

MSO Definable Text Languages*

Hendrik Jan Hoozeboom and Paulien ten Pas

Leiden University, Department of Computer Science
P.O.Box 9512, 2300 RA Leiden, The Netherlands
{hjh,pas}@rulwinw.leidenuniv.nl

Abstract. A *text* is a word together with an (additional) linear ordering. Each text has a generic tree representation, called its *shape*. We consider texts in a logical and an algebraic framework, and we prove that the classes of monadic second order definable and of recognizable text languages coincide. In particular we demonstrate that the construction of the shape of a text can be formalized in terms of our monadic second-order logic. We briefly consider right-linear grammars for texts.

Introduction

The theory of *2-structures*, introduced in [7], studies modular decompositions of graph-like structures. In this framework *texts* appear as representations for a certain subclass of 2-structures, see [8]. A text is in essence a word extended with an (additional) linear ordering of its positions.

In previous work grammars generating context-free text languages were studied [10]. The question addressed in this paper is: which text languages constitute the class of “*regular*” text languages? We give a logical definition of “*regular*” text languages, and show that the obtained class may equivalently be defined in an algebraic and in a grammatical framework. Crucial in our approach is that each text allows a hierarchical representation by a tree-structure, which indicates how the text is built from certain basic building blocks. Such a tree representation in general is not unique. There exists however a generic tree representation, called the *shape*, that can be constructed from the text. By associating to each text language K the tree language $sh(K)$ of corresponding shapes, we obtain a strong connection between text languages and tree languages.

The following theorem summarizes our results (where $\text{TXT}_\Pi(\Delta)$ denotes the set of all texts over the alphabet Δ and basic building blocks from a finite set Π , see Section 1).

Main Theorem *Let $K \subseteq \text{TXT}_\Pi(\Delta)$ be a text language. Equivalent are:*

- (i) K is mso definable.
- (ii) K is recognizable.
- (iii) K is right-linear.
- (iv) $sh(K)$ is regular.

* To appear in the proceedings of MFCS'94, Košice, Slovakia, 22-26 August 1994.
Research supported by the EBRA Working Group ASMICS 2.

For recognizable (or regular) word languages and tree languages there exist characterizations analogous to (i-iii) above, see [1, 6, 16, 18]. Since every word is a specific text, our Main Theorem generalizes the analogous result for word languages. Its proof, however, is based on the corresponding result for tree languages. The generalization is non-trivial: the underlying words of a recognizable text language do not necessarily form a regular word language (in fact they form a context-free language).

One may view texts as specific graphs, obtained by combining two linear orders. For graph languages in general there does not seem to be such a “stable” notion of recognizability, in the sense that it has an equivalent grammatical or logical characterization. These matters are discussed extensively by Courcelle in [4], where he conjectures that (i) and (ii) are equivalent for graph languages of bounded tree-width. In support of this conjecture he shows that recognizability and mso definability are equivalent within certain “parsable” sets of graphs [4, Theorem 4.8], and he gives examples of sets of graphs for which this property holds. Using this terminology, we show that each set $\text{TXT}_H(\Delta)$ is another example of a parsable set (Theorem 2.4). We may not apply the cited result to obtain the equivalence of (i) and (ii), as our respective algebraic frameworks differ.

In Section 2 we show that a text language K is mso definable iff the corresponding tree language $sh(K)$ is mso definable. Then, in Section 3, we observe that K is right-linear iff $sh(K)$ is regular. The equivalence of (i) and (iii) follows from the characterization of regular tree languages. Finally, we use an additional result from universal algebra in connecting the text language K with the set of all trees representing texts from K in the case of recognizability, which then will imply the equivalence of (i) and (ii).

1 Texts and Trees

In this preliminary section we present results on texts, and trees representing them, that are needed for this paper. We refer to [8, 9, 10] for more details.

We view a linear order as a non-empty sequence of distinct elements, which form the *domain* of the linear order. For a linear order $\rho = (x_1, \dots, x_n)$, $n \geq 1$, a *segment* of ρ is a set $\{x_j, x_{j+1}, \dots, x_k\}$ with $j \leq k$. A *suffix* of ρ is a segment containing x_n . If $i < j$ then we say that x_i *precedes* x_j in ρ .

