Logics and Automata for Totally Ordered Trees

Marco Kuhlmann, Joachim Niehren

Logics and Automata for Totally Ordered Trees

Marco Kuhlmann[1] and Joachim Niehren[2]

[1] Uppsala University, Sweden
[2] INRIA, Lille, France

Abstract A totally ordered tree is a tree equipped with an additional total order on its nodes. It provides a formal model for data that comes with both a hierarchical and a sequential structure; one example for such data are natural language sentences, where a sequential structure is given by word order, and a hierarchical structure is given by grammatical relations between words. In this paper, we study monadic second-order logic (MSO) for *totally ordered terms*. We show that the MSO satisfiability problem of unrestricted structures is undecidable, but give a decision procedure for practically relevant sub-classes, based on tree automata.

1 Introduction

A totally ordered tree is a tree equipped with an additional total order on its nodes. It provides a formal model for data that comes with both a hierarchical and a sequential structure. Depending on the application, the two structural aspects may be more or less dependent on each other: the total order may be obtained by a traversal of the tree, defined by a logic formula from tree relations, or completely independent.

The research reported in this paper is motivated by an application of totally ordered trees in computational linguistics, where they are used as formal models for *dependency structures*. A dependency structure is a representation of the syntactic structure of a natural-language sentence in terms of word-to-word dependencies, such as the dependency between a verb and its direct object. Such dependencies impose a tree-shaped hierarchical structure onto the words, while word order imposes a sequential structure. An example for a dependency structure is given in Fig. 1. Dependency-based representations have a long history in descriptive linguistics. Recently, they have received a lot of interest for many computational tasks, such as information extraction [1], machine translation [2], and, most prominently, data-driven parsing [3].

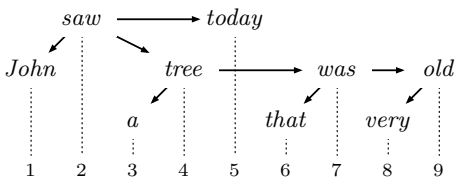


Figure 1: A dependency structure

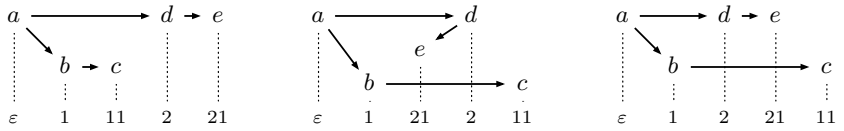
In this paper, we study monadic second-order logic (MSO) as a description language for *totally ordered terms (tots)*. Dependency structures can be understood as special cases of tots, where the sibling order is induced by the total order. Monadic second-order

logic is generally useful in order to express properties of graphs or trees [4,5,6,7]. Here we are particularly interested in lifting Doner, Thatcher, and Wright's theorem on the equivalent expressiveness of MSO and tree automata for finite trees to tots [4,8]. This theorem has first been proved for ground terms, and got extended to various kinds of trees and graphs of bounded tree width [9,10,7].

The new problem that we are faced with, is to deal with the addition of a total order to a finite term structure. The easy cases of this problem are those where the total order is MSO-definable from the term structure: MSO for ground terms is decidable, and thus MSO for ground terms with MSO-defined total orders is decidable as well. In the first part of the paper, we prove that MSO for general tots is *not* decidable. We show this by a reduction of the MSO satisfiability problem for *grids*; this problem is well-known to be undecidable (while first-order logic of grids can encoded into Presburger arithmetics). In the second part of the paper, we restrict the classes of models of MSO formulas to sets of tots with *bounded gap-degree* [11]. This means that the descendant sets of nodes are segmented into a bounded number of intervals in the total order. Our main contribution is the result that MSO satisfiablity of sets of tots with bounded gap-degree is decidable. In order to establish this, we introduce an algebraization for these sets. This leads us to a notion of tree automaton for gap-bounded sets of tots, which we show to have the same expressiveness as MSO.

Related work. Our algebraic perspective on automata goes back to the work of Mezei and Wright from the 1960s [16]. It was generalized by Courcelle [17], and applied to many kinds of trees, including unranked sibling-ordered trees as they appear in the context of XML [18]. Courcelle has proposed two different algebraizations for graphs [7] for which MSO can be reduced to finite automata. Graphs of bounded tree or clique width belong to these algebras, so that MSO satisfiability is decidable for them.

Dependency structures can be used to quantify the generative capacity of many grammar formalisms for natural language [12]: they are more informative than strings, but less formalism-specific than parse trees. The formal properties of dependency structures and sets of such structures have been studied only recently [12]. Automata for these structures define a notion of regularity. For regular sets of dependency structures, there is a direct relation between the gap-degree measure and string-generative capacity. In particular, regular sets of dependency structures with gap-degree 0 correspond exactly to the context-free languages, and regular sets of structures with gap-degree at most 1 and an additional property called *well-nestedness* give the string languages generated by Lexicalized Tree Adjoining Grammars (TAGs). More generally, every string language obtained from a regular set of dependency structures with bounded gap-degree is semilinear and can be recognized in polynomial time. Without gap restrictions, parsing quickly becomes NP-hard [13]. Recent work has shown that most dependency structures required for the analysis of natural language have a small gap-degree [15]. However, not all natural languages can be described by sets of dependency structures with bounded gap-degree; counterexamples have been given for Czech [14].



(a) preorder (b) swapping c and e (c) swapping c and d, e

Figure 2: Three total orders for the tree $a(b(c), d(e))$

2 Totally Ordered Terms

We start by introducing totally ordered terms, and MSO for their description. We make use of the following auxiliary notions and notations: The set of positive natural numbers is denoted by $\mathbb{N}$. Given some $n \in \mathbb{N}$, we write $[n]$ for the set $\{1, \ldots, n\}$. A *signature* Σ is a non-empty, finite set of function symbols σ, each equipped with a fixed, non-negative arity, denoted by $\mathrm{ar}(\sigma)$. The *set of (ground) terms over* Σ is the smallest set T_Σ such that if σ is a symbol of arity m and for each $i \in [m]$, $t_i \in T_\Sigma$, then $\sigma(t_1, \ldots t_m) \in T_\Sigma$. The set of *nodes* of a term $t \in T_\Sigma$ is a set of addresses in $\mathbb{N}^*$: the root node of t is addressed by the empty word ε, and the ith child of the node π is addressed by the extended address πi:

$$nod(\sigma(t_1, \ldots, t_m)) \;=\; \{\varepsilon\} \cup \{\, i\pi \in \mathbb{N}^* \mid i \in [m],\ \pi \in nod(t_i) \,\}\,.$$

A *linearization* of a finite set A is a word in A^* in which each element of A occurs exactly once. We note that the set of linearizations of A is isomorphic to the set of total orders on A in an obvious way.

