First-order logic with reachability for infinite-state systems

Emanuele D'Osualdo Roland Meyer
University of Kaiserslautern
dosualdo,meyer@cs.uni-kl.de

Georg Zetzsche[*]
LSV, CNRS & ENS Cachan, Université Paris-Saclay
zetzsche@lsv.fr

Abstract

First-order logic with the reachability predicate (FO(R)) is an important means of specification in system analysis. Its decidability status is known for some individual types of infinite-state systems such as pushdown (decidable) and vector addition systems (undecidable).

This work aims at a general understanding of which types of systems admit decidability. As a unifying model, we employ valence systems over graph monoids, which feature a finite-state control and are parameterized by a monoid to represent their storage mechanism. As special cases, this includes pushdown systems, various types of counter systems (such as vector addition systems) and combinations thereof. Our main result is a complete characterization of those graph monoids where FO(R) is decidable for the resulting transition systems.

Keywords first-order logic, reachability, transition systems, decidability, valence automata, graph monoids, automatic structures

1. Introduction

In the context of model checking, proving correctness of a system amounts to proving that a model of the system's behaviour, often in the form of an automaton, complies to a specification, typically given as a formula in a logic over the configuration graph of the automaton model. To obtain an automated correctness proof, care needs to be taken in choosing the expressivity of the automaton model and of the logics.

First-order logic with reachability (FO(R)) can uniformly specify a wide range of correctness properties. Safety specifications can be expressed with a simple existential quantification: Can some error configuration be reached? Violations of liveness specifications can be found by checking the existence of a loop that avoids desirable behaviour, that is, by solving a recurrent reachability problem, also expressible in FO(R). In general, quantifier alternation provides the means to specify how the system should react to an environment: No matter what the choices of the environment are ($\forall$), the system should be able to react ($\exists$) in a way that guarantees some objective is met. Moreover, FO(R) can provide a useful middle ground between first-order logics without reachability and monadic second-order logics: The latter increases expressivity significantly but often at the cost of being decidable on more restricted models.

Despite its importance in system analysis, the decidability status of FO(R) for infinite state systems is only known for a handful of prominent models. Using Caucal's interpretation technique, positive results have been shown for the (higher-order) pushdown automata family (Caucal 1996; Carayol and Wöhrle 2003). In the parallel setting, there are positive results for PA (Lugiez and Schnoebelen 2000) and vector addition systems with states (VASS) of dimension 2 (Leroux and Sutre 2004). For general VASS (equivalently, Petri Nets), close to no flavour of first-order logic (even without reachability) is decidable (Darondeau et al. 2011). However, to the best of our knowledge, there is no systematic account of decidability of FO(R) for general classes of storage mechanisms. The question we address in this paper is:

Which features of the storage mechanism of the automaton model determine the decidability status of FO(R)*?*

In order to investigate this question, we need a unifying framework that encompasses and generalizes all the aforementioned infinite state models. We thus set out to study FO(R) on the configuration graphs of *valence systems*: Finite state automata with auxiliary storage, where the storage mechanism is specified via a monoid. In this model, a configuration of a valence system over a monoid M consists of the couple (q, m) where q is a control state and m is the storage content, an element of M. A transition (q, m, q') has the effect of multiplying m to the current storage content. In order to model actions that are not allowed at the current storage content (such as popping the wrong stack symbol), we restrict the configurations to right-invertible monoid elements. As an example, a stack can be modelled by strings of non-barred (a) and barred ($\bar{a}$) versions of each stack symbol (representing a push and a pop respectively) with the rule $a\bar{a} = \mathbf{1}$. Then, the right-invertible elements are then the sequences over non-barred symbols. A counter over $\mathbb{N}$ with increment and decrement (which we call a *partially blind counter*) can be similarly modelled by having only one stack symbol. Analogously, a *blind counter* is a counter over $\mathbb{Z}$, represented by additionally having $a\bar{a} = \bar{a}a$.

Most importantly, new storage mechanisms can be built by composing monoids using two kinds of products. The direct product $M \times N$ of two monoids M and N allows the automaton to use the two storage mechanisms independently. This permits, for example, building counter systems of any finite dimension. The free product $M * N$ amounts, roughly speaking, to forming stacks whose entries alternate between elements of M and elements of N. A valence system over $M * N$ can push elements of N or M into the stack, manipulate the top entry using the storage mechanism offered by N and M and pop the top of the stack when empty.

To systematize our study, we consider the so-called graph monoids— monoids that are defined by graph presentations. Formally, a graph Γ induces a quotient $\mathbb{M}\Gamma$ over a free monoid: Each node represents a couple of dual letters a and $\bar{a}$, which always can-

[*] The third author is supported by a fellowship within the Postdoc-Program of the German Academic Exchange Service (DAAD).

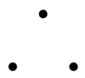
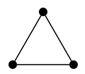
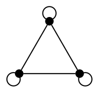
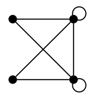
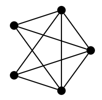
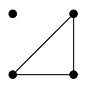
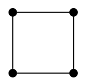
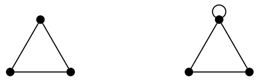
Graph Γ	Monoid $\mathbb{M}\Gamma$	Storage mechanism
	$\mathbb{B}^{(3)}$	Pushdown (with three symbols)
	$\mathbb{B}^3$	Three partially blind counters
	$\mathbb{Z}^3$	Three blind counters
	$\mathbb{B}^{(2)} \times \mathbb{Z}^2$	Pushdown (with two symbols) and two blind counters
	$\mathbb{B}^{(2)} \times \mathbb{B}^3$	Pushdown and three partially blind counters
	$\mathbb{B} * \mathbb{B}^3$	Stack of three partially blind counters
	$\mathbb{B}^{(2)} \times \mathbb{B}^{(2)}$	Two pushdowns (with two symbols each)

Table 1. Examples of storage mechanisms

cel out (i.e. $a\bar{a} \equiv \varepsilon$) — in other words, nodes represent copies of the bicyclic monoid $\mathbb{B}$. Edges represent a commutativity relation used to quotient the monoid. If two nodes u and v are adjacent, the letters they represent commute with each other, i.e. $x_u x_v \equiv x_v x_u$ for $x_w \in \{a_w, \bar{a_w}\}$, $w \in \{u, v\}$. if a node has a self loop, the two dual letters it represents commute, i.e. $a\bar{a} \equiv \bar{a}a$. A partially blind counter is thus represented by a graph with one node and no edges, while a blind counter by one with a node with a self loop. Table 1 shows several examples of storage mechanisms that can be modelled as graph monoids.

Contributions Our main result is a full characterization of the storage mechanisms $\mathbb{M}\Gamma$ that admit decidability of $\mathsf{FO}(\mathsf{R})$, based on a simple graph-theoretic property of Γ. We call a graph a $\mathbb{B}^2$-*triangle* if it is one of the following graphs:



A graph is said to be $\mathbb{B}^2$-*triangle-free* if it does not contain a $\mathbb{B}^2$-triangle as an induced subgraph. In Theorem 2.1 we prove that requiring Γ to be a disjoint union of cliques, where each clique is $\mathbb{B}^2$-triangle-free, is a *sufficient and necessary* condition for decidability of $\mathsf{FO}(\mathsf{R})$.

To establish our decidability results, we lift the idea of automaticity from transition systems to monoids. To be more precise, we define a notion of automatic rational multiplication — multiplication in the monoid being representable by a synchronous relation on a regular encoding. We then characterise the shape of storage mechanisms satisfying our condition as the closure under free products of the monoid class containing $\mathbb{B} \times \mathbb{Z}^k$ for all $k \in \mathbb{N}$ and $\mathbb{B} \times \mathbb{B}$. To show that, in the base case, $\mathbb{B} \times \mathbb{Z}^k$ and $\mathbb{B} \times \mathbb{B}$ have automatic rational multiplication, we establish Presburger-definability of the reachability relation. In the induction step, we give a direct construction for deriving automatic rational multiplication in $M * N$, assuming M, N have automatic rational multiplication.

For the undecidability results, we first show that the decomposition into cliques is needed. If Γ is not a disjoint union of cliques, then $\mathbb{M}\Gamma$ contains $\{a, b\}^* \times \{c\}^*$ as a submonoid of right-invertible elements, which, as we show, offers enough structure to make the Σ_1 fragment of $\mathsf{FO}(\mathsf{R})$ undecidable. On the other hand, if Γ decomposes into cliques but is not $\mathbb{B}^2$-triangle-free, then we show how to interpret the Σ_1 theory of arithmetic with addition and multiplication, $(\mathbb{N}, +, \cdot)$, in the Σ_2 fragment of $\mathsf{FO}(\mathsf{R})$ on valence systems. Interestingly, in both cases the undecidability results hold for a fixed valence systems.

One notable consequence of our result is that $\mathsf{FO}(\mathsf{R})$ is decidable for d-dimensional VASS if and only if $d \leq 2$.

Outline In Section 2 we recall the basic notions around valence systems and state our main theorem. Section 3 is devoted to proving the decidability result, while Section 4 proves the undecidability cases.

1.1 Related Work

A variety of positive results have been developed for the pushdown family. Notably, the Caucal hierarchy (Caucal 1996) admits a decidable MSO theory, via MSO-interpretations. The configuration graphs in the Caucal hierarchy are the ones generated by higher-order pushdown automata (Carayol and Wöhrle 2003), which therefore enjoy a decidable MSO theory. Walukiewicz (Walukiewicz 1996) obtained upper complexity bounds on μ-calculus model checking for pushdown systems. The interpretation idea has been generalized by Colcombet (2002) to regular ground tree rewriting. MSO is also known to be decidable on trees generated by higher-order recursion schemes (Ong 2006). Schulz (2010) showed, using an interpretation-based approach, that $\mathsf{FO}(\mathsf{R})$ is decidable over a class of grid-like structures. Beyond such iterated interpretation ideas, we are not aware of results that target general storage mechanisms.

In the concurrent setting, variants of first-order theories with reachability have been shown decidable by Lugiez and Schnoebelen (2000) for PA processes. For Petri nets, a thorough study of plain first-order logic without reachability showed that the logic is undecidable even for very weak predicates (Darondeau et al. 2011). Crucially, this holds only when one restricts the structures to the *reachable* configurations: when the unrestricted structure is considered, Petri net configuration graphs are automatic. For our setting, the restriction does not make a difference since the presence of the reachability predicate would allow us to restrict or relativise quantifiers within the logic.

Ideas akin to automaticity are applied in the area of regular model checking (Abdulla et al. 2004). There, the goal is to *approximate* the transitive closure of the transition relation, using widenings and acceleration based techniques on regular transducers. Instead of studying decidability, termination is enforced at the price of precision, obtaining constructions that are useful in practice.

Valence automata over graph monoids have recently been used to study other types of analysis. For example, there is a full characterization of those graph monoids that guarantee semilinear Parikh images (Buckheister and Zetzsche 2013) and partial characterizations of those with decidable reachability (Zetzsche 2015) and of those where ε-transitions can be eliminated (Zetzsche 2013).

2. Concepts and Main Result

A *finite automaton* is a tuple (X, Q, q_0, E, F) where X is a finite alphabet, Q is a finite set of states, $q_0 \in Q$ is the initial state, $E \subseteq Q \times X^* \times Q$ is a set of edges, and $F \subseteq Q$ is the set of final states. The language recognized by the automaton A is denoted $\mathcal{L}(A)$; for two states p and q of A, we denote by $\mathcal{L}_{pq}(A)$ the set of words that label paths from p to q in A.

Valence Systems

We represent storage mechanisms as monoids. Let M be a monoid; we denote multiplication in the monoid by juxtaposition, and the neutral element by $\mathbf{1}$. The elements of M are actions that can be applied to the storage content and the elements of

$$\mathcal{R}_1(M) := \{x \in M \mid \exists y \in M \colon xy = \mathbf{1}\}$$

correspond to valid storage contents. The elements of $\mathcal{R}_1(M)$ are said to be *right-invertible*. A *valence system over M* is a tuple $S = (Q, M, E)$, in which (i) Q is a finite set of *states*, (ii) E is a finite subset of $Q \times M \times Q$, called the set of *edges*. A *configuration of S* is a pair $(q, r) \in Q \times \mathcal{R}_1(M)$. For $p, q \in Q$ and $r, s \in \mathcal{R}_1(M)$, we write $(p, r) \to_S (q, s)$ if there is an edge $(p, m, q) \in E$ such that $s = rm$. By $\to_S^*$, we denote the reflexive transitive closure of $\to_S$.

Graphs

A *graph* is a pair $\Gamma = (V, E)$ where V is a finite set and E is a subset of $\{S \subseteq V \mid 1 \le |S| \le 2\}$. The elements of V are called *vertices* and those of E are called *edges*. Vertices $v, w \in V$ are *adjacent* if $\{v, w\} \in E$. If $\{v\} \in E$ for some $v \in V$, then v is called a *looped* vertex, otherwise it is *unlooped*. A *subgraph* of Γ is a graph (V', E') with $V' \subseteq V$ and $E' \subseteq E$. Such a subgraph is called *induced (by V')* if $E' = \{S \in E \mid S \subseteq V'\}$, i.e. E' contains all edges from E incident to vertices in V'. By $\Gamma \setminus \{v\}$, for $v \in V$, we denote the subgraph of Γ induced by $V \setminus \{v\}$. Moreover, a *clique* is a graph in which any two distinct vertices are adjacent. Finally, Γ^- denotes the graph obtained from Γ by deleting all self loops: We have $\Gamma^- := (V, E^-)$, where $E^- := \{S \in E \mid |S| = 2\}$. Finally, a graph is *transitive* if for $x, y, z \in V$, pairwise distinct, we have that $\{x, y\}, \{y, z\} \in E$ implies $\{x, z\} \in E$.

