $f: B^k \rightarrow B$  is *essentially unary* if there is an  $i \in \{1, \dots, k\}$  and a unary operation  $f_0$  such that  $f(x_1, \dots, x_k) = f_0(x_i)$  for all  $x_1, \dots, x_k \in B$ . Operations that are not essentially unary are called *essential*.<sup>4</sup> We say that  $f$  *depends on argument  $i$*  if there are  $r, s \in B^k$  such that  $f(r) \neq f(s)$  and  $r_j = s_j$  for all  $j \in \{1, \dots, k\} \setminus \{i\}$ .

**Lemma 5.4.** *Let  $f \in \mathcal{O}_B$  be an operation. Then the following are equivalent.*

1.  $f$  is essentially unary.
2.  $f$  preserves  $P_B^3 := \{(a, b, c) \in B^3 \mid a = b \text{ or } b = c\}$ .
3.  $f$  preserves  $P_B^4 := \{(a, b, c, d) \in B^4 \mid a = b \text{ or } c = d\}$ .
4.  $f$  depends on at most one argument.

<sup>4</sup>This is standard in clone theory, and it makes sense also when studying the complexity of CSPs, since the essential operations are those that are essential for complexity classification.

*Proof.* Let  $k$  be the arity of  $f$ . The implication from (1) to (2) is obvious, since unary operations clearly preserve  $P_B^3$ .

To show the implication from (2) to (3), we show the contrapositive, and assume that  $f$  violates  $P_B^4$ . By permuting arguments of  $f$ , we can assume that there are 4-tuples  $a^1, \dots, a^k \in P_B^4$  with  $f(a^1, \dots, a^k) \notin P_B^4$  and  $l \leq k$  such that in  $a^1, \dots, a^l$  the first two coordinates are equal, and in  $a^{l+1}, \dots, a^k$  the last two coordinates are equal. Let  $c := (a_1^1, \dots, a_1^l, a_4^{l+1}, \dots, a_4^k)$ . Since  $f(a^1, \dots, a^k) \notin P_B^4$  we have  $f(a_1^1, \dots, a_1^l) \neq f(a_2^1, \dots, a_2^l)$ , and therefore  $f(c) \neq f(a_1^1, \dots, a_1^l)$  or  $f(c) \neq f(a_2^1, \dots, a_2^l)$ . Let  $d = (a_1^1, \dots, a_1^l)$  in the first case, and  $d = (a_2^1, \dots, a_2^l)$  in the second case. Likewise, we have  $f(c) \neq f(a_3^1, \dots, a_3^l)$  or  $f(c) \neq f(a_4^1, \dots, a_4^l)$ , and let  $e = (a_3^1, \dots, a_3^l)$  in the first, and  $e = (a_4^1, \dots, a_4^l)$  in the second case. Then for each  $i \leq k$ , the tuple  $(d_i, c_i, e_i)$  is from  $P_B^3$ , but  $(f(d), f(c), f(e)) \notin P_B^3$ .

The proof of the implication from (3) to (4) is again by contraposition. Suppose  $f$  depends on the  $i$ -th and  $j$ -th argument,  $1 \leq i \neq j \leq k$ . Hence there exist tuples  $a_1, b_1, a_2, b_2 \in B^k$  such that  $a_1, b_1$  and  $a_2, b_2$  only differ at the entries  $i$  and  $j$ , respectively, and such that  $f(a_1) \neq f(b_1)$  and  $f(a_2) \neq f(b_2)$ . Then  $(a_1(l), b_1(l), a_2(l), b_2(l)) \in P_B^4$  for all  $l \leq k$ , but  $(f(a_1), f(b_1), f(a_2), f(b_2)) \notin P_B^4$ , which shows that  $f$  violates  $P_B^4$ .

For the implication from (4) to (1), suppose that  $f$  depends only on the first argument. Let  $i \leq k$  be minimal such that there is an operation  $g$  with  $f(x_1, \dots, x_k) = g(x_1, \dots, x_i)$ . If  $i = 1$  then  $f$  is essentially unary and we are done. Otherwise, observe that since  $f$  does not depend on the  $i$ -th argument, neither does  $g$ , and so there is an  $(i-1)$ -ary operation  $g'$  such that for all  $x_1, \dots, x_n \in B$  we have  $f(x_1, \dots, x_n) = g(x_1, \dots, x_i) = g'(x_1, \dots, x_{i-1})$ , contradicting the choice of  $i$ .  $\square$

## 5.4 Minimal Clones

A *trivial* clone is a clone all of whose operations are projections. Note that it follows from Lemma 5.4 that for any set  $B = \{b_1, \dots, b_n\}$  the clone  $\text{Pol}(B; P_B^4, \{b_1\}, \dots, \{b_n\})$  is trivial.

**Definition 5.5.** A clone  $\mathcal{C}$  is *minimal* if it is non-trivial, and for every non-trivial  $\mathcal{E} \subseteq \mathcal{C}$  we have  $\mathcal{E} = \mathcal{C}$ .

Recall that the smallest clone that contains a set of operations  $\mathcal{F}$  is called the *clone generated by  $\mathcal{F}$* , and denoted by  $\langle \mathcal{F} \rangle$ ; if  $g \in \langle \mathcal{F} \rangle$  and  $\mathcal{F} = \{f\}$ , then we say that  $f$  *generates*  $g$ .

**Definition 5.6.** An operation  $f \in \mathcal{O}_B$  is *minimal* if  $f$  is not a projection and of minimal arity such that every  $g$  generated by  $f$  is either a projection or generates  $f$ .

The following is straightforward from the definitions.

**Proposition 5.7.** *Every minimal  $f$  generates a minimal clone, and every minimal clone is generated by a minimal operation.*

**Theorem 5.8.** *Every non-trivial operation clone  $\mathcal{C} \subseteq \mathcal{O}_B$  over a finite set  $B$  contains a minimal operation.*

*Proof.* Consider the set of all clones contained in  $\mathcal{C}$ , partially ordered by inclusion. From this poset we remove the trivial clone; the resulting poset will be denoted by  $P$ . We use Zorn's lemma to show that  $P$  contains a minimal element. Observe that in  $P$ , all chains  $(\mathcal{C}_i)_{i \in \kappa}$  that are *descending*, i.e.,  $\mathcal{C}_i \supseteq \mathcal{C}_j$  for  $i < j$ , are *bounded*, i.e., for all such chains there exists a  $\mathcal{D} \in P$  such that  $\mathcal{C}_i \supseteq \mathcal{D}$  for all  $i \in \kappa$ . To see this, observe that the set  $\bigcup_{i \in \kappa} \text{Inv}(\mathcal{C}_i)$  is *closed under*

*primitive positive definability* in the sense that it is the set of relations that is primitively positively definable over some relational structure  $\mathfrak{B}$  (since only a finite number of relations can be mentioned in a formula, and since  $\text{Inv}(\mathcal{C}_i)$  is closed under primitive positive definability, for each  $i \in \kappa$ ). Moreover, one of the relations  $P_B^4, \{b_1\}, \dots, \{b_n\}$ , for  $B = \{b_1, \dots, b_n\}$ , is not contained in  $\bigcup_{i \in \kappa} \text{Inv}(\mathcal{C}_i)$ ; otherwise, there would be a  $j \in \kappa$  such that  $\text{Inv}(\mathcal{C}_j)$  contains all these relations, and hence  $\mathcal{C}_j$  is the trivial clone contrary to our assumptions. Hence,  $\text{Pol}(\mathfrak{B})$  is a non-trivial lower bound of the descending chain  $(\mathcal{C}_i)_{i \in \kappa}$ . By Zorn's lemma,  $P$  contains a minimal element, and this element contains a minimal operation in  $\mathcal{C}$ .  $\square$

**Remark 5.9.** Note that the statement above would be false if  $B$  is infinite: take for example the clone over the domain  $B := \mathbb{N}$  of the integers generated by the operation  $x \mapsto x + 1$ . Every operation in this clone is essentially unary, and every unary operation in this clone is of the form  $x \mapsto x + c$  for  $c \in \mathbb{N}$ . Note that for  $c > 0$ , the operation  $x \mapsto x + c$  generates  $x \mapsto x + 2c$ , but not vice versa, so the clone does not contain a minimal operation.

In the remainder of this section, we show that a minimal operation has one out of the following five types, due to Rosenberg [65]. An  $n$ -ary operation  $f$  is called a *semiprojection* if there exists an  $i \leq n$  such that  $f(x_1, \dots, x_n) = x_i$  whenever  $|\{x_1, \dots, x_n\}| < n$ . For the purpose of proving the next lemma, we call an  $n$ -ary operation  $f$  a *weak semiprojection* if for all distinct  $i, j \in \{1, \dots, n\}$  there exists an index  $s(i, j)$  and a unary non-constant operation  $g_{i,j}$  such that  $\forall x_1, \dots, x_n: f(x_1, \dots, x_n) = g_{i,j}(x_{s(i,j)})$  holds whenever  $x_i$  and  $x_j$  are the same variable. In the proof of the following lemma the following notation for weak semiprojections will be practical. Let  $f$  be a weak semiprojection, let  $S \subseteq \{1, \dots, n\}$  be of cardinality at least two, and let  $(x_1, \dots, x_n)$  be a tuple of variables such that  $x_i = x_j$  for all  $i, j \in S$ . Then for some  $k \in \{1, \dots, n\}$  it holds that  $\forall x_1, \dots, x_n: f(x_1, \dots, x_n) = g_{i,j}(x_k)$ . If  $k \in S$  define  $E(S) := S$ . Otherwise, define  $E(S) := \{k\}$ . Note that if  $S \subseteq T \subseteq \{1, \dots, n\}$ , then  $E(S) \subseteq E(T)$ . Also note that if there exists a  $k \in \{1, \dots, n\}$  such that  $k \in E(S)$  for every  $S \subseteq \{1, \dots, n\}$  with at least two elements, then  $f$  is a quasi semiprojection.

**Lemma 5.10.** *Let  $f$  be a weak semiprojection of arity at least  $n \geq 4$ . Then  $f$  is a quasi semiprojection.*

*Proof.* We first show that  $E(\{1, 2\}) \cap E(\{3, 4\}) \neq \emptyset$ . If  $E(\{1, 2, 3, 4\}) = \{\ell\}$  for some  $\ell \notin \{1, 2, 3, 4\}$ , then  $E(\{1, 2\}) = \{\ell\} = E(\{3, 4\})$  and we are done. So we assume that  $E(\{1, 2, 3, 4\}) = \{1, 2, 3, 4\}$ . First consider the case that  $E(\{1, 2\}) = \{i\} \subseteq \{3, 4\}$ . If  $E(\{3, 4\}) = \{j\} \subseteq \{1, 2\}$  then  $f(x, x, y, y, x_5, \dots, x_n) = g_{1,2}(x_i)$  and  $f(x, x, y, y, x_5, \dots, x_n) = g_{3,4}(x_j)$  for  $i \neq j$ , which is a contradiction since  $g_{1,2}$  and  $g_{3,4}$  are non-constant. Hence,  $E(\{3, 4\}) = \{3, 4\}$  and we have found  $i \in E(\{1, 2\}) \cap E(\{3, 4\})$ . Similarly we can treat the case that  $E(\{3, 4\}) = \{i\} \subseteq \{1, 2\}$ . If  $E(\{1, 2\}) = \{1, 2\}$  and  $E(\{3, 4\}) = \{3, 4\}$  then  $f(x, x, y, y, x_5, \dots, x_n) = g_{1,2}(x)$  because of  $E(\{1, 2\}) \subseteq \{1, 2\}$  and  $f(x, x, y, y, x_5, \dots, x_n) = g_{3,4}(y)$  because of  $E(\{3, 4\}) \subseteq \{3, 4\}$ , which is a contradiction since  $g_{1,2}$  and  $g_{3,4}$  are non-constant.

Let  $i \in E(\{1, 2\}) \cap E(\{3, 4\})$ . Note that if  $i \notin \{1, 2\}$ , then  $E(\{1, 2\}) = \{i\}$ . Similarly, if  $i \notin \{3, 4\}$  then  $E(\{3, 4\}) = \{i\}$ . We therefore have a set  $S \subseteq \{1, \dots, n\}$  of size two with  $E(S) = \{i\}$ . Let  $T \subseteq \{1, \dots, n\}$  be of cardinality at least two. We will show that  $i \in E(T)$ . Observe that if  $T \subseteq \{1, \dots, n\} \setminus \{i\}$ , then  $E(T) = E(\{1, \dots, n\} \setminus \{i\}) = E(S) = \{i\}$ . Now suppose that  $T = \{i, j\}$  for some  $j \in \{1, \dots, n\} \setminus \{i\}$ . Then  $\{1, \dots, n\} \setminus T$  has at least two elements (since  $n \geq 4$ ). We can therefore apply the argument from the first paragraph, up

to renaming argument, to conclude that  $E(\{i, j\}) \cap E(\{1, \dots, n\} \setminus \{i, j\})$  contains an element  $k$ . If  $k \notin \{i, j\}$ , then  $E(\{1, \dots, n\} \setminus \{i, j\}) = \{1, \dots, n\} \setminus \{i, j\}$ , which is in contradiction to  $E(\{1, \dots, n\} \setminus \{i\}) = \{i\}$ . Hence,  $E(\{i, j\}) = \{i, j\}$ . This implies that  $E(T) = T$  for all  $T \subseteq \{1, \dots, n\}$  of cardinality at least 2 containing  $i$ . We conclude that  $i \in E(T)$  for every  $T \subseteq \{1, \dots, n\}$  with at least two elements, so  $f$  is a semiprojection.  $\square$

**Theorem 5.11** (Rosenberg's five types theorem). *Let  $f$  be a minimal operation. Then  $f$  has one of the following types:*

1. a unary operation, which is either a permutation such that  $f^p(x) = x$ , for some prime  $p$ , or satisfies  $f(f(x)) = f(x)$  for all  $x$ ;
2. a binary idempotent operation;
3. a majority operation;
4. a minority operation;
5. a  $k$ -ary semiprojection, for  $k \geq 3$ , which is not a projection.

*Proof.* The statement is easy to prove if  $f$  is unary. If  $f$  is at least binary, then  $\hat{f}$  (see Exercise 37) must be the identity by the minimality of  $f$ , and hence  $f$  is idempotent. In particular, we are done if  $f$  is binary. If  $f$  is ternary, we have to show that  $f$  is majority, Maltsev, or a semiprojection. By minimality of  $f$ , the binary operation  $f_1(x, y) := f(y, x, x)$  is a projection, that is,  $f_1(x, y) = x$  or  $f_1(x, y) = y$ . Note that in particular  $f(x, x, x) = x$ . Similarly, the other operations  $f_2(x, y) := f(x, y, x)$ , and  $f_3(x, y) := f(x, x, y)$  obtained by identifications of two variables must be projections. We therefore distinguish eight cases.

1.  $f(y, x, x) = x, f(x, y, x) = x, f(x, x, y) = x$ .  
In this case,  $f$  is a majority.
2.  $f(y, x, x) = x, f(x, y, x) = x, f(x, x, y) = y$ .  
In this case,  $f$  is a semiprojection.
3.  $f(y, x, x) = x, f(x, y, x) = y, f(x, x, y) = x$ .  
In this case,  $f$  is a semiprojection.
4.  $f(y, x, x) = x, f(x, y, x) = y, f(x, x, y) = y$ .  
The operation  $g(x, y, z) := f(y, x, z)$  is a Maltsev operation.
5.  $f(y, x, x) = y, f(x, y, x) = x, f(x, x, y) = x$ .  
In this case,  $f$  is a semiprojection.
6.  $f(y, x, x) = y, f(x, y, x) = x, f(x, x, y) = y$ .  
In this case,  $f$  is a Maltsev operation.
7.  $f(y, x, x) = y, f(x, y, x) = y, f(x, x, y) = x$ .  
The operation  $g(x, y, z) := f(x, z, y)$  is a Maltsev operation.
8.  $f(y, x, x) = y, f(x, y, x) = y, f(x, x, y) = y$ .  
In this case,  $f$  is a Maltsev operation.

We claim that if  $f$  is a Maltsev operation, then either it is a minority operation (and we are done) or it generates a Majority operation. Indeed, if  $f$  is not a minority then minimality of  $f$  implies that  $f(x, y, x) = x$ . Now consider the function  $g$  defined by  $g(x, y, z) = f(x, f(x, y, z), z)$ . We have

$$\begin{aligned} g(x, x, y) &= f(x, f(x, x, y), y) = f(x, y, y) = x \\ g(x, y, x) &= f(x, f(x, y, x), x) = f(x, x, x) = x \\ g(y, x, x) &= f(y, f(y, x, x), x) = f(y, y, x) = x. \end{aligned}$$

Note that every ternary function generated by a majority is again a majority. Also note that a function cannot be a majority and a minority at the same time unless the domain has only one element, so we obtain in this case a contradiction to the minimality of  $f$ .

Finally, let  $f$  be  $k$ -ary, where  $k \geq 4$ . By minimality of  $f$ , the operations obtained from  $f$  by identifications of arguments of  $g$  must be projections. The lemma of Świerczkowski implies that  $f$  is a semiprojection.  $\square$

**Proposition 5.12.** *For all  $n \geq 3$ , the graph  $K_n$  is projective (i.e., all idempotent polymorphisms of  $K_n$  are projections). All relations that are preserved by  $\text{Sym}(\{0, \dots, n-1\})$  are primitive positive definable in  $K_n$ .*

This provides for example a solution to Exercise 75.

*Proof.* By Theorem 5.8, it suffices to show that the clone of idempotent polymorphisms of  $K_n$  does not contain a minimal operation. Hence, by Theorem 5.11, we have to verify that  $\text{Pol}(K_n)$  does not contain a binary idempotent, a Maltsev, a majority, or a  $k$ -ary semiprojection for  $k \geq 3$ .

1. Let  $f$  be a binary idempotent polymorphism of  $K_n$ .

**Observation 1.**  $f(u, v) \in \{u, v\}$ : otherwise,  $i := f(u, v)$  is adjacent to both  $u$  and  $v$ , but  $f(i, i) = i$  is not adjacent to  $i$ , in contradiction to  $f$  being a polymorphism.

**Observation 2.** If  $f(u, v) = u$ , then  $f(v, u) = v$ : this is clear if  $u = v$ , and if  $u \neq v$  it follows from  $f$  being a polymorphism.

By Observation 1, it suffices to show that there cannot be distinct  $u, v$  and distinct  $u', v'$  such that  $f(u, v) = u$  and  $f(u', v') = v'$ . Suppose for contradiction that there are such  $u, v, u', v'$ .

**Case 1.**  $u = u'$ . Since  $f(u, v') = f(u', v') = v'$ , we have  $f(v', u) = u$  by Observation 2. This is in contradiction to  $f(u, v) = u$  since  $u = u'$  is adjacent to  $v'$ , and  $E(v, u)$ .

**Case 2.**  $u \neq u'$ .

**Case 2.1.**  $f(u', u) = u$ : this is impossible because  $f(u, v) = u$ ,  $E(u, u')$ , and  $E(u, v)$ .

**Case 2.2.**  $f(u', u) = u'$ : this is impossible because  $f(v', u') = u'$ ,  $E(u', v')$ , and  $E(u', u)$ .

2. Since  $(1, 0), (1, 2), (0, 2) \in E(K_n)$ , but  $(0, 0) \notin E(K_n)$ , the graph  $K_n$  has no Maltsev polymorphism (it is not rectangular; see Section 3.3).
3. If  $f$  is a majority, note that  $f(0, 1, 2) = f(x_0, x_1, x_2)$  where  $x_i$  is some element distinct from  $i$  if  $f(0, 1, 2) = i$ , and  $x_i := f(0, 1, 2)$  otherwise. But  $(i, x_i) \in E(K_n)$ , so  $f$  is not a polymorphism of  $K_n$ .



4. Finally, let  $f$  be a  $k$ -ary semiprojection for  $k \geq 3$  which is not a projection. Suppose without loss of generality that  $f(x_1, \dots, x_k) = x_1$  whenever  $|\{x_1, \dots, x_k\}| < k$  (otherwise, permute the arguments of  $f$ ). Since  $f$  is not a projection, there exist pairwise distinct  $a_1, \dots, a_k \in V(K_n)$  such that  $c := f(a_1, \dots, a_k) \neq a_1$ . Let  $b_1, \dots, b_k$  be such that  $b_i$  is any element of  $V(K_n) \setminus \{c\}$  if  $c = a_i$ , and  $b_i := c$  otherwise. Note that  $b_1 = a_1$  since  $c \neq a_1$ , and that  $f(b_1, \dots, b_k) = b_1 = a_1$  because  $f$  is a semiprojection. But  $(a_i, b_i) \in E(K_n)$  for all  $i \leq k$ , so  $f$  is not a polymorphism of  $K_n$ .

The second part of the statement follows from Theorem 5.2.  $\square$

## 5.5 Schaefer's Theorem

Schaefer's theorem states that every CSP for a 2-element structure is either in P or NP-hard. By the general results in Section 5.2, most of the classification arguments in Schaefer's article follow from earlier work of Post [64], who classified all clones on a two-element domain. We present a short proof of Schaefer's theorem here.

Note that on Boolean domains, there is precisely one minority operation, and precisely one majority operation.

**Theorem 5.13** (Post [64]). *Every minimal operation on  $\{0, 1\}$  is among one of the following:*

- *a unary constant function.*
- *the unary function  $x \mapsto 1 - x$ .*
- *the binary function  $(x, y) \mapsto \min(x, y)$ .*
- *the binary function  $(x, y) \mapsto \max(x, y)$ .*
- *the Boolean minority operation.*
- *the Boolean majority operation.*

*Proof.* If  $f$  is unary the statement is trivial, so let  $f$  be a minimal at least binary idempotent function above  $\mathcal{C}$ . There are only four binary idempotent operations on  $\{0, 1\}$ , two of which are projections and therefore cannot be minimal. The other two operations are  $\min$  and  $\max$ . Next, note that a semiprojection of arity at least three on a Boolean domain must be a projection. Thus, Theorem 5.11 implies that  $f$  is the majority or a minority operation.  $\square$

A Boolean relation  $R \subseteq \{0, 1\}^n$  is called *affine* if it is the solution space of a system of linear equalities modulo 2.

**Lemma 5.14.** *A Boolean relation is affine if and only if it is preserved by the Boolean minority operation.*

*Proof.* This statement follows from basic facts in linear algebra. Let  $R$  be  $n$ -ary. We view  $R$  as a subset of the Boolean vector space  $\{0, 1\}^n$ . It is well-known that affine spaces are precisely those that are closed under *affine combinations*, i.e., linear combinations of the form  $\alpha_1 x_1 + \dots + \alpha_k x_k$  such that  $\alpha_1 + \dots + \alpha_k = 1$ . In particular, if  $R$  is affine then it is preserved by  $(x_1, x_2, x_3) \mapsto x_1 + x_2 + x_3$  which is the minority operation. Conversely, if  $R$  is preserved by the minority operation, then  $x_1 + \dots + x_k$ , for odd  $k$ , can be written as

$$\text{minority}(x_1, x_2, \text{minority}(x_3, x_4, \dots \text{minority}(x_{n-2}, x_{k-1}, x_k) \dots))$$

and hence  $R$  is preserved by all affine combinations, and thus affine.  $\square$

It is well-known and easy to see (see, for example, [15]) that for every relation  $R \subseteq \{0, 1\}^n$  there exists a propositional formula  $\phi(x_1, \dots, x_n)$  that defines  $R$ , and that  $\phi$  can even be chosen to be in *conjunctive normal form (CNF)*. That is, there is a conjunction of disjunctions of variables or negated variables from  $x_1, \dots, x_n$  such that a tuple  $(t_1, \dots, t_n) \in \{0, 1\}^n$  is in  $R$  if and only if the formula  $\phi$  evaluates to true after replacing  $x_i$  by  $t_i$ , for  $i \in \{1, \dots, n\}$ . The following definition is useful for proving that certain Boolean relations  $R$  can be defined in syntactically restricted propositional logic.

**Definition 5.15.** When  $\phi$  is a propositional formula in CNF that defines a Boolean relation  $R$ , we say that  $\phi$  is *reduced* if the following holds: whenever we remove a literal from a clause in  $\phi$ , then the resulting formula no longer defines  $R$ .

Clearly, every Boolean relation has a reduced definition: simply remove literals from any definition in CNF until the formula becomes reduced. A propositional formula in CNF is called *Horn* if every clause contains at most one positive literal.

**Lemma 5.16.** *A Boolean relation  $R$  has a Horn definition if and only if  $R$  is preserved by min.*

*Proof.* It is easy to see that min preserves every relation defined by clauses that contains at most one positive literal, and hence every relation with a Horn definition. Conversely, let  $R$  be a Boolean relation preserved by min. Let  $\phi$  be a reduced propositional formula in CNF that defines  $R$ . Now suppose for contradiction that  $\phi$  contains a clause  $C$  with two positive literals  $u$  and  $v$ . Since  $\phi$  is reduced, there is an assignment  $s_1$  that satisfies  $\phi$  such that  $s_1(u) = 1$ , and such that all other literals of  $C$  evaluate to 0. Similarly, there is a satisfying assignment  $s_2$  for  $\phi$  such that  $s_2(v) = 1$  and all other literals of  $C$  evaluate to 0. Then  $s_0: x \mapsto \min(s_1(x), s_2(x))$  does not satisfy  $C$ , and does not satisfy  $\phi$ , in contradiction to the assumption that min preserves  $R$ .  $\square$

A binary relation is called *bijunctive* if it can be defined by a propositional formula in CNF where each disjunction has at most two disjuncts.

**Lemma 5.17.** *A Boolean relation  $R$  is bijunctive if and only if it is preserved by the Boolean majority operation.*

*Proof.* It is easy to see that the majority operation preserves every Boolean relation of arity two, and hence every bijunctive Boolean relation. We present the proof that if  $R$  is preserved by majority, and  $\phi$  is a reduced definition of  $R$ , then all clauses  $C$  have at most two literals. Suppose for contradiction that  $C$  has three literals  $l_1, l_2, l_3$ . Since  $\phi$  is reduced, there must be satisfying assignments  $s_1, s_2, s_3$  to  $\phi$  such that under  $s_i$  all literals of  $C$  evaluate to 0 except for  $l_i$ . Then the mapping  $s_0: x \mapsto \text{majority}(s_1(x), s_2(x), s_3(x))$  does not satisfy  $C$  and therefore does not satisfy  $\phi$ , in contradiction to the assumption that majority preserves  $R$ .  $\square$

The following relation is called the (*Boolean*) *not-all-equal relation*.

$$\text{NAE} := \{(0, 0, 1), (0, 1, 0), (1, 0, 0), (1, 1, 0), (1, 0, 1), (0, 1, 1)\}$$

**Theorem 5.18** (Schaefer [66]). *Let  $\mathfrak{B}$  be a structure over the two-element universe  $\{0, 1\}$ . Then either  $(\{0, 1\}; \text{NAE})$  has a primitive positive definition in  $\mathfrak{B}$ , and  $\text{CSP}(\mathfrak{B})$  is NP-complete, or*

1.  $\mathfrak{B}$  is preserved by a constant operation.
2.  $\mathfrak{B}$  is preserved by min. Equivalently, every relation of  $\mathfrak{B}$  has a definition by a propositional Horn formula.
3.  $\mathfrak{B}$  is preserved by max. Equivalently, every relation of  $\mathfrak{B}$  has a definition by a dual-Horn formula, that is, by a propositional formula in CNF where every clause contains at most one negative literal.
4.  $\mathfrak{B}$  is preserved by the majority operation. Equivalently, every relation of  $\mathfrak{B}$  is bijunctive.
5.  $\mathfrak{B}$  is preserved by the minority operation. Equivalently, every relation of  $\mathfrak{B}$  can be defined by a conjunction of linear equations modulo 2.