Definition 1. *A* totally ordered term (tot) over Σ *is a pair* $\tau = \langle t, w \rangle$ *where* $t \in T_\Sigma$ *is a term, and* w *is a linearization of* $nod(t)$.

The set of all tots over Σ is denoted by TOT_Σ. Three examples for tots are visualized in Fig. 2; each of them provides a different linearization for the same term $a(b(c), d(e))$. Solid edges depict the term structure, dotted lines project nodes to their position in the linearization. In Fig. 2a, the nodes of $a(b(c), d(e))$ are ordered by the preorder traversal of the underlying term structure. The two examples in Fig. 2b and 2c are derived by swapping c with e and d, e, respectively.

In the following, we often identify a term $t \in T_\Sigma$ with a relational structure $\langle nod(t)\,; (:\sigma)_{\sigma\in\Sigma}\rangle$: for each m-ary symbol $\sigma \in \Sigma$, this structure provides a labelling relation $:\sigma$ with arity $m+1$. Given a node $\pi \in nod(t)$, labelling $\pi : \sigma(\pi 1, \ldots, \pi n)$ holds if and only if π is labelled by σ in t. A tot $\tau = (t, w)$ can be viewed as a relational structure that extends the structure corresponding to t by the total order $\preceq$ that is represented by the linearization w.

Monadic second-order logic (MSO) for tots is defined as usual. Let *Vars* be a set that contains infinitely many node variables $x, y, z \in \mathit{Vars}$, and infinitely many set variables $X, Y, Z \in \mathit{Vars}$. The formulas of MSO for tots are the following, where $\sigma \in \Sigma$ is a function symbol of arity m:

$$\phi, \phi' \;::=\; x : \sigma(x_1, \ldots, x_m) \mid x \preceq y \mid x \in X \mid \phi \wedge \phi' \mid \neg\phi \mid \exists x.\, \phi \mid \exists X.\, \phi$$

These formulas are interpreted in the usual Tarskian style in the relational structures corresponding to tots. Given a tot $\tau = (t, w)$ and a variable assignment $\alpha\colon \mathit{Vars} \to \mathit{nod}(t)$, we write $\tau, \alpha \models \phi$ if and only if the formula ϕ evaluates to true in τ under the assignment α, and $\tau \models \phi$ if $\tau, \alpha \models \phi$ holds for any assignment α.

MSO formulas with n free node variables over the signature Σ define n-ary relations for all tots over Σ. Most basically, we can define node equality ($=$), equality-or-precedence ($\preceq$), and the child relation $\lhd$:

$$x \lhd y \;=_{\text{def}}\; \bigvee_{\sigma\in\Sigma} \bigvee_{1\le i\le m=\mathrm{ar}(\sigma)} \exists\, x_1, \ldots, x_m.\, y = x_i \wedge x : \sigma(x_1, \ldots, x_m)) \,.$$

The dominance relation $\lhd^*$ of a term is the reflexive, transitive closure of $\lhd$, and thus definable in MSO.[3] This is in contrast to first-order logic, where the dominance relation is usually added to the relational structure. Let $C \subseteq \mathit{TOT}_\Sigma$ be a set of tots over Σ. The *MSO-satisfiability problem of* C is the problem to decide whether $\exists\, \tau \in C.\, \tau \models \phi$, where ϕ is some closed MSO formula.

3 Undecidability of MSO Satisfiability

We now present our first technical result, which is valid for all signatures Σ with at least one binary function symbol and one constant.

Theorem 1. *MSO satisfiability for the class of all tots over Σ is undecidable.*

For the proof, we make use of a simple tool to obtain undecidability results for graph-like structures. Let $m, n \in \mathbb{N}$. The *grid of dimensions* $m \times n$ is the structure

$$\begin{aligned} G_{m,n} &= \langle [m] \times [n] \,;\, \mathit{nextEast}, \mathit{nextSouth} \rangle \,, \quad \text{where} \\ \mathit{nextEast} &= \{\, \langle \langle i, j\rangle, \langle i', j'\rangle\rangle \mid i' = i+1,\, j' = j \,\} \quad \text{and} \\ \mathit{nextSouth} &= \{\, \langle \langle i, j\rangle, \langle i', j'\rangle\rangle \mid i' = i,\, j' = j+1 \,\} \,. \end{aligned}$$

We can view a grid as an $(m \times n)$-matrix in which we can navigate along columns (using the relation *nextEast*) and rows (using *nextSouth*). The *square grid of size* m is the grid of dimensions $m \times m$. The following result is standard:

Fact [7]. *The MSO satisfiability problem of every class of graphs that contains infinitely many square grids is undecidable.*

To make use of this result to prove Theorem 1, we show how to encode grids as tots (Proposition 1), and that every closed MSO formula interpreted on grids can be translated into a closed MSO formula interpreted on tots that has the same models modulo encoding (Proposition 2).

Without loss of generality, we use the signature $\Sigma = \{\mathsf{cons}, \circ, \mathsf{nil}\}$ with $\mathrm{ar}(\mathsf{cons}) = 2$, $\mathrm{ar}(\circ) = 1$, and $\mathrm{ar}(\mathsf{nil}) = 0$, and employ the following naming scheme for nodes: for $0 \le i$ and $1 \le j$,

$$\pi_{i,j} \;=\; \textbf{if } i = 0 \textbf{ then } 1^{j-1} \textbf{ else } 1^{j-1} 2 1^{i-1} \,.$$

[3] To see this, note that x dominates all elements of the set Y, where Y is the least set satisfying the equation $Y = \{x\} \cup \{\, z \mid \exists\, y.\, y \in Y \to y \lhd z \,\}$.

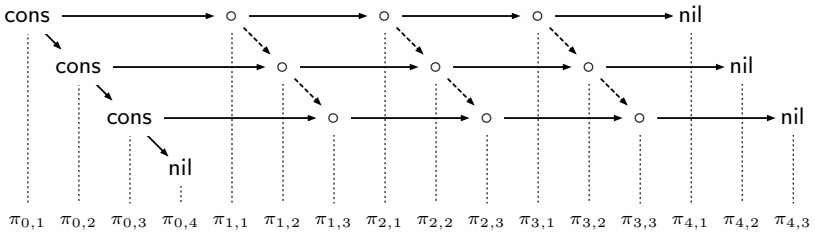


Figure 3: The encoding of the square grid $G_{3,3}$. The solid lines represent the term structure; the dashed lines visualize the relation *nextSouth*.

