Automata with group actions

Mikołaj Bojańczyk[*]
University of Warsaw

Bartek Klin[†]
University of Cambridge, University of Warsaw

Sławomir Lasota[‡]
University of Warsaw

***Abstract*—Our motivating question is a Myhill-Nerode theorem for infinite alphabets. We consider several kinds of those: alphabets whose letters can be compared only for equality, but also ones with more structure, such as a total order or a partial order. We develop a framework for studying such alphabets, where the key role is played by the automorphism group of the alphabet. This framework builds on the idea of nominal sets of Gabbay and Pitts; nominal sets are the special case of our framework where letters can be only compared for equality. We use the framework to uniformly generalize to infinite alphabets parts of automata theory, including decidability results. In the case of letters compared for equality, we obtain automata equivalent in expressive power to finite memory automata, as defined by Francez and Kaminski.**

I. Introduction

We study languages and automata over infinite alphabets. Each alphabet comes with some structure that can be accessed by recognizing devices such as automata or formulas of logic. We are particularly interested in three kinds of infinite alphabets:

- *Data values with equality.* There is an infinite set $\mathbb{D}$ whose elements are called *data values*. Words are elements of $\mathbb{D}^*$, or in some cases $(\Sigma \times \mathbb{D})^*$, for some finite set Σ. There is no structure on the data values, except for equality. A typical language is
$$\{d_1 \cdots d_n \in \mathbb{D}^* : d_{i+1} \neq d_i \text{ for all } i \in \{1, \dots, n-1\}\}.$$
- *Totally ordered data values.* The set of data values is equipped with a total order. A typical language is
$$\{d_1 \cdots d_n \in \mathbb{D}^* : d_{i+1} > d_i \text{ for all } i \in \{1, \dots, n-1\}\}.$$
- *Partially ordered data values.* The set of data values is equipped with a partial order.

Observe that, at least in the last two cases, the above descriptions do not describe the alphabets uniquely. One of the themes in this paper is the use of "universal" alphabets to obtain well-behaved notions of recognizing devices.

A device can only access data values through the given structure (e.g. equality, total order, or partial order). For instance, in the case of data values with equality, an automaton that accepts a two-letter word de with $d \neq e$, will also necessarily accept the word de' for any $e' \neq d$.

The notion of structure on an alphabet is naturally captured by the group of its automorphisms. For example, in the case of unordered data values, the group consists of all bijections on $\mathbb{D}$. In the case of totally ordered data values, it is the group of all monotone bijections on $\mathbb{D}$. For partially ordered data, automorphisms are bijections that preserve and reflect the order.

In general, we work with a set of data values $\mathbb{D}$, together with a group G of bijections of $\mathbb{D}$, which need not be the group of all bijections of $\mathbb{D}$. Such a pair $(\mathbb{D}, G)$ is called a *signature*. We then study sets X which are acted upon by the group G. A key example is the set $X = \mathbb{D}^*$, where G acts separately on every letter. As far as languages are concerned, we work with languages $L \subseteq \mathbb{D}^*$ that are closed under actions of the group G.

We now outline the main contributions of this paper.

Nominal sets for arbitrary signatures. When working with a signature $(\mathbb{D}, G)$ and a set X with an action of G, we pay attention to the interplay between the canonical action of G on $\mathbb{D}$ and the action of G on X. An example of this interplay is the definition of a nominal set. A set X is called nominal wrt. the signature if for every $x \in X$ there exists a *finite* set of data values $C \subseteq \mathbb{D}$, called a *support of* x, such that every $\pi \in G$ satisfies
$$\forall c \in C.\ \pi(c) = c \qquad \Rightarrow \qquad x \cdot \pi = x.$$
The left side of this implication uses the canonical action of π on $\mathbb{D}$, and the right side uses an action of π on X. The intuition is that x depends only on data values from C.

An example of a nominal set is $\mathbb{D}^*$, regardless of G: the support of a word can be chosen as the set of letters that appear in the word. In the case of data values with equality, where $\mathbb{D}$ is some infinite set and G is the group of all bijections on $\mathbb{D}$, the theory of nominal sets was developed by Gabbay and Pitts [14]. One of the contributions of this paper is a concept of nominal sets in different signatures.

Automata theory for arbitrary signatures. We study the theory of automata for a given signature. For basic definitions of automata and languages, we instantiate an abstract categorical theory of automata (see e.g. [1]) to categories of nominal sets. We thus connect notions of automata developed within category theory with some models that have been developed by automata theorists. We prove that, in the cases of unordered and ordered data values, the abstract definitions are expressively equivalent with existing definitions of finite memory automata [12], [8] and register automata over totally ordered data [2], [10]. Some minor changes to the latter models are needed; in fact, they help to make the automaton model robust. For instance, independently of the signature,

[*] Partially supported by the ERC Starting Grant Sosna.
[†] Supported by EPSRC grant EP/F042337/1.
[‡] Partially supported by the MNiSW grant N N206 567840.

our models admit minimization of deterministic automata. As one of our contributions, we provide an infinite-alphabet counterpart of the Myhill-Nerode theorem, thus concluding previous work on this theme [13], [2].

Effective representation and decidability. Actually, our framework has more applications than the theory of deterministic automata. We introduce a method of representing certain nominal sets, together with relations and functions on them, which can be manipulated by algorithms. We prove that an effective representation is possible in several signatures, including equality, total order and partial order. As a result we obtain a logical toolkit which, in the case of automata, uniformly proves decidability of problems such as emptiness for nondeterministic automata, language inclusion for deterministic ones or testing if a deterministic automaton is minimal. The toolkit can also be applied to study other structures such as grammars or Turing machines.

A. Background

Automata for infinite alphabets. Languages over infinite alphabets are a lively topic in the automata community. Two principal sources of motivation are XML and verification. An XML document is often modeled as a tree with labels from the (infinite) set of all Unicode strings that can appear as attribute values. In software verification, the infinite alphabet can refer to pointers or function parameters.

Many automata models have been developed for infinite alphabets, including: finite memory automata [12], automata for ordered data values [2], two-way automata and automata with pebbles [18], alternating register automata [8], data automata [5], etc. See [20] for a survey. There is no consensus as to which one is the “real” analogue of regular languages in the case of infinite alphabets. This question is a topic of debate, see e.g. [18] or [3].

Nominal sets and HD-automata. Nominal sets, studied until now in the case of unordered data values, are a convenient tool for capturing name generation and binding. They were introduced by Gabbay and Pitts [14] as a mathematical model of name-binding and α-conversion.

An fruitful line of research starting from [19] (see also [17] for an overview) uses a category equivalent to nominal sets for defining history-dependent (HD) automata, a syntax-free model of process calculi that create and pass names, like π-calculus. These are closely related to the notions of deterministic automata studied here. In fact, our representation of nominal sets, and consequently our notions of automata, are inspired by, and generalize, similar results for Gabbay-Pitts nominal sets as developed in [15], [21]. An initial connection between HD-automata and finite memory automata was made in [7].

Data monoids. The idea to use group actions in formal language theory for infinite alphabets appeared in [4], which is the closest relation to our current work. That paper already includes: a group action of bijections of data values on languages, a central role of finite supports, Myhill-Nerode congruence in the monoid setting. However, the main focus of [4] is the development of a monoid theory, including Green’s relations and an effective characterization of first-order definable word languages. This paper has a more fundamental approach. In particular we study: the connection with the literature on nominal sets, different kinds of alphabets, algorithms and methods of representing sets.