In case (1) to case (5), then for every finite-signature reduct  $\mathfrak{B}'$  of  $\mathfrak{B}$  the problem  $\text{CSP}(\mathfrak{B}')$  can be solved in polynomial time.

*Proof.* If  $\text{Pol}(\mathfrak{B})$  contains a constant operation, then we are in case one; so suppose in the following that this is not the case. If NAE is primitive positive definable in  $\mathfrak{B}$ , then  $\text{CSP}(\mathfrak{B})$  is NP-hard by reduction from positive not-all-equal-3SAT [42]. Otherwise, by Theorem 5.2 there is an operation  $f \in \text{Pol}(\mathfrak{B})$  that violates NAE. If  $\hat{f}$  defined as  $x \mapsto f(x, \dots, x)$  equals the identity then  $f$  is idempotent. Otherwise,  $\hat{f}$  equals  $\neg$ . But then  $\neg f \in \text{Pol}(\mathfrak{B})$  is idempotent and also violates NAE. So let us assume in the following that  $f$  is idempotent. Then  $f$  generates an at least binary minimal operation  $g \in \text{Pol}(\mathfrak{B})$ .

By Theorem 5.13, the operation  $g$  equals min, max, the Boolean minority, or the Boolean majority function.

- $g = \min$  or  $g = \max$ . By Lemma 5.16, the relations of  $\mathfrak{B}$  are preserved by min if and only if they can be defined by propositional Horn formulas. It is well-known that positive unit-resolution is a polynomial-time decision procedure for the satisfiability problem of propositional Horn-clauses [67]. The case that  $g = \max$  is dual to this case.
- $g = \text{majority}$ . By Lemma 5.17, the relations of  $\mathfrak{B}$  are preserved by majority if and only if they are bijunctive. Hence, in this case the instances of  $\text{CSP}(\mathfrak{B})$  can be viewed as instances of the 2SAT problem, and can be solved in linear time [3].
- $g = \text{minority}$ . By Lemma 5.14 every relation of  $\mathfrak{B}$  has a definition by a conjunction of linear equalities modulo 2. Then  $\text{CSP}(\mathfrak{B})$  can be solved in polynomial time by Gaussian elimination.

This concludes the proof of the statement. □

### Exercises.

83. The Rosenberg theorem is only a *preclassification* in the sense that not every operation which has one of the five types is minimal. For each of the following five questions, either present a proof or give a counterexample.
  - (a) Is every unary operation which is a permutation such that  $f^p(x) = x$  for some prime  $p$ , or satisfies  $f(f(x)) = f(x)$ , minimal?
  - (b) Is every binary idempotent operation minimal?

- (c) Is every majority operation minimal?
- (d) Is every minority operation minimal?
- (e) Is every  $k$ -ary semiprojection, for  $k \geq 3$ , which is not a projection, minimal?

84. Determine the complexity of the following CSPs.

$$\begin{aligned} &\text{CSP}(\{0, 1\}; \{(0, 0, 1, 1), (1, 1, 0, 0)\}) \\ &\text{CSP}(\{0, 1\}; \{(0, 0, 1), (0, 1, 0), (1, 0, 0), (1, 1, 1)\}, \{(0, 1), (1, 0)\}) \\ &\text{CSP}(\{0, 1\}; \{0, 1\}^3 \setminus \{(1, 1, 0)\}, \{(0, 1), (1, 0)\}). \end{aligned}$$

85. Show that a Boolean relation  $R \subseteq \{0, 1\}^k$  can be defined by a propositional Horn formula if and only if it is primitively positively definable in  $(\{0, 1\}; \{0, 1\}^3 \setminus \{(1, 1, 0)\}, \{0\}, \{1\})$ .

## 6 Maltsev Polymorphisms

Recall from Section 3.3 the definition of a Maltsev operation: a ternary operation  $f: D^3 \rightarrow D$  satisfying

$$\forall x, y \in D. f(y, x, x) = f(x, x, y) = y.$$

As we have seen in Theorem 3.18, every digraph with a Maltsev polymorphism can be solved by the path-consistency procedure. However, when considering arbitrary relational structures then there are many examples with a Maltsev polymorphism that cannot be solved by the path-consistency procedure [4]. In this section, we present the algorithm of Bulatov and Dalmau for  $\text{CSP}(\mathfrak{A})$  when  $\mathfrak{A}$  is preserved by a Maltsev polymorphism [25].

**Theorem 6.1.** *Let  $\mathfrak{A}$  be a finite structure with finite relational signature and a Maltsev polymorphism. Then  $\text{CSP}(\mathfrak{A})$  can be solved in polynomial time.*

### 6.1 Examples

The most prominent class of structures  $\mathfrak{A}$  with a Maltsev polymorphism comes from groups. For any group  $G$ , the operation  $m$  given by  $(x, y, z) \mapsto x + y^{-1} + z$  is obviously Maltsev and called an *affine Maltsev operation*. Note that if the group  $G$  is  $F = (\mathbb{Z}_p; +, 0)$  then the  $k$ -ary relations preserved by  $m$  are precisely the affine subspaces of  $F^k$  (with the same argument as given for Lemma 5.14 in the Boolean case). In this case one can use Gaussian elimination to solve  $\text{CSP}(\mathfrak{A})$ . If  $G$  is abelian, then  $\mathfrak{A}$  is called *affine Maltsev*.

For general finite groups  $G$ , and if all relations of  $\mathfrak{A}$  are cosets  $gH := \{gh \mid h \in H\}$  of subgroups  $H$  of  $G^k$ , then Feder and Vardi [40] showed how to solve  $\text{CSP}(\mathfrak{A})$  in polynomial time using a previously known algorithm to find small generating sets for a permutation group. We will not discuss this approach, but rather present the more general algorithm of Bulatov and Dalmau which works for all finite structures preserved by a Maltsev polymorphism. First we have a look at two examples of Maltsev polymorphisms that do not come from groups in the way described above.

**Example 6.2.** On the domain  $\{0, 1, 2\}$ , let  $m$  be a minority, and define  $m(x, y, z) = 2$  whenever  $|\{x, y, z\}| = 3$ . Note that  $m$  preserves the equivalence relation with the equivalence classes  $\{2\}$  and  $\{0, 1\}$ . Also note that  $m$  preserves all relations of the form (for  $i \in \{0, 1\}$ )

$$\{(2, \dots, 2)\} \cup \{(x_1, \dots, x_n) \in \{0, 1\}^n \mid x_1 + x_2 + \dots + x_n = i \pmod{2}\}. \quad \triangle$$

**Example 6.3.** On the domain  $\{0, 1, 2\}$ , let  $m$  be a minority, and define  $m(x, y, z) = x$  whenever  $|\{x, y, z\}| = 3$ . Then  $m$  preserves for every permutation  $\pi$  of  $\{0, 1, 2\}$  the relation  $\{(x, y) \mid \pi(x) = y\}$ . Moreover,  $m$  preserves the binary relations  $R$  that are maximal with the property that  $|\text{pr}_1 R|, |\text{pr}_2 R| \leq 2$ .  $\triangle$

### Exercises.

86. Check the claims made in Example 6.2. Are all relations that are preserved by  $m$  primitive positive definable over the given relations?
87. Check the claims made in Example 6.3. Are all relations that are preserved by  $m$  primitive positive definable over the given relations?
88. Show that if a Maltsev operation  $m$  preserves the digraph  $\{(x, y, z, u) \mid m'(x, y, z) = u\}$  of a Maltsev operation  $m'$  on the same domain, then  $m = m'$ .

## 6.2 Compact Representations of Relations

Our presentation of the proof closely follows that of Bulatov and Dalmau [25].

**Definition 6.4** (Forks and Representations). Let  $R \subseteq A^n$  be a relation.

- A *fork* of  $R$  is a triple  $(i, a, b)$  such that there exist  $s, t \in R$  with  $(s_1, \dots, s_{i-1}) = (t_1, \dots, t_{i-1})$ ,  $s_i = a$ , and  $t_i = b$ . We say that  $s$  and  $t$  *witness*  $(i, a, b)$ .
- $R' \subseteq R$  is called a *representation* of  $R$  if every fork of  $R$  is also a fork of  $R'$ .
- A representation  $R'$  of  $R$  is called *compact* if its cardinality is at most twice the number of forks of  $R$ .

Clearly, every relation has a compact representation.

**Definition 6.5.** For  $i_1, \dots, i_k \in \{1, \dots, n\}$ , we write  $\text{pr}_{i_1, \dots, i_k}$  for the function from  $A^n \rightarrow A^k$  defined by  $\text{pr}_{i_1, \dots, i_k}(t) := (t_{i_1}, \dots, t_{i_k})$ .

If  $R \subseteq A^n$ , we write  $\text{pr}_{i_1, \dots, i_k}(R)$  for  $\{\text{pr}_{i_1, \dots, i_k}(t) \mid t \in R\}$ . For an operation  $m: A^k \rightarrow A$  and a relation  $R$  on  $A$ , we write  $\langle R \rangle_m$  for the smallest relation that contains  $R$  and is preserved by  $m$ .

**Lemma 6.6.** Let  $A$  be a finite set and let  $m: A^3 \rightarrow A$  be a Maltsev operation. Let  $R \subseteq A^k$  be a relation preserved by  $m$ , and let  $R'$  be a representation of  $R$ . Then  $R = \langle R' \rangle_m$ .

*Proof.* We show by induction on  $i \in \{1, \dots, n\}$  that  $\text{pr}_{1, \dots, i} \langle R' \rangle_m = \text{pr}_{1, \dots, i}(R)$ . Clearly,  $\text{pr}_{1, \dots, i} \langle R' \rangle_m \subseteq \text{pr}_{1, \dots, i}(R)$ , so we only need to show the converse inclusion. The case  $i = 1$  follows from that fact that  $R$  has for every  $t \in R$  the fork  $(1, t_1, t_1)$ , and since  $R'$  must also have this fork it must contain a tuple  $t'$  such that  $t'_1 = t_1$ .

So let us assume that the statement holds for  $i < n$ . We have to show that for every  $t \in R$  we have  $(t_1, \dots, t_{i+1}) \in \text{pr}_{1, \dots, i+1} \langle R' \rangle_m$ . By induction hypothesis there exists a tuple  $s \in \langle R' \rangle_m$  such that  $(s_1, \dots, s_i) = (t_1, \dots, t_i)$ . Then  $(i+1, s_{i+1}, t_{i+1})$  is a fork of  $R$ , so there exist tuples  $s', s'' \in R'$  witnessing it. Then the tuple  $t' := m(s, s', s'') \in \langle R' \rangle_m$  is such that

$$\begin{aligned} (t'_1, \dots, t'_i, t'_{i+1}) &= (m(t_1, s'_1, s''_1), \dots, m(t_i, s'_i, s''_i), m(s_{i+1}, s_{i+1}, t_{i+1})) \\ &= (t_1, \dots, t_i, t_{i+1}) \end{aligned} \quad (\text{since } s'_i = s''_i).$$

Hence,  $(t_1, \dots, t_i, t_{i+1})$  is a tuple from  $\text{pr}_{1, \dots, i+1} \langle R' \rangle_m$ , as required.  $\square$

<pre> Procedure <i>Nonempty</i>(<math>R', i_1, \dots, i_k, S</math>).  Set <math>U := R'</math>. While <math>\exists r, s, t \in U</math> such that <math>\text{pr}_{i_1, \dots, i_k} m(r, s, t) \notin \text{pr}_{i_1, \dots, i_k} U</math>:     Set <math>U := U \cup \{m(r, s, t)\}</math> If <math>\exists t \in U</math> such that <math>(t_{i_1}, \dots, t_{i_k}) \in S</math> then return <math>t</math> else return 'No'. </pre>
--

Figure 7: The procedure *Nonempty*.

### Exercises.

89. Let  $A$  be a finite set. How many forks does the  $n$ -ary relation  $R := A^n$  have? Explicitly construct a compact representation for  $R$ .
90. Let  $R$  be the relation  $\{(x, y, z, u) \in \{0, 1\}^4 \mid x + y + z = 1 \pmod{2}\}$ . Find a smallest possible representation  $R'$  for  $R$ . Explicitly compute  $\langle R' \rangle_m$  where  $m$  is the Boolean minority.

### 6.3 The Bulatov-Dalmau Algorithm

Let  $\exists x_1, \dots, x_n (\phi_1 \wedge \dots \wedge \phi_n)$  be an instance of  $\text{CSP}(\mathfrak{A})$ . For  $\ell \leq n$ , we write  $R_\ell$  for the relation

$$\{(s_1, \dots, s_n) \in A^n \mid \mathfrak{A} \models (\phi_1 \wedge \dots \wedge \phi_\ell)(s_1, \dots, s_n)\}.$$

The idea of the algorithm is to inductively construct a compact representation  $R'_\ell$  of  $R_\ell$ , adding constraints one by one. Initially, for  $\ell = 0$ , we have  $R_\ell = A^n$ , and it is easy to come up with a compact representation for this relation. Note that when we manage to compute the compact representation  $R'_n$  for  $R_n$ , we can decide satisfiability of the instance: it is unsatisfiable if and only if  $R'_n$  is empty. For the inductive step, we need a procedure called *Next* which is more involved; we first introduce two auxiliary procedures.

#### The procedure *Nonempty*

The procedure *Nonempty* receives as input

- a compact representation  $R'$  of a relation  $R$ ,
- a sequence  $i_1, \dots, i_k$  of elements in  $[n]$  where  $n$  is the arity of  $R$ , and
- a  $k$ -ary relation  $S$  which is also preserved by  $m$ .

The output of the procedure is either a tuple  $t \in R$  such that  $(t_{i_1}, \dots, t_{i_k}) \in S$ , or 'No' if no such tuple exists. The procedure can be found in Figure 7. For its correctness we note the following:

- $R' \subseteq U \subseteq R$ : initially we start from  $U := R' \subseteq R$ , and only add tuples to  $U$  obtained by applying  $m$  to tuples in  $U$ , so the added tuples are again in  $R$ .

- It follows that if *Nonempty* returns a tuple  $(t_{i_1}, \dots, t_{i_k})$ , then this tuple is indeed from  $\text{pr}_{i_1, \dots, i_k} R$  and the output of the algorithm is correct.
- When the algorithm exits the while loop then  $\text{pr}_{i_1, \dots, i_k} \langle U \rangle_m = \text{pr}_{i_1, \dots, i_k} U$ . Since  $R' \subseteq U$  we have that  $\langle U \rangle_m = R$ . Hence, every tuple  $t \in \text{pr}_{i_1, \dots, i_k} R = \text{pr}_{i_1, \dots, i_k} \langle U \rangle_m$  is contained in  $\text{pr}_{i_1, \dots, i_k} U$ , and so the answer of the algorithm is also correct when it returns ‘No’.

We mention that this procedure does not use the particular properties of a Maltsev polymorphism, but works for any explicitly given polymorphism.

**Running time.** The number of iterations of the while loop can be bounded by the size  $|U|$  of the set  $U$  at the end of the execution of the procedure. Hence, when we want to use this procedure to obtain a polynomial-time running time, we have to make sure that the size of  $U$  remains polynomial in the input size. The way this is done in the Bulatov-Dalmau algorithm is to guarantee that at each call of *Nonempty* the size  $L$  of  $\text{pr}_{i_1, \dots, i_k} R$  is polynomial in the input size. Then  $|U|$  is bounded by  $L + |R'|$  which is also polynomial.

We have to test all tuples  $r, s, t \in U$ ; this can be implemented so that  $|U|^3$  steps suffice. In each step we have to compute  $m(r, s, t)$  and test whether  $\text{pr}_{i_1, \dots, i_k} m(r, s, t) \in \text{pr}_{i_1, \dots, i_k} U$ , which can be done in  $O(kL)$ . In the important case that  $L$  is bounded by a constant in the size of the input  $N$ , the running time of *Nonempty* is in  $O(N^4)$ .

### The procedure *Fix-values*

The procedure *Fix-values* receives as input

- a compact representation  $R'$  of an  $n$ -ary relation  $R$  preserved by  $m$ , and
- a sequence  $c_1, \dots, c_k \in A$  for  $k \leq n$ .

The output of *Fix-values* is a compact representation of the relation

$$R \cap (\{c_1\} \times \dots \times \{c_k\} \times A \times \dots \times A).$$

The procedure can be found in Figure 8. The algorithm computes inductively a compact representation  $U_j$  of the relation

$$R_j = R \cap (\{c_1\} \times \dots \times \{c_j\} \times A \times \dots \times A)$$

This is immediate for  $U_0 = R'$ , and the set  $U_k$  is the relation that we have to compute.

For its correctness, suppose inductively that  $U_j$  is a compact representation of  $R_j$ . We have to show that the set  $U_{j+1}$  computed by the procedure is a compact representation of  $R_{j+1}$ :

1.  $U_{j+1} \subseteq R_{j+1}$ . Suppose that the procedure adds  $\{r, m(r, s, t)\}$  to  $U_{j+1}$ , where  $r$  and  $s$  witness the fork  $(i, a, b)$  of  $U_j$  processed in the for-loop of the procedure. Note that  $r \in R_{j+1}$  since  $r \in U_j \subseteq R_j$  and  $r_{j+1} = c_{j+1}$ . Since  $m$  preserves  $R$  and is idempotent, it also preserves  $R_j$ , and since  $r, s, t \in R_j$  it follows that  $m(r, s, t) \in R_j$ . To show that  $m(r, s, t) \in R_{j+1}$  it suffices to show that  $s_{j+1} = t_{j+1}$  because then  $m(r, s, t)_{j+1} = r_{j+1} = c_{j+1}$  since  $m$  is Maltsev. If  $i > j + 1$  then we have that  $s_{j+1} = t_{j+1}$  since  $s, t$  witness  $(i, a, b)$ . Otherwise, we must have  $a = b = c_i$  because of the innermost if-clause of the procedure. But then  $s = t$  by the stipulation of the algorithm on the choice of  $s$  and  $t$ .

<pre> Procedure <i>Fix-values</i>(<math>R', c_1, \dots, c_k</math>).  Set <math>j := 0</math>; <math>U_j := R'</math>. While <math>j &lt; k</math> do:   Set <math>U_{j+1} := \emptyset</math>.   For each <math>(i, a, b) \in [n] \times A^2</math>:     If <math>\exists s, t \in U_j</math> witnessing <math>(i, a, b)</math> (assuming <math>s = t</math> if <math>a = b</math>):       If <math>r := \text{Nonempty}(U_j, j + 1, i, \{(c_{j+1}, a)\}) \neq \text{'No'}</math>         If <math>(i &gt; j + 1)</math> or <math>(a = b = c_i)</math>:           Set <math>U_{j+1} := U_{j+1} \cup \{r, m(r, s, t)\}</math>       Set <math>j := j + 1</math>. Return <math>U_k</math>. </pre>
--

Figure 8: The procedure *Fix-values*.

2. All forks  $(i, a, b)$  of  $R_{j+1}$  are forks of  $U_{j+1}$ . If  $R_{j+1}$  has the fork  $(i, a, b)$ , then by inductive assumption  $U_j$  must contain witnesses  $s, t$  for  $(i, a, b)$ . Therefore, the first if-clause of the procedure is positive. Moreover,  $s_{j+1} = c_{j+1}$  and  $s_i = a$ , so  $r := \text{Nonempty}(U_j, j + 1, i, \{(c_{j+1}, a)\}) \neq \text{'No'}$ . Also note that if  $i \leq j + 1$ , then  $a = s_i = c_i = t_i = b$ . So all the if-clauses of the procedure are positive, and the procedure adds  $r$  and  $m(r, s, t)$  to  $U_{j+1}$ . The tuples  $r$  and  $m(r, s, t)$  witness  $(i, a, b)$ . Since  $s, t$  witness  $(i, a, b)$  we have that  $(s_1, \dots, s_{i-1}) = (t_1, \dots, t_{i-1})$ . Hence,  $\text{pr}_{1, \dots, i-1}(m(r, s, t)) = (r_1, \dots, r_{i-1})$ . Furthermore, we have that  $\text{pr}_i(m(r, s, t)) = m(a, a, b) = b$ .
3. The representation  $U_{j+1}$  of  $R_{j+1}$  is compact since at most two tuples are added to  $U_{j+1}$  for each fork of  $R_{j+1}$ .

**Running time.** The while loop is performed  $k \leq n$  times; the inner for-loop is executed for each  $(i, a, b) \in [n] \times A^2$ , which is linear for fixed  $\mathfrak{A}$ . The cost of each iteration is dominated by the cost of calling the procedure *Nonempty*. Note that when calling *Nonempty*, the size of  $\text{pr}_{j+1, i} U_j$  is polynomial in the input size (even constant size when  $\mathfrak{A}$  is fixed), so the cost of *Nonempty* is in  $O(N^4)$  where  $N$  is the size of the input. Therefore, the total time complexity of the procedure *Fix-values* is polynomial in the input size (for fixed  $\mathfrak{A}$  it is in  $O(N^5)$ ).

### The procedure *Next*

Now comes the heart of the algorithm, which is the procedure *Next* that updates a compact representation of the solution space when constraints are added one by one. The input of the procedure is

- a compact representation  $R'$  of a relation  $R \subseteq A^n$  that is preserved by  $m$ ,
- a sequence  $i_1, \dots, i_k$  of elements from  $[n]$ ,
- a  $k$ -ary relation  $S$  which is also preserved by  $m$ .

The output of the procedure is a compact representation of the relation

$$R^* := \{t \in R \mid (t_{i_1}, \dots, t_{i_k}) \in S\}.$$



```

Procedure Next( $R', i_1, \dots, i_k, S$ ).

Set  $U := \emptyset$ .
For each  $(i, a, b) \in [n] \times A^2$ :
  If  $\text{Nonempty}(R', i_1, \dots, i_k, i, S \times \{a\}) \neq \text{'No'}$ :
    If  $\text{Nonempty}(\text{Fix-values}(R', t_1, \dots, t_{i-1}), i_1, \dots, i_k, i, S \times \{b\}) \neq \text{'No'}$ :
      Set  $U := U \cup \{t, t'\}$ .
Return  $U_k$ .

```

Figure 9: The procedure *Next*.

The procedure *Next* can be found in Figure 9. Observe that

- the condition  $\text{Nonempty}(R', i_1, \dots, i_k, i, S \times \{a\}) \neq \text{'No'}$  from the first if-clause is satisfied if and only if there exists a tuple  $t \in R$  such that  $(t_{i_1}, \dots, t_{i_k}) \in S$  and  $t_i = a$ . Hence, if such a tuple does not exist, then  $(i, a, b)$  cannot be a fork of  $R^*$ , and nothing needs to be done.
- the condition  $\text{Nonempty}(\text{Fix-values}(R', t_1, \dots, t_{i-1}), i_1, \dots, i_k, i, S \times \{b\}) \neq \text{'No'}$  from the second if-clause is satisfied if and only if there exists a tuple  $t' \in R$  such that

- $(t'_1, \dots, t'_{i-1}) = (t_1, \dots, t_{i-1})$ ,
- $(t'_{i_1}, \dots, t'_{i_k}) \in S$ , and
- $t'_i = b$ .

If this condition holds, and since  $t_i = a$ , we have that  $t$  and  $t'$  witness  $(i, a, b)$ . It only remains to show that if  $(i, a, b)$  is a fork of  $R^*$ , then such a tuple  $t'$  must exist. So let  $r$  and  $s$  be witnesses for  $(i, a, b)$  in  $R^*$ . Then the tuple  $t' := m(t, r, s)$  has the desired properties:

- for  $j < i$  we have that  $t'_j = m(t_j, r_j, s_j) = t_j$ ;
- $t' \in S$  because  $(r_{i_1}, \dots, r_{i_k}), (s_{i_1}, \dots, s_{i_k}), (t_{i_1}, \dots, t_{i_k}) \in S$  and  $m$  preserves  $S$ .
- $t'_i = m(t_i, r_i, s_i) = m(a, a, b) = b$ .

- The cardinality of  $U$  is bounded by twice the number of forks of  $R^*$ , so the representation computed by the algorithm is compact.

**Running time.** The for-loop of the procedure *Next* is performed  $n|A|^2$  times and the cost of each iteration is polynomial in the cost of *Nonempty* and *Fix-values*. Also note that  $k$  is bounded by the maximal arity of the relations in  $\mathfrak{A}$ , so constant for fixed  $\mathfrak{A}$ . It follows that  $\text{pr}_{i_1, \dots, i_k, i}(R)$  is polynomial, so the running time of the calls to *Nonempty* are polynomial. For fixed  $\mathfrak{A}$ , the global running time of the procedure *Next* is in  $O(N^6)$  where  $N$  is the size of the input.

*Proof of Theorem 6.1.* Starting from an empty list of constraints, we add constraints on the variables  $x_1, \dots, x_n$  one by one, and maintain a compact representation of the  $n$ -ary relation defined by the constraints considered so far. Initially, we start with a compact representation

of the full relation  $A^n$ . In later steps, we use the procedure *Next* to compute a compact representation when a constraint is added, in  $O(N^6)$  for fixed  $\mathfrak{A}$  and  $N$  the size of the input. The instance is unsatisfiable if and only if at the final stage we end up with an empty representation. The entire running time of the algorithm is in  $O(N^7)$ .  $\square$

### Exercises.

91. Let  $\mathfrak{A}$  be the structure  $(\{0, 1\}; L_0, L_1)$  where  $L_i := \{(x, y, z) \mid x + y + z = i \pmod{2}\}$ , which has the Boolean minority  $m$  as polymorphism. Consider the instance

$$\exists x_1, \dots, x_5 (L_1(x_1, x_2, x_3) \wedge L_1(x_2, x_3, x_4) \wedge L_1(x_3, x_4, x_5) \wedge L_0(x_1, x_3, x_5))$$

Compute compact representations  $R'_\ell$  of  $R_\ell$ , for  $\ell \in \{1, 2, 3, 4\}$ .

92. (\*) Let  $\mathfrak{B}$  be a structure with a Maltsev polymorphism  $f$  and an *infinite* relational signature. Note that we have defined  $\text{CSP}(\mathfrak{B})$  only if  $\mathfrak{B}$  has a finite signature. If we want to define  $\text{CSP}(\mathfrak{B})$  also for structures  $\mathfrak{B}$  with an infinite signature, it is important to discuss how the relation symbols in the signature of  $\mathfrak{B}$  are represented in the input. We choose to represent a relation symbol  $R$  from  $\mathfrak{B}$  by listing the tuples in  $R^{\mathfrak{B}}$ . Adapt the Dalmau algorithm such that it can solve  $\text{CSP}(\mathfrak{B})$  in polynomial time for this choice of representing the relations in  $\mathfrak{B}$ .
93. (\*) The *graph isomorphism problem (GI)* is a famous computational problem that is neither known to be solvable in polynomial time, nor expected to be NP-hard. An instance of GI consists of two graphs  $G$  and  $H$ , and the question is to decide whether  $G$  and  $H$  are isomorphic. Consider the variant of the graph-isomorphism problem where the vertices are coloured, each color appears at most  $k$  times for some constant  $k$ , and the isomorphism between  $H$  and  $G$  that we are looking for is required to additionally preserve the colours. Show that this problem can be solved in polynomial time using Dalmau's algorithm (use the previous exercise).

