RWTH Aachen

Department of Computer Science
Technical Report

Transforming structures by set interpretations

Thomas Colcombet and Christof Löding

ISSN 0935–3232 · Aachener Informatik Berichte · AIB-2006-07

RWTH Aachen · Department of Computer Science · May 22, 2006

The publications of the Department of Computer Science of *RWTH Aachen University* are in general accessible through the World Wide Web.

http://aib.informatik.rwth-aachen.de/

Transforming structures by set interpretations

Thomas Colcombet[1] and Christof Löding[2]

[1] Cnrs-Irisa
Irisa, Campus de Beaulieu
35042 Rennes, France
Thomas.Colcombet@irisa.fr
[2] Christof Löding
RWTH Aachen
loeding@informatik.rwth-aachen.de

Abstract. We consider a new kind of interpretation over relational structures: finite sets interpretations. Those interpretations are defined by weak monadic second-order (WMSO) formulas with free set variables. They transform a given structure into a structure with a domain consisting of finite sets of elements of the orignal structure. The definition of these interpretations directly implies that they send structures with a decidable WMSO theory to structures with a decidable first-order theory. In this paper, we investigate the expressive power of such interpretations applied to infinite deterministic trees. The results can be used in the study of automatic and tree-automatic structures.

1 Introduction

Computational model theory is concerned with the study of algorithmic properties of classes of infinite structures (cf. [BG04]), where the focus is on the problem of model checking such structures against specifications written in some logic, i.e., deciding for a given structure and logical formula if the formula holds in this structure. This problem setting has been studied for various instantiations of the two parameters, i.e., the way to represent the structures, and the logic to write the specifications. The most prominent logics in this context are first-order (FO) logic and monadic second-order (MSO) logic, and they have led to two tracks of research trying to identify classes of structures for which the respective logic is decidable.

One way of defining such classes uses, e.g., words or trees for representing the elements of the structure and uses simple transformations based on transducers or rewriting to define the relations of the structure. In this way, one obtains, e.g., the classes of automatic [Hod83,KN95,BG00] and tree-automatic [DT90,BG00] structures, for which the FO-theory is decidable, and the classes of pushdown-graphs [MS85] and prefix recognizable graphs [Cau96] with decidable MSO-theory.

In the case of automatic structures, the decidability results are based on the strong closure properties of finite automata, which are used to define the relations. Other techniques, e.g. in [KL02] for rewriting in trace monoids, are based on Gaifman's locality theorem.

The decidability results for MSO logic on pushdown and prefix recognizable graphs are derived from the results of Büchi and Rabin establishing the equivalence of monadic second-order logic with certain families of finite automata accepting infinite trees (cf. [Tho97]). These results are also underlying the more recent work [CT02] and [KNUW05] showing the decidability of MSO logic over certain classes of infinite words and infinite trees, respectively.

A different and more systematic approach for defining and studying classes of infinite structures is to use operations for transforming structures. An important operation of this kind is the model-theoretic interpretation. Such an interpretation defines a new structure 'inside' a given one by means of logical formulas describing the domain and the new relations. Depending on whether these defining formulas are FO or MSO one speaks of FO- and MSO-interpretations. An important property of these interpretations is that decidability results easily transfer from the given structure to the resulting structure, i.e., applying an FO-interpretation to a structure with a decidable FO-theory results in a structure with decidable FO-theory, and similarly for MSO.

As mentioned in [BG04], this suggests a new way of defining interesting classes of infinite structures: fix an underlying structure (with good algorithmic properties) and consider all structures that can be obtained by applying interpretations of a certain kind. In this way, one obtains the automatic structures by FO-interpretations from, e.g., a suitable extension of Presburger arithmetic [Blu99], and the prefix recognizable structures by MSO-interpretations from the infinite binary tree [Blu01]. This idea has been pursued further in [Cau02], where MSO-interpretations and unravelling of graphs are iterated, leading to an infinite hierarchy of graphs (or structures) with a decidable MSO-theory.

All the methods and results described so far can be separated into those concerned with FO logic (sometimes extended by a reachability relation [DT90,Col02]) and those dealing with MSO logic (sometimes with only restricted kind of set quantification as in [Mad03]). To our knowledge, there has been no systematic work on relating these two areas. In this paper, we bridge this gap by studying a new kind of interpretation, named finite sets interpretation, allowing to define classes of structures with decidable FO-theory from structures with decidable MSO-theory. To be more precise, we are considering weak MSO (WMSO) logic, i.e., MSO logic where quantification is restricted to finite sets. The idea for these interpretations is rather simple: the domain of the new structure does not consist of elements of the old structure but of finite sets of elements of the old structure. The relations are specified by WMSO-formulas with free set variables (the number of which corresponds to the arity of the relation). In this way, FO-formulas over the new structure can directly be translated into WMSO-formulas over the old structure.

Using the equivalence of WMSO logic and finite automata (over finite words and trees) it is not difficult to see that the classes of automatic and tree automatic structures can be obtained by finite sets interpretations from the naturals with successor relation, and from the infinite binary tree, respectively (see [Col04], and [Rub04] for the result on automatic structures).

This raises the question of what happens when we apply finite sets interpretations to other structures with decidable WMSO-theory, e.g., the structures from the hierarchy defined in [Cau02]. Though this hierarchy is strict, it is not a priori clear whether this is also true for the hierarchy obtained after applying finite sets interpretations. To answer questions of this kind one has to study the expressiveness of finite sets interpretations and to provide tools for showing that a structure cannot be obtained by such an interpretation applied to a given structure. By the relations mentioned above, results of this kind then also allow to answer questions on automatic structures.

The main result of this paper is of the above kind. It allows to reduce questions on definability by finite sets interpretations to questions on WMSO-interpretability, a notion that has already been studied intensively and is well understood. A precise formulation of our main result (in its simplest form) reads as follows: If the class of structures definable by finite sets interpretations from a structure $\mathcal{S}$ is included in the class of structures definable by finite sets interpretations from a tree t, then $\mathcal{S}$ is WMSO-interpretable in t.

This means that if we can show that $\mathcal{S}$ is not WMSO-interpretable in t, then there are structures that cannot be obtained by a finite sets interpretation from $\mathcal{S}$ but from t. A more technical formulation of the main result also explicitly gives such a structure.

We demonstrate the use of this result by showing some non-definability results, the strictness of the hierarchy mentioned above, and a result on intrinsic definability of relations related to similar questions studied for automatic structures (cf. [Bár06]).

The remainder of the paper is structured as follows. In Section 2 we give the basic definitions and introduce finite sets interpretations, and in Section 3 we give the connection to automatic structures. Section 4 is devoted to the study of finite sets interpretations applied to trees. In particular, the main result is stated in this section. In Section 5 we present some applications of our results, and Section 6 is devoted to the proof of the main result.

2 Definitions and elementary results

In this section we provide the basic definitions used in the paper, i.e., relational structures, trees, logic, automata, and finally interpretations, the main subject of this work. We end this section by giving some elementary results on finite sets interpretations.

2.1 Structures and trees

We consider (relational) *structures* $\mathcal{S} = (\mathcal{U}, R_1, \ldots, R_N)$ where $\mathcal{U}$ is the universe of the structure and for each i, $R_i \subseteq \mathcal{U}^{r_i}$ is a *relation* of *arity* r_i for a natural number r_i. The names of the R_i together with their arities form the *vocabulary* (or signature) of the structure. Trees, as defined below, can be seen in a natural way as particular instances of such structures.

We will be dealing with infinite binary labeled trees. From now, we simply write 'trees'. Formally, a tree labeled by a finite alphabet Σ is a partial mapping $t : \{0,1\}^* \to \Sigma$ with prefix closed domain $dom(t)$, and such that if $u1 \in dom(t)$ then also $u0 \in dom(t)$. The elements of the domain are called *nodes*. A node u such that $u0$ is also a node is called an *inner node*, else it is called a *leaf*. By $\sqsubseteq$ we denote the prefix ordering on nodes, also called the *ancestor* order. For technical simplifications we will mostly consider purely binary trees, i.e. such that every node is either a leaf or has two sons.

Seen as a structure a tree labeled by Σ has as universe the domain of the tree and contains the following relations: the unary relations S_0 and S_1 meaning 'being a left successor (resp. a right successor)' (for $i \in \{0,1\}$, $S_i(u,v)$ holds if $v = ui$) and for each $a \in \Sigma$ a unary relation a interpreted as the set of elements sent to a by t.

We will be considering two particular infinite trees, namely Δ_1 and Δ_2. The tree Δ_1 is the unlabeled tree of domain 0^*. We will identify in a natural way this tree with the structure $(\mathbb{N}, succ)$. The tree Δ_2 is the unlabeled tree of domain $\{0,1\}^*$, also called the *infinite binary tree*.

2.2 Logic and automata

We use the standard definitions for first-order (FO) and weak monadic second-order (WMSO) logic. FO-formulas are built up from atomic formulas using first-order variables (interpreted by elements of the structure and usually denoted by letters x, y, z) and the relation symbols from the vocabulary under consideration. Complex formulas are constructed using boolean connectives and quantification over first-order variables.

For WMSO-formulas one can additionally use monadic second-order variables (interpreted by *finite* sets of elements of the structure and usually denoted by capital letters X, Y, Z), quantification over such variables, and the membership relation $x \in X$. If the variables are interpreted by arbitrary sets instead of finite sets, then we speak of MSO.

In order to deal with WMSO-formulas on trees, we use automata. Those automata are more general than WMSO-formulas since they have the expressiveness of full monadic second-order logic on trees. But for our purpose this doesn't harm because we only use the translation in one direction, namely from formulas to automata.

Technically, we use *nondeterministic parity automata* (or simply *automata*), which are tuples $(\Sigma, Q, q^{\text{in}}, \delta, \Omega)$ with a finite set Q of *states*, *initial state* q^{in}, transition relation $\delta \subseteq Q \times Q \times \Sigma \times Q \uplus \Sigma \times Q$, and *priority mapping* $\Omega : Q \to \mathbb{N}$. Given a tree t and an automaton, a *run* of this automaton on t is a mapping $\rho : dom(t) \to Q$ such that $(\rho(u0), \rho(u1), t(u), \rho(u)) \in \delta$ for each inner node u, and $(t(v), \rho(v)) \in \delta$ for every leaf v. A run is *accepting* if $\rho(\epsilon) = q^{\text{in}}$ and for all infinite branches (maximal totally ordered sequences of nodes) $v_1, v_2, \ldots$, $\liminf_i \Omega(\rho(v_i))$ is even. We say that a tree t is accepted by an automaton if there is an accepting run of this automaton on t. For basic properties of such automata (such as closure under the Boolean operations) and their relation to logic, we refer the reader to [Tho97].

We are interested here in automata running on a fixed underlying tree t with additional markings representing (tuples of) subsets of its domain. To mark a certain subset X of $dom(t)$ we can put additional labels on the tree. Formally, the tree t labeled by X is the tree with the same domain as t and labels from $\Sigma \times \{0,1\}$, where a node u is labeled by the pair $(t(u), 0)$ if $u \notin X$ and $(t(u), 1)$ if $u \in X$. In the same way one can also label a tree by tuples of subsets of its domain using a separate $\{0,1\}$-component for each set.

If t is fixed and $X_1, \ldots, X_n$ are subsets of $dom(t)$, then we say that an automaton accepts the tuple $(X_1, \ldots, X_n)$ if it accepts t with the additional labelings corresponding to the tuple $(X_1, \ldots, X_n)$ as explained above. If we consider all the tuples accepted by an automaton, we obtain a relation over the subsets of $dom(t)$. We call this relation the relation recognized or accepted by the automaton.

A WMSO-formula with free set variables $X_1, \ldots, X_n$ also defines a relation over the subsets of $dom(t)$. Throughout the paper we make use of the following result stating that for each WMSO-formula there is an equivalent automaton. The proof of this can easily be inferred from the equivalence of MSO and automata over trees and from the fact that over trees each WMSO-formula can be translated into an equivalent MSO-formula.

Theorem 1 (cf. [Tho97]) *For each WMSO-formula there is an automaton such that for each tree t the relation over subsets of dom(t) defined by the formula is the same as the one accepted by the automaton.*

2.3 Interpretations

Interpretations are a standard tool in logic allowing to define transformations of structures by means of logical formulas. This technique allows easy transfer of theories from one structure onto another.

Definition 2 *An interpretation is a tuple $(\delta, \Phi_{R_1}, \ldots, \Phi_{R_N})$ with formulas δ and $\Phi_{R_1}, \ldots, \Phi_{R_N}$. The formula δ has only one free variable, and each formula Φ_{R_i} has r_i free variables. By our convention, for weak monadic variables we use capital letters X and $X_1, \ldots, X_{r_i}$, and small letters x and $x_1, \ldots, x_{r_i}$ in the case of first-order variables.*

An interpretation is FO if the formulas are first-order (and hence the free variables are also of first-order). An interpretation is WMSO if the formulas are weak monadic and the free variables are first-order. An interpretation is finite sets *if the formulas are weak monadic and the free variables are weak monadic.*

The application of an interpretation to a structure is defined in the standard way. The only difference is that for finite sets interpretations the elements of the obtained structure are subsets of the universe of the original structure instead of elements of the original structure. Formally, given a structure $\mathcal{S}$ and an interpretation $\mathcal{I} = (\delta, \Phi_{R_1}, \ldots, \Phi_{R_N})$, the structure $\mathcal{I}(\mathcal{S})$ has for universe

- $\{u \in \mathcal{U}^{\mathcal{S}} \ : \ \mathcal{S} \models \delta(u)\}$ if $\mathcal{I}$ is a FO or WMSO interpretation,
- $\{U \subseteq \mathcal{U}^{\mathcal{S}} \ : \ U \text{ finite}, \ \mathcal{S} \models \delta(U)\}$ if $\mathcal{I}$ is a finite sets interpretation,

and the interpretation of each symbol R_i is defined by

$$R_i = \{(U_1, \ldots, U_{r_i}) \in (\mathcal{U}^{\mathcal{I}(\mathcal{S})})^{r_i} \ : \ \mathcal{S} \models \Phi_i(U_1, \ldots, U_{r_i})\}.$$

One can note at this point that natural sets interpretations can as well be defined in a similar way, simply by removing the finiteness hypothesis on sets and using monadic second-order logic. This tool turns out to contain significant extra subtleties. We prefer in this work to concentrate on the finite sets interpretations, though some of the result presented here can be directly transfered to the more general framework of sets interpretations.

Example 1. We show how to obtain the structure $(\{0,1\}^*, S_0, S_1, \sqsubseteq, el)$, i.e., the infinite binary tree extended with the prefix and equal level relations, by a finite sets interpretation from the infinite unary tree Δ_1, i.e., from the natural numbers with successor $\Delta_1 = (\mathbb{N}, succ)$. To realize this we have to code the nodes of the tree by finite sets of natural numbers and to describe the relations S_0 (for left successor), S_1 (for right successor), $\sqsubseteq$ (for prefix), and *el* (for equal level) by means of WMSO formulas.

The coding of the nodes is depicted in Figure 1. A node $u \in \{0,1\}^*$ is represented by the set of positions corresponding to letter 1 in u and additionally by its length. For example, the node 100 is coded by $\{0,3\}$ because its length is 3 and only position 0 is labeled 1. We now define the finite sets interpretation

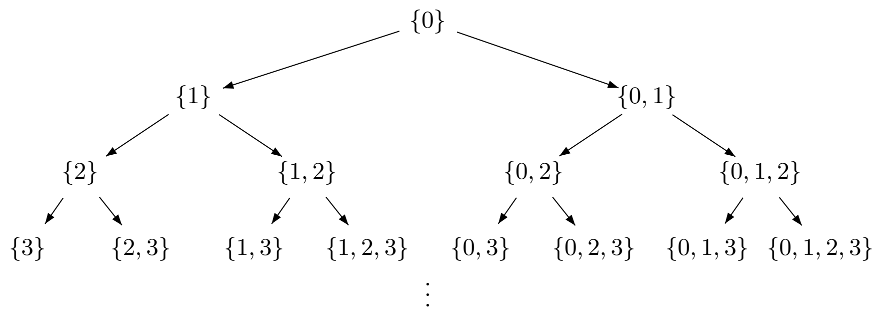

Fig. 1. The nodes of the infinite binary tree Δ_2 coded by sets of natural numbers

$\mathcal{I} = (\delta, \Phi_{S_0}, \Phi_{S_1}, \Phi_{\sqsubseteq}, \Phi_{el})$ such that $\mathcal{I}(\Delta_1)$ yields the binary tree depicted in Figure 1 together with the relations $\sqsubseteq$ and *el*. In the formulas we use abbreviations like $<$ and max that can easily be defined by WMSO-formulas in Δ_1.

- $\delta(X) := \exists x (x \in X)$ (all finite sets except $\emptyset$ are used in the coding).
- $\Phi_{\sqsubseteq}(X_1, X_2) := \max(X_1) < \max(X_2) \wedge \forall x (x < \max(X_1) \rightarrow (x \in X_1 \leftrightarrow x \in X_2))$.
- $\Phi_{S_0}(X_1, X_2) := \Phi_{\sqsubseteq}(X_1, X_2) \wedge \max(X_2) = \max(X_1) + 1 \wedge \max(X_1) \notin X_2$.
- $\Phi_{S_1}(X_1, X_2) := \Phi_{\sqsubseteq}(X_1, X_2) \wedge \max(X_2) = \max(X_1) + 1 \wedge \max(X_1) \in X_2$.
- $\Phi_{el}(X_1, X_2) := \max(X_1) = \max(X_2)$

Let us proceed with some elementary considerations. Obviously, finite sets interpretations are not closed under composition. But, as stated in the following proposition, applying an FO-interpretation after, or a WMSO-interpretation before a finite sets interpretation, does not give more expressive power.

Proposition 3 *Let $\mathcal{I}_1$ be a FO-interpretation, $\mathcal{I}_2$ be a WMSO-interpretation, and $\mathcal{I}$ be a finite sets interpretation. Then $\mathcal{I}_1 \circ \mathcal{I}$ and $\mathcal{I} \circ \mathcal{I}_2$ are effectively finite sets interpretations.*

Proof. As for standard interpretations. □

A straightforward as well as essential consequence of this is expressed in the following corollary.

Corollary 4 *The image of a structure of decidable WMSO-theory by a finite sets interpretation has a decidable FO-theory.*

To finish our elementary considerations on finite sets interpretations, we present Proposition 6 which is a form of converse to Proposition 3. It states that every finite sets interpretation can be described as the composition of a specific one, called the weak powerset interpretation, and a first-order interpretation.

Definition 5 *Let $\mathcal{P}^W$ be the finite sets interpretation that sends every structure $\mathcal{S}$ of signature Σ onto a structure of signature $\Sigma \cup \{\preceq\}$ where $\preceq$ is a new binary symbol, such that*

- *the universe is the set of finite subsets of the universe of $\mathcal{S}$,*
- *each symbol R in Σ has the same interpretation as in $\mathcal{S}$ but over singletons instead of elements,*
- *the interpretation of $\preceq$ corresponds to the subset ordering.*

The interpretation $\mathcal{P}^W$ is called the weak powerset *interpretation.*

This interpretation allows to reconstruct all other finite sets interpretations as stated in the following proposition which is obtained by a simple syntactic translation of formulas.

Proposition 6 *For each finite sets interpretation $\mathcal{I}$ there exists a FO-interpretation $\mathcal{I}_1$ such that $\mathcal{I}_1 \circ \mathcal{P}^W = \mathcal{I}$.*

We can remark that the adaption to MSO instead of WMSO of all the results from this section is straightforward.