Graph monoids

Let A be a (not necessarily finite) set of symbols and R be a subset of $A^* \times A^*$. The pair (A, R) is called a *(monoid) presentation*. The smallest congruence of A^* containing R is denoted by $\equiv_R$ and we will write $[w]_R$ for the congruence class of $w \in A^*$. The *monoid presented by* (A, R) is defined as $A^*/{\equiv_R}$. Note that since we did not impose a finiteness restriction on A, up to isomorphism, every monoid has a presentation. Furthermore, for monoids M_1, M_2 we can find presentations (A_1, R_1) and (A_2, R_2) such that $A_1 \cap A_2 = \emptyset$. We define the *free product* $M_1 * M_2$ to be presented by $(A_1 \cup A_2, R_1 \cup R_2)$. Note that $M_1 * M_2$ is well-defined up to isomorphism. In analogy to the n-fold direct product, we write $M^{(n)}$ for the n-fold free product of M.

To each graph $\Gamma = (V, E)$, we associate the alphabet $X_\Gamma := \{a_v, \bar{a}_v \mid v \in V\}$. Moreover, let $\equiv_\Gamma$ be the smallest congruence with

$$a_v \bar{a}_v \equiv_\Gamma \varepsilon \qquad \text{for each } v \in V\text{, and} \tag{1}$$

$$xy \equiv_\Gamma yx \qquad \text{for each } x \in \{a_v, \bar{a}_v\},\ y \in \{a_w, \bar{a}_w\} \text{ with } \{v, w\} \in E. \tag{2}$$

In particular, we have $a_v \bar{a}_v \equiv_\Gamma \bar{a}_v a_v$ whenever $\{v\} \in E$. To each graph Γ, we associate the monoid

$$\mathbb{M}\Gamma \;=\; X_\Gamma^*/{\equiv_\Gamma}.$$

The monoids of the form $\mathbb{M}\Gamma$ are called *graph monoids*.

Let us briefly discuss how to realize storage mechanisms by graph monoids. First, suppose Γ_0 and Γ_1 are disjoint graphs. If Γ is the union of Γ_0 and Γ_1, then $\mathbb{M}\Gamma \cong \mathbb{M}\Gamma_0 * \mathbb{M}\Gamma_1$ by definition. Moreover, if Γ is obtained from Γ_0 and Γ_1 by drawing an edge between each vertex of Γ_0 and each vertex of Γ_1, then $\mathbb{M}\Gamma \cong \mathbb{M}\Gamma_0 \times \mathbb{M}\Gamma_1$.

If Γ consists of one vertex v and has no edges, the only commutativity rule is $a_v \bar{a}_v \equiv_\Gamma \varepsilon$. In this case, $\mathbb{M}\Gamma$ is also denoted as $\mathbb{B}$ and we will refer to it as the *bicyclic monoid*. The generators a_v and $\bar{a}_v$ are then also written a and $\bar{a}$, respectively. Observe that $\mathcal{R}_1(\mathbb{B}) = \{a\}^*$ and thus $\mathcal{R}_1(\mathbb{B}) \cong \mathbb{N}$. Thus, it is not hard to see that $\mathbb{B}$ corresponds to a *partially blind counter*, i.e. one that attains only non-negative values. Moreover, if Γ consists of one looped vertex, then $\mathbb{M}\Gamma$ (and also $\mathcal{R}_1(\mathbb{M}\Gamma)$) is isomorphic to $\mathbb{Z}$ and thus realizes a *blind counter*, which can go below zero.

If one storage mechanism is realized by a monoid M, then the monoid $\mathbb{B} * M$ corresponds to the mechanism that *builds stacks*: A configuration of this new mechanism consists of a sequence $c_0 a c_1 \cdots a c_n$, where $c_0, \ldots, c_n$ are configurations of the mechanism realized by M. We interpret this as a stack with the entries $c_0, \ldots, c_n$. One can open a new stack entry on top (by multiplying $a \in \mathbb{B}$), remove the topmost entry if empty (by multiplying $\bar{a} \in \mathbb{B}$) and operate on the topmost entry using the old mechanism (by multiplying elements from M). In particular, $\mathbb{B} * \mathbb{B}$ describes a pushdown stack with two stack symbols. Observe that $\mathcal{R}_1(\mathbb{B} * \mathbb{B})$ is isomorphic to the monoid of words over two symbols, i.e. stack words. Table 1 shows few more examples. See (Zetzsche 2013) for details.

Logic

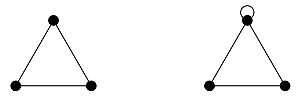
Let S be a valence system over the monoid M. Then the *first-order logic with reachability for S ($\mathsf{FO(R)}$)* consists of the first-order formulas over the following signature:

- The *constant symbols* are the configurations of S.
- $\mathsf{state}_q(\cdot)$: A unary *state predicate* for each $q \in Q$.
- $\mathsf{step}(\cdot,\cdot)$: The binary *step predicate* for one-step reachability.
- $\mathsf{reach}(\cdot,\cdot)$: The binary *reachability predicate* .

Given a valence system S, its *reachability structure* is the relational structure with domain $Q \times \mathcal{R}_1(M)$ over the above signature defined in the obvious way. For a first-order formula φ, we write $S \models \varphi$ when φ holds on the reachability structure of S. The *first-order theory with reachability over S* is then the set of formulas $\{\varphi \in \mathsf{FO} \mid \varphi \text{ closed}, S \models \varphi\}$.

2.1 Main Result

We call Γ a $\mathbb{B}^2$*-triangle* if it is one of the following graphs:



The graph Γ is said to be $\mathbb{B}^2$*-triangle-free* if it does not contain a $\mathbb{B}^2$-triangle as an induced subgraph.

Theorem 2.1. *Let Γ be a graph. The first-order theory with reachability for valence systems over $\mathbb{M}\Gamma$ is decidable if and only if Γ is a disjoint union of $\mathbb{B}^2$-triangle-free cliques.*

Before we move to the proof, let us elaborate on the concrete shape of the graphs satisfying the condition of Theorem 2.1. If Δ is a $\mathbb{B}^2$-triangle-free clique, then Δ either (i) consists of at most two unlooped vertices or (ii) contains at most one unlooped vertex and a number of looped vertices. Since all these vertices are adjacent, this means we either have $\mathbb{M}\Delta \cong \mathbb{B}^2$ or $\mathbb{M}\Delta \cong \mathbb{B}^r \times \mathbb{Z}^s$ for $r \in \{0, 1\}$ and $s \in \mathbb{N}$. If Γ is a disjoint union of $\mathbb{B}^2$-triangle-free cliques $\Delta_1, \ldots, \Delta_n$, then $\mathbb{M}\Gamma$ is the free product of the $\mathbb{M}\Delta_i$.

Operationally, the storage mechanism implemented by such an $\mathbb{M}\Gamma$ is composed of a stack, each entry of which is either (i) a pair of partially blind counters or (ii) a partially blind counter with a number of blind counters.

3. Decidability

In this section, we show that $\mathsf{FO}(\mathsf{R})$ is decidable for valence systems over $\mathbb{M}\Gamma$ whenever Γ is a disjoint union of $\mathbb{B}^2$-triangle-free cliques. More specifically, we prove that the transition systems–with the reachability relation–generated by such systems over $\mathbb{M}\Gamma$ are (effectively) automatic. We deduce this property from a slightly more abstract condition on the monoid, which we call *automatic rational multiplication*.

As we argued in the previous section, if Γ is a disjoint union of $\mathbb{B}^2$-triangle-free cliques, then $\mathbb{M}\Gamma$ is isomorphic to the free product of a finite number of monoids, each of which is isomorphic either to $\mathbb{B}^2$ or to $\mathbb{B}^r \times \mathbb{Z}^s$, for $r \in \{0, 1\}$ and $s \in \mathbb{N}$. Therefore, we proceed as follows:

(i) We show that $\mathbb{B} \times \mathbb{Z}^n$ has automatic rational multiplication for each $n \geq 0$. Here, we use a direct construction showing that the reachability relation for two fixed states is Presburger-definable and can thus conclude automaticity.

(ii) $\mathbb{B} \times \mathbb{B}$ has automatic rational multiplication. Note that $\mathcal{R}_1(\mathbb{B} \times \mathbb{B}) \cong \mathbb{N} \times \mathbb{N}$ and that valence systems over $\mathbb{B} \times \mathbb{B}$ are essentially two-dimensional vector addition systems with states. We can therefore apply a result of Leroux and Sutre (2004), stating that the binary reachability relation of such systems is effectively semilinear. Specifically, given a rational subset R of $\mathbb{B} \times \mathbb{B}$, one can easily construct a 2-dim. VASS whose binary reachability relation for two states is precisely all those $((x, y), (x', y'))$ for which there is an $r \in R$ with $(x, y)r = (x', y')$. Since this semilinearity implies Presburger-definability, we directly obtain an encoding with automatic rational multiplication.

(iii) We obtain a general transfer result, stating that if two monoids M_0 and M_1 each have automatic rational multiplication, then so does their free product $M_0 * M_1$. Here, we use a generalization of the saturation method for pushdown automata (Benois 1969; Bouajjani et al. 1997) to obtain automaticity.

Hence, for the decidability, remains to prove (i) and (iii), which we do in Sections 3.2 and 3.3, respectively. We begin with the required concepts.

3.1 Automaticity

Let X be an alphabet and $\diamond \notin X$. We define the alphabet $X(2, \diamond)$ as $(X \cup \{\diamond\})^2 \setminus \{(\diamond, \diamond)\}$ and the *convolution* $u \otimes v \in X(2, \diamond)^*$ of words $u, v \in X^*$ inductively as follows. We have

$$au \otimes bv = (a, b)(u \otimes v), \qquad \varepsilon \otimes \varepsilon = \varepsilon$$
$$\varepsilon \otimes bv = (\diamond, b)(\varepsilon \otimes v) \qquad au \otimes \varepsilon = (a, \diamond)(u \otimes \varepsilon)$$

for $a, b \in X$ and $u, v \in X^*$. For a relation $R \subseteq X^* \times X^*$, we write $R^{\otimes} = \{u \otimes v \mid (u, v) \in R\}$. Such a relation is called *synchronous rational* if $R^{\otimes}$ is a regular language.

Let $(D, R_1, \ldots, R_n)$ be a relational structure with domain D and relations $R_1, \ldots, R_n$ of arities $r_1, \ldots, r_n \leq 2$ (for our purposes, it suffices to consider arities ≤ 2). This structure is called *automatic* if there is a regular language $L \subseteq X^*$, and a bijection $\theta\colon L \to D$ such that each of the relations

$$R_i^{\theta} = \{(x_1, \ldots, x_{r_i}) \in L^{r_i} \mid (\theta(x_1), \ldots, \theta(x_{r_i})) \in R_i\}$$

is synchronously rational (or regular, if $r_i = 1$). The first-order theory of each automatic structure is decidable (Khoussainov and Nerode 1995), even when extended by the infinity quantifier (Blumensath and Grädel 2000) and the modulo quantifier (Khoussainov et al. 2004).

We will show that in the positive (decidable) case of Theorem 2.1, the rechability structure of each such valence system is (effectively) automatic. Here, it will be convenient to prove a more abstract condition, termed 'automatic rational multiplication', which implies the desired automaticity. Let M be a monoid. We say that M is *finitely generated* if there is an alphabet X and a surjective morphism $\varphi\colon X^* \to M$. Note that if M is finitely generated, we can denote elements of M by words over X. From now on, we assume that M is finitely generated and fix X and φ. While the following definitions make reference to X and φ, it is easy to see that they do not depend on this choice.

A subset $R \subseteq M$ is called *rational* if there is a regular language $L \subseteq X^*$ such that $R = \varphi(L)$. We will represent such a rational subset of M by a finite automaton for L.

Definition 3.1. *Let M be finitely generated. An* encoding *for M is a bijection $\theta\colon L \to \mathcal{R}_1(M)$ where L is a regular language. We say that θ has* automatic rational multiplication *if for each rational subset $R \subseteq M$, the relation*

$$R^{\odot} := \{(u, v) \in L \times L \mid \exists r \in R\colon \theta(u)r = \theta(v)\}$$

is effectively synchronous rational, i.e. one can compute a finite automaton for $(R^{\odot})^{\otimes}$. In this case, we also say that M has automatic rational multiplication.

As announced, we have the following.

Theorem 3.2. *Suppose M has an encoding with automatic rational multiplication. Then each valence system over M has an effectively automatic reachability structure.*

The proof of Theorem 3.2 is not difficult. One has to extend the encoding with the states of a given valence system. One then observes that the configurations in state q reachable from (p, r) are precisely those of the form (q, rm), where m is drawn from a rational set. Hence, the reachability relation is synchronous rational. Moreover, encodings of step relation and configurations follows from the special case where the rational set is a finite set.

3.2 Base Case $\mathbb{B} \times \mathbb{Z}^k$

We show that $\mathbb{B} \times \mathbb{Z}^k$ has automatic rational multiplication. The key observation is that a rational set over this monoid behaves like a 1-counter automaton (1CM) over $\mathbb{N}$ that is decorated with further counters over $\mathbb{Z}$. Decorated means only the $\mathbb{N}$-counter influences the behavior of the automaton, the $\mathbb{Z}$-counters do not.