## 7 Universal Algebra

### 7.1 Algebras and Clones

In universal algebra, an *algebra* is simply a structure with a purely functional signature. We will typically use bold font letters, like  $\mathbf{A}$ , to denote algebras, and the corresponding capital roman letters, like  $A$ , to denote their domain.

**Example 7.1 (Group).** A *group* is an algebra with a binary function symbol  $\circ$  for composition, a unary function symbol  $^{-1}$  for taking the inverse, and a constant denoted by  $e$ , satisfying

- $\forall x, y, z. x \circ (y \circ z) = (x \circ y) \circ z,$
- $\forall x. x \circ x^{-1} = e,$
- $\forall x. e \circ x = x,$  and  $\forall x. x \circ e = x.$

Note that all axioms are *universal* in the sense that all the variables are universally quantified (more on that comes later). The group is called *abelian* if it additionally satisfies

$$\forall x, y. x \circ y = y \circ x. \quad \triangle$$

**Example 7.2** (Ring). A (*commutative*) *ring* is an algebra  $\mathbf{A}$  with the signature  $\{\cdot, +, -, 0, 1\}$  where  $\cdot, +$  are binary,  $-$  is unary, and  $0, 1$  are constants, such that  $(A; +, -, 0)$  is an abelian group and additionally

$$\begin{aligned} \forall x, y, z. (xy)z &= x(yz) && \text{(associativity)} \\ \forall x. 1 \cdot x &= x && \text{(multiplicative unit)} \\ \forall x, y. xy &= yx && \text{(commutativity)} \\ \forall x, y, z. x(y + z) &= xy + xz && \text{(distributivity)} \end{aligned} \quad \triangle$$

**Example 7.3** (Module). Let  $\mathbf{R}$  be a ring. An  $\mathbf{R}$ -*module* is an algebra  $\mathbf{M}$  with the signature  $\{+, -, 0\} \cup \{f_r \mid r \in R\}$  such that  $(M; +, -, 0)$  is an abelian group and

$$\begin{aligned} \forall x, y. f_r(x + y) &= f_r(x) + f_r(y) \text{ holds for every } r \in R \\ \forall x. f_{r+s}(x) &= f_r(x) + f_s(x) \text{ holds for all } r, s \in R \\ \forall x. f_r(f_s(x)) &= f_{rs}(x) \text{ holds for all } r, s \in R. \end{aligned}$$

An  $\mathbf{R}$ -module is called *unitary* if it additionally satisfies  $\forall x. f_1(x) = x$ .  $\triangle$

**Example 7.4** (Semilattice). A *meet-semilattice*  $\mathfrak{S}$  is a  $\{\leq\}$ -structure with domain  $S$  such that  $\leq^{\mathfrak{S}}$  denotes a partial order where any two  $u, v \in S$  have a (unique) *greatest lower bound*  $u \wedge v$ , i.e., an element  $w$  such that  $w \leq u$ ,  $w \leq v$ , and for all  $w'$  with  $w' \leq u$  and  $w' \leq v$  we have  $w' \leq w$ . Dually, a *join-semilattice* is a partial order with least upper bounds, denoted by  $u \vee v$ . A *semilattice* is a meet-semilattice or a join-semilattice where the distinction between meet and join is either not essential or clear from the context.

Semilattices can also be characterised as  $\{\wedge\}$ -algebras where  $\wedge$  is a binary operation that must satisfy the following axioms

$$\begin{aligned} \forall x, y, z. x \wedge (y \wedge z) &= (x \wedge y) \wedge z && \text{(associativity)} \\ \forall x, y. x \wedge y &= y \wedge x && \text{(commutativity)} \\ \forall x. x \wedge x &= x && \text{(idempotency, or idempotence)}. \end{aligned}$$

Clearly, the operation  $\wedge^{\mathfrak{S}}$ , defined as above in a semilattice  $\mathfrak{S}$  viewed as a poset, satisfies these axioms. Conversely, if  $(S; \wedge)$  is a semilattice, then the formula  $x \wedge y = x$  defines a partial order on  $S$  which is a meet-semilattice (and  $x \wedge y = y$  defines a partial order on  $S$  which is a join-semilattice).

Note that the two ways of formalising semilattices differ when it comes to the notion of a substructure; a *subsemilattice* is referring to the substructure of a semilattice when formalised as an algebraic structure.  $\triangle$

**Example 7.5** (Lattice). A *lattice*  $\mathfrak{L}$  is a  $\{\leq\}$ -structure with domain  $L$  such that  $\leq^{\mathfrak{L}}$  denotes a partial order such that any two  $u, v \in L$  have a largest lower bound  $u \wedge v$  and a least upper bound, denoted by  $u \vee v$ . Lattices can also be characterised as  $\{\wedge, \vee\}$ -algebras where  $\wedge$  and  $\vee$  are semilattice operations (Example 7.4) that additionally satisfy

$$\forall x, y. x \wedge (x \vee y) = x \text{ and } x \vee (x \wedge y) = x \quad \text{(absorption).}$$

If  $\mathfrak{L}$  is a lattice and the operations  $\wedge$  and  $\vee$  are defined as above for semilattices, then these two operations also satisfy the absorption axiom. Conversely, if we are given an algebra  $(S; \wedge, \vee)$  satisfying the mentioned axioms, then the formula  $x \wedge y = x$  (equivalently, the formula  $x \vee y = y$ ) defines a partial order on  $S$  which is a lattice. Of course, there is potential danger of confusion of the symbols for lattice operations  $\wedge$  and  $\vee$  with the propositional connectives  $\wedge$  for conjunction and  $\vee$  for disjunction (which can be seen as lattice operations on the set  $\{0, 1\}$ ) which luckily should not cause trouble here. A lattice  $\mathfrak{L} = (L; \wedge, \vee)$  is called *distributive* if it satisfies

$$\forall x, y: x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z) \quad (\text{distributivity}). \quad \triangle$$

**The clone of an algebra.** If  $\mathbf{A}$  is an algebra with the signature  $\tau$ , then a  $\tau$ -term  $t(x_1, \dots, x_n)$  gives rise to a *term function*  $t^{\mathbf{A}}: A^n \rightarrow A$ ; the value of  $t^{\mathbf{A}}$  at  $a_1, \dots, a_n \in A$  can be obtained by replacing the variables  $x_1, \dots, x_n$  by  $a_1, \dots, a_n$  and evaluating in  $\mathbf{A}$ .

**Example 7.6.** If  $\mathbf{A}$  is a group, then the term function for the term  $(x \circ y^{-1}) \circ z$  is a Maltsev operation on  $A$ .  $\triangle$

**Example 7.7.** If  $t(x_1, x_2)$  is the term that just consists of the variable  $x_1$ , then  $t^{\mathbf{A}}$  equals the projection  $\pi_1^2$ .  $\triangle$

Algebras give rise to clones in the following way. We denote by  $\text{Clo}(\mathbf{A})$  the set of all term functions of  $\mathbf{A}$ . Clearly,  $\text{Clo}(\mathbf{A})$  is an operation clone since it is closed under compositions, and contains the projections.

**Polymorphism algebras.** In the context of complexity classification of CSPs, algebras arise as follows.

**Definition 7.8.** Let  $\mathfrak{B}$  be a relational structure with domain  $B$ . An algebra  $\mathbf{B}$  with domain  $B$  such that  $\text{Clo}(\mathbf{B}) = \text{Pol}(\mathfrak{B})$  is called a *polymorphism algebra* of  $\mathfrak{B}$ .

Note that a structure  $\mathfrak{B}$  has many different polymorphism algebras, since Definition 7.8 does not prescribe how to assign function symbols to the polymorphisms of  $\mathfrak{B}$ .

Any clone  $\mathcal{C}$  on a set  $D$  can be viewed as an algebra  $\mathbf{A}$  with domain  $D$  whose signature consists of the operations of  $\mathcal{C}$  themselves; that is, if  $f \in \mathcal{C}$ , then  $f^{\mathbf{A}} := f$ . We will therefore use concepts defined for algebras also for clones. In particular, the polymorphism clone  $\text{Pol}(\mathfrak{B})$  of a structure  $\mathfrak{B}$  might be viewed as an algebra, which we refer to as *the* polymorphism algebra of  $\mathfrak{B}$ . Note that the signature of the polymorphism algebra is always infinite, since we have polymorphisms of arbitrary finite arity.

## 7.2 Subalgebras, Products, Homomorphic Images

In this section we recall some basic universal-algebraic facts that will be used in the following subsections.

**Subalgebras.** Let  $\mathbf{A}$  be a  $\tau$ -algebra with domain  $A$ . A  $\tau$ -algebra  $\mathbf{B}$  with domain  $B \subseteq A$  is called a *subalgebra* of  $\mathbf{A}$  if for each  $f \in \tau$  of arity  $k$  we have  $f^{\mathbf{B}}(b_1, \dots, b_k) = f^{\mathbf{A}}(b_1, \dots, b_k)$  for all  $b_1, \dots, b_k \in B$ ; in this case, we write  $\mathbf{B} \leq \mathbf{A}$ . A *subuniverse* of  $\mathbf{A}$  is the domain of some subalgebra of  $\mathbf{A}$ . The smallest subuniverse of  $\mathbf{A}$  that contains a given set  $S \subseteq A$  is called the *subuniverse of  $\mathbf{A}$  generated by  $S$* , and the corresponding subalgebra is called the *subalgebra of  $\mathbf{A}$  generated by  $S$* .

**Products.** Let  $\mathbf{A}, \mathbf{B}$  be  $\tau$ -algebras with domain  $A$  and  $B$ , respectively. Then the *product*  $\mathbf{A} \times \mathbf{B}$  is the  $\tau$ -algebra with domain  $A \times B$  such that for each  $f \in \tau$  of arity  $k$  we have  $f^{\mathbf{A} \times \mathbf{B}}((a_1, b_1), \dots, (a_k, b_k)) = (f^{\mathbf{A}}(a_1, \dots, a_k), f^{\mathbf{B}}(b_1, \dots, b_k))$  for all  $a_1, \dots, a_k \in A$  and  $b_1, \dots, b_k \in B$ . More generally, when  $(\mathbf{A}_i)_{i \in I}$  is a sequence of  $\tau$ -algebras, indexed by some set  $I$ , then  $\prod_{i \in I} \mathbf{A}_i$  is the  $\tau$ -algebra  $\mathbf{A}$  with domain  $\prod_{i \in I} A_i$  such that  $f^{\mathbf{A}}((a_i^1)_{i \in I}, \dots, (a_i^k)_{i \in I}) = (f^{\mathbf{A}_i}(a_i^1, \dots, a_i^k))_{i \in I}$  for  $a_i^1, \dots, a_i^k \in A_i$ .

**Lemma 7.9.** *Let  $\mathbf{A}$  be the polymorphism algebra of a finite structure  $\mathfrak{A}$ . Then the (domains of the) subalgebras of  $\mathbf{A}^k$  are precisely the relations that have a primitive positive definition in  $\mathfrak{A}$ .*

*Proof.* A relation  $R \subseteq A^k$  is a subalgebra of  $\mathbf{A}^k$  if and only if for all  $m$ -ary  $f$  in the signature of  $\mathbf{A}$  and  $t^1, \dots, t^m \in R$ , we have  $(f(t_1^1, \dots, t_1^m), \dots, f(t_k^1, \dots, t_k^m)) \in R$ , which is the case if and only if  $R$  is preserved by all polymorphisms of  $\mathfrak{A}$ , which is the case if and only if  $R$  is primitive positive definable in  $\mathfrak{A}$  by Theorem 5.2.  $\square$

**Homomorphic Images.** Let  $\mathbf{A}$  and  $\mathbf{B}$  be  $\tau$ -algebras. Then a *homomorphism* from  $\mathbf{A}$  to  $\mathbf{B}$  is a mapping  $h: A \rightarrow B$  such that for all  $k$ -ary  $f \in \tau$  and  $a_1, \dots, a_k \in A$  we have

$$h(f^{\mathbf{A}}(a_1, \dots, a_k)) = f^{\mathbf{B}}(h(a_1), \dots, h(a_k)).$$

Note that if  $h$  is a homomorphism from  $\mathbf{A}$  to  $\mathbf{B}$  then the image of  $h$  is the domain of a subalgebra of  $\mathbf{B}$ ; this subalgebra is called a *homomorphic image* of  $\mathbf{A}$ .

**Definition 7.10.** A *congruence* of an algebra  $\mathbf{A}$  is an equivalence relation that is preserved by all operations in  $\mathbf{A}$ .

**Lemma 7.11.** *Let  $\mathfrak{B}$  be a finite structure, and  $\mathbf{B}$  be a polymorphism algebra of  $\mathfrak{B}$ . Then the congruences of  $\mathbf{B}$  are exactly the primitive positive definable equivalence relations over  $\mathfrak{B}$ .*

*Proof.* A direct consequence of Theorem 5.2.  $\square$

**Proposition 7.12** (see [29]). *Let  $\mathbf{A}$  be an algebra. Then  $E$  is a congruence of  $\mathbf{A}$  if and only if  $E$  is the kernel of a homomorphism from  $\mathbf{A}$  to some other algebra  $\mathbf{B}$ .*

**Example 7.13.** Let  $G = (V, E)$  be the undirected graph with  $V = \{a_1, \dots, a_4, b_1, \dots, b_4\}$  such that  $a_1, \dots, a_4$  and  $b_1, \dots, b_4$  induce a clique, for each  $i \in \{1, \dots, 4\}$  there is an edge between  $a_i$  and  $b_i$ , and otherwise there are no edges in  $G$ . Let  $\mathbf{A}$  be a polymorphism algebra of  $G$ . Then  $\mathbf{A}$  homomorphically maps to a two-element algebra  $\mathbf{B}$ . By Proposition 7.12, it suffices to show that  $\mathbf{A}$  has a congruence with two equivalence classes. By Lemma 7.11, it suffices to show that an equivalence relation of index two is primitive positive definable. Here is the primitive positive definition:

$$\exists u, v (E(x, u) \wedge E(y, u) \wedge E(x, v) \wedge E(y, v) \wedge E(u, v))$$

The equivalence classes of this relation are precisely  $\{a_1, \dots, a_4\}$  and  $\{b_1, \dots, b_4\}$ .  $\triangle$

**Example 7.14.** Let  $\mathbf{A}$  be the algebra with domain  $A := S_3 = \{\text{id}, (231), (312), (12), (23), (13)\}$  (the symmetric group on three elements), and a single binary operation, the composition function of permutations. Note that  $\mathbf{A}$  has the subalgebra induced by  $\{\text{id}, (123), (321)\}$ . Also note that  $\mathbf{A}$  homomorphically maps to  $(\{0, 1\}, +)$  where  $+$  is addition modulo 2: the preimage of 0 is  $\{\text{id}, (123), (321)\}$  and the preimage of 1 is  $\{(12), (23), (13)\}$ .  $\triangle$

When  $\mathbf{A}$  is a  $\tau$ -algebra, and  $h: A \rightarrow B$  is a mapping such that the kernel of  $h$  is a congruence of  $\mathbf{A}$ , we define the *quotient algebra*  $\mathbf{A}/h$  of  $\mathbf{A}$  under  $h$  to be the algebra with domain  $h(A)$  where

$$f^{\mathbf{A}/h}(h(a_1), \dots, h(a_k)) = h(f^{\mathbf{A}}(a_1, \dots, a_k))$$

where  $a_1, \dots, a_k \in A$  and  $f \in \tau$  is  $k$ -ary. This is well-defined since the kernel of  $h$  is preserved by all operations of  $\mathbf{A}$ . Note that  $h$  is a surjective homomorphism from  $\mathbf{A}$  to  $\mathbf{A}/h$ . The following is well known (see e.g. Theorem 6.3 in [29]).

**Lemma 7.15.** *Let  $\mathbf{A}$  and  $\mathbf{B}$  be algebras with the same signature, and let  $h: \mathbf{A} \rightarrow \mathbf{B}$  be a homomorphism. Then the image of any subalgebra  $\mathbf{A}'$  of  $\mathbf{A}$  under  $h$  is a subalgebra of  $\mathbf{B}$ , and the preimage of any subalgebra  $\mathbf{B}'$  of  $\mathbf{B}$  under  $h$  is a subalgebra of  $\mathbf{A}$ .*

*Proof.* Let  $f \in \tau$  be  $k$ -ary. Then for all  $a_1, \dots, a_k \in A'$ ,

$$f^{\mathbf{B}}(h(a_1), \dots, h(a_k)) = h(f^{\mathbf{A}}(a_1, \dots, a_k)) \in h(A'),$$

so  $h(A')$  is a subalgebra of  $\mathbf{B}$ . Now suppose that  $h(a_1), \dots, h(a_k)$  are elements of  $B'$ ; then  $f^{\mathbf{B}}(h(a_1), \dots, h(a_k)) \in B'$  and hence  $h(f^{\mathbf{A}}(a_1, \dots, a_k)) \in B'$ . So,  $f^{\mathbf{A}}(a_1, \dots, a_k) \in h^{-1}(B')$  which shows that  $h^{-1}(B')$  induces a subalgebra of  $\mathbf{A}$ .  $\square$

### 7.3 Pseudovarieties and Varieties

Varieties are a fascinatingly powerful concept to study classes of algebras. The fundamental result about varieties is Birkhoff's theorem, which links varieties with equational theories (Section 7.4). By Birkhoff's theorem, there is also a close relationship between varieties and the concept of an *abstract clone* (Section 7.5).

If  $\mathcal{K}$  is a class of algebras of the same signature, then

- $P(\mathcal{K})$  denotes the class of all products of algebras from  $\mathcal{K}$ .
- $P^{\text{fin}}(\mathcal{K})$  denotes the class of all finite products of algebras from  $\mathcal{K}$ .
- $S(\mathcal{K})$  denotes the class of all subalgebras of algebras from  $\mathcal{K}$ .
- $H(\mathcal{K})$  denotes the class of all homomorphic images of algebras from  $\mathcal{K}$ .

Note that closure under homomorphic images implies in particular closure under isomorphism. For the operators  $P$ ,  $P^{\text{fin}}$ ,  $S$  and  $H$  we often omit the brackets when applying them to single singleton classes that just contain one algebra, i.e., we write  $H(\mathbf{A})$  instead of  $H(\{\mathbf{A}\})$ . The elements of  $HS(\mathbf{A})$  are also called the *factors* of  $\mathbf{A}$ .

A class  $\mathcal{V}$  of algebras with the same signature  $\tau$  is called a *pseudovariety* if  $\mathcal{V}$  contains all homomorphic images, subalgebras, and direct products of algebras in  $\mathcal{V}$ , i.e.,  $H(\mathcal{V}) = S(\mathcal{V}) = P^{\text{fin}}(\mathcal{V}) = \mathcal{V}$ . The class  $\mathcal{V}$  is called a *variety* if  $\mathcal{V}$  also contains all (finite and infinite) products of algebras in  $\mathcal{V}$ . So the only difference between pseudovarieties and varieties is that pseudovarieties need not be closed under direct products of infinite cardinality. The smallest pseudovariety (variety) that contains an algebra  $\mathbf{A}$  is called the pseudovariety (variety) *generated* by  $\mathbf{A}$ .

**Lemma 7.16** (HSP lemma). *Let  $\mathbf{A}$  be an algebra.*

- The pseudovariety generated by  $\mathbf{A}$  equals  $\text{HSP}^{\text{fin}}(\mathbf{A})$ .
- The variety generated by  $\mathbf{A}$  equals  $\text{HSP}(\mathbf{A})$ .

*Proof.* Clearly,  $\text{HSP}^{\text{fin}}(\mathbf{A})$  is contained in the pseudovariety generated by  $\mathbf{A}$ , and  $\text{HSP}(\mathbf{A})$  is contained in the variety generated by  $\mathbf{A}$ . For the converse inclusion, it suffices to verify that  $\text{HSP}^{\text{fin}}(\mathbf{A})$  is closed under H, S, and  $\text{P}^{\text{fin}}$ . It is clear that  $\text{H}(\text{HSP}^{\text{fin}}(\mathbf{A})) = \text{HSP}^{\text{fin}}(\mathbf{A})$ . The second part of Lemma 7.15 implies that  $\text{S}(\text{HSP}^{\text{fin}}(\mathbf{A})) \subseteq \text{HS}(\text{SP}^{\text{fin}}(\mathbf{A})) = \text{HSP}^{\text{fin}}(\mathbf{A})$ . Finally,

$$\text{P}^{\text{fin}}(\text{HSP}^{\text{fin}}(\mathbf{A})) \subseteq \text{H P}^{\text{fin}} \text{S P}^{\text{fin}}(\mathbf{A}) \subseteq \text{HSP}^{\text{fin}} \text{P}^{\text{fin}}(\mathbf{A}) = \text{HSP}^{\text{fin}}(\mathbf{A}) .$$

The proof that  $\text{HSP}(\mathbf{A})$  is closed under H, S, and P is analogous. □

Pseudo-varieties are linked to the logic concepts from Section 4.

**Theorem 7.17.** *Let  $\mathfrak{C}$  be a finite structure with polymorphism algebra  $\mathbf{C}$ . Then  $\mathfrak{B} \in \text{I}(\mathfrak{C})$  if and only if there exists  $\mathbf{B} \in \text{HSP}^{\text{fin}}(\mathbf{C})$  such that  $\text{Clo}(\mathbf{B}) \subseteq \text{Pol}(\mathfrak{B})$ .*

*Proof.* We only prove the ‘if’ part of the statement here; the proof of the ‘only if’ part is similarly easy. There exists a finite number  $d \geq 1$ , a subalgebra  $\mathbf{D}$  of  $\mathbf{C}^d$ , and a surjective homomorphism  $h$  from  $\mathbf{D}$  to  $\mathbf{B}$ . We claim that  $\mathfrak{B}$  has a primitive positive interpretation  $I$  of dimension  $d$  in  $\mathfrak{C}$ . All operations of  $\mathbf{C}$  preserve  $D$  (viewed as a  $d$ -ary relation over  $\mathfrak{C}$ ), since  $\mathbf{D}$  is a subalgebra of  $\mathbf{C}^d$ . By Theorem 5.2, this implies that  $D$  has a primitive positive definition  $\delta(x_1, \dots, x_d)$  in  $\mathfrak{C}$ , which becomes the domain formula  $\delta_I$  of  $I$ . As coordinate map we choose the mapping  $h$ . Since  $h$  is an algebra homomorphism, the kernel  $K$  of  $h$  is a congruence of  $\mathbf{D}$ . It follows that  $K$ , viewed as a  $2d$ -ary relation over  $C$ , is preserved by all operations from  $\mathbf{C}$ . Theorem 5.2 implies that  $K$  has a primitive positive definition in  $\mathfrak{C}$ . This definition becomes the formula  $=_I$ . Finally, let  $R$  be a relation of  $\mathfrak{B}$  and let  $f$  be a function symbol from the signature of  $\mathbf{B}$ . By assumption,  $f^{\mathbf{B}}$  preserves  $R$ . It is easy to verify that then  $f^{\mathbf{C}}$  preserves  $h^{-1}(R)$ . Hence, all polymorphisms of  $\mathfrak{C}$  preserve  $h^{-1}(R)$ , and the relation  $h^{-1}(R)$  has a primitive positive definition in  $\mathfrak{C}$  (Theorem 5.2), which becomes the defining formula for the atomic formula  $R(x_1, \dots, x_k)$  in  $I$ . This concludes our construction of the primitive positive interpretation  $I$  of  $\mathfrak{B}$  in  $\mathfrak{C}$ . □

### Exercises.

94. Let  $B$  be a subuniverse of an algebra  $\mathbf{A}$  generated by  $S \subseteq A$ . Show that an element  $a \in A$  belongs to  $B$  if and only if there exists a term  $t(x_1, \dots, x_k)$  and elements  $s_1, \dots, s_k \in S$  such that  $a = t^{\mathbf{A}}(s_1, \dots, s_k)$ .
95. Show that the operators HS and SH are distinct.
96. Show that the operators SP and PS are distinct.

## 7.4 Birkhoff’s Theorem

A sentence in a functional signature is called an *identity* if it is of the form

$$\forall x_1, \dots, x_n: s = t$$

where  $s$  and  $t$  are  $\tau$ -terms. We present Birkhoff’s theorem for the special case of varieties generated by a single finite algebra.

**Theorem 7.18** (Birkhoff [13]; see e.g. [50] or [29]). *Let  $\tau$  be a functional signature, and  $\mathbf{A}$  and  $\mathbf{B}$  finite algebras with signature  $\tau$ . Then the following are equivalent.*

1. *All identities that hold in  $\mathbf{B}$  also hold in  $\mathbf{A}$ ;*
2.  $\mathbf{A} \in \text{HSP}^{\text{fin}}(\mathbf{B})$ ;
3.  $\mathbf{A} \in \text{HSP}(\mathbf{B})$ .

*Proof.* Trivially, 2. implies 3. To show that 3. implies 1., let  $\phi = \forall x_1, \dots, x_n. s = t$  be an identity that holds in  $\mathbf{B}$ . Then  $\phi$  is preserved in powers  $\mathbf{A} = \mathbf{B}^I$  of  $\mathbf{B}$ . To see this, let  $a_1, \dots, a_n \in A$  be arbitrary. Since  $\mathbf{B} \models \phi$  we have  $s^{\mathbf{B}}(a_1[j], \dots, a_n[j]) = t^{\mathbf{B}}(a_1[j], \dots, a_n[j])$  for all  $j \in I$ , and thus  $s^{\mathbf{A}}(a_1, \dots, a_n) = t^{\mathbf{A}}(a_1, \dots, a_n)$  by the definition of products. Since  $a_1, \dots, a_n$  were chosen arbitrarily, we have  $\mathbf{A} \models \phi$ . Moreover,  $\phi$  is true in subalgebras of algebras that satisfy  $\phi$  (this is true for universal sentences in general). Finally, suppose that  $\mathbf{B}$  is an algebra that satisfies  $\phi$ , and  $\mu$  is a surjective homomorphism from  $\mathbf{B}$  to some algebra  $\mathbf{A}$ . Let  $a_1, \dots, a_n \in A$ ; by surjectivity of  $\mu$  we can choose  $b_1, \dots, b_n$  such that  $\mu(b_i) = a_i$  for all  $i \leq n$ . Then

$$\begin{aligned} s^{\mathbf{B}}(b_1, \dots, b_n) = t^{\mathbf{B}}(b_1, \dots, b_n) &\Rightarrow \mu(s^{\mathbf{B}}(b_1, \dots, b_n)) = \mu(t^{\mathbf{B}}(b_1, \dots, b_n)) \\ &\Rightarrow t^{\mathbf{A}}(\mu(b_1), \dots, \mu(b_n)) = s^{\mathbf{A}}(\mu(b_1), \dots, \mu(b_n)) \\ &\Rightarrow t^{\mathbf{A}}(a_1, \dots, a_n) = s^{\mathbf{A}}(a_1, \dots, a_n). \end{aligned}$$

1. implies 2.: Let  $a_1, \dots, a_k$  be the elements of  $\mathbf{A}$ , define  $m := |B|^k$ , and let  $C$  be  $B^k$ . Let  $c^1, \dots, c^m$  be the elements of  $C$ ; write  $c_i$  for  $(c_i^1, \dots, c_i^m)$ . Let  $\mathbf{S}$  be the smallest subalgebra of  $\mathbf{B}^m$  that contains  $c_1, \dots, c_k$ ; so the elements of  $\mathbf{S}$  are precisely those of the form  $t^{\mathbf{B}^m}(c_1, \dots, c_k)$ , for a  $k$ -ary  $\tau$ -term  $t$ . Define  $\mu: S \rightarrow A$  by

$$\mu(t^{\mathbf{B}^m}(c_1, \dots, c_k)) := t^{\mathbf{A}}(a_1, \dots, a_k).$$

**Claim 1:**  $\mu$  is well-defined. Suppose that  $t^{\mathbf{B}^m}(c_1, \dots, c_k) = s^{\mathbf{B}^m}(c_1, \dots, c_k)$ ; then  $t^{\mathbf{B}} = s^{\mathbf{B}}$  by the choice of  $S$ , and by assumption we have  $t^{\mathbf{A}}(a_1, \dots, a_k) = s^{\mathbf{A}}(a_1, \dots, a_k)$ .