3 Automatic-like structures

The line of research that has inspired finite sets interpretations is the one of automatic structures. Automatic structures in their common acceptation are structures with a universe consisting of a regular languages of words, and the relations defined by half-synchronized transducers. The key reason for introducing such structures is that — thanks to the good closure properties of finite automata — they naturally possess decidable first-order theories.

Classical variants of those structures consider universes consisting of infinite words (ω-automatic structures), or consisting of trees, finite or infinite (namely the tree-automatic and ω-tree-automatic structures). Some definitions also allow to quotient the structure by a congruence (that is defined in the same way as the other relations). This extension does not increase the expressiveness of automatic nor tree-automatic structures (cf. Corollary 9 below). The question whether quotienting increases the expressive power of ω-automatic and ω-tree-automatic structures is open.

Historically, automatic as well as ω-automatic structures have been introduced by Hodgson [Hod83]. Khoussainov and Nerode introduce the notion of automatically presentable theory [KN95], starting the study of definability in automatic structures. The extension to tree-automatic structures can be traced back, in a different framework, to the work of Dauchet and Tison [DT90]. Blumensath and Grädel [Blu99,BG00] then formalize the notion of tree-automatic structure and add to it the family of ω-tree-automatic structures. Independently, the study of 3-manyfolds lead to the particular case of automatic groups [ECH$^+$92].

3.1 Word-automatic structures

In the case of words a relation $R \subseteq (\Sigma^*)^r$ is automatic if there is a finite automaton accepting exactly the tuples $(w_1, \ldots, w_r) \in R$, where the automaton reads all the words in parallel with the shorter words padded with a dummy symbol $\diamond$. Formally, for $w_1, \ldots, w_r \in \Sigma^*$ we define

$$w_1 \otimes \cdots \otimes w_r = \begin{bmatrix} a'_{11} \\ \vdots \\ a'_{r1} \end{bmatrix} \cdots \begin{bmatrix} a'_{1n} \\ \vdots \\ a'_{rn} \end{bmatrix} \in (\Sigma_\diamond^r)^*$$

where $\Sigma_\diamond = \Sigma \cup \{\diamond\}$, n is the maximal length of one of the words w_i, and a_{ij} is the jth letter of w_i if $j \leq |w_i|$ and $\diamond$ otherwise. A language $L \subseteq ((\Sigma \cup \{\diamond\})^r)^*$ defines a relation $R_L \subseteq (\Sigma^*)^r$ in the obvious way: $(w_1, \ldots, w_r) \in R_L$ iff $w_1 \otimes \cdots \otimes w_r \in L$. A tuple $(L, L_1, \ldots, L_n)$ of languages $L \subseteq \Sigma^*$ and $L_i \subseteq (\Sigma_\diamond^{r_i})^*$ defines a structure of universe L with the relations R_{L_i} of arity r_i. A structure $\mathcal{S} = (\mathcal{U}, R_1, \ldots, R_n)$ is automatic if it is isomorphic to a structure of the above kind for regular languages $L, L_1, \ldots, L_n$.

The class of ω-automatic structures is defined in the same way with infinite words instead of finite ones. In this case the definition is even simpler as there is no need for padding shorter words.

3.2 Tree-automatic structures

To define tree-automatic structures we need a way to code tuples of finite trees, i.e., we need an operation $\otimes$ for finite trees. For a tree $t : dom(t) \to \Sigma$ let $t^\diamond : \{0,1\}^* \to \Sigma_\diamond$ be defined by $t^\diamond(u) = t(u)$ if $u \in dom(t)$, and $t^\diamond(u) = \diamond$ otherwise. For finite Σ-labeled trees $t_1, \ldots, t_r$ we define the $\Sigma_\diamond^r$-labeled tree $t = t_1 \otimes \cdots \otimes t_r$ by $dom(t) = dom(t_1) \cup \cdots \cup dom(t_r)$ and $t(u) = (t_1^\diamond(u), \ldots, t_r^\diamond(u))$. When viewing words as unary trees, this definition corresponds to the operation $\otimes$ as defined for words. As in the case of words a set T of finite $\Sigma_\diamond^r$-labeled trees defines the relation R_T by $(t_1, \ldots, t_r) \in R_T$ iff $t_1 \otimes \cdots \otimes t_r \in T$. A structure is called tree-automatic if it is isomorphic to a structure given by a tuple $(T, T_1, \ldots, T_n)$ of regular tree languages in the same way as for words.

Again, the definitions for ω-tree-automatic structures are the same with ω-trees, i.e., trees of domain $\{0,1\}^*$ instead of finite trees.

One should note here that we only consider so called injective presentations of automatic structures. A more general definition as, e.g., in [Blu99] additionally uses a regular language $L_\sim \subseteq (\Sigma_\diamond^2)^*$ defining an equivalence relation identifying words representing the same element of the structure (and similarly for the other variants of automatic structures). It is known that injective presentations are sufficient for automatic structures [KN95] meaning that all structures that are automatic in the more general sense are also automatic according to our definition. The corresponding result for tree-automatic structures is established below, see Corollary 9.

3.3 Automaticity via interpretations

Recall that Δ_1 is the (unlabeled) infinite unary tree, i.e., the natural numbers with successor, and that Δ_2 is the (unlabeled) infinite binary tree. The following fact is a straightforward consequence of the definition of automatic structures and of the equivalences between WMSO-logic and automata. The first claim also appears in [Rub04].

Proposition 7 *The following holds up to isomorphism*

- *A structure is* automatic *iff it is finite sets interpretable in* Δ_1*.*
- *A structure is* tree-automatic *iff it is finite sets interpretable in* Δ_2*.*

Note that the ω-automatic and ω-tree-automatic structures satisfy the same equivalences where finite sets interpretations are replaced by sets interpretations.

4 Finite sets interpretations on trees

The power of finite sets interpretations makes it difficult to obtain results for the general case where such interpretations are applied to arbitrary structures. We consider here the special case of finite sets interpretations applied to deterministic trees.

This restriction can be justified in two ways. The first justification is that on trees there are specific tools suitable for treating WMSO questions: their automata equivalents. The second justification is that if Seese's conjecture [See91] holds — stating that all structures of decidable weak monadic second-order theory are WMSO-interpretations of trees — then the only structures that we can prove to have decidable first-order theory using Corollary 4 are finite sets interpretations of trees (for recent work on Seese's conjecture see [CO06]).

We give here two results concerning finite sets interpretations applied to deterministic trees. The first one — in Subsection 4.1 — shows that finite sets interpretations on deterministic trees followed by a quotient are simply finite sets interpretations. The second result — subject of Subsection 4.2 — concerns finite sets interpretations applied to trees leading to powerset lattices. The technical core of the proof is given in Section 6.

4.1 Quotienting finite sets interpretations on trees

We show here that if a structure is finite sets interpretable in a deterministic tree containing a symbol interpreted as a congruence on the structure, then it is possible to directly obtain the quotiented structure by a finite sets interpretation.

A *congruence* on a structure $\mathcal{S}$ is an equivalence relation $\sim$ such that for every symbol R of arity n and all elements $x_1, \ldots, x_n, y_1, \ldots, y_n$ of $\mathcal{S}$: if $x_1 \sim y_1, \ldots, x_n \sim y_n$, then $R^{\mathcal{S}}(x_1, \ldots, x_n)$ iff $R^{\mathcal{S}}(y_1, \ldots, y_n)$. We say that a symbol is a congruence if its interpretation in the structure is a congruence. For a congruence $\sim$ over a structure $\mathcal{S}$, we denote by $\mathcal{S}/_{\sim}$ the *quotient structure*, i.e., the structure which has as elements the equivalence classes of $\sim$, and the relations of which are the images of the relations on $\mathcal{S}$ under the canonical surjection induced by $\sim$.

Note that the operation of quotienting preserves the decidability of the first-order theory. For this reason we may wonder if constructing a structure by a finite sets interpretation followed by a quotient is more powerful than solely using a finite sets interpretation. Theorem 8 below shows that it is not the case when the original structure is a deterministic tree.

Theorem 8 *Given a finite sets interpretation $\mathcal{I}$, there exists a finite sets interpretation $\mathcal{I}'$ such that for every deterministic tree t, if the symbol $\sim$ is a congruence in $\mathcal{I}(t)$, then $\mathcal{I}'(t)$ is isomorphic to $\mathcal{I}(t)/_\sim$.*

Proof. Let $\mathcal{A}$ be a nondeterministic parity automaton corresponding to the formula $\Phi_\sim$ describing $\sim$ in $\mathcal{I}$. This automaton works on t additionally labeled by a pair (X, Y) of sets of nodes. We say that the automaton reads (X, Y).

For S a prefix closed subset of $dom(t)$, call its *frontier* the set of minimal nodes not in S. Let X be an element of the structure $\mathcal{I}(t)$, i.e., a finite set of nodes of t. Let us call its *shadow* $S(X)$ to be the least set of nodes containing X that is closed by prefix and such that no element of its frontier is the root of a finite subtree. This set happens to be finite. Now, let us consider an equivalence class c for $\sim$. Define the *shadow* $S(c)$ of the class c to be the intersection of the shadows of all the elements in the class. This is also a finite set of nodes of t. Let us call frontier of the class the frontier of its shadow. Denote by $F(c)$ the frontier of the class c.

Given an element X, its description is the triple (F, Y, f), where $F = F(c)$ for the class c of X, Y is $X \cap S(c)$, i.e., X restricted to the shadow of its class, and f maps each node $x \in F$ to the set of states q such that there is an accepting run of $\mathcal{A}$ on the subtree rooted in x starting with state q and reading $(\emptyset, X)$.

We claim that if two elements X and X' share the same description — say (F, Y, f) — then those elements are equivalent for $\sim$. Using the transitivity of $\sim$ and the finiteness of F it is sufficient to consider the case of elements coinciding everywhere but below one x in F. Since x is in the frontier of the class of X, there is an element Z equivalent to X such that Z does not contain any node below x. Since Z is equivalent to X, there is an accepting run ρ of $\mathcal{A}$ on t labeled by (Z, X). We aim at constructing a run of $\mathcal{A}$ witnessing the equivalence of Z and X', i.e., a run accepting t labeled by (Z, X'). This new run is constructed in the following way. On every element not below x, it coincides with ρ. This is a valid part of run since X and X' do coincide on this area. On the subtree rooted in x, Z coincides with $\emptyset$. Hence, as (F, Y, f) is a description of X, $\rho(x)$ belongs to $f(x)$. But as the same description holds also for X', there is a piece of run below x starting with $\rho(x)$ and accepting $\emptyset, X'$. We complete our new run by this piece of run. This new run witnesses as expected that $Z \sim X'$. It follows by symmetry and transitivity of $\sim$ that $X \sim X'$. This concludes the proof of the claim.

Let us remark now that a description (F, Y, f) can be encoded uniquely by a set of nodes: this set is $F \cup Y \cup \mathit{Coding}(f)$ where $\mathit{Coding}(f)$ contains exactly one element for each element x of the frontier, and this element is located in a place uniquely describing the value of $f(x)$ (e.g. the only node at distance $g(f(x))$ of x on the leftmost infinite branch below x where g is a numbering of 2^Q starting from 1). This is possible since the subtree rooted in x is infinite, and consequently, there is "room" below x for coding the information $f(x)$.

Note that associating to an element X the coding of its description is doable by means of a WMSO-formula. Note also that given a class c there is only a finite number of descriptions for the elements it contains. Hence we can chose the smallest description — smallest for a suitable total order — as unique representative for the class. From here, it is not difficult to reconstruct $\mathcal{I}(t)/_\sim$. $\square$

In combination with Proposition 7 we obtain the following.

Corollary 9 *Tree-automatic structures are effectively closed under quotient.*

In the terminology of [Blu99], this result is rephrased as "every tree-automatic structure admits an injective presentation." Let us remark that this result is announced in [Blu99], but unfortunately the proof proposed there contains an unrecoverable error.

4.2 Finite sets interpretations of powerset lattices

In this section we present our main result. For this, let $\mathcal{S}$ be a structure of signature $\preceq$, we say that $\mathcal{S}$ is a *finite powerset lattice* if it is isomorphic to $(\mathcal{P}^F(E), \subseteq)$ for some set E, where $\mathcal{P}^F(E)$ represents the finite subsets of E. Such a finite powerset lattice can be seen as a particular case of weak powerset generator applied to a vocabulary-free structure. We call *atoms* the elements corresponding to singletons in this isomorphic structure, i.e., the elements which have exactly one element strictly smaller with respect to $\preceq$.

Theorem 10 *For every finite sets interpretation $\mathcal{I} = (\delta(X), \phi_{\preceq}(X,Y))$, there exists a WMSO-formula $Code(X,x)$ such that, whenever for some tree t, $\mathcal{I}(t)$ is a finite powerset lattice, then $Code(X,x)$ evaluates on t to an injection mapping the atoms of $\mathcal{I}(t)$ to nodes of t.*

The proof of this result is long. Section 6 is dedicated to it.

We rarely use the theorem in this form. We rather use weakened versions of it, namely Corollary 11 and Corollary 12.

Corollary 11 *For every finite sets interpretation $\mathcal{I}$, there exists a WMSO-interpretation $\mathcal{I}_2$ such that whenever for some structure $\mathcal{S}$ and some tree t, $\mathcal{I}(t)$ is isomorphic to $\mathcal{P}^W(\mathcal{S})$ then $\mathcal{I}_2(t)$ is isomorphic to $\mathcal{S}$.*

Proof. If we remove all relations other than $\preceq$, the weak powerset generator is nothing but a finite powerset lattice. Hence we can obtain a formula $Code(X,x)$ by application of Theorem 10. It is then easy to transfer all relations defined on singletons to their image by $Code$.

Formally, we define the WMSO-interpretation $\mathcal{I}_2 = (\delta, \Phi_{R_1}, \dots, \Phi_{R_l})$ as follows:

- $\delta(x) = \exists X.Code(X,x)$,
- for each symbol R of arity r from the signature, $\Phi_R(x_1,\dots,x_r)$ is defined as

  $$\exists X_1,\dots,X_r.\Psi_R(X_1,\dots,X_r) \wedge \bigwedge_{i=1}^{r} Code(X_i,x_i)$$

  where each Ψ_R is the WMSO-formula in $\mathcal{I}$ defining the interpretation of the symbol R.

As $Code$ maps each element of $Atoms$ to a unique node, this interpretation indeed maps t to a structure isomorphic to $\mathcal{S}$. □

A weaker yet more readable formulation of the above corollary is provided in the following one.

Corollary 12 *If for a structure $\mathcal{S}$ and a tree t the class of structures that can be obtained by finite sets interpretations from $\mathcal{S}$ is contained in the class of structures that can be obtained by finite sets interpretations from t, then $\mathcal{S}$ is WMSO-interpretable in t.*

Proof. Assume that the hypothesis of the corollary holds. In particular, the structure $\mathcal{P}^W(\mathcal{S})$ is isomorphic to $\mathcal{I}(t)$ for some finite sets interpretation $\mathcal{I}$. By applying Corollary 11, the structure $\mathcal{S}$ is WMSO-interpretable in t. □

5 Applications

We present here several applications of the results above, ordered by level of complexity. The two first ones, showing that the free monoid is not obtainable by a finite sets interpretation of a tree (Section 5.1) and that a natural hierarchy of structures is strict (Section 5.2), are paradigmatic applications of Theorem 10. Section 5.3 establishes that the random graph is not finite sets interpretable in a tree, extending the known result for automatic structures. Finally, in Section 5.4, we study intrinsically definable relations in generators of the automatic structures.

Some of those results are known in the weaker context of automatic structures. We would like to underline here the fundamental methodological difference of our approach: in none of the applications below we perform a combinatorial analysis of finite sets interpretations. Instead, we systematically reduce the problem to a much easier one of WMSO-definability in trees.

5.1 The free monoid

We consider the free monoid as a structure $(\{a, b\}, \cdot, a, b)$ — the set of words over $\{a, b\}$ together with the ternary relation corresponding to the concatenation and the two words a and b identified by unary predicates — and want to answer the question whether this structure is isomorphic to a finite sets interpretation of a tree.

One should note that the FO theory of the free monoid is undecidable and hence we can directly conclude that it cannot be obtained by a finite sets interpretation from a tree with a decidable WMSO theory. However, this reasoning does not include trees with an undecidable WMSO theory.

The negative answer we give here to the above question is the simplest and in some sense the purest application of the results presented above and should be considered for this reason as a key example.

The following result was obtained in a discussion with Olivier Ly.

Proposition 13 *The free monoid over a two letter alphabet is not isomorphic to any finite sets interpretation of a tree.*

Proof. We first show how to obtain $\mathcal{P}^W(\mathbb{N}, +)$ from the free monoid by an FO-interpretation followed by a quotient. Then, assuming that our claim is false, we invoke the two results from the previous section and obtain a contradiction.

Let f be the function which to each word of the form $ba^{n_1}ba^{n_2}b \dots ba^{n_k}b$ over $\{a, b\}$ associates the set of naturals $\{n_1, n_2, \dots, n_k\}$. The domain of f is the set of elements satisfying $dom(x) = \exists y.x = byb$. The relation of inclusion is also first-order definable: $f(u) \subseteq f(v)$ iff $sub(u,v)$ holds with $sub(u,v) =$

$$\forall x \in a^*. \exists y, z.\ u = ybxbz \rightarrow \exists y', z'. v = y'bxbz' \ ,$$
$$\text{where } x \in a^* \ \text{ stands for } \ \forall\, y, z. x \neq ybz \ .$$

Let $\sim$ be the symmetric closure of sub, it is also first-order definable and is an equivalence relation. Finally the addition over singletons is definable. More precisely, $f(u) = \{i\}$, $f(v) = \{j\}$ and $f(w) = \{i+j\}$ iff $add(u,v,w)$ holds with $add(u,v,w) =$

$$\exists x \in a^*, y \in a^*. u \sim bxb \wedge v \sim byb \wedge w \sim bxyb.$$

Using those formulas, one can first-order interpret in the free monoid a structure which, when quotiented by $\sim$, is isomorphic to $\mathcal{P}^W(\mathbb{N},+)$.

Assume now that the free monoid can be finite sets interpreted in some tree t. Since structures obtainable by finite sets interpretations from t are closed under first-order interpretations (Proposition 3) and quotient (Theorem 8), this implies that $\mathcal{P}^W(\mathbb{N},+)$ is finite sets interpretable in t. By Corollary 11 we deduce that $(\mathbb{N},+)$ is WMSO-interpretable in t. This yields a contradiction since $(\mathbb{N},+)$ is not of bounded clique width. A direct argument to obtain a contradiction roughly looks as follows.

Assume that $(\mathbb{N},+)$ is WMSO-interpretable in t and let U denote the set of nodes of t that represent $\mathbb{N}$ in a corresponding interpretation. We first note that for each node of t at most one of its subtrees contains infinitely many elements from U. Otherwise, if the two subtrees of a node u contain both infinitely many elements from U, the successor relation on $\mathbb{N}$ (which is addition with one argument fixed to 1) would infinitely often jump between these two subtrees. If $\mathcal{A}$ is an automaton with n transitions accepting the successor relation, and if $x_0, \ldots, x_n \in U$ are in the left subtree of u, $y_0, \ldots, y_n \in U$ are in the right subtree of u, and all x_i, y_i are in the successor relation, then $\mathcal{A}$ also accepts a pair x_i, y_j with $i \neq j$ by a simple counting argument on the transitions used at u in accepting runs of $\mathcal{A}$.

By starting at the root and always proceeding to the unique subtree containing infinitely many elements from U we obtain an infinite branch B of t.