Definition 1.1 Let Δ be an alphabet. A *text* τ (over Δ) is a triple $(\lambda, \rho_1, \rho_2)$, where ρ_1 and ρ_2 are linear orders such that the domain of ρ_1 equals the domain of ρ_2 , and λ is a labeling function from this common domain to Δ . \square

For a text $\tau = (\lambda, \rho_1, \rho_2)$, the pair (ρ_1, ρ_2) determines its structural properties; such a pair (ρ_1, ρ_2) of linear orders with a common domain is called a *bi-order*. The common domain of ρ_1 and ρ_2 is the *domain* of the text τ , denoted by $dom(\tau)$. If $\rho_1 = (x_1, \dots, x_n)$, then the *word* of τ is the word $\lambda(x_1) \cdots \lambda(x_n)$.

By the *length* of a text (or bi-order) we mean the number of elements in its domain. For our purposes the identities of these elements are not important. Therefore, we usually assume that the domain of a text (or bi-order) of length

n equals $\{1, \dots, n\}$, and that the first order is $(1, \dots, n)$; we refer to this as the *standard form* of a text or bi-order. We may then represent a bi-order by its second order (i_1, \dots, i_n) and a text by the pair $(w, (i_1, \dots, i_n))$, where w is the word of the text.

For a bi-order π of length m and texts τ_1, \dots, τ_m , the *substitution* of τ_1, \dots, τ_m into π , denoted by $[\pi \leftarrow (\tau_1, \dots, \tau_m)]$, is the text constructed as follows: let π be in standard form, take copies of τ_1, \dots, τ_m with mutually disjoint domains X_1, \dots, X_m , and define the substitution text on the domain $\bigcup_{i=1}^m X_i$ such that the label of x from X_i is the label of x in τ_i , and for $k \in \{1, 2\}$, $x \in X_i$ precedes $y \in X_j$ in the k -th order if either i precedes j in the k -th order of π , or $i = j$ and x precedes y in the k -th order of τ_i .

Note that the above X_i 's become segments of both the first and the second order of the constructed text. For an arbitrary text τ , a non-empty set $X \subseteq \text{dom}(\tau)$ that is a segment in both the first and the second order of τ is called a *clan* of τ . Clearly, for each text τ , $\text{dom}(\tau)$ is a clan of τ and $\{x\}$ is a clan of τ for each $x \in \text{dom}(\tau)$. These clans are called the *trivial clans*. If the only clans of a text are the trivial clans, then the text is called *primitive*. We also speak of primitive bi-orders. Note that the two bi-orders of length 2, in standard form given by $(1, 2)$ and $(2, 1)$, respectively, are both primitive; they will be denoted by σ_f (f for “forward”) and σ_b (b for “backward”).

One might say that a text τ is “decomposable” if it can be obtained as the substitution of some (proper) subttexts into a bi-order π , or, equivalently, if the domains of these subttexts are clans forming a partition of the domain (π giving their relative first and second ordering). Clearly, primitive texts only allow such decomposition into singletons. By exhaustively decomposing subttexts, we obtain a tree-structure for a given text, called a *primitive representation* of the text, which indicates how it is built up from singletons and primitive (i.e., indecomposable) bi-orders.

Example 1.2 Let τ be the text $(acabaacbc, (5, 2, 4, 1, 3, 6, 7, 8, 9))$. In Fig. 1 two primitive representations of τ are given, where π is the primitive bi-order given by $(2, 4, 1, 3)$ in standard form. Consider the left tree t . At the root τ is decomposed into the subttexts corresponding with the clans $\{1, 2, 3, 4, 5\}$ and $\{6, 7, 8, 9\}$, in standard form $(acaba, (5, 2, 4, 1, 3))$ and $(acbc, (1, 2, 3, 4))$, respectively; note that τ is reobtained by substituting these subttexts into the root label σ_f which gives the relative orderings of the two clans. At the left child of the root the subttext $(acaba, (5, 2, 4, 1, 3))$ is decomposed into the subttexts corresponding with clans $\{1, 2, 3, 4\}$ and $\{5\}$ relatively ordered by σ_b ; the subttext corresponding with $\{1, 2, 3, 4\}$ is primitive with underlying bi-order π .

