Compositional Verification : Decidability issues using graph substitutions.

Olivier Ly

LaBRI – Bordeaux I University

Abstract. This paper deals with the *compositional verification* of sequential programs. This consists in deciding whether or not a given set of local structural properties of the functions of a program implies a given global behavioural property of the program. Here we consider properties expressed in monadic second-order logic dealing with the control flow of the program and the function calls occuring during its execution. This problem has been investigated in relation with the security of open multi-application smart cards. We show that the compositionality is *decidable* for sequential programs whose control-flow graphs are of tree-width less than a fixed integer value, which includes in particular structured programs. Formally, we prove the decidability of MSO theories of families of hypergraphs obtained by uniform substitution of hyperedges by hypergraphs specified by MSO formulas.

Keywords. Compositional Verification, Tree Automata, Monadic Second-Order Logic

Introduction

This paper deals with the *compositional verification* of sequential programs. This consists in deciding whether or not a given set of local structural properties of the functions of a program implies a given global behavioural property of the program. This aims at reducing the verification of a global property of a program to the verifications of some independent local properties of its components.

Compositional verification is a long standing problem in the area of concurrent systems (see *e.g.* [9, 1, 13]). Here we rather consider sequential programs and we restrict our study to behavioural and structural properties dealing with the control flow of the program and the function calls occurring during execution. This framework relies on the language-independent model of programs introduced in [12] in order to catch some classical security properties. It has been studied in [4, 3, 18] (see also [7]) for compositional reasoning about security of open multi-application smart cards. Like *e.g.* [19, 2, 15] for concurrent systems, a proof system dealing with modal μ-calculus has been set up in [4] for the framework that we consider. But the question of decidability was left open. Then a decision procedure has been proposed in [18] by restricting the study to properties expressed in simulation logic. Here we consider properties expressed in *monadic second-order logic* (see *e.g.* [10]) and we prove that for any fixed integer value k, the compositionality is a *decidable* problem for sequential programs whose control-flow graphs are of tree-width less than k.

This contributes to the studies mentioned above seeing that monadic second-order logic contains modal μ-calculus and *a fortiori* simulation logic. The limitation of our solution concerns the tree-width of the control-flow graphs. Roughly speaking, the tree-width is an integer value measuring how far a graph is from being a tree (see [17]). We claim that this limitation is reasonable, because reasonable programs are indeed of bounded tree-width as it has been shown in [21]: the tree-width of the control-flow graph of any goto-free C program is at most 6. Moreover, a study reported in [11] showed that the tree-width for control-flow graphs of the functions of the Java API source code does not exceed 5, and the average of it is only 2.7.

All the paper is devoted to the proof of the decidability result (Theorem 1), which relies on a variation of the classical link between monadic second-order logic and *automata* (see *e.g.* [20]). First, we remark that the compositionality problem as described above can be reformulated into the satisfiability problem for families of infinite hypergraphs (actually HR-equational hypergraphs) obtained by uniform substitution of hyperedges by hypergraphs specified by MSO formulas, which we call *MSO uniform expansions.* These families actually are the sets of transition systems which encode the behaviours of the programs made of functions satisfying a given set of structural properties. Second, we prove the decidability of MSO theories of MSO uniform expansions in the case of bounded tree-width. The first step of the proof is to translate the problem into a tree language problem, introducing for that a new kind of automata on infinite trees called *composite automata*. These automata encode MSO uniform expansions via the the concept of syntactic expressions, which is possible because of the bounded tree-width. Finally we prove the decidability of the emptyness of the intersection of the languages of a composite automaton and a Rabin automaton. We do that by using a variation of the classical point view of runs of automata in terms of *games* (see *e.g.* [20]). This result allows us to give out a bound for the length of solutions, and therefore to solve the problem by an enumeration method.

Acknowledgements. The author thanks B. Courcelle and G. Sénizergues for many helpful comments and suggestions.

1 Formal Definitions

Hypergraphs. A *hypergraph*, also called *relational structure*, is a tuple of the form $(V, L, \{R_\ell\}_{\ell \in L})$ where V is the set of vertices; L is the finite set of labels, and for each $\ell \in L$, R_ℓ is a relation over V, i.e., a subset of V^n for some strictly positive integer n. An element of R_ℓ of form $(v_1, ..., v_n)$ defines a *hyperedge* whose vertex list is $v_1, ..., v_n$; n is called the arity of the hyperedge. Let us note that the labelling of vertices is encoded by hyperedges of arity 1. *Graphs* are the hypergraphs whose hyperedges are all of arity 1 or 2. Here we actually deal with hypergraphs with *sources* which generalise pointed graphs; such a hypergraph is a hypergraph provided with a finite ordered list of pairwise distinct vertices. These distinguished vertices are called the sources of the hypergraph, the other ones are said to be internal. The sequence of sources of a hypergraph H is denoted by $src(H)$. The number of sources of a hypergraph is called its *type*. Let G be a hypergraph. A *tree-decomposition* is a pair (U, f) where U is an undirected tree and $f : V_U \to 2^{V_G}$ such that: $V_G = \bigcup_{i \in V_U} f(i)$; if a list of vertices $v_1 \ldots v_n$ of G are connected by a hyperedge, then there exists $i \in V_U$ such that $v_1, \ldots, v_n \in f(i)$; if $i, j, k \in V_U$ are such that j belongs to the shortest path between i and k, then $f(i) \cap f(k) \subseteq f(j)$. The *tree-width* (see [17]) of such a decomposition is defined to be $\max\{\text{card}(f(i)) \mid i \in V_U\} - 1$. The *tree-width* of G, denoted by $twd(G)$, is the minimum of the tree-widths of all its tree-decompositions.

HR-equational Graphs. The concept of *HR-equational hypergraph* extends those of context-free graphs (see [14]). It has been introduced in [6]. Such a hypergraph, possibly infinite, is defined according to a deterministic *hyperedge replacement graph grammar* (see [8]). Such a grammar is defined according to a finite set of symbols L which is divided into two subsets L_1 and L_2 whose elements are respectively called *terminal* symbols and *non-terminal* symbols. A deterministic hyperedge replacement grammar[1] is a tuple $(\ell_0, \{H_\ell\}_{\ell \in L_2})$ where $\ell_0 \in L_2$ is the initial symbol and for each $\ell \in L_2$, H_ℓ is a finite L-labelled hypergraph with source, such that for each $\ell \in L$, the arities of all the ℓ-labelled hyperedges of any $H_{\ell'}$ are equal; and if $\ell \in L_2$, this arity must be equal to the type of H_ℓ. The construction of the HR-equational hypergraph generated by such a grammar is as follows: First, one starts from H_{ℓ_0}. Then, according to the grammar, one replaces each hyperedge labelled by a non-terminal symbol by the associated hypergraph, gluing the sources of the hypergraph in place of the vertices of the hyperedge. One gets a new hypergraph, with possibly new hyperedges labelled by a non-terminal symbols. One replaces again these last ones as above, and so on, possibly infinitely many times, until there is no more such hyperedge.