Now let $\mathcal{A}_+$ be an automaton with n transitions accepting the relation $+$ on t and let $x_0, \ldots, x_n \in U$. For each x_i there are infinitely many y_i, z_i such that the triple (x_i, y_i, z_i) is accepted by $\mathcal{A}_+$. Choose a node v on B such that none of the x_i is below v and for each i choose y_i, z_i as above that are below v. Counting the possible transitions that are used at v in accepting runs of $\mathcal{A}_+$ on the tuples (x_i, y_i, z_i) we obtain that $\mathcal{A}_+$ also accepts (x_i, y_j, z_j) for some $i \neq j$. This gives a contradiction. □

5.2 A hierarchy of structures of decidable first-order theory

Caucal [Cau02] introduces a hierarchy of graphs/structures of decidable MSO-theory. Level 0 consists of finite structures, and level $n+1$ is defined as the MSO-interpretations of the unraveling of graphs of level n. As both MSO-interpretation and unraveling are transformations preserving the decidability of the MSO-theory, each structure of this hierarchy has a decidable MSO-theory. In [CW03], this hierarchy is shown to be strict. If in these definitions the MSO-interpretations are replaced by WMSO-interpretations, we obtain the same hierarchy. Hence, the WMSO-theory is decidable for each structure of this hierarchy.

From this hierarchy, it is easy to construct a corresponding tree-automatic hierarchy. The tree-automatic structures of level n are the image of the structures of level n of the Caucal hierarchy by finite sets interpretations. Let us denote the nth level of this tree-automatic hierarchy by $\text{TA}\textsc{ut}_n$. Since the trees on the first level of the Caucal hierarchy are regular, we can deduce from Proposition 7 that $\text{TA}\textsc{ut}_1$ coincide with the class of tree-automatic structures.

Furthermore, from Corollary 4 and the above considerations we can conclude the following decidability result.

Remark 14 *For each n, every structure in $TA\textsc{ut}_n$ has a decidable FO theory.*

A simple application of our result is the strictness of this tree-automatic hierarchy.

Theorem 15 *For each $n \geq 0$ the class $TA\textsc{ut}_n$ of structures on the nth level of the automatic hierarchy is strictly contained in $TA\textsc{ut}_{n+1}$.*

Proof. We know that each level n of the Caucal hierarchy contains a tree generator G_n, i.e., each structure of level n is WMSO-interpretable in G_n [Cau02]. Let us suppose that the automatic hierarchy collapses at some level n, i.e., $\text{TA}\textsc{ut}_n = \text{TA}\textsc{ut}_{n+1}$. This would imply that the structure $\mathcal{P}^W(G_{n+1})$ can be obtained by a finite sets interpretation from G_n. Then, by Corollary 12, we obtain G_{n+1} as a WMSO-interpretation of G_n and hence G_{n+1} is in the nth level of the hierarchy. This contradicts the strictness of the Caucal hierarchy. $\square$

5.3 Random graph

The *random graph* is a non-oriented unlabeled countable graph with the following fundamental property: for any two disjoint finite set of vertices E and F, there exists a vertex v that is connected to all the elements of E and to none of the elements of F. For the existence and basic properties of such a graph see, e.g., [Hod93]. We do not give in this work a more precise definition of the random graph. Anyhow, a direct consequence of the fundamental property stated above is that the random graph satisfies the quantifier elimination property, and the decidability of its first-order theory follows.

Since the random graph has a decidable first-order theory and since finite sets interpretations define a large number of structures also having this property, it is interesting to consider the question whether the random graph can be obtained by a finite sets interpretation from a tree. A partial answer to this question has been studied: one knows that the random graph is not isomorphic to any word-automatic structure [KNRS04].

In this section, we show that there is no tree from which the random graph can be generated by a finite sets interpretation. This proof was obtained in a discussion with Vince Bárány.

Theorem 16 *The random graph is not finite sets interpretable in a tree.*

Proof. Heading for contradiction, let us assume that there exists a finite sets interpretation $\mathcal{I}_R = (\delta(X), \Psi(X,Y))$ and a binary tree t_R such that $\mathcal{I}_R(t_R)$ is (isomorphic to) the random graph. Then, the following fact would hold.

Claim: There exists a finite sets interpretation $\mathcal{I}'$ such that for any finite non-oriented graph G there exists a tree t_G such that $\mathcal{I}'(t_G)$ is isomorphic to $\mathcal{P}^W(G)$.

Before we prove this claim we demonstrate how to use it to show Theorem 16. Let us apply Corollary 11 on the interpretation $\mathcal{I}'$. We obtain a WMSO-interpretation $\mathcal{I}''$ with the property that for any non-oriented unlabeled graph G, the graph $\mathcal{I}''(t_G)$ is isomorphic to G for a suitably chosen tree t_G. As trees have bounded clique-width and WMSO-interpretations applied to a class of graphs of bounded clique-width yield also a class of graphs of bounded clique-width (see e.g. [Cou97]), we obtain a contradiction to the fact that there exists non-oriented graphs of arbitrary high clique-width

Proof of the claim: Let us show first how we can encode any finite set of elements of $\mathcal{I}_R(t_R)$ by a pair of finite sets of nodes of t_R in such a way that the membership relation is "definable".

Let E be a finite set of vertices of $\mathcal{I}_R(t_R)$. Each vertex of $\mathcal{I}_R(t_R)$ is a finite set of nodes of t_R and therefore it makes sense to define D_E as the union of all the elements in E. Furthermore, let us chose I_E to be an element of $\mathcal{I}_R(t_R)$ which is connected to all elements of E and to none of the elements of $\mathcal{P}(D_E)/E$ (such an element exists since $\mathcal{I}_R(t_R)$ is the random graph). From D_E and I_E one can easily reconstruct the set E. More precisely, let X be an element of $\mathcal{I}_R(t_R)$. Then X belongs to E if and only if t_R models $\delta(X) \wedge X \subseteq D_E \wedge \Psi(X, I_E)$.

Let now G be a non-oriented finite graph. Since $\mathcal{I}_R(t_R)$ is the random graph, the graph G appears as an induced subgraph of $\mathcal{I}_R(t_R)$ (cf. [Hod93]). Let V be the set of vertices of $\mathcal{I}_R(t_R)$ inducing this subgraph.

For each subset F of V, one can construct an element v_F of $\mathcal{I}_R(t_R)$ which is connected to all the vertices in F and to no vertex in V/F. Knowing V, this element v_F completely characterizes F. Let now V' be the set of all the v_F's for all subsets F of V.

Let t_G be the tree t_R extended with markings describing $D_V, I_V, D_{V'}$ and $I_{V'}$. Using the trick mentioned above, we can define the formula $X \in V$ (similarly $X \in V'$) to be $\delta(X) \wedge X \subseteq D_V \wedge \Psi(X, I_V)$. We now want to finite sets interpret $\mathcal{P}^W(G)$ in t_G. Obviously, we can identify the elements of $\mathcal{P}^W(G)$ with the elements of V'. Pursuing this idea, we define the interpretation $\mathcal{I}' = (\delta'(X), \Psi'(X,Y), \Phi_{\subseteq}(X,Y))$ in the following way. The universe is defined by $\delta'(X) = X \in V'$. The subset relation is defined by $\Phi_{\subseteq}(X,Y) = \forall Z \in V. \Psi(Z,X) \rightarrow \Psi(Z,Y)$. Finally the edge relation is defined by $\Psi'(X,Y) = \mathit{Singleton}(X) \wedge \mathit{Singleton}(Y) \wedge \exists X', Y' \in V. \Psi(X',X) \wedge \Psi(Y',Y) \wedge (\Psi(X',Y'))$ where $\mathit{Singleton}(Z)$ stands for $Z \in V' \wedge \exists! Y \in V. \Psi(Y,Z)$ and $\exists!$ abbreviates "there exists one and only one".

Using the properties linking V and V', it is not difficult to see that $\mathcal{I}'(t_G)$ is (up to isomorphism) $\mathcal{P}^W(G)$. Furthermore, $\mathcal{I}'$ does not depend on G. This finishes the proof of the claim and hence the proof of the theorem. □

5.4 Intrinsic definability

Our last application of Theorem 10 concerns "intrinsic definability" of relations. In analogy to intrinsic regularity for automatic structures ([KRS04], [Bár06]), intrinsic definability considers relations that are definable in every possible presentation of a structure by a finite sets interpretation from a fixed tree t. Hence, in contrast to the previous sections, we now explicitly consider the presentations of elements of a structure, i.e., we distinguish different codings of the same structure.

Definition 17 *Given a structure $\mathcal{T}$, a $\mathcal{T}$-*presentation *of a structure $\mathcal{S}$ of universe $\mathcal{U}$ is an injection f from $\mathcal{U}$ to the finite subsets of $\mathcal{T}$ such that the set $f(\mathcal{U})$ as well as the image by f of each relation R of $\mathcal{S}$ are WMSO-definable on $\mathcal{T}$. That is, there is a formula $\phi^f_{\mathcal{U}}(X)$ over $\mathcal{T}$ defining the image of $\mathcal{U}$ under f, and for each relation R of $\mathcal{S}$ there is a formula $\phi^f_R(X_1, \ldots, X_r)$ using the signature of $\mathcal{T}$ such that for all $u_1, \ldots, u_r \in \mathcal{U}$*

$$(u_1, \ldots, u_r) \in R \quad \textit{iff} \quad \mathcal{T} \models \phi^f_R(f(u_1), \ldots, f(u_r)),$$

where r denotes the arity of R.

Given such a $\mathcal{T}$-presentation f of $\mathcal{S}$, it might be possible to add relations to $\mathcal{S}$ such that f is still a $\mathcal{T}$-presentation of this extended structure. Such relations are called *definable in f*, i.e., R' is called definable in f if f is a $\mathcal{T}$-presentation of $\mathcal{S}$ extended by the relation R'.

Note that to a $\mathcal{T}$-presentation f of a structure $\mathcal{S}$ we can directly associate a finite sets interpretation $\mathcal{I}_f$ sending $\mathcal{T}$ to $\mathcal{S}$ up to isomorphism (the isomorphism being f). An additional relation R' is definable in f if we can add to $\mathcal{I}_f$ a formula defining R'.

If $\mathcal{S}$ is the weak powerset structure $\mathcal{P}^W(\mathcal{T})$ of $\mathcal{T}$, then there is a canonical $\mathcal{T}$-presentation given by the identity mapping. We refer to this $\mathcal{T}$-presentation as the *standard presentation* of $\mathcal{P}^W(\mathcal{T})$.

The following lemma states that the "intrinsically definable" relations of $\mathcal{P}^W(t)$ for a tree t are exactly those that are regular in the standard presentation of $\mathcal{P}^W(t)$.

Lemma 18 *Let t be a tree and R be a relation over $\mathcal{P}^W(t)$. If R is definable in the standard presentation of $\mathcal{P}^W(t)$, then R is definable in all t'-presentations f of $\mathcal{P}^W(t)$ for all trees t'.*

Proof. Let R be a relation of arity r that is definable in the standard presentation of $\mathcal{P}^W(t)$ and let $\Phi_R(X_1, \ldots, X_r)$ be the defining formula, i.e., Φ_R is a formula over the signature of t such that $t \models \Phi_R(U_1, \ldots, U_r)$ iff $(U_1, \ldots, U_r) \in R$ for all $U_1, \ldots, U_r \subseteq dom(t)$. According to Proposition 6 we can construct an FO-interpretation $\mathcal{I}_1$ such that $\mathcal{I}_1(\mathcal{P}^W(t))$ is the structure $\mathcal{P}^W(t)$ augmented with the relation R.

Let now f be a t'-presentation of $\mathcal{P}^W(t)$ and let $\mathcal{I}_f$ be the finite sets interpretation sending t' to $\mathcal{P}^W(t)$. Then $\mathcal{I}_1 \circ \mathcal{I}_f$ sends t' to $\mathcal{I}_1(\mathcal{P}^W(t))$, witnessing the definability of R in the t'-presentation f of $\mathcal{P}^W(t)$. □

The symmetric of Lemma 18, where definable is replaced by not definable, is not true in general. This is already the case for instance for $t = t' = \Delta_2$, i.e., there is a relation R which is not definable in the standard presentation of $\mathcal{P}^W(\Delta_2)$ but is definable in some Δ_2-presentation of $\mathcal{P}^W(\Delta_2)$.

However, in the particular case of $t = \Delta_1$ and $t' = \Delta_2$ such a converse of Lemma 18 does hold as expressed in the following theorem. And here Theorem 10 comes into the play.

Theorem 19 *If R is definable in some Δ_2-presentation f of $\mathcal{P}^W(\Delta_1)$, it is definable in every Δ_2-presentation of $\mathcal{P}^W(\Delta_1)$.*

As we can expect, by application of Theorem 10, we will obtain a WMSO-interpretation sending Δ_2 to Δ_1. The two following lemmas study this kind of interpretations and how they preserve definability.

Lemma 20 *Let $\mathcal{I}_2$ be a WMSO-interpretation sending Δ_2 to Δ_1 and f_2 be the injection from $\mathbb{N}$ to $\{0,1\}^*$ witnessing the isomorphism. Then there exists naturals $m, n > 0$ and words $u, v, w_0, \ldots, w_{n-1} \in \{0,1\}^*$ such that for any naturals $k \geq 0$ and $p \in \{0, \ldots, n-1\}$,*

$$f_2(m + kn + p) = uv^{nk}w_p \ .$$

Proof. The general idea behind the proof is that the elements from the image of f_2 cannot be spread arbitrarily in Δ_2 because the successor relation from Δ_1 has to be definable in WMSO and hence must be recognizable by an automaton.

We denote by $U \subseteq \{0,1\}^*$ the image of f_2 and by V the closure of U by prefix. The set V defines a subtree of Δ_2. We now augment V by markings containing information on the successor relation in such a way that these markings are definable in WMSO. Hence, the marked tree is regular, i.e., it has only finitely many non-isomorphic subtrees (see [Tho97] for more information on regular trees). From this regular tree we can define the words $u, v, w_0, \ldots, w_{n-1}$.

Let $\Phi_{succ}(x, y)$ be the formula of $\mathcal{I}$ that defines the successor relation *succ* of Δ_1, and let $\mathcal{A}_{succ}$ be the equivalent tree automaton.

As *succ* is deterministic, one can easily show that V contains only one infinite branch B. Otherwise, the relation *succ* has to jump infinitely often between two infinite branches leading to a contradiction as $\mathcal{A}_{succ}$ would also accept pairs of nodes that are not in *succ*. Furthermore, this branch B is WMSO-definable.

Consider two nodes $x, y \in U$ such that $\Phi_{succ}(x, y)$ is satisfied, i.e., $f_2^{-1}(y)$ is the successor of $f_2^{-1}(x)$. We can describe how to get from x to y by a pair of words (z_x, z'_x) over $\{0,1\}$ meaning that $x = x'z_x$ and $y = x'z'_x$ for the greatest common ancestor x' of x and y. Again, using the determinism of *succ* one can show that the length of these words z_x, z'_x is bounded by some constant derived from the size of $\mathcal{A}_{succ}$. Hence, we can mark the vertices from U by this information (using sets $X_{z,z'}$ with $x \in X_{z,z'}$ iff $(z_x, z'_x) = (z, z')$). Obviously, this marking is WMSO-definable.

The last information that we attach to V is for each node $x \in B$ the word $b_x \in \{0,1\}^*$ such that the node xb_x is the smallest node in U (smallest referring to the position in Δ_1) such that all nodes bigger than xb_x are below x, i.e., for all $y \in U$, if $f_2^{-1}(xb_x) < f_2^{-1}(y)$, then $x \sqsubseteq y$. The length of these b_x is bounded because the relation that associates to each x the node b_x is WMSO-definable and deterministic. Hence, the marking of the nodes in B by using sets X_b with $x \in X_b$ iff $b_x = b$ is WMSO-definable.

The resulting tree t, consisting of the nodes in V with the markings described above is WMSO-definable and hence regular. Let $u, v \in \{0,1\}^*$ such that $u, uv \in$

B and the subtrees of t rooted at u and uv are isomorphic. Let $m = f_2^{-1}(ub_u)$, $m' = f_2^{-1}(uvb_{uv})$, and define $n = m' - m$. For $p \in \{0, \dots, n-1\}$ let $w_p \in \{0,1\}^*$ be such that $f_2(m+p) = uw_p$. By the choice of m, such a w_p always exists. In particular, $w_0 = b_u$.

By the choice of v, we know that $f_2(m + kn) = uv^{kn}w_0$ for all $k \geq 0$. Furthermore, as t is marked by the information on how to get from one node in U to its successor, we know that the ways to get from $f_2(m + kn + p)$ to $f_2(m + kn + p + 1)$ are the same for all k. Hence, $f_2(m + kn + p) = uv^{kn}w_p$. □

Lemma 21 *Let $\mathcal{I}_2$ be a WMSO-interpretation sending Δ_2 to Δ_1 and f_2 be the injection from $\mathbb{N}$ to $\{0,1\}^*$ witnessing the isomorphism. If R is a relation over finite subsets of $f_2(\mathbb{N})$ WMSO-definable in Δ_2, then its inverse image under f_2 is WMSO-definable in Δ_1.*

Proof. The formula $\Psi_R(X_1, \dots, X_r)$ defining R can be represented by a tree automaton $\mathcal{A}_R$ with state set Q_R. Using Lemma 20, this automaton can be simulated by an automaton $\mathcal{A}'_R$ on Δ_1 as we show in the following. On Δ_1 automata and WMSO have the same expressive power and hence the construction of $\mathcal{A}'_R$ suffices to prove the lemma. Let $u, v, w_0, \dots, w_{n-1}$ be as in Lemma 20. The states of $\mathcal{A}'_R$ correspond to partial runs on a finite subtree of Δ_2 that

- is rooted at ϵ and induced by the elements $f_2(0), \dots, f_2(m-1)$ for the first segment of Δ_1,
- rooted at uv^{kn} and induced by the words $v, w_0, \dots, w_{n-1}$ for the following segments of Δ_1.

For this purpose, let U_0 be the set of all nodes that are prefix of u or of some $f_2(i)$ for $0 \leq i < m$, and let U_1 be the set of all nodes that are prefix of v or one of $w_0, \dots, w_{n-1}$.

The automaton $\mathcal{A}'_R$ reads a word $\alpha \in (\{0,1\}^r)^\omega$ and has to decide if this labeling transferred by f_2 to Δ_2 corresponds to a tuple of sets in R.

The automaton starts by guessing a pair (ρ_0, λ_0) with mappings $\rho_0 : U_0 \to Q_R$ and $\lambda_0 : \{f_2(0), \dots, f_2(m-1)\} \to \{0,1\}^r$ such that ρ_0 corresponds to a partial run of $\mathcal{A}_R$ on U_0 with labels corresponding to λ_0. In the next steps, $\mathcal{A}'_R$ verifies if the guessed labeling is correct, i.e., if $\alpha(i) = \lambda_0(f_2(i))$. When reaching position m, $\mathcal{A}'_R$ guesses a new pair (ρ_1, λ_1), now with mappings $\rho_1 : U_1 \to Q_R$ and $\lambda_1 : \{w_0, \dots, w_{n-1}\} \to \{0,1\}^r$ such that ρ_1 is a possible continuation of ρ_0 on the subtree rooted at u that is induced by v and $w_0, \dots, w_{n-1}$ and labeled according to λ_1. The guessed labeling is again verified on the segment $m, \dots, m+n-1$ and then a pair (ρ_2, λ_2) of the same type as (ρ_1, λ_1) is guessed, and so on. The automaton accepts if the concatenation of the guessed partial runs on the path uv^ω satisfies the acceptance condition of $\mathcal{A}_R$. For this to work, the guessed partial runs have to be such that they can be continued to accepting runs on the "blank parts" of Δ_2, i.e., those infinite subtrees that do not contain a node from the image of f_2. □

Using this Lemma we can prove Theorem 19.