For each internal node of a primitive representation t its label provides two orderings on its children, where the first order is assumed to be the left-to-right order. In this way t is a “doubly” ordered tree, and we can recover the standard form of τ from t as follows: the word of τ is the yield of t , and the (second) order on the positions of this word is the ordering on the leaves induced by the local second orders in the obvious way, assuming that the leaves are named $1, \dots, 9$ from left to right. \square

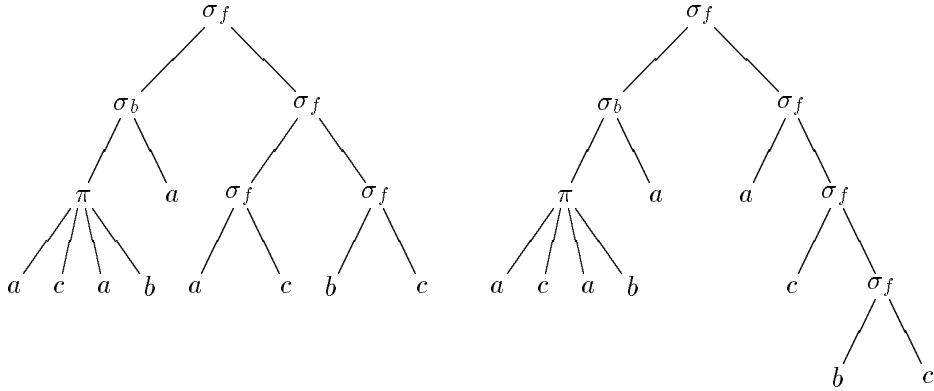


Fig. 1. Two primitive representations of τ

The trees used in this paper are directed ordered trees with node-labels. Therefore, they can be represented as terms over a ranked alphabet (cf. the definition of trees in [14]). A *ranked alphabet* Σ is a finite alphabet of operator symbols, where each operator symbol $\sigma \in \Sigma$ has a rank, which is a natural number. The set of Σ -terms, denoted by F_Σ , is the smallest set of words over Σ and the auxiliary symbols \langle and \rangle such that $\sigma \in F_\Sigma$ for every $\sigma \in \Sigma$ of rank 0, and $\sigma\langle t_1 \cdots t_m \rangle \in F_\Sigma$ for all $\sigma \in \Sigma$ of rank $m > 0$ and $t_1, \dots, t_m \in F_\Sigma$. For primitive representations we use ranked alphabets of the following specific form. For a finite set Π of primitive bi-orders of length ≥ 2 , and an alphabet Δ , let Σ be the ranked alphabet $\Pi \cup \Delta$ such that the rank of each $\sigma \in \Pi$ is its length and the rank of each $a \in \Delta$ is 0. Each tree t in F_Σ is then a primitive representation; the text represented by t is denoted by $\text{txt}(t)$ and is recursively defined by $\text{txt}(a) = (a, (1))$ for $a \in \Delta$, and $\text{txt}(\pi\langle t_1 \cdots t_n \rangle) = [\pi \leftarrow (\text{txt}(t_1), \dots, \text{txt}(t_n))]$ for $\pi \in \Pi$ of length $n \geq 2$, and $t_1, \dots, t_n \in F_\Sigma$.

The difference between primitive representations of one text is limited to binary subtrees containing only the operator symbol σ_f and binary subtrees containing only σ_b . This is a consequence of general results from the decomposition theory for 2-structures.

Proposition 1.3 *Two primitive representations of the same text differ only in binary subtrees.*

We fix one specific primitive representation for each text τ , called the *r-shape* of τ , denoted by $sh(\tau)$, by demanding that each binary subtree is in a right-most form, i.e., the r-shape of a text is the primitive representation such that it has no subtree of the form $\sigma_f\langle\sigma_f\langle t_1 t_2 \rangle t_3 \rangle$ or $\sigma_b\langle\sigma_b\langle t_1 t_2 \rangle t_3 \rangle$. E.g., the r-shape of the text τ from Example 1.2 is the right tree in Fig. 1. (Note that the r-shape slightly differs from the usual shape in [7].)

The r-shape of a text can be characterized in terms of clans. First observe that the nodes of a given primitive representation t correspond with clans of $\text{txt}(t)$, viz., the clans used in the decomposition reflected by t . A clan of a text τ is a *prime clan* if it is not overlapping with any other clan of τ , where sets X and Y are *overlapping* if $X \cap Y \neq \emptyset$, $X - Y \neq \emptyset$, and $Y - X \neq \emptyset$.