Graph Grammar Compositionality Problem. We consider *monadic second-order formulæ* on hypergraphs (MSO-formulæ for short, see *e.g.* [10]). These formulæ are constructed using individual variables and set variables, taking their values in the set of vertices. Atomic formulæ are of the forms $x \in A$, $A \subset B$, or else $\mathsf{edg}_\eta(x_1, \ldots, x_n)$ which encodes the fact that $x_1, \ldots, x_n$ are connected by an η-labelled hyperedge. Syntax is not restricted: we allow existential and universal quantifiers over individual and set variables, conjunctions and negations. The set of finite models (respectively infinite) of a formula φ is denoted by $\mathcal{M}(\varphi)$ (respectively $\mathcal{M}^\infty(\varphi)$). We shall also use the set $\mathcal{M}_{\text{twd}<k}(\varphi)$ (respectively $\mathcal{M}^\infty_{\text{twd}<k}(\varphi)$) of finite (respectively infinite) models of φ whose tree-width is less than a fixed value k. The MSO-theory of a family $\mathfrak{F}$ of hypergraphs, i.e., the set of MSO-formulæ which are satisfied by all the graphs of $\mathfrak{F}$, shall be denoted by $\mathcal{T}h_{\text{MSO}}(\mathfrak{F})$. Let us consider a finite set of symbols $\mathcal{M}$ given with an *arity* mapping $\alpha : \mathcal{M} \to \mathbb{N}$. Let us call *$\mathcal{M}$-vector of typed-graph languages* any tuple $(\mathfrak{L}_m)_{m \in \mathcal{M}}$ of sets of finite graphs such that for each $m \in \mathcal{M}$, all the graphs of $\mathfrak{L}_m$ are of type $\alpha(m)$. Any pair $\mathfrak{G} = (m_0, (\mathfrak{L}_m)_{m \in \mathcal{M}})$

[1] Seeing that the grammars that we consider are deterministic, i.e., it associates to each non-terminal symbol one and only one hypergraph, a production of form $\ell \to H_\ell$ is just denoted by H_ℓ.

made of a distinguished symbol $m_0 \in \mathcal{M}$ and a $\mathcal{M}$-vector of typed-graph languages $(\mathfrak{L}_m)_{m\in\mathcal{M}}$ defines a family of deterministic graph grammars as follows: $\mathcal{G}r(\mathfrak{S}) = \{(m_0, \{g_m\}_{m\in\mathcal{M}}) \mid \forall m \in \mathcal{M} : g_m \in \mathfrak{L}_m\}$. Let us now consider the set of all the HR-equational hypergraphs generated by any such graph grammar:

$$\mathcal{E}xp(\mathfrak{S}) = \{G \mid \text{there exists } (m_0, \{g_m\}_{m\in\mathcal{M}}) \in \mathcal{G}r(\mathfrak{S}) \text{ which generates } G\}$$

Informally, a graph of $\mathcal{E}xp(\mathfrak{S})$ is obtained by picking up one graph g_m in each $\mathfrak{L}_m$, once for all, and gluing the g_m's to each others infinitely many times. Such a graph shall be said to be obtained by *uniform expansion* from $\mathfrak{S}$ and the family $\mathcal{E}xp(\mathfrak{S})$ shall be called the uniform expansion of $\mathfrak{S}$. *The compositionality problem consists in determining the theory of $\mathcal{E}xp(\mathfrak{S})$ from the theories of the $\mathfrak{L}_m$'s.*

The $\mathfrak{L}_m$'s are *a priori* infinite[2]. A way to give a constructive version of the problem is to consider the case where $\mathcal{M}$-vectors of typed-graph languages are defined by logical formulæ, i.e., $\mathcal{M}$-vectors of the form $(\mathcal{M}(\varphi_m))_{m\in\mathcal{M}}$, where the φ_m's are MSO-formulæ. The family of hypergraphs obtained by uniform expansion from such a $\mathcal{M}$-vector shall be called the *MSO uniform expansion* of the $\mathcal{M}$-vector $(\varphi_m)_{m\in\mathcal{M}}$ according to m_0, it shall be denoted by $\mathcal{E}xp(m_0, (\varphi_m)_{m\in\mathcal{M}})$. It is easy to see that in general theories of MSO uniform expansions are not computable. However, if we bound the tree-width of the considered graphs, i.e., if we consider $\mathcal{M}$-vectors of the form $(\mathcal{M}_{\text{twd}<k}(\varphi_m))_{m\in\mathcal{M}}$ for some fixed k, one gets uniform expansions of form $\mathcal{E}xp(m_0, (\mathcal{M}_{\text{twd}<k}(\varphi_m))_{m\in\mathcal{M}})$, called *bounded-tree-width MSO uniform expansions*, whose MSO-theories are decidable; this is the main point of this paper:

Theorem 1. *Let $\mathcal{M}$ be a set of symbols, $m_0 \in \mathcal{M}$, k a positive integer and $(\varphi_m)_{m\in\mathcal{M}}$ an $\mathcal{M}$-vector of MSO-formulæ, then $\mathcal{T}h_{MSO}(\mathcal{E}xp(m_0, (\mathcal{M}_{twd<k}(\varphi_m)_{m\in\mathcal{M}})))$ is effectively computable.*

In other words, seeing that MSO logic is closed by negation, the following problem is effectively decidable:

Problem 1 (Bounded Tree-Width MSO-Compositionality). Given an integer k, a distinguished symbol m_0, a vector of MSO-formulæ $(\varphi_m)_{m\in\mathcal{M}}$ and a MSO-formula Ψ, does there exist $G \in \mathcal{E}xp(m_0, (\mathcal{M}_{\text{twd}<k}(\varphi_m))_{m\in\mathcal{M}})$ such that $G \models \Psi$?

2 Motivation: Program Models

Here we define program models. They generalise[3] the concepts introduced in [4] which was inspired by [12]. In particular, this model allows us to catch security properties considered in [4], see also [12, 3, 18].

Let $\mathcal{M}$ be a set of symbols whose elements are the *names* of functions. A *function* is a pair (m, G_m) where $m \in \mathcal{M}$ is the name of the function and G_m is its *control-flow graph*, i.e., a tuple $(V_m, \rightarrow_m, \lambda_m)$ such that V_m is the finite set of *program points* of m. The relation $\rightarrow_m \subset V_m \times V_m$ encode the *transfer edges*

[2] If all the $\mathfrak{L}_m$'s are finite, then the problem relies to the MSO-satisfiability of HR-equational hypergraphs, which is already known to be decidable (see [6]).

