Logical Methods in Computer Science
Vol. 3 (4:5) 2007, pp. 1–1–18
www.lmcs-online.org

Submitted Dec. 23, 2004
Published Nov. 5, 2007

MODEL CHECKING SYNCHRONIZED PRODUCTS OF INFINITE TRANSITION SYSTEMS *

STEFAN WÖHRLE AND WOLFGANG THOMAS

Informatik 7, RWTH Aachen, 52056 Aachen, Germany
e-mail address: {woehrle,thomas}@informatik.rwth-aachen.de

Abstract. Formal verification using the model checking paradigm has to deal with two aspects: The system models are structured, often as products of components, and the specification logic has to be expressive enough to allow the formalization of reachability properties. The present paper is a study on what can be achieved for infinite transition systems under these premises. As models we consider products of infinite transition systems with different synchronization constraints. We introduce finitely synchronized transition systems, i.e. product systems which contain only finitely many (parameterized) synchronized transitions, and show that the decidability of FO(R), first-order logic extended by reachability predicates, of the product system can be reduced to the decidability of FO(R) of the components. This result is optimal in the following sense: (1) If we allow semifinite synchronization, i.e. just in one component infinitely many transitions are synchronized, the FO(R)-theory of the product system is in general undecidable. (2) We cannot extend the expressive power of the logic under consideration. Already a weak extension of first-order logic with transitive closure, where we restrict the transitive closure operators to arity one and nesting depth two, is undecidable for an asynchronous (and hence finitely synchronized) product, namely for the infinite grid.

1. Introduction

In the theory of algorithmic verification, a standard framework for modeling systems is given by finite transition systems (often in the form of Kripke structures). Much effort is presently spent on extending this framework to cover infinite transition systems, and to deal adequately with the internal structure of the systems under consideration, such as their composition from several components. The present paper is a study on the scope of algorithmic model checking over transition systems that are composed from infinite components as products with various constraints on the synchronization of their transitions.

We consider transition graphs in the format $G = (V, (E_a)_{a\in\Sigma})$ where V is the set of states (or vertices) and $E_a \subseteq V \times V$ the set of a-labeled transitions. The direct product of two transition graphs has an a-labeled transition from (p, q) to (p', q') if there are such transitions from p to p' and from q to q'. This is the case of complete synchronization. The

2000 ACM Subject Classification: F.4.1.
Key words and phrases: Model checking, synchronized products, reachability, transitive closure logic.
* A preliminary version of the paper appeared in 19th IEEE Symposium on Logic in Computer Science, Turku, July 2004 [WT04].

DOI:10.2168/LMCS-3 (4:5) 2007

other extreme is the asynchronous product, where a transition in one component does not affect the other components. A main result below deals with the "intermediate" case where the component graphs are infinite and in each component only finitely many transitions are used for synchronization. We call these product structures "finitely synchronized". They arise whenever the local computations in the components involve infinite state-spaces but synchronization is restricted to a finite number of actions in each component.

We study the model checking problem for products of transition graphs with respect to several logics that are extensions of first-order logic FO. A basic requirement in verification is that reachability properties should be expressible. There are numerous ways to extend FO by features that allow to express reachability properties. We consider here four extensions that cover reachability relations, listed in the order of increasing expressiveness:

- Reachability logic FO(R), which is obtained from FO-logic by adjoining transitive closure operators Reach_Γ over subsets Γ of edge relations.
- FO(Reg) as a generalization of FO(R) in which path labels have to match a given regular expression.
- Transitive closure logic over binary relations, which allows to proceed from any definable relation (and not just from some edge relations) to its transitive closure.
- Monadic second-order logic MSO, which results from FO-logic by adjoining variables and quantifiers for sets (and in which transitive closure over binary relations can be expressed).

The purpose of this paper is to analyze for which types of products and for which of these logics $\mathcal{L}$ the decidability of the model checking problem for a product can be inferred from the decidability of the corresponding model checking problem for the components. In other words, we analyze for which kinds of products the decidability of the $\mathcal{L}$-theory of the product can be derived from the decidability of the $\mathcal{L}$-theories of the components.

Our first result is such a transfer result for the logic FO(R) over finitely synchronized products of transition graphs. For this, we use a technique of "composition" which resembles the method of Feferman and Vaught [FV59] in first-order model theory (see [CK73], [Hod93] for introductions and [Mak04] for a comprehensive survey). The Feferman-Vaught method (applied to FO) allows to determine the FO-theory of a product structure (e.g., a direct product) from the FO-theories of the components and some additional information on the index structure. Our proof involves a more detailed semantic analysis of the components, thereby exploiting the assumption on finite synchronization. The result extends a theorem of Rabinovich [Rab07] on propositional modal logic extended by the modality EF over asynchronous products.

We show that our result is optimal in two ways.

Firstly, the result does not extend to a case where we allow a slight liberalization of the constraint on finite synchronization: We consider "semi-finite synchronization", in which all components except one can synchronize via finitely many transitions. In the presence of a single component with infinitely many synchronizing transitions we may obtain a structure with undecidable FO(R) model checking problem, whereas the problem is decidable for the components individually.

Secondly, we investigate whether the logic FO(R) can be extended in the above mentioned preservation result. For a strong extension like MSO it is clear that decidability of the component theories does not carry over to the theory of the product system. As is well-known, we may work with the asynchronous product of the successor structure of the natural numbers, which is the infinite $(\omega \times \omega)$-grid. (Note that the asynchronous product is finitely synchronized with an empty set of synchronizing transitions.) The grid has an undecidable monadic theory, whereas the component structures have decidable monadic theories.

We clarify the situation for weaker extensions of FO(R), namely FO(Reg) and transitive closure logic. We show that asynchronous products do not preserve the decidability of the FO(Reg)-theory. For transitive closure logic this undecidability result can already be obtained for a very simple example of an asynchronous product, namely the infinite grid as considered above. Moreover, we show that this undecidability phenomenon only appears when the TC-operator is nested. For the fragment of transitive closure logic with unnested TC-operators interpreted over the infinite grid, we obtain a reduction to Presburger arithmetic and hence the decidability of the corresponding theory.

These undecidability results complement a theorem of Rabinovich [Rab07] where the corresponding fact is shown for propositional modal logic extended by the modality EG over finite grids.

In our results the component structures are assumed to have a decidable theory in one of the logics considered above. Let us summarize some of the relevant classes and their closure properties with respect to synchronization.

A fundamental result is that pushdown graphs have a decidable monadic second-order theory [MS85]. Since then several extensions like prefix recognizable graphs [Cau96] or Caucal graphs [Cau02] have been considered, see [Tho03] for an overview. These classes form an increasing sequence in this order, and all of them enjoy a decidable MSO-theory. None of these classes is closed under asynchronous products.

Two other classes of infinite graphs we like to mention are the graphs of ground term rewriting systems [Col02] for which the FO(R)-theory is decidable, and ground tree rewriting systems [Löd02] for which a temporal logic with reachability and recurrence operators is decidable. Both classes are closed under asynchronous products.

Classes which are closed under synchronized products are rational graphs [Mor00], graphs of Thue specifications [Pay00], or graphs of linear bounded machines [KP99]. However for all these classes already the FO-theory is undecidable and hence they are not suitable for model checking purposes.

The paper is organized as follows. In Section 2 we give the definition of a synchronized product of a family of graphs or transition systems, recall the definition of transitive closure logic, and define FO(R) and FO(Reg).

In Section 3 we show the composition theorem for finitely synchronized products and reachability logic and prove that this result cannot be extended to FO(Reg) or semifinite synchronization in general.

In Section 4 we investigate transitive closure logic over the infinite grid. We show that if we allow transitive closure operators of arity one without parameters but of nesting depth two the theory of the grid is undecidable. On the other hand we show that if no nesting of transitive closure operators is allowed, the respective theory is decidable even in presence of parameters in the scope of the transitive closure operators.

2. Preliminaries

Let $(V_i)_{1\le i\le n}$ be a family of sets. We denote by $\times_{1\le i\le n} V_i$ the Cartesian product of these sets. Tuples $(v_1,\dots,v_n)\in \times_{1\le i\le n} V_i$ are usually denoted by $\bar{v}$, and the ith component of $\bar{v}$ as v_i .

Let Σ be a finite set of labels. A *transition system* is a Σ-labeled directed graph $G=(V^G,(E^G_a)_{a\in\Sigma})$ where V^G is the set of vertices of G and $E^G_a \subseteq V^G\times V^G$ denotes the set of a-labeled edges in G.