Proposition 1.4 *A primitive representation t is the r -shape of a text τ iff the clans of τ corresponding with the nodes of t are precisely all clans of τ that are suffixes (w.r.t. the first order) of prime clans of τ .*

For a ranked alphabet $\Sigma = \Pi \cup \Delta$ as described above, we use $\text{TXT}_\Pi(\Delta)$ to denote the set of texts over Δ that have a primitive representation in F_Σ ; the subset of F_Σ consisting of the r -shapes of these texts is denoted by $\text{SH}_\Pi(\Delta)$; $\text{TXT}(\Delta)$ denotes the set of all texts over Δ .

2 MSO Definable Text Languages

In this section we view texts, and trees representing them, as graphs. We use monadic second-order logic on graphs to define the class of mso definable text languages, rather than introducing a separate logic for texts.

Let Ω and Γ be alphabets. A *graph* over Ω and Γ is a triple $g = (V, E, \lambda)$, where V is the set of nodes, $E \subseteq V \times \Gamma \times V$ the set of edges, and $\lambda : V \rightarrow \Omega$ the node labelling. The set of all graphs over Ω and Γ is denoted by $\text{GR}(\Omega, \Gamma)$.

The monadic second order logic $\text{MSO}(\Omega, \Gamma)$ expresses properties of graphs over Ω and Γ . The logic allows both first order variables x, y, \dots ranging over nodes, and (monadic) second order variables X, Y, \dots ranging over sets of nodes. There are four types of atomic formulas: $x = y$, expressing that nodes x and y are equal; $x \in X$, expressing x is an element of X ; $\text{lab}_a(x)$, expressing node x has label a (with $a \in \Omega$); and $\text{edge}_\gamma(x, y)$; expressing there is an edge from x to y with label γ (with $\gamma \in \Gamma$).

Formulas are built from atomic formulas with the propositional connectives $\neg, \wedge, \vee, \rightarrow$, using the quantifiers \forall and \exists both for node-variables and node set-variables. For better readability we use abbreviations like $X \subseteq Y$ and $X \cap Y = \emptyset$.

Example 2.1 A predicate \prec claiming the existence of a path from x to y :

$$x \prec y \equiv \forall X [x \in X \wedge \forall u \forall v (u \in X \wedge \text{edge}(u, v) \rightarrow v \in X) \rightarrow y \in X]$$

where $\text{edge}(u, v) \equiv \bigvee_{\gamma \in \Gamma} \text{edge}_\gamma(u, v)$. This example is from [18]. □

Given a closed formula φ of $\text{MSO}(\Omega, \Gamma)$, and a graph g from $\text{GR}(\Omega, \Gamma)$ we write $g \models \varphi$ if g satisfies φ , i.e., if φ is true when interpreted over g . Now φ defines the graph language of all graphs satisfying φ : $L(\varphi) = \{g \in \text{GR}(\Omega, \Gamma) \mid g \models \varphi\}$. Such a graph language is said to be *mso definable*.

To represent ordered trees as graphs we use edge-labels to explicitly determine the relative ordering of the children of each node: the natural numbers $1, \dots, m$ label the outgoing edges of a node of arity m . Thus, for the ranked alphabet Σ , trees in F_Σ are identified with specific graphs in $\text{GR}(\Sigma, \{1, \dots, M\})$, where M is the maximal rank of symbols in Σ .

Also texts have natural representations as graphs. We use edges to represent the two linear orders of the text, with edge-labels to identify the ordering. Formally, a text $\tau = (\lambda, \rho_1, \rho_2)$ over Δ is identified with the graph

$g = (V, E, \lambda)$ over Δ and $\{1, 2\}$, where $V = \text{dom}(\tau)$, and $E = \{(x, i, y) \mid x, y \in V, x \text{ precedes } y \text{ in } \rho_i, i \in \{1, 2\}\}$. Thus, a text language is *mso definable* if it is mso definable as set of graphs over $\text{GR}(\Delta, \{1, 2\})$.

The set of texts $\text{TXT}(\Delta)$ in $\text{GR}(\Delta, \{1, 2\})$ is easily seen to be mso definable, as we only have to specify the fact that the edges with each label form a linear ordering. Unfortunately, the definability of $\text{TXT}_\Pi(\Delta)$ within $\text{GR}(\Delta, \{1, 2\})$ is less transparent. We solve this problem in the proof of Theorem 2.4. On the other hand, both F_Σ and the set $\text{SH}_\Pi(\Delta)$ of r-shapes in F_Σ , where $\Sigma = \Pi \cup \Delta$ is a ranked alphabet for primitive representations, are mso definable.