II. Motivating Example

As an introduction to the topic, we describe some issues that appear when attempting to minimize a deterministic automaton over data values with equality.

In the classical setting, over finite alphabets, the Myhill-Nerode theorem says that the minimal deterministic finite automaton for a language is obtained from the Myhill-Nerode relation, with equivalence classes of words as states. We shall now see how this works for unordered data values.

Myhill-Nerode equivalence. The Myhill-Nerode equivalence relation also makes sense for an infinite alphabet $\mathbb{D}$. That is, we consider two words $w, w' \in \mathbb{D}^*$ to be equivalent with respect to a language $L \subseteq \mathbb{D}^*$, denoted by $w \equiv_L w'$, if

$$wv \in L \Leftrightarrow w'v \in L \qquad \text{for every } v \in \mathbb{D}^*.$$

This relation is a congruence with respect to appending new letters, i.e. if $w \equiv_L w'$ then $wd \equiv_L w'd$ holds for every letter $d \in \mathbb{D}$. We call it the *syntactic congruence of* L.

We would like to have an automaton model for words over data values with equality, such that in a minimal automaton, the configurations are in one-to-one correspondence with equivalence classes of the syntactic congruence.

Example 1. In the case of a language of data words, it is unreasonable to require finitely many equivalence classes under the syntactic congruence. Consider, for example, the following language of two letter data words.

$$L = \{de : d = e\} \subseteq \mathbb{D}^2$$

The equivalence classes of the syntactic congruence are:

$$\{\epsilon\}, \qquad L, \qquad \mathbb{D}^{\geq 2} - L, \qquad \text{and} \qquad \{d\} \text{ for every } d \in \mathbb{D}.$$

This language has infinitely many equivalence classes; the infinity arises from equivalence classes $\{d\}$ for single letters. On the other hand, up to renaming data values, there are just four equivalence classes. This more relaxed notion of finiteness is the one that we use in this paper.

Equivalence classes of the syntactic congruence are consistent with a natural deterministic register automaton that recognizes the language. For L as above, the automaton has one register, which is initially undefined. It begins in an initial state q_ϵ. Then, it reads the first letter d, enters a state q and puts the letter d into a register. When it reads the second letter e, it enters an accepting state q_L if $d = e$, and it enters a sink error state $q_\perp$ otherwise. Also, when entering either q_L or $q_\perp$, the automaton erases its register.

A *configuration* of such an automaton is a state together with a register valuation. Configurations of the automaton above are in one-to-one correspondence with equivalence classes of the syntactic congruence, and the automaton defined above is isomorphic to the syntactic automaton.

Note how it is important to erase the register when entering q_L or $q_\bot$. In particular, if we want to capture the syntactic automaton, our definition of register automata needs to allow undefined registers. □

Example 2. We now show that registers that store unordered pairs are sometimes needed to capture the syntactic congruence. Consider the following simple language:

$$L = \{def : d \neq e,\ f \in \{d, e\}\} \subseteq \mathbb{D}^3. \qquad (1)$$

If we were to use the classical model of register automata [12], [8], the automaton would first put d in one register, and then e in a second register. In particular, the configurations of the automaton after reading de and after reading ed would be different. However, in the syntactic congruence, the words de and ed are equivalent. □

A quick fix to the above problem seems to be allowing registers that store unordered pairs, or unordered triples etc. The following example shows that more is needed.

Example 3. Let G be any subgroup of permutations of $\{1, \dots, n\}$. Consider the following language

$$L = \{d_1 \cdots d_n e_1 \cdots e_n : \exists \pi \in G\ \forall i\ d_i = e_{\pi(i)}\} \subseteq \mathbb{D}^{2n}.$$

The equivalence class of an n-letter word $d_1 \dots d_n$ consists of all permutations of that word under action of the group G. This example shows that in order to capture minimal automata, our notion of automaton will need to be aware of permutation groups. □

III. Group Actions

In the previous section, we applied bijections of data values not only to data values themselves, but also to words or equivalence classes of the Myhill-Nerode relation. This is a special case of a group action. This section recalls the basics of group actions and shows how they can be used to generalize automata to infinite alphabets.

Group actions. A (right) action of a group G on a set X is a function $\cdot : X \times G \to X$, written infix, subject to axioms

$$x \cdot e = x \qquad x \cdot (\pi\sigma) = (x \cdot \pi) \cdot \sigma$$

for $x \in X$ and $\pi, \sigma \in G$, where e is the neutral element of G. A set equipped with such an action is called a *G-set*.

Let us assume that G is a subgroup of the symmetric group $\mathrm{Sym}(\mathbb{D})$ (i.e. the group of all bijections of some, usually infinite, set $\mathbb{D}$ of data values). In this case, a simple example of a G-set is the set $\mathbb{D}$ itself, with the action defined by $d \cdot \pi = \pi(d)$. The action of G on $\mathbb{D}$ extends point-wise to actions of G on tuples $\mathbb{D}^k$, words $\mathbb{D}^*$, infinite words $\mathbb{D}^\omega$, or sets $\mathcal{P}(\mathbb{D})$. For any G-sets X, Y, the Cartesian product $X \times Y$ and the disjoint union $X + Y$ are G-sets with actions defined point-wise and by cases, respectively. Also, any set X is a G-set with a trivial action defined by $x \cdot \pi = x$.

For any x in a G-set X, the set

$$x \cdot G = \{x \cdot \pi \mid \pi \in G\} \subseteq X$$

is called the *orbit* of x. We will mostly be interested in *orbit-finite* sets, i.e., those that have a finite number of orbits. In the world of G-sets these play the role of finite sets.

Example 4. If $G = \mathrm{Sym}(\mathbb{D})$ for some infinite set $\mathbb{D}$, elements of the powerset $\mathcal{P}(\mathbb{D})$ are in the same orbit if and only if they have the same cardinality. As a result, $\mathcal{P}(\mathbb{D})$ is not orbit-finite. □

Equivariant relations and functions. Suppose that X is a G-set. A subset $Y \subseteq X$ is called *equivariant* if it is preserved under group actions, i.e. $Y \cdot \pi = Y$ holds for every $\pi \in G$. In other words, Y is a union of orbits in X. This definition extends to the notion of an *equivariant relation* $R \subseteq X \times Y$, by using the action of G on the Cartesian product. In the special case when R is a function f, this definition says that

$$f(x \cdot \pi) = f(x) \cdot \pi \qquad \text{for } x \in X,\ \pi \in G$$

where the action on the left is taken in X and on the right in Y. The identity function on any G-set is equivariant, and the composition of two equivariant functions is again equivariant, therefore for any group G, G-sets and equivariant functions form a category, called G-**Set**.

Example 5. Assume again $G = \mathrm{Sym}(\mathbb{D})$, and consider the set $X = \mathbb{D}^2$. There are two orbits in X, namely the diagonal $\{(d, d) : d \in \mathbb{D}\}$ and its complement. Consequently, there are four equivariant subsets of X. Likewise, for any power $n \in \mathbb{N}$, there are finitely many equivariant subsets of $\mathbb{D}^n$. □

Languages and automata. We generalize the notion of a language to the world of G-sets. An *alphabet* is any orbit-finite G-set A. Examples of alphabets include the set of data values $\mathbb{D}$, any finite set Σ, or a product $\Sigma \times \mathbb{D}$ where Σ is finite. When A is an alphabet, the set of strings A^* is treated as a G-set, with the point-wise action of G. A G-language is any equivariant subset $L \subseteq A^*$.

