FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Ondřej Draganov

Finitely generated clones

Department of Algebra

Supervisor of the master thesis:	doc. Mgr. Libor Barto, Ph.D.
Study programme:	Mathematics
Study branch:	Mathematical Structures

Prague 2018

Title: Finitely generated clones

Author: Ondřej Draganov

Department: Department of Algebra

Supervisor: doc. Mgr. Libor Barto, Ph.D., Department of Algebra

Abstract: A clone is a set of finitary operations closed under composition and containing all projections. We say it is finitely generated if there exist a finite subset $\{f_1, \ldots, f_n\}$ such that all the other operations can be expressed as compositions of $f_1, \ldots, f_n$. We present examples of finitely and non-finitely genreated clones on finite sets. First, we demonstrate an explicit construction of operations in finitely generated clones. Secondly, we define relations such that the clones of compatible operations have restricted essential arity, and discuss several modifications. Lastly, for every binary operation f which cannot be composed to yield an essentially ternary operation, we find a maximal clone of essentially at most binary operations containing f.

Keywords: clone, finitely generated, non-finitely generated, essential variables

Contents

Introduction

Mathematicians often study objects that can be described as a set of elements A and a set of operations on A, say $\mathcal{A}$. Such tuple $(A, \mathcal{A})$ is called an *algebra*, and the field of mathematics studying these structures in general is usually being referred to as universal algebra.

For the properties studied in algebras, it is often not that important what is the base set of operations itself, but rather what is the set of term operations, i.e., what operations can be obtained by composing the basic operations. This leads to the notion of *clones* – sets of operations closed under composition, which contain all projections $\pi_i^n : (x_1, \ldots, x_n) \mapsto x_i$. For every algebra we can consider a clone of its term operations, and for every clone $\mathcal{C}$ on a set A we have an algebra $(A, \mathcal{C})$.

Clones on a fixed set form a lattice with respect to the inclusion; the greatest element is the clone of all operations, and the least is the clone of projections. Mathematicians are interested in describing the lattices of clones on finite sets. There is only countably many clones on a 2-element set and they were fully described already by Post [16]. However, the situation is more complicated for larger sets. It is known that already on 3-element set, there are $2^{\aleph_0}$ clones [10].

Given a clone $\mathcal{C}$, one of the interesting questions is whether it is finitely generated, that is, whether there exists a finite subset $\mathcal{S} \subset \mathcal{C}$ such that all the other operations in $\mathcal{C}$ can be expressed as compositions of the operations in $\mathcal{S}$. By [16], all clones on a 2-element set are finitely generated, and in [10] the first examples of non-finitely generated clones were provided.

One of the important results in clone theory is the complete description of maximal clones by Rosenberg's classification [17]. Since the clone of all operations is finitely generated, it follows that every proper subclone is contained in some maximal clone, and there are only finitely many maximal clones on a given finite set [15]. One of the classes of maximal clones are clones of monotone operations with respect to bounded partial orders. It was proved in [11] that all maximal clones, with the possible exception of the aforementioned clones of monotone operations, are finitely generated. For the clones of monotone operations, it was also shown in [11] that they are finitely generated if the partial order is a lattice order or if the poset has at most seven elements. An example of a non-finitely generated clone of monotone operations on an 8-element poset was provided by Tardos [18], and further generalized for example in [19, 5]. The classification of finitely generated maximal clones is still an open problem. One question in particular is whether every finitely generated clone of monotone operations of a bounded poset contains a near-unanimity operation.

Apart from the result of Tardos and the above mentioned generalizations, we have not found much recent research concerned with finitely and non-finitely generated clones. In this thesis we build on the clones presented in [7] and aim to give examples of both finitely and non-finitely generated clones. In Chapter 1 we provide basic definitions and important concepts of clone theory. Chapter 2 is concerned with finitely generated clones, namely with a simple explicit construction showing that a clone is finitely generated. The clones in this chapter are adopted from or inspired mainly by [7, 13]. In Chapter 3 we relationally define certain clones of operations with restricted essential arity, which yields non-finitely generated clones. Then we consider some possible modifications of the relations. Finally, in Chapter 4, given a binary operation f which cannot generate an operation that depends on three or more variables, we find a maximal clone of operations that depend on at most two variables, which contains f. We partially generalize the result to higher arities, which, again, yields examples of non-finitely generated clones.

1. General concepts of clone theory

1.1 Basic definitions and notation

Given a nonempty set A, an *operation* on A is a mapping $f : A^n \longrightarrow A$ from a power of A to A. We call n the *arity* of f and denote it by $\operatorname{ar}(f) = n$. We only consider operations of finite nonzero arity. In this thesis we also deal exclusively with operations on finite sets.

We denote tuples by boldface letters and, unless stated otherwise, its coordinates by the same letters indexed from 1 to the arity, e.g., for $\mathbf{a} \in A^n$ we have $\mathbf{a} = (a_1, a_2, \ldots, a_n)$. We use this convention also for variables, e.g., for an n-ary operation we often write $f(\mathbf{x})$ and mean $f(x_1, x_2, \ldots, x_n)$.

We will often evaluate an n-ary operation on k-tuples component-wise, and it will be convenient to use a shorthand notation. For $\mathbf{a}^i = (a^i_1, a^i_2, \ldots, a^i_k)$ and an n-ary operation f we write

$$f(\mathbf{a}^1, \mathbf{a}^2, \ldots, \mathbf{a}^n) = f\left(\begin{pmatrix} a^1_1 \\ a^1_2 \\ \vdots \\ a^1_k \end{pmatrix}, \begin{pmatrix} a^2_1 \\ a^2_2 \\ \vdots \\ a^2_k \end{pmatrix}, \ldots, \begin{pmatrix} a^n_1 \\ a^n_2 \\ \vdots \\ a^n_k \end{pmatrix}\right) = \begin{pmatrix} f(a^1_1, a^2_1, \ldots, a^n_1) \\ f(a^1_2, a^2_2, \ldots, a^n_2) \\ \vdots \\ f(a^1_k, a^2_k, \ldots, a^n_k) \end{pmatrix}.$$

For sets $A_1, \ldots, A_n \subseteq A$ we define a set

$$f(A_1, \ldots, A_n) = \{f(a_1, \ldots, a_n) \mid a_i \in A_i \text{ for } i = 1, \ldots, n\}$$

and similarly for sets of tuples using the above convention.

Another shorthand notation will be used for universal equalities, where we omit the universal quantifiers, e.g.,

$$\forall \mathbf{x} \in A^n : \; f(x_1, \ldots, x_n) = g(x_1, \ldots, x_n)$$

will be written as

$$f(x_1, \ldots, x_n) = g(x_1, \ldots, x_n).$$

By n-ary *projection* onto the i-th coordinate we mean the operation

$$\pi^n_i(x_1, \ldots, x_n) = x_i.$$

The central concept of this thesis is a structure called *clone*.

Definition 1. *Let A be a nonempty set and $\mathcal{C}$ be a set of operations on A. We say that $\mathcal{C}$ is a* clone *if*

- *$\mathcal{C}$ contains all projections, and*
- *$\mathcal{C}$ is closed under superposition, that is, if $f, g_1, \ldots, g_n \in \mathcal{C}$, where f is n-ary and $g_1, \ldots, g_n$ are k-ary, then also*

$$f(g_1, \ldots, g_n)(x_1, \ldots, x_k) := f(g_1(x_1, \ldots, x_k), \ldots, g_n(x_1, \ldots, x_k)) \in \mathcal{C}.$$

Using projections and superposition, we can see that a clone is closed under a general composition, i.e., an operation given by any term in some operations from a clone is again in the clone. For instance, given $f(x_1, x_2, x_3), g(y_1, y_2), h(z) \in \mathcal{C}$ we have also

$$\begin{aligned} &f(g(x_1, x_2), x_1, h(x_3)) = \\ &\qquad f(g(\pi^3_1(x_1, x_2, x_3), \pi^3_2(x_1, x_2, x_3)), \pi^3_1(x_1, x_2, x_3), h(\pi^3_3(x_1, x_2, x_3))) \in \mathcal{C}. \end{aligned}$$

It will be sometimes useful to depict composed operations as trees. The above example would be depicted as in Figure 1.1.

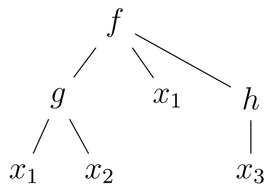


Figure 1.1: A tree depiction of $f(g(x_1, x_2), x_1, h(x_3))$

It will be convenient to also use a different definition of a clone, see e.g. [9, 12, 15].

Definition 2. *We say that a set of operations on a given set is a* clone*, if it contains identity and is closed under the following four procedures:*

- Substitution. *Given an n-ary operation f and an m-ary operation g, substitution forms an $(n + m - 1)$-ary operation*

$$(f * g)(x_1, x_2, \ldots, x_{m+n-1}) = f(g(x_1, \ldots, x_m), x_{m+1}, \ldots, x_{m+n-1}).$$

- Permutation of variables. *Given an n-ary operation f and a permutation σ on {1,. . . ,n}, we can permute the variables to obtain*

$$\mathrm{Perm}_\sigma(f)(x_1, x_2, \ldots, x_n) = f(x_{\sigma(1)}, x_{\sigma(2)}, \ldots, x_{\sigma(n)}).$$

- Identification of variables. *Given an n-ary operation f, we can identify the first two variables to obtain an $(n - 1)$-ary operation*

$$\Delta f(x_1, x_2, \ldots, x_{n-1}) = f(x_1, x_1, x_2, \ldots, x_{n-1}).$$

- Introduction of a “dummy” / fictitious variable. *Given an n-ary operation f, we can introduce a new variable to obtain an $(n + 1)$-ary operation*

$$\nabla f(x_1, x_2, \ldots, x_n, x_{n+1}) = f(x_1, x_2, \ldots, x_n).$$

The two definitions are, indeed, equivalent. We have seen how to use projections and superposition to obtain a general composition. It is clear that we can also carry out all the procedures in Definition 2 in a similar manner. For the other direction, we first realize that introducing $n - 1$ new variables to the identity and than swapping the first and the i-th variable, we get the projection

π_i^n. To obtain the superposition of f and $g_1, \dots, g_n$, we first use the substitution and permutation of variables to substitute g_i's into f, and than we identify the corresponding variables in each g_i.

We denote $\mathcal{P}_A$ the clone of all projections, also called the *trivial clone*, and $\mathcal{O}_A$ the clone of all operations on A, also called the *full clone*. It is easily seen that an intersection of clones is again a clone. For $\mathcal{S}$, an arbitrary set of operations on a given set, $\mathrm{Clo}(\mathcal{S})$ denotes the smallest clone (with respect to inclusion) containing $\mathcal{S}$. Every operation in $\mathrm{Clo}(\mathcal{S})$ is expressible as a term using operations from $\mathcal{S}$ – with a slight abuse of notation expressing any projection as just a single variable. We also say that $\mathcal{S}$ *generates* the clone $\mathrm{Clo}(\mathcal{S})$, which leads us to define another core notion for this text.

Definition 3. *Let $\mathcal{C}$ be a clone on a set A. We say that $\mathcal{C}$ is* finitely generated *if there exists a finite subset $\mathcal{S} \subseteq \mathcal{C}$ such that $\mathcal{C} = \mathrm{Clo}(\mathcal{S})$.*

In the opposite case, when there is no such a finite subset, the clone $\mathcal{C}$ is said to be non-finitely generated.

We denote
$$\mathcal{C}^{(k)} = \{f \in \mathcal{C} \mid f \text{ is at most } k\text{-ary}\}.$$
Given $f \in \mathcal{C}^{(k)}$, we may always assume f to be k-ary. If the arity of f is lower, we add fictitious variables to make it k-ary.

It is easy to observe that a clone $\mathcal{C}$ on a finite set is finitely generated if and only if there exist $k \in \mathbb{N}$ such that $\mathcal{C} = \mathrm{Clo}(\mathcal{C}^{(k)})$. If $\mathcal{C}$ is finitely generated by $f_1, \dots, f_n$, and m is the maximal arity of $f_1, \dots, f_n$, then $\mathcal{C} = \mathrm{Clo}(\mathcal{C}^{(m)})$. The converse holds since $\mathcal{C}^{(k)}$ is finite.

1.2 Relations, compatibility and Galois correspondence

There is a powerful tool for studying clones emerging from a connection between sets of operations and sets of relations on a given set [2], [6].

Given a set A, a *relation* on A of arity n is a subset $R \subseteq A^n$. We will sometimes write $R(a_1, a_2, \dots, a_n)$ for $(a_1, a_2, \dots, a_n) \in R$. Given a set of relations $\mathcal{R}$, a pair $\mathbb{A} = (A, \mathcal{R})$ is a *relational structure* and is often denoted by blackboard bold letters. By abuse of notation we will also treat $\mathbb{A}$ as the set of relations itself.

Definition 4. *We say an operation $f : A^n \to A$ is* compatible *with a relation $R \subseteq A^k$ if $f(\mathbf{a}^1, \mathbf{a}^2, \dots, \mathbf{a}^n) \in R$ whenever $\mathbf{a}^1, \mathbf{a}^2, \dots, \mathbf{a}^n \in R$, i.e., we have the following scheme:*

$$\begin{array}{cccccc}
a_1^1 & a_1^2 & \dots & a_1^n & \xrightarrow{f} & f(a_1^1, a_1^2, \dots, a_1^n) \\
a_2^1 & a_2^2 & \dots & a_2^n & \xrightarrow{f} & f(a_2^1, a_2^2, \dots, a_2^n) \\
\vdots & \vdots & & \vdots & \vdots & \vdots \\
a_k^1 & a_k^2 & \dots & a_k^n & \xrightarrow{f} & f(a_k^1, a_k^2, \dots, a_k^n) \\
\pitchfork & \pitchfork & & \pitchfork & & \pitchfork \\
R & R & \dots & R & & R
\end{array}.$$

We also say that R is preserved *by f.*

The schemes of the form used in the definition will be referred to as "compatibility schemes". We will call the right-hand side column the *resulting column.*

An operation compatible with all relations of a relational structure $\mathbb{A}$ is called a *polymorphism* of $\mathbb{A}$. The set of all polymorphisms of $\mathbb{A}$ is denoted $\mathrm{Pol}(\mathbb{A})$. Similarly a relation preserved by all operations from a given set $\mathcal{S}$ is said to be *invariant* under $\mathcal{S}$ and the set of all relations invariant under $\mathcal{S}$ is denoted $\mathrm{Inv}(\mathcal{S})$, i.e.,

$$\begin{aligned}\mathrm{Pol}(\mathbb{A}) &= \Big\{f : A^n \to A \,|\, n \in \mathbb{N},\ \forall R \in \mathbb{A} : f \text{ is compatible with } R\Big\},\\ \mathrm{Inv}(\mathcal{S}) &= \Big\{R \subseteq A^k \,|\, k \in \mathbb{N},\ \forall f \in \mathcal{S} : f \text{ is compatible with } R\Big\}.\end{aligned}$$

For a fixed set A, operators Pol and Inv realize a Galois correspondence between sets of operations on A and sets of relations on A.

Seeing this, an important question arises. What are the closed sets on either side of this correspondence? We are concerned only with the case where the domain A is finite. On the operations side the answer is clones. It is not difficult to observe that $\mathrm{Pol}(\mathbb{A})$ forms a clone for any set of relations $\mathbb{A}$. The more difficult part is to show that clones, indeed, form closed sets, i.e., that every clone is a set of polymorphisms of some set of relations.

Theorem 1 ([2], [6]). *Let A be a finite set and $\mathcal{S}$ be a set of operations on A. Then*

$$\mathrm{Clo}(\mathcal{S}) = \mathrm{Pol}(\mathrm{Inv}(\mathcal{S})).$$

A similar result can be derived for the relational side. The closed sets there are so called *relational clones*, which are sets of relations that are closed under pp-definitions (primitively positive definitions), and contain the empty relation. We say a relation R is *pp-definable* from $R_1, \ldots, R_k$ if there exists a formula defining R, which uses only existential quantifiers, conjunctions, equality and relations $R_1, \ldots, R_k$. We denote by $\mathrm{RelClo}(\mathbb{A})$ the smallest relational clone containing $\mathbb{A}$. It is, again, straightforward to check that $\mathrm{Inv}(\mathcal{S})$ is always a relational clone. The converse implication also holds.

Theorem 2 ([2], [6]). *Let A be a finite set and $\mathbb{A}$ be a set of relations on A. Then*

$$\mathrm{RelClo}(\mathbb{A}) = \mathrm{Inv}(\mathrm{Pol}(\mathbb{A})).$$

1.3 Near-unanimity operations and finitely generated clones

In this section we introduce the notion of near-unanimity operations and demonstrate how the Pol-Inv Galois correspondence can be utilized.

In universal algebra, it is often studied what we can derive about an algebra or a variety given the existence of an operation satisfying some interesting identity. One of the important operations is a so called *majority*, which is a ternary operation satisfying

$$m(y, x, x) = m(x, y, x) = m(x, x, y) = x.$$

This concept can be generalized in a straightforward manner to so called *near-unanimity* operations. We say that an operation f of arity $n \geq 3$ is a near-unanimity operation if the identity

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

holds for y being in any position. Note that majority is a special case of near-unanimity operation and it is actually the strongest one in the following sense. Let $\mathcal{C}$ be a clone and $f \in \mathcal{C}$ be an n-ary near-unanimity operation. Adding fictitious variables $y_1, \dots, y_k$, we can construct an operation $g \in \mathcal{C}$ of arity $n + k$ as

$$g(x_1, x_2, \dots, x_n, y_1, \dots, y_k) := f(x_1, x_2, \dots, x_n),$$

and g is then also a near-unanimity operation.

We now aim to prove the following known corollary of the results due to K. A. Baker and A. F. Pixley [1].

Proposition 3. *If a clone on a finite set contains a near-unanimity operation, then it is finitely generated.*

To prove the proposition, we will need the concept of a *projection* of a relation. Given an m-ary relation R and a set of coordinates $I = \{i_1, \dots, i_k\}$, the projection $\mathrm{Proj}_I(R)$ of R onto the coordinates I is the k-ary relation

$$\mathrm{Proj}_I(R) = \{(r_{i_1}, r_{i_2}, \dots, r_{i_k}) \mid (r_1, r_2, \dots, r_m) \in R\}.$$

Note that a projection of a relation can be always pp-defined from the relation. For example the projection onto the first two coordinates of a 5-ary relation R would be defined as

$$\mathrm{Proj}_{1,2}(R)(x_1, x_2) \;\equiv\; \exists y_3 \, \exists y_4 \, \exists y_5 : R(x_1, x_2, y_3, y_4, y_5).$$

Before we continue, let us go through a simple example to build up the intuition. Let us imagine we have two relations

$$R_1 = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \right\},$$
$$R_2 = \left\{ \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \right\}.$$

The projection of both R_1 and R_2 onto any two coordinates is

$$P = \left\{ \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 0 \\ 0 \end{pmatrix} \right\}.$$

This shows us that in general the projections of a relation do not uniquely determine the relation. However, if we have an unknown ternary relation $R_?$, we know that the projections of it onto any two coordinates is P, and, moreover, that it is compatible with a majority operation m, then we know that

$$\begin{array}{cccccc} ? & 0 & 0 & \overset{m}{\to} & 0 \\ 0 & ? & 0 & \overset{m}{\to} & 0 \\ 0 & 0 & ? & \overset{m}{\to} & 0\,, \\ \Cap & \Cap & \Cap & & \Cap \\ R_? & R_? & R_? & & R_? \end{array}$$

which rules out the option R_1. We then actually know now that necessarily $R_? = R_2$. The compatibility with a majority tells us the relation is "maximal" with given projections in the sense that every tuple consistent with the projections is in the relation. Note that this exactly means that $R_?$ can be pp-defined from its projections as

$$R_?(x_1, x_2, x_3) \equiv \mathrm{Proj}_{1,2}(R_?)(x_1, x_2) \,\wedge\, \mathrm{Proj}_{1,3}(R_?)(x_1, x_3) \,\wedge\, \mathrm{Proj}_{2,3}(R_?)(x_2, x_3).$$

This holds in general.

Lemma 4. *Let f be an n-ary near-unanimity operation which preserves an m-ary realtion R, where $n \leq m$. Then R is uniquely determined by its $(n-1)$-ary projections as*

$$R(x_1, \ldots, x_m) \equiv \bigwedge_{\{i_1,\ldots,i_{n-1}\} \subseteq \{1,\ldots,m\}} \mathrm{Proj}_{i_1,\ldots,i_{n-1}}(R)(x_{i_1}, x_{i_2}, \ldots, x_{i_{n-1}}).$$

Proof. We inductively prove that for all k, such that $n-1 \leq k < m$, the k-ary projections of R uniquely determine the $(k+1)$-ary projections of R in the sense of the pp-definition in the claim. It is enough to show that the projection onto the first $k+1$ coordinates is uniquely determined by all the projections to "all but one of the first $k+1$ coordinates". For the other cases, there are just different indices.

Suppose that $(a_1, a_2, \ldots, a_{k+1})$ is such that the projections of the tuple are in the corresponding projections of R, i.e., for all $i = 1, \ldots, k+1$ we have

$$(a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_{k+1}) \in \mathrm{Proj}_{1,\ldots,i-1,i-1,\ldots,k+1}(R).$$

Then we obtain

$$\begin{array}{cccccccc} ? & a_1 & a_1 & \ldots & a_1 & a_1 & \xrightarrow{f} & a_1 \\ a_2 & ? & a_2 & \ldots & a_2 & a_2 & \xrightarrow{f} & a_2 \\ a_3 & a_3 & ? & \ldots & a_3 & a_3 & \xrightarrow{f} & a_3 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots \\ a_k & a_k & a_k & \ldots & ? & a_k & \xrightarrow{f} & a_k \\ a_{k+1} & a_{k+1} & a_{k+1} & \ldots & a_{k+1} & ? & \xrightarrow{f} & a_{k+1}, \\ ? & ? & ? & \ldots & ? & ? & \xrightarrow{f} & ? \\ \vdots & \vdots & \vdots & \ldots & \vdots & \vdots & & \vdots \\ ? & ? & ? & \ldots & ? & ? & \xrightarrow{f} & ? \\ \Cap & \Cap & \Cap & & \Cap & \Cap & & \Cap \\ R & R & R & & R & R & \Rightarrow & R \end{array}$$

which means that $(a_1, \ldots, a_{k+1}) \in \mathrm{Proj}_{1,\ldots,k+1}(R)$, and that is what we wanted. $\square$

A direct consequence is the following lemma.

Lemma 5. *Let $\mathcal{S}$ be a set of operations on a finite set A containing an n-ary near-unanimity operation. Then*

$$\mathcal{S} = \mathrm{Pol}(\mathrm{Inv}(\mathcal{S})) = \mathrm{Pol}(\mathrm{Inv}(\mathcal{S})^{(n-1)}),$$

where $\mathrm{Inv}(\mathcal{S})^{(n-1)}$ *are all the at most* $(n-1)$*-ary relations from* $\mathrm{Inv}(\mathcal{S})$*.*

Proof. The inclusion "$\subseteq$" is trivial. For the other inclusion, let $R \in \mathrm{Inv}(\mathcal{S})$ be a relation of arity at least n. Then $\mathrm{Inv}(\mathcal{S})^{(n-1)}$ contains all of its $(n-1)$-ary projections, since $\mathrm{Inv}(\mathcal{S})$ is a relational clone, i.e., it is closed under pp-definitions. Since R is compatible with the near-unanimity operation in $\mathcal{S}$, by Lemma 4, R is pp-definable from these projections. This means that if we take a polymorphism of $\mathrm{Inv}(\mathcal{S})^{(n-1)}$, it will also be compatible with R. This proves the inclusion $\mathrm{Pol}(\mathrm{Inv}(\mathcal{S})) \supseteq \mathrm{Pol}(\mathrm{Inv}(\mathcal{S})^{(n-1)})$. □

Getting back to the proof of Proposition 3; since there are only finitely many at most $(n-1)$-ary relations on a finite set, there are also only finitely many sets of the form $\mathrm{Inv}(D)^{(n-1)}$. Recall that $\mathcal{C}^{(k)}$ denotes the set of all at most k-ary operations from $\mathcal{C}$. We have a chain

$$\mathcal{C}^{(1)} \subseteq \mathcal{C}^{(2)} \subseteq \mathcal{C}^{(3)} \subseteq \cdots \subseteq \mathcal{C}^{(k)} \subseteq \cdots \subseteq \mathcal{C},$$

and applying $\mathrm{Inv}(\text{-})^{(n-1)}$ we get

$$\mathrm{Inv}(\mathcal{C}^{(1)})^{(n-1)} \supseteq \mathrm{Inv}(\mathcal{C}^{(2)})^{(n-1)} \supseteq \ldots \supseteq \mathrm{Inv}(\mathcal{C}^{(k)})^{(n-1)} \supseteq \ldots \supseteq \mathrm{Inv}(\mathcal{C})^{(n-1)},$$

in which we can then only have finitely many strict inclusions. But that means there exists k such that

$$\mathrm{Inv}(\mathcal{C}^{(k)})^{(n-1)} = \mathrm{Inv}(\mathcal{C})^{(n-1)}. \tag{1.1}$$

We can assume that $k \geq n$, so $\mathcal{C}^{(k)}$ contains an n-ary near-unanimity operation. Using Theorem 1 and Lemma 5, we obtain

$$\begin{aligned}\mathrm{Clo}(\mathcal{C}^{(k)}) &\overset{\text{Theorem 1}}{=} \mathrm{Pol}(\mathrm{Inv}(\mathcal{C}^{(k)})) \overset{\text{Lemma 5}}{=} \mathrm{Pol}(\mathrm{Inv}(\mathcal{C}^{(k)})^{(n-1)}) \overset{(1.1)}{=} \\ &= \mathrm{Pol}(\mathrm{Inv}(\mathcal{C})^{(n-1)}) \overset{\text{Lemma 5}}{=} \mathrm{Pol}(\mathrm{Inv}(\mathcal{C})) \overset{\text{Theorem 1}}{=} \mathcal{C},\end{aligned}$$

which shows that $\mathcal{C}$ is finitely generated by its at most k-ary part, and hence finishes the proof of Proposition 3.

This result can be generalized to clones whose operations have restricted image as shown in [4]. For $B \subseteq A$ we say that f is B-near-unanimity operation if

$$\forall b, c \in B : f(b, \ldots, b, c, b, \ldots, b) = b$$

for every position of the different element c. We can then formulate the following.

Proposition 6 ([4])**.** *Let $B \subseteq A$ be finite sets and $\mathcal{C}$ be a clone on A. Let us denote*

$$\mathcal{C}[B] = \{f \in \mathcal{C} \mid \mathrm{Im}(f) \subseteq B\} \cup \mathcal{P}_A.$$

If $\mathcal{C}[B]$ contains a B-near-unanimity operation, then it is finitely generated.

1.4 Characterizations of non-finitely generated clones

In this section we introduce several known equivalent formulations of a clone being non-finitely generated. A simple observation which we use throughout Chapter 3 is the following lemma.

Lemma 7. *Let $\{\mathcal{C}_k\}_{k\in\mathbb{N}}$ be a sequence of clones such that*

$$\mathcal{C}_1 \subsetneq \mathcal{C}_2 \subsetneq \cdots \subsetneq \mathcal{C}_k \subsetneq \cdots \subsetneq \bigcup_{k\in\mathbb{N}} \mathcal{C}_k =: \mathcal{C}.$$

Then $\mathcal{C}$ is a non-finitely generated clone.

Proof. Clearly, $\mathcal{C}$ is a clone, since for any $f, g_1, \ldots, g_n \in \mathcal{C}$ there exists $k \in \mathbb{N}$ such that $f, g_1, \ldots, g_n \in \mathcal{C}_k$, i.e. also the superposition $f(g_1, \ldots, g_n) \in \mathcal{C}_k \subseteq \mathcal{C}$.

By the same argument, $\mathcal{C}$ is non-finitely generated, because given any finite number of operations $f_1, \ldots, f_n \in \mathcal{C}$, there exists $k \in \mathbb{N}$ such that

$$\mathrm{Clo}(f_1, \ldots, f_n) \subseteq \mathcal{C}_k \subsetneq \mathcal{C}.$$

□

In particular we can always consider the chain of clones generated by the at most m-ary parts. We then obtain a characterization.

Lemma 8. *A clone $\mathcal{C}$ on a finite set is non-finitely generated iff there are infinitely many strict inclusions in the chain*

$$\mathrm{Clo}(\mathcal{C}^{(1)}) \subseteq \mathrm{Clo}(\mathcal{C}^{(2)}) \subseteq \cdots \subseteq \mathrm{Clo}(\mathcal{C}^{(m)}) \subseteq \cdots \subseteq \bigcup_{m\geq 1} \mathrm{Clo}(\mathcal{C}^{(m)}) = \mathcal{C}.$$

Proof. By Lemma 7, we have the implication from right to left. The converse also holds, because if there is only finitely many strict inclusions in the chain, we have $\mathrm{Clo}(\mathcal{C}^{(m)}) = \mathcal{C}$ for some $m \in \mathbb{N}$, and hence $\mathcal{C}$ is finitely generated, since there are only finitely many at most m-ary operations on a finite set. □

Using the Pol-Inv Galois correspondence, we can prove another useful condition equivalent to $\mathcal{C}$ being non-finitely generated. The following characterization was used by Tardos in [18].