**Claim 2:**  $\mu$  is surjective. For all  $i \leq k$ , the element  $c_i$  is mapped to  $a_i$ .

**Claim 3:**  $\mu$  is a homomorphism from  $\mathbf{S}$  to  $\mathbf{A}$ . Let  $f \in \tau$  be of arity  $n$  and let  $s_1, \dots, s_n \in S$ . For  $i \leq n$ , write  $s_i = t_i^{\mathbf{S}}(c_1, \dots, c_k)$  for some  $\tau$ -term  $t_i$ . Then

$$\begin{aligned} \mu(f^{\mathbf{S}}(s_1, \dots, s_n)) &= \mu(f^{\mathbf{S}}(t_1^{\mathbf{S}}(c_1, \dots, c_k), \dots, t_n^{\mathbf{S}}(c_1, \dots, c_k))) \\ &= \mu(f^{\mathbf{S}}(t_1^{\mathbf{S}}, \dots, t_n^{\mathbf{S}})(c_1, \dots, c_k)) \\ &= \mu((f(t_1, \dots, t_n))^{\mathbf{S}}(c_1, \dots, c_k)) \\ &= (f(t_1, \dots, t_n))^{\mathbf{A}}(a_1, \dots, a_k) \\ &= f^{\mathbf{A}}(t_1^{\mathbf{A}}(a_1, \dots, a_k), \dots, t_n^{\mathbf{A}}(a_1, \dots, a_k)) \\ &= f^{\mathbf{A}}(\mu(s_1), \dots, \mu(s_n)). \end{aligned}$$

Therefore,  $\mathbf{A}$  is the homomorphic image of the subalgebra  $\mathbf{S}$  of  $\mathbf{B}^m$ , and so  $\mathbf{A} \in \text{HSP}^{\text{fin}}(\mathbf{B})$ .  $\square$

Theorem 7.18 is important for analysing the constraint satisfaction problem for a structure  $\mathfrak{B}$ , since it can be used to transform the ‘negative’ statement of not interpreting certain finite structures into a ‘positive’ statement of having polymorphisms satisfying non-trivial identities: this will be the content of Section 7.5 and Section 7.6.



## 7.5 (Abstract) Clones

Clones (in the literature often *abstract clones*) relate to operation clones in the same way as (abstract) groups relate to permutation groups: the elements of a clone correspond to the functions of an operation clone, and the signature contains composition symbols to code how functions compose. Since an operation clone contains functions of various arities, a clone will be formalized as a multi-sorted structure, with a sort for each arity.

**Definition 7.19.** A *clone*  $\mathbf{C}$  is a multi-sorted structure with sorts  $\{C^{(i)} \mid i \in \mathbb{N}\}$  and the signature  $\{\pi_i^k \mid 1 \leq i \leq k\} \cup \{\text{comp}_l^k \mid k, l \geq 1\}$ . The elements of the sort  $C^{(k)}$  will be called the *k-ary operations* of  $\mathbf{C}$ . We denote a clone by

$$\mathbf{C} = (C^{(0)}, C^{(1)}, \dots; (\pi_i^k)_{1 \leq i \leq k}, (\text{comp}_l^k)_{k, l \geq 1})$$

and require that  $\pi_i^k$  is a constant in  $C^{(k)}$ , and that  $\text{comp}_l^k: C^{(k)} \times (C^{(l)})^k \rightarrow C^{(l)}$  is an operation of arity  $k + 1$ . Moreover, it holds that

$$\text{comp}_k^k(f, \pi_1^k, \dots, \pi_k^k) = f \quad (1)$$

$$\text{comp}_l^k(\pi_i^k, f_1, \dots, f_k) = f_i \quad (2)$$

$$\begin{aligned} \text{comp}_l^k(f, \text{comp}_l^m(g_1, h_1, \dots, h_m), \dots, \text{comp}_l^m(g_k, h_1, \dots, h_m)) = \\ \text{comp}_l^m(\text{comp}_m^k(f, g_1, \dots, g_k), h_1, \dots, h_m). \end{aligned} \quad (3)$$

The final equation generalises associativity in groups and monoids, and we therefore refer to it by *associativity*. We also write  $f(g_1, \dots, g_k)$  instead of  $\text{comp}_l^k(f, g_1, \dots, g_k)$  when  $l$  is clear from the context. So associativity might be more readable as

$$f(g_1(h_1, \dots, h_m), \dots, g_k(h_1, \dots, h_m)) = f(g_1, \dots, g_k)(h_1, \dots, h_m).$$

Every operation clone  $\mathcal{C}$  gives rise to an abstract clone  $\mathbf{C}$  in the obvious way:  $\pi_i^k \in C^{(k)}$  denotes the  $k$ -ary  $i$ -th projection in  $\mathcal{C}$ , and  $\text{comp}_l^k(f, g_1, \dots, g_k) \in C^{(l)}$  denotes the composed function  $(x_1, \dots, x_l) \mapsto f(g_1(x_1, \dots, x_l), \dots, g_k(x_1, \dots, x_l)) \in \mathcal{C}$ .

**Example 7.20.**  $\forall x_1, x_2: f(x_1, x_2) = f(x_2, x_1)$  holds in an algebra  $\mathbf{A}$  if and only if

$$\text{Clo}(\mathbf{A}) \models \text{comp}_2^2(f^{\mathbf{A}}, \pi_1^2, \pi_2^2) = \text{comp}_2^2(f^{\mathbf{A}}, \pi_2^2, \pi_1^2). \quad \triangle$$

In the following, we will also use the term ‘abstract clone’ in situations where we want to stress that we are working with a clone and *not* with an operation clone.

**Definition 7.21.** Let  $\mathbf{C}$  and  $\mathbf{D}$  be clones. A function  $\xi: C \rightarrow D$  is called a (*clone*) *homomorphism* if

- $\xi$  preserves arities of functions, i.e.,  $\xi(C^{(i)}) \subseteq D^{(i)}$  for all  $i \in \mathbb{N}$ ;
- $\xi((\pi_i^k)^{\mathbf{C}}) = (\pi_i^k)^{\mathbf{D}}$  for all  $1 \leq i \leq k$ ;
- $\xi(f(g_1, \dots, g_n)) = \xi(f)(\xi(g_1), \dots, \xi(g_n))$  for all  $n, m \geq 1$ ,  $f \in C^{(n)}$ , and  $g_1, \dots, g_n \in C^{(m)}$ .

**Example 7.22.** We write **Proj** for the abstract clone of an algebra with at least two elements all of whose operations are projections; note that any such algebra has the same abstract clone (up to isomorphism), and that **Proj** has a homomorphism into any other clone.  $\triangle$

**Example 7.23.** All abstract clones of an algebra on a one-element are isomorphic, too, but of course not isomorphic to **Proj**. Any clone homomorphically maps to this trivial clone.  $\triangle$

**Proposition 7.24** (Formulation of Birkhoff’s theorem for clones). *Let  $\mathcal{C}$  and  $\mathcal{D}$  be operation clones on finite sets. Then the following are equivalent.*

1. *There is a surjective clone homomorphism from  $\mathcal{C}$  to  $\mathcal{D}$ ;*
2. *there are algebras  $\mathbf{A}$  and  $\mathbf{B}$  with the same signature  $\tau$  such that  $\text{Clo}(\mathbf{A}) = \mathcal{D}$ ,  $\text{Clo}(\mathbf{B}) = \mathcal{C}$ , and all universal conjunctive  $\tau$ -sentences that hold in  $\mathbf{B}$  also hold in  $\mathbf{A}$ ;*
3. *there are algebras  $\mathbf{A}$  and  $\mathbf{B}$  with the same signature such that  $\text{Clo}(\mathbf{A}) = \mathcal{D}$ ,  $\text{Clo}(\mathbf{B}) = \mathcal{C}$ , and  $\mathbf{A} \in \text{HSP}^{\text{fin}}(\mathbf{B})$  (equivalently,  $\mathbf{A} \in \text{HSP}(\mathbf{B})$ ).*

In the study of the complexity of CSPs, the equivalence between (1) and (3) in the above is the most relevant, since (3) is related to our most important tool to prove NP-hardness of CSPs (because of the link between pseudovarieties and primitive positive interpretations from Theorem 7.17), and since (1) is the universal-algebraic property that will be used in the following (see e.g. Theorem 7.30 below).

The following lemma is central for our applications of abstract clones when studying the complexity of CSPs.

**Lemma 7.25.** *Let  $\mathbf{C}$  be a clone and let  $\mathbf{F}$  be the clone of a finite algebra such that there is no clone homomorphism from  $\mathbf{C}$  to  $\mathbf{F}$ . Then there is a primitive positive sentence in the language  $\tau$  of (abstract) clones that holds in  $\mathbf{C}$  but not in  $\mathbf{F}$ .*

*Proof.* Let  $\mathbf{E}$  be the expansion of  $\mathbf{C}$  by constant symbols such that every element  $e$  of  $\mathbf{E}$  is named by a constant  $c_e$ . Let  $V$  be the set of atomic sentences that hold in  $\mathbf{E}$ . Let  $U$  be the first-order theory of  $\mathbf{F}$ . Suppose that  $U \cup V$  has a model  $\mathbf{M}$ . There might be elements in  $\mathbf{M}$  outside of  $\bigcup_i M^{(i)}$ . But the  $\tau$ -reduct of the restriction of  $\mathbf{M}$  to  $\bigcup_i M^{(i)}$  must be isomorphic to  $\mathbf{F}$ , since each of the  $M^{(i)}$  is finite; we identify it with  $\mathbf{F}$ . Note that for all constants  $c_e$  we have that  $c_e^{\mathbf{M}} \in \mathbf{F}$ . Since  $\mathbf{M}$  satisfies all atomic formulas that hold in  $\mathbf{E}$ , we have that the mapping  $e \mapsto c_e^{\mathbf{M}}$ , for  $e$  an element of  $\mathbf{E}$ , is a homomorphism from  $\mathbf{C}$  to  $\mathbf{F}$ , in contradiction to our assumptions.

So  $U \cup V$  is unsatisfiable, and by compactness of first-order logic there exists a finite subset  $V'$  of  $V$  such that  $V' \cup U$  is unsatisfiable. Replace each of the new constant symbols in  $V'$  by an existentially quantified variable; then the conjunction of the resulting sentences from  $V$  is a primitive positive sentence, and it must be false in  $\mathbf{F}$ .  $\square$

A set of identities  $\Sigma$  is called *trivial* if there exists an algebra  $\mathbf{A}$  that satisfies  $\Sigma$  and  $\text{Clo}(\mathbf{A})$  is isomorphic to **Proj**.

**Corollary 7.26.** *Let  $\mathbf{A}$  be an algebra. If there is no clone homomorphism from  $\text{Clo}(\mathbf{A})$  to **Proj**, then there exists a non-trivial finite set of identities that holds in  $\mathbf{A}$ .*

## 7.6 Taylor Terms

The following goes back to Walter Taylor [70]. We slightly deviate from the historic definition in that we do not require idempotency – this allows us to give stronger formulations of several results in the following.

**Definition 7.27** (Taylor operations). A function  $f: B^n \rightarrow B$ , for  $n \geq 2$ , is called a *Taylor operation* if for each  $1 \leq i \leq n$  there are variables  $z_1, \dots, z_n, z'_1, \dots, z'_n \in \{x, y\}$  with  $z_i \neq z'_i$  such that for all  $x, y \in B$

$$f(z_1, \dots, z_n) = f(z'_1, \dots, z'_n).$$

Examples for Taylor operations are binary commutative operations, majority operations, and Maltsev operations. We do not insist on idempotency for Taylor operations; so also quasi majority operations (Exercise 54) are examples of Taylor operations. Note that an  $n$ -ary operation  $f$  is Taylor if and only if it satisfies a set of  $n$  equations that can be written as

$$f \begin{pmatrix} x & ? & ? & \cdots & ? \\ ? & x & ? & & \vdots \\ \vdots & ? & \ddots & \ddots & \vdots \\ \vdots & & \ddots & x & ? \\ ? & \cdots & \cdots & ? & x \end{pmatrix} = f \begin{pmatrix} y & ? & ? & \cdots & ? \\ ? & y & ? & & \vdots \\ \vdots & ? & \ddots & \ddots & \vdots \\ \vdots & & \ddots & y & ? \\ ? & \cdots & \cdots & ? & y \end{pmatrix}$$

where  $f$  is applied row-wise and  $?$  stands for either  $x$  or  $y$ .

**Definition 7.28** (Taylor terms). Let  $\mathbf{B}$  be a  $\tau$ -algebra. A *Taylor term* of  $\mathbf{B}$  is a  $\tau$ -term  $t(x_1, \dots, x_n)$ , for  $n \geq 2$ , such that  $t^{\mathbf{B}}$  is a Taylor operation.

Note that a term  $t(x_1, \dots, x_n)$  is a Taylor term in  $\mathbf{B}$  if and only if  $\text{Clo}(\mathbf{B}) \models \Phi_n(t^{\mathbf{B}})$ , where  $\Phi_n(f)$  is the sentence

$$\bigwedge_{i \leq n} \exists z, z' \in \{\pi_1^2, \pi_2^2\}^n (z_i \neq z'_i \wedge \text{comp}_2^n(f, z_1, \dots, z_n) = \text{comp}_2^n(f, z'_1, \dots, z'_n)). \quad (4)$$

The following lemma is entirely obvious and I don't know why the formal proof is so long, there must be a better way of presenting it (if giving a proof is necessary at all).

**Lemma 7.29.**  $\Phi_n(f)$  is equivalent to

$$\bigwedge_{i \leq n} \exists v, v' \in \{\pi_1^n, \dots, \pi_n^n\}^n (v_i \neq v'_i \wedge \text{comp}_n^n(f, v_1, \dots, v_n) = \text{comp}_n^n(f, v'_1, \dots, v'_n)).$$

*Proof.* Let  $i \in \{1, \dots, n\}$  be arbitrary. Suppose that  $z, z' \in \{\pi_1^2, \pi_2^2\}^n$  are as in Equation 1. Then define  $v, v' \in \{\pi_1^n, \dots, \pi_n^n\}^n$  as follows: if  $z_j = \pi_k^2$  for  $k \in \{1, 2\}$ , then  $v_j := \pi_k^n$ , and similarly if  $z'_j = \pi_k^2$  for  $k \in \{1, 2\}$ , then  $v'_j := \pi_k^n$ . Then  $v$  and  $v'$  witness that the formula given in the statement is true:  $z_i \neq z'_i$  implies that  $v_i \neq v'_i$ . Moreover,  $\text{comp}_2^n(f, z_1, \dots, z_i) = \text{comp}_2^n(f, z'_1, \dots, z'_n)$  implies that

$$\text{comp}_n^2(\text{comp}_2^n(f, z_1, \dots, z_n), \pi_1^n, \pi_2^n) = \text{comp}_n^2(\text{comp}_2^n(f, z'_1, \dots, z'_n), \pi_1^n, \pi_2^n)$$

We compute

$$\begin{aligned} & \text{comp}_n^2(\text{comp}_2^n(f, z_1, \dots, z_n), \pi_1^n, \pi_2^n) \\ &= \text{comp}_n^2(f, \text{comp}_2^n(z_1, \pi_1^n, \pi_2^n), \dots, \text{comp}_2^n(z_n, \pi_1^n, \pi_2^n)) \\ &= \text{comp}_n^n(f, v_1, \dots, v_n) \end{aligned}$$

and similarly  $\text{comp}_n^2(\text{comp}_n^2(f, z'_1, \dots, z'_n), \pi_1^n, \pi_2^n) = \text{comp}_n^2(f, v'_1, \dots, v'_n)$  and hence

$$\text{comp}_n^2(f, v_1, \dots, v_n) = \text{comp}_n^2(f, v'_1, \dots, v'_n)$$

as required.

Conversely, let  $i \leq n$ , and suppose that  $v, v' \in \{\pi_1^n, \dots, \pi_n^n\}^n$  are as in the formula above. Define  $z, z' \in \{\pi_1^2, \pi_2^2\}^n$  as follows. If  $v_j = v_i$  define  $z_j := \pi_1^2$ , and  $z_j := \pi_2^2$  otherwise. Similarly, if  $v'_j = v'_i$  then  $z'_j := \pi_1^2$ , and  $z'_j := \pi_2^2$  otherwise. Then  $z$  and  $z'$  witness that  $\Phi_n(f)$  is true: since  $v_i \neq v'_i$  we have  $z_i \neq z'_i$ , and

$$\begin{aligned} & \text{comp}_2^n(\text{comp}_n^2(f, v_1, \dots, v_n), z_1, \dots, z_n) \\ &= \text{comp}_2^n(f, \text{comp}_2^n(v_1, z_1, \dots, z_n), \dots, \text{comp}_2^n(v_n, z_1, \dots, z_n)) \\ &= \text{comp}_2^n(f, z_1, \dots, z_n) \end{aligned}$$

Similarly,  $\text{comp}_2^n(\text{comp}_n^2(f, v'_1, \dots, v'_n), z'_1, \dots, z'_n) = \text{comp}_2^n(f, z'_1, \dots, z'_n)$ . Hence,

$$\text{comp}_2^n(f, z_1, \dots, z_n) = \text{comp}_2^n(f, z'_1, \dots, z'_n)$$

as required. □

Walter Taylor did not just introduce Taylor operations, but he proved a beautiful statement about their existence. The following is a slightly expanded presentation of the proof of Lemma 9.4 in [49]. An algebra is called *idempotent* if all of its operations are idempotent.

**Theorem 7.30.** *Let  $\mathbf{B}$  be an idempotent algebra. Then the following are equivalent.*

- (1) *there is no homomorphism from  $\text{Clo}(\mathbf{B})$  to  $\mathbf{Proj}$ ;*
- (2)  *$\mathbf{B}$  has a Taylor term.*

*Proof.* To show the easy implication from (2) to (1), suppose for contradiction that there is a homomorphism  $\xi$  from  $\text{Clo}(\mathbf{B})$  to  $\mathbf{Proj}$ . Let  $f$  be the element of  $\text{Clo}(\mathbf{B})$  that is denoted by  $t^{\mathbf{B}}$ . By definition of  $\mathbf{Proj}$  we have  $\xi(f) = \pi_l^n$  for some  $l \leq n$ . By assumption,  $\mathbf{B}$  satisfies

$$\forall x, y. t(z_1, \dots, z_n) = t(z'_1, \dots, z'_n) \tag{5}$$

for  $z_1, \dots, z_n, z'_1, \dots, z'_n \in \{x, y\}$  such that  $z_l \neq z'_l$ . Then  $\text{Clo}(\mathbf{B})$  satisfies

$$\text{comp}_2^n(f, \pi_{i_1}^2, \dots, \pi_{i_n}^2) = \text{comp}_2^n(f, \pi_{j_1}^2, \dots, \pi_{j_n}^2) \tag{6}$$

for  $i_1, \dots, i_n, j_1, \dots, j_n \in \{1, 2\}$  such that  $i_l = 1$  if and only if  $z_l = x$ ,  $j_l = 1$  if and only if  $z'_l = x$ , and  $i_l \neq j_l$ . Since  $\xi(f) = \pi_l^n$  we therefore obtain that  $\pi_1^2 = \pi_2^2$ , which does not hold in  $\mathbf{Proj}$ , a contradiction.

To show the implication from (1) to (2), suppose that  $\text{Clo}(\mathbf{B})$  does not homomorphically map to  $\mathbf{Proj}$ . Then Lemma 7.25 implies that there is a primitive positive sentence in the language of clones that holds in  $\text{Clo}(\mathbf{B})$  but not in  $\mathbf{Proj}$ . Note that by introducing new existentially quantified variables we can assume that this sentence is of the form  $\exists u_1, \dots, u_r. \phi$  where  $\phi$  is a conjunction of atoms of the form  $y = \text{comp}_l^m(x_0, x_1, \dots, x_m)$  or of the form  $y = \pi_l^m$  for  $y, x_0, x_1, \dots, x_m \in \{u_1, \dots, u_r\}$  and  $l, m \in \mathbb{N}$ . For example the equation

$$\text{comp}_2^2(u, \text{comp}_2^2(u, \pi_2^2, \pi_1^2)) = \text{comp}_2^2(\text{comp}_2^2(u, \pi_2^2, \pi_1^2), u)$$

involving the free variable  $u$  is equivalent to

$$\begin{aligned} \exists u_1, u_2, u_3 (u_1 = \text{comp}_2^2(u, \pi_2^2, \pi_1^2) \\ \wedge u_2 = \text{comp}_2^2(f, u, u_1) \\ \wedge u_3 = \text{comp}_2^2(u, u_1, u) \\ \wedge u_2 = u_3) . \end{aligned}$$

For  $l, m \in \mathbb{N}$  and  $n = lm$  we write  $f * g$  as a shortcut for

$$\text{comp}_n^l(f, \text{comp}_n^m(g, \pi_1^n, \dots, \pi_m^n), \dots, \text{comp}_n^m(g, \pi_{(l-1)m+1}^n, \dots, \pi_n^n)) .$$

Note that

$$\text{Clo}(\mathbf{B}) \models (g = \text{comp}_l^n(f * g, \pi_1^l, \dots, \pi_l^l, \pi_1^l, \dots, \pi_l^l, \dots, \pi_1^l, \dots, \pi_l^l)) \quad (7)$$

$$\text{and } \text{Clo}(\mathbf{B}) \models (f = \text{comp}_m^n(f * g, \pi_1^l, \dots, \pi_1^l, \pi_2^l, \dots, \pi_2^l, \dots, \pi_l^l, \dots, \pi_l^l)) \quad (8)$$

since  $\mathbf{B}$  is idempotent. For  $i \in \{1, \dots, r\}$ , let  $k_i \in \mathbb{N}$  be the arity of  $u_i$ , and define

$$u := u_1 * (u_2 * (\dots (u_{r-1} * u_r) \dots)) . \quad (9)$$

Observe that for each  $i$  we can obtain  $u_i$  from  $u$  by composing  $u$  with projections. In order to formalise this, we need a compact notation for strings of arguments consisting of projection constants. In this notation, (7) reads as  $y = \text{comp}_l^n(x * y, (1, \dots, l)^m)$ , and (8) reads as  $x = \text{comp}_m^n(x * y, 1^m, \dots, l^m)$ . Similarly, setting  $k := k_1 \cdots k_r$  we have

$$u_i = \text{comp}_{k_i}^{k_i}(u, \bar{p}_i) \text{ where } \bar{p}_i := (1^{k_1 \cdots k_{i-1}}, \dots, k_i^{k_1 \cdots k_{i-1}})^{k_{i+1} \cdots k_n} \in \{\pi_1^{k_i}, \dots, \pi_{k_i}^{k_i}\}^k .$$

Let  $n := k^2$ . Then every term of the form

$$t := \text{comp}_l^{k_i}(u_i, u_{i_1}, \dots, u_{i_{k_i}})$$

can be written as

$$\text{comp}_l^n(u * u, \bar{q}^t)$$

for  $\bar{q}^t \in \{\pi_1^l, \dots, \pi_l^l\}^n$  obtained from  $\bar{p}^i$  by replacing character  $l \leq k_i$  by the string  $\bar{p}^i$ ; see Figure 10 for an illustration. Similarly, every term of the form  $t := u_i$  can be written as  $\text{comp}_{k_i}^n(u * u, \bar{q}^t)$  for  $\bar{q}^t \in \{\pi_1^{k_i}, \dots, \pi_{k_i}^{k_i}\}^n$  obtained by  $k$  times concatenating  $\bar{p}^i$  with itself. In this way, every conjunct of  $\phi$  of the form

$$u_j = \text{comp}_{k_j}^{k_i}(u_i, u_{i_1}, \dots, u_{i_{k_i}})$$

can be written in the form

$$\text{comp}_{k_j}^n(u * u, \bar{q}^{t_1}) = \text{comp}_{k_j}^n(u * u, \bar{q}^{t_2})$$

for  $t_1 := u_j$  and  $t_2 := \text{comp}_{k_j}^{k_i}(u_i, u_{i_1}, \dots, u_{i_{k_i}})$ . Conjuncts of  $\phi$  of the form  $u_j = \pi_l^m$  can be rewritten similarly. Let  $\psi$  be the conjunction of all these equations; note that  $\phi$  implies  $\psi$ . Let  $\theta$  be the formula obtained from  $\psi$  by replacing each occurrence of  $u * u$  by a variable symbol  $f$ . Note that the sentence  $\exists f. \theta$  holds in  $\text{Clo}(\mathbf{B})$ . It suffices to show that every  $f \in \text{Clo}(\mathbf{B})$  that satisfies  $\theta$  is a Taylor operation.

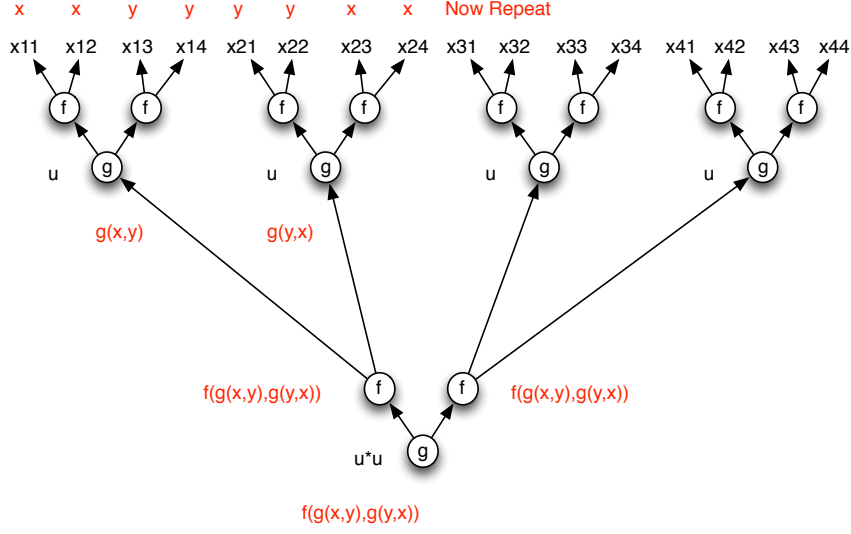


Figure 10: An illustration of  $u * u$  and  $u := f * g$  for the proof of Taylor's theorem, and how to recover the composed term  $f(g(x, y), g(y, x))$  from  $u * u$  by identifying arguments.