[3] We kept the formalism used in [4] to encode control-flow graphs and transition systems. However these concepts can be easily defined as relational structures.

of m. $\lambda_m : V_m \to \{\mathsf{entry}, \mathsf{seq}, \mathsf{ret}\} \cup \{\mathsf{call}\ m\}_{m \in \mathcal{M}}$ associates to each program point of m a program point type. The entry points are the entry points of the function. The call points are the points where the function calls an other function. The ret points are the returning points of the function. The seq points are all the other ones, which are not distinguished. A *program* $\mathcal{P}$ consists of a set of functions $\{(m, G_m)\}_{m \in \mathcal{M}}$. Let $V_{\mathcal{P}}$ denote the set of all the program points of all the functions of $\mathcal{P}$. A *state* of $\mathcal{P}$ is a pair (c, σ) consisting of a program point, i.e., an element of $V_{\mathcal{P}}$, and a call stack $\sigma \in V_{\mathcal{P}}^*$. A program $\mathcal{P}$ induces a *labelled transition system*, i.e., a graph $\mathcal{T}_{\mathcal{P}} = (\mathcal{S}_{\mathcal{P}}, \mathcal{L}_{\mathcal{P}}, \to_{\mathcal{P}})$ defined as follows: The set of *states* $\mathcal{S}_{\mathcal{P}}$ is the set of the states of $\mathcal{P}$ as defined above. The set of *edge labels* $\mathcal{L}_{\mathcal{P}}$ is defined to be $\{\tau, \mathsf{call}, \mathsf{ret}\}$. The call-labelled edges encode function calls. The ret-labelled edges encode function returns. Any other transition of the system is labelled by τ which is called the silent action (see [4]).

For each function name $m \in \mathcal{M}$, we consider the labelled transition system $\mathcal{T}_{\mathcal{P},m}$ defined to be the subgraph of all the vertices of $\mathcal{T}_{\mathcal{P}}$ which are accessible from a state of the form (c_i, ε) where c_i is an entry point of m.

Lemma 1. *$\mathcal{T}_{\mathcal{P},m}$ is a HR-equational graph.*

Remark 1. This result could actually be refined seeing that $\mathcal{T}_{\mathcal{P},m}$ is of finite degree: this implies that it is a context-free graph. Therefore, it can be constructed as the configuration graph of a pushdown automaton (see [14]), recovering the result of [4].

Any $\mathcal{M}$-vector $(\varphi_m)_{m \in \mathcal{M}}$ of MSO-formulæ specifies a set of programs $\mathfrak{P}((\varphi_m)_{m \in \mathcal{M}}) = \{(m, G_m)_{m \in \mathcal{M}} \mid \forall\, m \in \mathcal{M} : G_m \models \varphi_m\}$.

Problem 2 (Program Models Compositionality). Given a set of formulæ $(\varphi_m)_{m \in \mathcal{M}}$, a method name m_0, and a formula Ψ, does there exist[4] a program $\mathcal{P} \in \mathfrak{P}((\varphi_m)_{m \in \mathcal{M}})$ such that $\mathcal{T}_{\mathcal{P},m_0} \models \Psi$?

This problem is Turing reductible to $\mathcal{T}h_{\text{MSO}}(\mathcal{E}xp(m_0, (\varphi_m)_{m \in \mathcal{M}}))$; and actually, it is also undecidable. However, if we restrict the study to control flow graph of bounded tree-width, we get a problem which is Turing reductible to Problem 1, i.e. to the theory of a bounded-tree-width MSO uniform expansion, and therefore decidable.

3 Decidability of Compositionality Problem

Here we introduce some tools for dealing with decidability issues concerning monadic second-order logic: HR-syntactic expressions and tree automata.

HR-syntactic expressions are the expressions based on the following operations on hypergraphs (see [6] for complete definitions): the *redefinition of the sources* of a hypergraph, and the *fusion of the sources* of a hypergraph. These operators are typed by the types of hypergraphs considered (see [6] for details). We consider the set $\mathfrak{T}_{L,k}$ (respectively $\mathfrak{T}^{\infty}_{L,k}$) made of well-formed *finite terms* (respectively *infinite*) on the set of operators dealing with hypergraphs of type

[4] We state the compositionality problem in an existential form for convenience. In our framework, this is equivalent to the classic statement seeing that the monadic second-order logic is closed by negation of formulæ.

bounded by k, and L-labelled hyperedges. The finite hypergraph denoted by a finite term of $\mathfrak{T}_{L,k}$, i.e. the *value* of this term, is defined inductively according to the definitions of the semantics of each operator which has been given above (see [6]). The value of an infinite term $t \in \mathfrak{T}^{\infty}_{L,k}$, which is an infinite hypergraph, can be defined in two ways: as the inductive limit of a sequence of finite hypergraphs defined from an increasing sequence of sub-terms of t, or as the quotient of a graph constructed from the set of leaves of t (see [6]). The value of a term is denoted by $\bar{t}$. *Any* finite hypergraph is the value of a term of some $\mathfrak{T}_{L,k}$, and *any* HR-equational hypergraph[5] is the value of a (regular) term of some $\mathfrak{T}^{\infty}_{L,k}$ (cf. [6]). Besides, the concept of syntactic expressions allows to give an alternative definition of the *tree-width*: the tree-width of a hypergraph is the smallest k such that it is the value of a term of $\mathfrak{T}_{L,k}$ (see [6]). The syntactic expression operators are of arity 0, 1 or 2. Therefore, the terms on these operators can be encoded by possibly infinite ordered binary trees labelled by operators. Here we use automata on such trees: *top-down tree automata* (see [5]) and *Rabin automata* (see [16]). The classical link between automata and monadic second-order logic applies in this framework according to the following result (see [6]): For any formula φ and any integer k, there exists a top-down automaton (respectively a Rabin automaton) $\mathcal{A}$ such that $\mathcal{M}_{\text{twd}<k}(\varphi) = \{\bar{t} \mid t \in \mathfrak{T}_{L,k} \text{ and } t \in \mathcal{L}(\mathcal{A})\}$ (respectively $\mathcal{M}^{\infty}_{\text{twd}<k}(\varphi) = \{\bar{t} \mid t \in \mathfrak{T}^{\infty}_{L,k} \text{ and } t \in \mathcal{L}(\mathcal{A})\}$). One can compute $\mathcal{A}$ from φ and conversely. Therefore Problem 1 can be reformulated in terms of automata as follows:

Problem 3 (Automata Compositionality). Given a distinguished symbol m_0, a $\mathcal{M}$-vector of top-down automata $(\mathcal{A}_m)_{m\in\mathcal{M}}$ and a Rabin automaton $\mathcal{A}'$, does there exist $G \in \mathcal{E}xp(m_0, (\mathcal{L}(\mathcal{A}_m))_{m\in\mathcal{M}})$ such that $G = \bar{t}$ for some $t \in \mathcal{L}(\mathcal{A}')$?

3.1 Composite Automata.

Here we introduce the concept of *composite automata*. It is intended to catch sets of form $\mathcal{E}xp(m_0, (\mathcal{L}(\mathcal{A}_m))_{m\in\mathcal{M}})$ used in Problem 3 (see Lemma 2 below).