To prove automaticity, we show that the reachability relation in such decorated 1CM is Presburger definable. Automaticity now follows from automaticity of Presburger arithmetic. To establish Presburger definability of the reachability relation, we make use of semilinearity of the Parikh images of 1CM languages, combined with the equivalence of semilinear and Presburger-definable sets.

Proposition 3.3. *$\mathbb{B} \times \mathbb{Z}^k$ has automatic rational multiplication.*

The monoid $\mathbb{B} \times \mathbb{Z}^k$ is finitely generated by definition. Since we have an isomorphism $\mathcal{R}_1(\mathbb{B} \times \mathbb{Z}^k) \cong \mathbb{N} \times \mathbb{Z}^k$, we can identify the elements of the former set with pairs $(m, \vec{c}) \in \mathbb{N} \times \mathbb{Z}^k$. Consider a rational set $R \subseteq \mathbb{B} \times \mathbb{Z}^k$. We have to characterize the set of pairs $(m, \vec{c})$ and $(n, \vec{d})$ for which there is an element $r \in R$ so that $(m, \vec{c})r = (n, \vec{d})$. To be more precise, we first have to fix an encoding of $\mathbb{N} \times \mathbb{Z}^k$ as a regular language. Based on this encoding, characterizing the set means to compute a synchronously rational relation over the regular language that contains precisely the encoded pairs. Fortunately, we can show how to capture the set of pairs $((m, \vec{c}), (n, \vec{d}))$ at the monoid level, in a way that saves us from encoding issues, namely by proving Presburger definability. Presburger arithmetic is the first-order logic over $(\mathbb{Z}, \leq, +)$. The result is as follows.

Lemma 3.4. *Given a rational set $R \subseteq \mathbb{B} \times \mathbb{Z}^k$, one can construct a Presburger formula $\varphi_R(x_1, \vec{y_1}, x_2, \vec{y_2})$ (interpreted over $\mathbb{Z}$) so that for all $(m, \vec{c})$ and $(n, \vec{d})$ from $\mathbb{N} \times \mathbb{Z}^k$ we have*

$$\exists r \in R\colon (m, \vec{c})r = (n, \vec{d}) \quad \textit{iff} \quad \varphi_R(m, \vec{c}, n, \vec{d}) \textit{ holds.}$$

Before we turn to the construction, we explain why the result guarantees the encoding requirement. The structure $(\mathbb{Z}, +, \leq)$ has a first-order interpretation in $(\mathbb{N}, +, \leq)$, the more common formulation of Presburger arithmetic. (One idea is to provide two variables over the naturals for each integer). Automaticity of Presburger arithmetic over $\mathbb{N}$ goes back to (Büchi 1960). To be precise, there is a bijection between the naturals and the elements in a regular language. Moreover, there are automata for the encodings of solutions to Presburger-definable relations. Automaticity is preserved under first-order interpretations (Blumensath and Grädel 2000). A direct encoding of $(\mathbb{Z}, +, \leq)$ into automata (sign followed by most-significant bit) is given in (Boigelot and Wolper 1998). Finally, the set $\mathbb{N} \times \mathbb{Z}^k$ is first-order definable, meaning we may find an encoding of precisely the elements of $\mathcal{R}_1(\mathbb{B} \times \mathbb{Z}^k)$.

To compute the Presburger formula φ_R, we develop an understanding of the rational set $R \subseteq \mathbb{B} \times \mathbb{Z}^k$ as a 1CM. By definition, the rational set is represented by a finite automaton N over letters say a and $\bar{a}$ for $\mathbb{B}$ and b_i and $\bar{b_i}$ for each copy of $\mathbb{Z}$, $i = 1, \ldots, k$. We interpret $\mathbb{B}$ and $\mathbb{Z}$ as counters (partially blind and blind, respectively). With this point of view, a and $\bar{a}$ and similarly b_i and $\bar{b_i}$ are increment and decrement operations on the respective counters. An element $r \in R$ is thus represented by a sequence of increment and decrement operations on the counters that is accepted by N.

The formula φ_R is supposed to capture the pairs $(m, \vec{c})$ and $(n, \vec{d})$ from $\mathbb{N} \times \mathbb{Z}^k$ so that $(m, \vec{c})r = (n, \vec{d})$ for some $r \in R$. With the above discussion, finding such an element $r \in R$ means finding a sequence of increment and decrement operations in the automaton N that, when applied to the initial value $(m, \vec{c})$, yields $(n, \vec{d})$. The only requirement on this application is that the $\mathbb{N}$-counter is never decremented below zero. However, this is precisely the semantics of a 1CM over $\mathbb{N}$. We make this relationship between N and 1CM explicit, and show how it leads to the desired computability result.

Let the given finite automaton be $N = (X, Q, q_0, E, F)$ with alphabet $X = \{a, \bar{a}, b_1, \bar{b_1} \ldots, b_k, \bar{b_k}\}$ and edges $E \subseteq Q \times X \times Q$. A 1CM C is a finite automaton that has access to an $\mathbb{N}$-counter and where the edges carry increment, decrement, and noop-commands ($E \subseteq Q \times X \times \{+1, -1, 0\} \times Q$). The semantics is defined as expected. We write $(q_1, m_1) \xrightarrow{\sigma}_C (q_2, m_2)$ to mean that there is a sequence of transitions among the configurations of C that is labeled by $\sigma \in X^*$ and that takes (q_1, m_1) to (q_2, m_2). We use $\mathcal{L}(C, m) := \{\sigma \in X^* \mid \exists q_f \in F, i \in \mathbb{N}\colon (q_0, m) \xrightarrow{\sigma}_C (q_f, i)\}$ to denote the language of C from the initial counter value m. Automaton N induces the 1CM C_N defined as follows. Edges labeled by a are given command $+1$, because they correspond to increments on the one $\mathbb{N}$-counter. Edges labeled by $\bar{a}$ are given -1. The remaining edges are decorated with 0. Note that the labeling stays unchanged, and that we do not introduce commands for the $\mathbb{Z}$-counters. To state the relationship between N and C_N, we need the *Parikh images* of words $w \in X^*$. The Parikh image of w is the function $\psi(w)\colon X \to \mathbb{N}$ where $(\psi(w))(a)$ is the number of occurrences of letter a in word w.

Lemma 3.5. *There is $r \in R$ so that $(m, \vec{c})r = (n, \vec{d})$ if and only if there is $\sigma \in \mathcal{L}(C_N, m)$ so that*

$$y = x + (\psi(\sigma))(z) - (\psi(\sigma))(\bar{z})$$

for all $(x, y, z) \in \{(m, n, a), (c_1, d_1, b_1), \ldots, (c_k, d_k, b_k)\}$.

The lemma rephrases the existence of $r \in R$ as the existence of a C_N-computation. The main message is that the effect on the counters can be deduced from the Parikh image of the computation. This holds true even for the $\mathbb{N}$-counter. Nevertheless, we need the semantics of a 1CM (rather than all computations of the finite automaton N) to make sure the $\mathbb{N}$-counter stays non-negative also in all intermediary configurations.

Lemma 3.5 can be read as follows. For a fixed value m, it characterizes the suitable $\vec{c}$, $\vec{d}$, and n in terms of the Parikh images of the computations of C_N. To give a characterization for all m, we guess the initial value of the $\mathbb{N}$-counter. Let C'_N coincide with C_N except for a fresh initial state q_{new} and a fresh letter $\hat{a}$. The new state carries a loop $(q_{\mathsf{new}}, \hat{a}, +1, q_{\mathsf{new}})$ and has an ε-transition to the former initial state, $(q_{\mathsf{new}}, \varepsilon, 0, q_0)$.

Lemma 3.6. *There is $r \in R$ so that $(m, \vec{c})r = (n, \vec{d})$ if and only if there is $w \in \mathcal{L}(C'_N, 0)$ so that*

$$m = (\psi(w))(\hat{a})$$
$$n = (\psi(w))(\hat{a}) + (\psi(w))(a) - (\psi(w))(\bar{a})$$

and $d_i = c_i + (\psi(w))(b_i) - (\psi(w))(\bar{b_i})$ for all $i = 1, \ldots, k$.

Since C'_N is an ordinary 1CM, its language is context free. With Parikh's theorem (Parikh 1966), the Parikh image of this language is a semilinear set, and hence Presburger definable. The equations in the lemma explain how to turn a Presburger formula for $\psi(\mathcal{L}(C'_N, 0))$ into a formula for our set of pairs.

3.3 Induction Step

In this section, we show that the property of having automatic rational multiplication is passed on to free products. This means, we prove the following:

Theorem 3.7. *If the monoids M_0 and M_1 have automatic rational multiplication, then so does $M_0 * M_1$.*

Let us briefly sketch the proof steps. First, we devise a saturation procedure for rational sets (Lemma 3.9). This allows us to assume that in the automata over $M_0 * M_1$, every element can be accepted through a word in a normal form. This normal form is amenable to an analysis of how elements can cancel during multiplication (Lemma 3.10). Then, we use Lemma 3.10 to show that for each rational set, the set of encodings of its members and that of their inverses is effectively regular (Lemma 3.12). Finally, we use this and the possible cancelation cases from Lemma 3.10 again to construct a synchronous rational relation for each of these cases.

Let M_0 and M_1 be monoids with automatic rational multiplication. Then, M_0 and M_1 are finitely generated, so assume alphabets X_0 and X_1 with $X_0 \cap X_1 = \emptyset$ and a surjective morphism $\varphi_i\colon X_i^* \to M_i$ for each $i \in \{0, 1\}$. For the monoid $M = M_0 * M_1$, we pick the surjective morphism $\varphi\colon X^* \to M$ with $X = X_0 \cup X_1$ that extends φ_0 and φ_1. To simplify notation, we will write $[w]$ to mean $\varphi(w)$ for words $w \in X^*$. For $K \subseteq X^*$, we write $[K]$ for $\{[w] \mid w \in K\}$.

By our assumption, for each $i \in \{0, 1\}$, there is an encoding $\theta_i\colon L_i \to \mathcal{R}_1(M_i)$ with automatic rational multiplication. It is an easy exercise to show that we may assume $\varepsilon \in L_i$ and $\theta_i(\varepsilon) = \mathbf{1}$. Of course, we may also assume $Y_0 \cap Y_1 = \emptyset$. In our encoding for $M_0 * M_1$, we take $Y = Y_0 \cup Y_1$ as the alphabet and $L = (L_0 \cup L_1)^*$ as the regular language. In order to define $\theta\colon L \to \mathcal{R}_1(M_0 * M_1)$, we need some terminology.

Let $w \in X^*$ and $i \in \{0, 1\}$. A non-empty factor f of w is called an *i-block* (or *block of type i*) if $f \in X_i^*$ and f has no neighboring symbol in X_i. It is called a *block* if it is an i-block for some $i \in \{0, 1\}$. The unique decomposition $w = w_1 \cdots w_n$ where each w_j is a block of w is called w's *block decomposition*. Note that we also have $Y = Y_0 \cup Y_1$ and $Y_0 \cap Y_1 = \emptyset$, which allows us to apply the terms block and block decomposition analogously for words over Y. We call a word $w \in X^*$ *reducible* if for some

$i \in \{0,1\}$, it has a block f with $[f] = \mathbf{1}$. Otherwise, w is called *reduced*. The following follows easily from the definition of the free product.

Fact 3.8. *Suppose u and v are reduced and have block decompositions $u = u_1 \cdots u_n$ and $v = v_1 \cdots v_m$. Then $[u] = [v]$ if and only if $m = n$ and $[u_i] = [v_i]$ for every $i \in [1, n]$.*

We are now ready to define the map $\theta\colon L \to \mathcal{R}_1(M)$. It is well-known (and can be deduced from Lemma 3.10) that $\mathcal{R}_1(M_0 * M_1) = \mathcal{R}_1(M_0) * \mathcal{R}_1(M_1)$. For $w \in L_i^*$, $i \in \{0,1\}$, we define $\theta(w) = \theta_i(w)$. Recall that we have $L_0 \cap L_1 = \{\varepsilon\}$, but this causes no contradiction because the neutral elements of M_0 and M_1 are identified in $M_0 * M_1$. For the general case, suppose $w \in L$ with block decomposition $w = w_1 \cdots w_n$. Then we have $w_j \in L_0 \cup L_1$ for $j \in [1, n]$ and we set $\theta(w) := \theta(w_1) \cdots \theta(w_n)$. Observe that θ is surjective because θ_0 and θ_1 are. Moreover, it follows from Fact 3.8 that θ is injective.

Rational subsets of $M = M_0 * M_1$ are represented by finite automata over X. Of course, such an automaton can accept words that are not reduced. However, we will see that we can always construct an automaton A for R that is *saturated*, meaning that $[\mathcal{L}(A)] = R$ and whenever $m \in [\mathcal{L}_{pq}(A)]$ for states p, q, then there is a reduced word $w \in \mathcal{L}_{pq}(A)$ with $[w] = m$.

Lemma 3.9. *For each rational subset $R \subseteq M = M_0 * M_1$, we can compute a saturated automaton for R.*

Here, the idea is to introduce an ε-edge between p and q whenever $\mathbf{1} \in [\mathcal{L}_{pq}(A)]$. If we do this until we reach a fixpoint, we arrive at a saturated automaton.