Lemma 9. *Let $\mathcal{C}$ be a clone on a finite set A. Then $\mathcal{C}$ is non-finitely generated if and only if*

$$\forall m \in \mathbb{N}\ \exists R, \text{ a relation on } A:\ R \in \mathrm{Inv}(\mathcal{C}^{(m)}) \text{ and } R \notin \mathrm{Inv}(\mathcal{C}). \tag{1.2}$$

Proof. Let $\mathcal{C}$ be finitely generated by operations $f_1, \ldots, f_k$. We set

$$m = \max \mathrm{ar}(a_1), \ldots, \mathrm{ar}(a_k).$$

Let R be a relation on A such that $R \in \mathrm{Inv}(\mathcal{C}^{(m)})$. Then R is compatible with $f_1, \ldots, f_k$, and hence it is also compatible with $\mathrm{Clo}(f_1, \ldots, f_k) = \mathcal{C}$, i.e., $R \in \mathrm{Inv}(\mathcal{C})$.

For the converse implication, let m be such that for all relations R on A we have $R \in \operatorname{Inv}(\mathcal{C}^{(m)}) \Rightarrow R \in \operatorname{Inv}(\mathcal{C})$. Then $\operatorname{Inv}(\mathcal{C}^{(m)}) = \operatorname{Inv}(\mathcal{C})$, since "$\supseteq$" holds for every clone. We have

$$\mathcal{C} = \operatorname{Pol}(\operatorname{Inv}(\mathcal{C})) = \operatorname{Pol}(\operatorname{Inv}(\mathcal{C}^{(m)})) = \operatorname{Clo}(\mathcal{C}^{(m)}),$$

hence $\mathcal{C}$ is finitely generated by the set $\mathcal{C}^{(m)}$. □

Essentially, this is just a relational reformulation of the previous Lemma 8, since we have $\operatorname{Pol}(\operatorname{Inv}(\mathcal{C}^{(m)})) \subsetneq \operatorname{Pol}(\operatorname{Inv}(\mathcal{C}))$ iff $\operatorname{Inv}(\mathcal{C}^{(m)}) \supsetneq \operatorname{Inv}(\mathcal{C})$, so the condition (1.2) is equivalent to saying that there are infinitely many strict inclusions in the chain

$$\operatorname{Pol}(\operatorname{Inv}(\mathcal{C}^{(1)})) \subseteq \operatorname{Pol}(\operatorname{Inv}(\mathcal{C}^{(2)})) \subseteq \cdots \subseteq \operatorname{Pol}(\operatorname{Inv}(\mathcal{C}^{(m)})) \subseteq \cdots \subseteq \bigcup_{m\geq 1} \operatorname{Pol}(\operatorname{Inv}(\mathcal{C}^{(m)})) = \mathcal{C}.$$

Since $\operatorname{Pol}(\operatorname{Inv}(\mathcal{C}^{(m)})) = \operatorname{Clo}(\mathcal{C}^{(m)})$ by Theorem 1, this is the chain from Lemma 8.

As shown in [19], Lemma 9 can be reformulated using the notion of m-covers. For a set R and $m \in \mathbb{N}$ we say that $R_1, \ldots, R_s$ is *m-cover* of R, if

(i) $\bigcup_{i=1}^{s} R_i = R$, and

(ii) every m-element subset of R is contained in one of the sets $R_1, \ldots, R_s$.

This notion offers an equivalent condition to membership of a relation in $\operatorname{Inv}(\mathcal{C}^{(m)})$.

Lemma 10. *Let $\mathcal{C}$ be a clone on A, and R be a relation on A. Then $R \in \operatorname{Inv}(\mathcal{C}^{(m)})$ iff there exist $R_1, \ldots, R_s$, an m-cover of R, such that $R_1, \ldots, R_s \in \operatorname{Inv}(\mathcal{C})$.*

Proof. If $R \in \operatorname{Inv}(\mathcal{C}^{(m)})$, we set $R_{\mathbf{r}_1,\ldots,\mathbf{r}_m} = \left\{ f(\mathbf{r}_1, \ldots, \mathbf{r}_m) \mid f \in \mathcal{C}^{(m)} \right\}$ for every set of m elements $\mathbf{r}_1, \ldots, \mathbf{r}_m \in R$. Then clearly $\{\mathbf{r}_1, \ldots, \mathbf{r}_m\} \subseteq R_{\mathbf{r}_1,\ldots,\mathbf{r}_m}$ for any m element subset of R, since $\mathcal{C}^{(m)}$ contains projections. By $R \in \operatorname{Inv}(\mathcal{C}^{(m)})$, we have $R_{\mathbf{r}_1,\ldots,\mathbf{r}_m} \subseteq R$. Hence, it follows that $\{R_{\mathbf{r}_1,\ldots,\mathbf{r}_m} \mid \mathbf{r}_1, \ldots, \mathbf{r}_m \in R\}$ is m-cover of R.

Let $\mathbf{r}_1, \ldots, \mathbf{r}_m \in R$. We show that $R_{\mathbf{r}_1,\ldots,\mathbf{r}_m} \in \operatorname{Inv}(\mathcal{C})$. Let $g \in \mathcal{C}$ be an n-ary operation and $\mathbf{s}_1, \ldots, \mathbf{s}_n \in R_{\mathbf{r}_1,\ldots,\mathbf{r}_m}$. Then there exist m-ary operations $f_1, \ldots, f_n \in \mathcal{C}^{(m)}$ such that $f_i(\mathbf{r}_1, \ldots, \mathbf{r}_m) = \mathbf{s}_i$ for all $i = 1, \ldots, n$. Since the superposition $g(f_1, \ldots, f_n)$ is m-ary, that is, $g(f_1, \ldots, f_n) \in \mathcal{C}^{(m)}$, we have also

$$g(\mathbf{s}_1, \ldots, \mathbf{s}_n) = g(f_1, \ldots, f_n)(\mathbf{r}_1, \ldots, \mathbf{r}_m) \in R_{\mathbf{r}_1,\ldots,\mathbf{r}_m}.$$

On the other hand, let $R_1, \ldots, R_s \in \operatorname{Inv}(\mathcal{C})$ be an m-cover of R and $f \in \mathcal{C}^{(m)}$. Let $\mathbf{r}_1, \ldots, \mathbf{r}_m \in R$. Then there exists $i \in \{1, \ldots, s\}$ such that $\mathbf{r}_1, \ldots, \mathbf{r}_m \in R_i$. Since $R_i \in \operatorname{Inv}(\mathcal{C})$, we have

$$f(\mathbf{r}_1, \ldots, \mathbf{r}_m) \in R_i \subseteq R.$$

Therefore, $R \in \operatorname{Inv}(\mathcal{C}^{(m)})$. □

Putting Lemma 9 and Lemma 10 together yields the following characterization.

Corollary 11. *A clone $\mathcal{C}$ on a finite set is non-finitely generated iff for every $m \in \mathbb{N}$ there exist $R_1, \ldots, R_s \in \operatorname{Inv}(\mathcal{C})$, an m-cover of $R := \bigcup_{i=1}^{s} R_i$, such that $R \notin \operatorname{Inv}(\mathcal{C})$.*

2. Finitely generated clones

In this chapter we present several examples of finitely generated clones, describe an explicit construction showing how to express operations using the generators, and demonstrate how the construction can be modified.

2.1 The clone of all operations

As noted before, $\mathcal{O}_A$ – the set of all operations on a set A – is a clone, and, by Proposition 3, it is finitely generated for any finite A, since it contains a near-unanimity operation. There is, however, also a simple explicit construction showing that $\mathcal{O}_A$ is finitely generated, which can also be modified to show that some other clones without near-unanimity operations are finitely generated.

Boolean functions

Let us start with the binary case $\mathcal{O}_{\{0,1\}}$. It is a well known fact that we can construct any boolean function $f : \{0,1\}^n \longrightarrow \{0,1\}$ using just identity, negation, conjunction and disjunction. The trick is to take all the tuples on which f evaluates to 1, and, using the available operations, construct a logical formula saying "the tuple $\mathbf{x}$ is one of these".

For every $\mathbf{a}$ such that $f(\mathbf{a}) = 1$ we define an n-ary operation

$$g_{\mathbf{a}\to 1}(x_1, \ldots, x_n) := u_1(x_1) \wedge u_2(x_2) \wedge \cdots \wedge u_n(x_n),$$

where u_i is identity if $a_i = 1$ and negation if $a_i = 0$. Then $g_{\mathbf{a}\to 1}(\mathbf{x}) = 1$ if and only if $\mathbf{x} = \mathbf{a}$. This way we get $f = \bigvee_{\mathbf{a}: f(\mathbf{a})=1} g_{\mathbf{a}\to 1}$, where $\bigvee$ is just a composition of binary disjunctions.

The explicit construction for $\mathcal{O}_A$

Generalizing this construction to an arbitrary finite set A is quite straightforward. We will show that any operation on A is generated by some unary and binary operations. Let us fix an element $0 \in A$. In place of identity and negation we will need simple unary operations $u_{a\to b}$ given by

$$u_{a\to b}(x) = \begin{cases} b & \text{if } x = a, \\ 0 & \text{otherwise.} \end{cases}$$

As for the binary operations, we take a quite natural generalization of conjunction and disjunction defined by

$$\wedge(x,y) = \begin{cases} x & \text{if } x = y, \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \vee(x,y) = \begin{cases} x & \text{if } y = 0 \text{ or } x = y, \\ y & \text{if } x = 0 \text{ or } x = y, \\ 0 & \text{otherwise.} \end{cases} \tag{2.1}$$

Note that both operations are associative and by composing we get n-ary versions

$$\bigwedge(\mathbf{x}) = \begin{cases} a & \text{if } x_1 = \cdots = x_n = a, \\ 0 & \text{otherwise} \end{cases}, \quad \bigvee(\mathbf{x}) = \begin{cases} a & \text{if } \{x_1, \ldots, x_n\} = \{0, a\}, \\ 0 & \text{otherwise.} \end{cases}$$

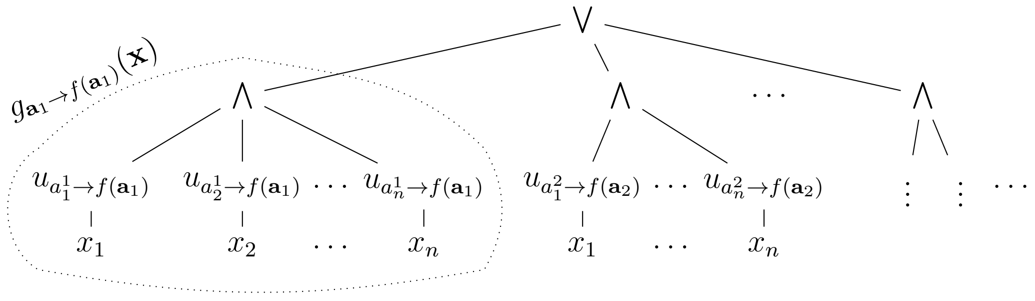
Even though these are formally different operations for each arity, for the sake of simplicity we will denote them by the same symbol for all $n \in \mathbb{N}$. Note that for $n = 2$ we have just the basic operation $\wedge/\vee$ and for $n = 1$ we define both as identity, which is always in a clone since it is a projection. The intended arity will be always clear from the context.

Given a tuple $\mathbf{a} \in A^n$ and a desired result $b \in A$ we can now generate an n-ary operation $g_{\mathbf{a}\to b}(\mathbf{x}) = u_{a_1\to b}(x_1) \wedge u_{a_2\to b}(x_2) \wedge \cdots \wedge u_{a_n\to b}(x_n)$ satisfying

$$g_{\mathbf{a}\to b}(\mathbf{x}) = \begin{cases} b & \text{if } \mathbf{x} = \mathbf{a}, \\ 0 & \text{otherwise.} \end{cases}$$

The definition works even for $g_{\mathbf{a}\to 0}$. We get a constant 0, but it is convenient for clarity to take these into account as well.

Let now $f \in \mathcal{O}_A$ be an arbitrary n-ary operation on A. It is easy to see that $f = \bigvee_{\mathbf{a}\in A^n} g_{\mathbf{a}\to f(\mathbf{a})}$, since for any tuple at most one "g" will be nonzero and so the disjunction returns the result of this "g", which is the result of f on the given tuple.



$\bigvee$
$g_{\mathbf{a}_1\to f(\mathbf{a}_1)}(\mathbf{x})$
$\bigwedge$
$\bigwedge$
$\cdots$
$\bigwedge$
$u_{a^1_1\to f(\mathbf{a}_1)}$
$u_{a^1_2\to f(\mathbf{a}_1)}$
$\cdots$
$u_{a^1_n\to f(\mathbf{a}_1)}$
$u_{a^2_1\to f(\mathbf{a}_2)}$
$\cdots$
$u_{a^2_n\to f(\mathbf{a}_2)}$
$\vdots$ $\vdots$ $\cdots$
x_1 x_2 $\cdots$ x_n
x_1 $\cdots$ x_n

Figure 2.1: Operation f generated by conjunction, disjunction and unary operations. The "$\bigwedge$-blocks" are $g_{\mathbf{a}\to f(\mathbf{a})}$ for all the different tuples $\mathbf{a} \in A^n$. The $\bigwedge$ and $\bigvee$ are compositions of the binary versions, so the tree is in fact much deeper then three levels.

We have shown that any operation is generated by unary and binary operations. Since there are only finitely many of these – and we do not even need all of them – this proves that the clone $\mathcal{O}_A$ is finitely generated.

2.2 Modifications of the construction

In this section we demonstrate how the above construction can be modified to show that various clones are finitely generated.

In Section 3.3 we talk about restricted image and about so called insensitivity, and show that clones of all operations with image restricted to some B which are also B-insensitive are non-finitely generated. Here we show that taking each of these two properties on its own yields finitely generated clones. The clones described here are inspired by [7].

Next we talk about "full cube-term blocker clones", studied in [13], which are interesting for classification of non-finitely related clones and are themselves non-finitely related (we explain this notion later). We show that they are finitely generated.

The clone of operations with restricted image

For a nonempty finite set $B \subsetneq A$ we define a clone by

$$\mathcal{C}_B = \{f : A^n \to A \,|\, n \in \mathbb{N},\ \mathrm{Im}(f) \subseteq B\} \cup \mathcal{P}_A.$$

The set $\mathcal{C}_B$ is indeed a clone, as if we take $f, g_1, \dots, g_n \in \mathcal{C}_B$ and the superposition $h(\mathbf{x}) = f(g_1(\mathbf{x}), \dots, g_n(\mathbf{x}))$, then either f is a projection onto the j-th variable, and we have $h = g_j \in \mathcal{C}$, or $\mathrm{Im}(f) \subseteq B$, and then we have also $\mathrm{Im}(h) \subseteq B$.

Although $\mathcal{C}_B$ does not contain any near-unanimity operation, it does contain a B-near-unanimity operation, so we know that the clone is finitely generated by Proposition 6. We can, however, modify the above construction in a very straightforward way to explicitly show that $\mathcal{C}_B$ is generated by its unary and binary operations and is, therefore, finitely generated.

We fix $0 \in B$ and define unary operations $u_{a\to b}$ for $a \in A$, $b \in B$ as before. We need to have $b \in B$ to have the image of the operation in B. But we only need operations of the form $u_{a\to f(\mathbf{a})}$ for a tuple $\mathbf{a}$ and an operation f with $\mathrm{Im}(f) \subseteq B$, which satisfy this condition.

The image of conjunction and disjunction as defined before is not in B but, we do not need to pass elements from $A \setminus B$ through them, so we take a slightly modified version where we just return 0 if we would normally return anything outside B, i.e.,

$$\wedge(x, y) = \begin{cases} x & \text{if } x = y \text{ and } x \in B, \\ 0 & \text{otherwise} \end{cases},$$

$$\vee(x, y) = \begin{cases} x & \text{if } (y = 0 \text{ or } x = y) \text{ and } x \in B, \\ y & \text{if } (x = 0 \text{ or } x = y) \text{ and } y \in B, \\ 0 & \text{otherwise.} \end{cases}$$

For any $\mathbf{a} \in A^n$ and $b \in B$ we can now construct

$$g_{\mathbf{a}\to b}(\mathbf{x}) = u_{a_1\to b}(x_1) \wedge \cdots \wedge u_{a_n\to b}(x_n),$$

and, as before, for any $f \in \mathcal{C}_B$ which is not a projection we get $f = \bigvee\limits_{\mathbf{a}\in A} g_{\mathbf{a}\to f(\mathbf{a})}$.

A union of clones of operations with restricted image

If we take a finite set A and several subsets $B_1, \dots, B_k \subsetneq A$, then we know that $\mathcal{C}_{B_i}$ is a finitely generated clone for all $i = 1, \dots, k$. Let us consider

$$\mathcal{C} = \mathcal{C}_{B_1} \cup \mathcal{C}_{B_2} \cup \cdots \cup \mathcal{C}_{B_k}.$$

First of all, $\mathcal{C}$ is a clone. Take $f, g_1, \dots, g_n \in \mathcal{C}$ and $h(\mathbf{x}) = f(g_1(\mathbf{x}), \dots, g_n(\mathbf{x}))$. Then either f is a projection, and we have $h = g_j \in \mathcal{C}$ for some j, or $f \in \mathcal{C}_{B_i}$ for some i, and then $\mathrm{Im}(f) \subseteq B_i$, hence also $\mathrm{Im}(h) \subseteq B_i$.

Now it is clear that $\mathcal{C}$ is finitely generated, because it is generated by the union of the finite sets of generators of $\mathcal{C}_{B_1}, \dots, \mathcal{C}_{B_k}$. We can also finish the construction by taking "translator functions", which is a concept that will be useful later.

Let us assume that B_1 is the largest of the subsets with respect to the number of elements. Then for each i we have a unary operation $t_i \in \mathcal{C}_{B_i}$ which "translates" the elements of B_1 to elements of B_i, meaning that the restriction $t\restriction_{B_1}: B_1 \to B_i$ is surjective. But then for any operation $f \in \mathcal{C}_{B_i}$ we have an operation $\tilde{f} \in \mathcal{C}_{B_1}$ such that $f = t_i(\tilde{f})$, hence $\mathcal{C}$ is generated by the generators of $\mathcal{C}_{B_1}$ and one unary translator operation t_i for each i.

The clone of B-insensitive operations

Let $B \subsetneq A$ be again nonempty finite sets. We say that f is B-insensitive if it cannot distinguish between different elements of B on the input, i.e., if

$$\forall i\ \forall \mathbf{a} \in A^n\ \forall b, b' \in B:$$
$$f(a_1, \dots, a_{i-1}, b, a_{i+1}, \dots, a_n) = f(a_1, \dots, a_{i-1}, b', a_{i+1}, \dots, a_n).$$

Let $\sim_B$ be the equivalence on A such that $a \sim_B a'$ if either $a = a'$ or $a, a' \in B$. For $\mathbf{a}, \mathbf{a}' \in A^n$ we write $\mathbf{a} \sim_B \mathbf{a}'$ if $a_i \sim_B a'_i$ for every $i \in \{1, \dots, n\}$. Now we can also say that f is B-insensitive iff $f(\mathbf{a}) = f(\mathbf{a}')$ whenever $\mathbf{a} \sim_B \mathbf{a}'$.

We will now study the set

$$\mathcal{C} = \{f : A^n \to A \mid n \in \mathbb{N},\ f \text{ is } B\text{-insensitive}\} \cup \mathcal{P}_A.$$

We need to add projections separately, since they are not B-insensitive.

To show that $\mathcal{C}$ is a clone, take $f, g_1, \dots, g_n \in C$ and consider the operation $h(\mathbf{x}) = f(g_1(\mathbf{x}), \dots, g_n(\mathbf{x}))$. If f is a projection, then $h = g_j \in \mathcal{C}$ for some j. Otherwise we show that h is B-insensitive.

Let $\mathbf{a} \sim_B \mathbf{a}'$. Now for each $i \in \{1, \dots, n\}$ either g_i is B-insensitive, and than the input to f is the same for $\mathbf{a}$ and $\mathbf{a}'$, or g_i is the projection onto the j-th variable, and than $g_i(\mathbf{a}) \sim_B g_i(\mathbf{a}')$, so switching from $\mathbf{a}$ to $\mathbf{a}'$ in g_i will not change the result, because f is B-insensitive. Hence $h(\mathbf{a}) = h(\mathbf{a}')$.

Showing that $\mathcal{C}$ is finitely generated by our explicit construction is somewhat tricky, since we can not directly pass elements from B through the operations of $\mathcal{C}$ as before. We need to incorporate some kind of coding into the construction to pass the information about which element from B should we output. There is no problem when the operation outputs an element from $A \setminus B$, but to make the construction clearer, we will encode all the elements of A. It is essential here to have some element outside of B to our disposal.

Let us fix $0 \in A$ and $0 \neq 1 \in A \setminus B$, and let $A = \{e_0, e_1, \dots, e_{k-1}\}$.

The unary operations "$u_{a \to b}$" will now only check if the input is right, but will not tell anything about the output. That is, we will only use operations $u_{a \to 1}$ defined by

$$u_{a \to 1}(x) = \begin{cases} 1 & \text{if } x \sim_B a, \\ 0 & \text{otherwise.} \end{cases}$$

For $b \in B$ the operation $u_{b \to 1}$ cannot distinguish between different elements of B, but neither can the operations we want to generate, so this is not a problem.

The conjunction and disjunction can be quite simple. We want them to be a regular conjunction and disjunction on $0, 1$ and they can behave arbitrarily on the other elements, since we never input any other elements in them. A natural way of defining them to make sure that they are in $\mathcal{C}$ is to let them "see" all elements in B as 0 and all the other elements as 1, i.e.,

$$\wedge(x,y) = \begin{cases} 1 & \text{if } x, y \notin B, \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad \vee(x,y) = \begin{cases} 1 & \text{if } x \notin B \text{ or } y \notin B, \\ 0 & \text{otherwise.} \end{cases}$$

Together we can compose $g_{\mathbf{a}\to 1}(\mathbf{x}) = u_{a_1\to 1}(x_1) \wedge \cdots \wedge u_{a_n \to 1}(x_n)$ so that

$$g_{\mathbf{a}\to 1}(\mathbf{x}) = \chi_{\mathbf{x}\sim_B \mathbf{a}}(\mathbf{x}) = \begin{cases} 1 & \text{if } \mathbf{x} \sim_B \mathbf{a}, \\ 0 & \text{otherwise.} \end{cases}$$

Now to the encoding. Let l be fixed such that $k \leq 2^l$. For $b_0, b_1, \ldots, b_l \in \{0,1\}$ we denote by $\overline{b_0 b_1 \ldots b_{l-1}}$ the number with a binary representation $b_0 b_1 \ldots b_{l-1}$. We define unary encoders c_i^j and l-ary decoder dec such that $\operatorname{dec}(c_0^j, c_1^j, \ldots, c_{l-1}^j)$ evaluates to a_j on $(1, \ldots, 1)$. We achieve this by letting

$$c_i^j(x) = \begin{cases} \text{the } i\text{-th digit of binary representation of } a_j & \text{if } x = 1, \\ 0 & \text{otherwise,} \end{cases}$$

and

$$\operatorname{dec}(x_0, \ldots, x_{l-1}) = a_{\overline{y_0, y_1, \ldots, y_l}}, \text{where } y_i = \chi_{x \notin B}(x_i) = \begin{cases} 1 & \text{if } x_i \notin B, \\ 0 & \text{if } x_i \in B. \end{cases}$$

There are actually only two coding operations: either it is a constant 0, or it maps 1 to 1 and anything else to 0. Note that all the defined operations are B-insensitive and that we have only one dec operation of a fixed arity l, so we have still defined only finitely many operations.

Let now f be an n-ary B-insensitive operation on A. We will construct f using the defined operations. The construction is illustrated in Figure 2.2.

Explicitly we will have $f = \operatorname{dec}(\operatorname{dig}_0^f, \operatorname{dig}_1^f, \ldots, \operatorname{dig}_l^f)$, where

$$\operatorname{dig}_i^f(\mathbf{x}) := \bigvee_{j \in \{0,1,\ldots,k-1\}} \left(c_i^j \;(\bigvee_{\mathbf{a}: f(\mathbf{a}) = e_j} g_{\mathbf{a}\to 1}(\mathbf{x})) \right).$$

What happens here is that given a tuple $\mathbf{a}$ such that $f(\mathbf{a}) = e_r$, 1 will be inputed into $c_0^r, \ldots, c_{l-1}^r$, and 0 will be inputed into every other c_i^j. This is true, because for all $\mathbf{b}$ such that $f(\mathbf{b}) \neq e_r$ we have necessarily $\mathbf{b} \not\sim_B \mathbf{a}$, and thus $g_{\mathbf{b}\to 1}(\mathbf{a}) = 0$. Therefore, for this $\mathbf{a}$ we have

$$\operatorname{dig}_i^f(\mathbf{a}) = c_i^r(1) \vee \bigvee_{j \neq r} c_i^j(0) = c_i^r(1) \vee \bigvee_{j\neq r} 0 = c_i^r(1),$$

i.e., all together

$$\operatorname{dec}(\operatorname{dig}_0^f, \operatorname{dig}_1^f, \ldots, \operatorname{dig}_{l-1}^f)(\mathbf{a}) = \operatorname{dec}(c_0^r(1), c_1^r(1), \ldots, c_{l-1}^r(1)) = e_r = f(\mathbf{a}),$$

which is what we wanted.

Let us remark that we could use any 01-encoding of the elements, which might be useful if we wanted to modify the construction for different clones. Another simple encoding we could use here is $e_i \longleftrightarrow (0, \ldots, 0, \overset{i-1}{1}, 0, \ldots, 0) \in \{0,1\}^k$.

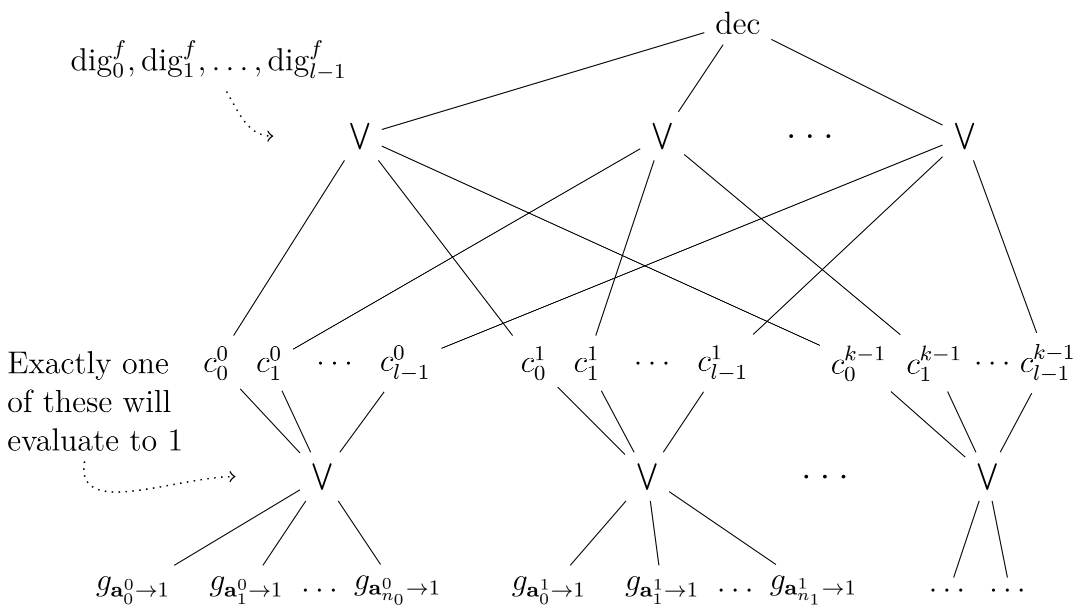

dec
$\mathrm{dig}_0^f, \mathrm{dig}_1^f, \dots, \mathrm{dig}_{l-1}^f$
$\bigvee$
$\bigvee$
$\cdots$
$\bigvee$
Exactly one of these will evaluate to 1
c_0^0 c_1^0 $\cdots$ c_{l-1}^0
c_0^1 c_1^1 $\cdots$ c_{l-1}^1
c_0^{k-1} c_1^{k-1} $\cdots$ c_{l-1}^{k-1}
$\bigvee$
$\bigvee$
$\cdots$
$\bigvee$
$g_{\mathbf{a}_0^0\to 1}$ $g_{\mathbf{a}_1^0\to 1}$ $\cdots$ $g_{\mathbf{a}_{n_0}^0\to 1}$
$g_{\mathbf{a}_0^1\to 1}$ $g_{\mathbf{a}_1^1\to 1}$ $\cdots$ $g_{\mathbf{a}_{n_1}^1\to 1}$
$\cdots$ $\cdots$

Figure 2.2: The construction of a given f from the defined operations. Each of the bottom blocks belongs to tuples on which f evaluates to a certain value, i.e., $\{\mathbf{a}_0^j, \mathbf{a}_1^j, \dots, \mathbf{a}_{n_j}^j\}$ are exactly the tuples for which $f(\mathbf{a}_i^j) = e_j$.