A classical deterministic automaton with a state space X and an alphabet A is a transition function $\delta : X \times A \to X$, a chosen initial state and a chosen subset of accepting states. Viewing an element of X as a function from a singleton set $1 = \{\star\}$ to X and a subset of X as a function from X to a two-element set 2, one can depict an automaton using a diagram:

$$\begin{tikzcd} & 1 \arrow[d, "\iota"] & \\ X \times A \arrow[r, "\delta"'] & X \arrow[r, "\alpha"'] & 2. \end{tikzcd} \qquad (2)$$

In the categorical approach to automata theory (see e.g. [1] and references therein), it is standard to define various kinds of sequential automata by instantiating this diagram in suitable categories. In this paper, we study the case of the category G-**Set**; this amounts to interpreting all objects in (2) as G-sets and arrows as equivariant functions. We consider the trivial G-action on the sets 1 and 2. This means that the initial configuration is a singleton orbit, and the set of accepting configurations is a union of orbits.

Definition 1 A *G-automaton* is any instance of the diagram (2) in G**-Set**.

The set X is called the set of *configurations*[1], and the set A is called the *input alphabet*.

Finally, just as X and A are typically assumed to be finite sets in the classical case, we may require them to be *orbit*-finite. An automaton itself is called orbit-finite if both X and A are so.

The transition function $\delta : X \times A \to X$ of an automaton extends from single letters to arbitrary words:

$$\delta^* : X \times A^* \to X.$$

The resulting function is equivariant. An automaton is called *reachable* if every configuration is equal to $\delta^*(\iota(\star), w)$ for some $w \in \mathbb{A}^*$. A word w is *accepted* by an automaton if $\delta^*(\iota(\star), w)$ is an accepting configuration. The set of accepted words is called the *language recognized by* $\mathcal{A}$. This language is a G-language, by equivariance of ι, δ^* and α.

A language is called *G-regular* if it is recognized by some orbit-finite G-automaton.

Example 6. Let $G = \mathrm{Sym}(\mathbb{D})$ for some infinite set $\mathbb{D}$. We describe a G-automaton recognizing the language

$$\{def : f \in \{d, e\}\}.$$

Its configurations are $\bot, \top$, as well as tuples of data values of size at most two:

$$X = \mathbb{D}^{\leq 2} \cup \{\top, \bot\}.$$

The idea is that the automaton, when reading the first two letters of its input, simply stores them in its configuration. Then, after the third letter, it has state $\top$ or $\bot$ depending on whether its input belongs to L or not. Such a transition function is easily seen to be equivariant. The configuration space X has six orbits: three singleton orbits

$$\{\epsilon\},\ \{\bot\},\ \{\top\},$$

and three infinite orbits

$$\{d : d \in \mathbb{D}\},\ \{(d,d) : d \in D\},\ \{(d,e) : d \neq e \in \mathbb{D}\}.$$

□

Example 7. Consider the same group G and the same language as in the previous example. We describe a different automaton for the language. Its configurations are $\bot, \top$, as well as *sets* of data values of size at most two. One can give an equivariant transition function on these configurations so that the resulting automaton recognizes L. Compared to the automaton from the previous example, the change is that instead of the orbit

$$O_1 = \{(d,e) : d \neq e \in \mathbb{D}\}$$

we have an orbit

$$O_2 = \{\{d,e\} : d \neq e \in \mathbb{D}\}.$$

In particular, both automata have six orbits. However, the new automaton is smaller in the following sense: there is an equivariant surjective function from O_1 to O_2, but there is no equivariant function from O_2 to O_1. □

Syntactic automaton. Suppose that $L \subseteq A^*$ is a G-language. We define the *syntactic automaton* of L as follows: its configurations are equivalence classes of A^* under Myhill-Nerode equivalence $\equiv_L$, its initial configuration is the equivalence class of the empty word, and accepting configurations are equivalence classes of the words in L. The transition function δ is defined by

$$\delta([w]_{\equiv_L}, a) = [wa]_{\equiv_L}.$$

The above definition is well formed, i.e. the value of the function does not depend on the choice of $w \in [w]_{\equiv_L}$.

Fact 2 The syntactic automaton of a G-language is a reachable G-automaton.

For the language in Example 6, the syntactic automaton is the one in Example 7, and not the one in Example 6.

Homomorphisms of automata. Suppose that $\mathcal{A}$ and $\mathcal{B}$ are G-automata over A. A homomorphism from $\mathcal{A}$ to $\mathcal{B}$ is an equivariant function from configurations of $\mathcal{A}$ to configurations of $\mathcal{B}$ that preserves and reflects initial and accepting configurations and commutes with the transition functions of $\mathcal{A}$ and $\mathcal{B}$.

The following result is an abstract counterpart of the Myhill-Nerode theorem for infinite alphabets:

Theorem 3 *Let L be a G-language. The syntactic automaton of L is a homomorphic image of any reachable G-automaton that recognizes L.*

Corollary 4 A G-language L is recognized by an orbit-finite G-automaton iff $A^*/\equiv_L$ is orbit-finite.

Finite representations. The notion of G-automaton presented above is quite abstract. When working with a model of computation, one expects it to have some kind of concrete presentation, e.g., in terms of states and registers. Such a presentation makes it easier to understand what the automaton does, and is necessary to design algorithms that work with automata, e.g., minimization algorithms.

One of the goals of this paper is to give a concrete presentation for orbit-finite G-sets, equivariant functions and automata, for well-behaved groups G.

Consider for instance the group G of all bijections of an infinite set $\mathbb{D}$. There is a concrete model in the literature,

[1] Why do we use the name configurations instead of states? In the sequel, some automata will be presented by giving a finite state space Q and a set of registers R. Then, a configuration of the automaton will consist of a state $q \in Q$ together with a valuation $v : R \to \mathbb{D}$ mapping registers to data values. The idea is that configurations will be orbit-finite, and states will be finite in the usual sense.

which uses registers, and which captures exactly the languages with orbit-finite syntactic automata: *finite memory automata* introduced by Francez and Kaminski [12]. However, as shown in Examples 6 and 7, a notion of symmetry on registers is necessary to describe syntactic automata, e.g. to captured unordered tuples. Such a notion of symmetry is missing from finite memory automata. Therefore, they recognize all the languages with orbit-finite syntactic automaton, but they do not minimize: the syntactic automaton is always a homomorphic image of a finite memory automaton, but in some cases it is not isomorphic to a finite memory automaton.

In Section VI, we will come back to the issue of concrete automata models. We will cover the case of data values with equality, as described in the previous case, but also a range of other cases, including total and partial orders.

A. Beyond deterministic automata

We introduced the name *G-automata* to describe automata that generalize, to the world of G-sets, the notion of a deterministic left-to-right finite automaton. Therefore, it would be more appropriate to use the name *deterministic left-to-right G-automaton*.

In the classical setting, deterministic left-to-right finite automata are equivalent to many different kinds of automata, e.g. nondeterministic, deterministic right-to-left, alternating, pebble automata, and so on. These notions can be generalized to G-sets. However, in G-sets, they all diverge, as we illustrate in the case $G = \mathrm{Sym}(\mathbb{D})$.