As in Problem 3, let us consider a pair $\mathfrak{S} = (m_0, (\mathcal{A}_m)_{m\in\mathcal{M}})$. Without loss of generality, one can assume that each $\mathcal{A}_m$ is such that no transition of it has an initial state in its target vector. Let us consider the automaton $\mathcal{A}(\mathfrak{S})$ constructed from the $\mathcal{A}_m$'s as follows:

1) First, one takes the disjoint union of all the $\mathcal{A}_m$'s. **2**) The terms recognized by $\mathcal{A}_m$ represents elements of $\mathcal{M}(\varphi_m)$. In such terms, the hyperedges encoding gluings regarding the grammar are encoded by sub-terms of form $E_{m'}$ for some m'. Since these last ones are of arity 0, these sub-terms actually are leaves of the complete term, when this last one is seen as a tree. Therefore, if we consider a run ρ of $\mathcal{A}_m$ on such a term, then ρ associates a terminal state to any sub-term of form $E_{m'}$. Let us consider such a terminal state q_f. Without loss of generality, one can assume that $\mathcal{A}_m$ is such that in any of its run, all the sub-terms to which q_f is associated are of the form $E_{m'}$ for a particular fixed m' depending

[5] However, some infinite hypergraphs such as the infinite grid have no syntactic expression in any $\mathfrak{T}^{\infty}_{L,k}$ (see [6].)

only of q_f. One deletes q_f and replaces its occurrences in the target vector of any transition by an initial state of $\mathcal{A}_{m'}$, duplicating the transition in question for each initial state. **3**) The initial states are those of $\mathcal{A}_{m_0}$. **4**) The accepting condition for $\mathcal{A}(\mathfrak{S})$ shall be defined in Definition 1 below.

We distinguish in $\mathcal{A}(\mathfrak{S})$ all the states which are initial states of some $\mathcal{A}_m$; such a state is called a *boundary state.* $\mathcal{A}(\mathfrak{S})$ is called a *composite automaton.*

Induced Partitions. Roughly speaking, a run of a composite automaton on a given infinite tree induces a partition of the tree into finite subtrees, each of them being recognized by a composant of the automaton, i.e., the states coming from a particular $\mathcal{A}_m$. Such a run is accepting if, during the whole run, each composant of the automaton recognizes always the same finite tree (see Definition 1 below). Formally, let T be an infinite tree. A *partition* of T is a set $\{t_i\}_{i\in I}$ of finite subtrees of T such that the t_i's are pairwise disjoint and $T = \bigcup_i t_i$. Let $\{t_i\}_{i\in I}$ be such a partition. For any $i \in I$, let r_i denote the root of t_i. Let us note that I has a canonical non-ordered tree structure in which, for any pair $i, i' \in I$, i is the father of i' if and only if $r_{i'}$ is the son of some node of t_i. The set $\{r_i\}_{i\in I}$ is called the *internal boundary* of $\{t_i\}_{i\in I}$, it is denoted by $\mathcal{F}(\{t_i\}_{i\in I})$. We say that $\{t_i\}_{i\in I}$ is a *tiling* if there exists a finite subset $K \subset I$, assumed to be minimal, such that there exists a mapping $\lambda : I \to K$ such that for any $i \in I$, there exists an isomorphism $\sigma_i : t_i \to t_{\lambda(i)}$. The t_k's are called the *tiles.* $|K|$ is called the *index* of the tiling, it is denoted by $idx(K)$. Such a tiling is said to be *deterministic*[6] if for any $k \in K$ there exists a mapping $\delta_k : t_k \to (K \cup \bot)^*$ where $\bot$ is a new symbol not belonging to K defined as follows: for any $\bar{x} \in t_k$, the n-th element of $\delta_k(\bar{x})$, denoted by $\delta_k(\bar{x})_n$, is equal to $\bot$ if and only if for any $i \in \lambda^{-1}(k)$ and any $x \in \sigma_i^{-1}(\bar{x})$: the n-th son of x still is in t_i. And $\delta_k(\bar{x})_n = k'$ if and only if for any $i \in \lambda^{-1}(k)$ and any $x \in \sigma_i^{-1}(\bar{x})$, the n-th son of x does not belong to t_i and is equal to some $r_{i'}$ with $\lambda(i') = k'$. Let $d_{\max} = \max_{k,x}\{|\delta_k(x)|\}$. Let us assume that there exists an run $\rho : T \to Q_{\mathcal{A}(\mathfrak{S})}$ of $\mathcal{A}(\mathfrak{S})$ on T such that there exists a partition made of finite trees defined by the property that $\mathcal{F}(\mathfrak{P}(\rho))$ is the inverse image by ρ of the set of boundary states of $\mathcal{A}(\mathfrak{S})$ This partition is called the *induced partition* of T regarding ρ, it is denoted by $\mathfrak{P}(\rho)$. Let us consider the mapping $\tau : Q_{\mathcal{A}(\mathfrak{S})} \to \mathcal{M}$ defined by $\tau(q) = m$ if and only if q belongs to $\mathcal{A}_m$.

Definition 1 (Rigid Accepting Condition). *A run $\rho : T \to Q_{\mathcal{A}(\mathfrak{S})}$ is said to be* accepting *if $\mathfrak{P}(\rho)$ is a tiling $(K, \lambda, \{\sigma_i\}_{i\in I})$, and for any pair $i_1, i_2 \in I$, $\tau(\rho(r_{i_1})) = \tau(\rho(r_{i_2}))$ implies that $\lambda(i_1) = \lambda(i_2)$. The language of $\mathcal{A}(\mathfrak{S})$, denoted by $\mathcal{L}^\infty(\mathcal{A}(\mathfrak{S}))$, is defined to be the set of trees for which such a run exists.*

Lemma 2. *A graph G belongs to $\mathcal{E}xp(m_0, (\mathcal{L}(\mathcal{A}_m))_{m\in\mathcal{M}})$ if and only if it has a syntactic expression tree $T \in \mathcal{L}^\infty(\mathcal{A}(\mathfrak{S}))$.*

Simplification of Accepting Runs. Let $\rho : T \to Q_{\mathcal{A}(\mathfrak{S})}$ be an accepting run. Here we keep the notations of Definition 1. For any $m \in \mathcal{M}$, let $k_m \in K$ be such that $\tau(\rho(r_{k_m})) = m$. Let us note that k_m is unique because K is assumed to be minimal. In particular $idx(\mathfrak{P}(\rho)) \leq |\mathcal{M}|$. In order to simplify notations, let t_m

[6] A tree provided with a deterministic tiling is not necessarily deterministic it-self. Indeed, several patterns could occur inside a particular tile.

(respectively r_m) denote t_{k_m} (respectively r_{k_m}). Let us define $\sigma : T \to \bigcup_m t_m$ the canonical extension of all the σ_i defined for any $n \in T$ by $\sigma(n) = \sigma_i(n)$ where t_i is the subtree of $\mathfrak{P}(\rho)$ which contains n. Let us now consider for any $m \in \mathcal{M}$ the mapping $\rho_m = \rho|_{t_m} : t_m \to Q_{\mathcal{A}_m} \subset Q_{\mathcal{A}(\mathfrak{S})}$. The family $\{\rho_m\}_{m\in\mathcal{M}}$ is denoted by Δ_ρ. Let $\lambda_\rho : I \to \mathcal{M}$ be defined by $\lambda_\rho(i) = \tau(\rho(r_i))$ for any $i \in I$. Finally, let us consider the mapping $\bar{\rho} : T \to Q_{\mathcal{A}(\mathfrak{S})}$ defined by $\forall n \in T : \bar{\rho}(n) = \rho_{\tau(\rho(n))}(\sigma(n))$.