Proof (Theorem 19). Assuming a Δ_2-presentation f of $\mathcal{P}^W(\Delta_1)$ and $\mathcal{I}$ the corresponding interpretation, one obtains by Corollary 11 a WMSO-interpretation

$\mathcal{I}_2$ sending Δ_2 to Δ_1. Let $f_2 : \mathbb{N} \to \{0,1\}^*$ be the injection witnessing this isomorphism and let $\tilde{f}_2$ be its extension to sets.

We now have two ways to obtain isomorphic copies of $\mathcal{P}^W(\Delta_1)$ from Δ_2: by applying $\mathcal{I}$ and by applying $\mathcal{I}_2$ followed by $\mathcal{P}^W$. These two ways yield isomorphic structures and hence there is an isomorphism h sending $\mathcal{I}(\Delta_2)$ to $\mathcal{P}^W(\mathcal{I}_2(\Delta_2))$. We obtain the following picture, where dashed arrows represent interpretations, while normal arrows are for isomorphisms:

$$\begin{tikzcd}
\mathcal{I}(\Delta_2) \arrow[r, "h"'] & \mathcal{P}^W(\mathcal{I}_2(\Delta_2)) & \mathcal{P}^W(\Delta_1) \arrow[l, "\tilde{f}_2"] \arrow[ll, bend right, "f"'] \\
\Delta_2 \arrow[u, dashed, "\mathcal{I}"'] \arrow[r, dashed, "\mathcal{I}_2"] & \mathcal{I}_2(\Delta_2) \arrow[u, dashed, "\mathcal{P}^W"'] & \Delta_1 \arrow[l, "f_2"'] \arrow[u, dashed, "\mathcal{P}^W"']
\end{tikzcd}$$

We now show that we can define the isomorphism h on Δ_2 by a WMSO-formula. To understand this consider a finite set U of naturals, i.e., an element of $\mathcal{P}^W(\Delta_1)$. This set U corresponds to two sets X and Y of nodes of Δ_2, namely $X = f(U)$ and $Y = \tilde{f}_2(U)$. The isomorphism h relates these two sets, i.e., $h(X) = Y$. This relation can be defined in WMSO using the formula *Code* that we obtain from Theorem 10 and that is used to construct $\mathcal{I}_2$ in Corollary 11. The formula $\mathit{Code}(X, x)$ itself already defines h restricted to atoms of $\mathcal{P}^W(\Delta_1)$. It is easy to extend this to sets:

$$\phi_h(X, Y) = \forall y(y \in Y \leftrightarrow \exists Z(\Phi_{\subseteq}(Z, X) \wedge \mathit{Code}(Z, y))).$$

Let now R be a relation over $\mathcal{P}^W(\Delta_1)$ which is definable in f. This means that $f(R)$ is WMSO-definable in Δ_2. Since h is WMSO-definable, it follows that $h(f(R))$ is also WMSO-definable in Δ_2. Finally, by Lemma 21 we obtain that $\tilde{f}_2^{-1}(h(f(R)))$ is WMSO-definable in Δ_1. Since $\tilde{f}_2^{-1} \circ h \circ f$ is obtained as a composition of isomorphisms, it is an automorphism of $\mathcal{P}^W(\Delta_1)$. Remark now that the identity is the only automorphism of $\mathcal{P}^W(\Delta_1)$ (a property inherited from Δ_1). It follows that $\tilde{f}_2^{-1}(h(f(R)))$ equals R. And consequently R is definable in the standard presentation. By Lemma 18 it is definable in every presentation. $\square$

A similar result has been shown in [Bár06] for all Δ_1-presentations of $(\Delta_2, \sqsubseteq, \mathrm{el})$, i.e., the infinite binary tree with binary relations for prefix and equal level. Our result is a bit stronger because we consider all Δ_2-presentations, i.e., all tree-automatic presentations of $\mathcal{P}^W(\Delta_1)$ and not only the word-automatic presentations.

6 Proof of the main result

The proof of Theorem 10 is rather complex and split into several parts. In Subsection 6.1, we introduce the key notions used afterwards while we make the scheme of the proof more precise. This will also be the occasion for explaining the content of Subsections 6.2, 6.3, 6.4, 6.5. In Subsection 6.6, things are put together and the proof of Theorem 10 is finally given.

6.1 First definitions and presentation of the proof

We assume from now that a finite sets interpretation $\mathcal{I} = (\delta(X), \phi_{\preceq}(X,Y))$ is fixed. Along the whole proof we use a tree t together with a set E and the isomorphism f that are assumed to satisfy the equality

$$f(\mathcal{P}^F(E)) = \mathcal{I}(t) \ .$$

The reader must keep in mind that none of the constructions we perform makes use of t, E, or f. Hence, the result will hold for any such tree, set, and isomorphism. This lightens the presentation of the proof by avoiding to systematically quantify over those objects. As mentioned above, for simplicity reasons, we assume that all the nodes of a tree t have either 2 successors or no successors, i.e., for all nodes u we have $u0 \in dom(t)$ iff $u1 \in dom(t)$. The general case is not different.

We consider *Atoms*, the set of finite subsets of t representing atoms of the powerset lattice, i.e.,

$$Atoms = \{f(\{u\}) \ : \ u \in E\} \ .$$

The set *Atoms* can be defined as the set of finite subsets of t which are minimal — for the $\phi_{\preceq}$ formula seen as an ordering — and distinct from the minimal element itself (which is $f(\emptyset)$). This description can be done in weak monadic second-order logic. Hence *Atoms* is regular in t and there exists an automaton

$$\mathcal{A}_{Atoms} = (Q_{Atoms}, q^{\text{in}}_{Atoms}, \delta_{Atoms}, \Omega_{Atoms})$$

accepting the language *Atoms*. We also consider the binary relation *Mem* over $\mathcal{I}(t)$ defined as the image under f of the $\in$ relation in $\mathcal{P}^F(E)$, i.e.,

$$Mem = \{(f(\{u\}), f(V)) \ : \ u \in V \subseteq E\} \ .$$

This *Mem* relation is also definable in weak monadic second-order logic, and consequently is regular. We fix

$$\mathcal{A}_{Mem} = (Q_{Mem}, q^{\text{in}}_{Mem}, \delta_{Mem}, \Omega_{Mem})$$

to be an automaton recognizing the relation *Mem*.

Recall that the theorem we want to prove claims the existence of a formula $Code(X,x)$ such that the corresponding relation is an injection from *Atoms* into $dom(t)$.

Our goal in the construction of *Code* is to uniquely attach to each X in *Atoms* an element in $dom(t)$ in a WMSO-definable way. As a first approximation, in Sections 6.2, 6.3 and 6.4, we define a mapping *Index* which assigns to each X in *Atoms* a node in $dom(t)$. Though the *Index* mapping is not in general an injection from *Atoms* into $dom(t)$, it does not either concentrate a lot of indices in the same area of the tree t. Formally, if you set $D(x)$ for $x \in t$ to be the cardinal of $Index^{-1}(x)$, then by Lemmas 27 and 33, D happens to be a sparse distribution (see Definition 22 below). Subsection 6.5 is dedicated to the study of sparse distributions. The central lemma of this part, Lemma 38, establishes that, given elements concentrated according to a sparse distribution, we can uniformly redistribute them in $dom(t)$ in a unique WMSO-definable way. Applied to our case, this means that the *Index* mapping can be transformed into an injection by use of WMSO-formulas. And this last step is used in Section 6.6 for terminating the proof of Theorem 10.

The key definition connecting the two main parts of the proof (definition of the *Index* mapping and turning it into an injection) is the notion of sparsity. This definition requires the notion of zone. A *zone* Z in t is a connected — where t is seen as an non-oriented graph — subset of $dom(t)$. That is, whenever $x \sqsubseteq y \sqsubseteq z$ for x, z in Z, then also $y \in Z$. A zone Z is completely characterized by its least element x, and by the minimal elements $x_1, \ldots, x_n$ that are below x and not in Z. The elements $\{x, x_1, \ldots, x_n\}$ are called the *frontier* of the zone. Given nodes $x, x_1, \ldots, x_n$ of t such that the x_i are pairwise incomparable and $x \sqsubseteq x_i$ for all i, we define $N_t^{x,x_1,\ldots,x_n}$ to be the set of nodes y such that $x \sqsubseteq y$ and $x_i \not\sqsubseteq y$ for all $i \in [n]$ where $[n]$ denotes the set $\{1, \ldots n\}$. By construction $N_t^{x,x_1,\ldots,x_n}$ is the sole zone which has frontier $\{x, x_1, \ldots, x_n\}$.

Definition 22 *A* distribution D *is a mapping from* $dom(t)$ *to* $\mathbb{N}$. *For* Z *a finite zone,* $D(Z)$ *stands for* $\sum_{x \in Z} D(x)$. *A distribution* D *is* K-sparse *for some* $K \in \mathbb{N}$ *if for every finite zone* Z *of frontier* F, $D(Z) \leq |Z| + K|F|$. *A distribution is* strongly K-sparse *if for any finite zone* Z *of frontier* F, $D(Z) \leq K|F|$.

Sparsity tells us that no finite zone contains more indices than its size plus a factor linearly depending on the size of the frontier.

6.2 Important nodes

In order to construct the mapping *Index*, given an element X of *Atoms*, we first define the set $I(X) \subseteq dom(t)$ of its important nodes via combinatorial constraints. Essentially we try to locate the places where "important coding decisions" are made by the automaton $\mathcal{A}_{Atoms}$ when reading X. In the present section, we provide the key combinatorial lemmas concerning important nodes.

Then, depending on the shape of the set $I(X)$ two cases are separated and two distinct definitions of *Index* are given. The first kind of indices are called standard indices — noted $SIndex(X)$ — and are the subject of Section 6.3. The others are called branch indices — noted $BIndex(X)$ — and are the subject of Section 6.4.

Let us first introduce a convenient notation for studying the behavior of the automata $\mathcal{A}_{Atoms}$ and $\mathcal{A}_{Mem}$ over zones: For a zone $Z = N_t^{x,x_1,\ldots,x_n}$ and states $q, q_1, \ldots, q_n \in Q_{Atoms}$, we denote by $Atoms(q, Z, q_1, \ldots, q_n)$ the set of all $X \subseteq Z$ such that there exists $X' \subseteq dom(t)$ with

- $X = X' \cap Z$, and
- X' is accepted by $\mathcal{A}_{Atoms}$ with a run ρ such that $\rho(x) = q$ and $\rho(x_i) = q_i$ for all $i \in [n]$.

Similarly, for $q, q_1, \ldots, q_n \in Q_{Mem}$ we denote by $Mem(q, Z, q_1, \ldots, q_n)$ the set of all pairs (X, Y) with $X, Y \subseteq Z$ such that there are $X', Y' \subseteq dom(t)$ with

- $X = X' \cap Z$, $Y = Y' \cap Z$, and
- (X', Y') is accepted by $\mathcal{A}_{Mem}$ with a run ρ such that $\rho(x) = q$ and $\rho(x_i) = q_i$ for all $i \in [n]$.

The definition of important nodes is then the following.

Definition 23 *Let* $K_{\mathrm{im}} = (2|Q_{Mem}|+1)|Q_{Atoms}|$. *Given* $X \in Atoms$, *a node* $x \in dom(t)$ *is called* important *for* X *if*

$$|\{Y \subseteq N_t^x \ : \ (X - N_t^x) \cup Y \in Atoms\}| > K_{\mathrm{im}} \ .$$

We denote by $I(X)$ *the set of important nodes for* X.

Hence a node x is important for X if there are many — i.e., more than K_{im} — ways to modify X below x while remaining in $Atoms$. Intuitively, without knowing how X looks like below x, we cannot say much about which atom is coded because there are too many possibilities left. Remark that the set $I(X)$ is by definition prefix closed. The fundamental property that we show in Lemma 25 is that for an important node x of X, the part of X that is not below x comes from a set of small size. To prove this lemma we need its combinatorial core stated in the following lemma.

Lemma 24 *Let* $K_{\mathrm{c}} = 2|Q_{Mem}| + 1$. *For any two disjoint zones* Z *and* Z' *of respective frontiers* $F = \{x, x_1, \ldots, x_n\}$ *and* $F' = \{x', x'_1, \ldots, x'_m\}$, *and all accepting runs* ρ *of* $\mathcal{A}_{Atoms}$

$$\begin{aligned} \textit{either} \quad & |Atoms(\rho(x), Z, \rho(x_1), \ldots, \rho(x_n))| < K_{\mathrm{c}} \ , \\ \textit{or} \quad & |Atoms(\rho(x'), Z', \rho(x'_1), \ldots, \rho(x'_m))| < K_{\mathrm{c}} \ . \end{aligned}$$

Proof. It is sufficient for us to prove the result for two complementary zones. This comes from the fact that increasing a zone also increases the number of possible projections w.r.t. a fixed run, i.e., $|Atoms(\rho(x), Z, \rho(x_1), \ldots, \rho(x_n))| \leq |Atoms(\rho(y), Z'', \rho(y_1), \ldots, \rho(y_\ell))|$ for a zone Z'' of frontier $\{y, y_1, \ldots, y_\ell\}$ with $Z \subseteq Z''$. Hence, we will assume Z to be $N_t^{\epsilon,x}$ and Z' to be N_t^x for some node x.

Let us assume that for some $K \geq K_{\mathrm{c}}$ we have distinct sets $X_1, \ldots, X_K$ in $Atoms(\rho(\epsilon), Z, \rho(x))$ and distinct sets $X'_1, \ldots, X'_K$ in $Atoms(\rho(x), Z')$. Then, for every $i, j \in [K]$, let $Y_{i,j}$ be $X_i \cup X'_j$. As the union of Z and Z' gives the whole domain of t, we have $Y_{i,j} \in Atoms$ for all $i, j \in [K]$.

Let us now consider the set $Comb$ of possible combinations of the $Y_{i,j}$, combination in the sense of the relation Mem. More precisely, $A \subseteq dom(t)$ is in $Comb$ if whenever $(Y, A) \in Mem$ holds for some atom Y, then $Y = Y_{i,j}$ for some i, j. The cardinality of $Comb$ is 2^{K^2}. We now show by a combinatorial argument that $\mathcal{A}_{Mem}$ cannot distinguish all the elements from $Comb$ because the amount of information that can be passed between the two zones Z and Z' is limited by the number of states in Q_{Mem}.

For this purpose, we define for each $A \in Comb$, $f_A : [K] \times Q_{Mem} \to \{0, 1\}$ and $g_A : Q_{Mem} \times [K] \to \{0, 1\}$ by

$$f_A(i, q) = \begin{cases} 1 & \text{if } (X_i, A \cap Z) \in Mem(q^{\mathrm{in}}_{Mem}, Z, q), \\ 0 & \text{else,} \end{cases}$$

$$g_A(q, j) = \begin{cases} 1 & \text{if } (X'_j, A \cap Z') \in Mem(q, Z'), \\ 0 & \text{else.} \end{cases}$$

It is obvious that if two sets $A, B \in \mathit{Comb}$ are such that $f_A = f_B$ and $g_A = g_B$, then $(Y_{i,j}, A) \in \mathit{Mem}$ iff $(Y_{i,j}, B) \in \mathit{Mem}$ for all $i, j \in [K]$. This means, by definition of Comb, that $A = B$.

However, there are only $2^{2|Q_{\mathit{Mem}}|K}$ different possible values for the pair f_A, g_A. Hence we obtain $|\mathit{Comb}| \leq 2^{2|Q_{\mathit{Mem}}|K}$. This contradicts $|\mathit{Comb}| = 2^{K^2}$. $\square$

The following lemma shows that the possibilities to code an atom 'above' an important node are bounded.

Lemma 25 *For each node x we have $|\{X \cap N_t^{\epsilon,x} \ : \ X \in \mathit{Atoms} \text{ and } x \in I(X)\}| < K_{\mathrm{im}}$.*

Proof. We are aiming at a contradiction to Lemma 24 for $Z = N_t^{\epsilon,x}$ and $Z' = N_t^x$.

For each X with $x \in I(X)$ there are more than K_{im} many $Y \subseteq N_t^x$ such that $X_{Y,x} := (X - N_t^x) \cup Y$ is in Atoms. Since $K_{\mathrm{im}} = |Q_{\mathit{Atoms}}| \cdot K_{\mathrm{c}}$ (with K_{c} from Lemma 24), we can choose a state $q_{X,x} \in Q_{\mathit{Atoms}}$ such that more than K_{c} of these $X_{Y,x}$ are accepted by $\mathcal{A}_{\mathit{Atoms}}$ with a run that labels x with $q_{X,x}$. This means that $|\mathit{Atoms}(q_{X,x}, N_t^x)| \geq K_{\mathrm{c}}$.

Now, assume that there are K_{im} different $X \in \mathit{Atoms}$ with $x \in I(X)$ that differ on $N_t^{\epsilon,x}$. Then there are at least K_{c} such sets $X_1, \ldots, X_{K_{\mathrm{c}}}$ with $q_{X_1,x} = \cdots = q_{X_{K_{\mathrm{c}}},x} =: q$. In particular, we obtain $|\mathit{Atoms}(q^{\mathrm{in}}_{\mathit{Atoms}}, N_t^x, q)| \geq K_{\mathrm{c}}$. Together with $|\mathit{Atoms}(q, N_t^x)| \geq K_{\mathrm{c}}$ from above we obtain the desired contradiction. $\square$

6.3 Standard indices

We now address the problem of computing $\mathit{Index}(X)$ for some atom X under the assumption that $I(X)$ is *not an infinite branch* (the case when $I(X)$ is an infinite branch is treated in Section 6.4). Since we call this case the standard case, we will denote the index defined for such atoms X by $\mathit{SIndex}(X)$. The simplest case is that $I(X)$ is totally ordered by $\sqsubseteq$, i.e., $I(X)$ is a finite path starting from the root. We simply define $\mathit{SIndex}(X)$ to be the last node on this path. The other case corresponds to $I(X)$ not being a finite path nor an infinite branch. This corresponds to $I(X)$ not being totally ordered by $\sqsubseteq$. In this situation, we define $\mathit{SIndex}(X)$ to be the first node at which $I(X)$ splits into two paths. Those two cases are unified in the following definition.

Definition 26 *For $X \in \mathit{Atoms}$ such that $I(X)$ is not an infinite branch, the* index *of X, written $\mathit{SIndex}(X)$, is the maximal element in $I(X)$ which is comparable to every element in $I(X)$.*

As already mentioned, the intention of this definition is that $\mathit{SIndex}(X)$ roughly locates in the tree where the main information concerning the atom coded by X lies. This location is far from being precise, and many elements of *Atoms* may have the same index. However, we will see that it is possible to obtain a good understanding of the repartition of the standard indices. The following lemma gives precise bounds on the quantity of indices that may occur in a zone, i.e., it states that the distribution assigning to each node x the number of X such that $\mathit{SIndex}(X) = x$ is sparse.

Lemma 27 (sparsity) *There is a constant K_{s} such that $|\mathit{SIndex}^{-1}(Z)| \leq |Z| + K_{\mathrm{s}}|F|$ for every finite zone Z of frontier F.*

Proof. Denote the elements of the frontier of Z by x and $x_1, \ldots, x_n$, i.e., $Z = N_t^{x,x_1,\ldots,x_n}$. The proof of the lemma consists of two steps. We first show that for atoms X such that $\mathit{SIndex}(X)$ is inside Z, the amount of information located outside Z is bounded. More precisely, we first show for $M := K_{\mathrm{im}} \cdot |Q_{\mathit{Atoms}}|$

(a) $|\{X \cap N_t^{\epsilon,x} \ : \ X \in \mathit{Atoms} \text{ and } \mathit{SIndex}(X) \in Z\}| < M$ and
(b) $|\{X \cap N_t^{x_i} \ : \ X \in \mathit{Atoms} \text{ and } \mathit{SIndex}(X) \in Z\}| < M$ for all $i \in [n]$.