The encoding of the general grid $G_{m,n}$ is the tot $[\![G_{m,n}]\!] = \langle t_{m,n}, w_{m,n} \rangle$, where

$$t_{m,n} = \textbf{if } n = 1 \textbf{ then } \mathsf{cons}(\mathsf{nil}, \circ^m(\mathsf{nil})) \textbf{ else } \mathsf{cons}(t_{m,n-1}, \circ^m(\mathsf{nil}))$$
$$w_{m,n} = \pi_{0,1} \cdots \pi_{0,n} \cdot \pi_{0,n+1} \cdot \pi_{1,1} \cdots \pi_{1,n} \cdots \pi_{m+1,1} \cdots \pi_{m+1,n}$$

For illustration, Fig. 3 shows the tot that encodes the square grid $G_{3,3}$. We will use the nodes that are labelled with the constructor $\circ$ (i.e., the nodes of the form $\pi_{i,j}$, $i \in [m]$, $j \in [n]$) as representatives for the entries of the grid proper.

The following MSO-defined sets are of general interest for the axiomatization of grid encodings. We use definitions by formulas with free set variables that may be parametrized by a free node variable x, so that the corresponding sets are parametrized by the node value of x.

$$\begin{aligned}
\mathit{FirstChild}(x) &= \{\, y \mid x : \circ(y) \,\} \cup \{\, y \mid \exists\, z.\, x : \mathsf{cons}(y, z) \,\} \\
\mathit{SecondChild}(x) &= \{\, y \mid \exists\, z.\, x : \mathsf{cons}(z, y) \,\}, \quad \mathit{Root} = \{\, x \mid \mathit{Father}(x) = \emptyset \,\} \\
\mathit{Father}(x) &= \{\, y \mid x \in \mathit{FirstChild}(y) \,\} \cup \{\, y \mid x \in \mathit{SecondChild}(y) \,\} \\
\mathit{Succ}(x) &= \{\, y \mid x \prec y,\ \neg\exists\, z.\, x \preceq z \preceq y \,\}, \quad \mathit{Cons} = \{\, x \mid x : \mathsf{cons}(_, _) \,\}, \\
\mathit{Grid} &= \{\, x \mid x : \circ(_) \,\}, \quad \mathit{Nil} = \{\, x \mid x : \mathsf{nil} \,\}.
\end{aligned}$$

We note that, since $\preceq$ is a total order, *Succ* is an injective partial function. The cons-labelled nodes provide the backbone of the tot; they 'glue' the lines of the grid together. The north of the grid is its first line; the south is its last line.

$$\begin{aligned}
\mathit{Backbone} &= (\mathit{Root} \cup \mathit{FirstChild}(\mathit{Backbone})) - \mathit{Nil} \\
\mathit{LastBackbone} &= \{\, x \in \mathit{Backbone} \mid \mathit{FirstChild}(x) \subseteq \mathit{Nil} \,\} \\
\mathit{North} &= (\mathit{SecondChild}(\mathit{Root}) \cup \mathit{FirstChild}(\mathit{North})) \cap \mathit{Grid} \\
\mathit{South} &= (\mathit{SecondChild}(\mathit{LastBackbone}) \cup \mathit{FirstChild}(\mathit{South})) \cap \mathit{Grid}
\end{aligned}$$

Recursive definitions are interpreted with respect to least fixed points, which is expressible in MSO, as is the reflexive-transitive closure of a relational image $S(x)$, denoted by $S(x)^*$. We next define sets of nodes by MSO formulas that permit us to navigate through the grid:

$$\begin{aligned}
\mathit{NextEast}(x) &= \{\, y \in \mathit{Grid} \cap \mathit{FirstChild}(x) \mid x \in \mathit{Grid} \,\} \\
\mathit{NextSouth}(x) &= \{\, y \in \mathit{Grid} \cap \mathit{Succ}(x) \mid x \in \mathit{Grid} - \mathit{South} \,\}
\end{aligned}$$

The *anchor* of a node x is the (uniquely determined) closest node on the backbone that dominates x. Via the anchors, we can navigate along the rows in a grid.

$$\begin{aligned} \mathit{Anchor}(x) &= \{\, y \in \mathit{Backbone} \mid x \in \mathit{FirstChild}^*(\mathit{SecondChild}(y)) \,\} \\ \mathit{NextRow}(x) &= \{\, y \in \mathit{Grid} \mid \mathit{FirstChild}(\mathit{Anchor}(x)) = \mathit{Anchor}(y) \,\} \\ \mathit{PreviousRow}(x) &= \{\, y \in \mathit{Grid} \mid \mathit{FirstChild}(\mathit{Anchor}(y)) = \mathit{Anchor}(x) \,\} \end{aligned}$$

We can now axiomatize the class of grid encodings in MSO:

(A1) The backbone consists of the nodes labelled with cons.

$\mathit{Backbone} = \mathit{Cons}$

(A2) For every node in the grid that does not lie in the south, taking the next step in the total order takes us to the next row.

$\forall\, x \in \mathit{Grid} - \mathit{South}.\, \exists\, y \in \mathit{NextRow}(x).\, \mathit{Succ}(x) = \{y\}$

(A3) Dually, every node in the grid that does not lie in the north can be reached from a node in the previous row.

$\forall\, x \in \mathit{Grid} - \mathit{North}.\, \exists\, y \in \mathit{PreviousRow}(x).\, \mathit{Succ}(y) = \{x\}$

(A4) The root node occupies the first position in the total order.

$\neg \exists\, x.\, \mathit{Succ}(x) = \mathit{Root}$

(A5) Every step to the first child of a cons node is a step in the total order.

$\forall\, x \in \mathit{Cons}.\, \mathit{Succ}(x) = \mathit{FirstChild}(x)$

(A6) The nil node terminating the backbone immediately precedes the north-western corner of the grid.

$\mathit{Succ}(\mathit{FirstChild}(\mathit{LastBackbone})) = \mathit{SecondChild}(\mathit{Root})$

(A7) The total order propagates along the rows of the grid.

$\forall\, x, y.\, \mathit{Succ}(\mathit{Father}(x)) = \mathit{Father}(y) \implies \mathit{Succ}(x) = \{y\}$

(A8) When being in the south, the next step with the total order leads to the first element of the next column eastwards.

$\forall\, x \in \mathit{North}.\, \mathit{Succ}(\mathit{NextSouth}^*(x) \cap \mathit{South}) = \mathit{FirstChild}(x)$

Proposition 1. *Every encoded grid satisfies the axioms A1–A8. Conversely, every tot that satisfies these axioms is a grid encoding.*

Proof. To verify that encoded grids satisfy the axioms A1–A8 is straightforward; here we show the converse. Let $\tau = (t, w)$ be a tot such that $\tau \models \mathrm{A1} \wedge \cdots \wedge \mathrm{A8}$. We show that there exists a grid $G_{m,n}$ such that $\tau = [\![G_{m,n}]\!]$. Axiom A_1 asserts that $\mathit{Backbone} = \mathit{Cons}$, which implies that t has the following form, where $t_i = \circ^{m_i}(\mathsf{nil})$ for some $m_i \geq 0$: $t = \mathsf{cons}(t_1, \mathsf{cons}(t_2, \ldots \mathsf{cons}(t_n, \mathsf{nil}) \ldots))$.