Lemma 3. *$\bar{\rho}$ is an accepting run of $\mathcal{A}(\mathfrak{S})$, called a* simplified *accepting run.*

3.2 Tree Tilings and Rabin Automata

By Lemma 2, Problem 1 is equivalent to decide whether the intersection of the languages of a given composite automaton and a given Rabin automaton is empty or not. Here we study the runs of a Rabin automaton on trees which are recognized by some composite automaton. We focus on the proof of Lemma 4 below which aims at simplifying such runs. This is a corner-stone of the proof of the Theorem 1. Roughly speaking, it says that if a tree which has a deterministic tiling is recognized by a Rabin automaton, then there is a run which uses a bounded number of partial runs on each tile; the bound only depending on the number of tiles (and not on their sizes) and on the number of states of the automaton. This applies to trees which are simultaneously recognized by a composite automaton and a Rabin automaton.

Formally, let us consider a Rabin automaton $\mathcal{A}'$. Let $T \in \mathcal{L}(\mathcal{A}')$, let us assume that there exists a tiling $\mathcal{T} = \{t_i\}_{i\in I}$ of T defined by a tuple $(K, \lambda, (\sigma_i)_{i\in I})$. Let $\rho' : T \to Q_{\mathcal{A}'}$ be an accepting run of $\mathcal{A}'$ on T. Let us consider the collection of mappings of the form $\rho' \circ \sigma_i^{-1} : t_{\lambda(i)} \to Q_{\mathcal{A}'}$ for $i \in I$. Since all the t_k's are finite and their number is also finite, this collection of mappings is actually finite. Let us index it by a finite set J and denote it by $\Delta'_{\mathcal{T},\rho'} = \{\rho'_j : t_{k_j} \to Q_{\mathcal{A}'}\}_{j\in J}$ where $k_j \in K$. According to this notation, let $\lambda'_{\mathcal{T},\rho'} : I \to J$ be the mapping such that for any $i \in I$: $\rho'|_{t_i} \equiv \rho'_{\lambda'_{\mathcal{T},\rho'}(i)} \circ \sigma_i$. $|\Delta'_{\mathcal{T},\rho'}|$ is called the *regularity index* of ρ' regarding $\mathcal{T}$, it is denoted by $reg(\mathcal{T}, \rho')$.

Lemma 4. *Let $\mathcal{A}'$ be a Rabin automaton and let $T \in \mathcal{L}(\mathcal{A}')$ having a deterministic tiling $\mathcal{T}$. Then there exists an accepting run $\rho' : T \to Q_{\mathcal{A}'}$ on T such that $reg(\mathcal{T}, \rho')$ is bounded by $2^{2\times|Q_{\mathcal{A}'}|\times|\Sigma_{\mathcal{A}'}|\times\max\{idx(K),|Q_{\mathcal{A}'}|\}^{d_{\max}}\times 2^{|Q_{\mathcal{A}'}|}}$.*

The Game Point of View. Let us be given with $T \in \mathcal{L}(\mathcal{A}')$ and a deterministic tiling $\mathcal{T} = \{t_i\}_{i\in I}$ defined by a tuple $(K, \lambda, \{\sigma_i\}_{i\in I}, \{\delta_k\}_{k\in K})$. Let us consider a finite set of mappings $\Delta = \{\rho'_j : t_{k_j} \to Q_{\mathcal{A}'}\}_{j\in J}$ where $k_j \in K$ such that for any $j \in J$, ρ'_j is a partial run of $\mathcal{A}'$ on t_{k_j}. For each $j \in J$ and for each node x of t_{k_j}, let us consider the tuple $\xi_{j,x} = (q, \eta, w, p) \in Q_{\mathcal{A}'} \times \Sigma_{\mathcal{A}'} \times (K \cup Q_{\mathcal{A}'})^{*<d_{\max}} \times \mathcal{P}(Q_{\mathcal{A}'})$ defined as follows: $q = \rho'_j(x)$ and η is the label of x. The word w is obtained as follows: by definition of $\delta_{k_j}(x)$, for any position n such that $\delta_{k_j}(x)_n = \perp$, the n-th son of x is defined in t_{k_j}, let us denote it by x'. Then one puts $\rho'_j(x')$ at the n-th position in $\delta_{k_j}(x)$ in the place of $\perp$. Finally $p = \{\rho'_j(y) \mid y \in \alpha\}$ where α denotes the path of t_{k_j} starting at the root and ending at x. Let ξ_j denote the set of all the $(q, \eta, w, p) = \xi_{j,x}$ such that $x \in t_{k_j}$ and w contains at least one element of K.

For any transition $\tau = (q, \eta, (q_1, \ldots, q_{n'}))$ of $\mathcal{A}'$ and any $w \in (K \cup Q_{\mathcal{A}'})^{*<d_{\max}}$,

let $\chi_{\tau,w} \subset J^{*<d_{\max}}$ be the set of words of form $j_0 \dots j_n$ where n is the number of positions where some element of K occurs in w, and for any $\ell \in [\,1...n\,]$, $j_\ell \in J$ is such that k_{j_ℓ} is equal to the ℓ-th element of K occurring in w and $(q_1, \dots, q_{n'})$ is obtained from w by replacing the ℓ-th element of K occurring in w by $\rho'_{j_\ell}(r_{k_{j_\ell}})$.