Right-to-left deterministic automata. Syntactically, a *right-to-left G-automaton* is the same object as a G-automaton. Only the semantics change: a reverse G-automaton reads its input from right to left. In other words, a language L is recognized by a reverse G-automaton if and only if its reverse

$$L^R = \{a_n \cdots a_1 : a_1 \cdots a_n \in L\}$$

is recognized by a G-automaton. It turns out that reverse orbit-finite G-automata do not recognize the same languages as orbit-finite G-automata. For instance, the language

$$\{a_1 \cdots a_n : a_i = a_n \text{ for some } i \in \{1, \ldots, n\}\}$$

is recognized by an orbit-finite reverse G-automaton, but not by an orbit-finite G-automaton (due to Corollary. 3.4).

Nondeterminism. A *nondeterministic G-automaton* is defined by allowing ι and δ in diagram (2) to be equivariant relations. This amounts to allowing two subsets of *initial* and *accepting* configurations, that are unions of orbits of X, and an equivariant *transition relation*

$$\delta \subseteq X \times A \times X.$$

A word $a_1 \cdots a_n$ is accepted by the automaton if there exists a sequence of configurations $x_0, \ldots, x_n$ such that x_0 is an initial configuration, x_n is an accepting configuration, and the triples $(x_0, a_1, x_1), \ldots, (x_{n-1}, a_n, x_n)$ belong to δ.

Let $G = \mathrm{Sym}(\mathbb{D})$. For any alphabet A, the language

$$\{a_1 \cdots a_n : a_i = a_j \text{ for some } i < j \in \{1, \ldots, n\}\}$$

is recognized by an orbit-finite nondeterministic G-automaton. The configurations of the automaton are letters from A, together with an initial configuration $\perp$ and an accepting configuration $\top$. By Corollary. 3.4, this language is not recognized by any deterministic orbit-finite G-automaton.

There are many other examples of situations where notions of automata diverge. Some of them have been studied in [18], in the case of $G = \mathrm{Sym}(\mathbb{D})$, and without the terminology of G-sets.

Beyond finite automata. Many other notions can be easily redefined in the world of G-sets, such as grammars, pushdown automata, or even Turing machines. For instance, the ingredients of a context-free G-grammar would be as in the classical definition, with 'finite' replaced by 'orbit-finite. Concretely: orbit-finite G-sets A and N of alphabet letters and nonterminals, respectively, a distinguished starting symbol $1 \longrightarrow N$, and a set of production rules given by an orbit-finite equivariant subset $P \subseteq N \times (N \cup A)^*$. More detailed investigation of further possibilities offered by G-sets is beyond the scope of this paper.

IV. Generalized Nominal Sets

From now on we shall consider permutation groups $G \leq \mathrm{Sym}(\mathbb{D})$ for some set $\mathbb{D}$ of data values. The pair $(\mathbb{D}, G)$ will be called a *signature*; we shall usually denote it simply by G if $\mathbb{D}$ is irrelevant or clear from the context.

In this paper, we focus on the following signatures:

- The set $\mathbb{D}$ is a countable set, say the natural numbers. The group G consists of all bijections on $\mathbb{D}$. We call this *equality signature*.
- The set $\mathbb{D} = \mathbb{Q}$ is the set of rational numbers, and G is the set of monotone bijections. We call this *total order signature*[2].
- The set $\mathbb{D}$ is the *universal partial order* (to be described below), and G is the set of its automorphisms. We call this *partial order signature*.
- The set $\mathbb{D} = \mathbb{Z}$ is the set of integers, and G is the set of translations $i \mapsto i + j$. The group G is isomorphic to the additive group of integers. We call this *integer signature*. It is used as a pathological example.

Our plan is to represent orbit-finite automata in some finite way, and run algorithms on these representations. Unfortunately the notion of G-set is too general and abstract to make this possible, as we shall now demonstrate.

Uncountably many non-isomorphic single-orbit sets. We show that in the total order signature there are uncountably many non-isomorphic single-orbit sets, which makes it impossible to represent these sets in any finite way.

Consider the powerset $\mathcal{P}(\mathbb{Q})$, with the point-wise action of G. From this powerset, extract the orbit of a single set $X \subseteq \mathbb{Q}$, i.e.

$$X \cdot G = \{X \cdot \pi : \pi \in G\} \subseteq \mathcal{P}(\mathbb{Q})$$

[2] Later it will become apparent why we chose the rational numbers, and not some other totally ordered set.

and consider it as single-orbit G-set. We want to know for which $X, Y \subseteq \mathbb{Q}$, the orbits $X \cdot G$ and $Y \cdot G$ are isomorphic.

Example 8. Consider sets $X = \{0, 1\}$, $Y = \{0, 1, 2\}$, and the closed unit interval $Z = [0; 1]$. There is no equivariant function from $X \cdot G$ to $Y \cdot G$, but there exists an equivariant isomorphism between the orbit of $X \cdot G$ and $Z \cdot G$; it is the function $X \cdot \pi \;\mapsto\; Z \cdot \pi$. □

We prove below that there are uncountably many of these G-sets, even up to isomorphism.

Theorem 5 *There is an uncountable family $\mathcal{X} \subseteq \mathcal{P}(\mathbb{Q})$ such that for any distinct $X, Y \in \mathcal{X}$, there is no equivariant isomorphism between the orbits $X \cdot G$ and $Y \cdot G$.*

As a result, there is no way to represent all single-orbit sets using data structures such as bit strings. If we plan on developing algorithms for representing G-sets, we must introduce new restrictions, so that not all of the sets in $\mathcal{X}$ need be represented. We present such a restriction now.

Generalized nominal sets. We recall the definition of a nominal set from the introduction. Consider a signature $(\mathbb{D}, G)$, and a G-set X. We say that a set $C \subseteq \mathbb{D}$ *supports* an element $x \in X$ if $x \cdot \pi = x$ for all $\pi \in G$ that act as identity on C. Observe that this definition talks about two group actions: the action of G on X, and the action of G on $\mathbb{D}$. A G-set is called *nominal* in the signature $(\mathbb{D}, G)$ if its every element has a finite support. By abuse of notation, we sometimes leave the set of data values $\mathbb{D}$ implicit, and simply talk about a nominal G-set.

The above definition generalizes the notion of nominal set introduced by Gabbay and Pitts [14]; the latter is precisely the case of equality signature.

Example 9. Consider any signature. Then $\mathbb{D}$ is a nominal G-set, since every element $d \in \mathbb{D}$ is supported by $\{d\} \subseteq \mathbb{D}$. Similarly $\{d_1, \ldots, d_k\}$ supports $(d_1, \ldots, d_k) \in \mathbb{D}^k$, hence $\mathbb{D}^k$ is also a nominal set. The same works for $\mathbb{D}^*$, but not for $\mathbb{D}^\omega$ or $\mathcal{P}(\mathbb{D})$ if $\mathbb{D}$ is infinite. □

Example 10. Consider the integer signature. If a translation $i \mapsto i + j$ preserves any single integer, then it is necessarily the identity. Therefore, any element of any set with an action of integers is supported by $\{5\}$ or $\{8\}$, etc. In the integer signature, all G-sets are nominal. □

Example 11. Consider the total order signature, and the element $x \in \mathcal{P}(\mathbb{Q})$ that is the union of two intervals $[0; 1] \cup [2; 3)$. One can show that this element is supported by the set $\{0, 1, 2, 3\}$. More generally, an element of $\mathcal{P}(\mathbb{Q})$ has a finite support if and only if it is a finite Boolean combination of intervals. In particular, there are countably many elements in $\mathcal{P}(\mathbb{Q})$ that have finite support, which eliminates the problem described in Thm. 4.1. □

Suppose that we change a signature $(\mathbb{D}, G)$ by keeping the set of data values $\mathbb{D}$, but considering a subgroup $H \leq G$. What happens to the nominal sets? If X is a G-set (and therefore also a H-set), then every G-support of $x \in X$ is also an H-support of x, therefore every nominal G-set is a nominal H-set. On the other hand, under the smaller group H, more sets might become nominal.