2.1. **Synchronized Products.** For $1\le i\le n$ let $G_i := (V_i,(E^i_a)_{a\in\Sigma_i})$ be a Σ_i-labeled graph. We assume that Σ_i is partitioned into a set Σ^l_i of *local* labels (or actions) and a set Σ^s_i of *synchronizing* labels, and to avoid notational complication we require the sets of local labels to be pairwise disjoint. An asynchronous transition labeled by $a\in\Sigma^l_i$ is applied only in the i-th component of a state $(v_1,\dots,v_n)$ of the product graph while the other components stay fixed. For synchronizing transitions we distinguish explicitly between the components where a joint change of states is issued and the components where the state does not change. To describe the latter, define $E^i_\varepsilon := \{(v,v)\mid v\in V_i\}$ and $\tilde{\Sigma}^s_i := \Sigma^s_i\cup\{\varepsilon\}$. A *synchronization constraint* is a set $C\subseteq \times_{1\le i\le n}\tilde{\Sigma}^s_i$. If $\bar{c}\in C$, a $\bar{c}$-labeled transtition induces a simultaneous change in the components i where $c_i\neq\varepsilon$ while the states do not change in the other components.

Formally, the *synchronized product* of $(G_i)_{1\le i\le n}$ defined by C is the graph G with vertex set $V := \times_{1\le i\le n} V_i$, asynchronous transitions with labels $a\in\bigcup_{1\le i\le n}\Sigma^l_i$ defined by $E^G_a\bar{v}\bar{w}$ if $E^i_a v_i w_i$ and $v_j=w_j$ for $j\neq i$, and synchronized transitions with labels $\bar{c}\in C$ defined by $E^G_{\bar{c}}\bar{v}\bar{w}$ if $E^i_{c_i}v_iw_i$ for every $1\le i\le n$. We denote the set of local transitions labels $\bigcup_{1\le i\le n}\Sigma^l_i$ of G by Σ^l, and the set $C\cup\Sigma^l$ of all transition labels by Σ. A product is asynchronous if $C=\emptyset$.

Note that we slightly deviate from the definition in [Arn94] since we require the sets of local labels and synchronizing labels to be disjoint, and implicitly assume an asynchronous behavior of local transitions.

Let $(G_i)_{1\le i\le n}$ be a family of graphs and $C\subseteq\times_{1\le i\le n}\tilde{\Sigma}^s_i$ be a synchronization constraint. For $\bar{c}\in C$ let $X_{\bar{c}} := \{i\mid c_i\neq\varepsilon\}$. For $C'\subseteq C$ we write $X_{C'}=\bigcup_{\bar{c}\in C'}X_{\bar{c}}$. Define

$$\bar{u}\sim_{\bar{c}}\bar{v} :\Leftrightarrow u[X_{\bar{c}}]=v[X_{\bar{c}}],$$

i.e. $\bar{u}\sim_{\bar{c}}\bar{v}$ if $\bar{u}$ and $\bar{v}$ agree on the synchronizing components. The synchronized product G of $(G_i)_{1\le i\le n}$ defined by C is called *finitely synchronized* if index$(\sim_{\bar{c}})$, i.e. the number of equivalence classes of $\sim_{\bar{c}}$, is finite for every $\bar{c}\in C$. In the conference version [WT04] of this paper, finitely synchronized products involve only finitely many individual synchronizing transitions, thus disallowing the label ε in the synchronization constraint. In the present treatment we allow finitely many parametrized synchronized transitions: The inclusion of constraints $\bar{c}$ with $c_i=\varepsilon$ means that in the i-th component the transition $\bar{c}$ applies to arbitrary states of V_i and hence possibly infinitely many individual synchronizing transitions may be present in a finitely synchronized product[1].

[1]Thus, the proof of Theorem 3.1 below involves more technicalities than the corresponding proof in [WT04].

We collect some technical preparations in the subsequent Lemma 2.1. For this we define for every $\emptyset \neq C' \subseteq C$ the eqivalence relation

$$\bar{u} \sim_{\bar{C}'} \bar{v} :\Leftrightarrow \bar{u} \sim_{\bar{c}} \bar{v} \text{ for every } \bar{c} \in C'$$

and restrict the relation $\sim_{C'}$ to the set of vertices of the synchronized product from which an outgoing transition exists for every $\bar{c} \in C'$, i.e. to the set

$$V_{C'} := \{\bar{u} \in \times_{1 \leq i \leq n} V_i \mid \forall \bar{c} \in C' \; \exists \bar{v} \text{ such that } (\bar{u}, \bar{v}) \in E_{\bar{c}}\}.$$

Lemma 2.1. *Let $(G_i)_{1 \leq i \leq n}$ be a family of graphs and $C \subseteq \times_{1 \leq i \leq n} \tilde{\Sigma}_i^s$ be a synchronization constraint.*

(a) If G is finitely synchronized, then $\operatorname{index}(\sim_{C'})$ is finite for every $\emptyset \neq C' \subseteq C$.
(b) For every subset $\emptyset \neq C' \subseteq C$, if $u \sim_{C'} v$ and $G \models \operatorname{Reach}_{\Sigma^l \cup C'}[u, w]$ there exists a w' such that $G \models \operatorname{Reach}_{\Sigma^l \cup C'}[v, w']$ and $w \sim_{C'} w'$.
(c) Let $\Gamma \subseteq \Sigma^l \cup C'$. If $G \models \operatorname{Reach}_\Gamma[u, v]$, $G \models \operatorname{Reach}_\Gamma[v, w]$ and $u \sim_{C'} v$ then $G \models \operatorname{Reach}_\Gamma[u, w]$ and the path from u to w can be chosen such that no intermediate vertex is $\sim_{C'}$-equivalent to u.

Proof. (a) If G is finitely synchronized, then $\operatorname{index}(\sim_{\bar{c}})$ is finite for every $\bar{c} \in C$. If $C' \subseteq C'' \subseteq C$ then C'' refines C' on $V_{C''} \subseteq V_{C'}$. Therefore, for every $C' \subseteq C$ the number of equivalence classes of $\sim_{C'}$ is bounded by $\times_{\bar{c} \in C} \operatorname{index}(\sim_{\bar{c}})$.

(c) is a direct consequence of (b) which remains to be shown. Let $u \sim_{C'} v$ and $G \models \operatorname{Reach}_\Gamma[u, w]$. Since transitions labeled with symbols from $\bigcup_{i \notin X_{C'}} \Sigma_i^l$ commute with transitions labeled by symbols from $\bigcup_{i \in X_{C'}} \Sigma_i^l \cup C'$ we may w.l.o.g. assume that the path from u to w is of the form

$$u = u_1 \xrightarrow{a_1} u_1 \xrightarrow{a_2} \ldots \xrightarrow{a_{m-1}} u_m = u_1' \xrightarrow{b_1} u_2' \xrightarrow{b_2} \ldots \xrightarrow{b_{n-1}} u_n' = w$$

and $a_j \in \bigcup_{i \notin X_{C'}} \Sigma_i^l$ for $1 \leq j \leq m$ and $b_j \in \bigcup_{i \in X_{C'}} \Sigma_i^l \cup C'$ for $1 \leq j \leq n$. Hence by definition of $\sim_{C'}$ we have $u[X_{C'}] = u_1'[X_{C'}] = v[X_{C'}]$. Thus there is a path $v' = v_1 \xrightarrow{b_1} v_2 \xrightarrow{b_2} \ldots \xrightarrow{n-1} v_n = w'$ in G and $w \sim_{C'} w'$. ☐

2.2. **First-Order Logic and Extensions.** We assume that the reader is familiar with first-order logic FO over graphs. We denote formulas by $\varphi(x_1, \ldots, x_n)$ to express that the free variables of φ are among $x_1, \ldots, x_n$. If G is a graph and $v_1, \ldots, v_n$ are the vertices assigned to the variables $x_1, \ldots, x_n$, we denote by $(G, v_1, \ldots, v_n) \models \varphi(x_1, \ldots, x_n)$ or shortly by $G \models \varphi[v_1, \ldots, v_n]$ that the formula φ is satisfied in G under the respective variable assignment.

Transitive closure logic FO(TC) is defined by extending FO with formulas of the type

$$\psi := [\mathrm{TC}_{\bar{x}, \bar{y}}\, \varphi(\bar{x}, \bar{y}, \bar{z})]\, \bar{s}, \bar{t}$$

where $\varphi(\bar{x}, \bar{y}, \bar{z})$ is a FO(TC)-formula, $\bar{x}, \bar{y}$ are disjoint tuples of free variables of the same length $k > 0$, $\bar{s}, \bar{t}$ are tuples of variables of length k and $\operatorname{free}(\psi) := (\operatorname{free}(\varphi) \setminus \{\bar{x}, \bar{y}\}) \cup \{\bar{s}, \bar{t}\}$. Note that in the notation $[\mathrm{TC}_{\bar{x}, \bar{y}}\, \varphi(\bar{x}, \bar{y}, \bar{z})]\, \bar{x}, \bar{y}$ the variables inside the square brackets are bound while the variables at the end of the formula occur free.