For (a) note that from $\mathit{SIndex}(X) \in Z$, the definition of SIndex, and the prefix closure of $I(X)$ we obtain that $x \in I(X)$. Therefore,

$$\begin{aligned}&\{X \cap N_t^{\epsilon,x} \ : \ X \in \mathit{Atoms} \text{ and } \mathit{SIndex}(X) \in Z\}\\ &\quad\subseteq \{X \cap N_t^{\epsilon,x} \ : \ X \in \mathit{Atoms} \text{ and } x \in I(X)\}\end{aligned}$$

and (a) follows from Lemma 25.

For (b) we show that for each $X \in \mathit{SIndex}^{-1}(Z)$ and each x_i there is a state $q \in Q_{\mathit{Atoms}}$ such that $X \cap N_t^{x_i} \in \mathit{Atoms}(q, N_t^{x_i})$ and $|\mathit{Atoms}(q, N_t^{x_i})| < K_{\mathrm{im}}$. From this, (b) follows because each $X \cap N_t^{x_i}$ comes from one of at most $|Q_{\mathit{Atoms}}|$ many sets of size less than K_{im}. We distinguish two cases.

If $x_i \notin I(X)$, then we take q to be the state at x_i in an accepting run of $\mathcal{A}_{\mathit{Atoms}}$ on X. From the definition of $I(X)$ we immediately obtain the desired property.

Else, if $x_i \in I(X)$, by definition of standard indices there must some $y \in I(x)$ incomparable to x_i, the index of X being the deepest common ancestor of x_i and y. From the definition of important nodes for X and from $K_{\mathrm{im}} = |Q_{\mathit{Atoms}}| \cdot K_{\mathrm{c}}$ we obtain that there exists a set $Y \subseteq N_t^y$ such that $(X - N_t^y) \cup Y$ is in Atoms and is accepted with a run of $\mathcal{A}_{\mathit{Atoms}}$ that labels y by a state q' such that $|\mathit{Atoms}(q', N_t^y)| \geq K_{\mathrm{c}}$. Let q be the state assumed at node x_i by this run. From Lemma 24 applied to the zones $N_t^{x_i}$, N_t^y, and to the aforementioned run, we can conclude that $|\mathit{Atoms}(q, N_t^{x_i})| < K_{\mathrm{c}}$. The desired property follows from $K_{\mathrm{c}} \leq K_{\mathrm{im}}$. This finishes the proof of (b).

Let us now turn ourselves to the conclusion of the lemma. We denote the elements from $\{X \cap N_t^{\epsilon,x} \ : \ \mathit{SIndex}(X) \in Z\}$ by $X^1, \ldots, X^M$ and the elements from $\{X \cap N_t^{x_i} \ : \ \mathit{SIndex}(X) \in Z\}$ by $X_i^1, \ldots, X_i^M$ (the same element can be represented more than once, what is important is that all elements are represented).

Now, consider the set Comb of all combinations of atoms from $\mathit{SIndex}^{-1}(Z)$ (in the same sense as in the proof of Lemma 24). A combination $A \in \mathit{Comb}$ is entirely characterized by the following objects

- the set $A \cap Z$,
- the mapping $f_{A,x} : [M] \times Q_{\mathit{Mem}} \to \{0, 1\}$ with

$$f_{A,x}(j, q) = \begin{cases} 1 & \text{if} \quad (X^j, A \cap N_t^{\epsilon,x}) \in \mathit{Mem}(q^{\mathrm{in}}_{\mathit{Mem}}, N_t^{\epsilon,x}, q), \\ 0 & \text{else,} \end{cases}$$

- and the mapping $f_{A,x_i} : Q_{\mathit{Mem}} \times [M] \to \{0, 1\}$ with

$$f_{A,x_i}(q, j) = \begin{cases} 1 & \text{if} \quad (X_i^j, A \cap N_t^{x_i}) \in \mathit{Mem}(q, N_t^{x_i}), \\ 0 & \text{else.} \end{cases}$$

Thus, $|Comb| \leq 2^{|Z|} \cdot 2^{M|Q_{Mem}|} \cdot \prod_{i=1}^{n} 2^{|Q_{Mem}|M}$. For $K_{\mathrm{s}} = |Q_{Mem}|M$ we obtain

$$2^{|SIndex^{-1}(Z)|} = |Comb| \leq 2^{|Z|+|F|K_{\mathrm{s}}}$$

and hence $|SIndex^{-1}(Z)| \leq |Z| + K_{\mathrm{s}}|F|$. □

6.4 Treatment of infinite branches

It is possible that for some $X \in Atoms$ the set $I(X)$ of important nodes is an infinite branch. For these X we also develop a notion of index, called $BIndex(X)$, and show that the distribution obtained in this way is strongly sparse. Since the sum of a K-sparse distribution and of a strongly K'-sparse distribution is a $K+K'$-sparse distribution, we can add the indices corresponding to infinite branches to the other indices without affecting the sparsity of the induced distribution.

In this section, we call *important branches* the infinite branches equal to $I(X)$ for some atom X. We start with the helpful observation that the number of elements of *Atoms* corresponding to the same important branch is bounded.

Lemma 28 *For every important branch B, $|I^{-1}(B)| < K_{\mathrm{im}}$.*

Proof. If there are K_{im} different sets in $I^{-1}(B)$, then we can pick a node x on B such that all these sets differ on the zone $N_t^{\epsilon,x}$. Since x is important for all X in $I^{-1}(B)$, we obtain a contradiction to Lemma 25. □

Our goal is to associate to every important branch B a node $VInd(B)$ on B such that

(1) at most K_{im} branches are mapped to the same node by $VInd$, and
(2) if some $VInd(B')$ is above $VInd(B)$, then $VInd(B)$ is not in B'.

Those two properties are established in Lemma 32. Then Lemma 33 uses those sole properties for concluding that $VInd \circ I$ has a strongly sparse distribution.

Our main tool for constructing $VInd$ is to produce a well-founded order for branches. For this, we define $RInd(B)$ for every important branch B by

$$RInd(B) = \min\{x \in B : \exists X \subseteq N_t^{\epsilon,x},\ I(X) = B\}.$$

Since we consider finite sets interpretations, $RInd(B)$ is always defined. Lemma 25 applied to the node $RInd(B)$ directly leads to the following lemma.

Lemma 29 *For all nodes x, $|RInd^{-1}(x)| < K_{\mathrm{im}}$.*

The well-foundedness argument announced above is then the following.

Lemma 30 *For every important branch B, there are finitely many important branches B' such that $RInd(B') \sqsubseteq RInd(B)$.*

Proof. One has that $RInd(B') \sqsubseteq RInd(B)$ iff B' belongs to $\cup_{x \sqsubseteq RInd(B)} RInd^{-1}(x)$. Lemma 29 shows that this set is finite. □

Now, we can define for a branch B the index $VInd(B)$ as being the first node in B below $RInd(B)$ which is not lying on an important branch strictly inferior with respect to comparing the $RInd$ values. Formally

$$VInd(B) = \min\{x \in B \ :\ RInd(B) \sqsubseteq x,\ \forall B'.\ RInd(B') \sqsubset RInd(B) \rightarrow x \notin B'\}.$$

This definition is sound thanks to Lemma 30. Furthermore, $VInd$ and $RInd$ can be related in the following way.

Lemma 31 *$\mathit{VInd}(B) \sqsubseteq \mathit{VInd}(B')$ implies $\mathit{RInd}(B) \sqsubseteq \mathit{RInd}(B')$.*

Proof. Assume $\mathit{VInd}(B) \sqsubseteq \mathit{VInd}(B')$. Since $\mathit{VInd}(B') \in B'$ we obtain $\mathit{VInd}(B) \in B'$. As by definition $\mathit{RInd}(B) \sqsubseteq \mathit{VInd}(B)$, we also have $\mathit{RInd}(B) \in B'$. Consequently $\mathit{RInd}(B)$ and $\mathit{RInd}(B')$ lie on the same branch B', and thus are comparable. For the sake of contradiction, suppose $\mathit{RInd}(B') \sqsubset \mathit{RInd}(B)$, then by definition of VInd we obtain $\mathit{VInd}(B) \notin B'$. Contradiction. The remaining case is the expected $\mathit{RInd}(B) \sqsubseteq \mathit{RInd}(B')$. □

We are ready to establish the two properties wanted for VInd.

Lemma 32 *The following holds.*

(1) For every node x, $|\mathit{VInd}^{-1}(x)| < K_{\mathrm{im}}$.
(2) If $\mathit{VInd}(B') \sqsubset \mathit{VInd}(B)$, then $\mathit{VInd}(B) \notin B'$.

Proof. (1): Let B be an infinite branch such that $\mathit{VInd}(B) = x$ and let y be $\mathit{RInd}(B)$. By Lemma 31, important branches with the same VInd also have the same RInd and hence $\mathit{VInd}^{-1}(x) \subseteq \mathit{RInd}^{-1}(y)$. The desired bound follows from Lemma 29.

(2): If $\mathit{VInd}(B) \in B'$, then $\mathit{RInd}(B) \sqsubseteq \mathit{RInd}(B')$ by definition of VInd. This implies $\mathit{VInd}(B) \sqsubseteq \mathit{VInd}(B')$, again by the definition of VInd. □

Now, for $X \in \mathit{Atoms}$ such that $I(X)$ is an infinite branch we define $\mathit{BIndex}(X)$ to be $\mathit{VInd}(I(X))$. The distribution induced by BIndex is strongly sparse:

Lemma 33 (strong sparsity) *For each finite zone Z of frontier F we have $|\mathit{BIndex}^{-1}(Z)| \leq K_{\mathrm{im}}^2 |F|$.*

Proof. Assume that $\mathit{VInd}(B), \mathit{VInd}(B') \in Z$ for two branches B and B'. From Lemma 32 (2) we can conclude that $B \cap F = B' \cap F$ implies $\mathit{VInd}(B) = \mathit{VInd}(B')$. And, according to Lemmas 28 and 32 (1), the number of atoms X that share the same value $\mathit{BIndex}(X)$ is bounded by K_{im}^2. This proves the claim. □

6.5 Car parking

In this section, our goal is to spread the indices around the tree such that each index ends in exactly one position. This can be seen as parking vehicles. In the beginning there are cars (indices) placed in the nodes of the tree, possibly more than one at the same position, and we aim at parking each of them in one node, i.e., attaching a single node to each of those cars. This is obviously not possible in general but we shall prove that, under a sparsity constraint on the distribution of vehicles, it is possible to attach a single parking place to each car, and furthermore that the mapping that, given a car, tells where to park it, can be described by a WMSO-formula.

For this purpose, we have to describe distributions and other kinds of mappings that involve integers in their domain or image by WMSO-formulas. These integers will always be bounded by some constant K and hence we can split the WMSO-definition into several formulas, one for each number that may be involved. Formally, we say that a relation $R \subseteq \mathit{dom}(t)^r \times I$ for some finite $I \subseteq \mathbb{Z}$ is WMSO-definable if there are WMSO-formulas $\phi_i(x_1, \dots, x_r)$ for each $i \in I$ such that $(u_1, \dots, u_r, i) \in R$ iff $t, u_1, \dots, u_r \models \phi_i(x_1, \dots, x_r)$. Note that a K-sparse distribution can be seen as a relation of this kind since $0 \leq D(x) \leq 3K$ for every $x \in \mathit{dom}(t)$.

Definition 34 *Given a distribution D, a* placement *P for D is an injective partial mapping from $dom(t) \times \mathbb{N}$ to $dom(t)$ such that $P(x,i)$ is defined iff $i \in [D(x)]$.*

A flow, defined below, can be seen as a kind of instruction on how to spread the values of a distribution to obtain a placement. In the vehicles description, this is the number of cars which will have to cross an edge in order to reach the final placement.

Definition 35 *A* flow *is a mapping f from the nodes of t to $\mathbb{Z}$. A flow f is* compatible *with a distribution D if for all inner nodes x, $D(x) + f(x) \leq 1 + f(x0) + f(x1)$ and for every leaf x, $D(x) + f(x) \leq 1$. A flow f is* bounded *by a constant K if $|f(x)| \leq K$ for each node x.*

In this definition, $f(x)$ is interpreted as the number of cars crossing the edge from the ancestor of x to x. In case of a negative value, $-f(x)$ cars are driving from x to its ancestor. The condition of f being compatible with D states that after distributing all the cars according to the flow there is at most one car remaining at each node. One should note here that, according to our definition, it is possible that $f(\epsilon) < 0$. With the above intuition this would mean that one has to send $-f(\epsilon)$ cars to the (non-existing) ancestor of the root. We need this case when constructing flows on finite subtrees of a given infinite tree.

In the following we show that for a K-sparse distribution there is a compatible flow that is bounded. From this flow we then compute a placement for the distribution. We start by defining a flow on finite trees (which can then also be used to deal with finite subtrees of a given infinite tree).

Lemma 36 *For every finite tree t and every WMSO-definable K-sparse distribution D over t, there exists a WMSO-definable flow f that is compatible with D, bounded by $2K+1$, and such that for each node x there is a zone Z_x rooted in x of frontier F_x with*

(1) $D(Z_x) + f(x) = |Z_x| + K(|F_x| - 1)$,
(2) and $f(y) \geq K$ for all $y \in F_x$ different from x.

Proof. First note that (1) implies $f(x) \geq -K$ for each x since D is K-sparse. We define the values $f(x)$ and the zones Z_x inductively. These definitions directly imply that f is compatible with D.

The base case of a leaf x is straightforward: we set $f(x)$ to be $1 - D(x)$ and $Z_x = \{x\}$. By the hypothesis of sparsity, we have that $D(x) \leq K+1$ and hence $|f(x)|$ is bounded by $2K+1$.

Let x be an inner node. Let $f'(x0)$ and $f'(x1)$ be respectively $\min(K, f(x0))$ and $\min(K, f(x1))$. Let us now define $f(x)$ to be $1 + f'(x0) + f'(x1) - D(x)$ and Z_x to contain the node x, the nodes of Z_{x0} if $f(x0) < K$, and the nodes of Z_{x1} if $f(x1) < K$. By the hypothesis of induction, we indeed obtain that $D(Z_x) + f(x) = |Z_x| + K(|F_x| - 1)$. We illustrate this only for the case $f'(x0) < K$ and $f'(x1) = K$, the other cases are similar. In this case $Z_x = Z_{x0} \cup \{x\}$, $|F_x| = |F_{x0}| + 1$, and $f(x) = 1 + f(x0) + K - D(x)$. From this we get the following sequence of equalities:

$$\begin{aligned}
D(Z_x) + f(x) &= D(Z_{x0}) + D(x) + 1 + f(x0) + K - D(x) \\
&= D(Z_{x0}) + f(x0) + 1 + K \\
&= |Z_{x0}| + K(|F_{x0}| - 1) + 1 + K \\
&= |Z_{x0}| + 1 + K(|F_{x0}|) \\
&= |Z_x| + K(|F_x| - 1).
\end{aligned}$$

From the definition of $f(x)$ it is clear that $f(x) \leq 2K + 1$. As mentioned before, $f(x) \geq -K$ and thus $|f(x)|$ is bounded by $2K + 1$.

It is obvious that this flow, which has only a bounded number of possible values and is defined inductively, is WMSO-definable. This definition can be done by requiring the existence of sets $X_{-K}, \ldots, X_{2K+1}$ such that a node x is in X_i iff $f(x) = i$. This can be directly expressed if x is a leaf. Otherwise it is a simple statement on the membership of the successors $x0$ and $x1$ of x. □

Lemma 37 *For every infinite tree t and every WMSO-definable K-sparse distribution D, there exists a WMSO-definable flow f bounded by $7K$ that is compatible with D such that $f(\epsilon) = 0$.*

Proof. As a K-sparse distribution restricted to a finite subtree of t is also K-sparse on this subtree, we can apply Lemma 36 to define the values of $f(x)$ for the nodes x that are not on infinite branches of t.

Let B be the set of nodes appearing in some infinite branch. We define inductively for any node $x \in B$ a flow $f(x)$ such that $0 \leq f(x) \leq 7K$. We only consider non-negative values since on infinite branches we never reach a leaf and hence there is no need for an upward flow.

We start by setting $f(\epsilon) = 0$. For $x \neq \epsilon$ let $y \in B$ be the father of x. Three cases may happen:

(a) If x is the only child of y in B, then we set $f(x) = \max(0, f(y) + D(y) - f(x') - 1)$ where x' is the other child of y.
(b) If the two children of y are in B and x is the left child, then we set $f(x) = \max(0, \min(5K, f(y) + D(y) - 1))$.
(c) If the two children of y are in B and x is the right child, then we set $f(x) = \max(0, f(y) + D(y) - 1 - 5K)$.

Obviously, $f(x) \geq 0$ in all cases. We show that if x is of type (b) or (c), then $f(x) \leq 5K$, and if x is of type (a), then $f(x) \leq 7K$.

In case (b), $f(x) \leq 5K$ follows directly from the definition and in case (c) from $f(y) \leq 7K$ (by induction) and $D(y) \leq 3K + 1$ (D is K-sparse).

For x as in (a) we cannot use a local argument but have to look at sequences of vertices of type (a) and then apply Lemma 36 together with the K-sparsity of D. For this purpose, let $y_n, \ldots, y_1$ be such that y_n is the father of x, y_{i-1} is the father of y_i for all $i \in \{2, \ldots, n\}$, $y_2, \ldots, y_n$ are of type (a), and y_1 is not of type (a), i.e., y_1 is of type (b) or (c), or y_1 is the root. Then we know that $f(y_1) \leq 5K$.

Let $x_1, \dots, x_n$ be such that y_i is the father of x_i, $x_n \neq x$, and $x_i \neq y_{i+1}$ for $i \in \{1, \dots, n-1\}$, i.e., x_i is the brother of y_{i+1}. For the x_i the flow is defined using Lemma 36. Hence there are zones Z_{x_i} rooted at x_i of frontier F_{x_i} such that

$$D(Z_{x_i}) + f(x_i) = |Z_{x_i}| + K(|F_{x_i}| - 1). \tag{1}$$

Let $Z = \bigcup_{i=1}^{n}(\{y_i\} \cup Z_{x_i})$ and let F be the frontier of Z. Then

$$|Z| = n + \sum_{i=1}^{n} |Z_{x_i}| \tag{2}$$

$$|F| = 2 + \sum_{i=1}^{n} (|F_{x_i}| - 1) \tag{3}$$

$$\sum_{i=1}^{n} D(y_i) = D(Z) - \sum_{i=1}^{n} D(Z_{x_i}) \tag{4}$$

Since $y_2, \dots, y_n$ are of type (a), we get

$$f(x) = f(y_1) + \sum_{i=1}^{n}(D(y_i) - 1 - f(x_i)).$$

We know that $f(y_1) \leq 5K$ and hence it remains to be shown that $\sum_{i=1}^{n}(D(y_i) - 1 - f(x_i)) \leq 2K$. This can be deduced as follows:

$$\begin{aligned}
\sum_{i=1}^{n}(D(y_i) - 1 - f(x_i)) &\overset{(4)}{=} D(Z) - n - \sum_{i=1}(D(Z_{x_i}) + f(x_i)) \\
&\overset{(1)}{=} D(Z) - n - \sum_{i=1}(|Z_{x_i}| + K(|F_{x_i}| - 1)) \\
&\overset{(3)}{=} D(Z) - n - K(|F| - 2) - \sum_{i=1}(|Z_{x_i}|) \\
&\overset{K\text{-sparse}}{\leq} |Z| + K|F| - n - K(|F| - 2) - \sum_{i=1}(|Z_{x_i}|) \\
&\overset{(2)}{=} 2K
\end{aligned}$$

That this flow f is compatible with D can be easily deduced from the definitions as well as the fact that it is WMSO-definable. □

We are now ready to establish our placement Lemma.