An *mso definable function* f , see [5, 11], specifies the construction of a new graph g' from a graph g using mso formulas. The construction starts by taking k copies of the nodes of g (k is fixed by f). Then for each copy, and each label a of g' , a formula $\varphi_a^i(x)$ determines whether the i -th copy x^i of node x of g is present in g' and has label a . Similarly, for each pair of copies and each edge label γ of g' , a formula $\varphi_\gamma^{i,j}(x, y)$ determines whether an edge with label γ is leading from the i -th copy x^i of x to the j -th copy y^j of y .

Definition 2.2 An *mso definable function* $f : \text{GR}(\Omega, \Gamma) \rightarrow \text{GR}(\Omega', \Gamma')$ is specified by a *domain formula* φ_{dom} , a *constant* $k \geq 1$, *node formulas* $\varphi_a^i(x)$, for every $a \in \Omega'$ and every $i \in \{1, \dots, k\}$, and *edge formulas* $\varphi_\gamma^{i,j}(x, y)$ for every $\gamma \in \Gamma'$ and all $i, j \in \{1, \dots, k\}$; all formulas are in $\text{MSO}(\Omega, \Gamma)$.

For $g \in L(\varphi_{\text{dom}})$ with node set V_g , the image $f(g)$ is $(\bigcup_{i \in \{1, \dots, k\}} V^i, E, \lambda)$, where for $i \in \{1, \dots, k\}$,

- $V^i = \{x^i \mid x \in V_g, \text{ there is exactly one } a \in \Omega' \text{ such that } g \models \varphi_a^i(x)\}$,
- $E = \{(x^i, \gamma, y^j) \mid x^i \in V^i, y^j \in V^j, i, j \in \{1, \dots, k\}, g \models \varphi_\gamma^{i,j}(x, y)\}$, and
- $\lambda(x^i) = a$ if $g \models \varphi_a^i(x)$, for $x^i \in V^i, i \in \{1, \dots, k\}$. □

Proposition 2.3 ([5]) *Let $f : \text{GR}(\Omega, \Gamma) \rightarrow \text{GR}(\Omega', \Gamma')$ be an mso definable function. If $L \subseteq \text{GR}(\Omega', \Gamma')$ is mso definable, then $f^{-1}(L) = \{g \in \text{GR}(\Omega, \Gamma) \mid f(g) \in L\}$ is mso definable.*

We verify that the function $\text{txt} : \text{GR}(\Sigma, \{1, \dots, M\}) \rightarrow \text{GR}(\Delta, \{1, 2\})$, that assigns to each tree in F_Σ the text it represents, is mso definable, like the translation from shape to 2-structure, see [13, Section 4.3]. Start with a single copy of each node of the tree. Internal nodes are removed (selecting nodes with labels in Δ) and the edges between a pair of nodes in the text are determined using the label associated to the least common ancestor of these nodes in the tree.

Formally, txt is fixed by a domain formula defining F_Σ , constant $k = 1$, node formulas $\varphi_a(x) \equiv \text{lab}_a(x)$, for $a \in \Delta$, and, for $m \in \{1, 2\}$, edge formulas $\varphi_m(x, y) \equiv$

$$\bigvee_{\pi \in \Pi} \bigvee_{i <_m^\pi j} [\exists z \exists x_1 \exists y_1 (\text{lab}_\pi(z) \wedge \text{edge}_i(z, x_1) \wedge \text{edge}_j(z, y_1) \wedge x_1 \prec x \wedge y_1 \prec y)]$$

where $x \prec y$ expresses that there is a path from x to y (cf. Example 2.1), and $i <_m^\pi j$ abbreviates “ i precedes j in the m -th order of π ”. Note that $i, j \in \{1, \dots, M\}$, hence the second disjunction is finite.