We have to show that all terms t_i have the same length in order to prove that $t = t_{m,n}$. This property is less obvious since it depends on the total order among the nodes. For $n = 0$, it follows that $t = \mathsf{nil}$, and there is nothing to show, so we can assume $n \geq 1$. The form of t implies that $\mathit{Backbone} = \{\, \pi_{0,j} \mid j \in [n] \,\}$,

$$\mathit{North} = \{\, \pi_{i,1} \mid i \in [m_1] \,\}, \qquad \text{and} \qquad \mathit{South} = \{\, \pi_{i,n} \mid i \in [m_n] \,\}.$$

Lemma 1. $m_1 = \cdots = m_n$

We first show that $|t_1| \leq |t_j|$ for all $j \in [n]$ by induction on n. The case $n = 1$ is trivial. For the case $n > 1$, it suffices to show that $|t_{n-1}| \leq |t_n|$. This follows from Axiom A2, which asserts that for each node π in t_{n-1}, there exists a node π' in t_n such that $\mathit{Succ}(\pi) = \{\pi'\}$, and from the observation that Succ is total for all nodes of t_1 by Axiom A4, functional and injective. Dually, using Axiom A3, we can show that $|t_j| \leq |t_1|$ for all $j \in [n]$. Consequently, $|t_1| = \cdots = |t_n|$.

It remains to show that w is indeed the total order that is imposed on the encoded grid. The following series of Lemmas together with Axiom A4 establishes that the total order on t is uniquely determined by the axioms. Since the order in the grid translation satisfies these axioms, it must be equal to this order.

Lemma 2. $\forall\, j \in [n].\, \mathit{Succ}(\pi_{0,j}) = \{\pi_{0,j+1}\}$ *and* $\mathit{Succ}(\pi_{0,n+1}) = \{\pi_{1,1}\}$

Since for each $j \in [n]$, the node $\pi_{0,j}$ is a cons node, the first half of the claim is stated in Axiom A5. The successor of $\pi_{0,n+1}$ follows from Axiom A6 in combination with the observation that $\mathit{SecondChild}(\mathit{Root}) \neq \{\pi_{1,1}\}$; this is so since $\mathit{Root} \subseteq \mathit{Cons}$ by Axiom A6, and $n \geq 1$.

Lemma 3. $\forall\, i \in [m].\, \forall\, j \in [n-1].\, \mathit{Succ}(\pi_{i,j}) = \{\pi_{i,j+1}\}$

We show that $\mathit{Succ}(\mathit{Father}(\pi_{i,j})) = \mathit{Father}(\pi_{i,j+1})$, and from this deduce the claim by Axiom A7. The proof proceeds by induction on i. In the case that $i = 1$, using the definition of Father and Lemma 2, we see that

$$\mathit{Succ}(\mathit{Father}(\pi_{1,j})) = \mathit{Succ}(\pi_{0,j}) = \{\pi_{0,j+1}\} = \mathit{Father}(\pi_{1,j+1}).$$

For $i > 1$, we may assume that $\mathit{Succ}(\pi_{i-1,j}) = \{\pi_{i-1,j+1}\}$. Thus,

$$\mathit{Succ}(\mathit{Father}(\pi_{i,j})) = \mathit{Succ}(\pi_{i-1,j}) = \{\pi_{i-1,j+1}\} = \mathit{Father}(\pi_{i,j+1}).$$

Lemma 4. $\forall\, i \in [m].\, \mathit{Succ}(\pi_{i,n}) = \{\pi_{i+1,1}\}$

We start by proving the following auxiliary claim by induction on j:

$$\forall\, j \in [n].\, \forall\, i \in [m].\, \mathit{NextSouth}^*(\pi_{i,j}) \cap \mathit{South} = \{\pi_{i,n}\}$$

For $j = n$, this is obvious, given that $\pi_{i,n} \in \mathit{South}$. For $j < n$, we may assume that $\mathit{NextSouth}^*(\pi_{i,j+1}) \cap \mathit{South} = \{\pi_{i,n}\}$. Moreover, $\pi_{i,j} \in \mathit{Grid} - \mathit{South}$, which, by definition of $\mathit{NextSouth}$, implies that $\mathit{NextSouth}(\pi_{i,j}) = \mathit{Succ}(\pi_{i,j})$. Thus,

$$\begin{aligned} &\mathit{NextSouth}^*(\pi_{i,j}) \cap \mathit{South} = \mathit{NextSouth}^*(\mathit{NextSouth}(\pi_{i,j})) \cap \mathit{South} \\ &= \mathit{NextSouth}^*(\mathit{Succ}(\pi_{i,j})) \cap \mathit{South} \overset{\text{Lemma 3}}{=} \mathit{NextSouth}^*(\pi_{i+1,j}) \cap \mathit{South} = \{\pi_{i,n}\}. \end{aligned}$$

$$
\begin{aligned}
[\![\mathit{nextEast}(x,y)]\!]_0 &= y \in \mathit{NextEast}(x) & [\![\forall\, x.\, \psi]\!]_0 &= \forall\, x \in \mathit{Grid}.\, [\![\psi]\!]_0 \\
[\![\mathit{nextSouth}(x,y)]\!]_0 &= y \in \mathit{NextSouth}(x) & [\![\forall\, X.\, \psi]\!]_0 &= \forall\, X \subseteq \mathit{Grid}.\, [\![\psi]\!]_0 \\
[\![x \in X]\!]_0 &= x \in X \cap \mathit{Grid} & [\![\psi \wedge \psi']\!]_0 &= [\![\psi]\!]_0 \wedge [\![\psi']\!]_0 \\
[\![\neg\psi]\!]_0 &= \neg [\![\psi]\!]_0
\end{aligned}
$$

Figure 4: Encoding MSO of grids into MSO of tots: the compositional part.

Instantiating Axiom A8 with $\pi_{i,1} \in \mathit{North}$, we see that

$$\mathit{Succ}(\mathit{NextSouth}^*(\pi_{i,1}) \cap \mathit{South}) \;=\; \mathit{FirstChild}(\pi_{i,1}) \;=\; \{\pi_{i+1,1}\}\,.$$

Since $\mathit{NextSouth}^*(\pi_{i,1}) \cap \mathit{South} = \{\pi_{i,n}\}$ by the auxiliary claim above, this implies that $\mathit{Succ}(\pi_{i,n}) = \{\pi_{i+1,1}\}$.

Together with Axiom A4, the preceding Lemmas establish that $\mathit{Succ}(\pi)$ is uniquely determined for all $\pi \neq \pi_{m,n}$. This ends the proof of Proposition 1. □.