Lemma 38 (car parking) *For every tree t and every WMSO-definable K-sparse distribution D, there exists a WMSO-definable placement for D. If t is finite, we additionally require that $D(dom(t)) \leq |dom(t)|$.*

Proof. According to Lemmas 36 and 37 we know that there is a WMSO-definable flow f that is compatible with D. For simplicity, we first assume that $f(\epsilon) = 0$, which is the case for infinite trees (Lemma 37). If $f(\epsilon) > 0$ for finite trees, then we can simply redefine $f(\epsilon) = 0$ without changing the property of f being compatible with D. At the end of the proof we briefly explain how to treat the case $f(\epsilon) < 0$ for finite trees.

The general strategy for defining the placement is the following:

- From each node we send all the cars except one to its neighbors. The number of cars sent to each neighbor is described by the flow.
- If we follow this strategy, then each edge in t is crossed by the cars in only one direction. Hence, a car cannot visit twice the same node. This means that it might be sent up in the tree for some steps and from some point onwards it is only sent downwards.
- To be sure that each car will be parked after finite time we order all the cars that cross a node (as described by D and f) according to a fixed strategy and we also fix a scheme for distributing the cars to the neighboring nodes.
- This ordering will ensure that the index of a car decreases each time it is sent down in the tree. As described above, each car is sent up in the tree only a finite number of times. Hence, if we always park the car that is first in the ordering at a specific node, then each car will eventually be parked.

To show that this strategy can be realized by WMSO-formulas we first describe the ordering of cars that we use and then define formulas

- $\text{send}_i(x, y)$ meaning that the ith car at node x is sent to node y.
- $\text{drive}_{i,j}(x, y)$ meaning that the ith car at node x is sent to y and is car number j at y.
- $\text{start}_i(x)$ meaning that the ith car at node x does not come from another node.
- $\text{itinerary}_i(x, X_1, \ldots, X_K, y)$ meaning that the ith car at node x will be parked at node y using an itinerary that is described by the sets $X_1, \ldots, X_K$.

To define these formulas we first have to introduce some notation. To avoid case distinctions we define $f^+(x) = \max(f(x), 0)$ and $f^-(x) = \min(f(x), 0)$. Furthermore, we assume by convention that for a leaf x the values $f^+(x0)$, $f^+(x1)$, $f^-(x0)$, $f^-(x1)$ are defined, and are all set to 0.

Then the number of cars crossing a node x is $f^+(x) + f^-(x0) + f^-(x1) + D(x)$. Since all the values involved in this expression are bounded and since we can increase K without affecting the K-sparsity of D, we can assume that $f^+(x) + f^-(x0) + f^-(x1) + D(x) \leq K$ for all x.

Since f is WMSO-definable, we can also assume that there are formulas $\phi_i^+(x)$ and $\phi_i^-(x)$ defining f^+ and f^-. Then expressions of the form $i_1 + f^+(x) + f^-(y) = i_2$ for $i_1, i_2 \in [K]$ can easily be expressed as Boolean combinations of the formulas ϕ_i^+ and ϕ_i^-. The use of expressions of this kind simplifies the presentation of the formulas.

We start by giving the orderings used in the definitions of the formulas send_i and $\text{drive}_{i,j}$. The cars that cross a node x will be distributed in the following order that we refer to as the distribution order:

$f^+(x)$	$f^-(x0)$	$f^-(x1)$	$D(x)$

That is, we first distribute the cars that come from the father of x, then the cars that come from the left child of x, and so on. It remains to fix where to send the cars.

1	$f^-(x)$	$f^+(x0)$	$f^+(x1)$

This means that the first car in the distribution order is parked in the node x. The next cars are sent to the father of x if $f(x)$ is negative. The following cars in the distribution order are send to the left child of x and the remaining cars to the right child of x. This is expressed by the following formulas, where the first two are only defined for $2 \leq i \leq K$ because the first car is always kept at the current position.

– For $2 \leq i \leq K$:

$$\begin{aligned}\mathrm{send}_i(x,y) := {} & [((x = y0) \vee (x = y1)) \wedge (1 < i \leq C_1)] \\ & \vee [(x0 = y) \wedge (C_1 < i \leq C_2)] \\ & \vee [(x1 = y) \wedge (C_2 < i \leq C_3)\end{aligned}$$

Here $C_1 = 1 + f^-(x)$, $C_2 = 1 + f^-(x) + f^+(x0)$, and $C_3 = 1 + f^-(x) + f^+(x0) + f^+(x1)$.

– For $2 \leq i \leq K$:

$$\begin{aligned}\mathrm{drive}_{i,j}(x,y) := {} & \mathrm{send}_i(x,y) \wedge \\ & \big([(x = y0) \wedge ((i-1) + f^+(y) = j)] \\ & \vee [(x = y1) \wedge ((i-1) + f^+(y) + f^-(y0) = j)] \\ & \vee [(x0 = y) \wedge (i - 1 - f^-(x) = j)] \\ & \vee [(x1 = y) \wedge (i - 1 - f^-(x) - f^+(x0) = j)]\big)\end{aligned}$$

– For $1 \leq i \leq K$:

$$\begin{aligned}\mathrm{start}_i(x) := {} & (i \leq f^+(x) + f^-(x0) + f^-(x1) + D(x)) \\ & \wedge (\neg\exists y \ : \ \textstyle\bigvee_{j=2}^K \mathrm{drive}_{j,i}(y,x))\end{aligned}$$

– For $1 \leq i \leq K$:

$$\begin{aligned}\mathrm{itinerary}_i(x, X_1, \ldots, X_K, y) := {} & \mathrm{disjoint(X_1, \ldots, X_K)} \wedge \\ & x \in X_i \wedge \mathrm{start}_i(x) \wedge X_1 = \{y\} \\ & \wedge \bigwedge_{j=2}^{K} (\forall z \in X_j \bigvee_{j' \in [K]} \exists z' \in X_{j'} \ : \ \mathrm{drive}_{j,j'}(z,z'))\end{aligned}$$

Then the formulas $\psi_i(x,y)$ defining the placement are given by

$$\psi_i(x,y) = \exists X_1, \ldots X_K (\mathrm{itinerary}_i(x, X_1, \ldots, X_K, y)).$$

As mentioned at the beginning of the proof we now discuss how to treat the case $f(\epsilon) < 0$. Recall that this may only happen for finite trees. In general, simply redefining $f(\epsilon) = 0$ may lead to a flow that is not compatible with D anymore. Therefore, we have to use a different strategy.

The strategy for distributing the cars described above would lead to $f^-(\epsilon)$ cars that "get stuck" at the root because, following the flow, they should be sent upwards, and this is not possible. This means that there are at most K cars (for simplicity assume exactly K cars) that start at some node but never arrive at some destination, i.e., there are nodes $x_1, \ldots, x_K$ and $i_1, \ldots, i_K \in [K]$ such that

$$\mathrm{start}_{i_j}(x_j) \wedge \neg\exists y(\psi_{i_j}(x_j, y))$$

is satisfied. From the assumption $D(dom(t)) \leq |dom(t)|$ we can conclude that there remain at least K nodes where no car is parked, i.e., K nodes which are not image of the function defined by the ψ_i's. Let $y_1, \ldots, y_K$ be the K first such nodes for some WMSO-definable order (this is possible since the tree is finite). We can now extend this function by ordering the $x_1, \ldots, x_K$ and map the i_jth car from x_j to y_j. Note that these definitions are expressible in WMSO. In this way we obtain an modification of the function defined by the ψ_i's into a placement for D. □

6.6 Proof of Theorem 10

We can now prove Theorem 10 as stated in Section 4 by combining the previous results.

Proof (of Theorem 10). For $X \in \mathit{Atoms}$ let

$$\mathit{Index}(X) = \begin{cases} \mathit{BIndex}(X) \text{ if } I(X) \text{ is an infinite branch,} \\ \mathit{SIndex}(X) \text{ otherwise.} \end{cases}$$

We construct a formula $\phi_{\mathrm{ind}}(X, y)$ that associates to each $X \in \mathit{Atoms}$ its index $y = \mathit{Index}(X)$. From the definitions of $I(X)$, $\mathit{SIndex}(X)$, and $\mathit{BIndex}(X)$ it is clear that we can construct such a formula. Note that in the definition of this formula, we do not have to explicitly represent infinite sets (though $I(X)$ may be infinite) because we can construct a WMSO formula $\phi_{imp}(X, x)$ that associates to $X \in \mathit{Atoms}$ its important nodes. From this one can construct WMSO definitions of $\mathit{SIndex}(X)$ and $\mathit{BIndex}(X)$, and hence the formula $\phi_{\mathrm{ind}}(X, y)$ is also WMSO.

Then, we compute the distribution D defined by $D(x) = |\mathit{Index}^{-1}(x)|$. Since this distribution is $(K_{\mathrm{s}} + K_{\mathrm{im}}^2)$-sparse by Lemmas 27 and 33, it is also WMSO-definable using the formula ϕ_{ind}. Let K be a constant such that $D(x) \leq K$ for all nodes x. Applying Lemma 38, we obtain a WMSO-definable placement P for D. One should note that for finite t the assumption $D(\mathit{dom}(t)) \leq |\mathit{dom}(t)|$ is satisfied because $D(\mathit{dom}(t))$ is the number of elements in the set E, and $\mathcal{I}(t)$ being isomorphic to $\mathcal{P}^F(E)$ implies that t has at least as many elements as E.

Let $\psi_i(x_1, x_2)$ for $i \in [K]$ be the formulas defining P. Now, we order all the $X \in \mathit{Atoms}$ with the same index. A possible definition for such an ordering is $X < Y$ if the lexicographically smallest node that is not in $X \cap Y$ is in X. Then one can construct WMSO-formulas $\theta_i(X)$ stating that X is the ith set in the ordering among those that have the same index as X.

The WMSO-formula $\mathit{Code}(X, x)$ that attaches X to its final position x is then defined as follows:

$$\mathit{Code}(X, x) = \bigvee_{i \in [K]} (\theta_i(X) \wedge \exists y(\phi_{\mathrm{ind}}(X, y) \wedge \psi_i(y, x)).$$

□

Acknowledgments

We are particularly grateful to Vince Bárány for his contribution to the random-graph proof, as well as his comments on previous versions of this work. We also thank Achim Blumensath for commenting on earlier stages of this work.

References

[Bár06] Vince Bárány. Invariants of automatic presentations and semi-synchronous transductions. In *STACS 2006*, volume 3884 of *LNCS*, pages 289–300. Springer, 2006.

[BG00] A. Blumensath and E. Grädel. Automatic Structures. In *Proceedings of 15th IEEE Symposium on Logic in Computer Science LICS 2000*, pages 51–62, 2000.

[BG04] Achim Blumensath and Erich Grädel. Finite presentations of infinite structures: Automata and interpretations. *Theory of Computing Systems*, 37:641 – 674, 2004.

[Blu99] Achim Blumensath. Automatic structures. Diploma thesis, RWTH-Aachen, 1999.

[Blu01] Achim Blumensath. Prefix-recognisable graphs and monadic second-order logic. Technical Report AIB-06-2001, RWTH Aachen, May 2001.

[Cau96] Didier Caucal. On infinite transition graphs having a decidable monadic theory. In *ICALP'96*, volume 1099 of *LNCS*, pages 194–205. Springer, 1996.

[Cau02] Didier Caucal. On infinite terms having a decidable monadic theory. In *MFCS'02*, volume 2420 of *LNCS*, pages 165–176. Springer, 2002.

[CO06] Bruno Courcelle and Sang-il Oum. Vertex-minors, monadic second-order logic and a conjecture by seese. *Journal of Combinatorial Theory, Series B*, 2006. to appear.

[Col02] Thomas Colcombet. On families of graphs having a decidable first order theory with reachability. In *Proceedings of ICALP 2002*, volume 2380 of *LNCS*, pages 98–109. Springer, 2002.

[Col04] T. Colcombet. Equational presentations of tree-automatic structures. In *Workshop on Automata, Structures and Logic, Auckland, New Zealand*, December 2004.

[Cou97] B. Courcelle. The expression of graph properties and graph transformations in monadic second-order logic. In G. Rozenberg, editor, *Handbook of graph grammars and computing by graph transformations, Vol. 1 : Foundations*, chapter 5, pages 313–400. World Scientific, 1997.

[CT02] Olivier Carton and Wolfgang Thomas. The monadic theory of morphic infinite words and generalizations. *Information and Computation*, 176(1):51–65, 2002.

[CW03] Arnaud Carayol and Stefan Wöhrle. The Caucal hierarchy of infinite graphs in terms of logic and higher-order pushdown automata. In *FSTTCS'03*, volume 2914 of *LNCS*, pages 112–123. Springer, 2003.

[DT90] Max Dauchet and Sophie Tison. The theory of ground rewrite systems is decidable. In *LICS'90*, pages 242–248. IEEE, 1990.

[ECH+92] D. Epstein, J.W. Cannon, D.F. Holt, S.V.F. Levy, M.S. Paterson, and Thurston. *Word processing in groups.* Jones and Barlett publishers, 1992.

[Hod83] Bernard R. Hodgson. Décidabilité par automate fini. *Ann. Sci. Math. Québec*, 7(3):39–57, 1983.

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

[KL02] Dietrich Kuske and Markus Lohrey. On the theory of one-step rewriting in trace monoids. In *ICALP'02*, volume 2380 of *LNCS*, pages 752–763. Springer, 2002.

[KN95] Bakhadyr Khoussainov and Anil Nerode. Automatic presentations of structures. In *Workshop LCC '94*, volume 960 of *LNCS*, pages 367–392. Springer, 1995.

[KNRS04] Bakhadyr Khoussainov, André Nies, Sasha Rubin, and Frank Stephan. Automatic structures: Richness and limitations. In *LICS'04*, pages 44–53. IEEE Computer Society, 2004.

[KNUW05] Teodor Knapik, Damian Niwiński, Paweł Urzyczyn, and Igor Walukiewicz. Unsafe grammars, panic automata, and decidability. In *ICALP'05*, volume 3580 of *LNCS*, pages 1450–1461. Springer, 2005.

[KRS04] Bakhadyr Khoussainov, Sasha Rubin, and Frank Stephan. Definability and regularity in automatic structures. In *STACS'04*, volume 2996 of *LNCS*, pages 440–451. Springer, 2004.

[Mad03] P. Madhusudan. Model-checking trace event structures. In *LICS'03*, pages 371–380. IEEE Computer Society, 2003.

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

[Rub04] S. Rubin. *Automatic Structures.* PhD thesis, University of Auckland, New Zealand, 2004.

[See91] Detlef Seese. The structure of models of decidable monadic theories of graphs. *Annals of Pure and Applied Logic*, 53:169–195, 1991.

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

Aachener Informatik-Berichte

This is a list of recent technical reports. To obtain copies of technical reports please consult http://aib.informatik.rwth-aachen.de/ or send your request to: Informatik-Bibliothek, RWTH Aachen, Ahornstr. 55, 52056 Aachen, Email: `biblio@informatik.rwth-aachen.de`

1987-01 *	Fachgruppe Informatik: Jahresbericht 1986
1987-02 *	David de Frutos Escrig, Klaus Indermark: Equivalence Relations of Non-Deterministic Ianov-Schemes
1987-03 *	Manfred Nagl: A Software Development Environment based on Graph Technology
1987-04 *	Claus Lewerentz, Manfred Nagl, Bernhard Westfechtel: On Integration Mechanisms within a Graph-Based Software Development Environment
1987-05 *	Reinhard Rinn: Über Eingabeanomalien bei verschiedenen Inferenzmodellen
1987-06 *	Werner Damm, Gert Döhmen: Specifying Distributed Computer Architectures in AADL*
1987-07 *	Gregor Engels, Claus Lewerentz, Wilhelm Schäfer: Graph Grammar Engineering: A Software Specification Method
1987-08 *	Manfred Nagl: Set Theoretic Approaches to Graph Grammars
1987-09 *	Claus Lewerentz, Andreas Schürr: Experiences with a Database System for Software Documents
1987-10 *	Herbert Klaeren, Klaus Indermark: A New Implementation Technique for Recursive Function Definitions
1987-11 *	Rita Loogen: Design of a Parallel Programmable Graph Reduction Machine with Distributed Memory
1987-12	J. Börstler, U. Möncke, R. Wilhelm: Table compression for tree automata
1988-01 *	Gabriele Esser, Johannes Rückert, Frank Wagner Gesellschaftliche Aspekte der Informatik
1988-02 *	Peter Martini, Otto Spaniol: Token-Passing in High-Speed Backbone Networks for Campus-Wide Environments
1988-03 *	Thomas Welzel: Simulation of a Multiple Token Ring Backbone
1988-04 *	Peter Martini: Performance Comparison for HSLAN Media Access Protocols
1988-05 *	Peter Martini: Performance Analysis of Multiple Token Rings
1988-06 *	Andreas Mann, Johannes Rückert, Otto Spaniol: Datenfunknetze
1988-07 *	Andreas Mann, Johannes Rückert: Packet Radio Networks for Data Exchange
1988-08 *	Andreas Mann, Johannes Rückert: Concurrent Slot Assignment Protocol for Packet Radio Networks
1988-09 *	W. Kremer, F. Reichert, J. Rückert, A. Mann: Entwurf einer Netzwerktopologie für ein Mobilfunknetz zur Unterstützung des öffentlichen Straßenverkehrs
1988-10 *	Kai Jakobs: Towards User-Friendly Networking
1988-11 *	Kai Jakobs: The Directory - Evolution of a Standard
1988-12 *	Kai Jakobs: Directory Services in Distributed Systems - A Survey
1988-13 *	Martine Schümmer: RS-511, a Protocol for the Plant Floor

1988-14 * U. Quernheim: Satellite Communication Protocols - A Performance Comparison Considering On-Board Processing
1988-15 * Peter Martini, Otto Spaniol, Thomas Welzel: File Transfer in High Speed Token Ring Networks: Performance Evaluation by Approximate Analysis and Simulation
1988-16 * Fachgruppe Informatik: Jahresbericht 1987
1988-17 * Wolfgang Thomas: Automata on Infinite Objects
1988-18 * Michael Sonnenschein: On Petri Nets and Data Flow Graphs
1988-19 * Heiko Vogler: Functional Distribution of the Contextual Analysis in Block-Structured Programming Languages: A Case Study of Tree Transducers
1988-20 * Thomas Welzel: Einsatz des Simulationswerkzeuges QNAP2 zur Leistungsbewertung von Kommunikationsprotokollen
1988-21 * Th. Janning, C. Lewerentz: Integrated Project Team Management in a Software Development Environment
1988-22 * Joost Engelfriet, Heiko Vogler: Modular Tree Transducers
1988-23 * Wolfgang Thomas: Automata and Quantifier Hierarchies
1988-24 * Uschi Heuter: Generalized Definite Tree Languages
1989-01 * Fachgruppe Informatik: Jahresbericht 1988
1989-02 * G. Esser, J. Rückert, F. Wagner (Hrsg.): Gesellschaftliche Aspekte der Informatik
1989-03 * Heiko Vogler: Bottom-Up Computation of Primitive Recursive Tree Functions
1989-04 * Andy Schürr: Introduction to PROGRESS, an Attribute Graph Grammar Based Specification Language
1989-05 J. Börstler: Reuse and Software Development - Problems, Solutions, and Bibliography (in German)
1989-06 * Kai Jakobs: OSI - An Appropriate Basis for Group Communication?
1989-07 * Kai Jakobs: ISO's Directory Proposal - Evolution, Current Status and Future Problems
1989-08 * Bernhard Westfechtel: Extension of a Graph Storage for Software Documents with Primitives for Undo/Redo and Revision Control
1989-09 * Peter Martini: High Speed Local Area Networks - A Tutorial
1989-10 * P. Davids, Th. Welzel: Performance Analysis of DQDB Based on Simulation
1989-11 * Manfred Nagl (Ed.): Abstracts of Talks presented at the WG '89 15th International Workshop on Graphtheoretic Concepts in Computer Science
1989-12 * Peter Martini: The DQDB Protocol - Is it Playing the Game?
1989-13 * Martine Schümmer: CNC/DNC Communication with MAP
1989-14 * Martine Schümmer: Local Area Networks for Manufactoring Environments with hard Real-Time Requirements
1989-15 * M. Schümmer, Th. Welzel, P. Martini: Integration of Field Bus and MAP Networks - Hierarchical Communication Systems in Production Environments
1989-16 * G. Vossen, K.-U. Witt: SUXESS: Towards a Sound Unification of Extensions of the Relational Data Model