B-insensitive operations with image restricted by set(s) different than B

Let $A, B, 0, 1$ and $\mathcal{C}$ be as above and let $D \subseteq A$ be such that $0, 1 \in D$. Except for the decoder, all the operations defined in the construction have image in $\{0, 1\} \subseteq D$. Therefore, when we restrict the decoder so that it only returns a_i if $a_i \in D$ and returns 0 otherwise, we immediately see that the clone

$$\begin{aligned}\mathcal{D}_D &= \{f \in \mathcal{C} \mid \mathrm{Im}(f) \subseteq D\} \cup \mathcal{P}_A \\ &= \{f : A^n \to A \mid n \in \mathbb{N}, f \text{ is } B\text{-insensitive and } \mathrm{Im}(f) \subseteq D\} \cup \mathcal{P}_A\end{aligned}$$

is finitely generated. The essential fact here is that $D \not\subseteq B$.

Note that if we would have $|B \cap D| \leq 1$, the construction would be much simpler. We would not need any encoding, since we could pass all needed elements through conjunction and disjunction. The construction for clones with just restricted image would suffice with a slight modification to make sure all the operations are B-insensitive – they would just treat the whole B as one element.

For example if $A = \{0, 1, 2\}, B = \{0, 1\}$ and $D = \{0, 2\}$, we get a clone used in [7]

$$\mathcal{D}_{02} = \{f \text{ operation on } \{0,1,2\} \mid f \text{ is } \{0,1\}\text{-insensitive, } \mathrm{Im}(f) \subseteq \{0, 2\}\} \cup \mathcal{P}_A,$$

which is thereby finitely generated.

We can now go one step further. Let A be a finite set, $B, D_1, \dots, D_k \subseteq A$ be nonempty such that $0, 1 \in D_1$ and $1 \notin B$, that is, in particular $D \not\subseteq B$. Let

$$\mathcal{D}_{D_j} = \{f : A^n \to A \mid n \in \mathbb{N},\ f \text{ is } B\text{-insensitive and } \mathrm{Im}(f) \subseteq D_j\}.$$

Then

$$\mathcal{D} = \mathcal{D}_{D_1} \cup \mathcal{D}_{D_2} \cup \cdots \cup \mathcal{D}_{D_k}$$

is a finitely generated clone. It is easy to see that $\mathcal{D}$ is a clone. We can use the same arguments we used for the clone of B-insensitive operations and the union of clones of operations with restricted image. Note, however, that we allow $D_j \subseteq B$ for $j \neq 1$. That means we do not know if $\mathcal{D}_{D_2}, \ldots, \mathcal{D}_{D_k}$ are finitely generated. Later we show that if $D_j \subseteq B$, then $\mathcal{D}_{D_j}$ is actually non-finitely generated.

To see that $\mathcal{D}$ is finitely generated, recall the "translator" unary operations we used for the union of clones of operations with restricted image. We could still use the same trick if $|D_1 \setminus B| \geq |D_j|$ for all $j \neq 1$. In general we do something similar. We just take a different decoder for each j. That is, for a given n-ary operation $f \in \mathcal{D}_{D_j}$, we will first map $\mathbf{a} \in A^n$ to the 01-encoding of $f(\mathbf{a})$, using the operations from $\mathcal{D}_{D_1}$, and then decode them to get the desired result by the decoder from $\mathcal{D}_{D_j}$.

The full cube-term blocker clone

For a clone $\mathcal{C}$, we ask whether there exist finitely many operations generating $\mathcal{C}$. Similarly, we can ask whether we can describe it in the terms of compatibility using only finitely many relations. We say that $\mathcal{C}$ is *finitely related* if there exists a finite set of relations $\mathcal{R}$ such that $\mathcal{C} = \operatorname{Inv}(\mathcal{R})$. Note that by Lemma 5, all clones containing a near-unanimity operation are finitely related.

Finitely related clones are studied in [13], using, among other concepts, so called *cube-term blockers*. For a clone $\mathcal{C}$ on A and subsets $\varnothing \neq D \subsetneq S \subseteq A$, we say that $\mathcal{C}$ has cube-term blocker (D, S) if all operations $f \in \mathcal{C}$ satisfy

$$\exists i: \ f(S, \ldots, S, \overset{i}{D}, S, \ldots, S) \subseteq D. \tag{2.2}$$

It is proved that the clone of all such operations, for a fixed (D, S), is non-finitely related. The paper is mainly concerned with the idempotent case. We say an operation is *idempotent*, if $f(a, \ldots, a) = a$ for every $a \in A$. A bit simplified, one of the main results in [13] is that the clones of all idempotent operations satisfying (2.2) for some $\varnothing \neq D \subsetneq S \subseteq A$ are maximal non-finitely related idempotent clones, in a sense that if we added another idempotent operation, the resulting clone would be finitely related.

We fix $B \subsetneq A$, and consider the clone of all operations satisfying (2.2) for (B, A). We call it the *full cube-term blocker clone*. We show that both the full cube-term blocker clone and its full idempotent reduct are finitely generated.

Let us denote the full cube-term blocker clone as

$$\mathcal{B} = \left\{ f : A^n \to A \,|\, n \in \mathbb{N},\ \exists i : f(A, \ldots, A, \overset{i}{B}, A, \ldots, A) \subseteq B \right\}.$$

It is easy to see that $\mathcal{B}$ is, indeed, a clone. The projections are in $\mathcal{B}$ and when we compose $f(g_1, \ldots, g_n)$ we can easily find a variable with the desired property. If f fulfills the condition in the i-th variable and g_i in the j-th variable, then the superposition fulfills it in the j-th variable. We now aim to modify our construction to show it is finitely generated. As before, we fix $0 \in B$.

The first problem we might notice is that all unary operations $u \in \mathcal{B}$ fulfill $u(B) \subseteq B$, which is inconvenient since we can have, for instance, f such that $f(a, b \dots, b) = a$ for some $b \in B, a \notin B$, and, therefore, we need $u_{b \to a}$.

The second problem is that $\vee \notin \mathcal{B}$ for $\vee$ as in (2.1), since for $a \notin B$ we have $a \vee 0 = 0 \vee a = a \notin B$, but $0 \in B$.

We fix both issues by introducing a new variable to the operations which will satisfy the desired condition. We will denote $\wedge$ and $\vee$ as in (2.1). We will use $\wedge$ as it is; there is no issue with it. We define new operations

$$\tilde{\vee}(x, y, z) = \begin{cases} \vee(x, y) & \text{if possible, i.e., if } z \notin B \text{ or } \vee(x, y) \in B, \\ 0 & \text{otherwise } (0 \in B), \end{cases}$$

$$\tilde{u}_{a \to b}(x, z) = u_{a \to b}(x) \qquad \text{for } b \in B,$$

$$\tilde{u}_{a \to b}(x, z) = \begin{cases} u_{a \to b}(x) & \text{if } z \notin B, \\ 0 & \text{otherwise } (0 \in B) \end{cases} \qquad \text{for } b \notin B.$$

The new operation $\tilde{\vee}$ being ternary, it is now not clear what do we mean by an "n-ary version of $\tilde{\vee}$". When composing $\tilde{\vee}$ we will plug the same variable into the third coordinate of all $\tilde{\vee}$'s composed. For a given n, we thus obtain an $(n+1)$-ary operation

$$\begin{aligned} \tilde{\bigvee}(x_1, x_2, \dots, x_n, z) &:= \tilde{\vee}(\tilde{\vee}(\dots \tilde{\vee}(\tilde{\vee}(x_1, x_2, z), x_3, z), \dots, z), x_n, z) \\ &= \begin{cases} \bigvee(\mathbf{x}) & \text{if possible, i.e., if } z \notin B \text{ or } \bigvee(\mathbf{x}) \in B, \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

Analogous to the full clone case we also denote $\tilde{\bigvee}(x_1, x_2, z) = \tilde{\vee}(x_1, x_2, z)$ and $\tilde{\bigvee}(x_1, z) = \pi^2_1$.

$$\tilde{\bigvee}(x_1, x_2, \dots, x_n, z) =$$

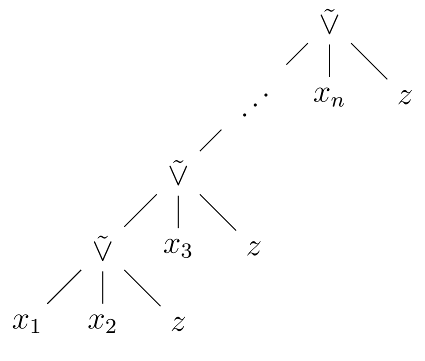


Figure 2.3: Tree depiction of how to compose $\tilde{\bigvee}$.

Let now f be an n-ary operation in $\mathcal{B}$ with the i-th variable satisfying the property $\forall \mathbf{a} \in A^n : a_i \in B \Rightarrow f(\mathbf{a}) \in B$. Let $\mathbf{a} \in A^n$. We will exploit the fact that $f(\mathbf{a}) \notin B \Rightarrow a_i \notin B$ to define n-ary operations $g_{\mathbf{a} \to f(\mathbf{a})}$ as

$$\begin{aligned} g_{\mathbf{a} \to f(\mathbf{a})}(x_1, \dots, x_n) &:= \tilde{u}_{a_1 \to f(\mathbf{a})}(x_1, x_i) \wedge \tilde{u}_{a_2 \to f(\mathbf{a})}(x_2, x_i) \wedge \dots \wedge \tilde{u}_{a_n \to f(\mathbf{a})}(x_n, x_i) \\ &= \begin{cases} f(\mathbf{a}) & \text{if } \mathbf{x} = \mathbf{a}, \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

This construction works because for $f(\mathbf{a}) \in B$ everything is defined as in the full clone case and for $f(\mathbf{a}) \notin B$ the operations $\tilde{u}_{a_j \to f(\mathbf{a})}$ have the element $a_i \notin B$ at disposal in the troublesome second coordinate so they have the freedom to output whatever $f(\mathbf{a})$ is.

The same trick used for $\tilde{\bigvee}$ suffices to generate f as

$$f(\mathbf{x}) = \tilde{\bigvee}(g_{\mathbf{a}_1 \to f(\mathbf{a}_1)}(\mathbf{x}), g_{\mathbf{a}_2 \to f(\mathbf{a}_2)}(\mathbf{x}), \ldots, g_{\mathbf{a}_{|A|^n} \to f(\mathbf{a}_{|A|^n})}(\mathbf{x}), x_i),$$

where $a_1, \ldots, a_{|A|^n}$ are all the tuples of A^n.

The idempotent reduct of the full cube-term blocker clone

For finite nonempty sets $B \subsetneq A$ and $\mathcal{B}$ as before, we set $\tilde{\mathcal{B}}$ to be the full idempotent reduct of $\mathcal{B}$, that is,

$$\tilde{\mathcal{B}} = \{f \in \mathcal{B} \,|\, \forall a \in A : f(a, \ldots, a) = a\} .$$

Note that idempotence is consistent with the defining property of $\mathcal{B}$ and $\tilde{\mathcal{B}}$ is therefore not trivial.

The operations $\wedge$ and $\tilde{\vee}$ as defined above are already idempotent, so we only need to modify the operations $\tilde{u}_{a \to b}$ by introducing another variable. We need to distinguish two cases here – $|B| = 1$ and $|B| \geq 2$.

For the first case let 0 be the only element of B and let us define

$$\tilde{\tilde{u}}_{a \to b}(x, z, w) = \begin{cases} x & \text{when forced by idempotence, i.e., if } x = z = w, \\ \tilde{u}_{a \to b}(x, z) & \text{otherwise} \end{cases}$$

Given n-ary $f \in \tilde{\mathcal{B}}$ with the i-th variable fulfilling the property for $\mathcal{B}$, we plug in x_i for z, as before, and we plug in $\bigwedge\limits_j x_j$ for w. For $\mathbf{a} \in A^n$ we obtain

$$\begin{aligned} g_{\mathbf{a} \to f(\mathbf{a})}(x_1, \ldots, x_n) :=& \\ & \tilde{\tilde{u}}_{a_1 \to f(\mathbf{a})}(x_1, x_i, \bigwedge_j x_j) \wedge \tilde{\tilde{u}}_{a_2 \to f(\mathbf{a})}(x_2, x_i, \bigwedge_j x_j) \wedge \cdots \wedge \tilde{\tilde{u}}_{a_n \to f(\mathbf{a})}(x_n, x_i, \bigwedge_j x_j) \\ &= \begin{cases} f(\mathbf{a}) & \text{if } \mathbf{x} = \mathbf{a}, \\ x_1 & \text{if } \mathbf{x} \text{ is constant, i.e., } x_1 = x_2 = \cdots = x_n, \\ 0 & \text{otherwise.} \end{cases} \end{aligned}$$

This indeed works because for a non-constant tuple $\mathbf{c} \in A^n$ with $c_i \neq 0$ we have two different inputs in $\tilde{\tilde{u}}$'s, $c_i \neq 0$ and $\bigwedge c_j = 0$, so $\tilde{\tilde{u}}$'s are not bound by the idempotency and may output whatever needed. For $\mathbf{c}$ with $c_i = 0$ the result of every used operation is 0, but also $f(\mathbf{c}) = 0$. For constant tuples $\mathbf{c} = (c, \ldots, c)$ everything outputs c, but also $f(c) = c$.

For the case $|B| \geq 2$ we have a problem for such f that fulfills the property for $\mathcal{B}$ in the first coordinate, and $f(0, a) = b \neq 0$. We would have $\tilde{\tilde{u}}_{0 \to b}(x_1, x_1, x_1 \wedge x_2)$ and for the tuple $(0, a)$ we would be forced to output $\tilde{\tilde{u}}_{0 \to b}(0, 0, 0) = 0$, which is not what we need.

We will fix this by adding yet another variable. We fix two different zeros $0, 0' \in B$ and define a second conjunction $\wedge'$ which is the same as $\wedge$, only it uses

$0'$ instead of 0. This allows us to have guaranteed two different elements $\bigwedge a_j$ and $\bigwedge' a_j$ for every non-constant $\mathbf{a}$. We then define

$$\tilde{\tilde{u}}_{a\to b}(x, z, w, w') =$$
$$= \begin{cases} x & \text{when forced by idempotence, i.e., if } x = z = w = w', \\ \tilde{u}_{a\to b}(x, z) & \text{otherwise} \end{cases}$$

and for a given $f \in \tilde{\mathcal{B}}$ with the special coordinate at i we take

$$g_{\mathbf{a}\to f(\mathbf{a})}(\mathbf{x}) := \tilde{\tilde{u}}_{a_1\to f(\mathbf{a})}(x_1, x_i, \bigwedge_j x_j, \bigwedge_j{}' x_j) \wedge \cdots \wedge \tilde{\tilde{u}}_{a_n\to f(\mathbf{a})}(x_n, x_i, \bigwedge_j x_j, \bigwedge_j{}' x_j).$$

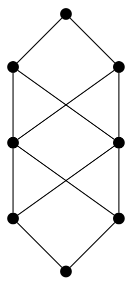
In both cases we finish the construction the same way as before by taking the generalized disjunction $\tilde{\bigvee}$ over all tuples $\mathbf{a} \in A^n$.

3. Non-finitely generated clones

In this chapter we look at examples of non-finitely generated clones. We show how B-insensitivity together with image restricted by B can yield non-finitely generated clones, as essentially shown in [7]. Inspired by the example, we find relations such that operations preserving them depend only on a limited number of variables. The construction yields non-finitely generated clones. Then we discuss possible modifications and generalizations of these relations.

3.1 Basic examples of non-finitely generated clones

Both the trivial and full clone are finitely generated. By the deep result of Tardos [18], a maximal clone can be non-finitely generated. Namely, the maximal clone of monotone operations of the following poset is non-finitely generated:



On the other hand, a minimal clone is always finitely generated, since minimal clones are always generated by one operation. Somewhat minimalistic example of a non-finitely generated clone can be constructed on $\mathbb{Z}_4$.

Non-finitely generated clones on $\mathbb{Z}_4$

The following are well known examples of non-finitely clones, see e.g. [3, 8, 14]. Consider {0,1,2,3} with standard $+, \cdot$ evaluated modulo 4. The clone generated by operations

$$f_k(x_1, x_2, \ldots, x_k) = 2 \cdot x_1 \cdot x_2 \cdot \cdots \cdot x_k.$$

is non-finitely generated. To see that, we just need to observe that f_n can not be composed using f_i's for $i < n$. This is clear, because composing f_i and f_j we always get a constant 0 – for example, substituting for the first variable, we get

$$f_i(f_j(x_1, \ldots, x_j), x_{j+1}, \ldots, x_{j+i}) = 2 \cdot 2 \cdot x_1 \cdot \cdots \cdot x_{j+1} = 0.$$

More generally, we can similarly take operations $p \cdot x_1 \cdot \cdots \cdot x_n$ in $\mathbb{Z}_{p^2}$, for any prime p. If we also include addition, the clone $\mathrm{Clo}(\{f_n \mid n \in \mathbb{N}\} \cup \{+\})$ is still non-finitely generated. The proof is, however, not that straightforward anymore.

We can also consider

$$g_n(x_1, x_2, \ldots, x_n) = 2 \cdot (x_1 + x_2 + \cdots + x_n) = 2 \cdot x_1 + 2 \cdot x_2 + \ldots 2 \cdot x_n.$$

Composing g_i and g_j we do not get a constant operation, but we still can not compose these into g_n for $n > i, j$, as for example

$$\begin{aligned} & g_i(x_1, \dots, x_{i-1}, g_j(y_1, \dots, y_j)) \\ & \qquad = 2 \cdot x_1 + \cdots + 2 \cdot x_{i-1} + 2 \cdot (2 \cdot y_1 + \cdots + 2 \cdot y_j) \\ & \qquad = 2 \cdot x_1 + \cdots + 2 \cdot x_{i-1} \\ & \qquad \equiv g_{i-1}(x_1 \dots, x_{i-1}). \end{aligned}$$

Using the alternative definition of a clone (Definition 2), it is not difficult to check that the operations in $\mathrm{Clo}(\{g_i \mid i \leq k\})$ are either projections or of the form

$$f(x_1, x_2, \dots, x_l) = g_j(x_{i_1}, x_{i_2}, \dots, x_{i_j}),$$

where $j \leq l$, $i_1, \dots, i_j \in \{1 \dots, l\}$. As a consequence, $g_n \notin \mathrm{Clo}(\{g_i \mid i \leq k\})$ for $n > k$, and hence $\mathrm{Clo}(\{g_n \mid n \in \mathbb{N}\})$ is non-finitely generated.

Note two important properties of the operations g_k. First, their image is in $\{0, 2\}$, and secondly, it does not matter if we plug 0 or 2 into the variables, i.e., the operations are 02-insensitive. We will later see that this combination does in general yield non-finitely generated clones.

An intersection of finitely generated clones

L. Haddad shows in [7] that an intersection of two finitely generated clones can be non-finitely generated by a following example. Let $\underline{\mathbf{3}} = \{0, 1, 2\}$ and

$$\begin{aligned} \mathcal{D}_{ij} &:= \{f : \underline{\mathbf{3}}^n \to \underline{\mathbf{3}} \mid n \in \mathbb{N},\ f \text{ is 01-insensitive and } \mathrm{Im}(f) \subseteq \{i, j\}\} \cup \mathcal{P}_A, \\ \mathcal{C}_{01} &:= \{f : \underline{\mathbf{3}}^n \to \underline{\mathbf{3}} \mid n \in \mathbb{N},\ \mathrm{Im}(f) \subseteq \{0, 1\}\} \cup \mathcal{P}_A, \\ \mathcal{D} &:= \mathcal{D}_{02} \cup \mathcal{D}_{01}. \end{aligned}$$

As shown in Section 2.2, $\mathcal{D}_{02}, \mathcal{D}$ and $\mathcal{C}_{01}$ are finitely generated clones. However

$$\mathcal{D}_{01} = \mathcal{D} \cap \mathcal{C}_{01}$$

is non-finitely generated, as follows from Proposition 12 in Section 3.3, where we will show that the clones of all B-insensitive operations with image in B, for some $B \subsetneq A$, are in general non-finitely generated.

We have also shown in Section 2.2 that

$$\mathcal{E}_{01} = \{f : \underline{\mathbf{3}}^n \to \underline{\mathbf{3}} \mid n \in \mathbb{N},\ f \text{ is 01-insensitive}\} \cup \mathcal{P}_A$$

is a finitely generated clone. So we have an even more straightforward intersection

$$\mathcal{D}_{01} = \mathcal{C}_{01} \cap \mathcal{E}_{01}.$$

3.2 Essential and insensitive operations

As described in Section 1.4, a clone $\mathcal{C}$ is non-finitely generated if and only if we can find an infinite chain of clones, union of which is $\mathcal{C}$. The arity of operations we add in each step has to gradually increase, since there is only finitely many operations of a given arity. It would, therefore, be natural to try taking all operations of given arity in each step. The problem is that they never form a clone, because we can always introduce new fictitious variables, and somewhat artificially increase arity of any operation. So what we are interested in is not the arity, but essential arity.

Definition 5. *Let A be a set and $f : A^n \to A$ an operation. We say that f is* essential *in the i-th variable, or that it* depends *on the i-th variable, if*

$$\exists \mathbf{a} \in A^n \ \exists b_i \in A : \quad f(a_1, \ldots, a_{i-1}, a_i, a_{i+1}, \ldots, a_n) \neq f(a_1, \ldots, a_{i-1}, b_i, a_{i+1}, \ldots, a_n),$$

that is, if the i-th variable is not fictitious.

We say that f is essentially k-ary *(the* essential arity *of f is k, f is* k-essential*) if it has exactly k essential variables.*

Let $\mathcal{C}$ be a clone on A. We define the k-essential part *of $\mathcal{C}$ as the set of all essentially at most k-ary operations in $\mathcal{C}$.*

Yet another way to formulate that a variable is essential is to say it can distinguish between some two elements. For sets $B \subsetneq A$ we can also introduce a notion of *B-essential* variable, which is a stronger property. Not only can the variable distinguish between a pair of elements, it can distinguish between two elements from B. The negation of that is the already mentioned B-insensitivity.

Definition 6. *Let $B \subseteq A$ be sets and $f : A^n \to A$ an operation. We say that f is* B-insensitive in the i-th variable *if*

$$\forall \mathbf{a} \in A^n \ \forall b_i, \tilde{b_i} \in B : \quad f(a_1, \ldots, a_{i-1}, b_i, a_{i+1}, \ldots, a_n) = f(a_1, \ldots, a_{i-1}, \tilde{b_i}, a_{i+1}, \ldots, a_n),$$

i.e., if f can not distinguish between different elements of B in the i-th variable.

We say that f is B-insensitive*, if it is B-insensitive in all variables.*

3.3 B-insensitive operations with image in B

One of the reasons a clone might be non-finitely generated is that its operations can not be composed to get an operation of higher essential arity. An easy way to guarantee this is to make sure the operations can not distinguish between the results they can output.

The following proposition is essentially contained in [7].

Proposition 12. *Let $B \subsetneq A$ be nonempty finite sets, $\tilde{\mathcal{C}}$ be a clone on A and*

$$\mathcal{C} = \left\{ f \in \tilde{\mathcal{C}} \,|\, f \text{ is } B\text{-insensitive and } \mathrm{Im}(f) \subseteq B \right\} \cup \mathcal{P}_A.$$

Then $\mathcal{C}$ is a clone, and if it contain operations of arbitrarily high essential arity, it is non-finitely generated.

Proof. First of all, $\mathcal{C}$ is a clone. We know that composing operations from $\tilde{\mathcal{C}}$, we obtain again operations in $\tilde{\mathcal{C}}$, so it is enough to prove that composing operations from $\mathcal{C}$ we get either a projection or a B-insensitive operation with the image in B. We check this for the four procedures from Definition 2.

It is clear the properties are preserved while permuting the variables or introducing new fictitious variables. It is also preserved by identification of the first two variables, since for an n-ary operation $f \in \mathcal{C}$ which is not a projection we have

$$\forall \mathbf{a} \in A^n \ \forall b, \tilde{b} \in B : f(b, b, a_3, \ldots, a_n) = f(b, \tilde{b}, a_3, \ldots, a_n) = f(\tilde{b}, \tilde{b}, a_3, \ldots, a_n),$$

so the glued variable is still B-insensitive.

Let now $f, g \in \mathcal{C}$ and

$$(f * g)(x_1, \ldots, x_{m+n-1}) = f(g(x_1, \ldots, x_m), x_{m+1}, \ldots, x_{m+n-1}).$$

If f is a projection, then $f * g$ is either also a projection or it is g with added fictitious variables. Assume that f is not a projection. Clearly, $\mathrm{Im}(f * g) \subseteq B$, and $x_{m+1}, \ldots, x_{m+n-1}$ are B-insensitive. For the first m variables, they are B-insensitive either because g is B-insensitive, or because f is B-insensitive in the first variable. Thus, $\mathcal{C}$ is a clone.

Let us denote

$$\mathcal{C}_k = \{f \in \mathcal{C} \mid f \text{ is essentialy at most } k\text{-ary}\}.$$

We assume that $\mathcal{C}$ contains an operation of arbitrarily high essential arity, hence we have an infinite strictly increasing sequence $k_1, k_2, \ldots$ such that there exists an essentially k_i-ary operation in $\mathcal{C}$ for every $i \in \mathbb{N}$. This yields a strictly increasing chain

$$\mathcal{C}_{k_1} \subsetneq \mathcal{C}_{k_2} \subsetneq \cdots \subsetneq \mathcal{C}_{k_i} \subsetneq \cdots \subsetneq \mathcal{C}.$$

Clearly $\bigcup_{i \in \mathbb{N}} \mathcal{C}_{k_i} = \mathcal{C}$. We show that $\mathcal{C}_k$ is a clone for every $k \in \mathbb{N}$, and hence $\mathcal{C}$ is non-finitely generated.

Let $k \in \mathbb{N}$. Projections are essentially unary, so they are in $\mathcal{C}_k$. Neither permutation and identification of variables, nor introduction of new fictitious variables increases the essential arity. We only need to check the substitution.

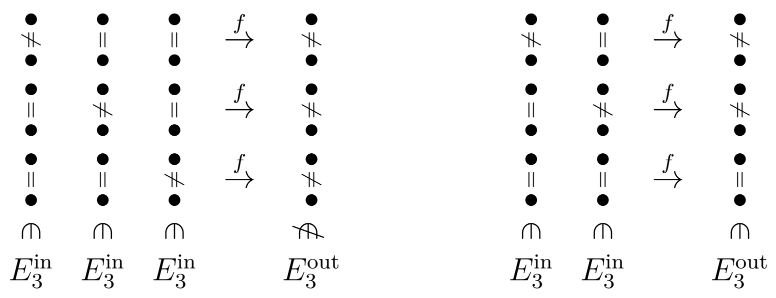
Let $f, g \in \mathcal{C}$ be essentially at most k-ary, $\mathrm{ar}(f) = n$, $\mathrm{ar}(g) = m$. If either f or g is a projection, then $f * g$ is still essentially at most k-ary. If they are not projections, then $\mathrm{Im}(g) \subseteq B$ and f is B-insensitive, i.e., f is $\mathrm{Im}(g)$-insensitive. Therefore, $\forall \mathbf{a} \in A^{m+n-1} : (f * g)(\mathbf{a}) = f(b, a_{m+1}, \ldots, a_{m+n-1})$ for some fixed $b \in B$. Hence, none of the first m variables are essential, and out of the others at most k are essential, i.e., $f * g$ is essentially at most k-ary. □

3.4 Relational description of restricted essential arity

In this section we describe certain relations and their polymorphisms. We aim to relationally describe some clones with restricted essential arity. We are looking for a relation E_k which can "cut off" all the operations of essential arity k or higher.

3.4.1 Definition and basic properties of E_k

Given essentially k-ary f, we want to exploit the pairs of tuples $\mathbf{a}_i$, $\mathbf{a}'_i$ which differ only at the i-th coordinate and for which $f(\mathbf{a}_i) \neq f(\mathbf{a}'_i)$. The idea is to define the relation as tuples of equality and nonequality pairs. For example, the elements of E_3 will consist of 3 pairs, i.e., it will be 6-ary. In the compatibility scheme we would like to only allow tuples with at most one nonequality on the input side, and allow all tuples with at least one equality on the output side; we would then obtain following schemes:


E_3^{in} E_3^{in} E_3^{in} E_3^{out}
E_3^{in} E_3^{in} E_3^{out}
f

On the left we have essentially ternary operation, so we have a pair of tuples $\mathbf{a}_i$, $\mathbf{a}'_i$ (rows), differing only in the i-th variable, such that $f(\mathbf{a}_i) \neq f(\mathbf{a}'_i)$ for $i = 1, 2, 3$. On the right we have essentially binary operation, so we have such tuples only for two variables.