Suppose for contradiction that for  $\ell \in \{1, \dots, n\}$  there are no  $\bar{v}, \bar{v}' \in \{\pi_1^m, \dots, \pi_m^m\}^n$ , for some  $m \leq n$  with  $\bar{v}_\ell \neq \bar{v}'_\ell$  such that  $\text{comp}_m^n(f, \bar{v}) = \text{comp}_m^n(f, \bar{v}')$  (otherwise,  $f$  is a Taylor operation by the argument from Lemma 7.29).

We claim that the assignment  $\rho$  that maps  $u_s$  to  $\text{comp}_{k_s}^n(\pi_\ell^n, \bar{q}^{u_s}) = \bar{q}_\ell^{u_s}$ , for  $s \in \{1, \dots, r\}$ , satisfies all conjuncts of  $\phi$ , contradicting our assumptions. First note that  $\bar{q}_\ell^{u_s} = \bar{p}_{\ell_1}^s$  where  $\ell_1 \in \{1, \dots, k\}$  is such that  $\ell = (\ell_1 - 1)k + \ell_2$  for some  $\ell_2 \in \{1, \dots, k\}$ . Indeed, consider a conjunct of  $\phi$

$$u_j = t \text{ for } t = \text{comp}_{k_j}^{k_i}(u_i, u_{i_1}, \dots, u_{i_{k_i}}).$$

By construction,  $\theta$  contains the conjunct

$$\text{comp}(f, \bar{q}^{u_j}) = \text{comp}(f, \bar{q}^t).$$

So by assumption we must have

$$\text{comp}(\pi_\ell^n, \bar{q}^{u_j}) = \bar{q}_\ell^{u_j} = \bar{q}_\ell^t = \text{comp}(\pi_\ell^n, \bar{q}^t). \quad (10)$$

Then  $\rho$  satisfies this conjunct since

$$\begin{aligned} \text{comp}_{k_j}^{k_i}(\rho(u_i), \rho(u_{i_1}), \dots) &= \text{comp}_{k_j}^{k_i}(\bar{q}_\ell^{u_i}, \bar{q}_\ell^{u_{i_1}}, \dots) \\ &= \text{comp}_{k_j}^{k_i}(\bar{p}_{\ell_1}^i, \bar{p}_{\ell_1}^{i_1}, \dots) \\ &= \bar{q}_\ell^t && \text{(by definition of } \bar{q}^t) \\ &= \bar{q}_\ell^{u_j} && \text{(by (10))} \\ &= \rho(u_j) \end{aligned} \quad \square$$

**Lemma 7.31.** *Let  $\mathfrak{B}$  and  $\mathfrak{C}$  be homomorphically equivalent structures. Then  $\mathfrak{B}$  has a Taylor polymorphism if and only if  $\mathfrak{C}$  has a Taylor polymorphism.*

*Proof.* Let  $h$  be a homomorphism from  $\mathfrak{B}$  to  $\mathfrak{C}$ , and  $g$  be a homomorphism from  $\mathfrak{C}$  to  $\mathfrak{B}$ . Suppose that  $f$  is a Taylor polymorphism for  $\mathfrak{B}$  of arity  $n$ . It suffices to show that the operation  $f'$  given by  $(x_1, \dots, x_n) \mapsto h(f(g(x_1), \dots, g(x_n)))$  is a Taylor polymorphism of  $\mathfrak{C}$ . Indeed, for all  $i \leq n$  we have that

$$\begin{aligned} f'(v_{1,i}, \dots, v_{n,i}) &= h(f(g(v_{1,i}), \dots, g(v_{n,i}))) \\ &= h(f(g(v'_{1,i}), \dots, g(v'_{n,i}))) \\ &= f'(v'_{1,i}, \dots, v'_{n,i}) \end{aligned} \quad \square$$

**Lemma 7.32.** *Let  $\mathfrak{B} \in C(\mathfrak{C})$  and suppose that  $\mathfrak{C}$  has a Taylor polymorphism. Then  $\mathfrak{B}$  has a Taylor polymorphism.*

*Proof.* Suppose that  $\mathfrak{C}$  is a core and that  $\mathfrak{B}$  has been obtained by expanding  $\mathfrak{C}$  by adding unary singleton relations. Let  $f$  be the Taylor polymorphism of  $\mathfrak{C}$ . Then  $\hat{f}$  (see Exercise 37) is an automorphism with inverse  $i \in \text{Pol}(\mathfrak{C})$ . The function  $i(f(x, \dots, x))$  is a Taylor operation and idempotent, and therefore a polymorphism of  $\mathfrak{B}$ .  $\square$

Alternatively, one can prove Lemma 7.32 using the fact that  $C(\mathfrak{C}) \subseteq H(I(\mathfrak{C}))$ , the observation that any  $\mathfrak{B}' \in I(\mathfrak{C})$  has a Taylor polymorphism, and Lemma 7.31.

**Corollary 7.33.** *Let  $\mathfrak{B}$  be a finite structure. Then the following are equivalent.*

- $K_3 \notin H(I(\mathfrak{B}))$
- $\mathfrak{B}$  has a Taylor polymorphism.

*Proof.* If  $\mathfrak{B}$  has a Taylor polymorphism, then so has any structure in  $I(\mathfrak{B})$ , and so has any structure in  $H(I(\mathfrak{B}))$  by Lemma 7.31. But all polymorphisms of  $K_3$  are essentially unary (Proposition 5.12), so  $K_3 \notin H(I(\mathfrak{B}))$ .

Conversely, suppose that  $K_3 \notin H(I(\mathfrak{B}))$ . If  $\mathfrak{B}'$  is the core of  $\mathfrak{B}$ , and  $\mathfrak{C}$  is the expansion of  $\mathfrak{B}'$  by all unary singleton relations, then  $C(H(\mathfrak{B})) \subseteq H(I(\mathfrak{B}))$  implies that  $K_3$  is not primitive positive interpretable in  $\mathfrak{C}$ . Hence, the idempotent clone  $\text{Pol}(\mathfrak{C})$  does not homomorphically map to **Proj**. Theorem 7.30 shows that  $\mathfrak{C}$  and thus also  $\mathfrak{B}'$  must have a Taylor polymorphism. Lemma 7.31 implies that  $\mathfrak{B}$  has a Taylor polymorphism.  $\square$

**Corollary 7.34.** *Let  $\mathfrak{B}$  be a finite structure. Then  $\mathfrak{B}$  has a Taylor polymorphism, or  $\text{CSP}(\mathfrak{B})$  is NP-hard.*

*Proof.* If  $K_3 \in H(I(\mathfrak{B}))$  then  $\text{CSP}(\mathfrak{B})$  is NP-hard by Corollary 4.15. Otherwise,  $\mathfrak{B}$  has a Taylor polymorphism by Corollary 7.33.  $\square$

## 7.7 The Tractability Conjecture

The following has been conjectured (in slightly different, but equivalent form) by Bulatov, Jeavons, and Krokhin in [27], and is known under the name *tractability conjecture*. Two solutions to this conjecture have been announced in 2017 by Bulatov [24] and by Zhuk [71].

**Conjecture 2** (Tractability Conjecture). *Let  $\mathfrak{B}$  be a finite relational structure. If  $\text{Pol}(\mathfrak{B})$  contains a Taylor polymorphism, then  $\text{CSP}(\mathfrak{B})$  is in P.*

For idempotent algebras  $\mathbf{A}$  there is yet another characterisation for the existence of Taylor polymorphisms that follows from the following result of Bulatov and Jeavons [26] (Proposition 4.14).

**Theorem 7.35.** *Let  $\mathbf{B}$  be an idempotent algebra. Then  $\text{HSP}^{\text{fin}}(\mathbf{B})$  contains an at least 2-element algebra all of whose operations are projections if and only if  $\text{HS}(\mathbf{B})$  does.*

*Proof.* Suppose that  $\mathbf{C} \in S(\mathbf{B}^d)$  for some  $d \in \mathbb{N}$  has a congruence  $K$  such that all operations of  $\mathbf{A} := \mathbf{C}/K$  are projections. Let  $I \subseteq \{1, \dots, d\}$  be a maximal set such that for each  $i \in I$  there exists  $b_i \in B$  such that the set

$$C((b_i)_{i \in I}) := \{c \in C \mid c_i = b_i \text{ for all } i \in I\}$$

is not contained in a class of  $K$ . Note that such a set exists because for  $I = \emptyset$  we have that  $C((b_i)_{i \in I}) = C$  is not contained in one class of  $K$  since  $\mathbf{C}/K$  has at least two elements.

Without loss of generality, we may assume that  $\{1, \dots, d\} \setminus I = \{1, \dots, k\}$ . Since  $\mathbf{B}$  is idempotent,  $C((b_i)_{i \in I})$  is the domain of a subalgebra  $\mathbf{C}'$  of  $\mathbf{C}$ . Let  $K' := K \cap (C')^2$ . All operations of  $\mathbf{A}' := \mathbf{C}'/K'$  are restrictions of operations in  $\mathbf{A}$  and hence projections. Let

$$B' := \{b \in B \mid (b_1, \dots, b_d) \in C'\}.$$

Then  $B'$  is the domain of a subalgebra  $\mathbf{B}'$  of  $\mathbf{B}$ . The image

$$K'' := \{(c_k, e_k) \in (B')^2 \mid (c, e) \in K'\}$$

of  $K'$  under the  $k$ -th projection is a congruence of  $\mathbf{B}'$ , and it must have more than one equivalence class: otherwise there would be  $b_1, \dots, b_d, b'_1, \dots, b'_{k-1} \in B$  such that the tuples  $(b_1, \dots, b_{k-1}, b_k, b_{k+1}, \dots, b_d)$  and  $(b'_1, \dots, b'_{k-1}, b_k, b_{k+1}, \dots, b_d)$  are in different  $K'$ -classes. But then  $I \cup \{k\}$  is such that  $C((b_i)_{i \in I \cup \{k\}})$  is not contained in one class of  $K$ , contradicting the maximality of  $I$ . Therefore,  $\mathbf{B}'/K'' \in \text{HS}(\mathbf{B})$  is an algebra with at least two elements all whose operations are projections.  $\square$

Since the size of the algebras in  $\text{HS}(\mathbf{B})$  is bounded by the size of  $\mathbf{B}$ , this leads to an algorithm that decides whether a given finite structure  $\mathfrak{B}$  satisfies the equivalent conditions in Theorem 7.30. We summarise various equivalent conditions for finite idempotent algebras that were treated in this chapter.

**Corollary 7.36.** *Let  $\mathbf{B}$  be a finite idempotent algebra. Then the following are equivalent.*

1.  $\mathbf{B}$  has a Taylor term.
2. there is no homomorphism from  $\text{Clo}(\mathbf{B})$  to  $\mathbf{Proj}$ .
3.  $\mathbf{B}$  satisfies some non-trivial finite set of identities.
4.  $\text{HSP}(\mathbf{B})$  does not contain an at least 2-element algebra all of whose operations are projections.
5.  $\text{HS}(\mathbf{B})$  does not contain an at least 2-element algebra all of whose operations are projections.

*Proof.* The equivalence of (1) and (2) is Theorem 7.30.

The equivalence of (2) and (3) follows from Corollary 7.26.

The equivalence of (2) and (4) follows from Proposition 7.24.

The equivalence of (4) and (5) follows from Theorem 7.18 combined with Theorem 7.35.  $\square$



**Exercises.**

97. Let  $\mathbf{A}$  be a finite idempotent algebra. Then there exists an operation  $f \in \text{Clo}(\mathbf{A})^{(k)}$  such that for every  $B \subseteq A$  and every  $b_0$  in the subalgebra of  $\mathbf{A}$  generated by  $B$  there exist  $b_1, \dots, b_k \in B$  such that  $b_0 = f(b_1, \dots, b_k)$ .

## 8 Undirected Graphs

This section contains a proof of the dichotomy for finite undirected graphs of Hell and Nešetřil, Theorem 1.5. We prove something stronger, namely that the tractability conjecture (Conjecture 2) is true for finite undirected graphs  $\mathfrak{B}$  [22]. More specifically, the following is true.

**Theorem 8.1.** *Let  $\mathfrak{B}$  be a finite undirected graph. Then either*

- $\mathfrak{B}$  is bipartite (i.e., homomorphic to  $K_2$ ) or has a loop, or
- $\text{H}(\text{I}(\mathfrak{B}))$  contains all finite structures.

Note that in combination with Corollary 7.33, this theorem implies that the tractability conjecture (Conjecture 2) holds for undirected graphs. This theorem also has a remarkable consequence in universal algebra, whose significance goes beyond the study of the complexity of CSPs, and which provides a strengthening of Taylor’s theorem (Theorem 7.30), discovered by Siggers in 2010 (see Section 8.2).

### 8.1 The Hell-Nešetřil Theorem

The graph  $K_4 - \{0, 1\}$  (a clique with four vertices where one edge is missing) is called a *diamond*. A graph is called *diamond-free* if it does not contain a copy of a diamond as a (not necessarily induced) subgraph. For every  $\ell \in \mathbb{N}$ , the graph  $(K_3)^\ell$  is an example of a diamond-free graph.

**Lemma 8.2.** *Let  $\mathfrak{B}$  be a finite undirected loopless graph which is not bipartite. Then  $\mathfrak{B}$  interprets primitively positively a graph that is homomorphically equivalent to a diamond-free core containing a triangle.*

*Proof.* For a binary relations  $R_1, R_2 \subseteq B^2$ , define  $R_1 \circ R_2$  to be the binary relation

$$\{(x, y) \mid \exists z(R_1(x, z) \wedge R_2(z, y))\}.$$

For  $R \subseteq B^2$  and  $k \geq 1$ , define  $R^1 := R$  and  $R^{k+1} := R^k \circ R$ . Note that  $R^k$  is primitively positively definable in  $(B; R)$ . We may assume that

1.  $\text{H}(\text{I}(\mathfrak{B}))$  does not contain a non-bipartite loopless graph with fewer vertices than  $\mathfrak{B}$ , since otherwise we could replace  $\mathfrak{B}$  by this graph. In particular,  $\mathfrak{B}$  must then be a core.
2.  $\mathfrak{B} = (V; E)$  contains a triangle: if the length of the shortest odd cycle in  $\mathfrak{B}$  is  $k$ , then  $(B; E^{k-2})$  is a graph and contains a triangle, so it can replace  $\mathfrak{B}$ .

**Claim 1.** Every vertex of  $\mathfrak{B}$  is contained in a triangle: Otherwise, we can replace  $\mathfrak{B}$  by the subgraph of  $\mathfrak{B}$  induced by set defined by the primitive positive formula  $\exists u, v (E(x, u) \wedge E(x, v) \wedge E(u, v))$  which still contains a triangle, contradicting our first assumption.

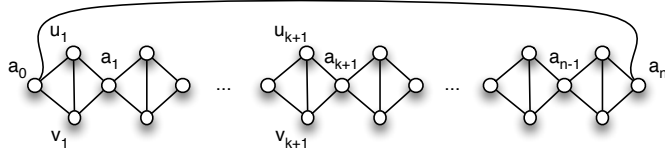


Figure 11: Diagram for the proof of Lemma 8.2.

**Claim 2.**  $\mathfrak{B}$  does not contain a copy of  $K_4$ . Otherwise, if  $a$  is an element from a copy of  $K_4$ , then the subgraph of  $\mathfrak{B}$  induced by the set defined by the primitive positive formula  $E(a, x)$  is a non-bipartite graph  $\mathfrak{A}$ , which has strictly less vertices than  $\mathfrak{B}$  because  $a \notin A$ . Moreover,  $\mathfrak{A}$  is by Theorem 4.24 homomorphically equivalent to a graph with a primitive positive interpretation in  $\mathfrak{B}$ , contrary to our initial assumption.

**Claim 3.** The graph  $\mathfrak{B}$  must also be diamond-free. To see this, let  $R$  be the binary relation with the primitive positive definition

$$R(x, y) :\Leftrightarrow \exists u, v (E(x, u) \wedge E(x, v) \wedge E(u, v) \wedge E(u, y) \wedge E(v, y))$$

and let  $T$  be the transitive closure of  $R$ . The relation  $T$  is clearly symmetric, and since every vertex of  $\mathfrak{B}$  is contained in a triangle, it is also reflexive, and hence an equivalence relation of  $\mathfrak{B}$ . Since  $B$  is finite, for some  $n$  the formula  $\exists u_1, \dots, u_n (R(x, u_1) \wedge R(u_1, u_2) \wedge \dots \wedge R(u_n, y))$  defines  $T$ , showing that  $T$  is primitively positively definable in  $\mathfrak{B}$ .

We claim that the graph  $\mathfrak{B}/T$  (see Example 4.12) does not contain loops. It suffices to show that  $T \cap E = \emptyset$ . Otherwise, let  $(a, b) \in T \cap E$ . Choose  $(a, b)$  in such a way that the shortest sequence  $a = a_0, a_1, \dots, a_n = b$  with  $R(a_0, a_1), R(a_1, a_2), \dots, R(a_{n-1}, a_n)$  in  $\mathfrak{B}$  is shortest possible; see Figure 11. This chain cannot have the form  $R(a_0, a_1)$  because  $\mathfrak{B}$  does not contain  $K_4$  subgraphs. Suppose first that  $n = 2k$  is even. Let the vertices  $u_1, v_1, u_{k+1}$  and  $v_{k+1}$  be as depicted in Figure 11. Let  $S$  be the set defined by

$$\exists x_1, \dots, x_k (E(u_{k+1}, x_1) \wedge E(v_{k+1}, x_1) \wedge R(x_1, x_2) \wedge \dots \wedge R(x_{k-1}, x_k) \wedge E(x_k, x)).$$

Note that  $a_0, u_1, v_1 \in S$  form a triangle. If  $a_n \in S$  then we obtain a contradiction to the minimal choice of  $n$ . Hence, the subgraph induced by the primitively positively definable set  $S$  is non-bipartite and strictly smaller than  $\mathfrak{B}$ , in contradiction to the initial assumption.

If  $n = 2k + 1$  is odd, we can argue analogously with the set  $S$  defined by the formula

$$\exists x_1, \dots, x_k (R(a_{k+1}, x_1) \wedge R(x_1, x_2) \wedge \dots \wedge R(x_{k-1}, x_k) \wedge E(x_k, x))$$

and again obtain a contradiction. So we conclude that  $\mathfrak{B}/T$  does not contain loops. It also follows that  $\mathfrak{B}/T$  contains a triangle, because  $\mathfrak{B}$  contains a triangle.

Thus, the initial assumption on  $\mathfrak{B}$  then implies that  $T$  must be the trivial equivalence relation, which in turn implies that  $\mathfrak{B}$  does not contain any diamonds.  $\square$

**Lemma 8.3** (from [22]). *Let  $\mathfrak{B}$  be a diamond-free undirected graph and let  $h: (K_3)^k \rightarrow \mathfrak{B}$  be a homomorphism. Then the image of  $h$  is isomorphic to  $(K_3)^m$  for some  $m \leq k$ .*

*Proof.* Let  $I \subseteq \{1, \dots, k\}$  be a maximal set such that  $\ker(h) \subseteq \ker(\text{pr}_I)$ . Note that  $\text{pr}_I$  is defined even if  $I = \emptyset$  (Definition 6.5). Such a set exists, because  $\ker(\text{pr}_\emptyset)$  is the total relation. We claim that  $\ker(h) = \ker(\text{pr}_I)$ ; this clearly implies the statement.

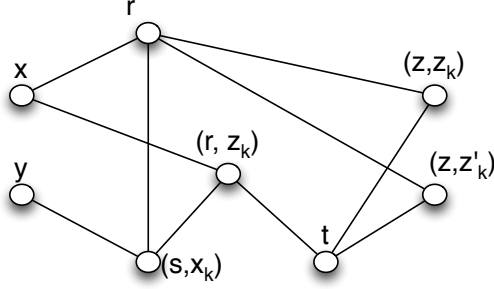


Figure 12: Diagram for the proof of Lemma 8.3.

By the maximality of  $I$ , for every  $j \in \{1, \dots, k\} \setminus I$  there are  $x, y \in (K_3)^k$  such that  $h(x) = h(y)$  and  $x_j \neq y_j$ . We have to show that for all  $z_1, \dots, z_k, z'_j \in \{0, 1, 2\}$

$$h(z_1, \dots, z_j, \dots, z_k) = h(z_1, \dots, z_{j-1}, z'_j, z_{j+1}, \dots, z_k).$$

We may suppose that  $z_j \neq x_j$  and  $z'_j = x_j$ . To simplify notation, we assume that  $j = k$ . As we have seen in Exercises 7 and 8, any two vertices in  $(K_3)^k$  have a common neighbour.

- Let  $r$  be a common neighbour of  $x$  and  $(z, z_k) := (z_1, \dots, z_k)$ . Note that  $r$  and  $(z, z'_k)$  are adjacent, too.
- For all  $i \neq k$  we choose an element  $s_i$  of  $K_3$  that is distinct from both  $r_i$  and  $y_i$ . Since  $x_k$  is distinct from  $r_k$  and  $y_k$  we have that  $(s, x_k) := (s_1, \dots, s_{k-1}, x_k)$  is a common neighbour of  $r$  and  $y$ .
- The tuple  $(r, z_k) := (r_1, \dots, r_{k-1}, z_k)$  is a common neighbour of both  $x$  and  $(s, x_k)$ .
- Finally, for  $i \neq k$  choose  $t_i$  to be distinct from  $z_i$  and  $r_i$ , and choose  $t_k$  to be distinct from  $z_k$  and from  $z'_k$ . Then  $t := (t_1, \dots, t_{k-1}, t_k)$  is a common neighbour of  $(z, z_k)$ , of  $(z, z'_k)$ , and of  $(r, z_k)$ .

The situation is illustrated in Figure 12. Since  $\mathfrak{B}$  is diamond-free,  $h(x) = h(y)$  implies that  $h(r) = h(r, z_k)$  and for the same reason  $h(z, z_k) = h(z, z'_k)$  which completes the proof.  $\square$

**Lemma 8.4** (from [22]). *If a finite diamond-free graph  $\mathfrak{B}$  contains a triangle, then for some  $k \in \mathbb{N}$  there is a primitive positive interpretation of  $(K_3)^k$  with constants in  $\mathfrak{B}$ .*

*Proof.* We construct a strictly increasing sequence of subgraphs  $G_1 \subset G_2 \subset \dots$  of  $\mathfrak{B}$  such that  $G_i$  is isomorphic to  $(K_3)^{k_i}$  for some  $k_i \in \mathbb{N}$ . Let  $G_1$  be any triangle in  $\mathfrak{B}$ . Suppose now that  $G_i$  has already been constructed. If the domain of  $G_i$  is primitively positively definable in  $\mathfrak{B}$  with constants, then we are done. Otherwise, there exists an idempotent polymorphism  $f$  of  $\mathfrak{B}$  and  $v_1, \dots, v_k \in G_i$  such that  $f(v_1, \dots, v_k) \notin G_i$ . The restriction of  $f$  to  $G_i$  is a homomorphism from  $(K_3)^{k_i}$  to the diamond-free graph  $\mathfrak{B}$ . Lemma 8.3 shows that  $G_{i+1} := f((G_i)^k)$  induces a copy of  $(K_3)^{k_{i+1}}$  for some  $k_{i+1} \leq k$ . Since  $f$  is idempotent, we have that  $G_i \subseteq G_{i+1}$ , and by the choice of  $f$  the containment is strict. Since  $\mathfrak{B}$  is finite, for some  $m$  the set  $G_m$  must have a primitive positive definition in  $\mathfrak{B}$  with constants.  $\square$

*Proof of Theorem 8.1.* Let  $\mathfrak{B}$  be a finite undirected graph that is not bipartite. By Lemma 8.2  $\mathfrak{B}$  interprets primitively positively a graph that is homomorphically equivalent to a diamond-free core  $\mathfrak{C}$  containing a triangle. Then Lemma 8.4 applied to  $\mathfrak{C}$  implies that for some  $k \in \mathbb{N}$  there is a primitive positive interpretation of  $(K_3)^k$  with constants in  $\mathfrak{C}$ . Since  $\mathfrak{C}$  is a core, and since  $(K_3)^k$  is homomorphically equivalent to  $K_3$ , it follows that there is a primitive positive interpretation of a structure that is homomorphically equivalent to  $K_3$  in  $\mathfrak{C}$ . The structure  $K_3$  interprets all finite structures primitive positively (Theorem 4.16), so Theorem 4.24 implies that  $\text{H}(\text{I}(\mathfrak{B}))$  contains all finite structures.  $\square$

## 8.2 Siggers Operations of Arity 6

An operation  $s: B^6 \rightarrow B$  is called *Siggers operation* if

$$s(x, y, x, z, y, z) = s(y, x, z, x, z, y)$$

holds for all  $x, y, z \in B$ .

**Theorem 8.5** (from [69]). *Let  $\mathfrak{B}$  be a finite structure. Then either  $\mathfrak{B}$  interprets all finite structures up to homomorphic equivalence, or  $\mathfrak{B}$  has a Siggers polymorphism.*

*Proof.* Pick  $k \geq 1$  and  $a, b, c \in B^k$  such that  $\{(a_i, b_i, c_i) \mid i \leq k\} = B^3$ . Let  $R$  be the binary relation on  $B^k$  such that  $(u, v) \in R$  if and only if there exists a 6-ary  $s \in \text{Pol}(\mathfrak{B})$  such that  $u = s(a, b, a, c, b, c)$  and  $v = s(b, a, c, a, c, b)$ . We make the following series of observations.

- The vertices  $a, b, c \in B^k$  induce in  $(B^k; R)$  a copy of  $K_3$ : each of the six edges of  $K_3$  is witnessed by one of the six 6-ary projections from  $\text{Pol}(\mathfrak{B})$ .
- The relation  $R$  is symmetric: Suppose that  $(u, v) \in R$  and let  $s \in \text{Pol}(\mathfrak{B})$  be such that  $u = s(a, b, a, c, b, c)$  and  $v = s(b, a, c, a, c, b)$ . Define  $s' \in \text{Pol}(\mathfrak{B})$  by  $s'(x_1, \dots, x_6) := s(x_2, x_1, x_4, x_3, x_6, x_5)$ ; then

$$v = s(b, a, c, a, c, b) = s'(a, b, a, c, b, c)$$

$$u = s(a, b, a, c, b, c) = s'(b, a, c, a, c, b)$$

and hence  $s'$  witnesses that  $(v, u) \in R$ .