Let G be a graph, let $\bar{c}$, $\bar{d}$, and $\bar{e}$ be the interpretations of the variables $\bar{z}$, $\bar{s}$, and $\bar{t}$ in φ. Let E be the relation on k-tuples defined by $E(\bar{c}) := \{(\bar{a}, \bar{b}) \mid (G, \bar{a}, \bar{b}, \bar{c}) \models \varphi(\bar{x}, \bar{y}, \bar{z})\}$, and $E'(\bar{c})$ be its transitive closure, i.e. $(\bar{a}, \bar{b}) \in E'(\bar{c})$ iff there exists a sequence $\bar{f}_0, \bar{f}_1, \ldots, \bar{f}_l$

such that $\bar{f}_0 = \bar{a}$, $(\bar{f}_i, \bar{f}_{i+1}) \in E(\bar{c})$ for $1 \leq i < l$, and $\bar{f}_l = \bar{b}$. The semantics of the FO(TC)-formula above is defined by

$$(G, \bar{c}, \bar{d}, \bar{e}) \models [\mathrm{TC}_{\bar{x},\bar{y}}\, \varphi(\bar{x}, \bar{y}, \bar{z})]\, \bar{s}, \bar{t} \Leftrightarrow (\bar{d}, \bar{e}) \in E'(\bar{c}).$$

We call the variables $\bar{z}$ parameters for the transitive closure operator. By $\mathrm{FO(TC)}_{(k)}$ be denote the fragment of FO(TC) where the transitive closure operation is only allowed to define relations over tuples of length $\leq k$, i.e. the length of the tuples $\bar{x}, \bar{y}$ in the definition above is bounded by k. For example, in $\mathrm{FO(TC)}_{(1)}$ we can only define binary relations using a transitive closure operator. For finite models the arity hierarchy $(\mathrm{FO(TC)}_{(k)})_{k \geq 0}$ is strict [Gro96].

By $\mathrm{FO(TC)}^{l}_{(k)}$ we denote the fragment of $\mathrm{FO(TC)}_{(k)}$ where the nesting depth of transitive closure operations is bounded by l.

In transitive closure logic we can express that from a vertex x a vertex y is reachable via a path with labels from some set $\Sigma' \subseteq \Sigma$ by

$$\mathrm{Reach}_{\Sigma'}(x, y) := \Big[\mathrm{TC}_{x,y}\, \big(x = y \vee \bigvee_{a \in \Sigma'} E_a xy\big)\Big] x, y.$$

We call the restriction of FO(TC) where the only transitive closure formulas allowed are of the form $\mathrm{Reach}_{\Sigma'}(x, y)$ for $\Sigma' \subseteq \Sigma$ *reachability logic* and denote it by FO(R).

The expressive power of the reachability predicates in FO(R) is limited, e.g. we cannot express that there is a path between vertex v and w in the graph whose labels form a word in a given regular language.

We denote by FO(Reg) first-oder logic extended by reachability predicates $\mathrm{Reach}_r(x, y)$ for regular expressions r over Σ, where $G \models \mathrm{Reach}_r[v, w]$ if there is a path in G from v to w labeled by a word contained in the language described by r.

3. Synchronization and FO(R)

In this section we show that synchronization preserves the decidability of the FO(R)-theory if (and only if) the product is finitely synchronized. For this case we prove a composition theorem that reduces the evaluation of a formula in the product graph to the evaluation of several formulas in the component graphs and a Boolean combination of these truth values. This result does not extend to the case of FO(Reg).

Furthermore we show that *semifinite* synchronization of two components, where in just one of the components infinitely many edges are allowed to be synchronized, does in general not preserve the decidability of the FO(R)-theory.

Theorem 3.1. *Let G be a finitely synchronized product of a family $(G_i)_{1 \leq i \leq n}$ of graphs with decidable FO(R)-theories. Then the FO(R)-theory of G is also decidable, and for an FO(R)-formula φ we can effectively construct sets of formulas Ψ_i and a Boolean formula α such that $G \models \varphi$ iff α is true under an Boolean interpretation defined by the truth values of the formulas in Ψ_i.*

Proof. Let $(G_i)_{1 \leq i \leq n}$ be a family of graphs whose signatures $\Sigma_i := \Sigma_i^l \cup \Sigma_i^s$ are partitioned into local and synchronizing labels. Let $C \subseteq \bigtimes_{1 \leq i \leq n} \tilde{\Sigma}_i^s$ be a synchronization constraint such that the product G of $(G_i)_{1 \leq i \leq n}$ is finitely synchronized with respect to C.

We show by induction that for every FO(R)-formula over Σ there are finite sets Ψ_i of Σ_i-formulas and a Boolean formula α over predicates $p_i(\psi^i_j)$ $(1 \le i \le n,\ 1 \le j \le |\Psi_i|)$ such that

$$(G, \bar{v}_1, \dots \bar{v}_m) \models \varphi(x_1, \dots, x_m) \Leftrightarrow I(\bar{v}_1, \dots, \bar{v}_m) \models \alpha \tag{3.1}$$

where $I(\bar{v}_1, \dots, \bar{v}_m)$ is the Boolean interpretation defined by

$$I(\bar{v}_1, \dots, \bar{v}_m)(p_i(\psi_j)) = \begin{cases} \text{true} & \text{if } (G_i, v^i_1, \dots, v^i_m) \models \psi^i_j \\ \text{false} & \text{otherwise.} \end{cases}$$

We start with the atomic formulas. For $x = y$ let $\psi_i := (x = y)$, for $E_a xy$ with $a \in \Sigma^l_i$ let $\psi_i := E_a xy$ and $\psi_j := (x = y)$ for $i \neq j$, and for $E_{\bar{c}} xy$ with $\bar{c} \in C$ let $\psi_i := E_{c_i} xy$. For every formula above let $\alpha := \bigwedge_{1 \le i \le n} p_i(\psi_i)$. Obviously (3.1) holds in all cases, so the remaining "atomic" formulas we have to take care of are of the form $\text{Reach}_\Gamma(x, y)$ for $\Gamma \subseteq \Sigma$.

For this part of the proof we proceed by induction on the number of synchronizing transitions from C which appear in Γ. We may assume that Γ comprises all local transition labels, i.e. that $\Sigma^l \subseteq \Gamma$; otherwise in the following every occurrence of Σ^l_i has to be replaced by $\Sigma^l_i \cup \Gamma$.

We first consider the case that there is only a single synchronizing transition $\bar{c} \in \Gamma$. By the definition of finitely synchronized product we know that $\text{index}(\sim_{\bar{c}})$ is finite, and by Lemma 2.1 (c) that we have to pass through every equivalence class at most once. Let $k = \text{index}(\sim_{\bar{c}})$. For $i \in X_{\bar{c}}$ and $1 \le m \le k$ define

$$\psi^i_{(\bar{c},m)}(x, y) := \exists z_1 \dots \exists z_m \Big(\text{Reach}_{\Sigma^l_i}(x, z_1) \wedge y = z_m \\ \wedge \bigwedge_{1 \le i < m} \exists w \big(E_{c_i} z_i w \wedge \text{Reach}_{\Sigma^l_i}(w, z_{i+1}) \big) \Big)$$

which expresses that on a path from x to y in component i exactly m vertices $z_1, \dots, z_m$ are passed from which a synchronized transition is possible. For $i \notin X_{\bar{c}}$ we set

$$\psi^i_{\bar{c},m}(x, y) := \text{Reach}_{\Sigma^l_i}(x, y)$$

and define $\Psi_i(\bar{c}) := \{\psi^i_{(\bar{c},m)}(x, y) \mid 1 \le m \le k\}$. Setting

$$\alpha(\bar{c}) := \bigvee_{1 \le m \le k} \bigwedge_{1 \le i \le n} p(\psi^i_{\bar{c},m})$$

ensures (3.1) for sets Γ which contain at most one synchronizing edge label $\bar{c}$.

Let now $C' = C \cap \Gamma$. By the induction hypothesis we may assume that for every subset $C'' \subset C'$ there are families of formulas $\Psi_i(C'') := \{\psi^i_{(C'',m)}(x, y) \mid m \le m(C'')\}$ and Boolean formulas $\alpha(C'')$ such that (3.1) holds, i.e.

$$G \models \text{Reach}_{C'' \cup \Sigma^l}[\bar{v}, \bar{w}] \Leftrightarrow I(\bar{v}, \bar{w}) \models \alpha(C''). \tag{3.2}$$

Let $k := \text{index}(\sim_{C'})$ and $l := \sum_{C'' \subset C'} \text{index}(\sim_{C''})$, for $1 \le r \le k$ let σ_1 be a mapping $\sigma_1 : \{1, \dots, r\} \to \{1, \dots, l\}$ and σ_2 a mapping $\sigma_2 : \{1, \dots, l\} \to \{(C'', s) \mid C'' \subset C',\ s \le m(C'')\}$. The number of vertices in $V_{C'}$ which are passed on the path from vertex $\bar{u}$ to $\bar{w}$ is r. The mapping σ_1 then determines the number of $\sim_{C''}$ equivalence classes which are passed on the path between consecutive vertices in $V_{C'}$ and σ_2 determines the order in which vertices from $\sim_{C''}$ eqivalence classes appear.