The problem is how to define the relation so that one part is only relevant on the input and the other only on the output side. Inspired by the clones of B-insensitive operations with image in B, we can have the "input tuples" consist of arbitrary elements, but the "output tuples" consist only of elements from B. If we take f B-insensitive with $\operatorname{Im}(f) \subseteq B$, it works as desired. The B-insensitivity allows us to mix "output tuples" into the input, since f can not distinguish between any of its elements, so it would be equivalent to inputing a constant tuple. The image in B allows us to restrict the tuples we allow on the output just to B.

We now define the relations E_k. For a set A, a *diagonal* on A is a binary relation given by

$$\Delta_A = \{(a, a) \mid a \in A\}.$$

We say that a pair (a, b) is *diagonal* if $a = b$. Otherwise we say it is *non-diagonal.*

Let $B \subsetneq A$ be nonempty finite sets, $|B| \geq 2$. For natural $k \geq 2$ we define $2k$-ary relations E_k as

$$\begin{aligned}
E_k^A &= \bigcup_{i=1}^{k} \underbrace{\Delta_A \times \cdots \times \Delta_A \times \overset{i}{A^2} \times \Delta_A \times \cdots \times \Delta_A}_{k},\\
E_k^B &= \bigcup_{i=1}^{k} \underbrace{B^2 \times \cdots \times B^2 \times \overset{i}{\Delta_B} \times B^2 \times \cdots \times B^2}_{k},\\
E_k &= E_k^A \cup E_k^B.
\end{aligned}$$

The "A-part", E_k^A, consists of all tuples of k pairs such that at most one pair is non-diagonal. It is stricter in the number of equalities, but more liberal in the elements themselves. The "B-part", E_k^B, consists of all tuples of k pairs in B such that at least one pair is diagonal. It is more liberal in the number of equalities, but restricts the elements we can use to B.

The two parts are not disjoint. In the intersection, we have all the tuples of k pairs in B with at most one non-diagonal pair. Note that for $k = 2$ we have $E_2 = E_2^A$.

To clarify the relations in yet another way, let us observe that $\mathbf{a} \notin E_k$ if and only if either none of the pairs in $\mathbf{a}$ is diagonal, or there are (at least) two non-diagonal pairs in $\mathbf{a}$ and $a_i \notin B$ for some i.

We define clones

$$\mathcal{E}_k = \mathrm{Pol}(E_{k+1}, E_{k+2}, \dots).$$

Defined this way, it is clear that $\mathcal{E}_k \subseteq \mathcal{E}_{k+1}$ for all k. It is also clear that $\mathcal{E}_k \subseteq \mathrm{Pol}(E_{k+1})$. We show later that the other inclusion holds as well for $k \geq 2$, but it is not immediately obvious.

Let us first check that E_k "cuts off" operations of essential arity k.

Lemma 13. *Let $k \geq 2$. Operations of essential arity k and higher are not compatible with E_k.*

Proof. Assume $n \geq k \geq 2$, f is an n-ary operation on A, and its first k variables are essential. We show that $f \notin \mathrm{Pol}(E_k)$. Since the first k variables of f are essential, there exist tuples $\mathbf{r}_1, \mathbf{r}'_1, \mathbf{r}_2, \mathbf{r}'_2, \dots, \mathbf{r}_k, \mathbf{r}'_k \in A^n$ such that $\mathbf{r}_i$ differs from $\mathbf{r}'_i$ only at the i-th coordinate, and $f(\mathbf{r}_i) \neq f(\mathbf{r}'_i)$ for all $i = 1, \dots, k$. Using these $2k$ tuples as rows, the following scheme is obtained:

$$\begin{array}{cccccccccccc}
\mathbf{r}_1 & = & \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & \bullet \\
 & & \nparallel & \| & & \| & \| & & \| & & \nparallel \\
\mathbf{r}'_1 & = & \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & \bullet \\
\\
\mathbf{r}_2 & = & \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & \bullet \\
 & & \| & \nparallel & & \| & \| & & \| & & \nparallel \\
\mathbf{r}'_2 & = & \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & \bullet \\
 & & \vdots & \vdots & \ddots & \vdots & \vdots & & \vdots & & \vdots \\
\mathbf{r}_k & = & \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & \bullet \\
 & & \| & \| & & \nparallel & \| & & \| & & \nparallel \\
\mathbf{r}'_k & = & \bullet & \bullet & \cdots & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & \bullet \\
 & & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\notin$} \\
 & & E_k & E_k & & E_k & E_k & & E_k & & E_k
\end{array}$$

Hence $f \notin \mathrm{Pol}(E_k)$. $\square$

Next, we show the following.

Lemma 14. *Let $k \in \mathbb{N}$ and f be a B-insensitive operation on A with $\mathrm{Im}(f) \subseteq B$. Then*

$$f \in \mathcal{E}_k \iff f \text{ is essentially at most } k\text{-ary.}$$

Proof. Let $\tilde{\mathcal{C}}$ be a clone and

$$\mathcal{C} = \Big\{ f \in \tilde{\mathcal{C}} \,|\, f \text{ is } B\text{-insensitive and } \mathrm{Im}(f) \subseteq B \Big\} \cup \mathcal{P}_A.$$

We show that $\mathcal{E}_k \cap \mathcal{C}$ consists exactly of all at most k-essential operations in $\mathcal{C}$. The inclusion "$\subseteq$" is a corollary of Lemma 13. We need to show the converse – that every essentially at most k-ary operation in $\mathcal{C}$ is compatible with E_l for every $l \geq k+1$.

When studying compatibility of E_k and operations in $\mathcal{C}$, we can restrict ourselves to only inputing tuples from E_k^A and only asking if the output is in E_k^B, that is,

$$f \in \operatorname{Pol}(E_k) \iff \forall \mathbf{a}_1, \dots, \mathbf{a}_{\operatorname{ar}(f)} \in E_k^A : f(\mathbf{a}_1, \dots, \mathbf{a}_{\operatorname{ar}(f)}) \in E_k^B. \tag{3.1}$$

The projections are trivially compatible with every relation and for non-trivial operations we can do the following. If we input $\mathbf{a}_i \in E_k^B$, we can equivalently take a constant tuple $(b, \dots, b) \in E_k^A$, for any $b \in B$, and the result will be the same, because f is B-insensitive. Since $\operatorname{Im}(f) \subseteq B$, the resulting tuple will always be in B^{2k}, and in that case it is in E_k if and only if it is in E_k^B.

Let $k \geq 2$, $f \in \mathcal{C}$ be n-ary with essential arity at most k, and $l \geq k+1$. We will assume all essential variables of f are among the first k. Let $\mathbf{a}_1, \dots, \mathbf{a}_n \in E_l^A$. We know that there are $l > k$ pairs in each $\mathbf{a}_i$ and only one can be non-diagonal in each $\mathbf{a}_i$. Thus, there exists $i \in \{1, \dots, l\}$ such that the i-th pair is diagonal in all $\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_k$. Since all essential variables of f are among the first k, also the i-th pair of $f(\mathbf{a}_1, \dots, \mathbf{a}_n)$ is diagonal, hence the tuple is in E_l^B. Using (3.1) we obtain $f \in \operatorname{Pol}(E_l)$.

We have proved that

$$\mathcal{E}_k \cap \mathcal{C} = \{ f \in \mathcal{C} \mid f \text{ is at essentialy most } k\text{-ary} \} .$$

□

This offers an alternative prove of Proposition 12, since $\mathcal{E}_k$, as a set of polymorphisms, is a clone, and an intersection of clones is also a clone. This yields a chain of clones

$$\mathcal{E}_1 \cap \mathcal{C} \subseteq \mathcal{E}_2 \cap \mathcal{C} \subseteq \cdots \subseteq \mathcal{E}_k \cap \mathcal{C} \subseteq \cdots \subseteq \mathcal{C},$$

which, given $\mathcal{C}$ has operations of arbitrarily high essential arity, has infinitely many strict inclusions. Furthermore, $\bigcup_{k \in \mathbb{N}} \mathcal{E}_k \cap \mathcal{C} = \mathcal{C}$, hence this yields that $\mathcal{C}$ is non-finitely generated.

3.4.2 Polymorphisms of E_k

We know that B-insensitivity and having the image in B are sufficient for an at most k-essential operation to be in $\mathcal{E}_k$. It is, however, not necessary for the operation to be B-insensitive in all variables. In this section we show that an operation can satisfy a different condition in one of the variables. The aim is now to describe the clone

$$\mathcal{E} = \bigcup_{k \in \mathbb{N}} \mathcal{E}_k = \bigcup_{k \in \mathbb{N}} \operatorname{Pol}(E_{k+1}, E_{k+2}, \dots).$$

We will investigate $\operatorname{Pol}(E_k)$. For the whole section, we fix nonempty finite sets $B \subsetneq A$, $|B| \geq 2$.

First, we examine essentially at most unary operations. In the following, we treat essentially at most unary operations as unary; we ignore the fictitious variables. There is no loss in generality, since non-essential variables do not influence compatibility in any way.

Lemma 15. *For $k \geq 3$ and an essentially at most unary operation u, we have*

$$u \in \mathrm{Pol}(E_k) \iff \begin{array}{l} \textit{either } u \textit{ is } B\textit{-insensitive (i.e., constant on } B\textit{),} \\ \textit{or } u(B) \subseteq B. \end{array}$$

Proof. Let $u(x)$ be essentially at most unary. If u is neither B-insensitive nor does it satisfy $u(B) \subseteq B$, then we have $b_1, b_2, b \in B$ such that $u(b_1) \neq u(b_2)$ and $u(b) \notin B$. Let $\mathbf{b} = (b_1, b_2, b_1, b_2, b, b, \dots, b, b)$ be $2k$-ary. Since the third pair is diagonal and all elements are in B, we have $\mathbf{b} \in E_k$. However, in the tuple

$$u(\mathbf{b}) = (u(b_1), u(b_2), u(b_1), u(b_2), u(b), u(b), \dots)$$

we have the first two pairs non-diagonal, therefore $u(\mathbf{b}) \notin E_k^A$, and there is also $u(b) \notin B$, so $u(\mathbf{b}) \notin E_k^B$. We have found $\mathbf{b} \in E_k$ such that $u(\mathbf{b}) \notin E_k$.

On the other hand, a unary operation maps diagonal pairs to diagonal pairs, so trivially $u(E_k^A) \subseteq E_k$, and we only need to check that $u(E_k^B) \subseteq E_k$. In the case $u(B) \subseteq B$, we get the inclusion using the same argument. If u is B-insensitive, we have $u(E_k^B) = \{(a, a, \dots, a)\} \subseteq E_k$ for some $a \in A$. □

Note that this description includes projections, which we had to add separately before. Also, all (essentially) unary operations are compatible with E_2 since $E_2 = E_2^A$ and unary mappings preserve diagonals. We will not consider this degenerate case from now on.

Next, let us look at essentially at least binary operations. We show that the image in B is a necessary condition for compatibility with E_k, $k \geq 3$. Let f be n-ary with the first two variables essential and $\mathbf{a} \in A^n$ be such that $f(\mathbf{a}) \neq B$. For $i = 1, 2$ there exist $\mathbf{r}_i, \mathbf{r}'_i \in A^n$ differing only at the i-th coordinate such that $f(\mathbf{r}_i) \neq f(\mathbf{r}'_i)$. The following scheme shows that f is not compatible with E_k for any $k \geq 3$.

$$\begin{array}{ccccccccl}
 & & & & & & & & \mathbf{u} \\
 & & & & & & & & \shortparallel \\
\mathbf{r}_1 & = & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & & f(\mathbf{r}_1) \\
 & & \nparallel & \shortparallel & & \shortparallel & & & \nparallel \\
\mathbf{r}'_1 & = & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & & f(\mathbf{r}'_1) \\
 & & & & & & & & \\
\mathbf{r}_2 & = & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & & f(\mathbf{r}_2) \\
 & & \shortparallel & \nparallel & & \shortparallel & & & \nparallel \\
\mathbf{r}'_2 & = & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & & f(\mathbf{r}'_2) \\
 & & & & & & & & \\
\mathbf{a} & = & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & & f(\mathbf{a}) \notin B \\
 & & \shortparallel & \shortparallel & & \shortparallel & & & \shortparallel \\
\mathbf{a} & = & \bullet & \bullet & \cdots & \bullet & \xrightarrow{f} & & f(\mathbf{a}) \notin B \\
 & & \vdots & \vdots & & \vdots & & & \vdots \\
 & & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & & \rotatebox{90}{$\notin$} \\
 & & E_k & E_k & & E_k & & & E_k
\end{array} \tag{3.2}$$

The resulting tuple, say $\mathbf{u}$, is not in E_k, because the first two pairs are non-diagonal, and hence $\mathbf{u} \notin E_k^A$, and it contains $f(\mathbf{a}) \notin B$, therefore $\mathbf{u} \notin E_k^B$.

The B-insensitivity can be weakened. We will show that in one of the variables the operation can be B-unary instead.

Definition 7. *We say an n-ary operation f is* B-unary *in the i-th variable if*

$$\forall \mathbf{a}, \mathbf{a}' \ \forall b \in B : f(a_1 \dots, a_{i-1}, b, a_{i+1}, \dots, a_n) = f(a'_1 \dots, a'_{i-1}, b, a'_{i+1}, \dots, a'_n).$$

The definition says that if we plug in an element from B to the i-th variable, the operation only depends on that variable.

We show that if $f \in \mathrm{Pol}(E_k)$, then it is either B-unary or B-insensitive in every variable. Assume that f is an n-ary operation, which is neither B-insensitive nor B-unary in the first variable. Then there exist tuples $\mathbf{b}, \mathbf{b}' \in A^n$ which only differ at the first coordinate, and $b_1, b'_1 \in B$, such that $f(\mathbf{b}) \neq f(\mathbf{b}')$. There also exist tuples $\mathbf{c}, \mathbf{c}' \in A^n$ such that $c := c_1 = c'_1 \in B$ and $f(\mathbf{c}) \neq f(\mathbf{c}')$. The following scheme shows that f is not compatible with E_k for any $k \geq 3$.

$$\begin{array}{cccccccc}
\mathbf{c} & = & \mathrm{c} & \bullet & \cdots & \bullet & \xrightarrow{f} & f(\mathbf{c}) \\
 & & \| & \nparallel & & \nparallel & & \nparallel \\
\mathbf{c}' & = & \mathrm{c} & \bullet & \cdots & \bullet & \xrightarrow{f} & f(\mathbf{c}') \\
 & & & & & & & \\
\mathbf{b} & = & b_1 & \bullet & \cdots & \bullet & \xrightarrow{f} & f(\mathbf{b}) \\
 & & \nparallel & \| & & \| & & \nparallel \\
\mathbf{b}' & = & b'_1 & \bullet & \cdots & \bullet & \xrightarrow{f} & f(\mathbf{b}') \\
 & & \vdots & \vdots & & \vdots & & \vdots \\
\mathbf{b} & = & b_1 & \bullet & \cdots & \bullet & \xrightarrow{f} & f(\mathbf{b}) \\
 & & \nparallel & \| & & \| & & \nparallel \\
\mathbf{b}' & = & b'_1 & \bullet & \cdots & \bullet & \xrightarrow{f} & f(\mathbf{b}') \\
 & & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\notin$} \\
 & & E_k^B & E_k^A & & E_k^A & & E_k
\end{array} \tag{3.3}$$

Note that there can be less inequalities among the pairs c_i,c'_i; we can even make sure there is only one, by changing the elements in each coordinate one by one, until the inequality for the results occurs. We observed that $f \in \mathrm{Pol}(E_k)$, $k \geq 3$, implies that f is in each variable either B-insensitive or B-unary.

Moreover, if f satisfies this property, then it can have at most one variable, which is not B-insensitive. If two variables were not B-insensitive, let us say the first two, then they would both have to be B-unary. For every $b, b' \in B$, $\mathbf{a} \in A^n$ we would get

$$f(b, a_2, \dots, a_n) \overset{\text{1st}}{=} f(b, b, a_3, \dots, a_n) \overset{\text{2nd}}{=} f(b', b, a_3, \dots, a_n) \overset{\text{1st}}{=} f(b', a_2, \dots, a_n),$$

where we used that f is B-unary in the first or second variable as indicated above the equal signs. But that proves the first variable is B-insensitive, which contradicts our assumption.

Finally, we show that the derived properties are sufficient for membership in $\mathrm{Pol}(E_k)$. Let f be an n-ary operation with essential arity $l \geq 2$ such that $\mathrm{Im}(f) \subseteq B$ and f is B-insensitive in all variables with possible exception of one variable, which is then B-unary. Let $k > l$.

If f is B-insensitive in all variables, then $f \in \operatorname{Pol}(E_k)$ by Lemma 14. If not, then one variable is B-unary instead. Let us assume it is the first one. In the proof of Lemma 14 we used insensitivity for the fact that we only needed to check the compatibility on the tuples from E_k^A. We can still use the proof to get $f(E_k^A, E_k, \ldots, E_k) \subseteq E_k$. It only remains to check that we can not spoil the compatibility when having a tuple from E_k^B at the first coordinate.

Let $\mathbf{b} \in E_k^B$ and $\mathbf{a}_2, \ldots, \mathbf{a}_n \in E_k$. There exists j such that the j-th pair of $\mathbf{b}$ is diagonal. Since f is B-unary in the first variable, necessarily also the j-th pair of the resulting tuple $\mathbf{u} = f(\mathbf{b}, \mathbf{a}_2, \ldots, \mathbf{a}_n)$ is diagonal. The elements of the tuple $\mathbf{u}$ are also all in B, hence $\mathbf{u} \in E_k^B \subseteq E_k$. Therefore, $f \in \operatorname{Pol}(E_k)$.

All together we have proved the following.

Proposition 16. *Let $k \geq 2$. Then $\operatorname{Pol}(E_{k+1})$ consists exactly of*

- *u essentially at most unary such that u is either B-insensitive or $u(B) \subseteq B$,*
- *f at most k-essential such that $\operatorname{Im}(f) \subseteq B$ and f is B-insensitive in all variables with possible exception of one variable, which is then B-unary.*

This yields several corollaries. First, for every $k \geq 3$, we have the inclusion $\operatorname{Pol}(E_k) \subseteq \operatorname{Pol}(E_{k+1})$, and hence $\mathcal{E}_{k-1} = \operatorname{Pol}(E_k)$.

Secondly, $\mathcal{E}$ consists exactly of the operations described in Proposition 16 without the restriction of essential arity.

Lastly, $\mathcal{E}_k$'s are the k-essential parts of $\mathcal{E}$, and hence $\mathcal{E}$ is non-finitely generated. Note that $E_{k+1} \notin \operatorname{Inv}(\mathcal{E})$ and $E_{k+1} \in \operatorname{Inv}(\mathcal{E}^{(k)})$, since if E_{k+1} is compatible with essentially at most k-ary operations in $\mathcal{E}$, then it is in particular compatible with at most k-ary operations in $\mathcal{E}$. Therefore, the relations E_{k+1} are as in Lemma 9 for the clone $\mathcal{E}$.

Moreover, we show in Chapter 4 that the clones $\mathcal{E}_k$ are maximal clones of essentially at most k-ary operations, in the sense that for any $f \notin \mathcal{E}_k$, the clone $\operatorname{Clo}(\mathcal{E}_k \cup \{f\})$ contains an operation of essential arity at least $k + 1$.

Remark. The polymorphisms of "A-part" or "B-part" of E_k alone are not too interesting. For every $k \geq 3$, $\operatorname{Pol}(E_k^A)$ is the clone of all essentially unary operations on A, and $\operatorname{Pol}(E_k^B)$ is the clone of all operations f such that $f(B, \ldots, B) \subseteq B$ and its restriction to B is an essentially unary operation on B.

3.4.3 Modifications and possible generalizations

In this section we will look at several ways how the relations E_k could be modified.

Replacing Δ_A with Δ_B

Let us replace Δ_B in the "B-part" with Δ_A, but not only at the one coordinate where we explicitly write "Δ_B" in the definition, but at all places. That is, let

$\tilde{E}_k$ be a $2k$-ary relation given by

$$\begin{aligned}
\tilde{E}_k^A &= \bigcup_{i=1}^{k} \Delta_A \times \cdots \times \Delta_A \times \overset{i}{A^2} \times \Delta_A \times \cdots \times \Delta_A, \\
\tilde{E}_k^B &= \bigcup_{i=1}^{k} (B^2 \cup \Delta_A) \times \cdots \times (B^2 \cup \Delta_A) \times \overset{i}{\Delta_A} \times (B^2 \cup \Delta_A) \times \cdots \times (B^2 \cup \Delta_A), \\
\tilde{E}_k &= \tilde{E}_k^A \cup \tilde{E}_k^B.
\end{aligned}$$

We have added new tuples in the relations. For the compatibility that means we have more liberty in the result, but we can also use more tuples as the input. The tuples $\mathbf{a} \notin \tilde{E}_k$ are exactly such that either all pairs are non-diagonal or there are (at least) two pairs non-diagonal and $a_i \notin B$ for some a_i in a non-diagonal pair. The requirement of $a_i \notin B$ being in one of the non-diagonal pairs is the difference compared to E_k.

What changes for the compatible operations is that we no longer allow the one variable which might not be B-insensitive.

Proposition 17. *For $k \geq 3$, $\mathrm{Pol}(\tilde{E}_{k+1})$ consists exactly of*

- *u essentially at most unary such that u is either B-insensitive or $u(B) \subseteq B$,*
- *f at most k-essential such that $\mathrm{Im}(f) \subseteq B$ and f is B-insensitive.*

Proof. The proof will be similar to the case of E_k. We will point out the differences. Let $k \geq 3$ be fixed.

Lemma 13 works the same for $\tilde{E}_k$. For the unary operations, the proof of Lemma 15 works if we realize that given $b_1, b_2, b \in B$ such that $u(b_1) \neq u(b_2)$ and $u(b) \neq B$, we can actually assume that $b \in \{b_1, b_2\}$, since necessarily either $u(b_1) \neq u(b)$ or $u(b_2) \neq u(b)$.

Now for the essentially at least binary operations. Let f be n-ary for $n \geq 2$ and for convenience let all the variables of f be essential. Again, the non-essential variables do not influence compatibility in any way.

We show that $\mathrm{Im}(f) \subseteq B$ is a necessary condition for $f \in \mathrm{Pol}(\tilde{E}_k)$. Let $\mathbf{a} \in A^n$ be such that $f(\mathbf{a}) \neq B$. In the previous, we have constructed scheme (3.2) in which the resulting tuple had the first two pairs non-diagonal and the elements of the third pair were not in B. Now we need one of the elements in the first two pairs not to be in B. However, for some $i \in \{1, \ldots, n\}$ there must be $\mathbf{r}_i, \mathbf{r}'_i \in A^n$ differing only at the i-th coordinate such that $f(\mathbf{r}_i) \neq f(\mathbf{r}'_i)$, and moreover, $f(\mathbf{r}_i) \notin B$. To find it, let $\mathbf{c} \in A^n$ be such that $f(\mathbf{a}) \neq f(\mathbf{c})$. Then there must be an inequality in the following chain

$$f(a_1, \ldots, a_n) \overset{?}{=} f(c_1, a_2, \ldots, a_n) \overset{?}{=} f(c_1, c_2, a_3, \ldots, a_n) \overset{?}{=} \ldots \overset{?}{=} f(c_1, \ldots, c_n).$$

Since $f(\mathbf{a}) \notin B$, the first inequality that occurs yields the desired tuples. Then we can use scheme (3.2).

The necessity of f being B-insensitive can be seen easily. We use scheme (3.3), where $\mathbf{b}, \mathbf{b}' \in A^n$ are, as before, such that they only differ at the first coordinate, $b_1, b'_1 \in B$ and $f(\mathbf{b}) \neq f(\mathbf{b}')$, but in the place of $\mathbf{c}, \mathbf{c}'$ we can now take any two tuples witnessing that the second variable is essential. We can do this, because now we can have $c_1 \in A$.

On the other hand, given f at most k-essential, B-insensitive, with $\mathrm{Im}(f) \subseteq B$, we have equivalence analogous to (3.1), since B-insensitivity tells us we only need to check tuples from $\tilde{E}^A_{k+1}$ as the input, and the image in B tells us we only need to check if there is at least one diagonal pair in the result. Therefore we show $f \in \mathrm{Pol}(\tilde{E}_{k+1})$ exactly as in the proof of Lemma 14 for E_{k+1}. $\square$

Replacing inequalities with a different relation

A natural modification of the relations is to require that the non-diagonal pairs are in some other relation; for example, we might require that the first element in non-diagonal pairs is greater then the second, with respect to some partial order.

We show that for certain relations R, the polymorphisms of modified relations $\tilde{E}_k$, that we get this way, are the operations described in Proposition 16 which are, moreover, also compatible with R. As R we use the edges of a weakly connected graph on A with loops in all vertices, such that, furthermore, B is a weakly connected component.

We say that a binary relation R on A is *reflexive*, if $(a, a) \in R$ for every $a \in A$. Two elements $a, a' \in A$ are said to be *weakly connected in* R via $c_0, \dots, c_n \in A$, if $c_0 = a$, $c_n = a'$ and either $(c_i, c_{i+1}) \in R$ or $(c_{i+1}, c_i) \in R$ for all $i = 0, 1, \dots, n-1$. A set $C \subseteq A$ is *weakly connected in* R, if all $c, c' \in C$ are weakly connected in R via elements in C. We say that R is *weakly connected*, if all elements in A are weakly connected in R, i.e., if A is weakly connected in R.

For $k \geq 3$ and a reflexive binary relation R, we can define the modified relations as

$$\begin{aligned}
\tilde{E}^A_k &= \bigcup_{i=1}^{k} \Delta_A \times \dots \times \Delta_A \times \overset{i}{R} \times \Delta_A \times \dots \times \Delta_A, \\
\tilde{E}^B_k &= \bigcup_{i=1}^{k} (R \cap B^2) \times \dots \times (R \cap B^2) \times \overset{i}{\Delta_B} \times (R \cap B^2) \times \dots \times (R \cap B^2), \\
\tilde{E}_k &= \tilde{E}^A_k \cup \tilde{E}^B_k.
\end{aligned}$$

Since R is reflexive, we actually obtain $\tilde{E}_k = E_k \cap R^k$.

In general, for any relations K, L of the same arity, the inclusion

$$\mathrm{Pol}(K) \cap \mathrm{Pol}(L) \subseteq \mathrm{Pol}(K \cap L)$$

holds. This is easily seen, as if we take f compatible with both K and L, and tuples $\mathbf{t}_1, \dots, \mathbf{t}_n \in K \cap L$, then $f(\mathbf{t}_1, \dots, \mathbf{t}_n)$ is in both K and L. If $K \cap L = \varnothing$, we have $\mathcal{O}_A$ on the right hand side. It is also trivial to check that $\mathrm{Pol}(K) = \mathrm{Pol}(K^m)$ for all $m \in \mathbb{N}$. Hence

$$\mathrm{Pol}(E_k) \cap \mathrm{Pol}(R) \subseteq \mathrm{Pol}(E_k \cap R^k).$$

We will now look more closely at the other inclusion.

Proposition 18. *Let $R \subseteq A^2$ be reflexive, weakly connected relation on A such that B is also weakly connected in R. Then*

$$\mathrm{Pol}(E_k) \cap \mathrm{Pol}(R) = \mathrm{Pol}(E_k \cap R^k)$$

for all $k \geq 3$.

Proof. We already have the inclusion "$\subseteq$" and will show the other.

First, let us check that $f \in \mathrm{Pol}(E_k \cap R^k)$ implies that $f \in \mathrm{Pol}(R)$. This is easy to see. We have $R = \mathrm{Proj}_{1,2}(E_k \cap R^k)$, since for $(r, r') \in R$ we have $(r, r', a, a, \dots, a)$ in $E_k \cap R^k$ for any $a \in A$. An operation compatible with a relation is also compatible with its projections.

To prove that $f \in \mathrm{Pol}(E_k \cap R^k)$ implies $f \in \mathrm{Pol}(E_k)$, we need to show that all the conditions for operations described in Proposition 16 are necessary for f to be compatible with $E_k \cap R^k$. It suffices to make sure we can use all the schemes we have used to prove this implication in Proposition 16.

We always assumed to have some pair of tuples $\mathbf{a}, \mathbf{a}'$ for which $f(\mathbf{a}) \neq f(\mathbf{a}')$. Moreover we either assumed or could adjust the tuples to only differ at one coordinate. It was also important that some of the coordinates were in B – if that was the case, then it held for both tuples for the given coordinate.

It is, therefore, enough to show that for any operation $f \in \mathrm{Pol}(E_k \cap R)$ the following holds. Let $\mathbf{a}, \mathbf{a}' \in A^n$ be such tuples that they only differ at the i-th coordinate, and $f(\mathbf{a}) \neq f(\mathbf{a}')$. Then there exist $(r, r') \in R$ such that for $\mathbf{r} = (a_1, \dots, a_{i-1}, r, a_{i+1}, \dots, a_n)$ and $\mathbf{r}' = (a_1, \dots, r', \dots, a_n)$ we still have $f(\mathbf{r}) \neq f(\mathbf{r}')$, and, moreover, if $a_i, a_i' \in B$ then we can find such r, r' also in B.