For any G, if X, Y are nominal G-sets then so are $X \times Y$ and $X + Y$. Indeed, if C supports $x \in X$ and D supports $y \in Y$ then $C \cup D$ supports $(x, y) \in X \times Y$, and also C supports $x \in X + Y$ and D supports $y \in X + Y$. A set X equipped with the trivial G-action is always nominal, with every element supported by the empty set.

Nominal G-sets and equivariant functions between them form a category G-**Nom**.

In the context of automata, the fact that the set of configurations is nominal means intuitively that only a finite amount of information about data values is stored in each configuration.

Minimal and least supports. An element of a nominal set always has a minimal support with respect to inclusion. As shown in Example 10, there may be many incomparable minimal supports (which means that there is no least support). Minimal supports of the same element might even have different cardinalities, as illustrated by the following example.

Example 12. Let $\mathbb{D} = \mathbb{N} \cup \mathbb{N} \times \mathbb{N}$ and let $G \leq \mathrm{Sym}(\mathbb{D})$ contain all permutations π satisfying the following conditions: $\pi(\mathbb{N}) = \mathbb{N}, \pi(\mathbb{N} \times \mathbb{N}) = \mathbb{N} \times \mathbb{N}$ and $\pi(n, m) = (\pi(n), \pi(m))$. Essentially, G contains all permutations of $\mathbb{N}$, extended coordinate-wise to $\mathbb{N} \times \mathbb{N}$. Consider $X = \mathbb{N} \times \mathbb{N}$ as a nominal G-set, with the action defined by $(m, n) \cdot \pi = \pi(m, n)$. The pair $(0, 1)$ has two minimal supports: the singleton $\{(0, 1)\}$ and the two-element set $\{0, 1\}$. □

A signature *admits least supports* if each element of every nominal G-set has the least support.

Theorem 6 *The equality signature, total order signature and partial order signature admit least supports.*

In the presence of least supports, the problems signified by Theorem 4.1 disappear.

Theorem 7 *In a signature that admits least supports, with a countable set of data values, there are only countably many orbit-finite nominal G-sets, up to an equivariant isomorphism. Furthermore, for any two orbit-finite nominal G-sets X and Y, there are only finitely many equivariant functions $f : X \to Y$.*

Having countably many orbit-finite nominal G-sets is only a necessary condition to represent these sets using data structures and to operate on them using algorithms. We revisit the topic of effective representations in the next section.

Integer pathologies. We briefly consider the integer signature as an illustration of pathologies. As we have already mentioned, it does not admit least supports, as every singleton from $\mathbb{D}$ is always a support.

As far as single-orbit nominal sets are concerned, the integer signature has a promisingly simple structure. One example of a single-orbit nominal set is $\mathbb{Z}$. Another example is the finite cyclic group $\mathbb{Z}_n$, for any $n \in \mathbb{N}$. It turns out that these are all the single-orbit sets:

Fact 8 Every single-orbit nominal set in the integer signature is isomorphic to $\mathbb{Z}$ or to $\mathbb{Z}_n$ for some $n \in \mathbb{N}$.

Equivariant functions between single-orbit sets are also simple. If the domain in $\mathbb{Z}$, these are all translations, possibly modulo n if the co-domain is $\mathbb{Z}_n$. If the domain is $\mathbb{Z}_n$, the co-domain must be necessarily $\mathbb{Z}_m$ for m a divisor of n.

The problems with the integer signature appear as soon as Cartesian products of nominal sets are considered. This has bad consequences for automata. Suppose that we are interested in automata where the set of configurations is $\mathbb{Z}$ and the input alphabet is also $\mathbb{Z}$. Both sets are single-orbit and nominal, so these are among the simplest automata in the integer signature. The transition function is any equivariant function $\delta : \mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$. What kind of functions δ can we expect? Suppose that we have defined δ for arguments of the form $(0, i)$. Then, by equivariance, this definition extends uniquely to all arguments:

$$\delta(i, j) = \delta((0, j - i) \cdot i) = \delta(0, j - i) + i.$$

However, there is no restriction on the value of $\delta(0, i)$, call it $g(i)$. It is not difficult to show that for any function $g : \mathbb{Z} \to \mathbb{Z}$, the function δ_g defined by $\delta_g(i, j) = g(j-i)+i$ is equivariant. In particular, there are uncountably many equivariant functions $\mathbb{Z} \times \mathbb{Z} \to \mathbb{Z}$.

V. Effective Representations

The key signatures studied in this paper are the equality, total order and partial order ones. It turns out that these are special cases of a *universal data domain*, to be defined in this section.

To avoid confusion with the name "signature", we use the name *vocabulary* for a set of relation names together with arities; all structures will be implicitly understood to be over some fixed finite vocabulary in this section.

For two relational structures $\mathfrak{A}$ and $\mathfrak{B}$, by an *embedding* $f : \mathfrak{A} \to \mathfrak{B}$ we mean an injective function from the carrier of $\mathfrak{A}$ to the carrier of $\mathfrak{B}$ that preserves and reflects all relations in the vocabulary. Assume a class $\mathcal{K}$ of finite structures that is closed under isomorphisms and that satisfies the following properties (we call $\mathcal{K}$ a *Fraïssé class*):

- *closure under substructures*: any substructure of a structure from $\mathcal{K}$ belongs to $\mathcal{K}$;
- *amalgamation*: if $f_{\mathfrak{B}} : \mathfrak{A} \to \mathfrak{B}$ and $f_{\mathfrak{C}} : \mathfrak{A} \to \mathfrak{C}$ are embeddings and $\mathfrak{A}, \mathfrak{B}, \mathfrak{C} \in \mathcal{K}$ then there is a structure $\mathfrak{D} \in \mathcal{K}$ together with two embeddings $f'_{\mathfrak{B}} : \mathfrak{B} \to \mathfrak{D}$ and $f'_{\mathfrak{C}} : \mathfrak{C} \to \mathfrak{D}$ that agree on the images of $f_{\mathfrak{B}}$ and $f_{\mathfrak{C}}$, i.e., $f_{\mathfrak{B}} f'_{\mathfrak{B}} = f_{\mathfrak{C}} f'_{\mathfrak{C}}$.

Examples of $\mathcal{K}$ are: finite sets (seen as relational structures over the empty vocabulary), finite total orders (seen as relational structures over a vocabulary with a single binary predicate $\leq$), finite partial orders (same vocabulary as for total order). If $\mathcal{K}$ is closed under substructures and has amalgamation, then there exists a unique, up to isomorphism, countable *universal structure* $\mathfrak{U}_{\mathcal{K}}$, known also as the Fraïssé limit of $\mathcal{K}$, with the property that the finite structures that embed into $\mathfrak{U}_{\mathcal{K}}$ are precisely those from $\mathcal{K}$ (see for instance [11])[3]. Equivalently: the set of induced finite substructures of $\mathfrak{U}_{\mathcal{K}}$ is precisely $\mathcal{K}$, up to isomorphism. Furthermore, $\mathfrak{U}_{\mathcal{K}}$ is *homogenous* in the sense that any isomorphism between two finite isomorphic substructures of $\mathfrak{U}_{\mathcal{K}}$ extends (not necessarily uniquely) to an automorphism of $\mathfrak{U}_{\mathcal{K}}$.