- If the graph  $(B^k; R)$  contains a loop  $(w, w) \in R$ , then there exists a 6-ary  $s \in \text{Pol}(\mathfrak{B})$  such that

$$s(a, b, a, c, b, c) = w = s(b, a, c, a, c, b).$$

The operation  $s$  is Siggers: for all  $x, y, z \in B$  there exists an  $i \leq k$  such that  $(x, y, z) = (a_i, b_i, c_i)$ , and the above implies that

$$s(a_i, b_i, a_i, c_i, b_i, c_i) = s(b_i, a_i, c_i, a_i, c_i, b_i)$$

and we are done in this case.

So we may assume in the following that  $(B^k; R)$  is a simple (i.e., undirected and loopless) graph that contains a copy of  $K_3$ . The relation  $R$  (as a  $2k$ -ary relation over  $B$ ) is preserved by  $\text{Pol}(\mathfrak{B})$ , and hence  $(B^k; R)$  has a primitive positive interpretation in  $\mathfrak{B}$ . By Theorem 8.1 applied to the undirected graph  $(B^k; R)$ , there is a primitive positive interpretation in  $(B^k; R)$  of all finite structures up to homomorphic equivalence, and hence also in  $\mathfrak{B}$ , and this concludes the proof.  $\square$

## 9 Congruence Lattices

Let  $\mathbf{A}$  be a  $\tau$ -Algebra. We write  $\text{Con}(\mathbf{A})$  for the set of all congruences of  $\mathbf{A}$  (Definition 7.10). Clearly,  $\text{Con}(\mathbf{A})$  is closed under arbitrary intersections. On the other hand, the union of two congruences is in general not a congruence.

**Definition 9.1.** Let  $R$  and  $S$  be binary relations. Then the *composition* of  $R$  and  $S$  is the binary relation  $R \circ S := \{(x, z) \mid \exists y (R(x, y) \wedge R(y, z))\}$ .

**Lemma 9.2.** Let  $\mathbf{A}$  be an algebra. Then  $(\text{Con}(\mathbf{A}), \subseteq)$  is a complete lattice (Example 7.5).

*Proof.* Let  $(E_i)_{i \in I}$  be a sequence of congruences of  $\mathbf{A}$ . Define

$$\bigvee_{i \in I} E_i = \bigcup \{E_{i_1} \circ \cdots \circ E_{i_k} \mid i_1, \dots, i_k \in I, k \in \mathbb{N}\}$$

Note that this is the smallest (with respect to inclusion) equivalence relation that contains all the  $E_i$ . Let  $f \in \tau$  be  $n$ -ary and  $(a_1, b_1), \dots, (a_n, b_n) \in E$ . Then there are  $i_1, \dots, i_k \in I$  such that for all  $j \in \{1, \dots, n\}$

$$(a_j, b_j) \in E_{i_1} \circ \cdots \circ E_{i_k}.$$

Hence,  $(f(a_1, \dots, a_n), f(b_1, \dots, b_n)) \in E_{i_1} \circ \cdots \circ E_{i_k}$  and  $E \in \text{Con}(\mathbf{A})$ .  $\square$

Every algebra has the following two congruences.

- $\Delta_A$ : the *diagonal relation*  $\{(a, a) \mid a \in A\}$ .
- $\nabla_A$ : the *universal relation*  $A^2$ .

Congruences that are different from  $\nabla_A$  are called *proper*. Algebras  $\mathbf{A}$  where  $\Delta_A$  and  $\nabla_A$  are the only congruences are called *simple*. For example, groups that are simple in the sense of group theory are simple in our more general sense. We present another example.

**Example 9.3.** Let  $G$  be a permutation group on the set  $A$ . Let  $\mathbf{A}$  be the algebra with domain  $A$  and signature  $G$ , and define  $g^{\mathbf{A}} := g$  for all  $g \in G$ . Then  $\mathbf{A}$  is simple if and only if  $G$  is primitive as a permutation group.  $\triangle$

**Definition 9.4.** Two algebras  $\mathbf{A}_1, \mathbf{A}_2$  with the same domain  $A$  are called *polynomially equivalent* if the expansions of  $\mathbf{A}_1, \mathbf{A}_2$  by constant operations for each element of  $A$  have the same term operations.

### Exercises.

98. Show that if  $\mathbf{A}$  is an idempotent algebra and  $C$  a congruence of  $\mathbf{A}$ , then every congruence class of  $C$  is a subalgebra of  $\mathbf{A}$ .
99. Show that if  $\mathbf{A}$  is an idempotent algebra, and  $\mathbf{B}$  is polynomially equivalent to  $\mathbf{A}$ , then  $\mathbf{A}$  and  $\mathbf{B}$  have the same congruence lattice.
100. Show that  $\text{Clo}(\mathbf{A})$  and  $\text{Clo}^{(1)}(\mathbf{A})$  have the same congruence lattice.

## 9.1 Congruence Permutability

Two congruences  $C_1, C_2 \in \text{Con}(\mathbf{A})$  *permute* if

$$C_1 \circ C_2 = C_2 \circ C_1.$$

An algebra  $\mathbf{A}$  is called *congruence permutable* if all pairs of congruences of  $\mathbf{A}$  permute.

**Lemma 9.5.** *Let  $\mathbf{A}$  be an algebra such that  $\text{Clo}(\mathbf{A})$  contains a Maltsev operation  $p$ . Then  $\mathbf{A}$  is congruence permutable.*

Note that the congruences of most classical algebras, such as groups, rings, fields, etc., do have a Maltsev term operation, and hence are congruence permutable.

*Proof.* Let  $C, E \in \text{Con}(\mathbf{A})$  and let  $(a, b) \in C \circ E$ . Then there exists  $c \in A$  with  $(a, c) \in C$  and  $(c, b) \in E$ . Note that

$$b = p^{\mathbf{A}}(c, c, b) \ C \ p^{\mathbf{A}}(a, c, b) \ E \ p^{\mathbf{A}}(a, b, b) = a$$

and thus  $(b, a) \in C \circ E$  and  $(a, b) \in E \circ C$ . □

**Theorem 9.6** (Maltsev). *Let  $\mathcal{K}$  be a class of  $\tau$ -algebras. Then all the algebras in  $\text{HSP}(\mathcal{K})$  are congruences permutable if and only if there exists a  $\tau$ -term  $t(x, y, z)$  such that every algebra in  $\mathcal{K}$  satisfies  $\forall x, y. t(y, x, x) = t(x, x, y) = y$ .*

*Proof.*  $\Leftarrow$ : if every algebra in  $\mathcal{K}$  has a Maltsev term operation, then so does  $\text{HSP}(\mathcal{K})$ , and hence the statement follows from Lemma 9.5.

$\Rightarrow$ : Let  $\mathbf{F} := F_{\mathcal{K}}(\{x, y, z\})$ . For  $\mathbf{F} := \mathbf{F}_{\mathcal{K}}(X)$  und  $u, v \in \{x, y, z\}$  we write  $C(u, v)$  for the smallest congruence of  $\mathbf{F}$  that contains  $(u, v)$ . Let  $C_1 := C(x, y) \in \text{Con}(\mathbf{F})$  and  $C_2 := C(y, z) \in \text{Con}(\mathbf{F})$ . Since  $(x, z) \in C_1 \circ C_2 = C_2 \circ C_1$  there exists  $b \in F$  with  $(x, b) \in C_2$  and  $(b, z) \in C_1$ . Since  $\mathbf{F}$  is generated by  $\{x, y, z\}$ , there is a  $\tau$ -term  $p(x, y, z)$  with  $b = p^{\mathbf{F}}(x, y, z)$ . We will show that  $\mathcal{K} \models \forall x, y. p(x, x, y) = y$ . Let  $\mathbf{A} \in \mathcal{K}$  and  $u, v \in A$ . Let  $f: \mathbf{F} \rightarrow \mathbf{A}$  be a homomorphism with  $f(x) = u, f(y) = u$ , and  $f(z) = v$ . Then  $f(p^{\mathbf{F}}(x, y, z)) = p^{\mathbf{A}}(u, u, v)$ . Since  $(x, y) \in \text{Ker}(f)$  we have  $C_1 \subseteq \text{Ker}(f)$ . Thus,  $(b, z) \in \text{Ker}(f)$  and

$$v = f(z) = f(b) = f(p^{\mathbf{F}}(x, y, z)) = p^{\mathbf{A}}(u, u, v).$$

$\mathcal{K} \models \forall x, y. p(y, x, x) = y$  can be shown similarly. □

## 9.2 Congruence Distributivity

A lattice  $(P; \wedge, \vee, 0, 1)$  is called *distributive* if

$$\begin{aligned} \forall x, y, z \ ((x \wedge y) \vee z &= (x \vee z) \wedge (y \vee z)) \\ \wedge \ ((x \vee y) \wedge z &= (x \wedge z) \vee (y \wedge z)). \end{aligned}$$

An example of a distributive lattice is the set of subsets of a set  $S$ , ordered by inclusion:  $(\mathcal{P}(S); \subseteq)$ . If the congruence lattice of  $\mathbf{A}$  is distributive, we call  $\mathbf{A}$  *congruence distributive*.

**Lemma 9.7.** *Every algebra with a majority term operation is congruence distributive.*

*Proof.* Seien  $C, D, E \in \text{Con}(\mathfrak{A})$  und  $(a, b) \in C \wedge (D \vee E)$ .  
Dann:  $(a, b) \in C$  und es gibt  $c_1, \dots, c_n$  mit  $aDc_1Ec_2Dc_3 \dots c_nEb$ .  
Für alle  $c \in A$  gilt

$$m^{\mathfrak{A}}(a, c, b)C \underbrace{m^{\mathfrak{A}}(a, c, a)}_{=a}$$

Damit

$$\begin{aligned} a &= m^{\mathfrak{A}}(a, a, b)(C \wedge D)m^{\mathfrak{A}}(a, c_1, b) && \text{(zwei Mal Beobachtung)} \\ &\quad (C \wedge E)m^{\mathfrak{A}}(a, c_2, b) \\ &\quad \dots \\ &\quad (C \wedge D)m^{\mathfrak{A}}(a, c_n, b) \\ &\quad (C \wedge E)m^{\mathfrak{A}}(a, b, b) = b \end{aligned}$$

Also  $(a, b) \in (C \wedge D) \vee (C \wedge E)$ . □

**Example 9.8.** Sei  $A$  durch  $<$  linear geordnet. Betrachte  $\mathfrak{A} := (A; f)$  mit

$$f^{\mathfrak{A}}(x, y, z) := \min(\max(x, y), \max(x, z), \max(y, z))$$

(die Medianfunktion). △

**Example 9.9.** Sei  $A$  beliebige Menge. Betrachte  $\mathfrak{A} := (A; f)$  mit

$$f^{\mathfrak{A}}(x, y, z) := \begin{cases} z & \text{falls } y = z \\ x & \text{sonst} \end{cases} \quad \triangle$$

**Theorem 9.10** (Jónsson). *Die Algebren in  $HSP(\mathcal{K})$  sind genau dann kongruenzdistributiv, wenn es ein  $n \in \mathbb{N}$  und  $\tau$ -Terme  $p_0, \dots, p_n$  gibt so dass*

$$\begin{aligned} \mathcal{K} \models \forall x, y. p_i(x, y, x) &= x && \text{für } i \in \{1, \dots, n\} \\ p_0(x, y, z) &= x \\ p_i(x, x, y) &= p_{i+1}(x, x, y) && \text{für } i \text{ gerade} \\ p_i(x, y, y) &= p_{i+1}(x, y, y) && \text{für } i \text{ ungerade} \\ p_n(x, y, z) &= z \end{aligned}$$

*Proof.* “ $\Rightarrow$ ”. Sei  $\mathbf{F} := F_{\mathcal{K}}(\{x, y, z\})$ .

$$C(x, z) \wedge (C(x, y) \vee C(y, z)) = (C(x, z) \wedge C(x, y)) \vee (C(x, z) \wedge C(y, z))$$

also  $(x, z) \in (C(x, z) \wedge C(x, y)) \vee (C(x, z) \wedge C(y, z))$  in  $\mathbf{F}$ .

Also gibt es  $p_1, \dots, p_{n-1} \in \mathbf{F}$  mit

$$\begin{aligned} x(C(x, z) \wedge C(x, y))p_1 & && x = p_1(x, y, x) = p_1(x, x, y), \\ p_1(C(x, z) \wedge C(y, z))p_2 & && p_1(x, y, x) = p_2(x, y, x), \\ & && p_1(x, y, y) = p_2(x, y, y), \\ \vdots & && \\ p_{n-1}(C(x, z) \wedge C(y, z))z & && p_{n-1}(x, y, x) = p_{n-1}(x, y, y) = z \end{aligned}$$

“ $\Leftarrow$ ”. Seien  $C_1, C_2, C_3 \in \text{Con}(\mathfrak{A})$  für  $\mathfrak{A} \in \text{HSP}(\mathcal{K})$ . R.Z.Z:

$$\begin{aligned} C_1 \wedge (C_2 \vee C_3) &\subseteq (C_1 \wedge C_2) \vee (C_1 \wedge C_3) \\ &\supseteq \text{gilt in jedem Verband} \end{aligned}$$

Sei  $(a, b) \in C_1 \wedge (C_2 \vee C_3)$ . D.h., es gibt  $c_1, \dots, c_t$  mit

$$aC_2c_1C_3c_2C_2 \cdots c_tC_3b$$

Also für  $i \in \{1, \dots, n\}$ :

$$p_i(a, a, b)C_2p_i(a, c_1, b)C_3p_i(a, c_2, b) \cdots C_3p_i(a, b, b)$$

und wegen  $p_i(a, c, b)C_1p_i(a, c, a) = a$ :

$$p_i(a, a, b)(C_1 \wedge C_2)p_i(a, c_1, b)(C_1 \wedge C_3)p_i(a, c_2, b) \cdots (C_1 \wedge C_3)p_i(a, b, b)$$

also

$$p_i(a, a, b)((C_1 \wedge C_2) \vee (C_1 \wedge C_3))p_i(a, b, b)$$

Gleichungen liefern dann:  $a((C_1 \wedge C_2) \vee (C_1 \wedge C_3))b$ .  $\square$

Jónsson chains have been discovered by Bjarni Jónsson; they provide an equivalent characterisation of congruence distributive varieties. If the variety is generated by the polymorphism clone of a finite structure  $\mathfrak{B}$  with finite relational signature, this condition has drastic consequences for  $\text{CSP}(\mathfrak{B})$ , similarly as in the previous section for congruence modular varieties. Barto [6] proved that in this case  $\mathfrak{B}$  must also have a near-unanimity polymorphism and hence can be solved in polynomial time by the methods that will be presented in Section ???. There are several equivalent definitions of Jónsson chains; we present a variant that follows the terminology in [55].

**Definition 9.11.** A sequence  $j_1, \dots, j_{2n+1}$  of ternary operations on a set  $B$  is called a *chain of Jónsson operations* if for all  $x, y, z \in B$

$$\begin{aligned} j_1(x, x, y) &= x \\ j_i(x, y, x) &= x && \text{for all } i \in \{1, \dots, 2n+1\} \\ j_{2i-1}(x, y, y) &= j_{2i}(x, y, y) && \text{for all } i \in \{1, \dots, n\} \\ j_{2i}(x, x, y) &= j_{2i+1}(x, x, y) && \text{for all } i \in \{1, \dots, n\} \\ j_{2n+1}(x, y, y) &= y. \end{aligned}$$

Note that if  $j_0, j_1, j_2$  is a Jónsson chain, then  $j_1$  is a majority operation. Clearly, the existence of a Jónsson chain is a minor condition, and it is non-trivial, so any clone that contains a Jónsson chain has no minor-preserving map to **Proj**.

A variety is *congruence distributive* if the congruence lattice of every algebra in the variety is distributive. Jónsson [53] showed that this is the case if and only if there exists a chain of terms which denotes a Jónsson chain in all algebras of the variety.

We therefore mention two further strengthenings of chains of quasi Jónsson operations whose impact on the complexity of  $\text{CSP}(\mathfrak{B})$  is also unclear.



**Definition 9.12.** A sequence  $d_1, \dots, d_n$  of ternary operations on a set  $B$  is called a *chain of directed Jónsson operations* if for all  $x, y, z \in B$

$$\begin{aligned} d_1(x, x, y) &= x \\ d_i(x, y, x) &= x && \text{for all } i \in \{1, \dots, n\} \\ d_i(x, y, y) &= d_{i+1}(x, x, y) && \text{for all } i \in \{1, \dots, n-1\} \\ d_n(x, y, y) &= y. \end{aligned}$$

**Proposition 9.13.** *If a clone contains a chain  $d_1, \dots, d_n$  of directed Jónsson operations then it also contains a chain  $j_1, \dots, j_{2n-1}$  of Jónsson operations.*

*Proof.* Define  $j_1(x, y, z) := d_1(x, y, z)$  and for  $i \in \{1, \dots, n-1\}$

$$\begin{aligned} j_{2i}(x, y, z) &:= d_{i+1}(x, x, z) \\ j_{2i+1}(x, y, z) &:= d_{i+1}(x, y, z). \end{aligned}$$

Then

$$\begin{aligned} j_1(x, x, y) &= d_1(x, x, y) = x, \\ j_{2n-1}(x, y, y) &= d_n(x, y, y) = d_n(y, y, y) = y, \end{aligned}$$

and for  $i \in \{1, \dots, n-1\}$ :

$$\begin{aligned} j_{2i}(x, y, x) &= x \\ j_{2i+1}(x, y, x) &= d_{i+1}(x, y, x) = x \\ j_{2i-1}(x, y, y) &= d_i(x, y, y) = d_{i+1}(x, x, y) = y \\ j_{2i}(x, x, y) &= d_{i+1}(x, x, y) = j_{2i+1}(x, x, y). \end{aligned} \quad \square$$

**Proposition 9.14.** *If a clone contains an  $n$ -ary near-unanimity operation  $f$ , then it also contains a chain  $d_1, \dots, d_n$  of directed Jónsson operations.*

*Proof.* For  $i \in \{1, \dots, n\}$  define  $d_i(x, y, z) := f(x, \dots, x, y, z, \dots, z)$  where the argument  $y$  is at position  $n-i+1$ .  $\square$

It has been shown by Kazda, Kozik, McKenzie, and Moore that a clone contains a chain of Jónsson operations if and only if it contains a chain of directed Jónsson operations [55]. A slight variation of the previous definition from [55] yields a much stronger condition.

**Definition 9.15.** A sequence  $p_1, \dots, p_n$  of ternary operations on a set  $B$  is called a *chain of Pixley operations* if for all  $x, y \in B$

$$\begin{aligned} p_1(x, y, y) &= x \\ p_i(x, y, x) &= x && \text{for all } i \in \{1, \dots, n\} \\ p_i(x, x, y) &= p_{i+1}(x, y, y) && \text{for all } i \in \{1, \dots, n-1\} \\ p_n(x, x, y) &= y. \end{aligned}$$

**Proposition 9.16.** *If a clone contains a chain  $p_1, \dots, p_n$  of Pixley operations, then it also contains a chain  $j_1, \dots, j_{2n+1}$  of Jónsson operations.*

*Proof.* Define

$$\begin{aligned}
 j_1(x, y, z) &:= x \\
 j_{2i}(x, y, z) &:= p_i(x, y, z) && \text{for } i \in \{1, \dots, n\} \\
 j_{2i+1}(x, y, z) &:= p_{i+1}(x, z, z) && \text{for } i \in \{1, \dots, n-1\} \\
 j_{2n+1}(x, y, z) &:= z.
 \end{aligned}$$

Then

$$\begin{aligned}
 j_1(x, x, y) &= x, \\
 j_{2n+1}(x, y, y) &= p_n(x, y, y) = y,
 \end{aligned}$$

and for  $i \in \{1, \dots, n\}$ :

$$\begin{aligned}
 j_{2i}(x, y, x) &= p_i(x, y, x) = x \\
 j_{2i+1}(x, y, x) &= x \\
 j_{2i-1}(x, y, y) &= p_i(x, y, y) = j_{2i}(x, y, y) \\
 j_{2i}(x, x, y) &= p_i(x, x, y) = p_{i+1}(x, y, y) = j_{2i+1}(x, x, y). \quad \square
 \end{aligned}$$

Note that if  $p_1, p_2, p_3$  is a chain of Pixley operations of length  $n = 3$ , then  $p_2$  is a Maltsev operation.

TODO: mention finite-signature collapse to NU.

### 9.3 Congruence Modularity

Ein Verband  $\mathbf{V}$  heißt *modular* falls gilt

$$\forall x, y. x \leq y \Rightarrow (x \vee (y \wedge z) = y \wedge (x \vee z))$$

äquivalent (da  $x \leq y \Leftrightarrow (x = x \wedge y)$ ):

$$\forall x, y, z. (x \wedge y) \vee (y \wedge z) = y \wedge ((x \wedge y) \vee z)$$

**Lemma 9.17.** *Distributivität impliziert Modularität.*

Sei  $x \leq y$ .

$$\begin{aligned}
 x \vee (y \wedge z) &= (x \vee y) \wedge (x \vee z) && \text{(D2)} \\
 &= y \wedge (x \vee z) && \text{(da } x \vee y = y)
 \end{aligned}$$

The lattice  $N_5$  (see Figure 13) is not modular:  $a \leq b$ , aber

$$\begin{aligned}
 a \vee (b \wedge c) &= a \\
 &\neq b \wedge (a \vee c)
 \end{aligned}$$

**Theorem 9.18** (Dedekind).  $\mathbf{V}$  ist genau dann modular, wenn es keine Einbettung  $N_5 \rightarrow \mathbf{V}$  gibt.

*Proof.* Angenommen,  $a, b, c \in V$  mit  $a \leq b$  und  $a_1 := a \vee (b \wedge c) < b_1 := b \wedge (a \vee c)$ . Nachrechnen:  $c \wedge b_1 = c \wedge b$  und  $c \vee a_1 = c \vee a$ .  $\square$

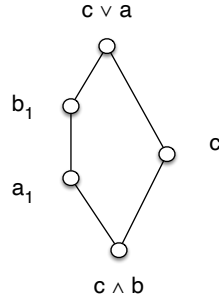


Figure 13: The lattice  $N_5$ .

**Proposition 9.19.** *Wenn  $\mathfrak{A}$  kongruenzpermutierbar, dann ist  $\mathfrak{A}$  kongruenzmodular.*

*Proof.* Seien  $C_1, C_2, C_3 \in \text{Con}(\mathfrak{A})$  mit  $C_1 \subseteq C_2$ . Z.Z.:

$$C_2 \wedge (C_1 \vee C_3) \subseteq C_1 \vee (C_2 \wedge C_3)$$

Sei also  $(a, b) \in C_2 \cap (C_1 \vee C_3)$ .

Kongruenzpermutierbarkeit:  $C_1 \vee C_3 = C_1 \circ C_3$ .

Also gibt es  $c$  mit  $aC_1cC_3b$ .

$(a, c) \in C_2$  da  $C_1 \subseteq C_2$ .

Da  $(a, b) \in C_2$  gilt  $(c, b) \in C_2$  da  $C_2$  Äquivalenzrelation.

Also  $(c, b) \in C_2 \cap C_3$ .

Da  $(a, c) \in C_1$  haben wir  $(a, b) \in C_1 \circ (C_2 \cap C_3) \subseteq C_1 \vee (C_2 \wedge C_3)$ . □

Sei  $\mathcal{K}$  eine Klasse von  $\tau$ -Algebren.

**Theorem 9.20** (Gumm). *Alle Algebren in  $HPS(\mathcal{K})$  sind kongruenzmodular gdw es gibt  $n \in \mathbb{N}$  und  $\tau$ -Terme  $q_0, \dots, q_n, p$  so dass für alle  $i \in \{0, 1, \dots, n\}$*

$$\begin{aligned} \mathcal{K} \models & \forall x, y. q_i(x, y, x) = x \\ & \wedge q_0(x, y, z) = x \\ & \wedge q_i(x, y, y) = q_{i+1}(x, y, y) && \text{für gerade } i \\ & \wedge q_i(x, x, y) = q_{i+1}(x, x, y) && \text{für ungerade } i \\ & \wedge q_n(x, y, y) = p(x, y, y) \\ & \wedge p(x, x, y) = y \end{aligned}$$

TODO: mention finite-signature collapse to edge terms, with forward reference to Section ??.

**Exercises.**

101. Show that  $SH(\mathcal{K}) \subseteq HS(\mathcal{K})$ ,  $PS(\mathcal{K}) \subseteq SP(\mathcal{K})$ , and  $PH(\mathcal{K}) \subseteq HP(\mathcal{K})$ .

102. Let  $f: \mathfrak{A} \rightarrow \prod_{i \in I} \mathfrak{A}_i$  be a homomorphism. Show that

$$\text{Ker}(f) = \bigcap_{i \in I} \text{Ker}(\pi_i \circ f)$$

103. Show that the variety of all lattices is congruence distributive, but not congruence permutable.
104. Show that the variety of Boolean algebras is both congruence permutable and congruence distributive.
105. Show that a lattice satisfies the identities

$$\begin{aligned}\forall x, y, z. x \wedge (y \vee z) &= (x \wedge y) \vee (x \wedge z) \\ \forall x, y, z. x \vee (y \wedge z) &= (x \vee y) \wedge (x \vee z)\end{aligned}$$

if and only if it satisfies one of those identities.

## 9.4 Congruence Semidistributivity

A lattice  $(L; \wedge, \vee)$  is called *meet semidistributive* ( $\text{SD}(\wedge)$ ) if it satisfies

$$\forall a, b, c (a \wedge b = a \wedge c \Rightarrow a \wedge (b \vee c) = a \wedge b).$$

*Join semidistributivity* ( $\text{SD}(\vee)$ ) is defined dually. The smallest lattices satisfying one, but not both, semidistributivity conditions are shown in Figure 14.

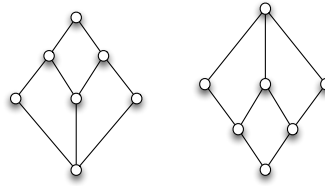


Figure 14: The left diagram shows the lattice  $D_1$  which fails  $\text{SD}(\wedge)$  and the right diagram shows the lattice  $D_2$  which fails  $\text{SD}(\vee)$  (and which is the congruence lattice of  $\mathfrak{S}^2$  where  $\mathfrak{S}$  is the 2-element semilattice).

TODO: forward reference to Section ??.

## 9.5 Abelian Algebras

**Definition 9.21.** An algebra  $\mathbf{A}$  is called *abelian* if  $\Delta_A := \{(a, a) \mid a \in A\}$  is a congruence class of a congruence of  $\mathbf{A}^2$ .

**Example 9.22.** The algebra  $(\{0, \dots, p-1\}; m)$ , where  $m: \{0, \dots, p-1\}^3 \rightarrow \{0, \dots, p-1\}$  is defined by  $m(x, y, z) := x - y + z$ , is abelian, witnessed by the congruence  $\theta$  defined as follows:

$$((x_1, x_2), (y_1, y_2)) \in \theta \Leftrightarrow (x_1 - x_2 = y_1 - y_2). \quad \triangle$$

With the following definition we generalise Example 9.22.