This is a straightforward consequence of the weak connectivity of R. We know that a_i, a_i' are weakly connected in R via some elements $c_0, c_1, \dots, c_l$. We have $c_0 = a_i, c_l = a_i'$ and either $(c_j, c_{j+1}) \in R$ or $(c_{j+1}, c_j) \in R$ for all $i = 0, \dots, l$. Because B is weakly connected in R, if $a_i, a_i' \in B$, then also $c_0, \dots, c_l \in B$. Let $\mathbf{c}_i = (a_1, \dots, a_{i-1}, c_i, a_{i+1}, \dots, a_n)$. Since $f(\mathbf{c}_0) = f(\mathbf{a}) \neq f(\mathbf{a}') = f(\mathbf{c}_l)$, there has to exist j such that $f(c_j) \neq f(c_{j+1})$. But now either $(c_j, c_{j+1}) \in R$, and we put $r := c_j, r' := c_{j+1}$, or $(c_{j+1}, c_j) \in R$, and then we put $r := c_{j+1}, r' := c_j$. □

Taking triplets or m-tuples instead of pairs

Another natural modification, especially after seeing the construction with replacing the inequalities with a different relation, is to take triplets, or in general m-tuples, instead of pairs. We show this does not change the polymorphisms.

For a set C and $m \geq 3$ we define an m-ary diagonal as

$$\Delta_C^m = \{\underbrace{(c, \dots, c)}_{m} \mid c \in C\},$$

and for $k \geq 3$ we define $(m \cdot k)$-ary relations given by

$$\begin{aligned}
\tilde{E}_k^A &= \bigcup_{i=1}^{k} \underbrace{\Delta_A^m \times \cdots \times \Delta_A^m \times \overset{i}{A^m} \times \Delta_A^m \times \cdots \times \Delta_A^m}_{k}, \\
\tilde{E}_k^B &= \bigcup_{i=1}^{k} \underbrace{B^m \times \cdots \times B^m \times \overset{i}{\Delta_B^m} \times B^m \times \cdots \times B^m}_{k}, \\
\tilde{E}_k &= \tilde{E}_k^A \cup \tilde{E}_k^B.
\end{aligned}$$

For $k \geq 3$ and $m \geq 3$ we claim that

$$\mathrm{Pol}(\tilde{E}_k) = \mathrm{Pol}(E_k). \tag{3.4}$$

For simplicity, we consider only the case $m = 3$. The analogy for $m > 3$ will be clear. We prove the claim by providing pp-definitions of $\tilde{E}_k$ using E_k, and vice versa.

Let $k \geq 3$. First, we pp-define $\tilde{E}_k$ using E_k. We will now use notation

$$\mathbf{x} = (x_1^1, x_2^1, x_3^1, x_1^2, x_2^2, x_3^2, \ldots, x_1^k, x_2^k, x_3^k).$$

Let

$$\tilde{P}_k(\mathbf{x}) :\equiv \bigwedge_{\substack{1 \leq i_l < j_l \leq 3 \\ l=1,\ldots,k}} E_k(x_{i_1}^1, x_{j_1}^1, x_{i_2}^2, x_{j_2}^2, \ldots, x_{i_k}^k, x_{j_k}^k),$$

i.e., for $\mathbf{x}$ to be in $\tilde{P}_k$, we choose two elements from each triplet independently and we want every such tuple to be in E_k.

Claim. $\tilde{P}_k = \tilde{E}_k$

Proof. Clearly $(x, y, z) \in \Delta_C^3 \iff (x, y), (x, z), (y, z) \in \Delta_C^2$. Given $\mathbf{x} \in \tilde{E}_k$, we easily see that if we arbitrary choose pairs out of the triplets, it always yields a tuple in E_k; hence $\mathbf{x} \in \tilde{P}_k$. On the other hand, if $\mathbf{x} \notin \tilde{E}_k$, there are two possibilities:

(i) none of the triplets is diagonal,

(ii) $\mathbf{x}$ contains an element in $A \setminus B$ and there are at least two non-diagonal triplets.

In (i) we choose two unequal elements from each triplet and get a tuple of k non-diagonal pairs which is not in E_k, i.e., $\mathbf{x} \notin \tilde{P}_k$. In (ii) we chose the non-diagonal pairs from the non-diagonal triplets and also chose the element not in B – again, $\mathbf{x} \notin \tilde{P}_k$. $\square$

To pp-define E_k using $\tilde{E}_k$ we can take the projection $\mathrm{Proj}_I(\tilde{E}_k)$, where I are the indices of the first two coordinates of each triplet, i.e.,

$$\mathrm{Proj}_I(\tilde{E}_k)(x_1^1, x_2^1, x_1^2, x_2^2, \ldots, x_1^k, x_2^k) \equiv \exists x_3^1, x_3^2, \ldots, x_3^k \; \tilde{E}_k(\mathbf{x}).$$

Claim. $\mathrm{Proj}_I(\tilde{E}_k) = E_k$

Proof. Indeed, if $(x_1^1, x_2^1, x_1^2, x_2^2, \ldots, x_1^k, x_2^k) \in E_k$, we set $x_3^j := x_1^j$ for $j = 1, \ldots, k$ and obtain $\mathbf{x} \in \tilde{E}_k$, hence $(x_1^1, x_2^1, x_1^2, x_2^2, \ldots, x_1^k, x_2^k) \in \mathrm{Proj}_I(\tilde{E}_k)$.

On the other hand, if $(x_1^1, x_2^1, x_1^2, x_2^2, \ldots, x_1^k, x_2^k) \in \mathrm{Proj}_I(\tilde{E}_k)$, then there exist x_3^j's such that $\mathbf{x} \in \tilde{E}_k$. From the previous we have $\tilde{E}_k = \tilde{P}_k$, and hence, choosing $i_1 = i_2 = \cdots = i_k = 1$ and $j_1 = j_2 = \cdots = j_k = 2$ in the definition of $\tilde{P}_k$, we obtain $(x_1^1, x_2^1, x_1^2, x_2^2, \ldots, x_1^k, x_2^k) \in E_k$. $\square$

We have pp-defined E_k and $\tilde{E}_k$ from each other, so (3.4) follows.

We could now do a construction similar to what we did before – restricting the non-diagonals to some relation. Let R be reflexive m-ary relation and $\tilde{E}_k$ be the m-tuple version of E_k for $k \geq 3$. Then we obtain $\mathrm{Pol}(\tilde{E}_k) \cap \mathrm{Pol}(R) \subseteq \mathrm{Pol}(\tilde{E}_k \cap R^k)$ and $\mathrm{Pol}(\tilde{E}_k \cap R^k) \subseteq \mathrm{Pol}(R)$ as before. The only question is what should R satisfy so that $\mathrm{Pol}(\tilde{E}_k \cap R^k) \subseteq \mathrm{Pol}(\tilde{E}_k)$ holds.

For operations not satisfying the conditions from Proposition 16, we have shown they are not compatible with E_k by exploiting tuples $\mathbf{a}, \mathbf{a}'$ differing only at the i-th coordinate and such that $f(\mathbf{a}) \neq f(\mathbf{a}')$. To carry out the prove the same way in this case, we would always need to find the tuples such that a_i, a_i' are both contained in a single tuple from R.

We can always find such tuples if R satisfies the following. For every $a, a' \in A$ there exist $c_0, \ldots, c_n$ such that $c_0 = a$ and $c_n = a'$, and for every $j = 0, \ldots, n-1$ there exists a tuple $\mathbf{d}_j \in R$ such that both c_j and c_{j+1} are in $\mathbf{d}_j$.

This way, given $\mathbf{a}, \mathbf{a}'$ as before, we can define $\mathbf{c}_j = (a_1, \ldots, a_{i-1}, c_j, a_{i+1}, \ldots, a_n)$ and for some j we obtain $f(\mathbf{c}_j) \neq f(\mathbf{c}_{j+1})$. We can then construct the needed scheme using the tuples $(a_1, \ldots, a_{i-1}, d_l, a_{i+1}, \ldots, a_n)$ as rows for $l = 1, \ldots, k$.

3.5 Unary version of the relations

In Section 3.4.1 we used the binary relations A^2, B^2, Δ_A and Δ_B to define E_k. They satisfy the following inclusions:

$$\begin{tikzcd}
& A^2 & \\
B^2 \arrow[ur, hook, "\neq"'] & \not\subseteq & \Delta_A \arrow[ul, hook', "\neq"] \\
& \Delta_B \arrow[ul, hook', "\neq"] \arrow[ur, hook, "\neq"'] &
\end{tikzcd}$$

In this section we define similar relations, but we replace the binary relations by sets. Using these relations, we then construct a non-finitely generated clone.

Let A, B, C, D be nonempty finite sets satisfying the inclusions

$$\begin{tikzcd}
& A & \\
B \arrow[ur, hook, "\neq"'] & \not\subseteq & C \arrow[ul, hook', "\neq"] \\
& D \arrow[ul, hook', "\neq"] \arrow[ur, hook, "\neq"'] &
\end{tikzcd}.$$

For $k \geq 3$ we define k-ary relations

$$F_k = \bigcup_{i=1}^{k} \underbrace{C \times \cdots \times C \times \overset{i}{A} \times C \times \cdots \times C}_{k} \cup \bigcup_{i=1}^{k} \underbrace{B \times \cdots \times B \times \overset{i}{D} \times B \times \cdots \times B}_{k},$$

i.e., a tuple is in F_k if either at most one element is not in C, or all elements are in B, and at least one is in D. For $k \geq 2$ we define clones

$$\mathcal{F}_k = \mathrm{Pol}(F_{k+1}, F_{k+2}, \ldots).$$

We have a chain of clones $\mathcal{F}_2 \subseteq \cdots \subseteq \mathcal{F}_k \subseteq \mathcal{F}_{k+1} \subseteq \cdots$.

Proposition 19. *The clone $\mathcal{F} = \bigcup_{k \geq 3} \mathcal{F}_k$ is non-finitely generated.*

Proof. We fix elements $a \in A \setminus C$, $b \in B \setminus C$, $c \in C \setminus D$ and $d \in D$, and for $k \geq 3$ define k-ary operations

$$f_k(\mathbf{x}) = \begin{cases} b & \text{if } \mathbf{x} = (c, \ldots, c, \overset{i}{a}, c, \ldots, c) \text{ for } i \in \{1, \ldots, k\}, \\ d & \text{otherwise.} \end{cases}$$

We show that $f_k \in \mathrm{Pol}(F_l)$ for every $l \geq k+1$, so $f_k \in \mathcal{F}_k$, but $f_k \notin \mathrm{Pol}(F_k)$, and therefore $f_k \notin \mathcal{F}_{k-1}$.

Let $k \geq 2$. The following scheme shows that $f_k \notin \mathrm{Pol}(F_k)$:

$$\begin{array}{cccccccc}
a & c & c & \dots & c & \overset{f_k}{\to} & b \\
c & a & c & \dots & c & \overset{f_k}{\to} & b \\
c & c & a & \dots & c & \overset{f_k}{\to} & b \\
\vdots & \vdots & \vdots & \ddots & \vdots & & \vdots \\
c & c & c & \dots & a & \overset{f_k}{\to} & b \\
\rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\notin$} \\
F_k & F_k & F_k & & F_k & & F_k
\end{array} \tag{3.5}$$

The tuple $(b, \dots, b)$ is indeed not in F_k, since $b \notin C$, hence it is not in the "AC-part", and $b \notin D$, therefore it is not in the "BD-part".

Let now $l > k$ and $\mathbf{c}_1, \dots, \mathbf{c}_k \in F_l$ are columns in a compatibility scheme. We know that $\mathrm{Im}(f_k) = \{b, d\}$, so the only way we could have $f(\mathbf{c}_1, \dots, \mathbf{c}_k) \notin F_l$ is if the resulting tuple was $(b, \dots, b)$. For this to happen, there would have to be the element a in every row. However, we have more rows then columns, so there would be a column $\mathbf{c}_j$ with more than one occurrence of a. But then $\mathbf{c}_j \notin F_l$. Therefore, necessarily $f(\mathbf{c}_1, \dots, \mathbf{c}_k) \in F_l$, and we have $f \in \mathrm{Pol}(F_l)$.

We have shown that for every $k \geq 3$ we have $\mathcal{F}_{k-1} \subsetneq \mathcal{F}_k$. Thus, $\mathcal{F}$ is non-finitely generated.

□

In the proof we have shown that $\mathrm{Pol}(F_{k+1}) \nsubseteq \mathrm{Pol}(F_k)$. One difference compared to the relations E_k is that we have also $\mathrm{Pol}(F_k) \nsubseteq \mathrm{Pol}(F_{k+1})$, so we really need to use clones $\mathcal{F}_k$ to define $\mathcal{F}$, not just $\mathrm{Pol}(F_k)$. We can show this easily. We fix $b \in B \setminus D$ and $d \in D$ and for $k \geq 3$ we define

$$g_k(\mathbf{x}) = \begin{cases} b & \text{if } \mathbf{x} = (b, \dots, b, \overset{i}{d}, b, \dots, b) \text{ for } i \in \{1, \dots, k\}, \\ b & \text{if } \mathbf{x} = (b, \dots, b), \\ d & \text{otherwise.} \end{cases}$$

Let us fix $k \geq 3$.

Now for $l \geq k$ we take tuples

$$\mathbf{c}_i = (\underbrace{b, \dots, b, \overset{i}{d}, b, \dots, b}_{l}) \in F_l$$

for $i = 1, \dots, k$ and we have $g_k(\mathbf{c}_1, \dots, \mathbf{c}_k) = (b, \dots, b) \notin F_l$, so $g_k \notin \mathrm{Pol}(F_l)$, in particular $g_k \notin \mathrm{Pol}(F_k)$.

On the other hand, we show $g_k \in \mathrm{Pol}(F_{k-1})$. Let $\mathbf{c}_1, \dots, \mathbf{c}_k \in F_{k-1}$. We have $\mathrm{Im}(g_k) = \{b, d\}$, so $g_k(\mathbf{c}_1, \dots, \mathbf{c}_k) \notin F_{k-1} \iff g_k(\mathbf{c}_1, \dots, \mathbf{c}_k) = (b, \dots, b)$. For that to happen, we can have only the elements b and d in the tuples $\mathbf{c}_j$. Since $\mathbf{c}_j \in F_{k-1}$, there must be at least one d in each of them. However, if we put $\mathbf{c}_j$'s as columns in a scheme, we have more columns then rows. That means there are at least two occurrences of d in one of the rows, and that row is then mapped by g_k to d. This proves that the result is not $(b, \dots, b)$, hence $g_k \in \mathrm{Pol}(F_{k-1})$. Indeed, we have showed that $\mathrm{Pol}(F_{k-1}) \not\subseteq \mathrm{Pol}(F_k)$.

It is not trivial to explicitly describe the operations in $\mathcal{F}$. We studied the clone in detail for the case

$$\begin{tikzcd}
 & \{0,1,2\} & \\
\{0,2\} \arrow[ur, hook, "/" description] & & \{0,1\} \arrow[ul, hook', "/" description] \\
 & \{0\} \arrow[ul, hook', "/" description] \arrow[ur, hook, "/" description] &
\end{tikzcd},$$

but were unsuccessful in finding an explicit description of all the operations.

3.6 Relations on a set with three distinct elements

We used a simple argument to prove that the clone defined in the previous section is non-finitely generated. In this section we use the same argument in a bit different setting. We start with relations defined in a particular way on three elements, which can behave arbitrarily on the other elements, and show they always yield a non-finitely generated clone.

Let A be a finite set and let $\{0, 1, 2\} \subseteq A$. Let $\{R_k\}_{k \geq 1}$ be a sequence of relations on A such that

(i) R_k is k-ary,

(ii) $R_k \cap (\{1,2\}^k \setminus \{\begin{pmatrix}1\\ \vdots \\ 1\end{pmatrix}\}) = \left\{ \begin{pmatrix}2\\1\\1\\ \vdots \\1\end{pmatrix}, \begin{pmatrix}1\\2\\1\\ \vdots \\1\end{pmatrix}, \dots, \begin{pmatrix}1\\1\\ \vdots \\1\\2\end{pmatrix} \right\}$, i.e., R_k restricted to the set $\{1, 2\}$ contains all the tuples with exactly one occurrence of 2, contains none of the tuples with more occurrences of 2, and might optionally contain a constant $(1, \dots, 1)$ tuple,

(iii) $\{0, 2\}^k \setminus \{(2, \dots, 2)\} \subseteq R_k$, i.e., R_k contains all 02-tuples except for the constant $(2, \dots, 2)$ that is ruled out by (ii).

The relations are defined uniquely on $\{1, 2\}$ and on $\{0, 2\}$, up to the optional $(1, \dots, 1)$ tuple, but can be arbitrary otherwise.

Note that we use the sequence of arities {1,2,3,...} for the sake of simplicity of the notation, but we could use any strictly increasing sequence of natural numbers.

Proposition 20. *Let us define clones*

$$\mathcal{R}_k = \mathrm{Pol}(R_{k+1}, R_{k+2}, \dots), k \geq 0,$$
$$\mathcal{R} = \bigcup_{k \geq 0} \mathcal{R}_k.$$

Then $\mathcal{R}$ is non-finitely generated.

Proof. First of all, $\mathcal{R}$ is a well-defined clone since $\mathcal{R}_k \subseteq \mathcal{R}_{k+1}$ for all $k \geq 0$.

For $k \geq 3$ we define k-ary operation

$$f_k(\mathbf{x}) = \begin{cases} 2 & \text{if } \mathbf{x} = (1, \dots, 1, \overset{i}{2}, 1, \dots, 1) \text{ for some } i = 1, \dots, k, \\ 0 & \text{otherwise.} \end{cases}$$

Let $k \geq 3$. We show $f_k \notin \mathcal{R}_{k-1}$, but $f_k \in \mathcal{R}_k$. The first follows from the scheme

$$\begin{array}{ccccccc} 2 & 1 & 1 & \dots & 1 & \xrightarrow{f_k} & 2 \\ 1 & 2 & 1 & \dots & 1 & \xrightarrow{f_k} & 2 \\ 1 & 1 & 2 & \dots & 1 & \xrightarrow{f_k} & 2 \\ \vdots & \vdots & \vdots & \ddots & \vdots & & \vdots \\ 1 & 1 & 1 & \dots & 2 & \xrightarrow{f_k} & 2 \\ \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\in$} & & \rotatebox{90}{$\notin$} \\ R_k & R_k & R_k & & R_k & & R_k \end{array},$$

which shows that f_k is not compatible with R_k.

For the second, let $l > k$. The image of f_k is $\{0, 2\}$, and the only 02-tuple not in R_l is the constant tuple $(2, \dots, 2)$. To obtain that as the result in a compatibility scheme, we would need to use a tuple of the form $(1, \dots, 1, 2, 1, \dots, 1)$ in every row. Since we have more rows then columns, there would be a 12-tuple with at least two occurrences of 2 as one of the columns. This column would not be in R_l, thus it could not be used in the compatibility scheme. Hence, $f_k \in R_l$. We have showed that $f_k \in \mathcal{R}_k = \operatorname{Pol}(R_{k+1}, R_{k+2}, \dots)$.

We obtain an infinite chain of strict inclusions

$$\mathcal{R}_2 \subsetneq \mathcal{R}_3 \subsetneq \dots \subsetneq \mathcal{R}_k \subsetneq \dots \subsetneq \bigcup_{k \geq 0} \mathcal{R}_k = \mathcal{R},$$

thus $\mathcal{R}$ is non-finitely generated. □

For example, this result allows us to observe that non-finitely generated clones might have "large finitely generated part", which can not "interact" enough with the operations causing the clone to be non-finitely generated. For a finite set A containing $0, 1, 2$, and $k \geq 3$, let

$$R_k = A^k \setminus \left\{ \mathbf{x} \in \{1, 2\}^k \,|\, \mathbf{x} \text{ contains at least two 2's} \right\}.$$

We define $\mathcal{R}_k$, $\mathcal{R}$ from these relations as above. Then for every $k \geq 3$, the clone $\mathcal{R}_k$ contains all the operations which do not have 2 in the image. The set of all such operations together with the projections is a finitely generated clone, as shown in Section 2.2. However, even using all these and $f_3, \dots, f_k$, we can not generate f_{k+1}.

The construction can be generalized a bit. We have already mentioned that we do not need to have R_k for every k. We only need to have relations with the desired properties for arbitrarily large arities. Another condition we can weaken is the omitting of the tuples. We omitted all the 12-tuples with at least two occurrences of the element 2. We might instead fix $l \geq 2$ and omit all 12-tuples with at least l occurrences of 2. Instead of f_k being in $\mathcal{R}_k$, we would obtain f_k

necessarily being in $\mathcal{R}_{(l-1)\cdot k}$, because for a compatibility scheme with more than $(l-1)\cdot k$ rows there would have to be a column with at least l occurrences of 2 in order to get the resulting tuple $(2,\ldots,2)$.

A different possible variant can be constructed using four elements. For A containing $0,1,2$ and 3, we take relations R_k satisfying (i) and (ii) as before, but instead of (iii) we require

$$\{0,3\}^k \setminus \{(3,\ldots,3)\} \subseteq R_k \text{ and } (3,\ldots,3) \notin R_k.$$

The only difference in the proof is that we define f_k to output 3 instead of 2.

4. Maximal clones of operations with restricted essential arity

We have seen in Section 3.3 that in a clone with B-insensitive operations with image in B we can not generate an operation of a given essential arity by composing operations of lower essential arities. In this chapter we look more closely at what is needed for operations not to generate another operation of a higher essential arity. In the first part we study essentially binary operations and construct maximal clones of essentially at most binary operations; maximal in the sense that if we added any other operation, we would be able to generate an essentially at least ternary operation.

In the second part we generalize a part of the construction for higher arities. We construct maximal clones of essentially at most k-ary operations. The union over $k \in \mathbb{N}$ will then yield non-finitely generated clones.

We will again work with the concept of insensitivity, but we will not require the operations to be insensitive to the whole image, only to parts of it separately. We will see that the insensitivity condition might even differ for each variable.

In this chapter we write some operations using a table. Our convention is that the first variable choses row, the second column.

$f(x,y)$	0	1	2	$\dots$
0	$f(0,0)$	$f(0,1)$	$f(0,2)$	$\dots$
1	$f(1,0)$	$f(1,1)$	$f(1,2)$	$\dots$
2	$f(2,0)$	$f(2,1)$	$f(2,2)$	$\dots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$

For $A_1, \dots, A_k \subseteq A$ we write equivalences on A in the form "$\alpha = A_1|A_2|\dots|A_k$". This means that α is the equivalence with equivalence classes $A_1, A_2, \dots, A_k$. We mostly consider to have $A = \{0, 1, \dots, l\}$ and use a slight abuse of the notation to write for example $01|23$ as the equivalence with equivalence classes $\{0,1\}$ and $\{2,3\}$.

For $a, b \in A$ we write $a \sim_\alpha b$ iff $(a,b) \in \alpha$.

For an equivalence $\alpha = A_1|A_2|\dots|A_k$ we say that an n-ary operation f is α-insensitive in the i-th variable, if it is A_j-insensitive in the i-th variable for all $j = 1, \dots, k$, that is, if

$$\begin{aligned}&\forall \mathbf{a} \in A^n \; \forall a_i' \in A : \\ &\quad a_i \sim_\alpha a_i' \implies f(a_1, \dots, a_{i-1}, a_i, a_{i+1}, \dots, a_n) = f(a_1, \dots, a_{i-1}, a_i', a_{i+1}, \dots, a_n).\end{aligned}$$

For an operation $f(x,y)$ on A and an element $a \in A$, we denote

$$\begin{aligned}&f(-,a) \text{ the unary operation given by } x \mapsto f(x,a),\\ &f(a,-) \text{ the unary operation given by } y \mapsto f(a,y).\end{aligned}$$

4.1 Binary case

4.1.1 Definitions, necessary conditions and examples

We start by analysing some necessary conditions to guarantee that a single binary operation can not generate an essentially at least ternary operation. Let f be a binary operation on a finite set A. We want $f(f(x,y),z)$ to be essentially at most binary, so we want one of the variables x, y or z to be fictitious, and the same for $f(z,f(x,y))$.

If $f(f(x,y),z)$ does not depend on z, it means that

$$\forall a_1,a_2,b,c \in A: \; f(f(a_1,a_2),b) = f(f(a_1,a_2),c),$$

which is equivalent to f being $\mathrm{Im}(f)$-unary in the first variable (see Definition 7 in Section 3.4.2).

If $f(f(x,y),z)$ does not depend on x, then we have

$$\forall a_2,a_3,b,c \in A: \; f(f(b,a_2),a_3) = f(f(c,a_2),a_3).$$

This is equivalent to

$$\forall a_2,a_3 \in A, \; \forall b,c \in f(A,a_2): \; f(b,a_3) = f(c,a_3),$$

that is, to f being $f(A,a)$-insensitive in the first variable for all $a \in A$, where we use the notation $f(A,a) = f(A,\{a\}) = \{f(b,a) \mid b \in A\}$. The situation is similar if the expression does not depend on y, and we have also symmetric conditions for $f(z,f(x,y))$.

All the equivalent conditions for one of the variables being non-essential in either of the expressions are described in Table 4.1.

does not depend on	$f(f(x,y),z)$	$f(z,f(x,y))$
x	$\forall a \in A: f$ is $f(A,a)$-insensitive in the 1st variable	$\forall a \in A: f$ is $f(A,a)$-insensitive in the 2nd variable
y	$\forall a \in A: f$ is $f(a,A)$-insensitive in the 1st variable	$\forall a \in A: f$ is $f(a,A)$-insensitive in the 2nd variable
z	f is $\mathrm{Im}(f)$-unary in the 1st variable	f is $\mathrm{Im}(f)$-unary in the 2nd variable

Table 4.1: Equivalent conditions for a given variable to be non-essential in the simplest compositions of a single binary operation.

The sets $f(A,a)$ and $f(A,b)$ can have a nonempty intersection for some $a \neq b$. In that case, given the operation is insensitive to both $f(A,a)$ and $f(A,b)$, it is even $(f(A,a) \cup f(A,b))$-insensitive. This gives rise to a definition of "row (resp. column) equivalence", which is the transitive closure of the relation connecting elements in the same row (resp. column) in the operation's table.

Definition 8. *Let f be a binary operation on A. On* $\mathrm{Im}(f)$ *we define equivalence $\iota_1(f)$ (resp. $\iota_2(f)$) as the transitive closure of the relation*

$$a \sim b \iff \exists x: \ a, b \in f(x, A) \ (\textit{resp. } f(A, x)),$$

i.e., it is the equivalence generated by

$$\{(f(x,a), f(x,b)) \mid a, b, x \in A\}$$
$$(\textit{resp.}\, \{(f(a,x), f(b,x)) \mid a, b, x \in A\}\,).$$

For example for the operation

$f(x,y)$	0	1	2	3
0	0	0	0	2
1	0	0	0	2
2	1	1	1	3
3	1	1	1	3

we have $\iota_1(f) = 02|13$, $\iota_2(f) = 01|23$.

The equivalence is denoted ι, as it is connected to the image of the operation. We can now reformulate the first two rows of Table 4.1 as in Table 4.2.

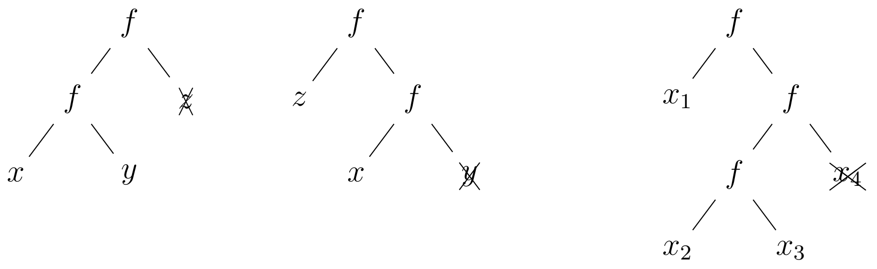
Note that the operation from the example above meets the first condition in the first column, but does not meet any condition from the second column of Table 4.1.

does not depend on	$f(f(x,y),z)$	$f(z,f(x,y))$
x	f is $\iota_2(f)$-insensitive in the 1st variable	f is $\iota_2(f)$-insensitive in the 2nd variable
y	f is $\iota_1(f)$-insensitive in the 1st variable	f is $\iota_1(f)$-insensitive in the 2nd variable

Table 4.2: The conditions from Table 4.1 for x and y being non-essential in the simple compositions equivalently reformulated using the equivalences $\iota_1(f)$, $\iota_2(f)$.

We know that if a binary operation f can not generate essentially ternary operation, then it meets at least one of the conditions in each of the columns in Table 4.1. The converse implication, however, does not hold; we demonstrate this in the following example.