For the rest of this section, fix a Fraïssé class $\mathcal{K}$. From $\mathcal{K}$ we obtain a nominal signature $(\mathbb{D}_{\mathcal{K}}, G_{\mathcal{K}})$, where $\mathbb{D}_{\mathcal{K}}$ is the carrier of $\mathfrak{U}_{\mathcal{K}}$ and $G_{\mathcal{K}}$ is its group of automorphisms. Our general development will be illustrated by the cases of $\mathcal{K}$ containing finite sets, total orders and partial orders.

Example 13. Consider the case when $\mathcal{K}$ is the class of finite sets, i.e., finite structures over the empty vocabulary. Then $\mathfrak{U}_{\mathcal{K}}$ is a countable set. Automorphisms of $\mathfrak{U}_{\mathcal{K}}$ are bijections of it. This corresponds to the equality signature. □

Example 14. Consider the case when $\mathcal{K}$ is the class of finite total orders, over a vocabulary with the predicate $\leq$. Then $\mathfrak{U}_{\mathcal{K}}$ is isomorphic to the rational numbers. Automorphisms of $\mathfrak{U}_{\mathcal{K}}$ are monotone bijections of the rational numbers. This corresponds to the total order signature. □

Example 15. Consider the case when $\mathcal{K}$ is the class of finite partial orders, over a vocabulary with the predicate $\leq$. The universal structure $\mathfrak{U}_{\mathcal{K}}$ is not easily described (see e.g. [16]), except that it is partially ordered and homogenous. □

We assume for this section that $\mathcal{K}$ has the following further properties: ($\mathcal{K}$ is called *well-behaved* in this case):

- $\mathcal{K}$ is *effective*, i.e., has decidable membership;
- the nominal signature of $\mathcal{K}$ admits least supports;
- $\mathcal{K}$ is *fungible*: for any $\mathfrak{A} \in \mathcal{K}$ and any element a of $\mathfrak{A}$ there is an extension $\mathfrak{B}$ of $\mathfrak{A}$ by one element, say b, such that $\mathfrak{A}$ and $\mathfrak{B} \setminus \{a\}$ are isomorphic via the bijection that maps a to b and is identity on $\mathfrak{A} \setminus \{a\}$.

Example 16. A non-fungible class $\mathcal{K}$ whose signature admits least supports is the class of finite structures over a vocabulary consisting of a single predicate symbol P, containing those structures where P holds for at most one element. In other words, $\mathcal{K}$ is the class of finite sets, possibly with a distinguished element. On the other hand, the class of finite sets equipped with an equivalence relation with at most two equivalence classes, although fungible, does not admit least supports. □

Fact 9 In each of Examples 13–15, the class $\mathcal{K}$ is effective, fungible and admits least supports.

Under the above assumption, we have a effective way of representing orbit-finite nominal sets and equivariant functions on them. This representation will be used in Sec. VI to define a representation of $G_{\mathcal{K}}$-automata, and in Sec. VII to develop decidability results regarding orbit-finite nominal sets.

Representation of single-orbit sets. A *single-orbit representation* consists of:

- a structure $\mathfrak{A}$ from the class $\mathcal{K}$, called the *shape*, and

[3] We are grateful to Thomas Colcombet, who suggested the Fraïssé limit.

- a subgroup $S \leq \mathrm{Aut}(\mathfrak{A})$ of automorphisms of $\mathfrak{A}$, called the *symmetry*.

The semantics of such a representation, written $[\![\mathfrak{A}, S]\!]$, is the $G_{\mathcal{K}}$-set of embeddings $u : \mathfrak{A} \to \mathfrak{U}_{\mathcal{K}}$, quotiented by the equivalence $\equiv_S$ defined by:

$$u \equiv_S v \Leftrightarrow \exists \pi \in S.\ u = \pi v,$$

with the $G_{\mathcal{K}}$-action defined by composition.

In the following, we shall denote the $\equiv_S$-equivalence class of u by $[u]_S$.

Theorem 10 *Every single-orbit nominal set is isomorphic to $[\![\mathfrak{A}, S]\!]$, for some single-orbit representation $(\mathfrak{A}, S)$.*

Example 17. Let $\mathcal{K}$ be the class of finite sets as in Example 13. A single-orbit representation is a finite set C together with any group $S \leq \mathrm{Sym}(C)$. The semantics $[\![C, S]\!]$ is isomorphic to the set of coordinate-wise distinct vectors $(\mathfrak{U}_{\mathcal{K}})^S$, quotiented under coordinate permutations from S. □

Example 18. Let $\mathcal{K}$ be as in Example 14. A single-orbit representation is a finite totally ordered set C, and the symmetry S is necessarily trivial. Its semantics is the set of monotone valuations $v : C \to \mathbb{Q}$. In other words, every single orbit nominal set is isomorphic to the set of ordered tuples of rational numbers, for some fixed tuple length. □

Example 19. Let $\mathcal{K}$ be as in Example 15. A single-orbit representation is a finite partially ordered set C, together with a group of its automorphisms. □

Representation of equivariant functions. We shall now describe a representation of equivariant functions between single-orbit sets. Suppose that $(\mathfrak{A}, S)$ and $(\mathfrak{B}, T)$ are single-orbit representations. An equivariant function representation is given by an embedding $u : \mathfrak{B} \to \mathfrak{A}$ that is *consistent with symmetries* in the following way:

$$\forall \sigma \in S\ \exists \tau \in T.\ u\sigma = \tau u, \tag{3}$$

in short: $uS \subseteq Tu$. Moreover, we do not want to distinguish two embeddings $u, v : \mathfrak{B} \to \mathfrak{A}$ if $u \equiv_T v$. Formally, representations of equivariant functions are $\equiv_T$-equivalence classes of embeddings.

The semantics of $[u]_T$ is the map

$$[\![u]\!] : [\![\mathfrak{A}, S]\!] \to [\![\mathfrak{B}, T]\!]$$

defined by composition:

$$[\![u]\!]([v]_S) = [uv]_T.$$

This is well-defined thanks to (3).

Theorem 11 *Let $(\mathfrak{A}, S)$ and $(\mathfrak{B}, T)$ be single-orbit representations. The mapping $[u]_T \mapsto [\![u]\!]$ is a bijection between representations and equivariant functions from $[\![\mathfrak{A}, S]\!]$ to $[\![\mathfrak{B}, T]\!]$.*

These results are easily extended to orbit-finite nominal sets and equivariant functions between them. Indeed, note that any G-set can be seen as a disjoint union of its orbits, with the G-action defined independently for each orbit. Also equivariant functions between G-sets preserve orbits. This means that orbit-finite nominal sets can be represented by finite families of single-orbit representations, and equivariant function by families of function representations.

The above results can usefully be formulated in the language of category theory, as an equivalence of the category of nominal $G_{\mathcal{K}}$-sets and a category of representations. Due to lack of space, we omit the categorical perspective here. It should be stressed that in the case of the equality signature, this equivalence of categories is known in the literature: our category of representations is then essentially the category of "named sets with symmetries" $\mathbf{NSet}$ introduced in [19], [9]. The only difference is that in $\mathbf{NSet}$, representations (C, S) are additionally quotiented by bijections of their shapes C, and as a result the "shape" of a named set is a natural number rather than a finite set. The equivalence of G-**Nom** and $\mathbf{NSet}$ was established in [15], [21]; our Theorems 5.2 and 5.3 provide a generalization to the setting of well-behaved Fraïssé classes.