**Definition 9.23.** An algebra  $\mathbf{A}$  is called *affine* if  $\mathbf{A}$  is polynomially equivalent (Definition 9.4) to a module (Example 7.3).

Clearly, every module  $\mathbf{M}$  has a Maltsev term, namely  $t(x, y, z) := x - y + z$ , and  $t^{\mathbf{M}}$  is an affine Maltsev operation as defined in Section 6.1. An algebra whose term operations are generated by an affine Maltsev operation is called an *affine Maltsev algebra*. Note that affine Maltsev algebras are affine in the sense of Definition 9.23.

**Proposition 9.24.** *An algebra  $\mathbf{A}$  is abelian if and only if for every term  $t$  of arity  $k + 1$ , all  $a, b \in A$  and tuples  $c, d \in A^k$*

$$t(a, c) = t(a, d) \Rightarrow t(b, c) = t(b, d).$$

*Proof.* TODO □

**Example 9.25.** We revisit the algebra  $\mathbf{A}$  from Example 9.22. Every term  $t$  in  $\mathbf{A}$  can be written as  $f(x_1, \dots, x_k) = \sum_{i=1}^k ax_i \pmod p$  where  $ka = 1 \pmod p$ . TODO: EXPLAIN. Then clearly if  $\sum_{i=1}^k ax_i = \sum_{i=1}^k ay_i$  and  $x_1 = y_1$ , then  $\sum_{i=2}^k ax_i = \sum_{i=2}^k ay_i$ , which proves the condition given in Proposition 9.24. △

Christian Herrmann [48] showed that if  $\mathbf{A}$  is a finite abelian idempotent algebra in a congruence modular variety, then  $\mathbf{A}$  is affine. We only need the following special case. (This is really nice, but do we *really* need it?)

**Lemma 9.26** (see [61]). *Let  $\mathbf{A}$  be a finite idempotent algebra with a Maltsev term. Then the following are equivalent.*

- $\mathbf{A}$  is abelian.
- If  $p(x, y, z)$  is a Maltsev term in  $\mathbf{A}$ , then  $p^{\mathbf{A}}$  is a homomorphism from  $\mathbf{A}^3 \rightarrow \mathbf{A}$ .
- $\mathbf{A}$  is affine.

*Proof.* Todo, easy. □

### Exercises.

106. Show that subalgebras of affine algebras are affine.

107. Show that an algebra  $\mathbf{A}$  is affine if and only if there exists an abelian group  $(A; +, -)$  such that

- $(x, y, z) \mapsto x - y + z$  is in  $\text{Clo}(\mathbf{A})$ , and
- for all  $a, b, c \in A^n$  we have  $f(a - b + z) = f(a) - f(b) + f(z)$  for every  $f \in \text{Clo}(\mathbf{A})^{(n)}$ .

108. Show that the second item in the previous exercise is equivalent to  $\{(x, y, u, v) \in A^4 \mid x + y = u + v\}$  being a subalgebra of  $\mathbf{A}^4$ .

109. Show that the second item in the Exercise 107 is equivalent to the condition that for every  $f \in \text{Clo}(\mathbf{A})$  there exist endomorphisms  $e_1, \dots, e_n$  of  $(A; +, -)$  and  $a \in A$  such that for all  $x_1, \dots, x_n \in A$

$$f(x_1, \dots, x_n) = \sum_{i=1}^n e_i(x_i) + a.$$

## 10 Absorption

*“The notion of absorption is, in a sense, complementary to abelianness”*  
(Barto and Kozik [9])

Absorption theory is an important topic in universal algebra, developed by Marcin Kozik and Libor Barto, which has powerful applications for the study of homomorphism problems. It can be seen as a tool to show the existence of certain solutions in instances of a CSP.

**Definition 10.1** (Absorbing subalgebras). Let  $\mathbf{A}$  be an algebra and  $f \in \text{Clo}(\mathbf{A})$ . A subalgebra  $\mathbf{B}$  of  $\mathbf{A}$  is called an *absorbing subalgebra of  $\mathbf{A}$  with respect to  $f$* , in symbols  $\mathbf{B} \triangleleft_f \mathbf{A}$ , if  $f(b_1, \dots, b_n) \in B$  whenever all but at most one out of  $b_1, \dots, b_n$  are from  $B$ . If such an  $f$  exists we say that  $\mathbf{B}$  *absorbs*  $\mathbf{A}$ , and write  $\mathbf{B} \triangleleft \mathbf{A}$ .

Since subalgebras are uniquely determined by their domain, we also use the notation  $B \triangleleft \mathbf{A}$  if  $B$  is the domain of an absorbing subalgebra  $\mathbf{B}$  of  $\mathbf{A}$ .

**Example 10.2.** If  $\mathbf{A} = (\{0, 1\}; \text{majority})$  then both  $\{0\} \triangleleft \mathbf{A}$  and  $\{1\} \triangleleft \mathbf{A}$ . If  $\mathbf{A} = (\{0, 1\}; \text{min})$  then  $\{0\} \triangleleft \mathbf{A}$ . △

In any algebra  $\mathbf{A}$  with a near-unanimity term  $t$ , every one-element subalgebra is absorbing with respect to  $t^{\mathbf{A}}$ . Following the presentation in [9], we will prove a converse to this statement; in the proof, we need a useful definition.

**Definition 10.3.** If  $s, t$  are operations (or terms) of arity  $p$  and  $q$ , respectively, then the *star composition* of  $s$  and  $t$  is the operation  $s * t$  (or term) of arity  $pq$  defined by

$$(x_{1,1}, \dots, x_{p,q}) \mapsto s(t(x_{1,1}, \dots, x_{1,q}), \dots, t(x_{p,1}, \dots, x_{p,q})).$$

**Proposition 10.4.** *Let  $\mathbf{A}$  be a finite algebra. If every one-element subset is the domain of an absorbing subalgebra of  $\mathbf{A}$ , then  $\mathbf{A}$  has a near-unanimity term.*

*Proof.* If  $\mathbf{B} \triangleleft_f \mathbf{A}$  and  $\mathbf{C} \triangleleft_g \mathbf{A}$ , where  $f$  is  $n$ -ary and  $g$  is  $m$ -ary, then  $\mathbf{B} \triangleleft_{f*g} \mathbf{A}$  and  $\mathbf{C} \triangleleft_{f*g} \mathbf{A}$ . Since  $\mathbf{A}$  is finite we can thus construct a single term operation  $h$  such that  $\mathbf{B} \triangleleft_h \mathbf{A}$  for every one-element subalgebra  $\mathbf{B}$ . But then  $h$  must be a near unanimity term. □

### 10.1 Absorption Transfer

We start with some warm-up exercises concerning absorption.

**Lemma 10.5.** *If  $\mathbf{C} \triangleleft \mathbf{B} \triangleleft \mathbf{A}$  then  $\mathbf{C} \triangleleft \mathbf{A}$ .*

*Proof.* If  $\mathbf{B}$  absorbs  $\mathbf{A}$  with respect to an operation  $t \in \text{Clo}(\mathbf{A})$  of arity  $n$  and  $\mathbf{C}$  absorbs  $\mathbf{B}$  with respect to an operation  $s \in \text{Clo}(\mathbf{B})$ , then  $\mathbf{A}$  absorbs  $\mathbf{C}$  with respect to  $s * t$ . □

**Corollary 10.6.** *If  $\mathbf{B} \triangleleft \mathbf{A}$  and  $\mathbf{C} \triangleleft \mathbf{A}$  then  $B \cap C \triangleleft \mathbf{A}$ .*

*Proof.* Note that  $B \cap C \triangleleft \mathbf{C}$  with respect to the same term as  $\mathbf{B} \triangleleft \mathbf{A}$ . Now the statement follows from Lemma 10.5. □

Let  $\mathbf{A}$  and  $\mathbf{B}$  be two algebras of the same signature, and let  $R$  be the domain of a subalgebra of  $\mathbf{A} \times \mathbf{B}$ . For  $X \subseteq A$  and  $Y \subseteq B$  we define the *neighborhoods* of  $X \subseteq A$  and  $Y \subseteq B$

$$\begin{aligned} X^{+R} &:= \{b \in B \mid \exists a \in X : R(a, b)\} \\ Y^{-R} &:= \{a \in A \mid \exists b \in Y : R(a, b)\}. \end{aligned}$$

**Lemma 10.7.** *Let  $R$  be a subalgebra of  $\mathbf{A} \times \mathbf{B}$ . If  $X \leq \mathbf{A}$  and  $Y \leq \mathbf{B}$  then  $X^+ \leq \mathbf{A}$  and  $Y^- \leq \mathbf{B}$ . If  $R \leq_S \mathbf{A} \times \mathbf{B}$  and  $X \triangleleft \mathbf{A}$  and  $Y \triangleleft \mathbf{B}$ , then  $X^{+R} \triangleleft \mathbf{B}$  and  $Y^- \triangleleft \mathbf{A}$ .*

*Proof.* □

**Lemma 10.8.** *Let  $\sim$  be a congruence of  $\mathbf{A}$  and suppose that  $\mathbf{B} \triangleleft (\mathbf{A}/\sim)$ . Then  $\bigcup_{b \in B} b/\sim \triangleleft \mathbf{A}$ .*

For the following lemma that generalises the previous two results, we pass to a presentation using relational structure  $\mathfrak{B}$ . Recall that if  $\mathbf{A}$  is an algebra such that  $\text{Clo}(\mathbf{A}) = \text{Pol}(\mathfrak{B})$ , then  $R \subseteq A^n$  is primitive positive definable in  $\mathfrak{B}$  if and only if  $R$  is the domain of a subalgebra of  $\mathbf{A}^n$ .

**Lemma 10.9.** *Let  $\mathfrak{A}$  be a finite relational  $\tau$ -structure and  $\phi$  a primitive positive  $\tau$ -formula. Let  $\mathfrak{A}'$  be a  $\tau$ -structure on the same domain such that for each  $R \in \tau$  we have  $R^{\mathfrak{A}'} \triangleleft R^{\mathfrak{A}}$ . If  $\phi$  defines  $S$  in  $\mathfrak{A}$  and defines  $S'$  in  $\mathfrak{A}'$ , then  $S' \triangleleft S$ .*

*Proof.* Todo. □

## 10.2 The Absorption Theorem

Let  $\mathbf{A}$  and  $\mathbf{B}$  be algebras, and let  $R \subseteq A \times B$  be a relation. We view  $R$  as the edge relation of a bipartite graph with color classes  $A$  and  $B$ . We say that  $R$  is *linked* if this graph is connected after removing isolated vertices. Note that when  $R$  is a subdirect subalgebra of  $\mathbf{A} \times \mathbf{B}$  then  $R$  has no isolated vertices.

Let  $\mathbf{A}$  and  $\mathbf{B}$  be  $\tau$ -algebras. Then a subalgebra  $\mathbf{R}$  of  $\mathbf{A} \times \mathbf{B}$  is called *subdirect* if

$$\begin{aligned} \{a \in A \mid \text{there exists } b \in B \text{ such that } (a, b) \in R\} &= A \\ \{b \in B \mid \text{there exists } a \in A \text{ such that } (a, b) \in R\} &= B. \end{aligned}$$

**Theorem 10.10** (Absorption theorem [8]). *Let  $\mathbf{C}$  be a finite algebra. Then  $\mathbf{C}$  has a Taylor term if and only if for all  $\mathbf{A}, \mathbf{B} \in \text{HSP}(\mathbf{C})$  and any linked  $R \leq_S \mathbf{A} \times \mathbf{B}$ :*

- $R = A \times B$ ;
- $\mathbf{A}$  has a proper absorbing subuniverse;
- $\mathbf{B}$  has a proper absorbing subuniverse.

## 10.3 Hereditarily Absorption-Free Algebras

An algebra  $\mathbf{A}$  is called *hereditarily absorption-free* if no subalgebra of  $\mathbf{A}$  has a proper absorbing subalgebra, i.e., whenever  $\mathbf{C}$  is an absorbing subalgebra of an subalgebra  $\mathbf{B}$  of  $\mathbf{A}$ , then  $C = B$ .

**Corollary 10.11** (Theorem 1.4 in [11]). *Let  $\mathbf{A}$  be a finite idempotent algebra with a Taylor term. If  $\mathbf{A}$  is hereditarily absorption-free, then  $\mathbf{A}$  has a Maltsev term.*

*Proof.* Let  $\mathbf{F} \in \text{HSP}^{\text{fin}}(\mathbf{A})$  be the free algebra on two generators  $x, y$ . We first show that  $\mathbf{F}$  is hereditarily absorption-free. TODO

Let  $\mathbf{R}$  be the subalgebra of  $\mathbf{F}^2$  generated by  $(x, y)$ ,  $(x, x)$ , and  $(y, x)$ . Every element of  $F$  can be written as  $t^{\mathbf{F}}(x, y)$ , and since  $(x, y) \in R$  and  $(y, x) \in R$  we have that  $(t^{\mathbf{F}}(x, y), t^{\mathbf{F}}(y, x)) \in R$ , showing that  $\mathbf{R}$  is a subdirect subalgebra of  $\mathbf{F}^2$ .

Every element of  $R$  can be written as  $s^{\mathbf{R}}((x, y), (x, x), (y, x)) = (s^{\mathbf{F}}(x, x, y), s^{\mathbf{F}}(y, x, x))$  for some term  $s$ . Since  $(x, x), (y, x) \in R$  we have  $(s^{\mathbf{F}}(x, x, y), s^{\mathbf{F}}(x, x, x)) \in R$  and since  $(x, y), (x, x) \in R$  we have  $(s^{\mathbf{F}}(x, x, x), s^{\mathbf{F}}(y, x, x)) \in R$ , and hence  $R$  is linked.

Since  $\mathbf{F}$  has no proper absorbing subalgebra, Theorem 10.10 implies that  $R = F \times F$ . Let  $m$  be a term such that  $m((x, y), (x, x), (y, x)) = (y, y)$ . Then  $m^{\mathbf{A}}$  is a Maltsev operation.  $\square$

The following is a special case of Lemma 4.1 in [11].

**Lemma 10.12.** *Let  $\mathbf{A}$  be a finite idempotent algebra. If  $\mathbf{A}$  is abelian then  $\mathbf{A}$  is hereditarily absorption-free.*

*Proof.* An easy short proof exists for the special case that abelian algebras cannot have 1-element absorbing subalgebras [9].  $\square$

**Corollary 10.13.** *Let  $\mathbf{A}$  be a finite idempotent algebra with a Taylor term. If  $\mathbf{A}$  is abelian then  $\mathbf{A}$  is affine.*

*Proof.* If  $\mathbf{A}$  is abelian, then by Lemma 10.12 it is hereditarily absorption-free. Since  $\mathbf{A}$  has a Taylor term, Corollary 10.11 implies that  $\mathbf{A}$  has a Maltsev term  $m$ . Lemma 9.26 implies that  $\mathbf{A}$  is affine.  $\square$

## 10.4 $n$ -Absorption

We define a refinement of absorption which takes the arity of the function into account.

**Definition 10.14.** Let  $\mathbf{A}$  be an algebra and  $B \subseteq A$ . Then  $B$  is called  $n$ -absorbing if there exists a term  $t$  such that  $t^{\mathbf{A}}(a) \in B$  whenever  $a \in A^n$  and  $|\{i \mid a_i \in B\}| \geq n - 1$ . If  $B$  is the domain of a subalgebra  $\mathbf{B}$  of  $\mathbf{A}$ , we write  $\mathbf{B} \triangleleft_n \mathbf{A}$ .

Clearly,  $\mathbf{B} \triangleleft \mathbf{A}$  (according to Definition 10.1) if there exists an  $n \in \mathbb{N}$  such that  $\mathbf{B} \triangleleft_n \mathbf{A}$ .

## 10.5 Near Unanimity Polymorphisms

Equivalence of  $k$ -decomposability and existence of  $(k + 1)$ -ary near unanimities.

# 11 Cyclic Polymorphisms

In this section, for several results that we present we do not give a proof.

## 11.1 Cyclic Terms

An operation  $c: A^n \rightarrow A$ , for  $n \geq 2$ , is *cyclic* if it satisfies for all  $a_1, \dots, a_n \in A$  that  $c(a_1, \dots, a_n) = c(a_2, \dots, a_n, a_1)$ . Cyclic operations are weak near unanimity operations, and in particular Taylor operations. Conversely, a deep result of Barto and Kozic (Theorem 11.5 below) implies that every Taylor operation on a finite set generates a cyclic operation.



We start with some easy but useful observations about cyclic terms. The *cyclic composition*  $s \circ t$  of  $s$  and  $t$  is the operation (or term) of arity  $q$  defined by

$$(x_1, \dots, x_q) \mapsto s(t(x_1, \dots, x_q), t(x_2, \dots, x_q, x_1), \dots).$$

The following is straightforward.

**Lemma 11.1.** *Let  $s: A^k \rightarrow A$  and  $t: A^l \rightarrow A$  be operations.*

- *If  $s$  and  $t$  are cyclic operations, then their star product  $s * t$  (Definition 10.3) is cyclic.*
- *If  $s$  is arbitrary and  $t$  is cyclic then  $s \circ t$  is cyclic.*
- *If  $s$  is cyclic,  $t$  is arbitrary, and  $l$  divides  $k$  then  $s \circ t$  is cyclic.*

**Exercises.**

110. Show that if  $\mathbf{A} = (\{0, 1\}; \min)$  and  $f \in \text{Clo}(\mathbf{A})^{(k)}$  is cyclic, then

$$f(x_1, \dots, x_k) = \min(x_1, \dots, x_k).$$

111. Suppose that  $\mathbf{A} = (\{0, 1\}; \text{majority})$  and  $f \in \text{Clo}(\mathbf{A})^{(k)}$  is cyclic. Show that

- $k \geq 3$ ;
- if  $r > k/2$  and  $c \in A^k$  is such that  $c_i = a$  for  $i \leq r$  and  $c_i = b$  otherwise, then  $f(c) = a$ ;
- if  $r, s, t$  are such that  $r + s > t$ ,  $s + t > r$ , and  $t + r > s$ , then the function

$$(x, y, z) \mapsto f(\underbrace{x, \dots, x}_r, \underbrace{y, \dots, y}_s, \underbrace{z, \dots, z}_t) \quad (11)$$

is the ternary majority operation on  $\{0, 1\}$ .

112. Suppose that  $p$  is a prime and  $\mathbf{A} = (\{0, \dots, p-1\}; m)$  where  $m: A^3 \rightarrow A$  is given by  $m(x, y, z) = x - y + z \pmod p$  and that  $f \in \text{Clo}(\mathbf{A})^{(k)}$  is cyclic. Show that if  $r, s, t$  are such that  $r = t = k \pmod p$  and  $s = -k \pmod p$ , then the ternary function defined in (11) equals  $x - y + z \pmod p$ .

113. Does previous exercise remain true if we drop the assumption that  $p$  is prime?

## 11.2 Cyclic Relations

When  $a = (a_0, a_1, \dots, a_{k-1})$  is a  $k$ -tuple, we write  $\rho(a)$  for the  $k$ -tuple  $(a_1, \dots, a_{k-1}, a_0)$ .

**Definition 11.2.** An  $n$ -ary relation  $R$  on a set  $A$  is called *cyclic* if for all  $a \in A^k$

$$a \in R \Rightarrow \rho(a) \in R.$$

**Lemma 11.3** (from [8]). *A finite idempotent algebra  $\mathbf{A}$  has a  $k$ -ary cyclic term if and only if every nonempty cyclic subalgebra of  $\mathbf{A}^k$  contains a constant tuple.*

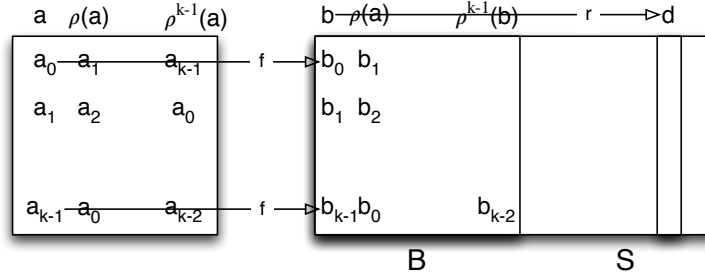


Figure 15: Diagram for the proof of Lemma 11.3.

*Proof.* Let  $\tau$  be the signature of  $\mathbf{A}$ . For the easy direction, suppose that  $\mathbf{A}$  has a cyclic  $\tau$ -term  $t(x_1, \dots, x_k)$ . Let  $a = (a_0, a_1, \dots, a_{k-1})$  be an arbitrary tuple in a cyclic subalgebra  $\mathbf{R}$  of  $\mathbf{A}^k$ . As  $R$  is cyclic,  $\rho(a), \dots, \rho^{k-1}(a) \in R$ , and since  $\mathbf{R}$  is a subalgebra,  $b := t^{\mathbf{A}^k}(a, \rho(a), \dots, \rho^{k-1}(a)) \in R$ . Since  $t$  is cyclic, the  $k$ -tuple  $b$  is constant.

To prove the converse direction, we assume that every nonempty cyclic subalgebra of  $\mathbf{A}^k$  contains a constant tuple. For a  $\tau$ -term  $f(x_0, x_1, \dots, x_{k-1})$ , let  $S(f)$  be the set of all  $a \in A^k$  such that  $f^{\mathbf{A}}(a) = f^{\mathbf{A}}(\rho(a)) = \dots = f^{\mathbf{A}}(\rho^{k-1}(a))$ . Choose  $f$  such that  $|S(f)|$  is maximal (here we use the assumption that  $A$  is finite). If  $|S(f)| = |A^k|$ , then  $f^{\mathbf{A}}$  is cyclic and we are done. Otherwise, arbitrarily pick  $a = (a_0, a_1, \dots, a_{k-1}) \in A^k \setminus S(f)$ . For  $i \in \{0, \dots, k-1\}$ , define  $b_i := f(\rho^i(a))$ , and let  $B := \{b, \rho(b), \dots, \rho^{k-1}(b)\}$ .

We claim that the smallest subalgebra  $\mathbf{C}$  of  $\mathbf{A}^k$  that contains  $B$  is cyclic. So let  $c \in C$  be arbitrary. Since  $\mathbf{C}$  is generated by  $B$ , there exists a  $\tau$ -term  $s(x_0, x_1, \dots, x_{k-1})$  such that  $c = s^{\mathbf{A}}(b, \rho(b), \dots, \rho^{k-1}(b))$ . Then  $\rho(c) = s^{\mathbf{A}}(\rho(b), \rho^2(b), \dots, \rho^{k-1}(b), b) \in C$ , proving the claim.

According to our assumption, the algebra  $\mathbf{C}$  contains a constant tuple  $d$ . Then there exists a  $\tau$ -term  $r(x_0, \dots, x_{k-1})$  such that  $d = r^{\mathbf{C}}(b, \rho(b), \dots, \rho^{k-1}(b))$ . Note that

$$r^{\mathbf{A}}(b_0) = r^{\mathbf{A}}(b_1) = \dots = r^{\mathbf{A}}(b_{k-1})$$

since  $d$  is constant (also see Figure 15). It follows that  $b \in S(r)$ .

Now consider the  $\tau$ -term  $t(x_0, x_1, \dots, x_{k-1})$  defined by

$$t(x) := r(f(x), f(\rho(x)), \dots, f(\rho^{k-1}(x))).$$

where  $x := (x_0, x_1, \dots, x_{k-1})$ . We claim that  $S(f) \subseteq S(t)$ , but also that  $a \in S(t)$ . This is clearly in contradiction to the maximality of  $|S(f)|$ . Let  $e \in S(f)$ . Then

$$\begin{aligned} t^{\mathbf{A}}(\rho^i(e)) &= r^{\mathbf{A}}(f^{\mathbf{A}}(\rho^i(e)), f^{\mathbf{A}}(\rho^{i+1}(e)), \dots, f^{\mathbf{A}}(\rho^{i-1}(e))) \\ &= r^{\mathbf{A}}(f^{\mathbf{A}}(e), f^{\mathbf{A}}(e), \dots, f^{\mathbf{A}}(e)) && \text{(since } e \in S(f)) \\ &= f^{\mathbf{A}}(e) && \text{(since } \mathbf{A} \text{ is idempotent)} \end{aligned}$$

for all  $i \in \{0, \dots, k-1\}$ . Therefore,  $e \in S(t)$ . On the other hand,

$$\begin{aligned} t^{\mathbf{A}}(\rho^i(a)) &= r^{\mathbf{A}}(f^{\mathbf{A}}(\rho^i(a)), f^{\mathbf{A}}(\rho^{i+1}(a)), \dots, f^{\mathbf{A}}(\rho^{i-1}(a))) \\ &= r^{\mathbf{A}}(b_i, b_{i+1}, \dots, b_{i-1}) \\ &= r^{\mathbf{A}}(\rho^i(b)) \end{aligned}$$

which is constant for all  $i$  by the choice of  $r$ . Therefore,  $a \in S(t)$  and the contradiction is established.  $\square$

### 11.3 Digraphs without Sources and Sinks

A *source* in a digraph is a vertex with no incoming edges, and a *sink* is a vertex with no outgoing edges. In this section we mention an important result about finite digraphs with no sources and no sinks. Note that undirected graphs  $(V, E)$ , viewed as directed graphs where for every  $\{u, v\} \in E$  we have  $(u, v) \in E$  and  $(v, u) \in E$ , are examples of such graphs.

**Theorem 11.4** (Barto, Kozik, Nieven [10]). *Let  $H$  be a digraph without sources and sinks. If  $H$  has a Taylor polymorphism, then  $H$  is homomorphically equivalent to a disjoint union of cycles.*

A proof of this deep theorem is out of the scope of this course. An important part of the proof is using *absorption theory*, which has been developed for this theorem.

If a graph  $H$  is homomorphically equivalent to a disjoint union of cycles, then  $\text{CSP}(H)$  is in P (e.g., we can use the algorithm  $\text{PC}_H$  to solve it; see Section 3). On the other hand, a digraph without a Taylor polymorphism has an NP-hard CSP. Therefore, Theorem 11.4 shows that the Feder-Vardi conjecture is true for digraphs without sources and sinks: their CSPs are in P or NP-complete.

### 11.4 The Cyclic Term Theorem

We present an improvement of the result in Section 8.2.

**Theorem 11.5** (of [8]). *Let  $\mathbf{A}$  be a finite algebra. Then the following are equivalent.*

- $\mathbf{A}$  has a Taylor term;
- $\mathbf{A}$  has a cyclic term;
- for all prime numbers  $p > |A|$ , the algebra  $\mathbf{A}$  has a  $p$ -ary cyclic term.

*Proof.* The statement can be found for finite *idempotent* algebras in [8], but since a structure has a Taylor polymorphism if and only if its core does (recall that in this text we do not require that Taylor operations are idempotent), and since a core has a Taylor polymorphism if and only if it has an idempotent Taylor polymorphism, the idempotent case implies the statement as given in the theorem. Again, the full proof of this theorem is out of the scope of our course. We mention that Barto and Kozik use absorption theory [8] and explicitly use a strong version of Theorem 11.4.  $\square$

As an application, we derive the classification of the complexity of  $H$ -colouring for finite undirected graphs  $H$  (Theorem 1.5).