Let us suppose we have a binary operation f, and we know that $f(f(x,y),z)$ does not depend on z and $f(z, f(x,y))$ does not depend on y. The problem now might be if we compose $f(x_1, f(f(x_2,x_3),x_4))$, since the only variable guaranteed to be non-essential by our assumptions is x_4. The situation is illustrated below. The crossed out variables are assumed or guaranteed to be non-essential.



Indeed, the term on the right will be essentially ternary, if we take f given by the following table:

$f(x,y)$	0	1	2	3
0	0	0	0	0
1	2	2	2	2
2	0	0	0	0
3	0	0	1	0

(4.1)

The operation is $\mathrm{Im}(f)$-unary in the first variable, and the row equivalence of f is $\iota_1(f) = 01|2$, so we see that f is $\iota_1(f)$-insensitive in the second variable. Since $f(f(x,y),z)$ is essentially binary, we can perceive it as a binary operation $\tilde{f}(x,y) = f(f(x,y),\cancel{z})$. The table of $\tilde{f}$ is:

$\tilde{f}(x,y)$	0	1	2	3
0	0	0	0	0
1	0	0	0	0
2	0	0	0	0
3	0	0	2	0

The operation $f(x_1, \tilde{f}(x_2,x_3))$ is essentially ternary. The variables x_2 and x_3 are essential, because $\tilde{f}$ can switch between the results 0 and 2 in both variables and f can distinguish between 0 and 2 in the second variable. Put differently, the operation f is neither $\iota_1(\tilde{f})$-insensitive nor $\iota_2(\tilde{f})$-insensitive in the second variable. The variable x_1 is obviously essential, since none of the columns in the table of f is constant.

The reason we were able to obtain $\tilde{f}$ with such properties is that f is able to generate the unary operation

x	0	1	2	3
$u(x)$	0	2	0	0

.

We know that f is $\iota_1(f)$-insensitive. However, $(u(0),u(1)) \notin \iota_1(f)$ even though $(0,1) \in \iota_1(f)$, and f is sensitive to $\{u(0),u(1)\}$.

Therefore, if we only assume that f is, for instance, $\iota_1(f)$-insensitive, and we want to guarantee that it can not generate an essentially ternary operation, we need to also assume that every unary operation u, that we can generate, meets the condition

$$\forall a,b \in A: \ (a,b) \in \iota_1(f) \Longrightarrow (u(a),u(b)) \in \iota_1(f),$$

i.e., that u is compatible with $\iota_1(f)$.

Given an essentially binary operation f, which can not generate an essentially ternary operation, we aim to find a maximal clone of essentially binary operations containing f. A necessary condition for g to be in such a clone is that none of the operations $f(g(x,y),z)$, $f(z,g(x,y))$, $g(f(x,y),z)$ and $g(z,f(x,y))$ is essentially ternary. The conditions for that are the same as for composing f alone and are, for clarity, summarized in Table 4.3.

We need to distinguish several cases. The simplest case is when $\mathrm{Im}(f) = A$ and f is $\iota_i(f)$-insensitive for either $i = 1$ or $i = 2$ in both variables. The main idea is then to fix the equivalence $\alpha = \iota_i(f)$ and consider essentially binary operations

does not depend on	$f(g(x,y),z)$	$f(z,g(x,y))$
x	f is $\iota_2(g)$-insensitive in the 1st variable	f is $\iota_2(g)$-insensitive in the 2nd variable
y	f is $\iota_1(g)$-insensitive in the 1st variable	f is $\iota_1(g)$-insensitive in the 2nd variable
z	f is $\mathrm{Im}(g)$-unary in the 1st variable	f is $\mathrm{Im}(g)$-unary in the 2nd variable

Table 4.3: Equivalent conditions for a given variable to be non-essential in the simplest compositions of two binary operations. We refer to f as the "top" operation and g as the "bottom" operation.

g such that the following two conditions hold. First, either $\iota_1(g) \subseteq \alpha$ or $\iota_2(g) \subseteq \alpha$. This is to satisfy the conditions in Table 4.3 where g is the "bottom" operation. Second, g is α-insensitive, which is for the case when g is the "top" operation.

For the more general case, when we also allow $\mathrm{Im}(f) \subsetneq A$, so f might also be $\mathrm{Im}(f)$-unary instead of $\iota_i(f)$-insensitive in one of the variables, we fix $B = \mathrm{Im}(f)$. We then also allow operations g which are B-unary instead of α-insensitive in one of the variables. We will look more closely at this case in Section 4.1.2. Let us remark that if g is A-unary in one of the variables, then it is essentially unary.

Weaker constraints may be applied for essentially unary operations – we just need to make sure they do not modify some of the binary operations to violate the binary conditions. The main idea is, as described after the example above, to take operations compatible with α.

Another case is when f is insensitive to different equivalences in each of the variables. Then we fix two equivalences. We examine this case in Section 4.1.3.

Note that if f satisfies at least one of the conditions in each column of Table 4.3, then we know that it is insensitive to either $\iota_1(g)$ or $\iota_2(g)$ in one of the variables. This is true, because if it is the third condition that holds in both columns in Table 4.3, i.e., if f is $\mathrm{Im}(g)$-unary in both variables, then it is constant on $\mathrm{Im}(g) \times A \cup A \times \mathrm{Im}(g)$, and, therefore, it is trivially also $\iota_i(g)$-insensitive for both $i = 1, 2$.

4.1.2 The clones with one equivalence

In this section we describe clones of essentially at most binary operations on a finite set A, defined for a given set $B \subseteq A$ and an equivalence α on A, which only glues elements in B. We show that the clones are maximal essentially binary in the sense, that adding any new operation would yield an essentially ternary operation in the clone.

In this and the next section we slightly abuse the notation, and ignore the fictitious variables. That is, for an essentially at most binary operation f we write $f = f(x, y)$, which means that all the essential variables of f are among x and y. Similarly for essentially at most unary.

Now we introduce assumptions for B, α for this section. Let $A = \{0, 1, \dots, n\}$,

$B = \{0, 1 \dots, k\} \subseteq A$ and α be an equivalence on A with equivalence classes $A_1, \dots, A_s$ such that all elements not in B are singletons, i.e.,

$$\begin{aligned}\alpha &= A_1|\dots|A_r|A_{r+1}|\dots|A_s\\ &= A_1|\dots|A_r|k+1|k+2|\dots|n.\end{aligned}$$

Furthermore, let $0, 1 \in A_1$, that is, we assume that $|B| \geq 2$ and there exist two elements in B, which we denote $0, 1$, such that $(0, 1) \in \alpha$. This will always be true if $\alpha = \iota_i(f)$ and $B = \operatorname{Im}(f)$ for some essentially binary operation f. Finally, we assume that $A \setminus A_1 \neq \varnothing$. This also does not involve any loss in generality, since if f is essentially binary operation such that both $\iota_1(f)$ and $\iota_2(f)$ are trivial equivalences connecting all elements, then it can generate an essentially ternary operation (see Table 4.1 and 4.2).

For $a, b \in A$ we denote $a \sim_\alpha b$ iff $(a, b) \in \alpha$.

Definition 9. *Let f be an essentially at most binary operation, $f = f(x, y)$. We define the following conditions:*

(im) *either $\iota_1(f) \subseteq \alpha$ or $\iota_2(f) \subseteq \alpha$,*

(ker) *for both variables, f is either α-insensitive or B-unary in this variable,*

(comp) *f is compatible with α, i.e., $\forall i, j\ \exists l :\ f(A_i, A_j) \subseteq A_l$.*

Let us remark that (im) is equivalent to a condition

$$\begin{aligned}&\text{either } \forall x \in A\ \exists j :\ f(x, A) \subseteq A_j\\ &\quad\text{or } \forall x \in A\ \exists j :\ f(A, x) \subseteq A_j,\end{aligned}$$

which is stronger than (comp) in the variable for which this holds. The condition (comp) is also trivially satisfied for a variable, in which f is α-insensitive. However, this does not mean that (im) and (ker) would imply (comp), as demonstrated by the operation (4.1) in the example in the previous section with $\alpha = 01|2|3$ and $B = \{0, 1, 2\}$.

Both conditions in (ker) state something about equalities of evaluations on certain tuples, so, as the name suggests, (ker) is a property of $\operatorname{Ker}(f)$.

We can now formulate the main claim of this section.

Proposition 21. *Let $\mathcal{E}(\alpha, B)$ be the set of operations on A given by*

$$\begin{aligned}&\mathcal{E}(\alpha, B) =\\ &\quad \{f \textit{ es. at most binary} \,|\, f \textit{ satisfies } (\text{im}), (\text{ker}), (\text{comp}) \textit{ and } \operatorname{Im}(f) \subseteq B\}\\ &\cup \{u \textit{ es. at most unary} \,|\, \textit{either } u(B) \subseteq B \textit{ and } u \textit{ is compatible with } \alpha\\ &\qquad\qquad\qquad\qquad \textit{or } u \textit{ is } \alpha\textit{-insensitive}\}\,.\end{aligned}$$

Then $\mathcal{E}(\alpha, B)$ is a clone and it is a maximal clone of essentially at most binary operations in the sense that adding any operation would yield an essentially at least ternary operation.

Note the similarity of the conditions with the conditions in Proposition 16 in Section 3.4.2. If we take $\alpha = 1_B$ trivial on B, we actually get $\mathcal{E}(1_B, B) = \mathrm{Pol}(E_3)$.

There are all constant operations in $\mathcal{E}(\alpha, B)$. If an essentially unary operation u satisfies the conditions for the essentially binary operations, then we have in particular $\mathrm{Im}(u) \subseteq B$ and (comp), which is stronger than the first condition for the essentially unary operations.

It is also worth noting that if an operation is α-insensitive, it is trivially compatible with α. All the operations in $\mathcal{E}(\alpha, B)$ are compatible with α, so any composition of the operations is also compatible with α.

Let us remark that in the case $A = B$ the conditions $\mathrm{Im}(f) \subseteq B$ and $u(B) \subseteq B$ are trivial, and we can also disregard the possibility of f being B-unary in a variable in the condition (ker), because if f is A-unary in some variable, it is essentially unary. This said, the following prove still covers this simpler case.

We now prove Proposition 21.

Claim 22. *$\mathcal{E}(\alpha, B)$ is a clone.*

Proof. We prove $\mathcal{E}(\alpha, B)$ is closed under the four procedures in Definition 2. It obviously contains an identity. Since all the conditions are symmetric and concern only essential variables, $\mathcal{E}(\alpha, B)$ is trivially closed under permutation of variables and introduction of fictitious variables.

Similarly, for identification of variables, we only need to check that if we identify the two essential variables of an essentially binary operation $f(x, y)$ in $\mathcal{E}(\alpha, B)$, the operation $\Delta f(x) = f(x, x)$ satisfies the conditions for essentially at most unary operations. It clearly satisfies the first condition, since compatibility with α is preserved by all the procedures, and $\mathrm{Im}(f) \subseteq B$ implies $\Delta f(B) \subseteq B$.

For the substitution we separately consider four possible combinations of essential arities of the two operations. Let $u, v \in \mathcal{E}(\alpha, B)$ be essentially at most unary. Then $(u * v)(x) = u(v(x))$ is again essentially at most unary. If v is α-insensitive, so is the composition. Let v be compatible with α and satisfy $v(B) \subseteq B$. Then either the same holds for u, and then it clearly also holds for the composition, or u is α-insensitive. Then for $a \sim_\alpha a'$ we have $v(a) \sim_\alpha v(a')$, and so $u(v(a)) = u(v(a'))$, hence $u * v$ is α-insensitive.

Let $u \in \mathcal{E}(\alpha, B)$ be again essentially at most unary, and $f \in \mathcal{E}(\alpha, B)$ be essentially binary. Both $u * f$ and $f * u$ are essentially at most binary and are compatible with α.

For the substitution $u * f = u(f)$, assume that $\iota_1(f) \subseteq \alpha$. Let first u be α-insensitive. Then it is also $\iota_1(f)$-insensitive, hence for all $a, b, c \in A$ we have $u(f(a, b)) = u(f(a, c))$, i.e., $u * f$ is essentially unary. Since f is compatible with α and u is α-insensitive, we have $u(f(a, c)) = u(f(b, c))$ whenever $a \sim_\alpha b$, so $u * f$ is α-insensitive. Hence, $u * f \in \mathcal{E}(\alpha, B)$.

In the other case we have $u(B) \subseteq B$, and therefore $\mathrm{Im}(u * f) \subseteq B$. Clearly, (ker) is preserved. We only need to show (im). We assume that $\iota_1(f) \subseteq \alpha$. Therefore, $f(a, b) \sim_\alpha f(a, c)$ for all $a, b, c \in A$. Then, by the compatibility of u with α, also $u(f(a, b)) \sim_\alpha u(f(a, c))$ for all $a, b, c \in A$, and since $\iota_1(u * f)$ is generated by such pairs, we have $\iota_1(u * f) \subseteq \alpha$.

For $(f * u)(x, y) = f(u(x), y)$, clearly $\mathrm{Im}(f * u) \subseteq B$. It is easy to observe that $\iota_i(f * u) \subseteq \iota_i(f)$, hence (im) is satisfied. It remains to show (ker). It is easy to see it is preserved for the second variable, we check it for the first. Let f first be α-insensitive in the first variable, and $a \sim_\alpha b$. Then $u(a) \sim_\alpha u(b)$, and so $f(u(a), c) = f(u(b), c)$ for every $c \in A$. Hence, $f * u$ is α-insensitive in the first variable. Now let f be B-unary in the first variable. The property is preserved, if $u(B) \subseteq B$. Otherwise u is α-insensitive, and then $f * u$ is α-insensitive in the first variable. So $f * u$ satisfies (ker).

Finally, let $f, g \in \mathcal{E}(\alpha, B)$ be essentially binary and, without loss of generality, $\iota_1(g) \subseteq \alpha$. First, let f be α-insensitive in the first variable. Then the operation $(f * g)(x, y, z) = f(g(x, y), z)$ does not depend on y. Let $a \in A$ and $\tilde{g} = g(-, a)$ be the essentially unary operation given by fixing a in the second coordinate of g. Then, up to adding fictitious variables, $f * g = f * \tilde{g}$. By the cases proved above, $\tilde{g} \in \mathcal{E}(\alpha, B)$, since all constants are in $\mathcal{E}(\alpha, B)$, and, consequently, also $f * \tilde{g} \in \mathcal{E}(\alpha, B)$.

Now let f be B-unary in the first coordinate. Then $(f * g)(x, y, z)$ does not depend on z. Similarly as before, we fix a constant in the second variable of f, and get an essentially unary operation $\tilde{f}$, for which $f * g = \tilde{f} * g \in \mathcal{E}(\alpha, B)$ follows from the cases above. □

Claim 23. *For any $g \notin \mathcal{E}(\alpha, B)$,* $\mathrm{Clo}(\mathcal{E}(\alpha, B) \cup \{g\})$ *contains an essentially at least ternary operation.*

Proof. We first show how to construct an essentially ternary operation, if we are given an essentially binary g which does not satisfy one of the conditions $\mathrm{Im}(g) \subseteq B$, (im) or (ker). Then we show that given an essentially unary operation not in $\mathcal{E}(\alpha, B)$, we can construct an essentially binary operation, which does not satisfy one of the aforementioned conditions.

If an essentially binary operation g does not satisfy (comp), then we have $g(b, a) \not\sim_\alpha g(c, a)$ or $g(a, b) \not\sim_\alpha g(a, c)$ for some $b \sim_\alpha c$ and a fixed $a \in A$. Using a constant, we can then generate an essentially unary operation $x \mapsto g(x, a)$ or $y \mapsto g(a, y)$, which is not compatible with α, i.e., is not in $\mathcal{E}(\alpha, B)$.

If we add an operation with essential arity greater then two, the conclusion is immediately satisfied.

Let $g(x, y)$ be an essentially binary operation, which does not satisfy (ker). We assume that it does not satisfy (ker) in the first variable. Because g is not B-unary in the first variable, there exist $b \in B$ and $c, c' \in A$ such that

$$g(b, c) \neq g(b, c'). \tag{4.2}$$

It is also not α-insensitive in the first variable, so it can distinguish between two elements in the same equivalence class of α in the first variable; without loss of generality, let us assume that

$$g(0, a) \neq g(1, a) \tag{4.3}$$

for some $a \in A$. We define a binary operation f by

$$f(x, y) = \begin{cases} 1 & \text{if } x \in A_1, y \in A_1, \\ 0 & \text{if } x \in A_1, y \notin A_1, \\ b & \text{if } x \notin A_1. \end{cases}$$

$f(x,y)$	A_1	A_2	$\dots$	A_s
A_1	1	0	$\dots$	0
A_2	b	b	$\dots$	b
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
A_s	b	b	$\dots$	b

We defined f on α-blocks, so it is clearly α-insensitive. The image of f is $\{0,1,b\} \subseteq B$, and $\iota_1(f) = 01|b \subseteq \alpha$. Hence, $f \in \mathcal{E}(\alpha, B)$. We show that $g(f(x,y),z)$ is essentially ternary. The variable z is essential, because we have $b \in \operatorname{Im}(f)$ and (4.2). Since f can "switch between 0 and 1 in the second variable", i.e., $f(0,0) = 1$ and $f(0,d) = 0$ for $d \notin A_1$, the variable y is essential in $g(f(x,y),z)$ by (4.3). Similarly, x is essential, since by (4.3), either $g(1,a) \neq g(b,a)$ or $g(0,a) \neq g(b,a)$.

Let now $g(x,y)$ be an essentially binary operation, which does not satisfy (im). From $\iota_2(g) \nsubseteq \alpha$, there exist $c, c' \in A$ such that $c \sim_{\iota_2(g)} c'$, but $c \not\sim_\alpha c'$. By Definition 8, this means there exist $c_0, \dots, c_l \in A$ and $p_0, \dots, p_{l-1} \in A$ such that $c = c_0$, $c' = c_l$ and $c_i, c_{i+1} \in g(A, p_i)$ for all $i = 0, \dots, l-1$. Since $c \not\sim_\alpha c'$, necessarily also $c_i \not\sim_\alpha c_{i+1}$ for some i, thus we have $g(a, p_i) \not\sim_\alpha g(a', p_i)$ for some $a, a' \in A$. Similarly for $\iota_1(g) \nsubseteq \alpha$. Altogether, there exist $a, a', p, b, b', q \in A$ such that

$$c := g(a,p) \not\sim_\alpha g(a',p) =: c', \tag{4.4}$$
$$d := g(q,b) \not\sim_\alpha g(q,b') =: d'. \tag{4.5}$$

There is no loss in generality assuming that also $c \not\sim_\alpha d'$ and $d \not\sim_\alpha c'$. We define a binary operation f such that $f(c,c) = f(d,c) = 1$ and $f(c',c) = f(d',c) = 0$. This is possible, since $c, d \not\sim_\alpha c', d'$. We can define it as

$$f(x,y) = \begin{cases} 1 & \text{if } x \sim_\alpha c \text{ or } x \sim_\alpha d, \text{ and } y \sim_\alpha c, \\ 0 & \text{otherwise.} \end{cases}$$

It is easy to check that $f \in \mathcal{E}(\alpha, B)$. Now $f(g(x,y),z)$ is essentially ternary, since

$$f(g(a,p),c) \overset{(4.4)}{=} f(c,c) = 1 \neq 0 = f(c',c) \overset{(4.4)}{=} f(g(a',p),c),$$
$$f(g(q,b),c) \overset{(4.5)}{=} f(d,c) = 1 \neq 0 = f(d',c) \overset{(4.5)}{=} f(g(q,b'),c),$$
$$f(g(a,p),c) \;=\; f(c,c) = 1 \neq 0 = f(c,c') \;=\; f(g(a,p),c').$$

Finally, let $g(x,y)$ be an essentially binary operation with $\operatorname{Im}(g) \nsubseteq B$. This only makes sense for the case $B \neq A$. We define f as a projection onto the first variable on B and a "diagonal" on the rest:

$$f(x,y) = \begin{cases} x & \text{if } x \in B, \\ 1 & \text{if } x \notin B \text{ and } x = y, \\ 0 & \text{otherwise.} \end{cases}$$

$f(x,y)$	0	1	$\dots$	k	$k{+}1$	$k{+}2$	$\dots$	n
0	0	0	$\dots$	0	0	0	$\dots$	0
1	1	1	$\dots$	1	1	1	$\dots$	1
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$		$\vdots$
k	k	k	$\dots$	k	k	k	$\dots$	k
$k{+}1$	0	0	$\dots$	0	1	0	$\dots$	0
$k{+}2$	0	0	$\dots$	0	0	1	$\dots$	0
$\vdots$	$\vdots$	$\vdots$		$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
n	0	0	$\dots$	0	0	0	$\dots$	1

We check that $f \in \mathcal{E}(\alpha, B)$. It is B-unary in the first variable and α-insensitive in the second, so it satisfies (ker). Since $0 \sim_\alpha 1$, we have $\iota_1(f) \subseteq \alpha$, and (im) is also satisfied. It is clearly compatible with α, and $\operatorname{Im}(f) = B$.

We show that the operation $f(g(x,y),z)$ is essentially ternary. First of all, some $d \notin B$ is in the image of g, and $f(d,0) = 0 \neq 1 = f(d,d)$. Thus, z is essential. The variables x and y are essential, because g is essential in both variables, and for all $a \neq b$ there exists $c \in A$ such that $f(a,c) \neq f(b,c)$. We check this easily. Let $a, b \in A$, $a \neq b$. Then

$$\begin{aligned} a, b \in B &\implies f(a,0) = a \neq b = f(b,0),\\ a, b \notin B &\implies f(a,b) = 0 \neq 1 = f(b,b),\\ a \in B \setminus \{1\}, b \notin B &\implies f(a,b) = a \neq 1 = f(b,b),\\ a = 1, b \notin B &\implies f(1,0) = 1 \neq 0 = f(b,0). \end{aligned}$$

It remains to show that given an essentially unary operation $u(x) \notin \mathcal{E}(\alpha, B)$, we can construct an essentially binary operation g, which violates one of the conditions (ker), (im) or $\mathrm{Im}(g) \subseteq B$.

If $u \notin \mathcal{E}(\alpha, B)$, then one of the following holds:

(a) u is not compatible with α,

(b) $u(B) \nsubseteq B$, and it is not α-insensitive.

For the case (a), we consider the "diagonal" operation

$$f(x,y) = \begin{cases} 1 & \text{if } x \sim_\alpha y,\\ 0 & \text{otherwise.} \end{cases}$$

Clearly, $f \in \mathcal{E}(\alpha, B)$. We claim that $g(x,y) = f(u(x), y)$ does not satisfy (ker). By (a) there exist $a, a' \in B$ such that $a \sim_\alpha a'$, but $u(a) \not\sim_\alpha u(a')$. The elements a, a' must be in B, otherwise they could not be equivalent in α. Then we have

$$g(a, u(a)) = f(u(a), u(a)) = 1 \neq 0 = f(u(a'), u(a)) = g(a', u(a)),$$

hence g is not α-insensitive in the first variable. Similarly, we have

$$g(a, u(a)) = f(u(a), u(a)) = 1 \neq 0 = f(u(a), u(a')) = g(a, u(a')),$$

and, therefore, g is not B-unary in the first variable. The inequalities also show that g is essentially binary.

Alternatively, we could define $f(x,y)$ to equal a if $x, y \in A_1$, and a' otherwise. This is clearly essentially binary operation in $\mathcal{E}(\alpha, B)$. The operation $u(f(x,y))$ is then still essentially binary, but it does not satisfy (im), since $u(a) \not\sim_\alpha u(a')$ and $(u(a), u(a')) \in \iota_1(f), \iota_2(f)$.

In the case (b), there exists $b \in B$ such that $u(b) \notin B$, and $a, a' \in B$ such that $a \sim_\alpha a'$ and $u(a) \neq u(a')$. We consider an operation

$$f(x,y) = \begin{cases} a & \text{if } x \in A_1, y \in A_1,\\ a' & \text{if } x \in A_1, y \notin A_1,\\ b & \text{if } x \notin A_1. \end{cases}$$

$f(x,y)$	A_1	A_2	$\dots$	A_s
A_1	a	a'	$\dots$	a'
A_2	b	b	$\dots$	b
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
A_s	b	b	$\dots$	b

Then f is α-insensitive, $\iota_1(f) \subseteq \alpha$ since $a \sim_\alpha a'$, and $\mathrm{Im}(f)$ is in B. Therefore, $f \in \mathcal{E}(\alpha, B)$. Let $g(x,y) = u(f(x,y))$. Then $u(b) \in \mathrm{Im}(g)$, so $\mathrm{Im}(g) \nsubseteq B$. Moreover, we show g is essentially binary. We have $0 \in A_1$. Let $c \in A_2$. Then

$$g(0,0) = u(f(0,0)) = u(a) \neq u(a') = u(f(0,c)) = g(0,c),$$

and since necessarily either $u(a) \neq u(b)$ or $u(a') \neq u(b)$, we have also either

$$g(0,0) = u(f(0,0)) = u(a) \neq u(b) = u(f(c,0)) = g(c,0),$$

or

$$g(0,c) = u(f(0,c)) = u(a') \neq u(b) = u(f(c,c)) = g(c,c).$$

□

This finishes the proof of Proposition 21.

4.1.3 The clones with two equivalences

In this section we describe similar clones as before, but with two different equivalences. As an example, take an operation f given by

$f(x,y)$	0	1	2	3
0	0	1	0	1
1	0	1	0	1
2	2	3	2	3
3	2	3	2	3

.

We have $\iota_1(f) = 01|23$ and $\iota_2(f) = 02|13$. The operation is $\iota_1(f)$-insensitive in the first variable, and $\iota_2(f)$-insensitive in the second variable. It is, however, neither $\iota_1(f)$-insensitive in the second variable, nor $\iota_2(f)$-insensitive in the first. Therefore, we can not take α such that f would be in $\mathcal{E}(\alpha, A)$. In this section we show that $\mathrm{Clo}(f)$ still contains only essentially at most binary operations, by finding a maximal clone of essentially at most binary operations, which contains f.

Note that we can describe equivalences $\iota_1(f)$ and $\iota_2(f)$ using a scheme

$$\begin{array}{cc} 0 & 1 \\ 2 & 3 \end{array},$$

where rows are equivalence classes of $\iota_1(f)$, and columns are equivalence classes of $\iota_2(f)$. We will now show that we can construct a similar scheme for every binary operation on A. Recall that we have defined $\iota_i(f)$ as an equivalence on $\mathrm{Im}(f)$, not necessarily on the whole A.

Lemma 24. *Let f be a binary operation on a finite set A. Let $A_1, \ldots, A_m$ be equivalence classes of $\iota_1(f)$, and $B_1, \ldots, B_n$ be equivalence classes of $\iota_2(f)$. Then $A_i \cap B_j \neq \varnothing$ for all $i = 1, \ldots, m$ and $j = 1, \ldots, n$, and there exist an $m \times n$ table*

$$M = (M_{ij})_{\substack{0 \le i \le m \\ 0 \le j \le n}},\ M_{ij} \subseteq \mathrm{Im}(f),$$

such that

- *the equivalence classes of $\iota_1(f)$ are exactly unions of rows of M, i.e.,*

$$A_i = \bigcup_{0\le j\le n} M_{i\,j}\ ,\ \ \textit{for all } i = 1,\dots,m,$$

- *the equivalence classes of $\iota_2(f)$ are exactly unions of columns of M, i.e.,*

$$B_j = \bigcup_{0\le i\le m} M_{i\,j}\ ,\ \ \textit{for all } j = 1,\dots,n,$$

- *every element of* $\mathrm{Im}(f)$ *is in exactly one of the sets $M_{i\,j}$,*
- *$M_{i\,j} \neq \varnothing$ for all $i = 1,\dots,m$, $j = 1,\dots,n$.*

Proof. We set $M_{i\,j} := A_i \cap B_j$. We show the intersection is always nonempty. Let $i \in \{1,\dots,m\}$, $j \in \{1,\dots,n\}$. Let $a \in A_i$ and $b \in B_j$. Then there exist $a_1, a_2, b_1, b_2 \in A$ such that $a = f(a_1,a_2)$ and $b = f(b_1,b_2)$. By Definition 8, $(f(a_1,a_2), f(a_1,b_2)) \in \iota_1(f)$ and $(f(b_1,b_2), f(a_1,b_2)) \in \iota_2(f)$, hence we get $f(a_1,b_2) \in A_i \cap B_j$.

Every element of $\mathrm{Im}(f)$ is in some equivalence class of both $\iota_1(f)$ and $\iota_2(f)$, so it is also in the intersection of these two classes.

Now we have

$$\bigcup_{0\le j\le n} M_{i\,j} = \bigcup_{0\le j\le n} (A_i \cap B_j) = A_i \cap \bigcup_{0\le j\le n} B_j = A_i \cap \mathrm{Im}(f) = A_i,$$

for all $i = 1,\dots,m$, and similarly for B_j's. □

Remark. It is not hard to see, that being able to describe two equivalences α, β by such a scheme is equivalent to commutativity of α and β and $\alpha\circ\beta = \beta\circ\alpha = 1$.