Let $\pi_1, \ldots, \pi_t$ be an enumeration of all mappings which can be obtained by composing the mappings σ_1 and σ_2. We define for $t' \leq t$, $\pi_{t'} := \sigma_2 \circ \sigma_1$ with σ_j as above and $1 \leq i \leq n$ the formula

$$\begin{aligned}\psi^i_{(C',t')}(x,y) := \exists y_1 \ldots y_r \quad & \Big[y_1 = x \wedge y_r = y \\ & \wedge \textstyle\bigwedge_{1\leq p<r} \Big(\exists z_1 \ldots z_{\sigma_1(p)}\big(z_1 = y_p \wedge z_{\sigma_1(p)} = y_{p+1} \\ & \wedge \textstyle\bigwedge_{1\leq q<\sigma_1(p)} \psi^i_{\sigma_2(q)}(z_q, z_{q+1})\big)\Big)\Big].\end{aligned}$$

The Boolean formula $\alpha(C')$ is then defined to be

$$\alpha(C') := \bigvee_{1\leq t'\leq t} \bigwedge_{1\leq i\leq n} p(\psi^i_{(C',t')}).$$

We claim now that for every $C' \subseteq C$

$$G \models \mathrm{Reach}_{\Sigma^l \cup C'}[\bar{u}, \bar{v}] \Leftrightarrow I(\bar{u}, \bar{v}) \models \alpha(C'). \tag{3.3}$$

We first consider the direction from right to left. Let $I(\bar{u}, \bar{v}) \models \alpha(C')$. The case $C' = \{\bar{c}\}$ has already been dealt with above. So assume that (3.3) holds for every $C'' \subset C'$. Then $I(\bar{u}, \bar{v}) \models \bigwedge_{1\leq i\leq n} p(\psi^i_{C',t'})$ for some t', i.e there exits an r and mappings $\sigma_1 : \{1, \ldots, r\} \to \{1, \ldots, l\}$ and $\sigma_2 : \{1, \ldots, l\} \to \{(C'', s) \mid C'' \subset C',\ s \leq m(C'')\}$ such that for $1 \leq i \leq n$

$$\begin{aligned}(G, u_i, v_i) \models \exists y_1 \ldots y_r \quad & \Big[y_1 = x \wedge y_r = y \\ & \wedge \textstyle\bigwedge_{1\leq p<r} \Big(\exists z_1 \ldots z_{\sigma_1(p)}\big(z_1 = y_p \wedge z_{\sigma_1(p)} = y_{p+1} \\ & \wedge \textstyle\bigwedge_{1\leq q<\sigma_1(p)} \psi^i_{\sigma_2(q)}(z_q, z_{q+1})\big)\Big)\Big].\end{aligned}$$

If we denote the the valuation of the variables z_j (respectively y_j) in G_i which make the formula above true by z^i_j (respectively y^i_j) and their n-tuple by $\bar{z}_j$ (respectively $\bar{y}_j$) we obtain that $I(\bar{z}_j, \bar{z}_{j+1}) \models \alpha(\sigma_2(q)_1)$ for $1 \leq j < \sigma_1(p)$ (here $\sigma_2(q)_1$ denotes the first component of $\sigma_2(q)$). Hence $G \models \mathrm{Reach}_{\Sigma^l \cup \sigma_2(q)_1}[\bar{z}_j, \bar{z}_{j+1}]$ for $1 \leq j \leq \sigma_1(p)$ and since $\bigcup_{1\leq q\leq\sigma_1(p)} \sigma_2(q)_1 \subseteq C'$ also $G \models \mathrm{Reach}_{\Sigma^l \cup C'}[\bar{y}_j, \bar{y}_{j+1}]$ for $1 \leq j \leq r$. Hence we obtain $G \models \mathrm{Reach}_{\Sigma^l \cup C'}[\bar{u}, \bar{v}]$.

For the direction from left to right suppose that $G \models \mathrm{Reach}_{\Sigma^l \cup C'}[\bar{u}, \bar{v}]$. By Lemma 2.1 (c) we know that there is a path from $\bar{u}$ to $\bar{v}$ in G which passes every $\sim_{C'}$ equivalence class ot most once. Let $\bar{y}_1, \ldots, \bar{y}_r$ be the sequence of these vertices from $V_{C'}$ on the path. We now consider for $1 \leq j < r$ the path segments between $\bar{y}_j$ and $\bar{y}_{j+1}$. Every such path segment can be further decomposed in the following way: Let $\bar{z}_1$ be the first vertex in the segment which is contained in some $V_{C''}$ for $\emptyset \neq C'' \subset C'$. If there is no such $\bar{z}_1$ only local labels can appear on the path from $\bar{y}_j$ to $\bar{y}_{j+1}$. In this case choose $\bar{z}_1 := \bar{y}_{j+1}$.

Then we choose $\bar{z}_2$ to be the last vertex on the path from $\bar{y}_j$ to $\bar{y}_{j+1}$ such that $G \models \mathrm{Reach}_{\Sigma^l \cup C''}[\bar{z}_1, \bar{z}_2]$, i.e. $z_2 \in V_{C'''}$ for some $C''' \subset C'$ with $C''' \setminus C'' \neq \emptyset$. This decomposition can be continued until $\bar{y}_{j+1}$ is reached.

Figure 1 shows such a decomposition of a path from $\bar{u}$ to $\bar{v}$. Every path segment from $\bar{y}_j$ to $\bar{y}_{j+1}$ is again partintioned as shown. For sake of readability we mention only the set of synchronizing labels allowed on the intermediate paths and write C' for $C' \cup \Sigma^l$.

By Lemma 2.1 (c) we again know that the number of intermediate vertices $\bar{z}$ can be bounded by $l := \sum_{C''\subset C'} \mathrm{index}(\sim_{C''})$. By the induction hypothesis on subsets $C'' \subset C'$ we know that for every pair of successive vertices $\bar{z}_j, \bar{z}_{j+1}$ with $\bar{z}_j \in V_{C''}$ there exists a conjunct

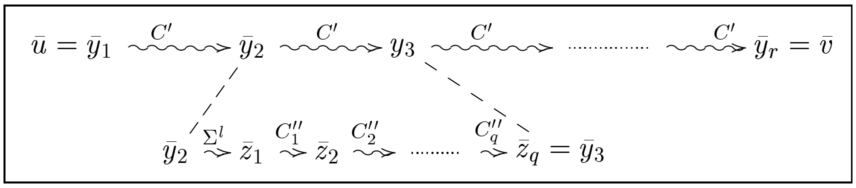

Figure 1: Sample decomposition of a path

of $\alpha(C'')$, i.e. some s such that

$$G \models \mathrm{Reach}_{C'' \cup \Sigma^l}[\bar{z}_j, \bar{z}_{j+1}] \Rightarrow I(\bar{z}_j, \bar{z}_{j+1}) \models \bigwedge_{1 \le i \le n} p(\psi^i_{(C'',s)}).$$

In particular we have $G_i \models \psi^i_{(C'',s)}[z^i_j, z^i_{j+1}]$ for every $1 \le i \le n$ and all inermediate vertices z_j.

Combining these decomposition results we obtain that there exists some r bounded by index$(\sim_{C'})$ (the number of vertices $\bar{y}$), a function $\sigma_1 : \{1, \ldots, r\} \to \{1, \ldots, l\}$ which determines the number of intermediate vertices $\bar{z}$ between the $\bar{y}$ vertices, and a function $\sigma_2 : \{1, \ldots, l\} \to \{(C'', s) \mid C'' \subseteq C'',\ s \le m(C'')\}$ which determines to which $V_{C''}$ an intermediate vertex $\bar{z}_j$ belongs and which conjunct of $\alpha(C'')$ is satisfied by the interpretation induced by $\bar{z}_j$ and $\bar{z}_{j+1}$. Thus we obtain that $G_i \models \psi^i_{(C',s)}[u_i, v_i]$ for $1 \le i \le n$ and some s an hence $I(\bar{u}, \bar{v}) \models \alpha(C')$.

The finishes the proof for atomic formulas. Formulas composed by Boolean connectives and existential quantification are now easy to handle.