We now consider our key result: the construction of the r-shape from a text is an mso definable function. We start by giving formulas that define the structure of the r-shape of a text, identifying the nodes of the r-shape with clans of the text. By Proposition 1.4, the nodes then are exactly the suffixes of prime clans. *nodes*. A set X is a clan of a text iff it is a segment in both orderings, which can easily be expressed as an mso formula $\text{clan}(X)$. Prime clans are by definition those clans that do not overlap other clans. We obtain the formulas:

$$\text{prim}(X) \equiv \text{clan}(X) \wedge \forall Y[\text{clan}(Y) \rightarrow (X \cap Y = \emptyset \vee X \subseteq Y \vee Y \subseteq X)]$$

$$\text{node}(X) \equiv \text{clan}(X) \wedge \exists Z(\text{prim}(Z) \wedge \text{suff}_1(X, Z))$$

where $\text{suff}_1(X, Z)$ expresses that X is a suffix of Z as segments in the first order. The usual child-parent relation $\text{child}(X, Y)$ can be expressed using set inclusion. *node labels*. The label of the (internal) node X in the r-shape is determined by the quotient bi-order of X with respect to its children, i.e. by the relative ordering of the children of X in the first and second ordering of the text. Thus, X has the associated bi-order $\pi = (i_1, \dots, i_n)$ in standard form, if the following predicate, denoted $\text{quot}_\pi(X)$, is satisfied:

$$\exists X_1 \dots X_n [(X = \bigcup_{j=1}^n X_j) \wedge \bigwedge_{j=1}^n \text{child}(X_j, X) \wedge (X_1 <_1 \dots X_n) \wedge (X_{i_1} <_2 \dots X_{i_n})]$$

where $X <_i Y$ abbreviates $\forall x \forall y (x \in X \wedge y \in Y \rightarrow \text{edge}_i(x, y))$, meaning that X precedes Y as a segment in the i -th order.

edges. The r-shape has edges connecting nodes to their children as usual in a tree. To order the children of a node, the predicate $\text{child}(X, Y)$ has to be extended to a predicate $\text{child}_k(X, Y)$ meaning that X is the k -th child of Y .

Theorem 2.4 *The mapping $sh : \text{GR}(\Delta, \{1, 2\}) \rightarrow \text{GR}(\Sigma, \{1, \dots, M\})$, that assigns to each text in $\text{TXT}_\Pi(\Delta)$ its r-shape, is an mso definable function.*

Proof. We avoid an unelegant duplication in our constructions by first considering the domain $\text{TXT}(\Delta)$ rather than $\text{TXT}_\Pi(\Delta)$. In a second step we show that a proper domain formula, defining $\text{TXT}_\Pi(\Delta)$, exists.

defining the mapping. We formalize the construction of the r-shape as an mso definable function. To this end we need to show how to obtain the nodes of the r-shape by duplicating the positions of the text.

We associate with each internal node of the r-shape a position in the text or, equivalently, we associate with each internal node a leaf (see also [17]). (Leaves are the singleton clans of the text.) If position x is associated to the internal node X , then the copy x^2 represents X , the first copy x^1 represents the leaf x .

The association we use is known (for binary trees) as the inorder successor of the node: it is the leaf that is found by first moving to the last child, and then repeatedly moving to the first child of the node visited. This gives an injective mapping of internal nodes to leaves of the tree, expressible by an mso formula:

$$\text{assoc}(x, X) \equiv \exists Y(\text{frst}_1(x, Y) \wedge \text{child}(Y, X) \wedge \text{suff}_1(Y, X))$$

where $\text{fst}_1(x, Y)$ expresses that x is the first element of Y (in the first ordering).

The mapping sh' is specified as an mso definable function as follows:
a domain function φ_{dom} defining $\text{TXT}(\Delta)$, the constant $k = 2$,
for $a \in \Delta$, and $\pi \in \Pi$, the node formulas $\varphi_a^1(x) \equiv \text{lab}_a(x)$, $\varphi_\pi^1(x) \equiv \varphi_a^2(x) \equiv$
false, and $\varphi_\pi^2(x) \equiv \exists X(\text{assoc}(x, X) \wedge \text{quot}_\pi(X))$,
for $k \in \{1, \dots, M\}$, the edge formulas $\varphi_k^{1,1}(x, y) \equiv \varphi_k^{1,2}(x, y) \equiv$ false,
 $\varphi_k^{2,1}(x, y) \equiv \exists X \exists Y(\text{assoc}(x, X) \wedge (Y = \{y\}) \wedge \text{child}_k(Y, X))$, and
 $\varphi_k^{2,2}(x, y) \equiv \exists X \exists Y(\text{assoc}(x, X) \wedge \text{assoc}(y, Y) \wedge \text{child}_k(Y, X))$
defining the domain. The mso definable function sh' constructed above does not
have the proper domain. For texts in $\text{TXT}_\Pi(\Delta)$ it satisfies our needs. For other
texts in $\text{TXT}(\Delta)$ the labelling formulas are undefined for internal nodes of the r-
shape that are labelled by operations not in Π . According to the definition of mso
definable functions, these nodes are then removed from the constructed graph,
leaving an unconnected graph. Hence, sh' constructs a tree if and only if the
input graph is an element of $\text{TXT}_\Pi(\Delta)$. Consequently, $\text{TXT}_\Pi(\Delta) = sh'^{-1}(F_\Sigma)$,
which, by Proposition 2.3, is an mso definable subset of $\text{GR}(\Delta, \{1, 2\})$.