Let $\mathcal{G}_\Delta$ be the finite two-players game defined by the finite game graph $\Gamma_\Delta = (V_0, V_1, E, c, C)$ and the winning ω-language W where $V_0 = Q_{\mathcal{A}'} \times \Sigma_{\mathcal{A}'} \times (K \cup Q_{\mathcal{A}'})^{*<d_{\max}} \times \mathcal{P}(Q_{\mathcal{A}'})$ is the set of positions for player$_1$, $V_1 = J^{*<d_{\max}}$ is the set of positions for player$_0$, the set of transitions E groups transitions of the form $(q, \eta, w, p) \to w'$ such that $w' \in \chi_{\tau,w}$ where τ is a transition of $\mathcal{A}'$ of origin (q, η), and transitions of the form $j_0 \dots j_n \to (q, \eta, w, p)$ such that $\exists e \in [\,1...n\,]$ such that $(q, \eta, w, p) \in \xi_{j_e}$, the colour set is $C = {Q_{\mathcal{A}'}}^*$ and $c : V \to C$ is defined by $c(v) = \varepsilon$ for any $v \in V_1$, and $c(q, \eta, w, p) = \tilde{p}$, where $\tilde{p} \in {Q_{\mathcal{A}'}}^*$ is defined to be the ordered sequence of all the elements of p according to a linear ordering of $Q_{\mathcal{A}'}$ fixed in advance once for all, finally, W is defined to be the language of infinite words accepted according to the acceptance condition of $\mathcal{A}'$ and starting at some initial state of $\mathcal{A}'$. Following the standard schema, we identify runs and strategies: Let ρ' be a run of $\mathcal{A}'$ on T. Let $\Delta'_{\mathcal{T},\rho'}$ and $\lambda'_{\mathcal{T},\rho'}$ be as defined page 8. Then ρ' defines a strategy $\mathcal{S}_{\rho'}$ in the game $\mathcal{G}_{\Delta'_{\mathcal{T},\rho'}}$ associated to $\Delta'_{\mathcal{T},\rho'}$. Here we define $\mathcal{S}_{\rho'}$ by describing how it drives a play: **1)** The first move of player$_0$ is $j_{0,0} = \rho'(\mathit{root}(T))$. **2)** Next, the first move of player$_1$ is defined as follows: According to the definition of $\mathcal{G}_{\Delta'_{\mathcal{T},\rho'}}$, player$_1$ picks a node $\bar{x}_1 \in t_{k_{j_{0,0}}}$ such that $\xi_{j_{0,0},\bar{x}_1} \in \xi_{j_{0,0}}$; and he plays the tuple $(q_1, \eta_1, w_1, p_1) = \xi_{j_{0,0},\bar{x}_1}$. **3)** Let us consider the play at the turn number $\ell \geq 1$: let $\gamma_\ell = j_{0,0}|\ (q_1, \eta_1, w_1, p_1)|\ \dots\ j_{\ell-1,0} j_{\ell-1,1} \dots j_{\ell-1,n_{\ell-1}}|\ (q_\ell, \eta_\ell, w_\ell, p_\ell)$ be the successive positions which have appeared before. By induction on $\ell \geq 0$, let us assume that **a)** there exists a sequence $t_{i_0}, \dots, t_{i_{\ell-1}}$ of ℓ elements of $\mathfrak{P}(\rho)$ such that i_0 is the root of I according to the natural tree structure of I, i.e., r_{i_0} is the root of T (cf. page 7), and for any $h \in [\,1...\ell-2\,]$: i_h is the father of i_{h+1}; in particular, the concatenation of all the α_h's is also a path of T; **b)** there exists a sequence of paths $\alpha_1 \subset t_{i_0}, \dots, \alpha_\ell \subset t_{i_{\ell-1}}$ such that for any $h \in [\,1...\ell\,]$, α_h starts at the root $r_{i_{h-1}}$ of $t_{i_{h-1}}$ and ends at a node x_h which is the father of r_{i_h} if $h \leq \ell - 1$; **c)** for any $h \in [\,0...\ell-1\,]$: there exists $e_h \in [\,1...n_h\,]$ such that $(q_{h+1}, \eta_{h+1}, w_{h+1}, p_{h+1}) = \xi_{j_{h,e_h}, \sigma_{i_h}(x_{h+1})}$.

The move of player$_0$. The strategy $\mathcal{S}_{\rho'}$ defined by ρ' is defined here: player$_0$ plays a word $j_{\ell,0} j_{\ell,1} \dots j_{\ell,n_\ell}$ defined as follows: n_ℓ is the number of elements of K occurring in w_ℓ. For any $h \in [\,1...n_\ell\,]$, let l_h be the position of the h-th element of K occurring in w_ℓ, and let $s_h \in I$ be such that r_{s_h} is h-th son of x_ℓ. Then for any $h \in [\,1...n_\ell\,]$, we define $j_h = \lambda'_{\mathcal{T},\rho'}(s_h)$. This move is correct: Let τ be the transition of $\mathcal{A}'$ used by ρ' at x_ℓ. Then by construction we have that η_ℓ is the label of x_ℓ and $q_\ell = \rho'(x_\ell)$. Therefore (q_ℓ, η_ℓ) is indeed the origin of τ. By a detailed checking of the definition of χ_{τ,w_ℓ}, one also can verify that $j_{\ell,0} j_{\ell,1} \dots j_{\ell,n_\ell} \in \chi_{\tau,w_\ell}$. The move is indeed correct.

The move of player$_1$. According to the definition of $\mathcal{G}_{\Delta'_{\mathcal{T},\rho'}}$, player$_1$ picks an integer $e_\ell \in [\,1...n_\ell\,]$ and a node $\bar{x}_{\ell+1} \in t_{k_{j_{\ell,e_\ell}}}$ such that $\xi_{j_{\ell,e_\ell},\bar{x}_{\ell+1}} \in \xi_{j_{\ell,e_\ell}}$; and he plays the tuple $(q_{\ell+1}, \eta_{\ell+1}, w_{\ell+1}, p_{\ell+1}) = \xi_{j_{\ell,e_\ell},\bar{x}_{\ell+1}}$.

Construction of $\alpha_{\ell+1}$ and t_{i_ℓ}. Let $\bar{\alpha}_{\ell+1}$ be the path going from the root of $t_{k_{j_{\ell,e_\ell}}}$ to $\bar{x}_{\ell+1}$. Let e'_ℓ be position in w_ℓ of the e_ℓ-th element of K occurring in it. Then i_ℓ is defined by the condition that r_{i_ℓ} is the e'_ℓ-th son of x_ℓ. Let us note that we have $\lambda(i_\ell) = (w_\ell)_{e'_\ell}$. Finally $\alpha_{\ell+1}$ is defined to be $\sigma_{i_\ell}^{-1}(\bar{\alpha})$. And $x_{\ell+1} = \sigma_{i_\ell}^{-1}(\bar{x}_{\ell+1})$. The induction hypothesis are obviously verified by construction.

Lemma 5. *For any run ρ' of $\mathcal{A}'$, if ρ' is an accepting run then $\mathcal{S}_{\rho'}$ is a winning strategy.*

Let us turn to the converse. Let $\mathcal{S}$ be a strategy in $\mathcal{G}_\Delta$. Let $j_0 = \mathcal{S}(\varepsilon)$. Let us suppose that $\rho'_{j_0}(r_{k_{j_0}})$ is an initial state of $\mathcal{A}'$. Such a strategy is said to *have the property $\mathcal{P}_{init}$.* Assuming that, $\mathcal{S}$ defines a run $\rho'_\mathcal{S}$ of $\mathcal{A}'$ such that $\Delta'_{\mathcal{T},\rho'_\mathcal{S}} \subset \{\rho'_j \in \Delta \mid j \in \mathcal{D}om(\mathcal{S})\}$. To define $\rho'_\mathcal{S}$, we shall construct the mapping $\lambda'_{\mathcal{T},\rho'_\mathcal{S}} : I \to J$ associated to $\rho'_\mathcal{S}$.