Proof. We are given R as a finite automaton $A = (X, Q, q_0, E, F)$ such that $[\mathcal{L}(A)] = R$. We define, for states $p, q \in Q$, the finite automaton A_{pq} as $(X, Q, \{p\}, E, \{q\})$. In other words, we take A and designate p and q as initial and (only) final state, respectively.

First, observe that if N is a monoid with automatic rational multiplication, then given a rational subset $R \subseteq N$, it is decidable whether $\mathbf{1} \in R$: We have $\mathbf{1} \in R$ if and only if $(\varepsilon, \varepsilon) \in R^{\odot}$, which we can check because $(R^{\odot})^{\otimes}$ is effectively regular.

We apply a saturation procedure, in which we successively add edges to E. We begin with $E_0 = E$ and assume we have constructed E_i. Consider the automaton $A_i = (X, Q, q_0, E_i, F)$. We check whether there is a pair (p, q) of states such that there is no edge (p, ε, q) and we have $\mathbf{1} \in [\mathcal{L}_{pq}(A_i)]$. This is decidable by our observation since $[\mathcal{L}_{pq}(A_i)]$ is rational. If we find such a pair of states, we set $E_{i+1} = E_i \cup \{(p, \varepsilon, q)\}$. If there is no such pair, the procedure terminates.

We clearly have $E_0 \subseteq E_1 \subseteq \cdots$ and since we add no states to the automaton, the procedure has to terminate with some E_k. We claim that then, A_k has the desired property.

First of all, it is clear that in each step, we have $[\mathcal{L}(A_i)] = [\mathcal{L}(A_{i+1})]$. The inclusion "$\subseteq$" holds trivially and the other inclusion holds by the choice of p and q. In particular, we have $[\mathcal{L}(A_k)] = R$. Now let $p, q \in Q$ and $m \in [\mathcal{L}_{pq}(A_k)]$ and choose a word $w \in \mathcal{L}_{pq}(A_k)$ with $[w] = m$ such that w has a minimal number of blocks. Suppose w is not reduced, i.e. $w = ufv$ with a block f such that $[f] = \mathbf{1}$. By construction of A_k, this means $uv \in \mathcal{L}_{pq}(A_k)$. However, since $[uv] = [w]$ and uv has at least one block less than w, this contradicts the choice of w. Hence, w must be reduced. ☐

For our proof, we need to understand how multiplication works in $M_0 * M_1$ in terms of reduced words. We identify four cases, the simplest being the "non-merging" case. Let $u = u_1 \cdots u_m$ and $v = v_1 \cdots v_n$ be block decompositions of $u, v \in X^*$. We call u and v *non-merging* if u_m and v_1 are blocks of distinct types. Otherwise, we say that u and v are *merging*. Of course, when u and v are non-merging, then uv is again a reduced word and represents $[u][v]$. As above, since we have $Y_0 \cap Y_1 = \emptyset$, the notion of merging words also applies to encodings, i.e. members of L. Note that if x and y are non-merging and $x, y \in L$, then $xy \in L$ and $\theta(xy) = \theta(x)\theta(y)$. We now want to understand multiplication in the merging case. Let $w = w_1 \cdots w_k$ be the block decomposition of w. For a proof, see Appendix B.

Lemma 3.10. *Let $u, v, w \in X^*$ be reduced and let u and v be merging. Then we have $[uv] = [w]$ if and only if one of the following holds:*

(i) We have $m \le n$ and
- *(a) for every $j \in [1, m]$, we have $[u_{m-j+1}v_j] = \mathbf{1}$,*
- *(b) $k = n - m$, and*
- *(c) $[w_1 \cdots w_k] = [v_{m+1} \cdots v_n]$.*

(ii) We have $m > n$ and
- *(a) for every $j \in [1, n]$, we have $[u_{m-j+1}v_j] = \mathbf{1}$,*
- *(b) $k = m - n$, and*
- *(c) $[w_1 \cdots w_k] = [u_1 \cdots u_k]$.*

(iii) There is an $i \in [0, \min(m, n)]$ such that
- *(a) for every $j \in [1, i]$, we have $[u_{m-j+1}v_j] = \mathbf{1}$*
- *(b) $[u_{m-i}v_{i+1}] \neq \mathbf{1}$*
- *(c) $k = (m - i - 1) + 1 + (n - i - 1)$,*
- *(d) $[w_1 \cdots w_{m-i-1}] = [u_1 \cdots u_{m-i-1}]$,*
- *(e) $[w_{m-i}] = [u_{m-i}v_{i+1}]$,*
- *(f) $[w_{m-i+1} \cdots w_k] = [v_{i+2} \cdots v_n]$.*

For relations $S, T \subseteq Y^* \times Y^*$, we define

$$S \cdot T := \{(x_1x_2, y_1y_2) \mid (x_1, y_1) \in S,\ (x_2, y_2) \in T\}.$$

When constructing synchronous rational relations, we employ the following fact, which is well known and easy to see.

Fact 3.11. *Let $S \subseteq Y^* \times Y^*$ be synchronous rational and let $C \subseteq Y^*$ and $D \subseteq Y^*$ be regular. Then, the relation $S \cdot (C \times D)$ is synchronous rational as well. Moreover, if every pair $(x, y) \in S$ satisfies $|x| = |y|$ and $T \subseteq Y^* \times Y^*$ is synchronous rational, then $S \cdot T$ is also synchronous rational.*

We now show that for the encoding θ, the right-invertible members of a rational set as well left-inverses of their members have effectively regular encodings. Let us define this precisely. Suppose N is a monoid and we have an encoding $\eta\colon K \to \mathcal{R}_1(N)$. Then, for rational subsets $R \subseteq N$, we define

$$R^{\triangleright} := \{w \in K \mid \eta(w) \in R\},$$
$$R^{\triangleleft} := \{w \in K \mid \exists r \in R\colon \eta(w)r = \mathbf{1}\}.$$

If the encoding function has automatic rational multiplication, these sets are always regular. This is because the set $R^{\triangleright}$ is the projection of $R^{\odot} \cap (\{\eta^{-1}(1)\} \times X^*)$ onto the right component. Hence, $R^{\triangleright}$ is regular. Analogously, $R^{\triangleleft}$ is the projection of the relation $R^{\odot} \cap (X^* \times \{\eta^{-1}(1)\})$ onto the left component. We now lift this effective regularity to our encoding for $M_0 * M_1$. The idea is to (i) decompose each run of an automaton for R into blocks representing elements $\neq \mathbf{1}$ (which is possible thanks to Lemma 3.9) and (ii) replace each of them with the encoding (employing effective regularity for M_0 and M_1). In the case of $R^{\triangleleft}$, we have to reverse the sequence of blocks.

Lemma 3.12. *For each rational subset $R \subseteq M = M_0 * M_1$, the sets $R^{\triangleright}$ and $R^{\triangleleft}$ are effectively regular.*

Proof. Suppose that $R = [\mathcal{L}(A)]$ for the finite automaton $A = (X, Q, q_0, E, F)$. By Lemma 3.9, we may assume that every element of R is represented by a reduced word. Observe that it is decidable whether $\mathbf{1} \in R$: The only reduced word that can represent $\mathbf{1} \in M$ is the empty word, meaning we need to check whether

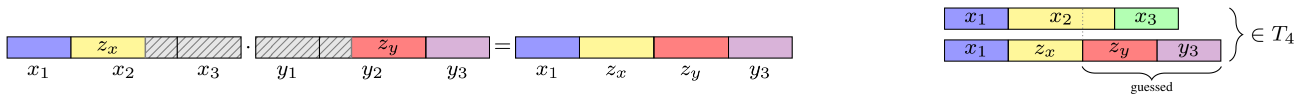


Figure 1. Anatomy of rational multiplication, merging case. The gray area reduces to $\mathbf{1}$.

$\varepsilon \in \mathcal{L}(A)$. Recall that $X = X_0 \cup X_1$. By $A|_i$, we denote the automaton obtained from A by deleting all edges labeled with X_{1-i}.

We begin with the set $R^{\triangleright}$. We will construct a regular language K and a regular substitution σ such that $\sigma(K) = R^{\triangleright}$, which implies effective regularity of $R^{\triangleright}$. The set K is a language over $Z = Q \times \{0,1\} \times Q$ and consists of all non-empty words

$$(p_0, i_1, p_1)(p_1, i_2, p_2) \cdots (p_{n-1}, i_n, p_n)$$

such that $p_0 = q_0$, $p_n \in F$, and for every $j \in [1, n-1]$, we have $i_{j+1} = 1 - i_j$. Moreover, we add ε to K if and only if $\mathbf{1} \in R$. We will use the rational set $R_{piq} \subseteq M_i$, defined by $R_{piq} \coloneqq [\mathcal{L}_{pq}(A|_i)]$. The substitution $\sigma\colon Z \to \mathbb{P}(Y^*)$ is defined as follows. For $(p,i,q) \in Z$, we set $\sigma(z) \coloneqq R^{\triangleright}_{piq} \setminus \{\varepsilon\}$, where $R^{\triangleright}_{piq}$ is effectively regular by our observation above. One can now show that $R^{\triangleright} = \sigma(K)$ (see Appendix C).

Let us now show regularity of $R^{\triangleleft}$. We use the language K from above, but in reverse, and instead of σ, we use the substitution $\tau\colon Z \to \mathbb{P}(X^*)$ where for $z = (p,i,q)$, we set $\tau(z) \coloneqq R^{\triangleleft}_{piq} \setminus \{\varepsilon\}$. Again, one can show that $R^{\triangleleft} = \tau(K^{\mathsf{rev}})$ (see Appendix C). □

We are now ready to show Theorem 3.7, i.e. regularity of $R^{\odot}$. Here, our strategy is to construct a synchronous rational relation for each of the four cases of a multiplication ("non-merging" and the three cases of Lemma 3.10). In order to show that the constructed relations are synchronous rational, we first introduce some auxiliary sets. Let $R = [\mathcal{L}(A)]$ be a rational subset of $M = M_0 * M_1$, where $A = (X, Q, q_0, E, F)$. We may assume that A is saturated (Lemma 3.9) and that A has only one final state, $F = \{f\}$. The first set we use consists of all encoding pairs that are equal and either empty or end in an ℓ-block:

$$E_\ell \coloneqq \{(u,u) \mid u \in \{\varepsilon\} \cup (L_0 \cup L_1)^* L_\ell\}. \tag{3}$$

E_ℓ is clearly synchronous rational. In order to describe the result of canceling elements of R with others, we employ the set $R_{p,q,\ell} = [\mathcal{L}_{pq}(A|_\ell)]$, which is a rational subset of M_ℓ. Since we assume θ_ℓ to have automatic rational multiplication, the following set is effectively synchronous rational:

$$P_{p,q,\ell} \coloneqq \{(u,v) \in L_\ell \times (L_\ell \setminus \{\varepsilon\}) \mid \exists r \in R_{p,q,\ell}\colon \\ \theta_\ell(u)r = \theta_\ell(v)\} \tag{4}$$

Note that the right component is required to be non-empty, so this set is not quite the same as $R^{\odot}_{p,q,\ell}$, but it can be obtained from the latter by an intersection with the synchronous rational relation $L_\ell \times (L_\ell \setminus \{\})$ (recall that synchronous rational relations are closed under intersection (Elgot and Mezei 1965)). We also need two variants of $R^{\triangleleft}_{p,q,\ell}$, namely those that pick only encodings that are empty or start (end) in an ℓ-block:

$$I_{p,q,\ell} = R^{\triangleleft}_{p,q,\ell} \cap (\{\varepsilon\} \cup L_\ell(L_0 \cup L_1)^*) \tag{5}$$

$$I'_{p,q,\ell} = R^{\triangleleft}_{p,q,\ell} \cap (\{\varepsilon\} \cup (L_0 \cup L_1)^* L_\ell) \tag{6}$$

Finally, we use a variant of $R^{\triangleright}_{p,q,\ell}$ that requires encodings to be empty or to start in an ℓ-block:

$$W_{p,q,\ell} = R^{\triangleright}_{p,q,\ell} \cap (\{\varepsilon\} \cup L_\ell(L_0 \cup L_1)^*). \tag{7}$$

According to Lemma 3.12, the sets in Eqs. (5) to (7) are effectively regular.

Our task is to show that the set of pairs (u,v) such that there is an $r \in R$ with $\theta(u)r = \theta(v)$ is synchronous rational. Suppose $\theta(u)$, r, and $\theta(v)$ are given by reduced words x, y, and z, respectively. We have two ways of representing elements, via θ and as sequences of generators. However, since for a word $w \in Y^*$ and a block decomposition $w = w_1 \cdots w_n$, with $w_1 \in Y_i^*$, we have $\theta(w) = \theta_i(w_1)\theta_{1-i}(w_2)\theta_i(w_3)\cdots$, the product of two encodings via θ is built according to the same pattern of cancelation and of multiplication for sequences of generators.

Thus, Lemma 3.10 (and the remark above it) tell us that there are four possible situations. We call these cases *non-merging* (if x and y are non-merging), *suffix* (case (i) of Lemma 3.10), *prefix* (case (ii)), and *mixed* (case (iii)).