Proposition 1 states that we can define the encodings of grids using MSO for tots. We now show how to express MSO formulas for grids using MSO formulas for tots. The translation of a closed MSO formula ψ for grids is as follows:

$$[\![\psi]\!] =_{\text{def}} \exists \mathit{Grid} \exists \mathit{NextSouth} \exists \mathit{NextEast} \exists \ldots.\ (\text{Defs} \wedge \text{A1} \wedge \cdots \wedge \text{A8} \wedge [\![\psi]\!]_0)$$

where $[\![\psi]\!]_0$ is given in Fig. 4, and Defs contains the above definitions for all occuring set variables, all of which are existentially quantified in the outermost quantifier prefix.

Proposition 2. *For every closed formula ψ from the set of all MSO formulas over grids and every grid G, $\psi \models G$ implies that $[\![\psi]\!] \models [\![G]\!]$.*

Proving this proposition from Proposition 1 is routine, since it is largely independent of the particularities of the encoding. Note however that the proposition does not hold for formulas with free variables. For these, quantification would need to be restricted to nodes in *Grid*, and exclude nodes in $\mathit{Cons} \cup \mathit{Nil}$.

4 Bounded Gap-Degree

Given the undecidability of the MSO satisfiability problem for the class of unrestricted tots, we are interested in restricted classes for which decidability can be obtained. A family of such classes that is relevant for applications in computational linguistics is obtained from the *gap-degree measure* [11,12].

Let τ be a tot, and let π, π_1, π_2 be nodes of τ. The set of *descendants of* π, denoted by $\mathit{desc}(\pi)$, is the set of all nodes $\pi' \in \mathit{nod}(t)$ such that $\pi \lhd^* \pi'$. The *interval with endpoints* π_1 *and* π_2, denoted by $[\pi_1, \pi_2]$, is the set of all nodes $\pi' \in \mathit{nod}(\tau)$ such that $\pi_1 \preceq \pi' \preceq \pi_2$. Note that for terms ordered by a pre-order

traversal, each descendant set forms an interval. In the general case, though, a descendant set $desc(\pi)$ may be partitioned into a sequence of (maximal) intervals, which we call the *segments* of π. We define MSO formulas $y_1 \equiv_x y_2$ stating that y_1 and y_2 belong to the same segment of x:

$$y_1 \equiv_x y_2 =_{\text{def}} \forall z.\, (y_1 \preceq z \preceq y_2 \vee y_2 \preceq z \preceq y_1) \rightarrow x \lhd^* z$$

This formula defines an equivalence relation $\equiv_\pi$ on the descendant set $desc(\pi)$ with the property that each equivalence class forms a (maximal) interval. We call such a relation a *segmentation*.

Definition 2. *Let τ be a tot, and let π be a node of τ. The* gap-degree of π, *$deg(\pi)$, is defined as the index of the relation $\equiv_\pi$, minus one. The* gap-degree of τ, *$deg(\tau)$, is the maximum among the gap-degrees of its nodes. A set L of tots is* gap-bounded, *if there is a constant g_L such that $deg(\tau) \leq g_L$, for every $\tau \in L$.*

The tot in Fig. 2a has gap-degree 0, while those in Figs. 2b and 2c have gap-degree 1. In Figs. 2b and Figs. 2c, the node b has two segments ($\{b\}$ and $\{c\}$); in Fig. 2a, it has only one ($\{b, c\}$).

Lemma 5. *The gap-degree of the tot $[\![G_{n,n}]\!]$ is $n + 1$.*

This Lemma shows that our undecidability proof fails when we restrict the models of our MSO formulas to classes of tots that are gap-bounded. Even better, this restriction implies decidability:

Theorem 2. *The MSO satisfiability problem of every gap-bounded class of tots is decidable.*

The remainder of this paper is concerned with the proof of this result. To do so, we establish a link between MSO for gap-bounded classes of tots on the one hand, and an algebraic notion of automata on the other.

5 Segmented Tots and Subtots

We develop a notion of 'substructure' for tots that will serve as intermediate results when constructing tots algebraically. They may be little more general than tots, in that their linearisations may be segmented, so we call them segmented tots.

For illustration, let us reconsider the tot in Fig. 2c. Its first subterm is $b(c)$. The linearization of the nodes of this subterm is segmented into two parts, while leaving a gap between the b-node and the c node, into which external nodes may be plugged, such as the nodes of $d(e)$ when reconstructing the original tot.

A *k-segmented linearization* over a finite set A is a k-tuple $\langle w_1, \ldots, w_k \rangle$ of non-empty words over A such that $w_1 \cdots w_k$ forms a linearization. Every k-segmented linearization w defines a total ordering $\preceq$ on A, which satisfies $a_1 \preceq a_2$ if a_1 occurs left of a_2 in w. In addition, it defines an equivalence relation $\equiv$ on A, such

that $a_1 \equiv a_2$ if they occur in the same segment of w. The segments of w are the equivalence classes of $\equiv$, so that the index of $\equiv$ is k. The equivalence classes form intervals with respect to the total ordering, i.e.:

$$a_1 \equiv a_2 \;\Rightarrow\; \forall\, a.\, (a_1 \preceq a \preceq a_2 \vee a_2 \preceq a \preceq a_1) \rightarrow a_1 \equiv a \equiv a_2$$

Conversely, every pair of total ordering $\preceq$ on A and an equivalence relation $\equiv$ on A of index k that satisfy the above condition define an k-segmented linearization on A.

Definition 3. *A* k-segmented tot *over a signature* Σ *is a pair* $\langle t, w\rangle$ *where* t *is a term over* Σ *and* w *a* k*-segmented linearization of* $nod(t)$*.*

The relational structure corresponding to a segmented tot (t, w) is the structure $\langle nod(t)\,; \preceq, \equiv, (:\sigma)_{\sigma\in\Sigma}\rangle$; it provides the total ordering $\preceq$ and the equivalence relation $\equiv$ defined by w and the labeling predicates. The MSO of segmented tots is the MSO with symbols for all these relations and interpreted over these relational structures.

Substructures of segmented tots can now be defined by extending the notion of subterms. Let $\tau = \langle t, w\rangle$ be a segmented tot and node $\pi \in nod(t)$. The set $desc(\pi)$ is segmented by the relation $\equiv^{\pi}$, which is defined by an MSO formula with three free variables:

$$y_1 \equiv^{x} y_2 \;=_{\text{def}}\; y_1 \equiv y_2 \wedge \forall\, z.\, (y_1 \preceq z \preceq y_2 \vee y_2 \preceq z \preceq y_1) \rightarrow x \lhd^{*} z\,.$$

This means that segments of $desc(\pi)$ are either separated by nodes of τ external to $desc(\pi)$ or by the segmentation of τ.