Let $i \in I$. Let β be the path starting at the root of T and ending at r_i, the root of t_i. Let $t_{i_0}, ..., t_{i_\theta}$ be the sequence of elements of $\mathfrak{P}(\rho)$ across which β goes; β ends at r_{i_θ}. For each $h \in [1...\theta]$, let $x_h \in t_{i_{h-1}}$ be the father of r_{i_h}. Then $x_h \in \beta$. Let e_h be such that r_{i_h} is the e_h-th son of x_h which does not belong to $t_{i_{h-1}}$; and let $e'_h \in [1...n_h]$ be such that r_{i_h} is the e'_h-th son of x_h. Let us consider the partial play defined as follows: According to $\mathcal{S}$, the first move of player$_0$ is $j_{0,0} = \mathcal{S}(\varepsilon)$. Then, inductively, let us consider the play at the turn number $\ell \le \theta$: let $\gamma_\ell = j_{0,0}|\ (q_1, \eta_1, w_1, p_1)| \ldots |(q_\ell, \eta_\ell, w_\ell, p_\ell)|\ j_{\ell,0} j_{\ell,1} \ldots j_{\ell,n_\ell}$ be the successive positions which have appeared before. We assume that the successive game positions $j_{0,0}|\ j_{1,0} j_{1,1} \ldots j_{1,n_1}| \ldots |j_{\ell,0} j_{\ell,1} \ldots j_{\ell,n_\ell}$ have been played by player$_0$ according to $\mathcal{S}$. If $\ell = \theta$, then the construction is finite. Otherwise, we define the next moves of player$_0$ and player$_1$: First, let us deal with the next move of player$_1$. Let us note that $\lambda(i_\ell) = k_{j_{\ell,e_\ell}}$. Therefore $\bar{x}_{\ell+1} = \sigma_{i_\ell}(x_{\ell+1})$ belongs to $t_{k_{j_{\ell,e_\ell}}}$. Then we define the next move of player$_1$ to be $(q_{\ell+1}, \eta_{\ell+1}, w_{\ell+1}, p_{\ell+1}) = \xi_{j_{\ell,e_\ell}, \bar{x}_{\ell+1}}$. Second, player$_0$ plays according $\mathcal{S}$: $j_{\ell+1,0} j_{\ell+1,1} \ldots j_{\ell+1,n_{\ell+1}} = \mathcal{S}((q_1, \eta_1, w_1, p_1) \ldots (q_{\ell+1}, \eta_{\ell+1}, w_{\ell+1}, p_{\ell+1}))$ By induction, we construct a play γ_θ. And finally, we set: $\lambda'_{\mathcal{T},\rho'}(i) = j_{\theta,e_\theta}$.

Lemma 6. *For any strategy $\mathcal{S}$ of player$_0$ in $\mathcal{G}_\Delta$, if $\mathcal{S}$ is a winning strategy and has the property $\mathcal{P}_{init}$, then $\rho'_\mathcal{S}$ is an accepting run.*

Lemma 7 (Strategy Reduction). *Let $\mathcal{G}$ be a game defined by a game graph $\Gamma = (V_0, V_1, E, c, C)$ with $|V_1|$ and $|C|$ finite. Let us suppose that there exists a winning strategy $\mathcal{S}$ for player$_0$ in $\mathcal{G}$. Then there exists an other strategy $\mathcal{S}'$ for player$_0$ in $\mathcal{G}$ such that $|\mathcal{S}'(\mathcal{D}om(\mathcal{S}'))| \le |c(V_1)| \times 2^{2 \times |V_0|}$. Moreover, $\mathcal{S}'$ can be computed in an effective way from $\mathcal{S}$.*

Proof of Lemma 4. Let us be given with an accepting run $\tilde{\rho}'$ on $T \in \mathcal{L}(\mathcal{A}')$. By Lemma 5, $\tilde{\rho}'$ defines a winning strategy $\mathcal{S}_{\tilde{\rho}'}$ for player$_0$ in $\mathcal{G}_{\Delta'_{\mathcal{T},\tilde{\rho}'}}$. Moreover, $\mathcal{S}_{\tilde{\rho}'}$ has the property $\mathcal{P}_{\text{init}}$. By Lemma 7, we can construct from $\mathcal{S}_{\tilde{\rho}'}$ a new winning strategy $\mathcal{S}'$ for player$_0$ such that $|\mathcal{S}'(\mathcal{D}om(\mathcal{S}'))|$ is bounded by $2^{2 \times |Q_{\mathcal{A}'}| \times |\Sigma_{\mathcal{A}'}| \times \max\{idx(K), |Q_{\mathcal{A}'}|\}^{d_{\max}} \times 2^{|Q_{\mathcal{A}'}|}}$. Taking into account Remark 8, one can assume that $\mathcal{S}'$ has the property $\mathcal{P}_{\text{init}}$. By Lemma 6, $\mathcal{S}'$ defines an accepting run $\rho_{\mathcal{S}'}$ which satisfies the conditions of Lemma 4 □

3.3 Decidability

Decidability results from a pumping property obtained from Lemma 4: if the langages of a composite automaton and a Rabin automaton are not disjoint, then there is a tree in their intersection which has a deterministic tiling with *bounded* tiles (Lemma 8 below). Deciding if such a tree exists is therefore possible by enumeration.

Let $(\mathcal{A}_m)_{m\in\mathcal{M}}$ and $\mathcal{A}(\mathfrak{S})$ be such as in Section 3.1. Let $T \in \mathcal{L}^\infty(\mathcal{A}(\mathfrak{S}))$. Let $\rho : T \to Q_{\mathcal{A}(\mathfrak{S})}$ be an accepting run of $\mathcal{A}(\mathfrak{S})$ on T, and let $\mathfrak{P}(\rho) = \{t_i\}_{i\in I}$ be the induced partition of T associated to ρ. $\mathfrak{P}(\rho)$ is a deterministic tiling, let $(K, \lambda, \{\sigma_i\}_{i\in I}, \{\delta_k\}_{k\in K})$ be a tuple associated to it. We also consider the families $\{t_m\}_{m\in\mathcal{M}}$ and $\Delta_\rho = \{\rho_m : t_m \to Q_{\mathcal{A}_m}\}_{m\in\mathcal{M}}$, and the mapping $\lambda_\rho : I \to \mathcal{M}$ as defined at page 8. Let $\mathcal{A}'$ be a Rabin automaton, let us assume that $T \in \mathcal{L}(\mathcal{A}')$. Let $\rho' : T \to Q_{\mathcal{A}'}$ be an accepting run satisfying Lemma 4 regarding the deterministic tiling defined by $\mathfrak{P}(\rho)$. We keep the notations of Section 3.3. Let us consider $\Delta'_{\mathfrak{P}(\rho),\rho'} = \{\rho'_j : t_{k_j} \to Q_{\mathcal{A}'}\}_{j\in J}$ defined in Section and the associated mapping $\lambda'_{\mathfrak{P}(\rho),\rho'} : I \to J$.