The simplest case is the non-merging case, since we just have to concatenate encodings x and y to get z. This means, the two components have a common prefix (corresponding to x) and then the second component proceeds with the encoding of an element of R. Hence, these pairs are represented in the following set:

$$T_1 \coloneqq \bigcup_{\ell \in \{0,1\}} E_\ell \cdot (\{\varepsilon\} \times W_{q_0,f,1-\ell}).$$

In the suffix case, y is split up in $y = y_1 y_2$ such that x cancels with y_1, so that the result is y_2. Therefore, we first guess the type ℓ of the last block of y_1 and we guess the state p that A is in after it has read the part y_1. Then, in the left component, we generate x as an element that can be canceled by one that is read from q_0 to p. In the right component, we generate y_2 as an element that is read by A from p to the final state:

$$T_2 \coloneqq \bigcup_{\ell \in \{0,1\}} \bigcup_{p \in Q} I'_{q_0,p,\ell} \times W_{p,f,1-\ell}.$$

Now, the prefix case works similarly to the suffix case: Here, x is split up as $x = x_1 x_2$ such that x_2 cancels with y:

$$T_3 \coloneqq \bigcup_{\ell \in \{0,1\}} E_\ell \cdot (I_{q_0,f,1-\ell} \times \{\varepsilon\}).$$

The mixed case is more involved. As depicted in Fig. 1, x and y decompose as $x = x_1 x_2 x_3$ and $y = y_1 y_2 y_3$ (which is not necessarily a block decomposition) such that (i) x_3 cancels with y_1, (ii) x_2 and y_2 are blocks of the same type ℓ that are multiplied as in M_ℓ and yield a result $\neq \mathbf{1}$, and (iii) the prefix x_1 of x and the suffix y_3 of y are passed to the result without modification. This is realized as follows. First, we guess the block type ℓ of x_2 and y_2 and which states the automaton A is in after it reads the part y_1 (p) and after it reads y_2 (q). The relation $E_{1-\ell}$ copies the unchanged prefix x_1. Afterwards, the relation $P_{p,q,\ell}$ generates x_2 on the left side and the product of x_2 and y_2 on the right side. Then, x_3 is described by $I_{q_0,p,\ell}$: Recall that x_3 cancels with y_1, which is read from q_0 to p. Finally, y_3 is described by $W_{q,f,1-\ell}$:

$$T_4 \coloneqq \bigcup_{\ell \in \{0,1\}} \bigcup_{p,q \in Q} E_{1-\ell} \cdot P_{p,q,\ell} \cdot (I_{q_0,p,1-\ell} \times W_{q,f,1-\ell}).$$

By Fact 3.11, each of the relations T_1, T_2, T_3, T_4, and hence their union, is synchronous rational. It can be verified that $R^{\odot}$ equals $T_1 \cup T_2 \cup T_3 \cup T_4$. See Appendix D for a detailed proof.

4. Undecidability

We show that if Γ is not a disjoint union of $\mathbb{B}^2$-triangle-free cliques, the first-order theory with reachability is undecidable — even for a fixed valence system over $\mathbb{M}\Gamma$. A graph is a disjoint union of cliques if and only if it is transitive. Therefore, a graph can fail to be a disjoint union of $\mathbb{B}^2$-triangle-free cliques for two reasons: Either Γ^- contains the graph •—•—• as an induced subgraph, or Γ contains a $\mathbb{B}^2$-triangle. To each of these cases we devote one section.

4.1 Undecidability: Non-Transitive Graphs

Suppose Γ^- contains •—•—• as an induced subgraph. This means $\mathbb{M}\Gamma$ contains a submonoid $(M_0 * M_1) \times M_2$, where each M_i is either $\mathbb{B}$ or $\mathbb{Z}$. In any case, $\mathcal{R}_1(\mathbb{M}\Gamma)$ contains a submonoid that is isomorphic to $\{a,b\}^* \times \{c\}^*$. We will show that undecidability can be proved by just relying on the submonoid $\{a,b\}^* \times \{c\}^*$; operationally, this means we will restrict ourselves to automata that never make use of barred symbols. That is, if we interpret a and b as the two symbols of a stack and c as representing a counter, the systems we will construct never perform a pop, nor a decrement. This is in sharp contrast to other questions in automata theory, where even bounding reversals yields decidability results (Ibarra 1978). Our main finding is that we can in fact give two arguments for undecidability, one where the formula is fixed, and one for a fixed valence system.

Theorem 4.1. *Assume $\mathbb{M}\Gamma$ is as discussed above.*

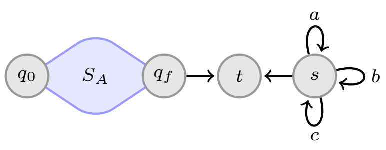
(i) There is a fixed $\mathsf{FO(R)}$*-formula that cannot be checked for valence systems over $\mathbb{M}\Gamma$.*

(ii) There is a fixed *valence system over $\mathbb{M}\Gamma$ with an undecidable first-order theory with reachability.*

For the first claim, we reduce the following undecidable problem to ours:

Theorem 4.2 (Sakarovitch (1992))**.** *Given a rational subset R of $\{a,b\}^* \times \{c\}^*$, it is undecidable to determine whether R equals $\{a,b\}^* \times \{c\}^*$.*

Let $R \subseteq \{a,b\}^* \times \{c\}^*$ be a rational set as recognized by the finite state automaton A. The automaton A can be turned into a valence system S_A with states q_f and q_0, such that (q_f, m) is reachable from $(q_0, \mathbf{1})$ if and only if $m \in R$. We now construct the following valence system:



In this new system, we can use the transitions on s to reach, from $(s, \mathbf{1})$ all and only the configurations (t, m) with $m \in \{a,b\}^* \times \{c\}^*$; moreover, from $(q_0, \mathbf{1})$ we can reach (t, m) if and only if $m \in R$. We therefore obtain that $S_A^P \models \forall c \colon \mathsf{state}_t(c) \wedge \mathsf{reach}((s,\mathbf{1}),c) \rightarrow \mathsf{reach}((q_0,\mathbf{1}),c)$ holds if and only if $R = P$. Note that the formula can be evaluated on any valence system having three states named t, s and q_0 respectively.

To establish the second claim, we reduce a variant of Post's Correspondence Problem (PCP). The *initialized PCP* that we rely on is the following.

Theorem 4.3 (Harju et al. (1996)[1])**.** *There is a fixed alphabet X, a fixed alphabet Y, and fixed morphisms $\alpha, \beta \colon X^* \rightarrow Y^*$ such that the following problem is undecidable. Given a word $u \in Y^*$, is there a word $w \in X^*$ satisfying $\alpha(w)u = \beta(w)$.*

[1] In the original result, the problem is finding w such that $u\alpha(w) = \beta(w)$. Our variant is equivalent via string reversal.

We encode an instance of initialized PCP into checking a Σ_1 formula over a valence system. The precise statement is in the next lemma, a proof of which concludes the proof of Theorem 4.1.

Lemma 4.4. *Take X, Y, α, and β from Theorem 4.3. There is a fixed valence system $S_{\alpha\beta}$ and a mapping from $u \in Y^*$ to the Σ_1 formula φ_u so that $S_{\alpha\beta} \models \varphi_u$ iff $\alpha(w)u = \beta(w)$ for some w.*

We give a construction that depends on X, Y, α, and β. By choosing the parameters appropriately, we arrive at the lemma. We begin with the construction of $S_{\alpha\beta}$. The alphabets X and Y can be assumed to be disjoint. We encode their union $X \cup Y = \{a_1, \ldots, a_n\}$ as words over the alphabet $\{a,b\}$. (Remember that we are sure to have the submonoid $\{a,b\}^* \times \{c\}^*$.) The encoding is via the morphism $\gamma \colon (X \cup Y)^* \rightarrow \{a,b\}^*$ with $\gamma(a_i) = ab^i$.

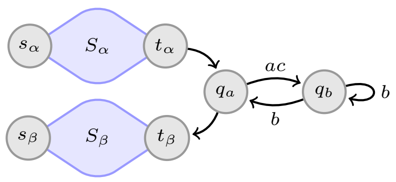
We now show how to represent morphisms $\mu \colon X^* \rightarrow Y^*$ by rational sets. In a first step, we represent μ as $W_\mu \subseteq \{a,b\}^* \times \{c\}^*$. The set encodes the pairs $w\mu(w)$ together with information about the length of $\mu(w)$. Formally, we have

$$W_\mu := \{(\gamma(w\mu(w)), c^k) \mid w \in X^*, k = |\mu(w)|\}.$$

In a second step, and this is the point in the definition of W_μ, we observe that the complement $\overline{W_\mu} = (\{a,b\}^* \times \{c\}^*) \setminus W_\mu$ is rational and effectively computable from μ, due to Sakarovitch (1992). Intuitively, since we have negation in the logic, it will not matter whether we use W_μ or its complement to represent μ.

An automaton recognising $\overline{W_\mu}$ can be turned into a valence system S_μ. To this end, we introduce distinguished states s_μ and t_μ such that the t_μ-configurations reachable from $(s_\mu, \mathbf{1})$ are exactly of the form (t_μ, m) with $m \in \overline{W_\mu}$. We can make sure t_μ does not have outgoing edges.

We construct $S_{\alpha\beta}$ by taking the union of S_α and S_β obtained as above, and adding new states q_a and q_b as below:



Note that $m \in W_\alpha$ iff (t_α, m) is *not* reachable from $(s_\alpha, \mathbf{1})$, and similar for (t_β, m).

We now construct the formula φ_u for the given word $u \in Y^*$. To explain the idea, consider a configuration $e_1 = (t_\alpha, (v_1, c^{k_1}))$ that is not reachable from $(s_\alpha, \mathbf{1})$. With the previous note, this means $v_1 = \gamma(w\alpha(w))$ and $k_1 = |\alpha(w)|$ for some $w \in X^*$. Assume e_1 leads to $e_2 = (t_\beta, (v_2, c^{k_2}))$ via a path that multiplies $\gamma(u)$ to the store. Then we have $v_2 = v_1\gamma(u) = \gamma(w\alpha(w)u)$ and $k_2 = k_1 + |u| = |\alpha(w)u|$. The relationship with the instance of the initialized PCP problem is as follows. Configuration e_2 is not reachable from $(s_\beta, \mathbf{1})$ iff $v_2 = \gamma(w'\beta(w'))$ and $k_2 = |\beta(w')|$ for some $w' \in X^*$. But since we also have $v_2 = \gamma(w\alpha(w)u)$ and $k_2 = |\alpha(w)u|$, we can conclude $w' = w$. We found a solution, namely $\beta(w) = \alpha(w)u$, to the initialized PCP instance.

Formula φ_u phrases the above setting in first-order logic. It is Σ_1 as we only have to existentially quantify over e_1 and e_2. Assume for the moment we have a predicate $\mathsf{path}_{\gamma(u)}(e_1, e_2)$ that guarantees the following. If it holds for configurations $e_1 = (t_\alpha, (v_1, c^{k_1}))$ and $e_2 = (t_\beta, (v_2, c^{k_2}))$, then $v_2 = v_1\gamma(u)$ and $k_2 = k_1 + |u|$. With this predicate, formula φ_u is

$$\begin{aligned}\exists e_1, e_2 \colon\ & \mathsf{state}_{t_\alpha}(e_1) \wedge \neg\mathsf{reach}((s_\alpha, \mathbf{1}), e_1) \\ & \wedge \mathsf{state}_{t_\beta}(e_2) \wedge \neg\mathsf{reach}((s_\beta, \mathbf{1}), e_2) \\ & \wedge \mathsf{path}_{\gamma(u)}(e_1, e_2)\ .\end{aligned}$$

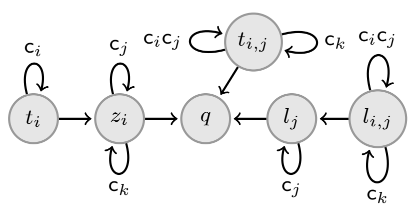


Figure 2. Gadget supporting auxiliary properties in $S_{\mathbb{N}}$, for all i, j, k such that $\{i, j, k\} = \{1, 2, 3\}$.

The argumentation in the previous paragraph derives the desired equivalence in Lemma 4.4.

It remains to define the predicate $\mathsf{path}_{\gamma(u)}(e, e')$. Let $\gamma(u)$ be the word $x_1 \dots x_n$ from $\{a, b\}^*$. The trick is to refer to the control states q_a and q_b:

$$\exists e_1 \dots e_{n+1} \colon \mathsf{step}(e, e_1) \wedge \mathsf{step}(e_{n+1}, e') \wedge \mathsf{state}_{q_a}(e_{n+1}) \\ \wedge \textstyle\bigwedge_{i=1}^{n} \mathsf{step}(e_i, e_{i+1}) \wedge \bigwedge_{i=1}^{n} \mathsf{state}_{q_{x_i}}(e_i).$$

It is readily checked that the predicate satisfies the requirements.

4.2 Undecidability: $\mathbb{B}^2$-Triangle

Suppose Γ contains a $\mathbb{B}^2$-triangle as an induced subgraph. Then, $\mathbb{M}\Gamma$ contains $\mathbb{B}^2 \times \mathbb{Z}$ or $\mathbb{B}^3$ as a submonoid. In any case, it contains the monoid $\mathbb{B} \times \mathbb{B} \times \mathbb{N}$ as a submonoid. By making use of this substructure, we construct a valence system $S_{\mathbb{N}}$ over $\mathbb{M}\Gamma$ in which the structure $(\mathbb{N}, +, \cdot)$ of the natural numbers with addition and multiplication can be interpreted. The Σ_1 fragment of arithmetic with addition and multiplication is undecidable (Matiyasevich 1993), which, as we show, implies undecidability of the Σ_2 fragment over $\mathbb{M}\Gamma$.