It is not difficult to see that given a table M as in Lemma 24, we can construct a binary operation f such that M represents its equivalences $\iota_1(f)$, $\iota_2(f)$, and f is $\iota_1(f)$-insensitive in the first variable and $\iota_2(f)$-insensitive in the second. If all the sets $M_{i\,j}$ have just one element, we already have the table of the operation – we set $f(x,y) \in M_{i\,j}$ whenever $x \in A_i$ and $y \in B_j$. Otherwise, we need to take an operation f on some larger set A. We show just a simple example. Given

012	3
4	56
78	9

we take two more elements 10, 11, and define f on $\{1,\dots,11\}$, for example, by

$f(x,y)$	B_1	B_2	10	11
A_1	0	1	3	3
A_2	1	2	3	3
A_3	4	4	5	6
10	7	8	9	9
11	7	8	9	9 .

We will need a following simple observation, in which we also demonstrate how can we use Lemma 24.

Lemma 25. *Let α_1 and α_2 be equivalences on a set B, for which $\alpha_1 \circ \alpha_2 = \alpha_2 \circ \alpha_1 = 1$, i.e., there exists a table M for α_1, α_2 as in Lemma 24.*

1. *Let $\sim$ be any equivalence on B, and $a, a', b, b' \in B$ be such that $(a, a') \in \alpha_1$, $(b, b') \in \alpha_2$, $a \not\sim a'$ and $b \not\sim b'$. Then there exist $c, c', c'' \in B$ such that $(c, c') \in \alpha_1$, $(c, c'') \in \alpha_2$, $c \not\sim c'$ and $c \not\sim c''$.*

2. *Let $c, c', c'' \in B$ be such that $(c, c') \in \alpha_1$ and $(c, c'') \in \alpha_2$. Then there exists $d \in B$ such that $(c'', d) \in \alpha_1$ and $(c', d) \in \alpha_2$.*

$$\begin{array}{ccc} c & \sim_{\alpha_1} & c' \\ \wr_{\alpha_2} & & \wr_{\alpha_2} \\ c'' & \sim_{\alpha_1} & \exists d \end{array} \tag{4.6}$$

Proof. We use just the fact that every equivalence class of α_1 has a nonempty intersection with every class of α_2. The second claim is just a reformulation of that; we take arbitrary $d \in [c']_{\alpha_2} \cap [c'']_{\alpha_1}$, where $[e]_\gamma$ denotes an equivalence class of γ containing e.

For the first claim, let $c \in [a]_{\alpha_1} \cap [b]_{\alpha_2}$. Then we have $(c, a), (c, a') \in \alpha_1$ and $(c, b), (c, b') \in \alpha_2$. Since $a \not\sim a'$, we have either $a \not\sim c$, and set $c' := a$, or $a' \not\sim c$, and then we set $c' := a'$. Similarly, either $b \not\sim c$ or $b' \not\sim c$, and we set c'' accordingly. □

The construction of clones $\mathcal{E}(\alpha_1, \alpha_2, B)$

We now aim to define the maximal clones of essentially at most binary operations $\mathcal{E}(\alpha_1, \alpha_2, B)$. Let us fix a finite set A, nonempty subset $B \subseteq A$ and equivalences α_1 and α_2 on A such that they only identify elements in B, i.e., all the elements not in B are singletons in both α_1, α_2. Moreover, restricted to B, let both α_1 and α_2 be nontrivial, and satisfy $\alpha_1 \circ \alpha_2 = \alpha_2 \circ \alpha_1 = 1$, i.e., the conclusion of Lemma 24 holds for them. Moreover, let either $\alpha_1 \cap \alpha_2$ be nontrivial, or $B = A$.

Remark. The case with one of the equivalences connecting all the elements is covered by Section 4.1.2. If one of the equivalences did not connect any elements, by the conditions we will define, all the operations would be essentially at most unary.

We denote $a \sim_{\alpha_i} b$ iff $(a, b) \in \alpha_i$, for $i \in \{1, 2\}$.

We define analogous conditions to (im), (ker) and (comp) from Section 4.1.2. For the one equivalence case, we had compatibility with α. For a unary operation u, it means that $u(\alpha) \subseteq \alpha$. The analogy in the two equivalence case is that for both $i = 1, 2$ either α_1 or α_2 maps into α_i:

(comp 2) $(u(\alpha_1) \subseteq \alpha_1$ or $u(\alpha_2) \subseteq \alpha_1)$ and $(u(\alpha_1) \subseteq \alpha_2$ or $u(\alpha_2) \subseteq \alpha_2)$

For an essentially binary operation $f(x, y)$ we define the following conditions

(im 2) $(\iota_1(f) \subseteq \alpha_1$ or $\iota_2(f) \subseteq \alpha_1)$ and $(\iota_1(f) \subseteq \alpha_2$ or $\iota_2(f) \subseteq \alpha_2)$,

(ker 2) for both variables, f is either α_1-insensitive, α_2-insensitive or B-unary in the variable,

(comp 2) for all $a \in A$ the essentially unary operations $f(-,a), f(a,-)$ satisfy the unary version of (comp 2).

As in the case with one equivalence, the condition (comp 2) is only necessary for B-unary variables, since if an operation $u(x)$ is α_1-insensitive or α_2-insensitive, it trivially satisfies (comp 2). If f is B-unary in, let us say, the first variable, then (comp 2) actually implies even stronger condition – we can change the position of the quantifier $\forall a$:

$$\begin{gathered}(\forall a : f(\alpha_1, a) \subseteq \alpha_1 \text{ or } \forall a : f(\alpha_2, a) \subseteq \alpha_1)\\ \text{and } (\forall a : f(\alpha_1, a) \subseteq \alpha_2 \text{ or } \forall a : f(\alpha_2, a) \subseteq \alpha_2),\end{gathered}$$

since $f(-,a)$ restricted to B is the same unary operation for all $a \in A$. Analogously if f is B-unary in the second variable.

Proposition 26. *Let $\mathcal{E}(\alpha_1, \alpha_2, B)$ be the set of operations on A given by*

$$\begin{aligned}&\mathcal{E}(\alpha_1, \alpha_2, B) =\\ &\quad \{f \textit{ es. at most binary} \,|\, f \textit{ satisfies } (\text{im } 2), (\text{ker } 2), (\text{comp } 2) \textit{ and } \operatorname{Im}(f) \subseteq B\}\\ &\ \cup \{u \textit{ es. at most unary} \,|\, \textit{either } u(B) \subseteq B \textit{ and } u \textit{ satisfies } (\text{comp } 2)\\ &\qquad\qquad\qquad\qquad \textit{or } u \textit{ is either } \alpha_1\textit{-insensitive or } \alpha_2\textit{-insensitive}\}\,.\end{aligned}$$

Then $\mathcal{E}(\alpha_1, \alpha_2, B)$ is a clone and it is a maximal clone of essentially at most binary operations in the sense that adding any operation would yield an essentially at least ternary operation.

Claim 27. *$\mathcal{E}(\alpha_1, \alpha_2, B)$ is a clone.*

Proof. The steps of the proof will be the same as in the proof of Claim 22. Clearly, $\mathcal{E}(\alpha_1, \alpha_2, B)$ contains the identity, and we show that it is closed under the four procedures described in Definition 2. It is obviously closed under permutation of variables and introduction of a new fictitious variable.

Let us first check that identifying the two essential variables of an essentially binary operation $f(x,y) \in \mathcal{E}(\alpha_1, \alpha_2, B)$ yields an operation satisfying the unary conditions. Since $\operatorname{Im}(f) \subseteq B$, clearly $\Delta f(B) \subseteq B$. We check (comp 2) for Δf. Let $i \in \{1,2\}$. By (im 2) either $\iota_1(f) \subseteq \alpha_i$ or $\iota_2(f) \subseteq \alpha_i$. Let us assume the former. By the property (comp 2) and the remark after the definition, there exists $j \in \{1,2\}$, such that $f(\alpha_j, a) \subseteq \alpha_i$ for all $a \in A$. Let $a, b \in A$, $a \sim_{\alpha_j} b$. Then

$$\Delta f(a) = f(a,a) \overset{(\text{comp } 2)}{\sim_{\alpha_i}} f(b,a) \overset{(\text{im } 2)}{\sim_{\alpha_i}} f(b,b) = \Delta f(b),$$

hence $\Delta f(\alpha_j) \subseteq \Delta f(\alpha_i)$, and $\Delta f \in \mathcal{E}(\alpha_1, \alpha_2, B)$.

We show that $\mathcal{E}(\alpha_1, \alpha_2, B)$ is closed under substitution. If $u, v \in \mathcal{E}(\alpha_1, \alpha_2, B)$ are essentially at most unary and v is α_i-insensitive for $i \in \{1,2\}$, then $u * v$ is also α_i-insensitive. If u is α_i insensitive and v satisfies (comp 2), then we have $v(\alpha_j) \subseteq \alpha_i$ for either $j = 1$ or $j = 2$, so $(u * v)(\alpha_j) = u(v(\alpha_j)) \subseteq u(\alpha_i) = \{a\}$ for some $a \in A$, i.e., $u * v$ is α_j-insensitive. Otherwise, both u and v satisfy (comp 2) and $u(B), v(B) \subseteq B$. Clearly $(u * v)(B) \subseteq B$. Let $i \in \{1,2\}$. Then for some $j \in \{1,2\}$, $u(\alpha_j) \subseteq \alpha_i$, and for some $k \in \{1,2\}$, $v(\alpha_k) \subseteq \alpha_j$. Therefore, $(u * v)(\alpha_k) = u(v(\alpha_k)) \subseteq u(\alpha_j) \subseteq \alpha_i$, i.e., $u * v$ satisfies (comp 2).

Let $u(x) \in \mathcal{E}(\alpha_1, \alpha_2, B)$ be essentially unary, $f(x, y) \in \mathcal{E}(\alpha_1, \alpha_2, B)$ be essentially binary. If u is α_j-insensitive for some $j \in \{1, 2\}$, then, by (im 2), we take $i \in \{1, 2\}$ such that $\iota_i(f) \subseteq \alpha_j$. Let us assume that $\iota_1(f) \subseteq \alpha_j$. Then the second variable in the operation $(u * f)(x, y)$ is not essential, so $u * f$ is essentially unary, and, up to adding fictitious variables, equal to $u * v$, where $v \in \mathcal{E}(\alpha_1, \alpha_2, B)$ is the essentially unary operation created by substituting a constant to the second variable of f. Then $u * f \in \mathcal{E}(\alpha_1, \alpha_2, B)$ by the previous case.

Otherwise, u satisfies (comp 2) and $u(B) \subseteq B$. Clearly, $u * f$ satisfies (ker 2) and $\mathrm{Im}(u * f) \subseteq B$. Checking (comp 2) is analogous to the case of two essentially unary operations. We check (im 2). Let $i \in \{1, 2\}$. By (comp 2) for u, $u(\alpha_j) \subseteq \alpha_i$ for some $j \in \{1, 2\}$, and by (im 2) for f, $\iota_k(f) \subseteq \alpha_j$ for some $k \in \{1, 2\}$. Then $\iota_k(u * f) \subseteq u(\iota_k(f)) \subseteq u(\alpha_j) \subseteq \alpha_i$.

For the substitution $f * u$, we immediately have $\mathrm{Im}(f * u) \subseteq B$, and also (im 2), since $\iota_i(f * u) \subseteq \iota_i(f)$ for every operation u. The condition (comp 2) follows from the case of two essentially unary operations, so we only need to check (ker 2). It is preserved for the second variable, we check it for the first. It is clearly satisfied, if u is α_1-insensitive or α_2-insensitive. Otherwise, f is either B-unary in the first variable, and then $f * u$ is also B-unary in the first variable, since $u(B) \subseteq B$, or f is α_i-insensitive in the first variable, and then we have $u(\alpha_j) \subseteq \alpha_i$ for some $i, j \in \{1, 2\}$, hence $f * u$ is α_j insensitive in the first variable.

Let $f, g \in \mathcal{E}(\alpha_1, \alpha_2, B)$ be essentially binary. We show that $f * g$ is essentially at most binary. Then $f * g \in \mathcal{E}(\alpha_1, \alpha_2, B)$ will follow from the previous cases as in the proof of Claim 22.

If f is α_i insensitive in the first variable, $i \in \{1, 2\}$, then either the first or the second variable of $(f * g)(x, y, z)$ is not essential, since either $\iota_2(g) \subseteq \alpha_i$ or $\iota_1(g) \subseteq \alpha_i$, respectively. If f is B-unary in the first variable, then z is not essential in $(f * g)(x, y, z)$, since $\mathrm{Im}(g) \subseteq B$. $\square$

Claim 28. *For any $g \notin \mathcal{E}(\alpha_1, \alpha_2, B)$, $\mathrm{Clo}(\mathcal{E}(\alpha_1, \alpha_2, B) \cup \{g\})$ contains an essentially at least ternary operation.*

Proof. As in the proof of Claim 23, we only need to show that given an essentially binary operation which does not satisfy one of the conditions (ker 2), (im 2) or $\mathrm{Im}(f) \subseteq B$, we can construct an essentially ternary operation, and given an essentially unary $u \notin \mathcal{E}(\alpha_1, \alpha_2, B)$, we can construct an essentially binary operation not satisfying one of the mentioned conditions. If an essentially binary operation does not satisfy (comp 2), we can generate essentially an unary operation not satisfying (comp 2), by substituting a constant in one of the variables.

Let g be essentially binary and does not satisfy (ker 2). We assume that it does not satisfy (ker 2) in the first coordinate. By Lemma 25, if we set $a \sim b$ iff $g(a, -) = g(b, -)$, there exist $c, c', c'' \in B$ such that $c \sim_{\alpha_1} c'$, $c \sim_{\alpha_2} c''$, but $g(c, -) \neq g(c', -)$ and $g(c, -) \neq g(c'', -)$. We also have $d \in B$ such that $c' \sim_{\alpha_1} d$ and $c'' \sim_{\alpha_2} d$. We denote $A_1, \ldots, A_s$ the equivalent classes of α_1, and define

$$f(x,y) = \begin{cases} c & \text{if } x \in A_1, y \in A_1, \\ c' & \text{if } x \in A_1, y \notin A_1, \\ c'' & \text{if } x \notin A_1, y \in A_1, \\ d & \text{if } x \notin A_1, y \notin A_1. \end{cases} \qquad \begin{array}{c|cccc} f(x,y) & A_1 & A_2 & \ldots & A_s \\ \hline A_1 & c & c' & \ldots & c' \\ A_2 & c'' & d & \ldots & d \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ A_s & c'' & d & \ldots & d \end{array} \tag{4.7}$$

By the assumptions on c, c', c'', d, we have $\iota_1(f) \subseteq \alpha_1$, $\iota_2(f) \subseteq \alpha_2$ and $\mathrm{Im}(f) \subseteq B$. The operation is also obviously α_1-insensitive in both variables, hence we have $f \in \mathcal{E}(\alpha_1, \alpha_2, B)$. The operation $g(f(x,y),z)$ now obviously depends on x and y. It also depends on z, if $g(a,-)$ is not a constant for one of the elements $a \in \{c, c', c'', d\}$. We show that we can always find elements c, c', c'', d such that they satisfy the conditions above, and, moreover, $g(a,-)$ is not constant for one of them.

Let $g(c,-)$ be constant. Because g is essentially binary, we know that there exists $b \in B$ such that $g(b,-)$ is not a constant. Using the table from Lemma 24, we can find $e, f, d', d'' \in B$ such that we have a following scheme, where elements in the same row are connected by α_1, and in the same column by α_2:

$$\begin{array}{ccc} c & c' & e \\ c'' & d & d' \\ f & d'' & b \end{array} \tag{4.8}$$

If $g(e,-)$ is not constant, then $g(c,-) \neq g(e,-)$, and we take c, e, c'', d' in place of c, c', c'', d, respectively. If $g(f,-)$ is not constant, we can similarly take c, c', f, d''. Otherwise both $g(e,-)$ and $g(f,-)$ are constant, hence $g(e,-) \neq g(b,-) \neq g(f,-)$, and we can take b, e, f, c instead of c, c', d, d''.

Let g be essentially binary such that it does not satisfy (im 2); without loss of generality, let it violate the condition for α_1, i.e., $\iota_1(g) \not\subseteq \alpha_1$ and $\iota_2(g) \not\subseteq \alpha_1$. Then there exist $a, a', p, b, b', q \in A$ such that

$$\begin{aligned} g(a,p) &:= c \not\sim_{\alpha_1} c' =: g(a',p), \\ g(q,b) &:= d \not\sim_{\alpha_1} d' =: g(q,b'), \end{aligned}$$

and $c \not\sim_{\alpha_1} d'$, $d \not\sim_{\alpha_1} c'$. If $\alpha_1 \cap \alpha_2$ is nontrivial, we can construct essentially ternary operation the same way as in the proof of Claim 23 (where we would need $(0,1) \in \alpha_1 \cap \alpha_2$). In general, we need to slightly modify the construction. We assume that both equivalences are nontrivial on B, so there exist four distinct elements $0, 1, 2, 3 \in B$ such that

$$\begin{array}{ccc} 0 & \sim_{\alpha_1} & 1 \\ \wr_{\alpha_2} & & \wr_{\alpha_2} \\ 2 & \sim_{\alpha_1} & 3 \end{array} .$$

We define f as

$$f(x,y) = \begin{cases} 0 & \text{if } (x \not\sim_{\alpha_1} c \text{ and } x \not\sim_{\alpha_1} d) \text{ and } y \sim_{\alpha_1} c, \\ 2 & \text{if } (x \sim_{\alpha_1} c \text{ or } x \sim_{\alpha_1} d) \text{ and } y \sim_{\alpha_1} c, \\ 1 & \text{if } (x \not\sim_{\alpha_1} c \text{ and } x \not\sim_{\alpha_1} d) \text{ and } y \not\sim_{\alpha_1} c, \\ 3 & \text{if } (x \sim_{\alpha_1} c \text{ or } x \sim_{\alpha_1} d) \text{ and } y \not\sim_{\alpha_1} c. \end{cases}$$

f	$[c]_{\alpha_1}$	$\dots$
$[c']_{\alpha_1}$	0	1
$[d']_{\alpha_1}$	0	1
$[c]_{\alpha_1}$	2	3
$[d]_{\alpha_1}$	2	3
$\vdots$	0	1

Then $f \in \mathcal{E}(\alpha_1, \alpha_2, B)$, and $f(g(x,y),z)$ is essentially ternary.

Let now $B \subsetneq A$ and g be essentially binary such that $\mathrm{Im}(g) \not\subseteq B$. In that case we assume that $\alpha_1 \cap \alpha_2$ is nontrivial. Let $0, 1 \in B$ be such that $0 \sim_{\alpha_1} 1$ and

$0 \sim_{\alpha_2} 1$. Let us define f as the projection onto the first variable on B and as "diagonal" on the rest:

$$f(x,y) = \begin{cases} x & \text{if } x \in B, \\ 1 & \text{if } x \notin B \text{ and } x = y, \\ 0 & \text{if } x \notin B \text{ and } x \neq y. \end{cases}$$

Then $f \in \mathcal{E}(\alpha_1, \alpha_2, B)$; note in particular, that $\iota_1(f) \subseteq \alpha_1, \alpha_2$. For all $a, b \in A$, $a \neq b$, we have either $f(a,a) \neq f(b,a)$ or $f(a,b) \neq f(b,b)$. Therefore, x and y are essential in $f(g(x,y),z)$, since g is essentially binary. We assume that there exists $c \in \operatorname{Im}(g)$ such that $c \notin B$. Since $f(c,c) = 1 \neq 0 = f(c,0)$, also z is essential in $f(g(x,y),z)$.

Finally, let $u \notin \mathcal{E}(\alpha_1, \alpha_2, B)$ be essentially unary. Then one of the following holds:

(a) u does not satisfy (comp 2), i.e., there exist $i \in \{1,2\}$ and elements a, a', b, b' such that $a \sim_{\alpha_1} a'$, $b \sim_{\alpha_2} b'$, $u(a) \not\sim_{\alpha_i} u(a')$ and $u(b) \not\sim_{\alpha_i} u(b')$,

(b) $u(B) \nsubseteq B$ and it is neither α_1-insensitive nor α_2-insensitive, i.e., there exist $a, a', b, b' \in A$, $a \sim_{\alpha_1} a'$, $d \sim_{\alpha_2} d'$, $u(a) \neq u(a')$ and $u(d) \neq u(d')$.

In the case (a), we use Lemma 25 to obtain c, c', c'', d as in (4.6) such that $u(c) \not\sim_{\alpha_i} u(c')$, $u(c) \not\sim_{\alpha_i} u(c')$. We define f as in (4.7) above. Then $u(f(x,y))$ is essentially binary, but does not satisfy (im 2).

In the case (b), we also use Lemma 25 to obtain c, c', c'', d as in (4.6) such that $u(c) \neq u(c')$, $u(c) \neq u(c')$. Again, we define f as in (4.7), and by the same construction as above, using the scheme (4.8), we can guarantee that $u(d) \notin B$. We then have $u(f(x,y))$ essentially binary such that $\operatorname{Im}(f) \nsubseteq B$. □

4.1.4 Conclusion

We finish the binary case by showing that we have, indeed, found a maximal clone of essentially at most binary operations for each essentially binary operation which can not generate an essentially ternary operation.

Theorem 29. *Let f be an essentially binary operation on a finite set A such that* $\operatorname{Clo}(f)$ *only contains essentially at most binary operations. Then there either exist equivalences α_1, α_2 on A such that $f \in \mathcal{E}(\alpha_1, \alpha_2, A)$, or there exist a set $B \subseteq A$ and an equivalence α on A such that $f \in \mathcal{E}(\alpha, B)$.*

We now prove Theorem 29. Let $f(x,y)$ be as in the theorem. We have noticed in Section 4.1.1 that if f cannot generate an essentially ternary operation, then it necessarily satisfies at least one of the conditions in each column of Table 4.1. Let f first satisfy insensitivity conditions in both variables. If it is $\iota_i(f)$-insensitive in both variables for the same $i \in \{1,2\}$, we set

$$\alpha = \iota_i(f) \cup \{(a,a) \mid a \in A \setminus \operatorname{Im}(f)\}.$$

Then we have $\iota_i(f) \subseteq \alpha$ and f is α-insensitive in both variables, i.e., $f \in \mathcal{E}(\alpha, A)$. The clone $f \in \mathcal{E}(\alpha, A)$ is a maximal clone of essentially at most binary operations by Proposition 21.

Similarly, if f is $\iota_1(f)$-insensitive in one variable and $\iota_2(f)$-insensitive in the other, we set

$$\begin{aligned}\alpha_1 &= \iota_1(f) \cup \{(a,a) \mid a \in A \setminus \operatorname{Im}(f)\}\,,\\ \alpha_2 &= \iota_2(f) \cup \{(a,a) \mid a \in A \setminus \operatorname{Im}(f)\}\,,\end{aligned}$$

and have $f \in \mathcal{E}(\alpha_1, \alpha_2, A)$, which is a maximal clone of essentially at most binary operations according to Proposition 26.

Instead of A, we could also take any B such that $\operatorname{Im}(f) \subseteq B \subseteq A$ as the set in the definition of the clones. Similarly, instead of the equivalences $\iota_i(f)$ we might take any α such that $\iota_i(f) \subseteq \alpha$ and f is still α-insensitive in the corresponding variables.

In both mentioned cases, the conditions (comp), (comp 2), respectively, trivially follow from the insensitivity. They are, however, not guaranteed in the remaining case. As we have already noticed before, if an operation is $\operatorname{Im}(f)$-unary in both variables, then it is also $\operatorname{Im}(f)$-insensitive in both variables, so we have this case already covered.

Let now f be $\operatorname{Im}(f)$-unary in the first variable and either $\iota_1(f)$-insensitive or $\iota_2(f)$-insensitive in the second variable. We define an equivalence

$$\kappa_2(f) = \Big\{(a,b) \in \operatorname{Im}(f)^2 \,\big|\, \forall c \in A:\ f(c,a) = f(c,b)\Big\}\,,$$

i.e., for $a, b \in \operatorname{Im}(f)$, $a \sim_{\kappa_2(f)} b$ iff the columns a and b are identical in the table of f. It is the maximal equivalence on $\operatorname{Im}(f)$ to which f is insensitive in the second variable. The condition that f is $\iota_i(f)$-insensitive in the second variable is equivalent to $\iota_i(f) \subseteq \kappa_2(f)$.

If f is compatible with $\kappa_2(f)$, then we can set

$$\alpha = \kappa_2(f) \cup \{(a,a) \mid a \in A \setminus \operatorname{Im}(f)\}$$

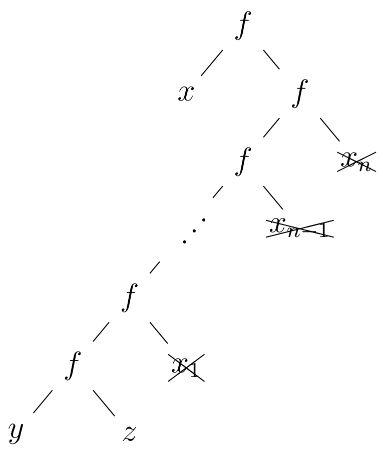
and have $f \in \mathcal{E}(\alpha, \operatorname{Im}(f))$.

Let us assume that f is not compatible with $\kappa_2(f)$. Since f is $\kappa_2(f)$-insensitive in the second variable, it is the first variable which violates the compatibility, i.e., there exist $a, a', b \in A$ such that $f(a,b) \not\sim_{\kappa_2(f)} f(a',b)$. This means, in particular, that we have $\kappa_2(f) \subsetneq B^2$ and $\iota_2(f) \not\subseteq \kappa_2(f)$, since $f(a,b) \sim_{\iota_2(f)} f(a',b)$ by the definition of $\iota_2(f)$. We have assumed f to be $\iota_i(f)$-insensitive in the second variable for either $i = 1$ or $i = 2$, so necessarily $\iota_1(f) \subseteq \kappa_2(f)$. If f is compatible with any equivalence β on $\operatorname{Im}(f)$ such that $\iota_1(f) \subseteq \beta \subseteq \kappa_2(f)$, we set

$$\alpha = \beta \cup \{(a,a) \mid a \in A \setminus \operatorname{Im}(f)\}\,,$$

and have $f \in \mathcal{E}(\alpha, \operatorname{Im}(f))$. We show that otherwise f generates an essentially ternary operation, which will conclude the proof of Theorem 29.

Let us assume that for all equivalences β on $\operatorname{Im}(f)$ such that $\iota_1(f) \subseteq \beta \subseteq \kappa_2(f)$, f is not compatible with β. We construct an essentially ternary operation similarly as for the operation (4.1) in the example in Section 4.1.1.



Let us define a unary operation $u(x)$ as a mapping $x \mapsto f(x, a)$ for an arbitrary fixed $a \in A$. We only need u as an operation on $\mathrm{Im}(f)$ and f is $\mathrm{Im}(f)$-unary in the first variable, so the choice of a does not affect anything. If $\iota_1(f) \subseteq \beta \subseteq \kappa_2(f)$ is an equivalence, then, by our assumption, we have $f(\beta, b) \not\subseteq \beta$ for some $b \in A$. Since $\beta \subseteq \mathrm{Im}(f)^2$ and f is $\mathrm{Im}(f)$-unary, we have even $f(\beta, b) \not\subseteq \beta$ for all $b \in A$. Therefore, also $u(\beta) \not\subseteq \beta$.

For $k \in \mathbb{N}$ we write $u^k(x) = \underbrace{u(u(\dots u}_{k}(x)))$ and $u^0(x) = x$.

Claim 30. *There exist $(a_0, a_0') \in \iota_1(f)$ and $n \in \mathbb{N}$ such that $u^n(a_0) \not\sim_{\kappa_2(f)} u^n(a_0')$.*

We prove the claim later. Since $(a_0, a_0') \in \iota_1(f)$, there exist $c, d, d' \in A$ such that $f(c, d) = a_0$ and $f(c, d') = a_0'$. Let us define

$$\tilde{f}(y, z) = \underbrace{f(f(\dots f(f}_{n+1}(y, z), x_1), \dots, x_n), x_{n+1}).$$

We regard $\tilde{f}$ as binary, since $x_1, \dots, x_{n+1}$ are not essential. We have

$$\tilde{f}(c, d) = u^n(f(c, d)) = u^n(a_0) \not\sim_{\kappa_2(f)} u^n(a_0') = u^n(f(c, d')) = \tilde{f}(c, d'). \qquad (4.9)$$

Let $b \in \mathrm{Im}(f)$. The operation $\tilde{f}(y, z)$ is still $\mathrm{Im}(f)$-unary in the variable y. Therefore, we have the following scheme

$$\begin{array}{ccc} \tilde{f}(b, d) & = & \tilde{f}(b, d') \\ ?\wr_{\kappa_2(f)} & & ?\wr_{\kappa_2(f)} \\ \tilde{f}(c, d) & \not\sim_{\kappa_2(f)} & \tilde{f}(c, d') \end{array},$$

and, therefore, necessarily for either $e = d$ or $e = d'$ we have

$$\tilde{f}(b, e) \not\sim_{\kappa_2(f)} \tilde{f}(c, e). \qquad (4.10)$$

We show that $f(x, \tilde{f}(y, z))$ is essentially ternary. By the definition of $\kappa_2(f)$, for every pair $p \not\sim_{\kappa_2(f)} q$, there exists $r \in A$ such that $f(r, p) \neq f(r, q)$. Therefore, by (4.9), $f(x, \tilde{f}(y, z))$ depends on z, and by (4.10), it depends on y. It clearly depends on x, since $f(\kappa_2(f), b) \not\subseteq \kappa_2(f)$ for every $b \in A$.