Let $t|_\pi$ the subterm of t at node π, i.e., $t|_\varepsilon = t$ and $\sigma(t_1, \ldots, t_m)|_{i\pi} = t_i|_\pi$, where $i \in [m] = \mathrm{ar}(\sigma)$. Recall that $nod(t|_\pi) = \{\, \pi' \mid \pi\pi' \in nod(t)\,\}$. The subtot of τ at π is given by the subterm $t_{|\pi}$ and the segmented linearization of its nodes induced by $\equiv^{\pi}$ and $\preceq$. The of gap-degree of a segmented tot τ is one less then the maximal number of segments of its subtots $\tau_{|\pi}$.

6 Segmented Words

We introduce segmented words and define operators for them that will be useful for the algebraization of tots. Let A be a set and k a natural numbers.

A *k-segmented word* over A is a k-tuple of non-empty words over A. The *positions* of a k-segmented word $\psi = \langle a_1^1 \cdots a_1^{n_1}, \ldots, a_k^1 \cdots a_k^{n_k}\rangle$ are pairs of natural numbers: $pos(\psi) = \{\, \langle i, j\rangle \mid i \in [k],\ j \in [n_i]\,\}$. The set of positions is totally ordered by the lexicographic order, i.e. $\langle i', j'\rangle < \langle i, j\rangle$ iff $i' < i$ or $i' = i \wedge j' < j$. It is partitioned into k classes by the equivalence $\langle i', j'\rangle \equiv \langle i, j\rangle$ iff $i' = i$. For every $a \in A$, the relation $Q_a \subseteq pos(\psi)$ contains all positions $\langle i, j\rangle$ where a occurs, i.e., $a_i^j = a$.

We now define operators for k-segmented words, where we assume $k \in [l]$ for a fixed natural number l. Let $W(k)$ be the set of k-segmented words over A. We define a multi-sorted algebra $\mathscr{S}$ with domains $W(1), \ldots, W(l)$; the functions

of this algebra are defined by segmented words. Each function first shuffles the segments of its arguments into a new segmented word, and then fuses adjacent segments to arrive at a segmented word with at most l segments. To make this formal, we need three auxiliary notions: The *multiplicity* of a letter $a \in A$ in a segmented word ψ over A is the number of positions $\langle i, j\rangle \in pos(\psi)$ such that $\langle i, j\rangle \in Q_a$. A *multiset* over a finite set B is a function $\mu\colon B \to \mathbb{N}$. A *$k$-segmented linearization of of a multiset* μ is a k-segmented word ψ over B such that for all $b \in B$, the multiplicity of b in ψ is $\mu(b)$.

Let $k \in [l]$ and $m \in \mathbb{N}$, and $k_1, \ldots, k_m \in [l]$. Every k-segmented linearization $\psi \in ([m]^+)^k$ of the multiset $\{\, i \mapsto k_i \mid i \in [m] \,\}$ defines an operator $\psi^{\mathscr{S}}$ of type:

$$\psi^{\mathscr{S}}\colon W(k_1) \times \cdots \times W(k_m) \to W(k)\,.$$

To give a definition of this function, let the *occurrence number* $\mathrm{occ}(i, j) \in \mathbb{N}$ of a position $\langle i, j\rangle \in pos(\psi)$ with $\langle i, j\rangle \in Q_a$ be the number of positions $\langle i', j'\rangle \in pos(\psi)$ such that $\langle i', j'\rangle \leq \langle i, j\rangle$ and $\langle i', j'\rangle \in Q_a$. If $\psi = \langle a_1^1 \cdots a_1^{n_1}, \ldots, a_k^1 \cdots a_k^{n_k}\rangle$, then the application of the function $\psi^{\mathscr{S}}$ to segmented words $S_i \in W(k_i)$, $i \in [m]$, is defined as

$$\psi^{\mathscr{S}}(\langle S_1^1, \ldots, S_1^{k_1}\rangle, \ldots, \langle S_m^1, \ldots, S_m^{k_n}\rangle) \;=\; \langle T_1^1 \cdots T_1^{n_1}, \ldots, T_k^1 \cdots T_k^{n_k}\rangle\,,$$

where $T_i^j = S_{a_i^j}^{\mathrm{occ}(i,j)}$. Note that, for each $i \in [k]$, $T_1^1 \cdots T_1^{n_i} \in A^+$ is a concatenation of n_i words, while $\langle S_1^1, \ldots, S_1^{k_i}\rangle \in W(k_i)$ is a k_i-tuple. The number of functions of $\mathscr{S}$ is infinite as long as we do not bound the maximal arity $m \in \mathbb{N}$. Furthermore, $\mathscr{S}$ does not contain any constant, so it is useless standalone.

7 Algebra of Segmented Tots

For the algebraization of tots, we fix a signature Σ of function symbols and a gap bound $l \in \mathbb{N}$. For all $k \in \mathbb{N}$ let $T(k) \subseteq TOT_\Sigma$ be the set of tots over Σ with k segments. We define a multi-sorted algebra $\mathscr{T}$ with domains $T(1), \ldots, T(l)$.

For every tot $\tau = \langle t, w\rangle$ in $T(k)$, let $term(\tau) = t$ and $seg(\tau) = w$. The algebra $\mathscr{T}$ provides operations of type $\langle \sigma, \psi\rangle^{\mathscr{T}} : T(k_1) \times \cdots \times T(k_m) \to T(k)$ for all $\sigma \in \Sigma$ of arity m, k-segmentation ψ of the multiset $\{0 \mapsto 1\} \cup \{\, i \mapsto k_i \mid i \in [m] \,\}$, and $k, k_1, \ldots, k_m \in [l]$. Specifically, we define the value $\langle \sigma, \psi\rangle^{\mathscr{T}}(\tau_1, \ldots, \tau_n) = \tau$ as follows:

$$\begin{aligned} term(\tau) \;&=\; \sigma(term(\tau_1), \ldots term(\tau_n)) \\ seg(\tau) \;&=\; \psi^{\mathscr{S}}((\varepsilon), pfx_1(seg(\tau_1)), \ldots, pfx_n(seg(\tau_n)))\,, \end{aligned}$$

where pfx_i is the function that takes a sequence of nodes and prefixes every node in this sequence by the address i. This algebra has a finite number of functions, whose maximal arity is bounded by the maximal arity in Σ. The constants of this algebra are of the form $\langle \sigma, \langle 1\rangle\rangle$, where σ is a constant in Σ.