Theorem 4.5. *For the (fixed) valence system $S_{\mathbb{N}}$ constructed below, checking the Σ_2 fragment of* $\mathsf{FO}(\mathsf{R})$ *is undecidable.*

To derive the undecidability, it suffices to interpret the structure $(\mathbb{N}, +, \cdot^2)$ with addition and squaring ($n \mapsto n^2$). This is because the identity $2ab = (a+b)^2 - a^2 - b^2$ allows us to express multiplication in terms of squaring and addition. In the remainder of the section we thus show how to set up $S_{\mathbb{N}}$ so that addition and squaring can be interpreted as first-order properties of reachable configurations.

We interpret the number $n \in \mathbb{N}$ as the configuration $(q, n, 0, 0)$ of the submonoid $\mathbb{B} \times \mathbb{B} \times \mathbb{N}$ of $\mathbb{M}\Gamma$. So we fix a distinguished control state q. We will refer to configurations (q, n_1, n_2, n_3) as q-configurations. For a q-configuration $c = (q, n_1, n_2, n_3)$, let the projection $\pi_i(c) = n_i$ return the value of the i-th counter with $i \in \{1, 2, 3\}$. We write c_i and $\bar{\mathsf{c}}_i$ for the increment and decrement operation on the i-th counter with $i \in \{1, 2, 3\}$.

The valence system $S_{\mathbb{N}}$ consists of the gadgets in Figs. 2 and 3 that we explain one by one. The first gadget to include in $S_{\mathbb{N}}$ is depicted in Fig. 2. To be precise, we add a copy of these states and transitions for every combinatation i, j, k such that $\{i, j, k\} = \{1, 2, 3\}$. (The state q is the distinguished one. It is shared among the copies.) These transitions allow us to express useful auxiliary properties of q-configurations. In what follows, we implicitly require $\mathsf{state}_q(c)$ and/or $\mathsf{state}_q(d)$:

(i) We can express $\pi_i(c) = \pi_j(c)$ with $\mathsf{reach}((t_{i,j}, 0, 0, 0), c)$.

(ii) We can express $\pi_i(c) = 0$ with $\mathsf{reach}((z_i, 0, 0, 0), c)$.

(iii) We can express $\pi_i(c) = \pi_i(d)$ with $\exists e \colon \mathsf{reach}((t_i, 0, 0, 0), e) \wedge \mathsf{state}_{z_i}(e) \wedge \mathsf{reach}(e, c) \wedge \mathsf{reach}(e, d)$.

(iv) We can express $\pi_i(c) = \pi_j(d)$ with $\exists e \colon \pi_i(c) = \pi_i(e) \wedge \pi_i(e) = \pi_j(e) \wedge \pi_j(e) = \pi_j(d)$.

(v) We can express $\pi_i(c) \leq \pi_j(c)$ with $\mathsf{reach}((l_{i,j}, 0, 0, 0), c)$.

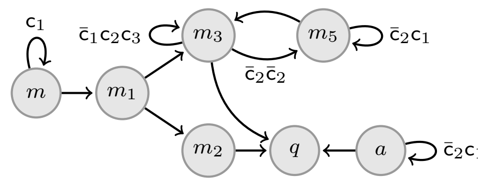

m
m_1
m_2
m_3
m_5
q
a
c_1
$\bar{\mathsf{c}}_1\mathsf{c}_2\mathsf{c}_3$
$\bar{\mathsf{c}}_2\mathsf{c}_1$
$\bar{\mathsf{c}}_2\bar{\mathsf{c}}_2$
$\bar{\mathsf{c}}_2\mathsf{c}_1$

Figure 3. Gadget supporting addition and squaring in $S_{\mathbb{N}}$

(vi) We can express $\pi_i(c) \leq \pi_i(d)$ with $\exists e \colon \pi_i(e) \leq \pi_j(e) \wedge \pi_i(e) = \pi_i(c) \wedge \pi_j(e) = \pi_j(d)$.

Note how all these formulas belong to the Σ_1 fragment of the logic. Now we are ready to show how to interpret first-order logic over the naturals.

Domain We represent the domain of the naturals with the configurations satisfying $\mathsf{state}_q(x) \wedge \pi_2(x) = 0 \wedge \pi_3(x) = 0$.

Addition To represent addition, we include in $S_{\mathbb{N}}$ the state a and the relevant portion of the gadget in Fig. 3, again sharing q with the rest of the construction. We can then express $\pi_1(d) = \pi_1(c) + \pi_2(c)$ with the Σ_1 formula

$$\exists e_1\ e_2\ d' \colon \mathsf{state}_a(e_1) \wedge \mathsf{step}(e_1, c) \\ \wedge \mathsf{state}_a(e_2) \wedge \mathsf{reach}(e_1, e_2) \\ \wedge \mathsf{state}_q(d') \wedge \mathsf{step}(e_2, d') \\ \wedge \pi_2(d') = 0 \wedge \pi_1(d) = \pi_1(d').$$

The trick is to add the second to the first counter of c, and make sure the first counter of d matches this value. To be more precise, we guess a configuration e_1 that has the same counter values as c. This is guaranteed by $\mathsf{step}(e_1, c)$. Then we transfer the value of the second counter in e_1 to the first counter, using the path from e_1 to e_2. We have to check that we have transferred the full counter value $\pi_2(e_1)$. To this end, we perform a transition from e_2 to the q-configuration d' and check $\pi_2(d') = 0$. Note that the auxiliary predicate $\pi_2(d') = 0$ can only be used, because d' is a q-configuration. It cannot be used for e_2. All that remains is to compare the value of the first counter in d to the first counter in d', again using one of the auxiliary predicates.

With the above, we can interpret $\pi_1(e) = \pi_1(c) + \pi_1(d)$ in Σ_1 with the formula $\exists e' \colon \pi_1(e') = \pi_1(c) \wedge \pi_2(e') = \pi_1(d) \wedge \pi_1(e) = \pi_1(e') + \pi_2(e')$.

Squaring To interpret squaring we add the transitions in Fig. 3 to $S_{\mathbb{N}}$. We then express squaring in two steps.

First, we express $\pi_1(d) \leq \pi_1(c)^2$. We rely on the fact that $n^2 = \sum_{i=0}^{n-1} 2i + 1$. With this equation, it is sufficient to reach all configurations $(q, 0, 0, n')$ where $n' \leq \sum_{i=1}^{n} 2(n-i) + 1$. Indeed, the desired inequality can now be phrased as $\pi_1(d)$ being one of the values n'. To compute the sum, we define a configuration c', using the interpretation for addition above, such that $c' = (q, 2n-1, 0, 0)$. We write this property as $\mathsf{init}(c, c')$.

The trick for making sure that $n' \leq \sum_{i=1}^{n} 2(n-i) + 1$ is the following. We start with the value $2n - 1$ in the first counter. We think of the counter values as tokens that can be moved, so that we have $2n - 1$ tokens in the first component. Intuitively, we move the counter value back and forth between the first and the second component. This happens between the states m_3 and m_5. However, each time we move the tokens from one component to the other, we lose 2 tokens. Moreover, each time we move one token from the first to the second component, we increment the third. Now if we could make sure to move all tokens in every iteration, we would end up with counter value $\sum_{i=1}^{n}(2i - 1) = n^2$ in the third component. This we cannot guarantee, but we know that each time we take *at most* the tokens that are there (recall that the first two components of our storage are $\mathbb{B}$, i.e. partially blind counters). Hence, we end up with counter value at most n^2 in the third component.

We return to our formula. Starting in configuration $(m, 0, 0, 0)$ we increment the first counter to value n_1 and move to $c_1 = (m_1, n_1, 0, 0)$. We have to ensure that $n_1 = 2n - 1$. To this end, we require the existence of a path from c_1 via m_2 to a configuration c' that satisfies $\mathsf{init}(c, c')$. That the path is via m_2 guarantees $n_1 = \pi_1(c')$. That the init predicate holds yields $\pi_1(c') = 2n - 1$. Together, this is the equality we need.

We have to do the summation. From $c_1 = (m_1, n_1, 0, 0)$, we go to $c_3 = (m_3, n_1, 0, 0)$ from which we execute the m_3-m_5-loop. Eventually, we leave the loop and get to $d' = (q, \_, \_, n')$. Since this is a q-configuration, we can use the auxiliary predicates and require $\pi_1(d) = \pi_3(d')$. The above argumenation on the construction of the gadget together with the fact that $n_1 = 2n - 1$ guarantees that we leave the loop with $n' \leq \sum_{i=1}^{n} 2(n - i) + 1 = n^2$.

Formally, we interpret $\pi_1(d) \leq \pi_1(c)^2$ with the Σ_1 formula $\varphi_{\leq \mathrm{sq}}(c, d)$ defined as follows:

$$\begin{aligned}\exists c'\ d'\ c_1\ c_2\ c_3 \colon & \mathsf{init}(c, c') \wedge \mathsf{state}_q(d') \wedge \pi_1(d) = \pi_3(d') \\ & \wedge \mathsf{reach}((m, 0, 0, 0), c_1) \wedge \mathsf{state}_{m_1}(c_1) \\ & \wedge \mathsf{step}(c_1, c_2) \wedge \mathsf{state}_{m_2}(c_2) \wedge \mathsf{step}(c_2, c') \\ & \wedge \mathsf{step}(c_1, c_3) \wedge \mathsf{state}_{m_3}(c_3) \wedge \mathsf{reach}(c_3, d') \ .\end{aligned}$$

In Appendix E, we elaborate on the correctness of the construction.

To complete the construction we express $\pi_1(d) = \pi_1(c)^2$ in Π_1. We state that d is the q-configuration with the maximal value in the first counter that is below $\pi_1(c)^2$. Formally, for every q-configuration e we either have $\pi_1(e) > \pi_1(c)^2$ or $\pi_1(e) \leq \pi_1(d)$. The corresponding Π_1 formula is

$$\varphi_{\mathrm{sq}}(c, d) := \varphi_{\leq \mathrm{sq}}(c, d) \wedge \forall e \colon \neg\varphi_{\leq \mathrm{sq}}(c, e) \vee \pi_1(e) \leq \pi_1(d) \ .$$

Formula $\varphi_{\mathrm{sq}}(c, d)$ is the needed interpretation of $\pi_1(d) = \pi_1(c)^2$.

If we now take a Σ_1 formula over $(\mathbb{N}, +, \cdot)$ and express addition as above and multiplication via $\varphi_{\mathrm{sq}}(\cdot, \cdot)$ and the identity $2ab = (a + b)^2 - a^2 - b^2$, we arrive at a Σ_2 formula. On the whole, we obtain the following result.

Lemma 4.6. *Assume Γ is not $\mathbb{B}^2$-triangle-free and construct the (fixed) valence system $S_{\mathbb{N}}$ over $\mathbb{M}\Gamma$ above. For each first-order formula φ over the naturals with addition and multiplication, we can produce a φ' over the configuration graph so that*

$$(\mathbb{N}, +, \cdot) \models \varphi \quad \textit{iff} \quad S_{\mathbb{N}} \models \varphi'.$$

Moreover, if φ belongs to the Σ_1 fragment, φ' is in Σ_2.

To deduce undecidability of the Σ_2 fragment of $\mathsf{FO(R)}$ for a fixed valence system, note that the Σ_1 fragment of arithmetic with addition and multiplication is undecidable (Matiyasevich 1993). This concludes the proof of Theorem 4.5.

5. Discussion and Future Work

We provided a sufficient and necessary condition on a graph Γ such that $\mathsf{FO(R)}$ for valence systems over the monoid $\mathbb{M}\Gamma$ is decidable. This result generalizes previous results on verification of infinite-state systems and provides a full characterization of the shape of the storage mechanisms enjoying decidability of $\mathsf{FO(R)}$. The techniques employed in the proofs are robust enough to support extensions. For example, the results would still hold when adding a finite input alphabet X to valence systems and adding labeled predicates $\mathsf{step}_x(\cdot)$ with $x \in X$ and $\mathsf{reach}_R(\cdot, \cdot)$ for a regular language $R \subseteq X^*$.

As future work, it could be interesting to study whether similar characterizations exist for ordinary first-order logic or richer branching-time logics, such as the modal μ-calculus.

References

P. A. Abdulla, B. Jonsson, M. Nilsson, and M. Saksena. A survey of regular model checking. In *CONCUR*, volume 3170 of *LNCS*, pages 35–48. Springer, 2004.

M. Benois. Parties rationnelles du groupe libre. *CR Acad. Sci. Paris*, 269: 1188–1190, 1969.

A. Blumensath and E. Grädel. Automatic structures. In *LICS*, pages 51–62. IEEE, 2000.

B. Boigelot and P. Wolper. On the expressiveness of real and integer arithmetic automata. In *ICALP*, volume 1443 of *LNCS*, pages 152–163. Springer, 1998.

A. Bouajjani, J. Esparza, and O. Maler. Reachability analysis of pushdown automata: Application to model-checking. In *CONCUR*, volume 1243 of *LNCS*, pages 135–150. Springer, 1997.

J. R. Büchi. Weak seond-order arithmeti and finite automata. *Math. Logik und Grundlagen der Mathematik*, 6:66–92, 1960.

P. Buckheister and G. Zetzsche. Semilinearity and context-freeness of languages accepted by valence automata. In *MFCS*, volume 8087 of *LNCS*, pages 231–242. Springer, 2013.

A. Carayol and S. Wöhrle. The caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In *FSTTCS*, volume 2914 of *LNCS*, pages 112–123. Springer, 2003.

D. Caucal. On infinite transition graphs having a decidable monadic theory. In *ICALP*, volume 1099 of *LNCS*, pages 194–205. Springer, 1996.