The case of Boolean connectives may be solved in the standard way. Let $\varphi_1(\bar{x})$ and $\varphi_2(\bar{y})$ be FO(R)-formulas and α_1, $(\Psi^1_i)_{1 \le i \le n}$ as well as α_2, $(\Psi^2_i)_{1 \le i \le n}$ be given by the induction hypothesis. Then, for $\neg\varphi_1(\bar{x})$ we can choose the same $(\Psi^1_i)_{1 \le i \le n}$ and the Boolean formula to be $\neg\alpha_1$, and for $\varphi_1(\bar{x}) \vee \varphi_2(\bar{y})$ we choose $\Psi_i := \Psi^1_i \cup \Psi^2_i$ and $\alpha = \alpha_1 \vee \alpha_2$.

To finish the proof let $\varphi(x_1, \ldots, x_n) := \exists x_{n+1} \varphi_1(x_1, \ldots, x_{n+1})$. Let Ψ^1_i and α_1 be the formulas computed for $\varphi_1(x_1, \ldots, x_{n+1})$. Let $\mathcal{I}$ be the set of all satisfying assignments for α_1. For every $I \in \mathcal{I}$ let $I_i := \{j \mid I(p^j_i) = \text{true}\}$. Then sets Ψ_i for $1 \le i \le n$ are constructed by adding for every $I \in \mathcal{I}$ the formula

$$\psi^I_i(x_1, \ldots, x_n) := \exists x_{n+1} \Big(\bigwedge_{j \in I_i} \psi^j_i \wedge \bigwedge_{j \notin I_i} \neg\psi^j_i \Big).$$

Then we can define $\alpha := \bigvee_{I \in \mathcal{I}} \bigwedge_{1 \le i \le n} p(\psi^I_i)$. □

For a complexity analysis of this algorithm, note that even in the special case in which the synchronization constraint does not contain ε, the number of formulas which have to be evaluated in the components cannot be bounded by an elementary function. This is due to the exponential increase of the sets Ψ_i which result from dealing with existential quantifiers.

It is easy to see that Theorem 3.1 also covers FO(Reg)-formulas with regular expressions built from Γ^*_i for $\Gamma_i \subseteq \Sigma$ using $\cdot$ and $+$. However, if we allow reachability predicates with regular expressions of the form $(\Gamma_1 \cdot \Gamma_2)^*$ the decidability of the corresponding theory will be lost.

Theorem 3.2. *Asynchronous products do not preserve the decidability of the FO(Reg)-theory.*

Proof. We use a 2-PDA $\mathcal{A}$ (pushdown automaton with two stacks) that simulates a universal Turing machine (cf. [HU79]). Formally a 2-PDA is a tuple $\mathcal{A} = (Q, \Sigma, \Gamma, q_0, \Delta, f)$ where Q is a finite set of states, Σ and Γ the input alphabet, respectively stack alphabet, q_0 is the initial state, f is the final state, and $\Delta \subseteq Q \times \Sigma \times (\Gamma \cup \{\varepsilon\})^2 \times (\Gamma \cup \{\varepsilon\})^2 \times Q$ the transition relation. The configuration with state p and stack contents u, v (discarding the stack bottom symbols) is denoted by (p, u, v) (similarly a pair (p, u) is a configuration of a standard PDA). We assume that Turing machines (as well as 2-PDA's) are normalized, i.e. that each state is reachable from the initial state q_0, the only sink state is the final state f and there are no incoming transitions to q_0.

Input words for the universal 2-PDA $\mathcal{A}$ are of the form $w_1\$w_2\#$ where w_1 is the code of a Turing machine and w_2 an input word for the Turing machine. We assume that $\mathcal{A}$ processes such an input word in two phases: First $w_1\$w_2\#$ is written into the first stack (in reverse order) and then transferred into the second stack (with the first letter of w_1 on top of the stack). With this configuration the second phase starts (and we call its initial state q_0^2), realizing the actual simulation of the universal Turing machine. It is well-known that the reachability problem for $\mathcal{A}$ ("Given $w_1\$w_2\#$ as input, does $\mathcal{A}$ reach the final state?") is undecidable.

To reduce this reachability problem for $\mathcal{A}$ to the model checking problem for FO(Reg) over an asynchronous product of graphs with decidable FO(Reg)-theory, we split $\mathcal{A}$ into two component pushdown automata

$$\mathcal{A}_1 = (Q, \Sigma \times \Delta, \Gamma, q_0, \Delta_1, f)$$
$$\mathcal{A}_2 = (Q, \bar{\Sigma} \times \bar{\Delta}, \Gamma, q_0, \Delta_2, \bar{f})$$

where for every $\delta = (q, a, \gamma_1, \gamma_2, \gamma_3, \gamma_4, p) \in \Delta$ the following transitions are included:

$$(q, (a, \delta), \gamma_1, \gamma_3, p) \text{ to } \Delta_1,$$
$$(q, (\bar{a}, \bar{\delta}), \gamma_2, \gamma_4, p) \text{ to } \Delta_2.$$

Each of the graphs generated by $\mathcal{A}_1$ and $\mathcal{A}_2$ has a decidable MSO-theory and therefore also a decidable FO(Reg)-theory. Let $\mathcal{B}$ their asynchronous product.

Let r be the regular expression

$$r = \Big(\bigvee_{\substack{\delta \in \Delta \\ a \in \Sigma}} (a, \delta)(\bar{a}, \bar{\delta}) \Big)^*$$

which states that a transition of $\mathcal{A}_1$ is followed by the corresponding transition of $\mathcal{A}_2$.

We obtain that

$$\mathcal{B} \models \mathrm{Reach}_r(x, y)[((q, u), (q, v)), ((q', u'), (q'', v'))]$$

iff $q' = q''$ and $\mathcal{A}$ can reach from configuration (q, u, v) the configuration (q', u', v').

It is now easy to construct for every word $w_1\$w_2\#$ a first-order formula $\varphi_{w_1\$w_2\#}(x, y)$ such that

$$\mathcal{B} \models \varphi_{w_1\$w_2\#}(x, y)[((q_0, \varepsilon), (q_0, \varepsilon)), ((q, u_1), (q, u_2))]$$

iff $u_1 = \varepsilon$, $u_2 = w_1\$w_2\#$ and $q = q_0^2$. Then we obtain that

$$\mathcal{B} \models \exists z_1 \exists z_2 \exists z_3 \Big(\varphi_{w_1\$w_2\#}(((q_0,\varepsilon),(q_0,\varepsilon)), z_1) \wedge \mathrm{Reach}_r(z_1,z_2) \wedge \bigvee_{\substack{\delta\in\Delta \\ a\in\Sigma}} \big(E_{(a,\delta)} z_2 z_3 \wedge E_{(\bar{a},\bar{\delta})} z_3 ((f,u),(f,v)) \big) \Big)$$

iff $\mathcal{A}$ reaches a halting configuration after processing $w_1\$w_2\#$. Note that since $\mathcal{A}$ is normalized we can ensure that the initial configuration $((q_0,\varepsilon),(q_0,\varepsilon))$ and all final configurations $((f,u),(f,v))$ are first-order definable. □

We now turn to the proof that semifinite synchronization in general does not preserve the decidability of the FO(R)-theory. We reduce the halting problem of deterministic Turing machines to the model checking problem for FO(R) for synchronized products of finite graphs and infinite graphs which are generated by ground tree rewriting systems (GTRS). The GTRS graphs we will construct are of finite out-degree and hence have a decidable FO(R)-theory [Löd02, Löd03].

The GTRS graph will encode computations of the Turing machine M, but not all of them are valid. We will use the synchronization with a finite graph to eliminate computations which are not valid.

Our construction of the GTRS graph encoding computations of M follows ideas of [Löd03]. Before we start the proof we give a short definition of the Turing machine model we use and of ground tree rewriting systems. For a more detailed description we refer to [HU79] and [Löd03].

A *deterministic Turing machine* is a tuple $M = (Q, \Gamma, q_0, q_f, \delta)$ where Q is a finite set of states, Γ is an alphabet containing a designated blank symbol ␣ , q_0 is the initial state, q_f is the halting state, and $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the transition function. A *configuration* of M is a sequence $a_1, \dots a_k, q, b_l, b_{l-1} \dots b_1$ where $a_i, b_i \in \Gamma$, $q \in Q$ and b_l denotes the symbol currently read by the head of the machine. We consider two configurations to be equivalent if they differ only in heading or trailing blank symbols, and do not distinguish between equivalent configurations.