Proposition 3. *Let $k \in [l]$. For every tot $\tau \in T_k$, there exist a symbol $\sigma \in \Sigma$ $ar(\sigma) = m$, natural numbers $k_1, \ldots, k_m \in [l]$, a k-segmentation ψ of the multiset $\{0 \mapsto 1\} \cup \{\, i \mapsto k_i \mid i \in [m] \,\}$, and tots $\tau_1 \in T_{k_1}, \ldots, \tau_m \in T_{k_m}$ such that $\tau = \langle \sigma, \psi \rangle^{\mathscr{T}}(\tau_1, \ldots, \tau_m)$.*

This means that every k-segmented tot can be constructed from the operators of the algebra $\mathscr{T}$ if $k \in [l]$. For instance, the tot in Fig. 2b is equal to:

$$\langle a, \langle 0121 \rangle \rangle^{\mathscr{T}}(\langle b, \langle 0, 1 \rangle \rangle^{\mathscr{T}}(\langle c, \langle 0 \rangle \rangle^{\mathscr{T}}), \langle d, \langle 10 \rangle \rangle^{\mathscr{T}}(\langle e, \langle 0 \rangle \rangle^{\mathscr{T}})) \,.$$

The substructure at b, i.e., the first child of the root, has gap-degree 1. This is reflected by the segmented word $\langle 0, 1 \rangle$ whose comma matches the gap.

Proof. Let $\tau = (t, w)$, where $t = \sigma(t_1, \ldots, t_n)$ for some $\sigma \in \Sigma$, and $t_1, \ldots, t_n \in T_\Sigma$. For all $i \in [k]$, let k_i be the number of segments of $\tau|_i$, so that

$$pfx_i(seg(\tau|_i)) \;=\; \langle S_i^1, \ldots, S_i^{k_i} \rangle$$

for some $k_i \in [l]$ and $S_i^1, \ldots, S_i^{k_i} \in nod(t)^+$. Furthermore, let $k_0 = 1$ and $S_0^1 = \varepsilon$. The set $nod(t)$ is partitioned into segments S_i^j, where $0 \leq i \leq n$ and $j \in [k_i]$. The segmented linearization w of τ must be of the following form, where $T_i^j = S_{a_i^j}^{\mathrm{occ}(i,j)}$:

$$seg(\tau) \;=\; \langle T_1^1 \cdots T_1^{n_1}, \ldots, T_k^1 \cdots T_k^{n_k} \rangle \,.$$

This holds, since each segment of $seg(\tau)$ must be a concatenation of words of S_i^js due to convexity, and since the segments of the same substructure must appear in their original order. Let

$$\psi \;=\; \langle a_1^1 \cdots a_1^{n_1}, \ldots, a_k^1 \cdots a_k^{n_k} \rangle \,.$$

It then follows that $\tau = \langle \sigma, \psi \rangle^{\mathscr{T}}(\tau|_1, \ldots, \tau|_n)$. □

The decomposition of segmented tots does not need to be unique; for example,

$$\langle a, \langle 011 \rangle \rangle^{\mathscr{T}}(\langle a, \langle 0, 1 \rangle \rangle^{\mathscr{T}}(\langle b, \langle 0 \rangle \rangle^{\mathscr{T}})) \;=\; \langle a, \langle 01 \rangle \rangle^{\mathscr{T}}(\langle a, \langle 01 \rangle \rangle^{\mathscr{T}}(\langle b, \langle 0 \rangle \rangle^{\mathscr{T}})) \,.$$

Both expressions describe the same tot with term $a(a(b))$ and the node order $\varepsilon \preceq 1 \preceq 11$. In the first expression, the subtot of the first child is artificially split into two subsequent segments, which are then fused without insertion of nodes into the gap. On the right, this artificial split is avoided. It is not difficult to see that every segmented tot can be constructed in a unique manner, when disallowing immediate repetitions in constructors ψ, i.e. segments in $\mathbb{N}^* i i \mathbb{N}^*$, for every $i \in \mathbb{N}$. Let $\mathscr{T}'$ be the restriction of $\mathscr{T}$ to operators (σ, ψ) without immediate repetitions in ψ.

Theorem 3. *The algebra of k-segmented tots $\mathscr{T}'$ is isomorphic to a $[k]$-sorted term algebra T_Δ.*

Proof (Sketch). The multi-sorted signature Δ contains all symbols (σ, ψ) of type $k_1 \times \cdots \times k_m \to k$, where $\sigma \in \Sigma$ has arity m, and ψ is a k-segmentation of the multiset $\{0 \mapsto 1\} \cup \{\, i \mapsto k_i \mid i \in [m] \,\}$ without immediate repetitions. The interpretation function $[\![\cdot]\!]^{\mathscr{T}'} \colon T_\Delta \to \mathscr{T}'$ is a homomorphism, which is onto by Proposition 3, and one-to-one since immediate repetitions are forbidden.

8 Automata for Segmented Tots

We define automata for the algebra of segmented tots $\mathscr{T}'$ as automata for the multi-sorted term algebra T_Δ. Since well-sorted terms over Δ are recognizable, we can use standard tree automata that recognized languages of well-sorted terms.

We call a set of segmented tots $L \subseteq \mathscr{T}'$ with bounded gap-degree *recognizable*, if and only if the corresponding set of term encodings $\{ t \in T_\Delta \mid t^{\mathscr{T}'} \in L \}$ is recognizable by a tree automaton. Standard results on tree automata [8] show that recognizable set of segmented tots are closed under union, intersection, complementation and projection. Note also, that the set of all tots (i.e., segmented tots without gaps) is recognizable

Theorem 4. *A gap-bounded set of segmented tots is MSO-definable if and only if it is recognizable. The transformations from formulas to automata and back are effective.*

Proof. The transformation of automata for T_Δ into MSO for segmented tots needs to express the rules of the automata. This works as usual, except that one has to express the operators $\langle \sigma, \psi \rangle^{\mathscr{T}'}$ in the MSO of segmented tots. This is straightforward.

The transformation of MSO formulas to automata works as for Thatcher and Wright's theorem [4]: Boolean connectives and monadic second-order quantifiers are mapped to the closure operators for union, complementation and projection. What remains to show is that we can construct automata that check the atomic predicates $x \preceq y$ and $x \equiv y$. The fundamental insight in this construction is that it suffices to remember, by means of the state information that the tree automaton provides, for each node π, in which segments of the subtot $\tau|_\pi$ the nodes $\alpha(x), \alpha(y)$ occur (if any). Based on this information, the question whether $\alpha(x) \prec \alpha(y)$ can be decided by looking at the label $\langle \sigma, \psi \rangle$ at π: if $\alpha(x)$ occurs in the j_xth segment of the i_xth substructure of $\tau|_\pi$, and $\alpha(y)$ occurs in the j_yth segment of the i_yth substructure, then $\alpha(x) \prec \alpha(y)$ iff the j_xth occurrence of the symbol i_x precedes the j_yth occurrence of the symbol i_y; if $\alpha(x)$ or $\alpha(y)$ does not occur in some substructure, but equals π, then the relevant occurrence is the (single) occurrence of the symbol 0 in ψ. A similar argument holds for the case $\alpha(x) \equiv \alpha(y)$. As long as the number of gaps (and hence, the number of segments) in a tot is bounded, the required state set is of bounded size.