T. Colcombet. On families of graphs having a decidable first order theory with reachability. In *ICALP*, volume 2380 of *LNCS*, pages 98–109. Springer, 2002.

P. Darondeau, S. Demri, R. Meyer, and C. Morvan. Petri net reachability graphs: Decidability status of FO properties. In *FSTTCS*, volume 13 of *LIPIcs*, pages 140–151. Schloss Dagstuhl, 2011.

C. Elgot and J. E. Mezei. On relations defined by generalized finite automata. *IBM Journal of Research and Development*, 9(1):47–68, Jan 1965.

T. Harju, J. Karhumäki, and D. Krob. Remarks on generalized post correspondence problem. In *STACS*, volume 1046 of *LNCS*, pages 39–48. Springer, 1996.

O. H. Ibarra. Reversal-bounded multicounter machines and their decision problems. *Journal of the ACM (JACM)*, 25(1):116–133, 1978.

B. Khoussainov and A. Nerode. Automatic presentations of structures. In *LCC: International Workshop on Logic and Computational Complexity*, volume 960 of *LNCS*, pages 367–392. Springer, 1995.

B. Khoussainov, S. Rubin, and F. Stephan. Definability and regularity in automatic structures. In *STACS*, pages 440–451. Springer, 2004.

J. Leroux and G. Sutre. On flatness for 2-dimensional vector addition systems with states. In *CONCUR*, volume 3170 of *LNCS*, pages 402–416. Springer, 2004.

D. Lugiez and P. Schnoebelen. Decidable first-order transition logics for pa-processes. In *ICALP*, volume 1853 of *LNCS*, pages 342–353. Springer, 2000.

Y. V. Matiyasevich. *Hilbert's Tenth Problem.* MIT Press, Cambridge, 1993.

C.-H. L. Ong. On model-checking trees generated by higher-order recursion schemes. In *LICS*, pages 81–90. IEEE Computer Society, 2006.

R. J. Parikh. On context-free languages. *Journal of the ACM (JACM)*, 13(4): 570–581, 1966.

J. Sakarovitch. The "last" decision problem for rational trace languages. In *Proc. of LATIN*, volume 583 of *LNCS*, pages 460–473. Springer, 1992.

S. Schulz. First-order logic with reachability predicates on infinite systems. In *FSTTCS*, volume 8 of *LIPIcs*, pages 493–504. Schloss Dagstuhl, 2010.

I. Walukiewicz. Pushdown processes: Games and model checking. In *CAV*, volume 1102 of *LNCS*, pages 62–74. Springer, 1996.

G. Zetzsche. Silent transitions in automata with storage. In *ICALP*, volume 7966 of *LNCS*, pages 434–445. Springer, 2013.

G. Zetzsche. The emptiness problem for valence automata or: Another decidable extension of Petri nets. In *Reachability Problems*, volume 9328 of *LNCS*, pages 166–178. Springer, 2015.

A. Proof of Theorem 3.2

Suppose the monoid M has an encoding $\theta\colon L \to \mathcal{R}_1(M)$, $L \subseteq Y^*$, with automatic rational multiplication and let $S = (Q, M, E)$ be a valence system over M. Since M is finitely generated, we have a surjective morphism $\varphi\colon X^* \to M$.

The reachability structure for S has the domain $D = Q \times \mathcal{R}_1(M)$, so we assume $Q \cap Y = \emptyset$ and take the regular language $QL = \{qx \mid q \in Q, x \in L\}$ to represent D. As expected, the bijection is then $\eta\colon QL \to D$ with $\eta(qx) = (q, \theta(x))$, where $q \in Q$ and $x \in L$. We have to verify that each of the relations in the reachability structure have synchronous rational encodings.

(i) The $\mathsf{state}_q(\cdot)$ predicate. Clearly, the set of its encodings is qL, which is regular.

(ii) The rechability predicate $\mathsf{reach}(\cdot,\cdot)$. For each pair $p, q \in Q$, the set R_{pq} of all $m \in M$ such that $(p, 1) \to_S^* (q, m)$ is a rational set: Take the automaton A with state set Q and whenever there is an edge (r, m, s) in S, with $\varphi(w)$, create an edge labeled w between r and s. This automaton clearly satisfies $\varphi(\mathcal{L}_{pq}(A)) = R_{pq}$, so that R is rational. The word relation corresponding to $\mathsf{reach}(\cdot,\cdot)$ can now be written as

$$\begin{aligned}&\bigcup_{p,q\in Q} \{(px, qy) \in QL \times QL \mid \exists r \in R_{pq}\colon \theta(x)r = \theta(y)\}\\ =&\bigcup_{p,q\in Q} \{(px, qy) \in QL \times QL \mid (\theta(x), \theta(y)) \in R_{pq}\}\\ =&\bigcup_{p,q\in Q} \{(p, q)\} \cdot R_{pq}^{\odot},\end{aligned}$$

which is synchronous rational (Fact 3.11).

(iii) The one-step reachability relation $\mathsf{step}(\cdot,\cdot)$. Here, we can proceed as in the previous case, except that instead of the above R_{pq}, we take the finite set of all $m \in M$ for which there is a transition (p, m, q) in S. Note that every finite set is rational.

(iv) Constants. Again, we use the fact that one-element subsets of M are rational. Hence, given a state $q \in Q$ and a word $w \in X^*$ such that $\varphi(w) \in \mathcal{R}_1(M)$, we can clearly compute the word $x \in L$ such that $\theta(x) = \varphi(w)$ and thus $\eta(qx) = (q, \varphi(w))$.

B. Proof of Lemma 3.10

Proof. The "if" statement can be checked straightforwardly. We prove the "only if" direction. Let $i \in [0, \min(m, n)]$ be maximal with the property that for all $j \in [1, i]$, we have $[u_{m-j+1}v_j] = \mathbf{1}$. We distinguish three cases.

- Suppose $i = \min(m, n)$ and $m \le n$. We claim that we are in situation (i) above. The first condition is clearly fulfilled. Moreover, the equations $[u_{m-j+1}v_j] = \mathbf{1}$ for $j \in [1, m]$ mean that
$$[w_1 \cdots w_k] = [uv] = [u_1 \cdots u_m v_1 \cdots v_n] = [v_{m+1} \cdots v_n].$$
Hence, Fact 3.8 yields the second and third condition.
- Suppose $i = \min(m, n)$ and $m > n$. We claim that we are in situation (ii) above. This can be shown analogously to the second case.
- Suppose $i < \min(m, n)$. We claim that we are in situation (iii) above. The first three conditions are clearly met. Furthermore, the equations $[u_{m-j+1}v_j] = \mathbf{1}$ for $j \in [1, i]$ imply that
$$[w_1 \cdots w_k] = [uv] = [u_1 \cdots u_{m-i}v_{i+1} \cdots v_n]$$
and since $[u_{m-i}v_{i+1}] \neq \mathbf{1}$ by maximality of i, the word $u_1 \cdots u_{m-i}v_{i+1} \cdots v_n$ is reduced. According to Fact 3.8, this entails the last three conditions.

□

C. Proof of Lemma 3.12

We begin with the identity $R^{\triangleright} = \sigma(K)$. Suppose $w \in R^{\triangleright}$ and let $w = w_1 \cdots w_n$ be the block decomposition. Then $\theta(w) = \theta(w_1) \cdots \theta(w_n)$ and $\theta(w_i) \neq \mathbf{1}$ for $i \in [1, n]$. This means there is a word $v \in X^*$ with block decomposition $v = v_1 \cdots v_n$ such that $[v_i] = \theta(w_i)$ for $i \in [1, n]$. Note that v is reduced. Since $[v] = \theta(w) \in R$, there is a word $u \in \mathcal{L}(A)$ with $[v] = [u]$. Because of our saturation, we may assume that u is reduced. If $u = u_1 \cdots u_m$ is its block decomposition, then this implies $m = n$ and $[u_i] = [v_i]$ for $i \in [1, n]$. We distinguish two cases.

- If $n = 0$, then we have $[v] = [u] = \mathbf{1}$ and thus $w = \theta^{-1}([v]) = \varepsilon$. On the other hand, this means $\mathbf{1} \in R$ and therefore $\varepsilon \in K$. Thus, we have $w = \varepsilon \in \sigma(K)$.
- Suppose $n > 0$ and consider a computation of A that reads $u = u_1 \cdots u_n$:
$$q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} \cdots \xrightarrow{u_n} q_n.$$
Moreover, let $i_j \in \{0, 1\}$ be the number with $u_j \in X_i^*$ for $j \in [1, n]$. Then, clearly, the word $x = (q_0, i_1, q_1)(q_1, i_2, q_2) \cdots (q_{n-1}, i_n, q_n)$ is contained in K. Moreover, since $[u_j] = [v_j] = \theta(w_j)$ and $[u_j] \in [\mathcal{L}_{pq}(A|_{i_j})] = R_{pi_jq}$, we have $w_j \in \sigma((q_{j-1}, i_j, q_j))$ (recall that $w_j \neq \varepsilon$ because $\theta(w_j) \neq \mathbf{1}$). This means $w = w_1 \cdots w_n \in \sigma(x) \subseteq \sigma(K)$.

This proves $R^{\triangleright} \subseteq \sigma(K)$. We turn to the converse inclusion. Let $w \in \sigma(K)$. If $w = \varepsilon$, then $\varepsilon \in K$ and hence $\mathbf{1} \in R$ by definition of K. This means $w = \varepsilon \in R^{\triangleright}$. Suppose $w \neq \varepsilon$ and let $x = (p_0, i_1, p_1)(p_1, i_2, p_2) \cdots (p_{n-1}, i_n, p_n) \in K$ such that $w \in \sigma(x)$. This means $w = w_1 \cdots w_n$ such that $w_j \in \sigma((p_{j-1}, i_j, p_j))$. This implies that for each $j \in [1, n]$, we have $\theta(w_j) = [u_j]$ for some $u_j \in \mathcal{L}_{p_{j-1}p_j}(A|_{i_j})$. Then the word $u = u_1 \cdots u_n$ is accepted by A and satisfies $[u] = \theta(w)$. Since this means $\theta(w) = [u] \in R$, we have $w \in R^{\triangleright}$. This completes the proof of $R^{\triangleright} = \sigma(K)$.

We now turn to the identity $R^{\triangleleft} = \tau(K^{\mathsf{rev}})$. Let $w \in R^{\triangleleft}$ and let $w = w_1 \cdots w_n$ be the block decomposition. Then $\theta(w) = \theta(w_1) \cdots \theta(w_n)$ and $\theta(w_i) \neq \mathbf{1}$ for $i \in [1, n]$. This means there is a word $v \in X^*$ with block decomposition $v = v_1 \cdots v_n$ such that $[v_i] = \theta(w_i)$ for $i \in [1, n]$. Note that v is reduced. Since there is an $r \in R$ with $[v]r = \theta(w)r = \mathbf{1}$, there is a word $u \in \mathcal{L}(A)$ with $[u] = r$ and thus $[vu] = [v][u] = \mathbf{1}$. Since we saturated A, we may assume that u is reduced. Let $u = u_1 \cdots u_m$ be its block decomposition. Now we have u and v reduced and we know $[vu] = \mathbf{1} = [\varepsilon]$. The words v and u clearly have to be merging, since otherwise the equality $[vu] = [\varepsilon]$ would contradict Fact 3.8. We can therefore apply Lemma 3.10. Note that the only case where the resulting word can be empty is case (i). This implies that $m = n$ and $[v_{n-j+1}u_j] = \mathbf{1}$ for all $j \in [1, n]$. As above, we distinguish two cases.

- If $n = 0$, then we have $[v] = [u] = \mathbf{1}$ and thus $w = \theta^{-1}([v]) = \varepsilon$. On the other hand, this means $\mathbf{1} \in R$ and therefore $\varepsilon \in K^{\mathsf{rev}}$. Thus, we have $w = \varepsilon \in \tau(K^{\mathsf{rev}})$.
- Suppose $n > 0$ and consider a computation of A that reads $u = u_1 \cdots u_n$:
$$q_0 \xrightarrow{u_1} q_1 \xrightarrow{u_2} \cdots \xrightarrow{u_n} q_n.$$
Moreover, let $i_j \in \{0, 1\}$ be the number with $u_j \in X_i^*$ for $j \in [1, n]$. Then, clearly, the word $x = (q_{n-1}, i_n, q_n)(q_{n-2}, i_{n-1}, q_{n-1}) \cdots (q_0, i_1, q_1)$ is contained in K^{rev}. Moreover, since $\theta(w_{n-j+1})[u_j] = [v_{n-j+1}][u_j] = \mathbf{1}$ and $[u_j] \in [\mathcal{L}_{pq}(A|_{i_j})] = R_{pi_jq}$, we have $w_{n-j+1} \in \tau((q_{j-1}, i_j, q_j))$ for all $j \in [1, n]$ (recall that $w_{n-j+1} \neq \varepsilon$ because $\theta(w_{n-j+1}) \neq \mathbf{1}$). In other words, we have $w_j \in$

$\tau((q_{n-j}, i_{n-j+1}, q_{n-j+1})$ for all $j \in [1,n]$. This means $w = w_1 \cdots w_n \in \tau(x) \subseteq \tau(K^{\mathsf{rev}})$.