A *ground tree rewriting system* is a tuple $\mathcal{R} = (A, \Sigma, R, t_0)$ where A is a ranked alphabet, Σ is a set of labels for the rules, R is a finite set of rules, and t_0 is a finite tree over A. We denote the set of all finite trees over A by T_A. A *rewriting rule* r is of the form $t \xrightarrow{b} t'$ with $t, t' \in T_A$ and $b \in \Sigma$. A rule r is applicable to a tree s if there is a subtree s_1 of s equal to t, and the result of an application of r to s is a tree s' obtained from s by replacing s_1 with t'. $\mathcal{R}$ generates a Σ-labeled graph whose vertices are the trees that can be obtained from t_0 by applying rewriting rules from R, with a b-labeled edge between s and s' if s' results from s by an application of a rule of the form $t \xrightarrow{b} t' \in R$.

Theorem 3.3. *Semifinite synchronization does not preserve the decidability of the FO(R)-theory.*

Proof. Let $M = (Q, \Gamma, q_0, q_f, \delta)$ be a deterministic Turing machine. We assume that $q_0 \neq q_f$, $Q \cap \Gamma = \emptyset$, $X \notin Q \cup \Gamma$ and encode a configuration $a_1, \ldots, a_k, q, b_l, b_{l-1}, \ldots b_1$ of M by a tree

$$\begin{array}{ccc} & \bullet & \\ / & & \backslash \\ X & & X \\ \vert & & \vert \\ a_1 & & b_1 \\ \vdots & & \vdots \\ a_k & & b_l \\ & & \vert \\ & & q \end{array}$$

Every transition of the Turing machine will be simulated by the rewriting system in two steps, by first rewriting the right branch of the configuration tree, and then rewriting the left branch. The labels of the rewriting rules will indicate which letter from Γ has to be added $(+)$ or removed $(-)$ from the left branch of the configuration tree, and $\top$ respectively $\perp$ indicate whether the halting state has been reached or not.

More precisely we define a GTRS $\mathcal{R} = (A, \Sigma, R, t_0)$ where $A_2 = \{\bullet\}$, $A_1 = \Gamma \cup \{X\}$, $A_0 = A_1 \cup Q$, $\Sigma = \{+, -\} \times (\Gamma \cup \bar{\Gamma}) \times \{\perp, \top\}$ and

$$t_0 := \begin{array}{ccc} & \bullet & \\ / & & \backslash \\ X & & X \\ & & \vert \\ & & q_0 \end{array} .$$

The set R is defined by adding for $\delta(q, b) = (p, c, L)$ and every $a \in \Gamma$ the rules

$$\begin{array}{c} b \\ \vert \\ q \end{array} \xrightarrow{(-,a,*)} \begin{array}{c} c \\ \vert \\ a \\ \vert \\ p \end{array} \text{ and } \begin{array}{c} X \\ \vert \\ q \end{array} \xrightarrow{(-,a,*)} \begin{array}{c} c \\ \vert \\ a \\ \vert \\ p \end{array} \text{ if } b = \text{␣},$$

and for $\delta(q, b) = (p, c, R)$ and every $a \in \Gamma$ the rules

$$\begin{array}{c} a \\ \vert \\ b \\ \vert \\ q \end{array} \xrightarrow{(+,c,*)} \begin{array}{c} a \\ \vert \\ p \end{array} \text{ and } \begin{array}{c} X \\ \vert \\ q \end{array} \xrightarrow{(+,c,*)} \begin{array}{c} X \\ \vert \\ p \end{array} \text{ if } b = \text{␣}$$

where $* = \top$ if $p = q_f$ and $* = \perp$ otherwise. Note that these rules can only be applied to the right branch of a configuration tree. For the left branch we add for every $a, c \in \Gamma$ and $* \in \{\perp, \top\}$ the rules

$$a \xrightarrow{(-,\bar{a},*)} \varepsilon \text{ and } X \xrightarrow{(-,\bar{a},*)} X \text{ if } a = \text{␣},$$

as well as

$$a \xrightarrow{(+,\bar{c},*)} \begin{array}{c} a \\ \vert \\ c \end{array}$$

and

$$X \xrightarrow{(+,\bar{c},*)} \begin{array}{c} X \\ \vert \\ c \end{array} \text{ if } c \neq \text{␣} \text{ and } X \xrightarrow{(+,\bar{c},*)} X \text{ if } c = \text{␣}.$$

By construction, a path through the graph G generated by $\mathcal{R}$ corresponds to a valid computation of M started on the empty tape iff every transition with label $(+, a, *)$ respectively $(-, a, *)$ is followed by its counterpart labeled $(+, \bar{a}, *)$ respectively $(-, \bar{a}, *)$. Let H be the star graph with $|\Sigma| + 1$ many vertices where the center vertex v has for every $(\$, a, *) \in \{+, -\} \times \Gamma \times \{\perp, \top\}$ a single outgoing edge with this label to a vertex w and the single corresponding incoming edge from w labeled $(\$, \bar{a}, *)$. If we define the synchronization constraint $C := \{(\sigma, \sigma) \mid \sigma \in \Sigma\}$, the synchronized product of G and H will contain exactly the valid computations of M. To decide whether M halts on the empty tape we thus have to check the truth of the formula

$$\exists xy\Big[\forall z \bigwedge_{\sigma\in\Sigma} \neg E_\sigma zx \wedge \exists z\Big(\operatorname{Reach}_\Sigma(x,z) \wedge \bigwedge_{\sigma\in\{+,-\}\times\bar{\Gamma}\times\{\top\}} E_\sigma zy\Big)\Big]$$

in the semifinitely synchronized product of G and H. □

4. Transitive Closure Logic over the Infinite Grid

The infinite grid is the structure $\mathcal{G} = (\omega^2, S_1, S_2)$ with two successor relations S_1 and S_2. It can be viewed as the asynchronous and hence finitely synchronized product of two copies of the natural numbers with successor relation, $\mathcal{N}_1 = (\omega, S_1)$ and $\mathcal{N}_2 = (\omega, S_2)$, defined by the empty synchronization constraint.

We show in this section how to interpret the first-order theory of addition and multiplication of the natural numbers in $\mathrm{FO(TC)}^2_{(1)}$ (without parameters) over the infinite grid. $\mathrm{FO(TC)}^2_{(1)}$ allows only transitive closure operators of arity one and a nesting depth of two.

It is well known that the FO-theory of addition and multiplication of $\mathcal{N}$ is undecidable. However, since $\mathrm{FO(TC)}_{(1)}$ can be interpreted in MSO, $\mathrm{FO(TC)}_{(1)}$ is decidable over $\mathcal{N}$. From these results we can conclude that the $\mathrm{FO(TC)}^2_{(1)}$-theory is not preserved by finitely synchronized products and thus obtain that we cannot extend FO(R) much without losing decidability for finitely synchronized products.

To interpret the theory of addition and multiplication in $\mathrm{FO(TC)}^2_{(1)}$ over the infinite grid we first connect the transitive closure theories of $\mathcal{N}$ and $\mathcal{G}$.

Lemma 4.1. *Let $k \geq 1$.*

(a) For every $FO(TC)^n_{(k)}$-sentence φ there is a $FO(TC)^n_{(2k)}$-sentence $\tilde{\varphi}$ such that $\mathcal{G} \models \varphi \Leftrightarrow \mathcal{N} \models \tilde{\varphi}$.

(b) For every $FO(TC)^n_{(2k)}$-sentence φ there is a $FO(TC)^n_{(k)}$-sentence $\hat{\varphi}$ such that $\mathcal{N} \models \varphi \Leftrightarrow \mathcal{G} \models \hat{\varphi}$.

Proof. For (a) there is almost nothing to show. It suffices to split every variable x (interpreted as vertex of the grid) into coordinate variables x_1 and x_2 (interpreted as natural numbers) and to replace the atomic formulas S_1xy by Sx_1y_1 and S_2xy by Sx_2y_2.

For (b) we identify every $x \in \omega$ with $(x, 0) \in \omega^2$. To reduce the number of variables needed in a TC operator we represent a pair of variables x_1, x_2 by a single variable $x = (x_1, x_2)$ to be interpreted as a vertex of the grid.

To finish the proof it suffices to show that the following operations are $\mathrm{FO(TC)}_{(1)}$ definable:

(i) π_i with $\pi_1((x_1, x_2)) := (x_1, 0)$ and $\pi_2((x_1, x_2)) := (0, x_2)$,
(ii) swap_i with $\mathrm{swap}_1((x, 0)) := (0, x)$ and $\mathrm{swap}_2((0, x)) := (x, 0)$,
(iii) comb with $\mathrm{comb}((x, 0), (0, y)) := (x, y)$

Then a $\mathrm{FO(TC)}_{(2k)}$ formula $[\mathrm{TC}_{\bar{x},\bar{y}}\, \varphi(\bar{x}, \bar{y}, \bar{z})]\bar{x}, \bar{y}$ is equivalent to the $\mathrm{FO(TC)}_{(k)}$ formula

$$\exists \bar{u}\bar{v}\Big(\bigwedge_{1\le i\le k} (u_i = \mathrm{comb}(x_{2i-1}, \mathrm{swap}_2(x_{2i})) \wedge v_i = \mathrm{comb}(y_{2i-1}, \mathrm{swap}_2(y_{2i}))\big) \\ \wedge [\mathrm{TC}_{\bar{u},\bar{v}}\, \tilde{\varphi}(\bar{u},\bar{v},\bar{z})]\bar{u},\bar{v}\Big)$$

where

$$\tilde{\varphi} := \exists \bar{x}\bar{y}\Big(\bigwedge_{1\le i\le k} (x_{2i-1} = \pi_1(u_i) \wedge x_{2i} = \mathrm{swap}_2(\pi_2(u_i)) \wedge y_{2i-1} = \pi_1(v_i) \\ \wedge y_{2i} = \mathrm{swap}_2(\pi_2(v_i))\big) \wedge \varphi(\bar{x},\bar{y})\Big)$$

and in φ every occurrence of the symbol S is replaced by S_1.