The final conclusion made in the above proof is interesting in its own right.

Corollary 2.5 $\text{TXT}_\Pi(\Delta)$ is an mso definable subset of $\text{GR}(\Delta, \{1, 2\})$.

We are ready to prove the main result of this section.

Theorem 2.6 Let $K \subseteq \text{TXT}_\Pi(\Delta)$. K is mso definable iff $sh(K)$ is mso definable iff $\text{txt}^{-1}(K)$ is mso definable.

Proof. The functions txt and sh are mso definable, and the mso definable sets are closed under the Boolean operations. Thus, by Proposition 2.3, if K is mso definable, then $\text{txt}^{-1}(K)$ is mso definable. The other implications follow from the equalities $sh(K) = \text{txt}^{-1}(K) \cap \text{SH}_\Pi(\Delta)$ and $K = sh^{-1}(sh(K))$ (sh is injective).

The structure of a text interpreted as graph is independent of node-labels. Using [12] one obtains an operational characterization of the family of mso definable subsets of $\text{TXT}_\Pi(\Delta)$: it is the smallest family containing certain elementary languages that is closed under intersection, difference, and (node) relabellings.

3 Recognizable and Right-linear Text Languages

Using the results from the previous section we now explain the equivalences given in the Main Theorem.

We first define the algebraic and grammatical notions involved in the Main Theorem. We make use of the general definition of recognizability of subsets in a Σ -algebra. For a ranked alphabet Σ , a Σ -algebra \mathcal{A} is a pair (A, Σ) , where A is a set and each operator $\sigma \in \Sigma$ of rank $m \geq 0$ defines a mapping $\sigma^{\mathcal{A}} : A^m \rightarrow A$ (see, e.g., [2]). For a Σ -algebra $\mathcal{A} = (A, \Sigma)$, a subset $K \subseteq A$ is *recognizable* if

there is a finite Σ -algebra $\mathcal{Q} = (Q, \Sigma)$, a homomorphism $h : \mathcal{A} \rightarrow \mathcal{Q}$, and a subset $F \subseteq Q$ such that $h^{-1}(F) = K$.

This definition applies to tree languages in F_Σ , being subsets of the Σ -algebra of terms (F_Σ, Σ) , denoted by \mathcal{F}_Σ ; the finite algebra \mathcal{Q} corresponds with a so-called (deterministic) bottom-up tree recognizer.

To obtain an algebraic structure on texts we let $\Sigma = \Pi \cup \Delta$ be a ranked alphabet as before: the Σ -algebra $\mathcal{T}_\Sigma = (\text{TXT}_\Pi(\Delta), \Sigma)$ is defined by $a^{\mathcal{T}_\Sigma} = (a, (1))$ for $a \in \Delta$, and for $\pi \in \Pi$ of rank m , $\tau_1, \dots, \tau_m \in \text{TXT}_\Pi(\Delta)$, $\pi^{\mathcal{T}_\Sigma}(\tau_1, \dots, \tau_m) = [\pi \leftarrow (\tau_1, \dots, \tau_m)]$. A text language $K \subseteq \text{TXT}_\Pi(\Delta)$ is *recognizable* if it is recognizable in the Σ -algebra \mathcal{T}_Σ .

Note that the mapping $\text{txt} : F_\Sigma \rightarrow \text{TXT}_\Pi(\Delta)$ which assigns to each $t \in F_\Sigma$ the text $\text{txt}(t)$ is precisely the homomorphism evaluating terms of \mathcal{F}_Σ in \mathcal{T}_Σ . As txt is surjective we may apply the more general result [3, Proposition 1.7] to the algebra \mathcal{T}_Σ .