1989-17 * J. Derissen, P. Hruschka, M.v.d. Beeck, Th. Janning, M. Nagl: Integrating Structured Analysis and Information Modelling
1989-18 A. Maassen: Programming with Higher Order Functions
1989-19 * Mario Rodriguez-Artalejo, Heiko Vogler: A Narrowing Machine for Syntax Directed BABEL
1989-20 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Graph-based Implementation of a Functional Logic Language
1990-01 * Fachgruppe Informatik: Jahresbericht 1989
1990-02 * Vera Jansen, Andreas Potthoff, Wolfgang Thomas, Udo Wermuth: A Short Guide to the AMORE System (Computing Automata, MOnoids and Regular Expressions)
1990-03 * Jerzy Skurczynski: On Three Hierarchies of Weak SkS Formulas
1990-04 R. Loogen: Stack-based Implementation of Narrowing
1990-05 H. Kuchen, A. Wagener: Comparison of Dynamic Load Balancing Strategies
1990-06 * Kai Jakobs, Frank Reichert: Directory Services for Mobile Communication
1990-07 * Kai Jakobs: What's Beyond the Interface - OSI Networks to Support Cooperative Work
1990-08 * Kai Jakobs: Directory Names and Schema - An Evaluation
1990-09 * Ulrich Quernheim, Dieter Kreuer: Das CCITT - Signalisierungssystem Nr. 7 auf Satellitenstrecken; Simulation der Zeichengabestrecke
1990-11 H. Kuchen, R. Loogen, J.J. Moreno Navarro, M. Rodriguez Artalejo: Lazy Narrowing in a Graph Machine
1990-12 * Kai Jakobs, Josef Kaltwasser, Frank Reichert, Otto Spaniol: Der Computer fährt mit
1990-13 * Rudolf Mathar, Andreas Mann: Analyzing a Distributed Slot Assignment Protocol by Markov Chains
1990-14 A. Maassen: Compilerentwicklung in Miranda - ein Praktikum in funktionaler Programmierung (written in german)
1990-15 * Manfred Nagl, Andreas Schürr: A Specification Environment for Graph Grammars
1990-16 A. Schürr: PROGRESS: A VHL-Language Based on Graph Grammars
1990-17 * Marita Möller: Ein Ebenenmodell wissensbasierter Konsultationen - Unterstützung für Wissensakquisition und Erklärungsfähigkeit
1990-18 * Eric Kowalewski: Entwurf und Interpretation einer Sprache zur Beschreibung von Konsultationsphasen in Expertensystemen
1990-20 Y. Ortega Mallen, D. de Frutos Escrig: A Complete Proof System for Timed Observations
1990-21 * Manfred Nagl: Modelling of Software Architectures: Importance, Notions, Experiences
1990-22 H. Fassbender, H. Vogler: A Call-by-need Implementation of Syntax Directed Functional Programming
1991-01 Guenther Geiler (ed.), Fachgruppe Informatik: Jahresbericht 1990
1991-03 B. Steffen, A. Ingolfsdottir: Characteristic Formulae for Processes with Divergence
1991-04 M. Portz: A new class of cryptosystems based on interconnection networks

1991-05 H. Kuchen, G. Geiler: Distributed Applicative Arrays

1991-06 * Ludwig Staiger: Kolmogorov Complexity and Hausdorff Dimension

1991-07 * Ludwig Staiger: Syntactic Congruences for w-languages

1991-09 * Eila Kuikka: A Proposal for a Syntax-Directed Text Processing System

1991-10 K. Gladitz, H. Fassbender, H. Vogler: Compiler-based Implementation of Syntax-Directed Functional Programming

1991-11 R. Loogen, St. Winkler: Dynamic Detection of Determinism in Functional Logic Languages

1991-12 * K. Indermark, M. Rodriguez Artalejo (Eds.): Granada Workshop on the Integration of Functional and Logic Programming

1991-13 * Rolf Hager, Wolfgang Kremer: The Adaptive Priority Scheduler: A More Fair Priority Service Discipline

1991-14 * Andreas Fasbender, Wolfgang Kremer: A New Approximation Algorithm for Tandem Networks with Priority Nodes

1991-15 J. Börstler, A. Zündorf: Revisiting extensions to Modula-2 to support reusability

1991-16 J. Börstler, Th. Janning: Bridging the gap between Requirements Analysis and Design

1991-17 A. Zündorf, A. Schürr: Nondeterministic Control Structures for Graph Rewriting Systems

1991-18 * Matthias Jarke, John Mylopoulos, Joachim W. Schmidt, Yannis Vassiliou: DAIDA: An Environment for Evolving Information Systems

1991-19 M. Jeusfeld, M. Jarke: From Relational to Object-Oriented Integrity Simplification

1991-20 G. Hogen, A. Kindler, R. Loogen: Automatic Parallelization of Lazy Functional Programs

1991-21 * Prof. Dr. rer. nat. Otto Spaniol: ODP (Open Distributed Processing): Yet another Viewpoint

1991-22 H. Kuchen, F. Lücking, H. Stoltze: The Topology Description Language TDL

1991-23 S. Graf, B. Steffen: Compositional Minimization of Finite State Systems

1991-24 R. Cleaveland, J. Parrow, B. Steffen: The Concurrency Workbench: A Semantics Based Tool for the Verification of Concurrent Systems

1991-25 * Rudolf Mathar, Jürgen Mattfeldt: Optimal Transmission Ranges for Mobile Communication in Linear Multihop Packet Radio Networks

1991-26 M. Jeusfeld, M. Staudt: Query Optimization in Deductive Object Bases

1991-27 J. Knoop, B. Steffen: The Interprocedural Coincidence Theorem

1991-28 J. Knoop, B. Steffen: Unifying Strength Reduction and Semantic Code Motion

1991-30 T. Margaria: First-Order theories for the verification of complex FSMs

1991-31 B. Steffen: Generating Data Flow Analysis Algorithms from Modal Specifications

1992-01 Stefan Eherer (ed.), Fachgruppe Informatik: Jahresbericht 1991

1992-02 * Bernhard Westfechtel: Basismechanismen zur Datenverwaltung in strukturbezogenen Hypertextsystemen

1992-04 S. A. Smolka, B. Steffen: Priority as Extremal Probability

1992-05 * Matthias Jarke, Carlos Maltzahn, Thomas Rose: Sharing Processes: Team Coordination in Design Repositories

1992-06 O. Burkart, B. Steffen: Model Checking for Context-Free Processes

1992-07 * Matthias Jarke, Klaus Pohl: Information Systems Quality and Quality Information Systems

1992-08 * Rudolf Mathar, Jürgen Mattfeldt: Analyzing Routing Strategy NFP in Multihop Packet Radio Networks on a Line

1992-09 * Alfons Kemper, Guido Moerkotte: Grundlagen objektorientierter Datenbanksysteme

1992-10 Matthias Jarke, Manfred Jeusfeld, Andreas Miethsam, Michael Gocek: Towards a logic-based reconstruction of software configuration management

1992-11 Werner Hans: A Complete Indexing Scheme for WAM-based Abstract Machines

1992-12 W. Hans, R. Loogen, St. Winkler: On the Interaction of Lazy Evaluation and Backtracking

1992-13 * Matthias Jarke, Thomas Rose: Specification Management with CAD

1992-14 Th. Noll, H. Vogler: Top-down Parsing with Simultaneous Evaluation on Noncircular Attribute Grammars

1992-15 A. Schuerr, B. Westfechtel: Graphgrammatiken und Graphersetzungssysteme(written in german)

1992-16 * Graduiertenkolleg Informatik und Technik (Hrsg.): Forschungsprojekte des Graduiertenkollegs Informatik und Technik

1992-17 M. Jarke (ed.): ConceptBase V3.1 User Manual

1992-18 * Clarence A. Ellis, Matthias Jarke (Eds.): Distributed Cooperation in Integrated Information Systems - Proceedings of the Third International Workshop on Intelligent and Cooperative Information Systems

1992-19-00 H. Kuchen, R. Loogen (eds.): Proceedings of the 4th Int. Workshop on the Parallel Implementation of Functional Languages

1992-19-01 G. Hogen, R. Loogen: PASTEL - A Parallel Stack-Based Implementation of Eager Functional Programs with Lazy Data Structures (Extended Abstract)

1992-19-02 H. Kuchen, K. Gladitz: Implementing Bags on a Shared Memory MIMD-Machine

1992-19-03 C. Rathsack, S.B. Scholz: LISA - A Lazy Interpreter for a Full-Fledged Lambda-Calculus

1992-19-04 T.A. Bratvold: Determining Useful Parallelism in Higher Order Functions

1992-19-05 S. Kahrs: Polymorphic Type Checking by Interpretation of Code

1992-19-06 M. Chakravarty, M. Köhler: Equational Constraints, Residuation, and the Parallel JUMP-Machine

1992-19-07 J. Seward: Polymorphic Strictness Analysis using Frontiers (Draft Version)

1992-19-08 D. Gärtner, A. Kimms, W. Kluge: pi-Red^+ - A Compiling Graph-Reduction System for a Full Fledged Lambda-Calculus

1992-19-09 D. Howe, G. Burn: Experiments with strict STG code

1992-19-10 J. Glauert: Parallel Implementation of Functional Languages Using Small Processes

1992-19-11 M. Joy, T. Axford: A Parallel Graph Reduction Machine

1992-19-12 A. Bennett, P. Kelly: Simulation of Multicache Parallel Reduction

1992-19-13 K. Langendoen, D.J. Agterkamp: Cache Behaviour of Lazy Functional Programs (Working Paper)

1992-19-14 K. Hammond, S. Peyton Jones: Profiling scheduling strategies on the GRIP parallel reducer

1992-19-15 S. Mintchev: Using Strictness Information in the STG-machine

1992-19-16 D. Rushall: An Attribute Grammar Evaluator in Haskell

1992-19-17 J. Wild, H. Glaser, P. Hartel: Statistics on storage management in a lazy functional language implementation

1992-19-18 W.S. Martins: Parallel Implementations of Functional Languages

1992-19-19 D. Lester: Distributed Garbage Collection of Cyclic Structures (Draft version)

1992-19-20 J.C. Glas, R.F.H. Hofman, W.G. Vree: Parallelization of Branch-and-Bound Algorithms in a Functional Programming Environment

1992-19-21 S. Hwang, D. Rushall: The nu-STG machine: a parallelized Spineless Tagless Graph Reduction Machine in a distributed memory architecture (Draft version)

1992-19-22 G. Burn, D. Le Metayer: Cps-Translation and the Correctness of Optimising Compilers

1992-19-23 S.L. Peyton Jones, P. Wadler: Imperative functional programming (Brief summary)

1992-19-24 W. Damm, F. Liu, Th. Peikenkamp: Evaluation and Parallelization of Functions in Functional + Logic Languages (abstract)

1992-19-25 M. Kesseler: Communication Issues Regarding Parallel Functional Graph Rewriting

1992-19-26 Th. Peikenkamp: Charakterizing and representing neededness in functional loginc languages (abstract)

1992-19-27 H. Doerr: Monitoring with Graph-Grammars as formal operational Models

1992-19-28 J. van Groningen: Some implementation aspects of Concurrent Clean on distributed memory architectures

1992-19-29 G. Ostheimer: Load Bounding for Implicit Parallelism (abstract)

1992-20 H. Kuchen, F.J. Lopez Fraguas, J.J. Moreno Navarro, M. Rodriguez Artalejo: Implementing Disequality in a Lazy Functional Logic Language

1992-21 H. Kuchen, F.J. Lopez Fraguas: Result Directed Computing in a Functional Logic Language

1992-22 H. Kuchen, J.J. Moreno Navarro, M.V. Hermenegildo: Independent AND-Parallel Narrowing

1992-23 T. Margaria, B. Steffen: Distinguishing Formulas for Free

1992-24 K. Pohl: The Three Dimensions of Requirements Engineering

1992-25 * R. Stainov: A Dynamic Configuration Facility for Multimedia Communications

1992-26 * Michael von der Beeck: Integration of Structured Analysis and Timed Statecharts for Real-Time and Concurrency Specification

1992-27 W. Hans, St. Winkler: Aliasing and Groundness Analysis of Logic Programs through Abstract Interpretation and its Safety

1992-28 * Gerhard Steinke, Matthias Jarke: Support for Security Modeling in Information Systems Design

1992-29 B. Schinzel: Warum Frauenforschung in Naturwissenschaft und Technik

1992-30 A. Kemper, G. Moerkotte, K. Peithner: Object-Orientation Axiomatised by Dynamic Logic

1992-32 * Bernd Heinrichs, Kai Jakobs: Timer Handling in High-Performance Transport Systems

1992-33 * B. Heinrichs, K. Jakobs, K. Lenßen, W. Reinhardt, A. Spinner: Euro-Bridge: Communication Services for Multimedia Applications

1992-34 C. Gerlhof, A. Kemper, Ch. Kilger, G. Moerkotte: Partition-Based Clustering in Object Bases: From Theory to Practice

1992-35 J. Börstler: Feature-Oriented Classification and Reuse in IPSEN

1992-36 M. Jarke, J. Bubenko, C. Rolland, A. Sutcliffe, Y. Vassiliou: Theories Underlying Requirements Engineering: An Overview of NATURE at Genesis

1992-37 * K. Pohl, M. Jarke: Quality Information Systems: Repository Support for Evolving Process Models

1992-38 A. Zuendorf: Implementation of the imperative / rule based language PROGRES

1992-39 P. Koch: Intelligentes Backtracking bei der Auswertung funktional-logischer Programme

1992-40 * Rudolf Mathar, Jürgen Mattfeldt: Channel Assignment in Cellular Radio Networks

1992-41 * Gerhard Friedrich, Wolfgang Neidl: Constructive Utility in Model-Based Diagnosis Repair Systems

1992-42 * P. S. Chen, R. Hennicker, M. Jarke: On the Retrieval of Reusable Software Components

1992-43 W. Hans, St.Winkler: Abstract Interpretation of Functional Logic Languages

1992-44 N. Kiesel, A. Schuerr, B. Westfechtel: Design and Evaluation of GRAS, a Graph-Oriented Database System for Engineering Applications

1993-01 * Fachgruppe Informatik: Jahresbericht 1992

1993-02 * Patrick Shicheng Chen: On Inference Rules of Logic-Based Information Retrieval Systems

1993-03 G. Hogen, R. Loogen: A New Stack Technique for the Management of Runtime Structures in Distributed Environments

1993-05 A. Zündorf: A Heuristic for the Subgraph Isomorphism Problem in Executing PROGRES

1993-06 A. Kemper, D. Kossmann: Adaptable Pointer Swizzling Strategies in Object Bases: Design, Realization, and Quantitative Analysis

1993-07 * Graduiertenkolleg Informatik und Technik (Hrsg.): Graduiertenkolleg Informatik und Technik

1993-08 * Matthias Berger: k-Coloring Vertices using a Neural Network with Convergence to Valid Solutions

1993-09 M. Buchheit, M. Jeusfeld, W. Nutt, M. Staudt: Subsumption between Queries to Object-Oriented Databases

1993-10 O. Burkart, B. Steffen: Pushdown Processes: Parallel Composition and Model Checking

1993-11 * R. Große-Wienker, O. Hermanns, D. Menzenbach, A. Pollacks, S. Repetzki, J. Schwartz, K. Sonnenschein, B. Westfechtel: Das SUKITS-Projekt: A-posteriori-Integration heterogener CIM-Anwendungssysteme

1993-12 * Rudolf Mathar, Jürgen Mattfeldt: On the Distribution of Cumulated Interference Power in Rayleigh Fading Channels
1993-13 O. Maler, L. Staiger: On Syntactic Congruences for omega-languages
1993-14 M. Jarke, St. Eherer, R. Gallersdoerfer, M. Jeusfeld, M. Staudt: ConceptBase - A Deductive Object Base Manager
1993-15 M. Staudt, H.W. Nissen, M.A. Jeusfeld: Query by Class, Rule and Concept
1993-16 * M. Jarke, K. Pohl, St. Jacobs et al.: Requirements Engineering: An Integrated View of Representation Process and Domain
1993-17 * M. Jarke, K. Pohl: Establishing Vision in Context: Towards a Model of Requirements Processes
1993-18 W. Hans, H. Kuchen, St. Winkler: Full Indexing for Lazy Narrowing
1993-19 W. Hans, J.J. Ruz, F. Saenz, St. Winkler: A VHDL Specification of a Shared Memory Parallel Machine for Babel
1993-20 * K. Finke, M. Jarke, P. Szczurko, R. Soltysiak: Quality Management for Expert Systems in Process Control
1993-21 M. Jarke, M.A. Jeusfeld, P. Szczurko: Three Aspects of Intelligent Cooperation in the Quality Cycle
1994-01 Margit Generet, Sven Martin (eds.), Fachgruppe Informatik: Jahresbericht 1993
1994-02 M. Lefering: Development of Incremental Integration Tools Using Formal Specifications
1994-03 * P. Constantopoulos, M. Jarke, J. Mylopoulos, Y. Vassiliou: The Software Information Base: A Server for Reuse
1994-04 * Rolf Hager, Rudolf Mathar, Jürgen Mattfeldt: Intelligent Cruise Control and Reliable Communication of Mobile Stations
1994-05 * Rolf Hager, Peter Hermesmann, Michael Portz: Feasibility of Authentication Procedures within Advanced Transport Telematics
1994-06 * Claudia Popien, Bernd Meyer, Axel Kuepper: A Formal Approach to Service Import in ODP Trader Federations
1994-07 P. Peters, P. Szczurko: Integrating Models of Quality Management Methods by an Object-Oriented Repository
1994-08 * Manfred Nagl, Bernhard Westfechtel: A Universal Component for the Administration in Distributed and Integrated Development Environments
1994-09 * Patrick Horster, Holger Petersen: Signatur- und Authentifikationsverfahren auf der Basis des diskreten Logarithmusproblems
1994-11 A. Schürr: PROGRES, A Visual Language and Environment for PROgramming with Graph REwrite Systems
1994-12 A. Schürr: Specification of Graph Translators with Triple Graph Grammars
1994-13 A. Schürr: Logic Based Programmed Structure Rewriting Systems
1994-14 L. Staiger: Codes, Simplifying Words, and Open Set Condition
1994-15 * Bernhard Westfechtel: A Graph-Based System for Managing Configurations of Engineering Design Documents
1994-16 P. Klein: Designing Software with Modula-3
1994-17 I. Litovsky, L. Staiger: Finite acceptance of infinite words

1994-18 G. Hogen, R. Loogen: Parallel Functional Implementations: Graphbased vs. Stackbased Reduction