For effective representations of $G_{\mathcal{K}}$-automata, but also of algorithms that work on them, it is necessary to provide representations of Cartesian products of nominal sets in terms of representations of the components. Under our assumptions on the class $\mathcal{K}$, such representations can be derived.

Theorem 12 *Let U_1, U_2 be representations. One can compute a representation $U_1 \otimes U_2$ such that $[\![U_1 \otimes U_2]\!]$ is isomorphic to the Cartesian product $[\![U_1]\!] \times [\![U_2]\!]$. Also the equivariant projection functions can be effectively represented.*

In particular, the product of orbit-finite nominal $G_{\mathcal{K}}$-sets is orbit-finite. This is not a property of nominal sets in general. For instance, in the integer signature, the product of two single-orbit nominal sets $\mathbb{Z} \times \mathbb{Z}$ has infinitely many orbits.

Due to lack of space we omit a detailed description of the product representation; it is similar to the construction of products for $\mathbf{NSet}$ developed in [6].

VI. An Application to Automata

A nominal G-automaton, understood as in (2), is a simple combination of a few nominal G-sets and equivariant functions between them, involving a simple Cartesian product. It is therefore natural that an effective representation of nominal sets, equivariant functions and Cartesian products extends to a similar representation of automata. In this section we sketch the result of this extension in the case of signatures obtained from Fraïssé classes of relational structures.

Fraïssé automata. Fix for the rest of this section a well-behaved Fraïssé class $\mathcal{K}$ of structures. Our goal is to develop a syntax for deterministic left-to-right automata in the nominal signature of $\mathcal{K}$.

For the sake of presentation, we study automata over the alphabet $\mathbb{D}_{\mathcal{K}}$. The general case, when the alphabet is an arbitrary orbit-finite nominal $G_{\mathcal{K}}$-set, like $(\mathbb{D}_{\mathcal{K}})^2$ or $\mathbb{D}_{\mathcal{K}} \uplus \mathbb{D}_{\mathcal{K}}$, may be dealt with in essentially the same way.

The basic intuition is that the class $\mathcal{K}$ describes all possible "memory shapes" of an automaton.

A Fraïssé $\mathcal{K}$-*automaton* has a finite set Q of states. Each state $q \in Q$ comes with a single-orbit representation $(\mathfrak{A}_q, S_q)$. The set of configurations in state q is the nominal set $[\![\mathfrak{A}_q, S_q]\!]$. Elements of $\mathfrak{A}_q$ are called *registers* of state q, and the group S_q is the *register symmetry*. The set of configurations is:

$$X = \coprod_{q \in Q} [\![\mathfrak{A}_q, S_q]\!]. \tag{4}$$

At the risk of repeating material from the previous section, we unravel the above definition. A configuration consists of a state $q \in Q$, together with a valuation $\mathfrak{A}_q \to \mathfrak{U}_{\mathcal{K}}$ that maps registers to data values, and preserves and reflects the structure of $\mathfrak{A}_q$, with the proviso that valuations are considered equal if they differ only by a register symmetry.

The automaton has a set of accepting states, and an initial state. The structure of registers $\mathfrak{A}_q$ in the initial state must be empty. A further ingredient of the Fraïssé automaton is the *symbolic transition function* $s = \{s_q\}_{q \in Q}$ that is used to represent the equivariant transition function

$$\delta_s : X \times \mathbb{D}_{\mathcal{K}} \to X. \tag{5}$$

When defining the symbolic transition function, we use a notion of *annotation*. Annotations enumerate orbits of the product $X \times \mathbb{D}_{\mathcal{K}}$. Intuitively speaking, annotations describe the ways how the newly read input data value may compare to the data values in the registers. An *annotation* of $\mathfrak{A} \in \mathcal{K}$ is one of two kinds of structures: either a conservative extension $\mathfrak{A}^* \in \mathcal{K}$ of $\mathfrak{A}$ by one element, denoted $*$; or the structure $\mathfrak{A}$ itself with additionally one distinguished element, that we denote by $*$ as well. In the latter case, we identify two annotations if they are related by an automorphism from S_q. There are finitely many annotations as the vocabulary is finite.

The domain of s_q contains all annotations of $\mathfrak{A}_q$. For any annotation $\mathfrak{A}^*$, the function $s_q(\mathfrak{A}^*)$ is an embedding

$$s_q(\mathfrak{A}^*) : \mathfrak{A}_p \to \mathfrak{A}^*, \quad \text{for some state } p, \tag{6}$$

that is consistent with symmetries, cf. (3). Summing up:

Definition 13 A Fraïssé $\mathcal{K}$-automaton consists of:

- a finite set of states Q;
- for each state $q \in Q$, a representation $(\mathfrak{A}_q, S_q)$;
- an initial state $q_I \in Q$ with $\mathfrak{A}_{q_I}$ the empty structure;
- a set of accepting states $F \subseteq Q$;
- a symbolic transition function $s = \{s_q\}_{q \in Q}$ as above.

These ingredients induce naturally a $G_{\mathcal{K}}$-automaton. The transition function (5) is defined in the following way. Suppose that the state in the current configuration is $q \in Q$ and the valuation is represented, up to register symmetry, by $\eta : \mathfrak{A} \to \mathbb{D}_{\mathcal{K}}$. The automaton reads an input letter $d \in \mathbb{D}_{\mathcal{K}}$. Let η^* extend η by mapping $*$ to d, thus $\eta^* : \mathfrak{A}^* \to \mathbb{D}_{\mathcal{K}}$ is an embedding, for some $\mathfrak{A}^* \in \mathcal{K}^*_q$. Apply s_q to $\mathfrak{A}^*$, yielding a function (6). The new state is p, and the p-valuation is obtained by composing $s_q(\mathfrak{A}^*)$ with the extended valuation η^*, that is $s_q(\mathfrak{A}^*)\, \eta^* : \mathfrak{A}_p \to \mathbb{D}_{\mathcal{K}}$. The new valuation is an embedding, as a composition of embeddings, and its equivalence class depends only on the equivalence class of η, due to consistency with symmetries.

Theorem 14 *Suppose that the Fraïssé class $\mathcal{K}$ is well-behaved. Every reachable orbit-finite $G_{\mathcal{K}}$-automaton over input alphabet $\mathbb{D}_{\mathcal{K}}$ is isomorphic to a Fraïssé $\mathcal{K}$-automaton.*

A comment is in order here. The representation results from Section V apply to nominal $G_{\mathcal{K}}$-sets only, while the notion of $G_{\mathcal{K}}$-automaton lives in $G_{\mathcal{K}}$-sets and does not require finite supports. However, Theorem 6.2 still holds. Indeed, whenever the alphabet has finite supports, any reachable automaton also has finite supports, which in turn follows from the general fact that equivariant functions preserve supports; and from the empty support of the initial configuration.

By Theorems 6.2 and 3.3 one directly obtains:

Corollary 15 When $\mathcal{K}$ is well-behaved, the following conditions are equivalent for a $G_{\mathcal{K}}$-language $L \subseteq \mathbb{D}^*$:

(1) L is recognized by an orbit-finite $G_{\mathcal{K}}$-automaton
(2) the syntactic quotient $\mathbb{D}^*/\!\equiv_L$ is orbit-finite
(3) the syntactic automaton of L is isomorphic to a Fraïssé $\mathcal{K}$-automaton.