One perhaps surprising consequence of this Theorem is that total orders of MSO-defined sets of tots with bounded gap-degree are always definable by MSO over terms without the order. Given a term $t \in T_\Sigma$, variables x_1, x_2 and nodes $\pi, \pi' \in \mathit{nod}(t)$, let $t * [x \mapsto \pi, y \mapsto \pi'] \in T_{\Sigma \times 2^{\{x,y\}}}$ be the tree obtained from t by annotating all its nodes by the set of variables that are mapped to it. Similarly, we define $\tau * [x \mapsto \pi, y \mapsto \pi'] \in TOT_{\Sigma \times 2^{\{x,y\}}}$ to be the tot $\langle \mathit{term}(\tau) * [x \mapsto \pi, y \mapsto \pi'], \mathit{seg}(\tau) \rangle$ in which the variable assignment is annotated.

Corollary 1. *Let ϕ be a closed MSO formula for tots in TOT_Σ with bounded gap degree. Then the following term language $\{ \mathit{term}(\tau) \in T_\Sigma \mid \tau \models \phi \}$ is regular as well as $\{ \mathit{term}(\tau) * [x \mapsto \alpha(x), y \mapsto \alpha(y)] \in T_{\Sigma \times 2^{\{x,y\}}} \mid \tau \models \phi \}$.*

Proof. The set $L_1 = \{ \tau \in TOT_\Sigma \mid \tau \models \phi \}$ is an MSO defined set of tots over Σ. Let A_1 be an tree automaton over Δ that recognizes $\{ s \in T_\Delta \mid s^{\mathscr{T}} \in L_1 \}$ according to Theorem 4. The projection A_1 to Σ recognizes $\{ \mathit{term}(\tau) \in T_\Sigma \mid \tau \in L_1 \}$ as required, where the rules of the projected automaton B_1 are as follows:

$$\frac{\langle \sigma, \psi \rangle(p_1, \ldots, p_m) \to p \in \mathit{Rules}(A_1)}{\sigma(\langle p_1, \psi_1 \rangle, \ldots, \langle p_m, \psi_m \rangle) \to \langle p, \psi \rangle \in \mathit{Rules}(B_1)}$$

For the second statement, let $L_2 = \{ \tau * [x \mapsto \alpha(x), y \mapsto \alpha(y)] \in TOT_{\Sigma \times 2^{\{x,y\}}} \mid \tau, \alpha \models \phi \wedge x \preceq y \}$ and automaton A_2 recognize $\{ s \in T_{\Delta \times 2^{\{x,y\}}} \mid s^{\mathscr{T}} \in L_2 \}$ according to Theorem 4. The projection A_2 to $\Sigma \times 2^{\{x,y\}}$ recognizes the second term language of the corollary $\{ \mathit{term}(\tau) * [x \mapsto \alpha(x), y \mapsto \alpha(y)] \in T_{\Sigma \times 2^{\{x,y\}}} \mid \tau, \alpha \models \phi \wedge x \preceq y) \}$, where the projection automaton B_2 has the following rules:

$$\frac{\langle \langle \sigma, \psi \rangle, V \rangle(p_1, \ldots, p_m) \to p \in \mathit{Rules}(A_2)}{\langle \sigma, V \rangle(\langle p_1, \psi_1 \rangle, \ldots, \langle p_n, \psi_m \rangle) \to \langle p, \psi \rangle \in \mathit{Rules}(B_2)}$$

Final remark. We have shown that the MSO satisfiability problem for tots with bounded gap-degree is decidable, while the general case is undecidable. A question that we need to leave unanswered for now is whether the first-order satisfiability problem of general tots is decidable in contrast to the case of MSO.

References

1. Culotta, A., Sorensen, J.: Dependency tree kernels for relation extraction. In 42nd Annual Meeting of the ACL (2004) 423–429
2. Quirk, C., Menezes, A., Cherry, C.: Dependency treelet translation: Syntactically informed phrasal SMT. In 43rd Annual Meeting of the ACL (2005) 271–279
3. Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel, S., Yuret, D.: The CoNLL 2007 shared task on dependency parsing. In Joint Conference on Empirical Methods in NLP and Computational Natural Language Learning (2007) 915–932
4. Thatcher, J.W., Wright, J.B.: Generalized finite automata with an application to a decision problem of second-order logic. Math. System Theory **2** (1968) 57–82
5. Courcelle, B.: Handbook of graph grammars and computing by graph transformations, volume 1: Foundations. Handbook of Graph Grammars. (1997)
6. Gottlob, G., Koch, C.: Monadic queries over tree-structured data. In 17th Annual IEEE Symposium on Logic in Computer Science (2002) 189–202
7. Courcelle, B.: Graph Grammars and Logic. Book in preparation (2008)
8. Comon, H., Dauchet, M., Gilleron, R., Löding, C., Jacquemard, F., Lugiez, D., Tison, S., Tommasi, M.: Tree automata techniques and applications. (1997/2007).
9. Rabin, M.: Decidability of Second-Order Theories and Automata on Infinite Trees. Transactions of the American Mathematical Society **141** (1969) 1–35
10. Gottlob, G., Koch, C.: Monadic datalog and the expressive power of languages for web information extraction. In: 21rd ACM PODS (2002) 17–28
11. Plátek, M., Holan, T., Kuboň, V.: On relaxability of word order by D-Grammars. In 3rd Int. Conf. on Combinatorics, Computability and Logic. DMTCS (2001) 159–174

12. Kuhlmann, M.: Dependency Structures and Lexicalized Grammars. Doctoral dissertation, Saarland University, Saarbrücken, Germany (2007)
13. Koller, A., Striegnitz, K.: Generation as dependency parsing. In 40th Annual Meeting of the ACL (2002) 17–24
14. Holan, T., Kuboň, V., Oliva, K., Plátek, M.: Two useful measures of word order complexity. Work. on Processing of Dependency-Based Grammars (1998) 21–29
15. Kuhlmann, M., Nivre, J.: Mildly non-projective dependency structures. In 21st COLING-ACL, Main Conference Poster Sessions (2006) 507–514
16. Mezei, J., Wright, J.B.: Algebraic automata and context-free sets. Information and Control **11** (1967) 3–29
17. Courcelle, B.: Recognizable sets of unrooted trees. In Nivat, M., Podelski, A., eds.: Tree Automata and Languages. Elsevier Science (1992)
18. Carme, J., Niehren, J., Tommasi, M.: Querying unranked trees with stepwise tree automata. In 19th RTA, Vol 3091 of LNCS (2004) 105 – 118