1994-19 M. Jeusfeld, U. Johnen: An Executable Meta Model for Re-Engineering of Database Schemas

1994-20 * R. Gallersdörfer, M. Jarke, K. Klabunde: Intelligent Networks as a Data Intensive Application (INDIA)

1994-21 M. Mohnen: Proving the Correctness of the Static Link Technique Using Evolving Algebras

1994-22 H. Fernau, L. Staiger: Valuations and Unambiguity of Languages, with Applications to Fractal Geometry

1994-24 * M. Jarke, K. Pohl, R. Dömges, St. Jacobs, H. W. Nissen: Requirements Information Management: The NATURE Approach

1994-25 * M. Jarke, K. Pohl, C. Rolland, J.-R. Schmitt: Experience-Based Method Evaluation and Improvement: A Process Modeling Approach

1994-26 * St. Jacobs, St. Kethers: Improving Communication and Decision Making within Quality Function Deployment

1994-27 * M. Jarke, H. W. Nissen, K. Pohl: Tool Integration in Evolving Information Systems Environments

1994-28 O. Burkart, D. Caucal, B. Steffen: An Elementary Bisimulation Decision Procedure for Arbitrary Context-Free Processes

1995-01 * Fachgruppe Informatik: Jahresbericht 1994

1995-02 Andy Schürr, Andreas J. Winter, Albert Zündorf: Graph Grammar Engineering with PROGRES

1995-03 Ludwig Staiger: A Tight Upper Bound on Kolmogorov Complexity by Hausdorff Dimension and Uniformly Optimal Prediction

1995-04 Birgitta König-Ries, Sven Helmer, Guido Moerkotte: An experimental study on the complexity of left-deep join ordering problems for cyclic queries

1995-05 Sophie Cluet, Guido Moerkotte: Efficient Evaluation of Aggregates on Bulk Types

1995-06 Sophie Cluet, Guido Moerkotte: Nested Queries in Object Bases

1995-07 Sophie Cluet, Guido Moerkotte: Query Optimization Techniques Exploiting Class Hierarchies

1995-08 Markus Mohnen: Efficient Compile-Time Garbage Collection for Arbitrary Data Structures

1995-09 Markus Mohnen: Functional Specification of Imperative Programs: An Alternative Point of View of Functional Languages

1995-10 Rainer Gallersdörfer, Matthias Nicola: Improving Performance in Replicated Databases through Relaxed Coherency

1995-11 * M.Staudt, K.von Thadden: Subsumption Checking in Knowledge Bases

1995-12 * G.V.Zemanek, H.W.Nissen, H.Hubert, M.Jarke: Requirements Analysis from Multiple Perspectives: Experiences with Conceptual Modeling Technology

1995-13 * M.Staudt, M.Jarke: Incremental Maintenance of Externally Materialized Views

1995-14 * P.Peters, P.Szczurko, M.Jeusfeld: Oriented Information Management: Conceptual Models at Work

1995-15 * Matthias Jarke, Sudha Ram (Hrsg.): WITS 95 Proceedings of the 5th Annual Workshop on Information Technologies and Systems
1995-16 * W.Hans, St.Winkler, F.Saenz: Distributed Execution in Functional Logic Programming
1996-01 * Jahresbericht 1995
1996-02 Michael Hanus, Christian Prehofer: Higher-Order Narrowing with Definitional Trees
1996-03 * W.Scheufele, G.Moerkotte: Optimal Ordering of Selections and Joins in Acyclic Queries with Expensive Predicates
1996-04 Klaus Pohl: PRO-ART: Enabling Requirements Pre-Traceability
1996-05 Klaus Pohl: Requirements Engineering: An Overview
1996-06 * M.Jarke, W.Marquardt: Design and Evaluation of Computer–Aided Process Modelling Tools
1996-07 Olaf Chitil: The Sigma-Semantics: A Comprehensive Semantics for Functional Programs
1996-08 * S.Sripada: On Entropy and the Limitations of the Second Law of Thermodynamics
1996-09 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP96 - Fifth International Conference on Algebraic and Logic Programming
1996-09-0 Michael Hanus (Ed.): Proceedings of the Poster Session of ALP 96 - Fifth International Conference on Algebraic and Logic Programming: Introduction and table of contents
1996-09-1 Ilies Alouini: An Implementation of Conditional Concurrent Rewriting on Distributed Memory Machines
1996-09-2 Olivier Danvy, Karoline Malmkjær: On the Idempotence of the CPS Transformation
1996-09-3 Víctor M. Gulías, José L. Freire: Concurrent Programming in Haskell
1996-09-4 Sébastien Limet, Pierre Réty: On Decidability of Unifiability Modulo Rewrite Systems
1996-09-5 Alexandre Tessier: Declarative Debugging in Constraint Logic Programming
1996-10 Reidar Conradi, Bernhard Westfechtel: Version Models for Software Configuration Management
1996-11 * C.Weise, D.Lenzkes: A Fast Decision Algorithm for Timed Refinement
1996-12 * R.Dömges, K.Pohl, M.Jarke, B.Lohmann, W.Marquardt: PRO-ART/CE* — An Environment for Managing the Evolution of Chemical Process Simulation Models
1996-13 * K.Pohl, R.Klamma, K.Weidenhaupt, R.Dömges, P.Haumer, M.Jarke: A Framework for Process-Integrated Tools
1996-14 * R.Gallersdörfer, K.Klabunde, A.Stolz, M.Eßmajor: INDIA — Intelligent Networks as a Data Intensive Application, Final Project Report, June 1996
1996-15 * H.Schimpe, M.Staudt: VAREX: An Environment for Validating and Refining Rule Bases
1996-16 * M.Jarke, M.Gebhardt, S.Jacobs, H.Nissen: Conflict Analysis Across Heterogeneous Viewpoints: Formalization and Visualization
1996-17 Manfred A. Jeusfeld, Tung X. Bui: Decision Support Components on the Internet

1996-18 Manfred A. Jeusfeld, Mike Papazoglou: Information Brokering: Design, Search and Transformation

1996-19 * P.Peters, M.Jarke: Simulating the impact of information flows in networked organizations

1996-20 Matthias Jarke, Peter Peters, Manfred A. Jeusfeld: Model-driven planning and design of cooperative information systems

1996-21 * G.de Michelis, E.Dubois, M.Jarke, F.Matthes, J.Mylopoulos, K.Pohl, J.Schmidt, C.Woo, E.Yu: Cooperative information systems: a manifesto

1996-22 * S.Jacobs, M.Gebhardt, S.Kethers, W.Rzasa: Filling HTML forms simultaneously: CoWeb architecture and functionality

1996-23 * M.Gebhardt, S.Jacobs: Conflict Management in Design

1997-01 Michael Hanus, Frank Zartmann (eds.): Jahresbericht 1996

1997-02 Johannes Faassen: Using full parallel Boltzmann Machines for Optimization

1997-03 Andreas Winter, Andy Schürr: Modules and Updatable Graph Views for PROgrammed Graph REwriting Systems

1997-04 Markus Mohnen, Stefan Tobies: Implementing Context Patterns in the Glasgow Haskell Compiler

1997-05 * S.Gruner: Schemakorrespondenzaxiome unterstützen die paargrammatische Spezifikation inkrementeller Integrationswerkzeuge

1997-06 Matthias Nicola, Matthias Jarke: Design and Evaluation of Wireless Health Care Information Systems in Developing Countries

1997-07 Petra Hofstedt: Taskparallele Skelette für irregulär strukturierte Probleme in deklarativen Sprachen

1997-08 Dorothea Blostein, Andy Schürr: Computing with Graphs and Graph Rewriting

1997-09 Carl-Arndt Krapp, Bernhard Westfechtel: Feedback Handling in Dynamic Task Nets

1997-10 Matthias Nicola, Matthias Jarke: Integrating Replication and Communication in Performance Models of Distributed Databases

1997-11 * R. Klamma, P. Peters, M. Jarke: Workflow Support for Failure Management in Federated Organizations

1997-13 Markus Mohnen: Optimising the Memory Management of Higher-Order Functional Programs

1997-14 Roland Baumann: Client/Server Distribution in a Structure-Oriented Database Management System

1997-15 George Botorog: High-Level Parallel Programming and the Efficient Implementation of Numerical Algorithms

1998-01 * Fachgruppe Informatik: Jahresbericht 1997

1998-02 Stefan Gruner, Manfred Nagel, Andy Schürr: Fine-grained and Structure-Oriented Document Integration Tools are Needed for Development Processes

1998-03 Stefan Gruner: Einige Anmerkungen zur graphgrammatischen Spezifikation von Integrationswerkzeugen nach Westfechtel, Janning, Lefering und Schürr

1998-04 * O. Kubitz: Mobile Robots in Dynamic Environments

1998-05 Martin Leucker, Stephan Tobies: Truth - A Verification Platform for Distributed Systems

1998-06 * Matthias Oliver Berger: DECT in the Factory of the Future
1998-07 M. Arnold, M. Erdmann, M. Glinz, P. Haumer, R. Knoll, B. Paech, K. Pohl, J. Ryser, R. Studer, K. Weidenhaupt: Survey on the Scenario Use in Twelve Selected Industrial Projects
1998-09 * Th. Lehmann: Geometrische Ausrichtung medizinischer Bilder am Beispiel intraoraler Radiographien
1998-10 * M. Nicola, M. Jarke: Performance Modeling of Distributed and Replicated Databases
1998-11 * Ansgar Schleicher, Bernhard Westfechtel, Dirk Jäger: Modeling Dynamic Software Processes in UML
1998-12 * W. Appelt, M. Jarke: Interoperable Tools for Cooperation Support using the World Wide Web
1998-13 Klaus Indermark: Semantik rekursiver Funktionsdefinitionen mit Striktheitsinformation
1999-01 * Jahresbericht 1998
1999-02 * F. Huch: Verifcation of Erlang Programs using Abstract Interpretation and Model Checking — Extended Version
1999-03 * R. Gallersdörfer, M. Jarke, M. Nicola: The ADR Replication Manager
1999-04 María Alpuente, Michael Hanus, Salvador Lucas, Germán Vidal: Specialization of Functional Logic Programs Based on Needed Narrowing
1999-05 * W. Thomas (Ed.): DLT 99 - Developments in Language Theory Fourth International Conference
1999-06 * Kai Jakobs, Klaus-Dieter Kleefeld: Informationssysteme für die angewandte historische Geographie
1999-07 Thomas Wilke: CTL+ is exponentially more succinct than CTL
1999-08 Oliver Matz: Dot-Depth and Monadic Quantifier Alternation over Pictures
2000-01 * Jahresbericht 1999
2000-02 Jens Vöge, Marcin Jurdzinski A Discrete Strategy Improvement Algorithm for Solving Parity Games
2000-03 D. Jäger, A. Schleicher, B. Westfechtel: UPGRADE: A Framework for Building Graph-Based Software Engineering Tools
2000-04 Andreas Becks, Stefan Sklorz, Matthias Jarke: Exploring the Semantic Structure of Technical Document Collections: A Cooperative Systems Approach
2000-05 Mareike Schoop: Cooperative Document Management
2000-06 Mareike Schoop, Christoph Quix (eds.): Proceedings of the Fifth International Workshop on the Language-Action Perspective on Communication Modelling
2000-07 * Markus Mohnen, Pieter Koopman (Eds.): Proceedings of the 12th International Workshop of Functional Languages
2000-08 Thomas Arts, Thomas Noll: Verifying Generic Erlang Client-Server Implementations
2001-01 * Jahresbericht 2000
2001-02 Benedikt Bollig, Martin Leucker: Deciding LTL over Mazurkiewicz Traces
2001-03 Thierry Cachat: The power of one-letter rational languages

2001-04	Benedikt Bollig, Martin Leucker, Michael Weber: Local Parallel Model Checking for the Alternation Free mu-Calculus
2001-05	Benedikt Bollig, Martin Leucker, Thomas Noll: Regular MSC Languages
2001-06	Achim Blumensath: Prefix-Recognisable Graphs and Monadic Second-Order Logic
2001-07	Martin Grohe, Stefan Wöhrle: An Existential Locality Theorem
2001-08	Mareike Schoop, James Taylor (eds.): Proceedings of the Sixth International Workshop on the Language-Action Perspective on Communication Modelling
2001-09	Thomas Arts, Jürgen Giesl: A collection of examples for termination of term rewriting using dependency pairs
2001-10	Achim Blumensath: Axiomatising Tree-interpretable Structures
2001-11	Klaus Indermark, Thomas Noll (eds.): Kolloquium Programmiersprachen und Grundlagen der Programmierung
2002-01 *	Jahresbericht 2001
2002-02	Jürgen Giesl, Aart Middeldorp: Transformation Techniques for Context-Sensitive Rewrite Systems
2002-03	Benedikt Bollig, Martin Leucker, Thomas Noll: Generalised Regular MSC Languages
2002-04	Jürgen Giesl, Aart Middeldorp: Innermost Termination of Context-Sensitive Rewriting
2002-05	Horst Lichter, Thomas von der Maßen, Thomas Weiler: Modelling Requirements and Architectures for Software Product Lines
2002-06	Henry N. Adorna: 3-Party Message Complexity is Better than 2-Party Ones for Proving Lower Bounds on the Size of Minimal Nondeterministic Finite Automata
2002-07	Jörg Dahmen: Invariant Image Object Recognition using Gaussian Mixture Densities
2002-08	Markus Mohnen: An Open Framework for Data-Flow Analysis in Java
2002-09	Markus Mohnen: Interfaces with Default Implementations in Java
2002-10	Martin Leucker: Logics for Mazurkiewicz traces
2002-11	Jürgen Giesl, Hans Zantema: Liveness in Rewriting
2003-01 *	Jahresbericht 2002
2003-02	Jürgen Giesl, René Thiemann: Size-Change Termination for Term Rewriting
2003-03	Jürgen Giesl, Deepak Kapur: Deciding Inductive Validity of Equations
2003-04	Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Improving Dependency Pairs
2003-05	Christof Löding, Philipp Rohde: Solving the Sabotage Game is PSPACE-hard
2003-06	Franz Josef Och: Statistical Machine Translation: From Single-Word Models to Alignment Templates
2003-07	Horst Lichter, Thomas von der Maßen, Alexander Nyßen, Thomas Weiler: Vergleich von Ansätzen zur Feature Modellierung bei der Softwareproduktlinienentwicklung
2003-08	Jürgen Giesl, René Thiemann, Peter Schneider-Kamp, Stephan Falke: Mechanizing Dependency Pairs
2004-01 *	Fachgruppe Informatik: Jahresbericht 2003

2004-02 Benedikt Bollig, Martin Leucker: Message-Passing Automata are expressively equivalent to EMSO logic

2004-03 Delia Kesner, Femke van Raamsdonk, Joe Wells (eds.): HOR 2004 – 2nd International Workshop on Higher-Order Rewriting

2004-04 Slim Abdennadher, Christophe Ringeissen (eds.): RULE 04 – Fifth International Workshop on Rule-Based Programming

2004-05 Herbert Kuchen (ed.): WFLP 04 – 13th International Workshop on Functional and (Constraint) Logic Programming

2004-06 Sergio Antoy, Yoshihito Toyama (eds.): WRS 04 – 4th International Workshop on Reduction Strategies in Rewriting and Programming

2004-07 Michael Codish, Aart Middeldorp (eds.): WST 04 – 7th International Workshop on Termination

2004-08 Klaus Indermark, Thomas Noll: Algebraic Correctness Proofs for Compiling Recursive Function Definitions with Strictness Information

2004-09 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Parameterized Power Domination Complexity

2004-10 Zinaida Benenson, Felix C. Gärtner, Dogan Kesdogan: Secure Multi-Party Computation with Security Modules

2005-01 * Fachgruppe Informatik: Jahresbericht 2004

2005-02 Maximillian Dornseif, Felix C. Gärtner, Thorsten Holz, Martin Mink: An Offensive Approach to Teaching Information Security: "Aachen Summer School Applied IT Security"

2005-03 Jürgen Giesl, René Thiemann, Peter Schneider-Kamp: Proving and Disproving Termination of Higher-Order Functions

2005-04 Daniel Mölle, Stefan Richter, Peter Rossmanith: A Faster Algorithm for the Steiner Tree Problem

2005-05 Fabien Pouget, Thorsten Holz: A Pointillist Approach for Comparing Honeypots

2005-06 Simon Fischer, Berthold Vöcking: Adaptive Routing with Stale Information

2005-07 Felix C. Freiling, Thorsten Holz, Georg Wicherski: Botnet Tracking: Exploring a Root-Cause Methodology to Prevent Distributed Denial-of-Service Attacks

2005-08 Joachim Kneis, Peter Rossmanith: A New Satisfiability Algorithm With Applications To Max-Cut

2005-09 Klaus Kursawe, Felix C. Freiling: Byzantine Fault Tolerance on General Hybrid Adversary Structures

2005-10 Benedikt Bollig: Automata and Logics for Message Sequence Charts

2005-11 Simon Fischer, Berthold Vöcking: A Counterexample to the Fully Mixed Nash Equilibrium Conjecture

2005-12 Neeraj Mittal, Felix Freiling, S. Venkatesan, Lucia Draque Penso: Efficient Reductions for Wait-Free Termination Detection in Faulty Distributed Systems

2005-13 Carole Delporte-Gallet, Hugues Fauconnier, Felix C. Freiling: Revisiting Failure Detection and Consensus in Omission Failure Environments

2005-14 Felix C. Freiling, Sukumar Ghosh: Code Stabilization

2005-15 Uwe Naumann: The Complexity of Derivative Computation

2005-16 Uwe Naumann: Syntax-Directed Derivative Code (Part I: Tangent-Linear Code)

2005-17 Uwe Naumann: Syntax-directed Derivative Code (Part II: Intraprocedural Adjoint Code)

2005-18 Thomas von der Maßen, Klaus Müller, John MacGregor, Eva Geisberger, Jörg Dörr, Frank Houdek, Harbhajan Singh, Holger Wußmann, Hans-Veit Bacher, Barbara Paech: Einsatz von Features im Software-Entwicklungsprozess - Abschlußbericht des GI-Arbeitskreises "Features"

2005-19 Uwe Naumann, Andre Vehreschild: Tangent-Linear Code by Augmented LL-Parsers

2005-20 Felix C. Freiling, Martin Mink: Bericht über den Workshop zur Ausbildung im Bereich IT-Sicherheit Hochschulausbildung, berufliche Weiterbildung, Zertifizierung von Ausbildungsangeboten am 11. und 12. August 2005 in Köln organisiert von RWTH Aachen in Kooperation mit BITKOM, BSI, DLR und Gesellschaft fuer Informatik (GI) e.V.

2005-21 Thomas Noll, Stefan Rieger: Optimization of Straight-Line Code Revisited

2005-22 Felix Freiling, Maurice Herlihy, Lucia Draque Penso: Optimal Randomized Fair Exchange with Secret Shared Coins

2005-23 Heiner Ackermann, Alantha Newman, Heiko Röglin, Berthold Vöcking: Decision Making Based on Approximate and Smoothed Pareto Curves

2005-24 Alexander Becher, Zinaida Benenson, Maximillian Dornseif: Tampering with Motes: Real-World Physical Attacks on Wireless Sensor Networks

2006-03 Uwe Naumann: Intraprocedural Adjoint Code Generated by the Differentiation-Enabled NAGWare Fortran Compiler

2006-04 Ebadollah Varnik, Uwe Naumann, Andrew Lyons: Toward Low Static Memory Jacobian Accumulution

2006-06 Joachim Kneis, Daniel Mölle, Stefan Richter, Peter Rossmanith: Divide-and-Color

* These reports are only available as a printed version.

Please contact `biblio@informatik.rwth-aachen.de` to obtain copies.