Let us now define the operations above:

$$\begin{aligned} \pi_1(x) = y &\leftrightarrow& y \le_2 x \wedge \forall z(z \le_2 x \rightarrow z = y) \\ \mathrm{swap}_1(x) = y &\leftrightarrow& \forall z(z \le_2 x \rightarrow z = x) \wedge [\mathrm{TC}_{x,y}\, \exists z(S_1xz \wedge S_2yz)]x,y \\ && \wedge \forall z(z \le_1 y \rightarrow z = y) \\ \mathrm{comb}(x,y) = z &\leftrightarrow& \forall u(u \le_2 x \rightarrow u = x) \wedge \forall u(u \le_1 y \rightarrow u = y) \\ && \wedge x \le_1 z \wedge y \le_2 z \end{aligned}$$

Observe that if the formula φ has no TC operators with parameters, then neither $\tilde{\varphi}$ nor $\hat{\varphi}$ has (in $\tilde{\varphi}$ only TC-formulas without parameters are introduced), and that the nesting depth is not increased. □

Let us now turn to the undecidability proof.

Theorem 4.2. *The $FO(TC)^2_{(1)}$-theory of the infinite grid is undecidable.*

Proof. We define addition and multiplication in $\mathrm{FO(TC)}_{(1)}$ over $\mathcal{G}$ without the use of parameters. By Lemma 4.1 it is enough to define these operations in $\mathrm{FO(TC)}_{(2)}$ over $\mathcal{N}$. The definition of addition is straightforward.

$$a + b = c \leftrightarrow \mathcal{N} \models [\mathrm{TC}_{x_1x_2,y_1y_2}\, Sx_1y_1 \wedge Sx_2y_2]0\,a, b\,c$$

To define multiplication note that $x \cdot y = \frac{(x+y)^2 - x^2 - y^2}{2}$, hence it suffices to define the square function. To define x^2 note that $x^2 = \sum_{i=0}^{x-1} 2i + 1$. The formula

$$\psi(x,y) = [\mathrm{TC}_{x_1x_2,y_1y_2}\, y_2 = x_2 + (x_2 - x_1) + 2 \wedge y_1 = x_2]0\,1, xy$$

defines all pairs of square numbers

$$\Big(\sum_{i=1}^{k-2} 2i + 1, \sum_{i=1}^{k-1} 2i + 1\Big) \text{ for } k \ge 3.$$

Hence $\mathcal{N} \models \psi[a,b]$ iff $b - a = 2k - 1$ for some $k \ge 2$. Let

$$\chi(x,y) = \exists z_1\Big(\psi(z_1,y) \wedge \frac{y - z_1 + 1}{2} = x\Big).$$

Then $\mathcal{N} \models \chi[a,b]$ iff $b = a^2$. □

A similar technique was used in [Avr03] to define multiplication in $(\omega, +, 0)$ using a transitive closure operator of arity one.

The nesting of transitive closure operators in the previous proof is necessary. If we disallow nesting, even in the presence of parameters in the transitive closure formulas, the theory of the infinite grid is decidable.

Theorem 4.3. *The $FO(TC)^1_{(1)}$-theory of the infinite grid is decidable.*

Proof. We reduce the $\mathrm{FO(TC)}^1_{(1)}$-theory of the infinite grid $\mathcal{G}$ to Presburger arithmetic, the first-order theory of $\mathcal{N}_+ = (\omega, +, 0)$, in the following sense: For every $\mathrm{FO(TC)}^1_{(1)}$-formula $\varphi(x_1, \ldots, x_n)$ one can construct a Presburger formula $\tilde{\varphi}(x_{11}, x_{12}, \ldots, x_{n1}, x_{n2})$ such that

$$\mathcal{G} \models \varphi[(k_1, l_1), \ldots, (k_n, l_n)] \Leftrightarrow \mathcal{N}_+ \models \tilde{\varphi}[k_1, l_1, \ldots, k_n, l_n]. \tag{4.1}$$

In order to construct $\tilde{\varphi}$ it suffices to consider the case

$$\varphi(x_1, \ldots, x_n) = [\mathrm{TC}_{x_1, x_2}\, \psi(x_1, \ldots, x_n)] x_1, x_2,$$

or for better readability

$$\varphi(x_1, \ldots, x_n) = [\mathrm{TC}_{x,y}\, \psi(x, y, x_3, \ldots, x_n)] x_1, x_2$$

where ψ is a first-order formula. The second notation emphasizes that $x_3, \ldots, x_n$ serve as parameters in the transitive closure formula.

In a first step we rewrite ψ in a normal form, applying Hanf's Theorem for first-order logic over graphs (see [Han65, EF95, Tho97]).

For this purpose we recall some definitions. The r-sphere $r\text{-sph}(d)$ around a vertex $d \in \omega^2$ is the set of grid vertices which are of distance less or equal to r from d, where we allow to traverse the edges in either direction. Invoking the distributive normal form and Hanf's Theorem, there exists a suitable $r > 0$ such that $\psi(x_1, \ldots, x_n)$ is equivalent to a disjunction of formulas $\varphi_\tau(x_1, \ldots, x_n)$ where each φ_τ describes the isomorphism type τ of $\bigcup_{1 \le i \le n} r\text{-sph}(c_i)$ for some tuple $c_1, \ldots, c_n$ of grid vertices. Let T be the set of all such types. Since T is finite it suffices to consider only finitely many tuples $c_1, \ldots, c_n$.

Remark. In the general case, over an arbitrary graph instead of the infinite grid, Hanf's Theorem involves a statement on the number (up to a certain threshold) of spheres outside $\bigcup_{1 \le i \le n} r\text{-sph}(c_i)$. This statement is superfluous here due to the regular structure of the infinite grid. (For technical convenience we assume that $(0,0)$ is included in the set of parameters, so every isomorphism type realizable in $\mathcal{G}$ outside $\bigcup_{1 \le i \le n} r\text{-sph}(c_i)$ occurs an infinite number of times.)

Due to the special structure of the grid, which we depict as a diagram with the bottom row and left column as margins, open upwards and to the right, every formula $\varphi_\tau(x_1, \ldots, x_n)$ can be expressed by conditions on the vertices $x_1, \ldots, x_n$ which fix their distances up to the radius r from the left margin as well as the bottom margin, and their relative distances up to $2r$.

It is convenient to express $\varphi_\tau(x_1, \ldots, x_n)$ in terms of the $2n$ components of the vertices, obtaining a formula $\tilde{\varphi}_\tau(x_{11}, x_{12}, \ldots, x_{n1}, x_{n2})$ The formula $\tilde{\varphi}_\tau$ is interpreted over ω and equivalent to φ in the sense of (4.1) above. It is a conjunction of statements

- $x_{ih} = k$ for $0 \le k \le r$ or $x_{ih} > r$
- $(x_{i1}, x_{i2}) = (x_{j1}, x_{j2}) + (k, l)$ for $-2r \le k, l \le 2r$
- $\mathrm{dist}((x_{i1}, x_{i2}), (x_{j1}, x_{j2})) > 2r$

where $1 \leq i, j \leq n$ and $h \in \{1, 2\}$.

We now have to evaluate formulas of the form

$$\Big[\mathrm{TC}_{(x_{11},x_{12}),(x_{21},x_{22})} \bigvee_{\tau \in T'} \tilde{\varphi}_\tau(x_{11}, x_{12}, \ldots, x_{n1}, x_{n2})\Big](s,t),(u,v) \tag{4.2}$$

for some $T' \subseteq T$.

In a first step we note that it is possible to add disjuncts to (4.2) such that vertices tied to occur in a $2r$-sphere around a parameter (x_{i1}, x_{i2}) for $i > 2$ only need to appear as start vertex or as end vertex of any path described by (4.2). Hence vertices tied to parameters can be handled without the use of TC, by an appropriate modification of the formula.