Path Collapsing. Let $\alpha = x_0...x_n$ be a depth-increasing path in T; x_0 is the higher node and x_n the lower one. *Collapsing* α in T consists in deleting T_{x_0} and gluing T_{x_n} in place of it in T. The resulting tree is denoted by $\zeta_\alpha(T)$. Let $\pi_\alpha : T \to \zeta_\alpha(T)$ be the associated projection. Let $\rho : T \to Q'_{\mathcal{A}}$ be a run. The operation of collapsing α in T is said to be *compatible* with ρ if $\rho(x_0) = \rho(x_n)$. If this holds, then ρ induces a run on $\zeta_\alpha(T)$ which shall be denoted by $\zeta_\alpha(\rho)$. For any family $\mathcal{P}$ of pairwise disjoint paths of T, possibly infinite, the operation of collapsing all the paths of $\mathcal{P}$ simultaneously in T, denoted by $\zeta_\mathcal{P}$, is well defined by considering the equivalence relation made of the union of all the equivalence relations (seen as subsets of $T\times T$) associated to the paths of $\mathcal{P}$: $\mathcal{R}_\mathcal{P} = \bigcup_{\alpha\in\mathcal{P}} \mathcal{R}_\alpha$. Let $\pi_\mathcal{P} : T \to \zeta_\mathcal{P}(T)$ denote the associated projection. If ζ_α is compatible with ρ for any $\alpha \in \mathcal{P}$, then one consider the canonical run $\zeta_\mathcal{P}(\rho)$ of $\mathcal{A}'$ on $\zeta_\mathcal{P}(T)$. Let α be a path of t_m for some fixed m. Such a path defines a family of pairwise disjoint paths in T defined by $\sigma^{-1}(\alpha) = \{\sigma_i^{-1}(\alpha) \subset T \mid \lambda_\rho(i) = m\}$. Let us consider the simultaneous collapsing of all the paths of $\sigma^{-1}(\alpha)$ in T, i.e., $\zeta_{\sigma^{-1}(\alpha)}$; let it be denoted by ζ_α^∞. We also consider the associated projection $\pi_\alpha^\infty : T \to \zeta_{\sigma^{-1}(\alpha)}(T)$ and the associated run $\zeta_\alpha^\infty(\rho)$. Similarly, ζ_α^∞ is compatible with ρ' if and only if ζ_α is compatible with ρ'_j for any $j \in J$ such that $\lambda_\rho(k_j) = m$. In this case $\zeta_\alpha^\infty(\rho')$ still is a run of $\mathcal{A}'$ on $\zeta_\alpha^\infty(T)$, but non necessarily accepting.

Lemma 8. *Let* $\mathcal{B}(\mathcal{M}, \mathcal{A}') = 2^{2\times|Q_{\mathcal{A}'}|\times|\Sigma_{\mathcal{A}'}|\times\max\{|\mathcal{M}|,|Q_{\mathcal{A}'}|\}^{d_{\max}}\times 2^{|Q_{\mathcal{A}'}|}}$. *If the height of* t_m *for some* $m \in \mathcal{M}$ *is greater than* $(\mathcal{B}(\mathcal{M}, \mathcal{A}')|Q_{\mathcal{A}'}| + 2)|Q_{\mathcal{A}_m}||Q_{\mathcal{A}'}|^{\mathcal{B}(\mathcal{M},\mathcal{A}')}$ *then there exists a depth-increasing path* $\alpha \subset t_m$ *such that* ζ_α^∞ *is compatible with* ρ *and* ρ'*, and such that* $\zeta_\alpha^\infty(\rho')$ *is a Rabin-accepting run of* $\mathcal{A}'$ *on* $\zeta_\alpha^\infty(T)$.

This lemma gives us a bound for enumeration and thus, allows us to decide Problem 3 in an effective way. This achieves the proof of Theorem 1.

References

1. M. Abadi and L. Lamport. Composing Specifications. *ACM Transactions on Prog. Lang. and Systems (TOPLAS)*, 15(1):73–132, 1993.

2. H. R. Andersen, C. Stirling, and G. Winskel. A compositional proof system for the modal mu-calculus. In *9th Symp. on Logic in Comp. Sci. (LICS'94)*, pages 144–153. IEEE Comp. Soc. Press, 1994.
3. G. Barthe, P. Courtieu, G. Dufay, M. Huisman, S. Mello de Sousa, G. Chugunov, L.-A. Fredlund, and D. Gurov. Temporal Logic and Toolset for Applet Verification: Compositional Reasoning, Model Checking, Abstract Interpretation. Technical report, VERIFICARD Project, `http://www.verificard.org/`, Sept 2002. Deliverable 4.1.
4. G. Barthe, D. Gurov, and M. Huisman. Compositional Verification of Secure Applet Interactions. In *Fundamental Approaches to Soft. Eng. (FASE'02)*, volume LNCS 2306, pages 15–32, 2002.
5. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree Automata Techniques and Applications. Technical report, LIFL – France, 2003. `http://www.grappa.univ-lille3.fr/tata/`.
6. B. Courcelle. The Monadic Second-order Logic of Graphs II : Infinite Graphs of Bounded Width. *Math. Syst. Theory*, 21:187–221, 1989.
7. M. Dam and D. Gurov. Compositional Verification of CCS processes. In *Proceedings of PSI'99*, volume LNCS 1755, pages 247–256, 1999.
8. F. Drewes, H.-J. Kreowski, and Habel A. Hyperedge Replacement Graph Grammars. In G. Rozenberg, editor, *Handbook of Graph Grammars and Computing by Graph Transformation*, volume 1, pages 95–162. World Scientific, 1997.
9. O. Grumberg and D. Long. Model Checking and Modular Verification. *ACM Trans. on Prog. Lang. & Syst.*, 16(3):843–871, 1994.
10. Y. Gurevich. Monadic Second-Order Theories. In J. Barwise and S. Feferman, editors, *Model Theoretic Logic*, pages 479–506. Springer, 1985.
11. J. Gustedt, O.A. Maehle, and J.A. Telle. The treewidth of java programs. In *Proc. of the 4th Workshop on Algorithm Engineering and Experiments (ALENEX'02), San Francisco*, 2002. To appear in LNCS.
12. T. Jensen, D. Le Métayer, and T. Thorn. Verifying Security Properties of Control-Flow Graphs. In *Proc. of the 20th Symposium on Security and Privacy, Berkeley*, pages 89–103. IEEE Computer Society Press, 1999.
13. Clarke E. M., D. E. Long, and K. L. McMillan. Compositional Model Checking. In *Proc. of the 4th Symp. on Logic in Comp. Sci. (LICS'89*, pages 353–362, 1989.
14. D. E. Muller and P. E. Schupp. The theory of ends, pushdown automata, and second-order logic. *Theoretical Comp. Sci.*, 37:51–75, 1985.
15. Kedar S. Namjoshi and Richard J. Trefler. On the completeness of compositional reasoning. In *Proc. of the 12th Int. Conference on Computer Aided Verification (CAV'00)*, number 1855, pages 139–153. Springer-Verlag, 2000.
16. M. O. Rabin. Decidability of Second-Order Theories and Automata on Infinite Trees. *Trans. of the A.M.S.*, 141:1–35, 1969.
17. N. Robertson and P. Seymour. Some New Results on the Well-Quasi Ordering of Graphs. *Annals of Discrete Math.*, 23:343–354, 1984.
18. C. Sprenger, D. Gurov, and M. Huisman. Simulation Logic, Applets and Compositional Verification. Technical Report 4890, INRIA, 2003.
19. C. Stirling. A complete compositional modal proof system for a subset of CCS. *LNCS*, 194:475–486, 1985.
20. W. Thomas. Languages, automata, and logic. In *Rozenberg and Salomaa ed., Handbook of Formal Languages. Springer Verlag*, volume 3, pages 389–455, 1997.
21. M. Thorup. All structured programs have small tree-width and good register allocation. *Inf. and Comp.*, 142(2):159–181, 1998.