We now inspect in some more detail the three example classes $\mathcal{K}$: finite sets, total orders and partial orders.

Equality signature. The model of automaton resembles the finite memory automata of Francez and Kaminski. The differences are: the number of registers varies from state to state (thus no need for undefined register values), and symmetries are imposed on registers. These are inessential as far as expressive power is concerned:

Fact 16 When $\mathcal{K}$ is all finite sets, Fraïssé automata (and hence also $G_{\mathcal{K}}$-automata, by Thm 6.2) are expressively equivalent to deterministic finite memory automata of [12], [8], over singleton alphabet.

It should be noted that a similar, albeit weaker, connection to finite memory automata has been made within the framework of history-dependent automata in [7].

Total order signature. The model of automaton has a totally ordered set of registers in each state, and valuations are monotonic. The automata are capable to compare data values with respect to data ordering. One easily verifies:

Fact 17 When $\mathcal{K}$ is all finite total orders, Fraïssé automata (and hence also $G_{\mathcal{K}}$-automata, by Thm 6.2) are expressively equivalent to deterministic finite memory automata of [2], [10] over totally ordered data and singleton alphabet.

Partial order signature. In this case, the model of automaton has a partially ordered set of registers in each state, and the valuations are partial-order embeddings. This model generalizes both previous ones. To our best knowledge, this kind of automaton is new.

VII. Decidable theories

In this section we use the representation from Section V to sketch a logical framework for decidability results for problems that involve orbit-finite nominal sets. Problems covered by our framework include testing a nondeterministic automaton for emptiness, or testing if a deterministic automaton is minimal.

Fix for the rest of this section a nominal signature that admits representations of sets, functions and Cartesian products, as described in Section V. This includes the equality, total order and partial order nominal signatures. We shall prove that the first-order theory is decidable for every relational structure whose carrier is orbit-finite and nominal, and where every relation is equivariant. This result may be used to infer decidability of certain problems concerning automata.

Nominal relational structures. Consider a vocabulary Σ, i.e. a set of predicate names for relations. A *nominal relational structure* $\mathfrak{A}$ over a vocabulary Σ is a relational structure over Σ where the carrier is a nominal set, and every predicate is an equivariant relation. The structure is called *orbit-finite* if its carrier is. We use the representation of Section V to represent relational structures.

The vocabulary Σ should not be confused with vocabularies used in the definition of Fraïssé classes and nominal sets in Section V. Also, relational structures used in the representation results of Section V are not the same as nominal relational structures we wish to consider here. Rather, the former are used to represent the carriers of the latter.

For example, automata can be interpreted as relational structures. Consider an automaton $\mathcal{A}$, where both the configurations and input alphabet are orbit-finite and nominal. We do not assume that the automaton is deterministic, but its transition relation must be equivariant. The *automaton structure* of $\mathcal{A}$ is defined as follows. Its carrier is the disjoint union of configurations and input alphabet (essentially, we are simulating a two-sorted relational structure in a one-sorted one). The structure has four unary relations that identify: configurations, initial configurations, final configurations, letters of the input alphabet; and a ternary relation that identifies the transition relation of the automaton.

First-order theory. The *first-order theory* of a relational structure is the set of sentences of first-order logic that are true in it. Using first-order logic one can express some rudimentary property of automata. For instance, the transition relation of an automaton is deterministic if the theory of its automaton structure contains the sentence

$$\forall x \forall y \forall z_1 \forall z_2 \qquad \delta(x,y,z_1) \wedge \delta(x,y,z_2) \Rightarrow z_1 = z_2,$$

where δ stands for the transition relation.

As it turns out, first-order logic is decidable for every orbit-finite nominal relational structure with equivariant relations. In particular, this holds for automaton structures.

Theorem 18 *The first-order theory of every nominal orbit-finite relational structure is decidable.*

In the full version of this paper, we also prove that a variant of second-order theory is decidable, where quantifiers may range over equivariant relations only.

VIII. Future work

We would like to see in more detail what happens to various computation models in the world of nominal sets. Are context-free grammars equivalent to pushdown automata? Are regular expressions equivalent to nondeterministic automata? Is the P=NP question the same as in the classical case? What is the complexity zoo?

In another direction, we plan to apply our approach to timed automata. There seem to be at least two approaches. One is to apply the Fraïssé construction to some smartly chosen class of finite structures. Another is to consider a nominal signature where the data values are rational numbers, and the bijections preserve order and the binary predicate $x - y = 1$. This signature has very few of the good properties surveyed in this paper.

References

[1] J. Adamek and V. Trnkova. *Automata and Algebras in Categories*. Kluwer Academic Publishers, 1990.

[2] M. Benedikt, C. Ley, and G. Puppis. What you must remember when processing data words. In *AMW*, 2010.

[3] H. Björklund and T. Schwentick. On notions of regularity for data languages. *TCS*, 411(4-5):702–715, 2010.

[4] M. Bojańczyk. Data monoids. In *STACS*, 2011.

[5] M. Bojańczyk, A. Muscholl, T. Schwentick, L. Segoufin, and C. David. Two-variable logic on words with data. In *LICS*, pages 7–16, 2006.

[6] V. Ciancia. *Accessible functors and final coalgebras for named sets*. PhD thesis, University of Pisa, 2008.

[7] V. Ciancia and E. Tuosto. A novel class of automata for languages on infinite alphabets. Technical Report CS-09-003, University of Leicester, 2009.

[8] S. Demri and R. Lazic. LTL with the freeze quantifier and register automata. In *LICS*, pages 17–26, 2006.

[9] G. L. Ferrari, U. Montanari, and M. Pistore. Minimizing transition systems for name passing calculi: A co-algebraic formulation. In *FoSSaCS*, volume 2303 of *LNCS*, pages 129–158, 2002.

[10] D. Figueira, P. Hofman, and S. Lasota. Relating timed and register automata. In *Proc. EXPRESS'10*, volume 41 of *EPTCS*, pages 61–75, 2010.

[11] R. Fraïssé. *Theory of relations*. North-Holland, 1953.

[12] N. Francez and M. Kaminski. Finite-memory automata. *TCS*, 134(2):329–363, 1994.

[13] N. Francez and M. Kaminski. An algebraic characterization of deterministic regular languages over infinite alphabets. *TCS*, 306(1-3):155–175, 2003.

[14] M. Gabbay and A. M. Pitts. A new approach to abstract syntax with variable binding. *Formal Asp. Comput.*, 13(3-5):341–363, 2002.

[15] F. Gadducci, M. Miculan, and U. Montanari. About permutation algebras, (pre)sheaves and named sets. *Higher-Order and Symbolic Computation*, 19(2-3):283–304, 2006.

[16] J. Hubička and J. Nešetřil. Universal partial order represented by means of oriented trees and other simple graphs. *Eur. J. Comb.*, 26:765–778, 2005.

[17] U. Montanari and M. Pistore. History-dependent automata: An introduction. In *SFM*, pages 1–28, 2005.

[18] F. Neven, T. Schwentick, and V. Vianu. Towards regular languages over infinite alphabets. In *MFCS*, pages 560–572, 2001.

[19] M. Pistore. *History Dependent Automata*. PhD thesis, University of Pisa, 1999.

[20] L. Segoufin. Automata and logics for words and trees over an infinite alphabet. In *CSL*, pages 41–57, 2006.

[21] S. Staton. *Name-passing process calculi: operational models and structural operational semantics*. PhD thesis, University of Cambridge, 2007.