Let I be an initial segment of the grid encompassing the $2r$-spheres around parameters (x_{i1}, x_{i2}) for $i > 2$. Outside this initial segment, in a second step, it suffices to consider formulas (4.2) in which only type formulas $\tilde{\varphi}_\tau$ which contain

$$x_{11} = k_1 \wedge x_{12} > r \text{ or } x_{11} > r \wedge x_{12} = k_2 \text{ or } x_{11} > r \wedge x_{12} > r \text{ for } k_1, k_2 \leq r$$

and

$$x_{21} = l_1 \wedge x_{22} > r \text{ or } x_{21} > r \wedge x_{22} = l_2 \text{ or } x_{21} > r \wedge x_{22} > r \text{ for } l_1, l_2 \leq r$$

and

$$\mathrm{dist}((x_{11}, x_{12}), (x_{21}, x_{22})) > 2r \text{ or } (x_{11}, x_{12}) = (x_{21}, x_{22}) + (k, l) \text{ for } -2r \leq k, l \leq 2r$$

appear.

It is now possible to apply a finite saturation process to obtain a formula

$$\Big[\mathrm{TC}_{(x_{11},x_{12}),(x_{21},x_{22})} \bigvee_{1 \leq j \leq m} \tilde{\varphi}_j(x_{11}, x_{12}, \ldots, x_{n1}, x_{n2})\Big](s,t),(u,v) \tag{4.3}$$

which is equivalent to (4.2) and where TC and $\bigvee$ commute, i.e.

$$\mathcal{G} \models \Big[\mathrm{TC}_{(x_{11},x_{12}),(x_{21},x_{22})} \bigvee_{1 \leq j \leq m} \tilde{\varphi}_j\Big](s,t),(u,v) \Leftrightarrow$$

$$\mathcal{G} \models \bigvee_{1 \leq j \leq m} \Big[\mathrm{TC}_{(x_{11},x_{12}),(x_{21},x_{22})} \tilde{\varphi}_j\Big](s,t),(u,v).$$

The subformulas $\tilde{\varphi}_j$ in (4.3) have the same format as the subformulas $\tilde{\varphi}_\tau$ in (4.2) except that the center of the excluded $2r$-sphere around (x_{11}, x_{12}) may be shifted by a bounded distance from (x_{11}, x_{12}) or be missing, or $\tilde{\varphi}_\tau$ defines the complete relation outside I and the border stripes of width r. Thus it remains to consider two cases.

Case 1. If $\tilde{\varphi}_j$ contains a conjunct excluding some $2r$-sphere then the relation defined by $[\mathrm{TC}_{(x_{11},x_{12}),(x_{21},x_{22})} \tilde{\varphi}_j](s,t),(u,v)$ is cofinite (w.r.t. the grid excluding I and border stripes of width r, or a fixed line in one of the border stripes) and hence definable without the use of a transitive closure operator.

Case 2. If $\tilde{\varphi}_j$ fixes relations of the form

$$(x_{21}, x_{22}) = (x_{11}, x_{12}) + (k_i, l_i) \tag{4.4}$$

for $i = 1, \ldots, N$ and $-2r \leq k_i, l_i \leq 2r$. the formula

$$[\mathrm{TC}_{(x_{11},x_{12}),(x_{21},x_{22})} \tilde{\varphi}_j](s,t),(u,v)$$

expresses that there is a path from (s,t) to (u,v) consisting of steps of the form (4.3). The set of vertices (u,v) reachable in this way from (s,t) can be represented as the union of paths in the finite initial segment I of the grid and finitely many sets of the form

$$\{(u,v) \mid (u,v) = (s',t') + y_1(k_1,l_1) + \ldots + y_N(k_N,l_N)\}.$$

Here $y_i \geq 0$, the (s',t') range over boundary vertices of I, and the (k_i,l_i) are from (4.4). It follows that the relation defined by (4.2) is definable in Presburger arithmetic. □

5. Conclusion

We have proved a result on compositional model checking for a logic including reachability predicates, and we have shown tight limitations for possible extensions of this result.

Let us mention some questions left open in this paper:

(1) The composition result (Theorem 3.1) should be generalized to infinite products.
(2) For an extension of Theorem 3.1, one can enrich FO(R) by an operator for "recurrent reachability" (existence of an infinite path which visits a designated set infinitely often), or one can consider stronger logics like (fragments of) CTL.
(3) Interesting subcases of Theorem 3.1 should be found where the mentioned blow-up of complexity can be avoided.
(4) The distinction between products which are asynchronous, finitely synchronized, or synchronized should be refined, by allowing other means of coordination between component structures, also incorporating the special case of synchronization of parameterized systems composed from identical components.

Acknowledgment

We thank C. Löding for pointing us to GTRS-graphs to prove Theorem 3.3 and the anonymous referees (both of the conference version and the journal version of this paper) for many helpful comments and pointers to related literature.

References

[Arn94] A. Arnold. *Finite Transition Systems.* Prentice Hall, 1994.

[Avr03] A. Avron. Transitive closure and the mechanization of mathematics. In F. Kamareddine, editor, *Thirty Five Years of Automating Mathematics*, pages 149–171. Kluwer Academic Publishers, 2003.

[Cau96] D. Caucal. On infinite transition graphs having a decidable monadic theory. In *Proceedings of the 23rd International Colloquium on Automata, Languages and Programming*, volume 1099 of *Lecture Notes in Computer Science*, pages 194–205, 1996.

[Cau02] D. Caucal. On infinite terms having a decidable theory. In *Proceedings of the 27th International Symnposium on Mathematical Foundations of Computer Science*, volume 2420 of *Lecture Notes in Computer Science*, pages 165–176. Springer, 2002.

[CK73] C. Chang and H. Keisler. *Model Theory.* North-Holland, 1973.

[Col02] T. Colcombet. On families of graphs having a decidable first order theory with reachability. In *Proceedings of the 29th International Conference on Automata, Languages, and Programming*, volume 2380 of *Lecture Notes in Computer Science*, pages 98–109, 2002.

[EF95] H.D. Ebbinghaus and J. Flum. *Finite Model Theory.* Springer, 1995.

[FV59] S. Feferman and R.L. Vaught. The first-order properties of products of algebraic systems. *Fundamenta Mathematicae*, 47:57–103, 1959.

[Gro96] M. Grohe. Arity hierarchies. *Annals of Pure and Applied Logic*, 82:103–163, 1996.

[Han65] W. Hanf. Model-theoretic methods in the study of elementary logic. In *Proceedings of the Symposium on the Theory of Models*, pages 132–145. North Holland, 1965.

[Hod93] W. Hodges. *Model Theory*. Cambridge University Press, 1993.

[HU79] J.E. Hopcroft and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison-Wesley, 1979.

[KP99] T. Knapik and É. Payet. Synchronized product of linear bounded machines. In *Proceedings of the 12th International Symposium on Fundamentals of Computation Theory*, volume 1684 of *Lecture Notes in Computer Science*, pages 362–373. Springer, 1999.

[Löd02] C. Löding. Model-checking infinite systems generated by ground tree rewriting. In *Proceedings of the 5th International Conference on Foundations of Software Science and Computation Structures*, volume 2303 of *Lecture Notes in Computer Science*, pages 280–294. Springer, 2002.

[Löd03] C. Löding. *Infinite Graphs Generated by Tree Rewriting*. PhD thesis, RWTH Aachen, 2003.

[Mak04] J.A. Makowsky. Algorithmic aspects of the Feferman-Vaught theorem. *Annals of Pure and Applied Logic*, 126:159–213, 2004.

[Mor00] C. Morvan. On rational graphs. In *Proceedings of the 3rd International Conference on Foundations of Software Science and Computation Structures*, volume 1784 of *Lecture Notes in Computer Science*, pages 252–266. Springer, 2000.

[MS85] D.E. Muller and P.E. Schupp. The theory of ends, pushdown automata, and second-order logic. *Theoretical Computer Science*, 37:51–75, 1985.

[Pay00] É. Payet. Thue specifications, infinite graphs and synchronized product. *Fundamenta Informaticae*, 44:265–290, 2000.

[Rab07] Alexander Rabinovich. On compositionality and its limitations. *ACM Transactions on Computational Logic*, 8(1), 2007.

[Tho97] W. Thomas. Languages, automata, and logic. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 3, pages 389–455. Springer, 1997.

[Tho03] W. Thomas. Constructing infinite graphs with a decidable MSO-theory. In *Proceedings of the 28th International Symposium on Mathematical Foundations of Computer Science*, volume 2747 of *Lecture Notes in Computer Science*, pages 113–124. Springer, 2003.

[WT04] S. Wöhrle and W. Thomas. Model checking synchronized products of infinite transition systems. In *Proceedings of the 19th Annual Symposium on Logic in Computer Science*, pages 2–11. IEEE Computer Society, 2004.