This proves $R^{\triangleright} \subseteq \tau(K^{\mathsf{rev}})$. Now suppose $w \in \tau(K^{\mathsf{rev}})$. Again, $w = \varepsilon$ clearly implies $\varepsilon \in K$ and hence $\mathbf{1} \in R$ and thus $w = \varepsilon \in R^{\triangleright}$. Suppose $w \neq \varepsilon$. Then there is an $x = (q_{n-1}, i_n, q_n)(q_{n-2}, i_{n-1}, q_{n-1}) \cdots (q_0, i_1, q_1)$ such that $w = w_1 \cdots w_n$ with

$$w_j \in \tau((q_{n-j}, i_{n-j+1}, q_{n-j+1})).$$

This implies that for each $j \in [1,n]$, there is a

$$u_j \in \mathcal{L}_{q_{n-j}q_{n-j+1}}(A|_{i_{n-j+1}})$$

with $\theta(w_j)[u_j] = \mathbf{1}$. Therefore, if we set $u = u_n \cdots u_1$, then we have $u \in \mathcal{L}(A)$ and thus $[u] \in R$. Hence

$$\theta(w)[u] = \theta(w_1) \cdots \theta(w_n)[u_n] \cdots [u_1] = \mathbf{1}$$

and thus $w \in R^{\triangleleft}$. This completes the proof of $R^{\triangleleft} = \tau(K^{\mathsf{rev}})$.

D. Proof of Theorem 3.7

It remains to be shown that $R^{\odot} = T_1 \cup T_2 \cup T_3 \cup T_4$. We begin with the inclusion "$\subseteq$", so we assume that $x, y, z \in L$ with $\theta(x)\theta(y) = \theta(z)$ for some $\theta(y) \in R$. We want to show that $(x,z) \in T_1 \cup T_2 \cup T_3 \cup T_4$. There are reduced words $u, w \in X^*$ and $v \in \mathcal{L}(A)$ with $[u] = \theta(x)$, $[v] = \theta(y)$, and $[w] = \theta(z)$ (recall that A is saturated). Let

$$u = u_1 \cdots u_m, \qquad x = x_1 \cdots x_m,$$
$$v = v_1 \cdots v_n, \qquad y = y_1 \cdots y_n,$$
$$w = w_1 \cdots w_k, \qquad z = z_1 \cdots z_k,$$

be block decompositions (note that since u, v, and w are reduced, they have the same number of blocks as x, y, and z, respectively). Then we have $[u_i] = \theta(x_i)$, $[v_i] = \theta(y_i)$, and $[w_i] = \theta(z_i)$ for all possible indices i. Since $v \in \mathcal{L}(A)$, there is a computation

$$q_0 \xrightarrow{v_1} q_1 \xrightarrow{v_2} q_2 \xrightarrow{v_3} \cdots \xrightarrow{v_n} q_n \tag{8}$$

with $q_n = f$ in A.

Since $[u][v] = [w]$, we can apply Lemma 3.10, which leaves us with four cases.

(i) Non-merging. If u and v are non-merging, then $w = uv$ and hence $z = xy$. Note that $y \in W_{q_0,f,1-\ell}$ for some $\ell \in \{0,1\}$, whether y is empty or not. Then, since u and v are non-merging, we have $(x,x) \in E_\ell$. Thus, $(x,z) = (x,xy) \in T_1$.

(ii) The suffix case (case (i) of Lemma 3.10). We have $m \leq n$ and $[u_{m-j+1}v_j] = \mathbf{1}$ for $j \in [1,m]$. This means that $\theta(x)\theta(y_1 \cdots y_m) = \mathbf{1}$ and thus $x \in R^{\triangleleft}_{q_0,q_m}$ since $\theta(y_1 \cdots y_m) \in R_{q_0,q_m}$.

Note that $y_{m+1} \cdots y_n \in R^{\triangleright}_{q_m,f}$. There is clearly an $\ell \in \{0,1\}$ such that $x \in I'_{q_0,q_m,\ell}$ and $y_{m+1} \cdots y_n \in W_{q_m,f,1-\ell}$. Hence, we have

$$(x,z) = (x, y_{m+1} \cdots y_n) \in T_2.$$

(iii) The prefix case (case (ii)). We have $m > n$ and and $[u_{m-j+1}v_j] = \mathbf{1}$ for $j \in [1,n]$. This means that $\theta(x)\theta(y) = \theta(x_1 \cdots x_{m-n})$ and hence $z = x_1 \cdots x_{m-n}$. Since $m > n$, z is non-empty, so there is a unique $\ell \in \{0,1\}$ with $(z,z) \in E_\ell$. Since $[u_{m-n+1} \cdots u_m][v] = \mathbf{1}$ and $[v] \in R$, we have $x_{m-n+1} \cdots x_m \in I_{q_0,f,\ell}$. Therefore,

$$(x,z) = (x_1 \cdots x_{m-n}x_{m-n+1} \cdots x_n, x_1 \cdots x_{m-n}) \in T_3.$$

(iv) The mixed case. We have an $i \in [0, \min(m,n)]$ so that $[u_{m-j+1}v_j] = \mathbf{1}$ for $j \in [1,i]$ and $[u_{m-i}v_{i+1}] \neq \mathbf{1}$. From conditions (iii)d and (iii)f, Fact 3.8, and the injectivity of θ, we may conclude

$$z_1 \cdots z_{m-i-1} = x_1 \cdots x_{m-i-1},$$
$$z_{m-i+1} \cdots z_k = y_{i+2} \cdots y_n.$$

Let $\ell \in \{0,1\}$ be the type of the block v_{i+1} (and hence of u_{m-i} and x_{m-i}). Then x_{m-i-1} is of type $1-\ell$ and thus

$$(x_1 \cdots x_{m-i-1}, z_1 \cdots z_{m-i-1}) \in E_{1-\ell}. \tag{9}$$

Note that since we have the computation (8), if we set $p = q_i$ and $q = q_{i+1}$, then we have $[v_{i+1}] \in R_{p,q,\ell}$. Since

$$\theta(x_{m-i})[v_{i+1}] = [u_{m-i}v_{i+1}] = [w_{m-i}] = \theta(z_{m-i})$$

we have

$$(x_{m-i}, z_{m-i}) \in P_{p,q,\ell} \tag{10}$$

Furthermore, $[v_1 \cdots v_i] \in R_{q_0,p,1-\ell}$ and

$$\theta(x_{m-i+1} \cdots x_m)[v_1 \cdots v_i]$$
$$= [u_{m-i+1} \cdots u_m][v_1 \cdots v_i] = \mathbf{1},$$

which together implies

$$x_{m-i+1} \cdots x_m \in I_{q_0,p,1-\ell}. \tag{11}$$

Finally, we have $[v_{i+2} \cdots v_n] \in R_{p,f,1-\ell}$ and hence

$$z_{m-i+1} \cdots z_k = y_{i+2} \cdots y_n \in W_{q,f,1-\ell}. \tag{12}$$

Together, Eqs. (9) to (12) imply that $(x,z) \in T_4$.

This proves that $R^{\odot} \subseteq T_1 \cup T_2 \cup T_3 \cup T_4$.

Suppose $(x,z) \in T_1$. This means we have $z = xy$, x is either empty or ends in an ℓ-block, and $y \in W_{q_0,f,1-\ell}$. In particular, we have $\theta(y) \in R$, so that there is a reduced word $v \in \mathcal{L}(A)$ with $[v] = \theta(y)$. Let $u, w \in X^*$ be reduced words with $[u] = \theta(x)$ and $[w] = \theta(z)$. Since y is empty or begins in a block of type $1-\ell$, the same is true of v. For similar reasons, u is empty or ends in an ℓ-block. Hence, u and v are non-merging and we have

$$\theta(x)\theta(y) = [u][v] = [z] = \theta(z),$$

which confirms $(x,z) \in R^{\odot}$.

Now assume $(x,z) \in T_2$. Then there is an $\ell \in \{0,1\}$ and a $p \in Q$ such that $x \in I'_{q_0,p,\ell}$ and $z \in W_{p,f,1-\ell}$. The former yields an element $r \in R_{q_0,p}$ with $\theta(x)r = \mathbf{1}$ and the latter implies $\theta(z) \in R_{p,f,1-\ell}$. Since A is saturated, we can therefore pick reduced words $v \in \mathcal{L}_{q_0,p}(A)$ and $w \in \mathcal{L}_{p,f}(A)$ such that $[v] = r$ and $[w] = \theta(z)$. Then we have $vw \in \mathcal{L}(A)$ and thus $[vw] \in R$. Therefore,

$$\theta(x)[vw] = (\theta(x)r)\theta(z) = \theta(z)$$

and thus $(x,z) \in R^{\odot}$.

Suppose $(x,z) \in T_3$. Then we can decompose $x = zx'$ so that $x' \in I_{q_0,f,1-\ell}$. The latter means that there is an $r \in R$ with $\theta(x')r = \mathbf{1}$. Moreover, the last block of z and the first block of x' are of different types; or one of the words is empty. This implies $\theta(zx') = \theta(z)\theta(x')$ and hence

$$\theta(x)r = \theta(zx')r = \theta(z)\theta(x')r = \theta(z)$$

meaning $(x,z) \in R^{\odot}$.

Finally, let $(x,z) \in T_4$. Then there are $\ell \in \{0,1\}$, $p, q \in Q$, and words $x_1, x_2, x_3, z_1, z_2, z_3 \in Y^*$ such that $x = x_1x_2x_3$, $z = z_1z_2z_3$, $x_1 = z_1$, $(x_2, z_2) \in R^{\odot}_{p,q}$, $x_3 \in R^{\triangleleft}_{q_0,p}$, and $z_3 \in R^{\triangleright}_{q,f}$. Moreover, x_2, z_2 are ℓ-blocks and we know that x_i and x_{i+1} as well as z_i and z_{i+1} are non-merging for $i \in \{1,2\}$ and that $\theta(x_2), \theta(z_2) \in M_\ell$.

We want to construct an $r \in R$ with $\theta(x)r = \theta(z)$. We shall construct an encoding for r in the form $y_1y_2z_3$ such that in this decomposition, each factor is non-merging with the next. Since

$x_3 \in R^{\triangleleft}_{q_0,p}$, there is a $y_1 \in R^{\triangleright}_{q_0,p}$ with $\theta(x_3)\theta(y_1) = \mathbf{1}$. Observe that x_3 is either empty or begins with a $(1-\ell)$-block. Hence, y_1 is either empty or ends with an $1-\ell$-block. Since $(x_2, z_2) \in R^{\odot}_{p,q}$, we can find a $y_2 \in R^{\triangleright}_{p,q}$ with $\theta(x_2)\theta(y_2) = \theta(z_2)$. This means $\theta(y_2) \in M_\ell$, so that y_1 and y_2 are non-merging. Moreover, z_3 is empty or begins with a $(1-\ell)$-block, so that also y_2 and z_3 are non-merging.

We have seen that $y_1 \in R^{\triangleright}_{q_0,p}$, $y_2 \in R^{\triangleright}_{p,q}$, and $z_3 \in R^{\triangleright}_{q,f}$. Together with the non-merging relationships, this implies

$$\theta(y_1y_2z_3) = \theta(y_1)\theta(y_2)\theta(z_3) \in R$$

and furthermore

$$\begin{aligned}
\theta(x)\theta(y) &= \theta(x_1x_2x_3)\theta(y_1y_2z_3) \\
&= \theta(x_1)\theta(x_2)\theta(x_3)\theta(y_1)\theta(y_2)\theta(z_3) \\
&= \theta(x_1)\theta(x_2)\theta(y_2)\theta(z_3) \\
&= \theta(x_1)\theta(z_2)\theta(z_3) \\
&= \theta(z_1)\theta(z_2)\theta(z_3) \\
&= \theta(z),
\end{aligned}$$

confirming $(x, z) \in R^{\odot}$. This proves $R^{\odot} = T_1 \cup T_2 \cup T_3 \cup T_4$.

E. Correctness of squaring gadget

Let $c_1 = (m_1, 2n-1, 0, 0)$, as guaranteed by $\mathsf{init}(c, c')$, $\mathsf{step}(c_1, c_2)$, and $\mathsf{step}(c_2, c')$. Any run going from c_1 to a q-configuration passing through an m_3-configuration will be of the following shape: After a first transition $c_1 \to (m_3, 2n-1, 0, 0)$, there will be a number of executions of the m_3-m_5-loop. At the i-th execution, the self loops at m_3 and m_5 are fired k_i and h_i times, respectively. Let us write the configurations before the i-th m_3-m_5-loop as (m_3, x_i, y_i, z_i). The cumulative effect of an execution of the m_3-m_5-loop is then $(\bar{\mathsf{c}}_1\mathsf{c}_2\mathsf{c}_3)^{k_i}\bar{\mathsf{c}}_2\bar{\mathsf{c}}_2(\bar{\mathsf{c}}_2\mathsf{c}_1)^{h_i}$, with $k_i \le x_i$ and $h_i \le y_i + k_i - 2$. We obtain that $x_{i+1} + y_{i+1} = x_i + y_i - 2$, so $x_{i+1} \le x_i + y_i - 2$. The consequence of this estimation is that (i) the maximum number of iterations is given by $\lfloor (x_0 + y_0)/2 \rfloor$ and (ii) at each iteration we add k_i (so, at most x_i) to z_i. Hence, starting from $(m_3, 2n-1, 0, 0)$ ($x_0 = 2n-1$, $y_0 = 0$, $z_0 = 0$, and thus $\lfloor (x_0 + y_0)/2 \rfloor = n-1$) we can reach precisely those q-configurations d' where $\pi_3(d')$ is any number between 0 and n^2.