Indeed, $\mathrm{Clo}(f)$ contains an essentially ternary operation. It only remains to prove Claim 30.

Proof of Claim 30. For two equivalences α, β we use the standard notation $\alpha \vee \beta$ to denote the least equivalence containing both α and β. It is the equivalence generated by the edges in $\alpha \cup \beta$ and

$$a \sim_{\alpha\vee\beta} b \iff \exists c_0, \dots, c_k : \ a = c_0 \sim_\alpha c_1 \sim_\beta c_2 \sim_\alpha \cdots \sim_\beta c_k = b. \tag{4.11}$$

We assume that u is not compatible with β for every equivalence such that $\iota_1(f) \subseteq \beta \subseteq \kappa_2(f)$.

Let $\beta_0 = \iota_1(f)$. For $i \geq 0$ we define $\beta_{i+1} := \beta_i \vee u(\beta_i)$. For each $i \in \mathbb{N}$ we have either $\iota_1(f) \subseteq \beta_i \subseteq \kappa_2(f)$, and then, by our assumption, $u(\beta_i) \not\subseteq \beta_i$, hence $\beta_i \subsetneq \beta_{i+1}$, or we have $\beta_i \not\subseteq \kappa_2(f)$. Since $\mathrm{Im}(f)$ is finite, there is only finitely many equivalences on $\mathrm{Im}(f)$, and, therefore, there exists $n \in \mathbb{N}$ such that $\beta_n \not\subseteq \kappa_2(f)$. Let n be the least number such that $\beta_n \not\subseteq \kappa_2(f)$.

We now prove by induction that for all $i \in \{n, n-1, \dots, 0\}$ there exists $(a_i, a_i') \in \beta_i$ such that

$$u^{n-i}(a_i) \not\sim_{\kappa_2(f)} u^{n-i}(a_i').$$

For n we have $\beta_n \not\subseteq \kappa_2(f)$ by assumption, hence there exists $(a_n, a_n') \in \beta_n$ such that $a_n \not\sim_{\kappa_2(f)} a_n'$, i.e., $u^0(a_n) \not\sim_{\kappa_2(f)} u^0(a_n')$.

Let $1 \leq i \leq n$ and $(a_i, a_i') \in \beta_i$ be such that $u^{n-i}(a_i) \not\sim_{\kappa_2(f)} u^{n-i}(a_i')$. Then by (4.11) there exist $c_0, \dots, c_k \in \mathrm{Im}(f)$ such that

$$a_i = c_0 \sim_{\beta_{i-1}} c_1 \sim_{u(\beta_{i-1})} c_2 \sim_{\beta_{i-1}} \cdots \sim_{u(\beta_{i-1})} c_k = a_i'.$$

Since $u^{n-i}(a_i) \not\sim_{\kappa_2(f)} u^{n-i}(a_i')$, we must have $u^{n-i}(c_j) \not\sim_{\kappa_2(f)} u^{n-i}(c_{j+1})$ for some $j \in \{1, \dots, k-1\}$. Clearly, $u^{n-i}(\beta_{i-1}) \subseteq \beta_{n-1}$, thus for $(c_j, c_{j+1}) \in \beta_{i-1}$ we have

$$(u^{n-i}(c_j), u^{n-1}(c_{j+1})) \in \beta_{n-1} \subseteq \kappa_2(f).$$

Hence, there necessarily exists a tuple $(c_j, c_{j+1}) \in u(\beta_{i-1})$ such that we have $u^{n-i}(c_j) \not\sim_{\kappa_2(f)} u^{n-i}(c_{j+1})$. Thus, we get $(a_{i-1}, a_{i-1}') \in \beta_{i-1}$ such that $u(a_{i-1}) = c_j$, $u(a_{i-1}') = c_{j+1}$, and, therefore, $u^{n-i+1}(a_{i-1}) \not\sim_{\kappa_2(f)} u^{n-i+1}(a_{i-1}')$.

For $i = 0$ we have obtained the desired $(a_0, a_0') \in \beta_0 = \iota_1(f)$ such that $u^n(a_0) \not\sim_{\kappa_2(f)} u^n(a_0')$. $\square$

This finishes the proof of Theorem 29.

In the theorem we use the clones with two equivalences only for the full set A. We have, however, proved that $\mathcal{E}(\alpha_1, \alpha_2, B)$ are maximal clones of essentially binary operations for arbitrary $B \subseteq A$, $|B| \geq 2$, given that $\alpha_1 \cap \alpha_2$ is nontrivial. This is useful when we seek such a clone containing two different essentially binary operations which can not generate an essentially ternary operation together. For example, let us consider the following binary operations:

f	0	1	2	3	4	5
0	0	0	2	0	2	1
1	0	0	2	0	2	1
2	0	0	2	0	2	1
3	3	3	4	3	4	3
4	3	3	4	3	4	3
5	1	1	2	1	2	1

g	0	1	2	3	4	5
0	0	0	0	0	0	0
1	1	1	1	1	1	1
2	2	2	2	2	2	2
3	3	3	3	3	3	3
4	4	4	4	4	4	4
5	0	0	0	1	1	1

h	0	1	2	3	4	5
0	0	0	0	1	1	1
1	0	0	0	1	1	1
2	0	0	0	1	1	1
3	1	1	1	1	1	1
4	1	1	1	1	1	1
5	5	5	5	5	5	5

We have $\iota_1(f) = 012|34$ and $\iota_2(f) = 013|24$, f is $\iota_1(f)$-insensitive in the first variable and $\iota_2(f)$-insensitive in the second variable. The clone that we find for f in the proof of Theorem 29 is

$$\mathcal{E}(012|34|5, 013|24|5, \{0, \dots, 5\}) =: \mathcal{E}_f.$$

We can easily see, that $f, h \in \mathcal{E}_f$. On the other hand, $g \notin \mathcal{E}_f$, since it is neither $\{0, \dots, 5\}$-unary nor insensitive to any of the equivalences in the first variable. However, this does not mean that f and g could generate an essentially ternary operation together. To see that they can not, observe that

$$f, g \in \mathcal{E}(012|34|5, 013|24|5, \{0, \dots, 4\}) =: \mathcal{E}_g.$$

Since $5 \in \operatorname{Im}(h)$, it follows that $h \notin \mathcal{E}_g$.

We have found a maximal clone of essentially at most binary operations for the pair f, h and for the pair f, g. We cannot put all three operations in such a clone together, since we can easily see that h, g generate an essentially ternary operation $g(h(x, y), z)$. It is essentially ternary since

$$\begin{aligned} g(h(0,0),0) = 0 \neq 1 = g(h(3,0),0), \\ g(h(0,0),0) = 0 \neq 1 = g(h(0,3),0), \\ g(h(5,5),0) = 0 \neq 1 = g(h(5,5),5). \end{aligned}$$

4.2 Generalization for higher arities

In this section we show how the clones $\mathcal{E}(\alpha, B)$ with one equivalence can be generalized to maximal clones of essentially at most k-ary operations. As in Section 4.1.2, we assume that A is a finite set, $B \subseteq A$ is a subset, $|B| \geq 2$, and α is a nontrivial equivalence on A, which only connects elements in B, i.e., the equivalence classes for elements in $A \setminus B$ are just singletons. We denote $\alpha = A_1|A_2|\dots|A_s$ and assume that $0, 1 \in A_1$.

We use the simplification of notation where we ignore fictitious variables, so if an operation f is essentially k-ary, we perceive it as k-ary operation $f(x_1, \dots, x_k)$.

We need to define k-ary variant of the "row" and "column" equivalences.

Definition 10. *Let f be a k-ary operation on A. On* $\operatorname{Im}(f)$ *we define equivalence $\iota_i(f)$ as the transitive closure of the relation*

$$a \sim b \iff \exists x:\ a, b \in f(A, \dots, A, \overset{i}{x}, A, \dots, A).$$

We use analogous conditions to those described in Definition 9. For an essentially k-ary $f(x_1, \dots, x_k)$, we define:

(im) there exists $i \in \{1, \dots, k\}$ such that $\iota_i(f) \subseteq \alpha$,

(ker) for each variable, f is either α-insensitive or B-unary in this variable,

(comp) f is compatible with α.

We prove an analogy of Proposition 21.

Proposition 31. *Let $k \geq 2$ and $\mathcal{E}_k(\alpha, B)$ be the set of operations on A given by*

$$\begin{aligned}&\mathcal{E}_k(\alpha,B) = \\ &\qquad \{f \text{ es. at most } k\text{-ary} \,|\, f \text{ satisfies (im), (ker), (comp) and } \mathrm{Im}(f) \subseteq B\} \\ &\quad \cup \{u \text{ es. at most unary} \,|\, \text{either } u(B) \subseteq B \text{ and } u \text{ is compatible with } \alpha \\ &\qquad\qquad\qquad\qquad\qquad\qquad \text{or } u \text{ is } \alpha\text{-insensitive}\}.\end{aligned}$$

Then $\mathcal{E}_k(\alpha, B)$ is a clone and it is a maximal clone of essentially at most k-ary operations in the sense that adding any operation would yield an essentially at least $(k+1)$-ary operation.

The binary clone from Proposition 21 is a special case of these clones, i.e., $\mathcal{E}_2(\alpha, B) = \mathcal{E}(\alpha, B)$.

We clearly have an infinite chain of clones

$$\mathcal{E}_2(\alpha,B) \subsetneq \mathcal{E}_3(\alpha,B) \subsetneq \cdots \subsetneq \mathcal{E}_k(\alpha,B) \subsetneq \cdots \subsetneq \bigcup_{k\geq 2} \mathcal{E}_k(\alpha,B),$$

and, therefore, $\bigcup\limits_{k\geq 2} \mathcal{E}_k(\alpha, B)$ is non-finitely generated.

If $B \subsetneq A$ and we set α trivial on B, the condition (ker) says that an operation is either B-insensitive or B-unary in each variable. Also, compatibility with α is then always satisfied if $\mathrm{Im}(f) \subseteq B$. Hence, the clones $\mathcal{E}_k(\alpha, B)$ are in that case exactly the clones described in Proposition 16 in Section 3.4.2.

Note that we can not generalize the two equivalence case in such straightforward manner. The identification of variables becomes a problem for insensitivity. For example if we have a ternary operation $f(x, y, z)$ which is α_1-insensitive in x, z and α_2-insensitive in y, it can happen that for $a \sim_{\alpha_1} a'$, $c \in A$, we get $f(a, a, c) = f(a', a, c)$, but $f(a', a, c) \neq f(a', a', c)$, hence $\Delta f(a, c) \neq \Delta f(a', c)$. We did not have this problem in the binary case, since we could only get an essentially unary operation by identifying essential variables, for which we have weaker conditions.

Proof of Proposition 31. The arguments are very similar to the proof of Proposition 21. We directly use parts of the previous proof and point out how to generalize the other parts.

We fix $k \geq 2$. Clearly, $\mathcal{E}_k(\alpha, B)$ is closed under introduction of fictitious variables and under permutation of variables. It is closed under identification of variables, because if $f(x_1, \dots, x_n) \in \mathcal{E}_k(\alpha, B)$, $n \geq 2$, then either one of the first two variables is B-unary, and then the first variable Δf is also B-unary, or they are both α-insensitive, and then for all $\mathbf{a} \in A^{n-1}$ and $a_1' \in A$ such that $a_1 \sim_\alpha a_1'$ we have

$$\begin{aligned}&\Delta f(a_1, a_2, \dots, a_{n-1}) = f(a_1, a_1, a_2, \dots, a_{n-1}) = \\ &\qquad = f(a_1, a_1', a_2, \dots, a_{n-1}) = f(a_1', a_1', a_2, \dots, a_{n-1}) = \Delta f(a_1', a_2, \dots, a_{n-1}),\end{aligned}$$

hence Δf satisfies (ker). It clearly also satisfies (comp) and $\mathrm{Im}(\Delta f) \subseteq B$. It satisfies (im), since $\iota_1(\Delta f) \subseteq \iota_1(f), \iota_2(f)$ and $\iota_i(\Delta f) \subseteq \iota_{i+1}(f)$ for $i \geq 2$.

Regarding the substitution, the case for two essentially unary operations is already shown in the proof of Proposition 21, since the essentially unary operations are the same in $\mathcal{E}_l(\alpha, B)$ for all $l \geq 2$. The cases for one essentially unary and one essentially n-ary operation, $k \geq n \geq 2$, are proved the same way as in Proposition 21. The remaining case is also very similar. Let $f, g \in \mathcal{E}_k(\alpha, B)$ be essentially n-ary and m-ary, respectively, for $k \geq n, m \geq 2$. We have

$$(f * g)(x_1, x_2, \dots, x_{m+n-1}) = f(g(x_1, \dots, x_m), x_{m+1}, \dots, x_{m+n-1}).$$

If f is α-insensitive in the first coordinate and $\iota_i(g) \subseteq \alpha$, then $f * g$ does not depend on $x_1, \dots, x_{i-1}, x_{i+1}, \dots, x_m$, so it is essentially at most n-ary. If f is B-unary in the first variable, then $f * g$ does not depend on $x_{m+1}, \dots, x_{m+n-1}$, so it is essentially at most m-ary. Moreover, in both cases we can replace either g or f by a unary operation created by substituting constants to all but one variable, and then $f * g \in \mathcal{E}_k(\alpha, B)$ follows form the previous cases.

Now we show the maximality. Essentially unary and binary operations are the same in $\mathcal{E}_n(\alpha, B)$ for all $n \geq 2$. Thus, given an essentially unary operation which is not in $\mathcal{E}_k(\alpha, B)$, we construct an essentially binary operation not in $\mathcal{E}_k(\alpha, B)$ the same way we did in the proof of Proposition 21. Also, if we are given an essentially n-ary operation g which is not compatible with α, then there exist $i \in \{1, \dots, n\}$, $\mathbf{a} \in A^n$ and $a'_i \in A$ such that $a_i \sim_\alpha a'_i$ and

$$g(a_1, \dots, a_{i-1}, a_i, a_{i+1}, \dots, a_n) \not\sim_\alpha g(a_1, \dots, a_{i-1}, a'_i, a_{i+1}, \dots, a_n),$$

so we can take essentially unary operation $u(x) = g(a_1, \dots, a_{i-1}, x, a_{i+1}, \dots, a_n)$, and then u is not compatible with α, hence $u \notin \mathcal{E}_k(\alpha, B)$.

It remains to show how to generate essentially at least $(k+1)$-ary operation, given an n-ary operation, $k \geq n \geq 2$, which does not satisfy one of the conditions (ker), (im) or $\mathrm{Im}(g) \subseteq B$. For the rest of the proof, g will be an essentially n-ary operation, $k \geq n \geq 2$.

If g does not satisfy (ker) in the first variable, then it is neither B-unary nor α-insensitive in the first variable, hence there exist $b \in A$, $\mathbf{c}, \mathbf{c}' \in A^{n-1}$ and $d, d' \in A$, $\mathbf{a} \in A^{n-1}$ such that $d \sim_\alpha d'$ and

$$g(b, \mathbf{c}) \neq g(b, \mathbf{c}'), \tag{4.12}$$
$$g(d, \mathbf{a}) \neq g(d', \mathbf{a}). \tag{4.13}$$

For simplicity, assume that $d = 0, d' = 1$. Let us define k-ary operation

$$f(x_1, x_2, \dots, x_k) = \begin{cases} 1 & \text{if } x_1, x_2, \dots, x_k \in A_1, \\ 0 & \text{if } x_1 \in A_1,\ x_i \notin A_1 \text{ for some } i \in \{2, \dots, k\}, \\ b & \text{if } x_1 \notin A_1. \end{cases}$$

We can define such f, since α is nontrivial, hence has at least two equivalence classes. Clearly, f is α-insensitive. We have $\iota_1(f) = 01|b \subseteq \alpha$, so it satisfies (im), and $\mathrm{Im}(f) = \{0, 1, b\} \subseteq B$. We show that $g * f$ is essentially at least $(k+1)$-ary. Let $e \in A_2$ be arbitrary. Then for all $i \in \{1, \dots, k\}$ we have

$$g(f(0, \dots, 0, \overset{i}{0}, 0, \dots, 0), \mathbf{a}) = g(0, \mathbf{a}) \overset{(4.13)}{\neq} g(1, \mathbf{a}) = g(f(0, \dots, 0, \overset{i}{e}, 0, \dots, 0), \mathbf{a}),$$

so the first k variables are essential, and

$$g(f(e, \dots, e), \mathbf{c}) = g(b, \mathbf{c}) \overset{(4.12)}{\neq} g(b, \mathbf{c}') = g(f(e, \dots, e), \mathbf{c}'),$$

so at least one of the other $n-1$ variables is essential.

Let us now assume that g does not satisfy (im), that is, $\iota_i(g) \nsubseteq \alpha$ for all $i = 1, \dots, n$. Similarly as in the proof of Proposition 21, this means there exist $c_i, c_i', p_i \in A$ such that $c_i, c_i' \in g(A \dots, A, p_i, A, \dots, A)$ and $c_i \not\sim_\alpha c_i'$ for all $i = 1, \dots, n$.

For c_1, c_1' we find $\tilde{a}_2, \dots, \tilde{a}_n, \tilde{a}_2', \dots, \tilde{a}_n' \in A$ such that

$$g(p_1, \tilde{a}_2, \dots, \tilde{a}_n) = c_1 \not\sim_\alpha c_1' = g(p_1, \tilde{a}_2', \dots, \tilde{a}_n').$$

By changing $\tilde{a}_i$ for $\tilde{a}_i'$ one variable at a time, we find tuples $\mathbf{a}, \mathbf{a}' \in A^n$ which only differ in the i-th variable, for some $i \in \{1, \dots, n\}$, and for which $g(\mathbf{a}) \not\sim_\alpha g(\mathbf{a}')$. Similarly, starting with c_i, c_i', we find $\mathbf{b}, \mathbf{b}'$ differing only in the j-th variable such that $g(\mathbf{b}) \not\sim_\alpha g(\mathbf{b}')$, where $j \neq i$, since we had p_i at the i-th variable for all considered tuples. We have

$$\begin{aligned} c := g(\mathbf{a}) &\not\sim_\alpha g(\mathbf{a}') =: c', \\ d := g(\mathbf{b}) &\not\sim_\alpha g(\mathbf{b}') =: d'. \end{aligned}$$

We may assume that $c \not\sim_\alpha d'$ and $d \not\sim_\alpha c'$. We define a k-ary operation

$$f(x_1, x_2, \dots, x_k) = \begin{cases} 1 & \text{if } (x_1 \sim_\alpha c \text{ or } x_1 \sim_\alpha d) \text{ and } x_l \sim_\alpha c \text{ for all } l = 2, \dots, k, \\ 0 & \text{otherwise.} \end{cases}$$

We can easily check that $f \in \mathcal{E}_k(\alpha, B)$. The operation $(f * g)(x_1, \dots, x_{n+k-1})$ is essentially at least $(k+1)$-ary. The variables $x_{n+1}, \dots, x_{n+k-1}$ are essential, since for all $l = n+1, \dots, n+k-1$ we have

$$f(g(\mathbf{a}), c, \dots, c, \overset{l}{c}, c, \dots, c) = 1 \neq 0 = f(g(\mathbf{a}), c, \dots, c, \overset{l}{c'}, c, \dots, c).$$

The variables x_i and x_j are essential, since

$$\begin{aligned} f(g(\mathbf{a}), c, \dots, c) = f(c, c, \dots, c) = 0 \neq 1 = f(c', c, \dots, c) = f(g(\mathbf{a}'), c, \dots, c), \\ f(g(\mathbf{b}), c, \dots, c) = f(d, c, \dots, c) = 0 \neq 1 = f(d', c, \dots, c) = f(g(\mathbf{b}'), c, \dots, c). \end{aligned}$$

Finally, let $B \neq A$ and $\mathrm{Im}(g) \nsubseteq B$, that is, there exists $\mathbf{a} \in A^n$ such that $g(\mathbf{a}) \notin B$. We define a k-ary operation

$$f(x_1, x_2, \dots, x_k) = \begin{cases} x_1 & \text{if } x_1 \in B, \\ 1 & \text{if } x_1 \notin B \text{ and } x_1 = x_2 = \dots = x_k, \\ 0 & \text{otherwise.} \end{cases}$$

We have $\iota_1(f) \subseteq \alpha$, since it only connects 0 and 1. We have f B-unary in the first variable, and α-insensitive in the other variables, since if x_1 is fixed, then either $x_1 \in B$ and changing other variables has no effect, or $x_1 \notin B$ and then we can possibly change the result only by alternating between $x_i = x_1$ and $x_i \neq x_1$. It is clearly compatible with α, and $\mathrm{Im}(f) \subseteq B$, hence $f \in \mathcal{E}_k(\alpha, B)$.

As the binary version of f in Proposition 21, f has the property, that for every $a, b \in A$, $a \neq b$, there exists $\mathbf{c} \in A^{k-1}$ such that $f(a, \mathbf{c}) \neq f(b, \mathbf{c})$. This holds, for example because $\tilde{f}(x, y) = f(x, y, \dots, y)$ is the binary version of the operation.

Thus, operation $f(g(x_1, \ldots, x_n), x_{n+1}, \ldots, x_{n+k-1})$ depends on the variables $x_1, \ldots, x_n$, which are essential for g. The other variables, $x_{n+1}, \ldots, x_{n+k-1}$, are also essential, because if we set $b := g(\mathbf{a}) \notin B$, we have

$$f(g(\mathbf{a}), b, \ldots, b, \overset{i}{b}, b \ldots, b) = 1 \neq 0 = f(g(\mathbf{a}), b, \ldots, b, \overset{i}{0}, b, \ldots, b),$$

for all $i = n+1, \ldots, n+k-1$. Therefore, $f * g$ is essentially $(n+k-1)$-ary.

□

Conclusion

We have summarized several tools that are useful for determining whether a clone is finitely generated or not, presented examples of finitely and non-finitely generated clones, and found maximal clones of essentially binary operations.

In Chapter 2 we have studied several finitely generated clones. We have demonstrated how to modify a simple explicit construction with which we can find generators of the clones, and express all the operations of the clones as compositions of these generators. We have proved that for finite sets $B \subsetneq A$ the maximal non-finitely related clones and the idempotent reducts, defined in [13] as $\mathcal{C}(B, A)$ and $\mathcal{C}_{\mathcal{I}}(B, A)$, are finitely generated.

Inspired by the clones used in [7], in Chapter 3 we have defined relations, polymorphisms of which have restricted essential arity. It is later proved in Chapter 4 that the clones of polymorphisms of these relations are in fact maximal clones of operations with essential arity restricted by a chosen $k \geq 2$.

We have discussed possible modifications of the relations, replacing A^2, B^2, Δ_A and Δ_B by different relations. As seen in Chapter 4, restricting essential arity always involves some kind of insensitivity, which is rather limiting. We have seen, however, that other properties might be restricted to a certain number of variables – for instance, we have defined relations F_k by replacing $A^2, B^2, \Delta_A, \Delta_B$ by sets A, B, C, D, such that

$$\begin{tikzcd}
 & A & \\
B \arrow[ur, phantom, sloped, "\subsetneq"] & \nsubseteq & C \arrow[ul, phantom, sloped, "\supsetneq"] \\
 & D \arrow[ul, phantom, sloped, "\supsetneq"] \arrow[ur, phantom, sloped, "\subsetneq"] &
\end{tikzcd},$$

and then compatibility of an operation f with the relation F_k implies that the property

$$f(C, \ldots, C, \overset{i}{A}, C, \ldots, C) \nsubseteq C$$

can hold for at most $k - 1$ variables.

Note that the condition "a property P holds for at most k variables" is always trivially satisfied by at most k-ary operations. In the light of Lemma 9 in Section 1.4, the reason why a clone $\mathcal{C}$ is non-finitely generated is, in a sense, always that we have a chain of clones of operations satisfying some property in at most k variables, and for every k we have an operation in $\mathcal{C}$ satisfying the property in more than k variables.

It would be interesting to look more deeply into what kind of other property might be restricted to a certain number of variables, if we replaced $A^2, B^2, \Delta_A, \Delta_B$ by different relations. Also, for F_k, the "unary version" of the relations, the clones of polymorphisms themselves seem to be worth looking at. Despite a simple relational description, they appear to be rather problematic to describe explicitly.

In Chapter 4 we have studied clones of operations with restricted essential arity in general. For every essentially binary operation f such that $\mathrm{Clo}(f)$ contains only essentially at most binary operations, we have found a maximal clone of essentially at most binary operations containing f. At the end of Section 4.1.4 we have presented an example which shows that we can also find such a maximal clone for some pairs of operations such that they can only generate essentially at most binary operations together. The question is, whether we can do this for any such pair, or, in general, a larger tuple of operations, and whether there exists some other maximal clone of essentially at most binary operations on a finite set.

Finally, we have generalized part of the clones to larger arities, obtaining clones of essentially at most k-ary operations. These generalize the clones we have relationally described in Chapter 3. We could not, however, generalize the case with two equivalences, which offers another possibility for further study.

Bibliography

[1] K. A. Baker and A. F. Pixley. Polynomial interpolation and the chinese remainder theorem for algebraic systems. *Mathematische Zeitschrift*, 143(2): 165–174, 1975.

[2] V. G. Bodnarčuk, L. A. Kalužnin, V. N. Kotov, and B. A. Romov. Galois theory for post algebras. i, ii. (russian). *Kibernetika*, 3:1–10, 1969.

[3] A. Bulatov. Polynomial clones containing the mal'tsev operation of the groups $\mathbb{Z}_{p^2}$ and $\mathbb{Z}_p \times \mathbb{Z}_p$. 8:193–221, 01 2002.

[4] J. Demetrovics, L. Hannák, and L. Rónyai. On algebraic properties of monotone clones. *Order*, 3(3):219–225, 1986.

[5] O. S. Dudakova. Classes of functions of k-valued logic that are monotone with respect to sets of width two. *Vestnik Moskov. Univ. Ser. I Mat. Mekh.*, 63(1):31–37, 72, 2008. ISSN 0579-9368.

[6] D. Geiger. Closed systems of functions and predicates. *Pacific J. Math.*, 27: 95–100, 1968.

[7] L. Haddad. Intersections of finitely generated clones. *Algebra Universalis*, 27:171–179, 1990.

[8] P. M. Idziak. Clones containing Mal'tsev operations. *Internat. J. Algebra Comput.*, 9(2):213–226, 1999. ISSN 0218-1967.

[9] S. W. Jablonski, G. P. Gawrilov, and W. B. Kudrjawzew. *Boolesche Funktionen und Postsche Klassen.* Akademie-Verlag, Berlin, 1970.

[10] J. I. Janov and A. A. Mučnik. Existence of k-valued closed classes without a finite basis. *Dokl. Akad. Nauk SSSR*, 127:44–46, 1959. ISSN 0002-3264.

[11] D. Lau. Bestimmung der Ordnung maximaler Klassen von Funktionen der k-wertigen Logik. *Z. Math. Logik Grundlag. Math.*, 24(1):79–96, 1978. ISSN 0044-3050.

[12] A. I. Mal'cev. *Iterative Post algebras.* Novosibirsk. Gos. Univ., Novosibirsk, 1976.

[13] P. Marković, M. Maróti, and R. McKenzie. Finitely related clones and algebras with cube terms. *Order*, 29(2):345–359, 2012. ISSN 0167-8094.

[14] P. Mayr. Polynomial clones on squarefree groups. *Internat. J. Algebra Comput.*, 18(4):759–777, 2008. ISSN 0218-1967.

[15] R. Pöschel and L. A. Kalužnin. *Funktionen- und Relationenalgebren.* Matematische Monographien, VEB Deutsche Verlag der Wissenschaften, Berlin, 1979.

[16] E. L. Post. *The Two-Valued Iterative Systems of Mathematical Logic.* Annals of Mathematics Studies, no. 5. Princeton University Press, Princeton, N. J., 1941.

[17] I. G. Rosenberg. Über die funktionale vollständigkeit in den mehrwertigen logiken. *Rozpravy Československé Akad. věd, Ser. Math. Nat. Sci.*, 80(4): 3–93, 1970.

[18] G. Tardos. A maximal clone of monotone operations which is not finitely generated. *Order*, 3(3):211–218, 1986. ISSN 0167-8094.

[19] L. Zádori. Series parallel posets with non-finitely generated clones. *Order*, 10(4):305–316, 1993. ISSN 0167-8094.