*Proof.* If the core  $G$  of  $H$  equals  $K_2$  or has just one vertex, then  $\text{CSP}(H)$  can be solved in polynomial time, e.g. by the Path Consistency Procedure, Section 3. Otherwise,  $G$  is not bipartite and there exists a cycle  $a_0, a_1, \dots, a_{2k}, a_0$  of odd length in  $H$ . If  $H$  has no Taylor polymorphism, then by Corollary 7.34,  $\text{CSP}(H)$  is NP-hard.

Otherwise, if  $H$  has a Taylor polymorphism, then Theorem 11.5 asserts that there exists a  $p$ -ary cyclic polymorphism  $c$  of  $H$  where  $p$  is a prime number greater than  $\max\{2k, |A|\}$ .

Since the edges in  $H$  are undirected, we can also find a cycle  $a_0, a_1, \dots, a_{p-1}, a_0$  in  $H$ . Then  $c(a_0, a_1, \dots, a_{p-1}) = c(a_1, \dots, a_{p-1}, a_0)$ , which implies that  $H$  contains a loop, a contradiction to the assumption that the core of  $H$  has more than one element.  $\square$

### Exercises.

114. Show that if  $\mathbf{A}$  has a cyclic term and  $\mathbf{B}$  has a cyclic term, then  $\mathbf{A} \times \mathbf{B}$  has a cyclic term.

## 11.5 Siggers Operations of Arity 4

Interestingly, whether a finite algebra has a Taylor term (equivalently: a weak near unanimity term, or a cyclic term) can be tested by searching for a single 4-ary term  $s$  that satisfies

$$\mathbf{A} \models \forall x, y, z. s(x, y, z, y) = s(y, z, x, x),$$

a so-called *4-ary Siggers term*. In particular, the question whether a given finite structure has a cyclic term is in the complexity class NP. Siggers originally found a 6-ary term (see Section 8.2), which has been improved later to the 4-ary term given above. The proof given below is from [56], where the authors also prove that the result is optimal in the sense that there is no equivalent characterisation using a single ternary Taylor term.

**Theorem 11.6** (due to [69]; see [56]). *Let  $\mathbf{A}$  be a finite algebra. Then  $\mathbf{A}$  has a cyclic term if and only if  $\mathbf{A}$  has a 4-ary Siggers term.*

*Proof.* Suppose that  $\mathbf{A}$  has a cyclic term. Let  $p = 3k + 2$  be some prime number larger than  $|A|$  and let  $c(x_1, \dots, x_p)$  be a cyclic term of  $\mathbf{A}$ , which exists by Theorem 11.5. Define  $s(x, y, z, w)$  to be the term

$$c(x, x, \dots, x, y, \dots, y, w, z, \dots, z),$$

where the variable  $x$  occurs  $k + 1$  times, the variable  $w$  occurs once, and the variables  $y$  and  $z$  occur  $k$  times each. Then

$$\begin{aligned} s(x, y, z, y) &= c(x, x, \dots, x, y, y, \dots, y, y, z, \dots, z) \quad (k + 1 \text{ times } x, k + 1 \text{ times } y, k \text{ times } z) \\ &= c(y, y, \dots, y, z, z, \dots, z, x, x, \dots, x) \quad (\text{by cyclicity of } c) \\ &= s(y, z, x, x). \end{aligned}$$

Conversely, a Siggers term is a Taylor term, and therefore the other direction follows from Theorem 11.5.  $\square$

## 12 Open Problems

The Feder and Vardi dichotomy conjecture [40] has been the outstanding open problem in the field; it was solved in 2017 by Bulatov [24] and, independently, by Zhuk [71]. However, there are many interesting problems in the field that are still left open. We start with open problems where all the relevant concepts have already been introduced in the course.

1. Is the class of all finite structures, ordered by pp-constructability and factored by the respective equivalence relation, a lattice [17–19]? Is it countably infinite or uncountably infinite? Are there infinite ascending chains?

2. What is the computational complexity of determining whether a given finite core structure  $H$  has tree duality? Is this problem in P? Is it in P if  $H$  is a digraph or even an orientation of a tree?
3. Is there a polynomial-time algorithm to determine whether a given core structure  $H$  has a Siggers polymorphism? Is this true for the special case where  $H$  is a digraph or an orientation of a tree?
4. (Bulin [28]) Is it true that the CSP of an orientation of a tree is in P if and only if it can be solved by Datalog?

Here come some open problems that require knowledge of concepts that have not been covered in this course; however, references are provided where these concepts are defined formally.

1. Prove that a finite-domain CSP is in P if and only if it can be expressed in *Choiceless Polynomial Time* [14].
2. Is it true that if  $\text{CSP}(H)$  is in NL, then  $\text{CSP}(H)$  is in linear Datalog (Dalmau [33])? Is this at least true for digraphs  $H$ ? The same question would already be interesting for orientations of trees.
3. (Egri-Larose-Tesson [36]) Is it true that if  $\text{CSP}(H)$  is in L, then  $\text{CSP}(H)$  is in symmetric Datalog? Is this at least true for digraphs  $H$ ? It would already be interesting for orientations of trees.
4. (Larose-Tesson [58]) Is it true that if the polymorphism algebra of  $H$  generates a congruence join-semidistributive variety, then  $\text{CSP}(H)$  is in linear Datalog? Is this at least true for digraphs  $H$ ? It would already be interesting for orientations of trees.
5. Is it true that if  $\text{CSP}(H)$  is not P-hard under logspace reductions, then it is in NC? It is known that NC is closed under logspace reductions, and it is believed that P is different from NP. Moreover, the CSP for the structure  $(\{0, 1\}; \{0, 1\}^3 \setminus \{(1, 1, 0)\}, \{0\}, \{1\})$  is P-hard (see Exercise 85). Is it true that if  $\text{CSP}(H)$  does not pp-construct this structure then  $\text{CSP}(H)$  is in NC?

For CSPs over infinite domains, there are numerous open problems, and I invite the reader to have a look at [16].

## References

- [1] S. Aaronson.  $P \stackrel{?}{=} NP$ . *Electronic Colloquium on Computational Complexity (ECCC)*, 24:4, 2017.
- [2] G. Ahlbrandt and M. Ziegler. Quasi-finitely axiomatizable totally categorical theories. *Annals of Pure and Applied Logic*, 30(1):63–82, 1986.
- [3] B. Aspvall, M. F. Plass, and R. E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3):121–123, 1979.

- [4] A. Atserias, A. A. Bulatov, and A. Dawar. Affine systems of equations and counting infinitary logic. *Theoretical Computer Science*, 410(18):1666–1683, 2009.
- [5] L. Barto. The dichotomy for conservative constraint satisfaction problems revisited. In *Proceedings of the Symposium on Logic in Computer Science (LICS)*, Toronto, Canada, 2011.
- [6] L. Barto. Finitely related algebras in congruence distributive varieties have near unanimity terms. *Canadian Journal of Mathematics*, 65(1):3–21, 2013.
- [7] L. Barto and M. Kozik. Constraint satisfaction problems of bounded width. In *Proceedings of the Annual Symposium on Foundations of Computer Science (FOCS)*, pages 595–603, 2009.
- [8] L. Barto and M. Kozik. Absorbing subalgebras, cyclic terms and the constraint satisfaction problem. *Logical Methods in Computer Science*, 8/1(07):1–26, 2012.
- [9] L. Barto and M. Kozik. Absorption in universal algebra and CSP. In *The Constraint Satisfaction Problem: Complexity and Approximability*, volume 7 of *Dagstuhl Follow-Ups*, pages 45–77, 2017.
- [10] L. Barto, M. Kozik, and T. Niven. The CSP dichotomy holds for digraphs with no sources and no sinks (a positive answer to a conjecture of Bang-Jensen and Hell). *SIAM Journal on Computing*, 38(5), 2009.
- [11] L. Barto, M. Kozik, and D. Stanovský. Mal'tsev conditions, lack of absorption, and solvability. *Algebra Universalis*, 74:185–206, 2015.
- [12] L. Barto, J. Opršal, and M. Pinsker. The wonderland of reflections. *Israel Journal of Mathematics*, 223(1):363–398, 2018.
- [13] G. Birkhoff. On the structure of abstract algebras. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31(4):433–454, 1935.
- [14] A. Blass, Y. Gurevich, and S. Shelah. Choiceless polynomial time. *Annals of Pure and Applied Logic*, 100(1-3):141–187, 1999.
- [15] M. Bodirsky. Diskrete Strukturen, 2020. Skript zur Vorlesung, TU Dresden.
- [16] M. Bodirsky. *Complexity of Infinite-Domain Constraint Satisfaction*. Lecture Notes in Logic (52). Cambridge University Press, 2021.
- [17] M. Bodirsky and F. Starke. Maximal digraphs with respect to primitive positive constructability. *Submitted*, 2021.
- [18] M. Bodirsky, F. Starke, and A. Vucaj. Smooth digraphs modulo primitive positive constructability. *International Journal on Algebra and Computation (to appear)*, 2021. Preprint available at ArXiv:1906.05699.
- [19] M. Bodirsky and A. Vucaj. Two-element structures modulo primitive positive constructability. *Algebra Universalis*, 81(20), 2020. Preprint available at ArXiv:1905.12333.

- [20] V. G. Bodnarčuk, L. A. Kalužnin, V. N. Kotov, and B. A. Romov. Galois theory for Post algebras, part I and II. *Cybernetics*, 5:243–539, 1969.
- [21] A. A. Bulatov. Tractable conservative constraint satisfaction problems. In *Proceedings of the Symposium on Logic in Computer Science (LICS)*, pages 321–330, Ottawa, Canada, 2003.
- [22] A. A. Bulatov. H-coloring dichotomy revisited. *Theoretical Computer Science*, 349(1):31–39, 2005.
- [23] A. A. Bulatov. Conservative constraint satisfaction re-visited. *Journal Computer and System Sciences*, 82(2):347–356, 2016. ArXiv:1408.3690.
- [24] A. A. Bulatov. A dichotomy theorem for nonuniform CSPs. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17*, pages 319–330, 2017.
- [25] A. A. Bulatov and V. Dalmau. A simple algorithm for Mal’tsev constraints. *SIAM Journal on Computing*, 36(1):16–27, 2006.
- [26] A. A. Bulatov and P. Jeavons. Algebraic structures in combinatorial problems. Technical report MATH-AL-4-2001, Technische Universität Dresden, 2001.
- [27] A. A. Bulatov, A. A. Krokhin, and P. G. Jeavons. Classifying the complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34:720–742, 2005.
- [28] J. Bulin. On the complexity of  $H$ -coloring for special oriented trees. *Eur. J. Comb.*, 69:54–75, 2018.
- [29] S. N. Burris and H. P. Sankappanavar. *A Course in Universal Algebra*. Springer Verlag, Berlin, 1981.
- [30] C. Carvalho, L. Egri, M. Jackson, and T. Niven. On Maltsev digraphs. *Electr. J. Comb.*, 22(1):P1.47, 2015.
- [31] H. Chen and B. Larose. Asking the metaquestions in constraint tractability. *TOCT*, 9(3):11:1–11:27, 2017.
- [32] D. A. Cohen, M. C. Cooper, P. G. Jeavons, and S. Živný. Binarisation via dualisation for valued constraints. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 3731–3737, 2015.
- [33] V. Dalmau. Linear datalog and bounded path duality of relational structures. *Logical Methods in Computer Science*, 1(1), 2005.
- [34] V. Dalmau and J. Pearson. Closure functions and width 1 problems. In *Proceedings of the International Conference on Principles and Practice of Constraint Programming (CP)*, pages 159–173, 1999.
- [35] R. Dechter. *Constraint Processing*. Morgan Kaufmann, 2003.

- [36] L. Egri, B. Larose, and P. Tesson. Symmetric datalog and constraint satisfaction problems in logspace. In *Proceedings of the Symposium on Logic in Computer Science (LICS)*, pages 193–202, 2007.
- [37] M. M. El-Zahar and N. Sauer. The chromatic number of the product of two 4-chromatic graphs is 4. *Combinatorica*, 5(2):121–126, 1985.
- [38] T. Feder. Classification of homomorphisms to oriented cycles and of  $k$ -partite satisfiability. *SIAM Journal on Discrete Mathematics*, 14(4):471–480, 2001.
- [39] T. Feder and M. Y. Vardi. Monotone monadic SNP and constraint satisfaction. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 612 – 622, 1993.
- [40] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM Journal on Computing*, 28:57–104, 1999.
- [41] J. Fischer. CSPs of orientations of trees. Master thesis, TU Dresden, 2015.
- [42] M. Garey and D. Johnson. *A guide to NP-completeness*. CSLI Press, Stanford, 1978.
- [43] D. Geiger. Closed systems of functions and predicates. *Pacific Journal of Mathematics*, 27:95–100, 1968.
- [44] G. H. Hardy and E. M. Wright. *An introduction to the theory of numbers*. Oxford University Press, 2008. Sixth edition.
- [45] P. Hell and J. Nešetřil. On the complexity of H-coloring. *Journal of Combinatorial Theory, Series B*, 48:92–110, 1990.
- [46] P. Hell and J. Nešetřil. The core of a graph. *Discrete Mathematics*, 109:117–126, 1992.
- [47] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, Oxford, 2004.
- [48] C. Herrmann. Affine algebras in congruence modular varieties. *Acta Sci. Math. (Szeged)*, 41(1-2):119–125, 1979.
- [49] D. Hobby and R. McKenzie. *The structure of finite algebras*, volume 76 of *Contemporary Mathematics*. American Mathematical Society, 1988.
- [50] W. Hodges. *Model theory*. Cambridge University Press, Cambridge, 1993.
- [51] W. Hodges. *A shorter model theory*. Cambridge University Press, Cambridge, 1997.
- [52] P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.
- [53] B. Jónsson. Algebras whose congruence lattices are distributive. *Mathematica Scandinavica*, 21:110–121, 1967.
- [54] A. Kazda. Maltsev digraphs have a majority polymorphism. *European Journal of Combinatorics*, 32:390–397, 2011.



- [55] A. Kazda, M. Kozik, R. McKenzie, and M. Moore. Absorption and directed Jónsson terms. *Outstanding Contributions to Logic*, 16:203–220, 2018.
- [56] M. Kozik, A. Krokhin, M. Valeriote, and R. Willard. Characterizations of several Maltsev conditions. *Algebra universalis*, 73(3):205–224, 2015.
- [57] B. Larose, C. Loten, and C. Tardif. A characterisation of first-order constraint satisfaction problems. *Logical Methods in Computer Science*, 3(4:6), 2007.
- [58] B. Larose and P. Tesson. Universal algebra and hardness results for constraint satisfaction problems. *Theoretical Computer Science*, 410(18):1629–1647, 2009.
- [59] L. Libkin. *Elements of Finite Model Theory*. Springer, 2004.
- [60] A. K. Mackworth. Consistency in networks of relations. *Artificial Intelligence*, 8:99–118, 1977.
- [61] R. N. McKeznie, G. F. McNulty, and W. F. Taylor. *Algebras, Lattices, Varieties (Volume 1)*. American Mathematical Society, 1987.
- [62] U. Montanari. Networks of constraints: Fundamental properties and applications to picture processing. *Information Sciences*, 7:95–132, 1974.
- [63] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley, 1994.
- [64] E. L. Post. *The two-valued iterative systems of mathematical logic*, volume 5. Princeton University Press, Princeton, 1941.
- [65] I. G. Rosenberg. Minimal clones I: the five types. *Lectures in Universal Algebra (Proc. Conf. Szeged, 1983), Colloq. Math. Soc. J. Bolyai*, 43:405–427, 1986.
- [66] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings of the Symposium on Theory of Computing (STOC)*, pages 216–226, 1978.
- [67] U. Schöning. *Logic for Computer Scientists*. Springer, 1989.
- [68] Y. Shitov. Counterexamples to hedetniemi’s conjecture, 2019. arXiv:1905.02167.
- [69] M. H. Siggers. A strong Mal’cev condition for varieties omitting the unary type. *Algebra Universalis*, 64(1):15–20, 2010.
- [70] W. Taylor. Varieties obeying homotopy laws. *Canadian Journal of Mathematics*, 29:498–527, 1977.
- [71] D. N. Zhuk. A proof of CSP dichotomy conjecture. In *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17*, pages 331–342, 2017. <https://arxiv.org/abs/1704.01914>.

## A O-notation

The letters  $o$  and  $O$  stand for the *order* of growth of the function. The *big-O notation* is used to express upper bounds, and the *little-o notation* to express lower bounds. We mention that there exists related notation to describe other kinds of bounds on asymptotic growth, e.g.,  $\Theta$ ,  $\Omega$ ,  $\omega$ , of which we only need  $\Theta$  in this text, so we skip the definitions of the others.

Let  $g: \mathbb{R} \rightarrow \mathbb{R}$  (we use  $\mathbb{R}$  for convenience; similar definitions exist for other domains such as  $\mathbb{N}$  and  $\mathbb{Q}$ , etc). Then  $O(g)$  is the set of all functions  $f: \mathbb{R} \rightarrow \mathbb{R}$  such that there exists  $c, x_0 \in \mathbb{R}$  such that  $|f(x)| \leq cg(x)$  for all  $x \geq x_0$ . Note that

$$f \in O(g) \Leftrightarrow \limsup_{x \rightarrow \infty} \left| \frac{f(x)}{g(x)} \right| < \infty.$$

In typical usage, the formal definition of  $O(g)$  is not used directly; rather, we first use the following simplification rules:

- if  $g(x)$  is a sum of several terms, if there is one with largest growth rate, then we drop all other terms;
- if  $g(x) = c \cdot f(x)$  and  $c$  is a constant that does not depend on  $x$ , then  $c$  can be omitted.

When we write  $O(g)$ , we typically choose  $g$  to be as simple as possible.  $O$ -notation can also be used within arithmetic terms. For example,  $h + O(g)$  denotes the set of functions of the form  $h + f$  for  $f \in O(g)$ . In other words,  $k \in h + O(g)$  is equivalent to  $k - h \in O(g)$ .

We write  $o(g)$  for the set of all functions  $f: \mathbb{R} \rightarrow \mathbb{R}$  such that for every  $\epsilon \in \mathbb{R}_{>0}$  there exists  $x_0 \in \mathbb{R}$  such that  $|f(x)| \leq \epsilon g(x)$  for all  $x \geq x_0$ . Informally,  $f \in o(g)$  means that  $g$  grows much faster than  $f$ . For example,  $x \mapsto 2x$  is in  $o(x \mapsto x^2)$ , and  $x \mapsto 1/x$  is in  $o(1)$ . Note that  $o(g) \subseteq O(g)$ , and that

$$f \in o(g) \Leftrightarrow \lim_{x \rightarrow \infty} \frac{f(x)}{g(x)} = 0.$$

Similarly as in the case of the  $O$ -notation we may use the  $o$ -notation in arithmetic expressions. Note that if  $f \in o(g)$  and  $c$  is a constant, then  $cf \in o(g)$ . Frequent notation is to write  $f \ll g$  (or  $g \gg f$ ) if  $f \in o(g)$ .

We write  $\Theta(g)$  for the set of all functions  $f$  such that there are constants  $c, C$  and  $x_0 \in \mathbb{R}$  such that  $cg(x) \leq f(x) \leq Cg(x)$  for every  $x \geq x_0$ . In other words,  $f \in \Theta(g)$  if  $f \in O(g)$  and  $g \in O(f)$ .

Finally, we write  $f(x) \sim g(x)$  if

$$\lim_{x \rightarrow \infty} \frac{f(x)}{g(x)} = 1$$

and we say that  $f$  and  $g$  are *asymptotically equivalent* (for  $x \rightarrow \infty$ ).

## B Basics of Complexity Theory

For a set  $A$ , we write  $A^*$  for the set of all words over the alphabet  $A$ . A *word over  $A$*  can be seen as a function from  $\{1, \dots, n\} \rightarrow A$ , for some  $n \in \mathbb{N}$ . We write  $\epsilon$  for the empty word (i.e., for the function with the empty domain).

The most classical setting of complexity theory is the study of the computational complexity of functions  $f$  from  $\{0, 1\}^* \rightarrow \{0, 1\}$ . Alternatively, we may view  $f$  as a set of words, namely that set of words  $w$  such that  $f(w) = 1$ ; such sets are also called *formal languages*. There are several mathematically rigorous machine models to formalise the set of such functions that are computable or efficiently computable. The first insight is that most of these machine models lead to the same, or to closely related classes of functions. Complexity theory maps out the landscape of the resulting classes of functions. Typically the first machine model that is introduced in introductory courses are *Turing machines*. They strike a good balance between the following two (almost contradictory!) requirements that a theoretician has for these machine models:

- the model should be relatively simple, so that it is easy to show that it can be simulated by many other machine models.
- the model should be relatively powerful, so that it is easy to show that it can simulate many other machine models.

Turing machines are simple, but still the definition does not easily fit into a few lines. On the other hand, today academics are most likely to already have a very good idea of what a computer program can do (in polynomially many steps); and this coincides with what a Turing machine  $M$  can do (in polynomially many computational steps). In a nutshell, a Turing machine

- has an unboundedly large memory containing values from  $\{-1, 0, 1\}$  (the symbol  $-1$  will be called the *blank* symbol);
- has finitely many *states*  $Q$ ;
- has a *read/write* head;
- has a finite transition function  $\delta: Q \times \{-1, 0, 1\} \rightarrow \Sigma \times Q \times \{l, r\}$ ;
- has a *accept* state  $y \in Q$ .
- has a *start* state  $s \in Q$ .

Initially, the memory just contains the word  $w \in \{0, 1\}^*$ , i.e., in the first cell there is  $w_1$ , in the second cell there is  $w_2$ , etc, and in all further memory cells there is  $-1$ , and the machine *is in state*  $s$ . Depending on its state  $u \in Q$  and the tape content  $c$  under the read-write head, let  $(v, d, m) := \delta(u, c)$ ; then

1. the machine changes to state  $v$ ;
2. the tape content under the read-write head is changed from  $c$  to  $d$ ,
3. the read-write tape moves one cell to the left if  $m = l$ , and one to the right if  $m = r$ .

If the machine reaches state  $y$  it accepts. Every Turing machine describes a formal language, namely the function  $f: \{0, 1\}^* \rightarrow \{0, 1\}$  such that  $f(w) = 1$  if and only if when running the machine on input  $w$  it eventually accepts. We also say that  $M$  *computes*  $f$ , and we then sometimes write  $M(f)$  instead of  $f(w)$ . More generally, Turing machines can be used to describe functions  $f$  from  $\{0, 1\}^*$  to  $\{0, 1\}^*$  where  $f(w)$ , for a given word  $w$ , is the string

that is written on the output tape when the Turing machine accepts (here we require that the machine terminates on every input after finitely many steps, and again we say that  $M$  computes  $f$ ).

So we will pretend in the following that the reader already knows what Turing machines  $M$  are. It turns out that despite the simplicity of Turing machines, they can simulate most of the other machine models, and they can simulate any machine that humans ever constructed (even when neglecting the restriction that we one have some fixed finite maximal memory size in this universe).

In complexity theory we are interested in the number of computation steps that  $M$  needs to perform to compute  $f(w)$ , which corresponds to computation time. For example, we say that a Turing machine runs *in polynomial time* if the number of computation steps is in  $O(|w|^k)$  for some  $k \in \mathbb{N}$ . The class of such functions is denoted by  $P$ .

**Coding.** In the combinatorics course we have met computational complexity for example in the section about colorability. We mentioned that 2-colorability is in  $P$  and that  $k$ -colorability, for  $k \geq 3$ , is NP-hard. But these were problems about finite graphs, whereas in the above we only treated formal languages. But this is just a matter of coding. We first observe that we can simulate any alphabet by our alphabet  $\{0, 1\}$ , by just grouping bits together to represent a richer alphabet. In particular, we will typically use the letter  $\#$  to separate different numbers in the input. One way to represent a graph as a word is to first write the number  $n$  of vertices, followed by the symbol  $\#$ , followed by a sequence of  $n^2$  bits for the adjacency matrix.

The second most important complexity class is NP.

**Definition B.1.** NP (for *nondeterministic polynomial time*) stands for the class of all functions  $f: \{0, 1\}^* \rightarrow \{0, 1\}$  such that there exists a polynomial-time Turing machine  $M$  and a  $d \in \mathbb{N}$  such that for every  $w \in \{0, 1\}^*$  there exists a  $a \in \{0, 1\}^*$  with  $|a| \in O(n^d)$  such that  $f(w) = M(w\#a)$ .

It is a famous open problem whether  $P = NP$ , and it is widely conjectured that  $P \neq NP$ . To explain the significance of this conjecture, we need a couple of more concepts. Let  $f_1, f_2: \{0, 1\}^* \rightarrow \{0, 1\}$ . A *reduction* from  $f_1$  to  $f_2$  is a function  $g: \{0, 1\}^* \rightarrow \{0, 1\}^*$  such that  $f_1(w) = f_2(g(w))$ . A reduction  $g$  is *polynomial-time* if  $g$  can be computed a Turing machine that runs in polynomial time.

**Definition B.2.** A function  $f: \{0, 1\}^* \rightarrow \{0, 1\}$  is *NP-hard* if every function  $g$  in NP has a polynomial-time reduction to  $f$ . A function is called *NP-complete* if it is in NP and NP-hard.

The class *coNP* is dual to NP: it is the class of all functions  $f$  such that  $1 - f$  is in NP. There is an analogous definition for any complexity class  $K$ : a function is in *co-K* if  $1 - f$  is in  $K$ . Clearly, every function in  $P$  is both in NP and in *co-NP*.

A class of finite graphs  $\mathcal{C}$  is in NP if there exists a formal language in NP such that each word in the language codes a graph in  $\mathcal{C}$  (say in the way we described above), and every graph in  $\mathcal{C}$  is coded by some word in the language. Unlike the class  $P$ , it is possible to define the class of all graph classes in NP transparently and fully formally in a few lines (without any reference to Turing machines).

**Theorem B.3** (Fagin). *A class of finite graphs  $\mathcal{C}$  is in NP if and only if there exists an existential second-order sentence  $\Phi$  such that for every finite graph  $G$  we have*

$$G \in \mathcal{C} \text{ if and only if } G \models \Phi.$$

We do not define *existential second-order logic* here. The interested reader is referred to a textbook on finite model theory to learn more about such connections between logic and complexity theory, e.g. [59].

We now return to the question why most researchers believe that  $P \neq NP$ . In order to show that  $P=NP$  suffices to provide for *any* of the known NP-complete problems a polynomial-time algorithm. There are many NP-complete problems that are of central importance in optimisation, scheduling, cryptography, bioinformatics, artificial intelligence and many more areas. If  $P=NP$ , then this would mean a simultaneous breakthrough in all of these areas. It is fair to say that every day, thousands of researchers are directly or indirectly working on proving that  $P=NP$  (since they work on things that are related to the better understanding of some NP-complete problem). The fact that nobody has succeeded (not even came close to) is one of the reasons why we believe that  $P$  cannot be equal to  $NP$ . A world where  $P = NP$  would probably be drastically different from the world we live in. On the other hand, we also have no clue on how to possibly prove that  $P \neq NP$ . And quite a bit is known about approaches to proving  $P \neq NP$  that must fail (see [1]; great read, free download at <https://www.scottaaronson.com/papers/pnp.pdf>).