Proposition 3.1 *Let $K \subseteq \text{TXT}_\Pi(\Delta)$. K is recognizable in \mathcal{T}_Σ iff $\text{txt}^{-1}(K)$ is recognizable in \mathcal{F}_Σ .*

Now we turn to grammars defining tree and text languages.

A *regular tree grammar* is a 4-tuple $G = (N, \Sigma, P, S)$, where N is a set of nonterminal symbols, Σ is a ranked alphabet, P consists of productions of the form $A \rightarrow t$, where $A \in N$ and $t \in F_{\Sigma \cup N}$ (interpreting the nonterminals as operators of rank 0), and $S \in N$ is the startsymbol. G generates trees in F_Σ starting from S , where in one step a tree $t' \in F_{\Sigma \cup N}$ is derived from a $t \in F_{\Sigma \cup N}$ if there is a production $A \rightarrow u$ in P such that t' is obtained by attaching the tree u at a leaf of t with label A . A tree language is called *regular* if it is generated by a regular tree grammar.

For texts we have the notion of *context-free text grammar* (see [10]), which is a 4-tuple $G = (N, \Delta, P, \tau_0)$, where Δ is the alphabet of terminals, the productions in P are of the form $A \rightarrow \tau$, where $A \in N$ and $\tau \in \text{TXT}(N \cup \Delta)$, and the startsymbol τ_0 is a text of length 1 over N . For $\tau, \tau' \in \text{TXT}(N \cup \Delta)$, τ derives τ' (in G) if there is a production $A \rightarrow \nu \in P$ such that τ' is obtained by substituting ν in τ at a position with label A . The text language over Δ generated by such a context-free text grammar may be non-recognizable. Therefore, we restrict ourselves to *right-linear text grammars* (generating *right-linear text languages*), where each right-handside is primitive and there is neither a couple of productions of the form $A \rightarrow (BC, (1, 2))$, $B \rightarrow (DE, (1, 2))$ nor a couple $A \rightarrow (BC, (2, 1))$, $B \rightarrow (DE, (2, 1))$.

From the compatibility of substitution for texts and for trees, and from the observation that a right-linear text grammar corresponds with a regular tree grammar that generates r-shapes, we obtain the following result (the equivalence of (iii) and (iv) of the Main Theorem).

Theorem 3.2 *Let $K \subseteq \text{TXT}_\Pi(\Delta)$. K is right-linear iff $sh(K)$ is regular.*

As to the remaining equivalences in the Main Theorem, these finally follow from well-known similar equivalences for tree languages.

Proposition 3.3 ([6, 16, 18]) *Let $T \subseteq F_\Sigma$. T is mso definable iff T is recognizable iff T is regular.*

By this proposition and by Theorem 2.6, Proposition 3.1 implies the equivalence of (i) and (ii) of the Main Theorem, and Theorem 3.2 implies the equivalence of (i) and (iii) of the Main Theorem. An independent proof of the latter equivalence is given in [15], where for a right-linear text grammar a finite Σ -algebra recognizing its language is directly constructed.

References

1. J.R. Büchi, Weak second-order arithmetic and finite automata, *Zeitschrift für Mathematik, Logik und Grundlagen der Mathematik* 6 (1960) 66–92.
2. P.M. Cohn, *Universal Algebra*, Harper & Row, New York, 1965.
3. B. Courcelle, The monadic second-order logic of graphs. I. Recognizable sets of finite graphs, *Information and Computation* 85 (1990) 12–75.
4. B. Courcelle, The monadic second-order logic of graphs V: on closing the gap between definability and recognizability, *Theoret. Comput. Sci.* 80 (1991) 153–202.
5. B. Courcelle, Monadic second-order definable graph transductions, *Lecture Notes in Computer Science* 581 (1992) 124–144.
6. J. Doner, Tree acceptors and some of their applications, *J. of Comp. and System Sci.* 4 (1970) 406–451.
7. A. Ehrenfeucht and G. Rozenberg, Theory of 2-structures, Parts I and II, *Theoret. Comput. Sci.* 70 (1990) 277–342.
8. A. Ehrenfeucht and G. Rozenberg, T-functions, T-structures, and texts, *Theoret. Comput. Sci.* 116 (1993) 227–290.
9. A. Ehrenfeucht, P. ten Pas, and G. Rozenberg, Combinatorial properties of texts, *RAIRO, Theor. Inf.* 27 (1993) 433–464.
10. A. Ehrenfeucht, P. ten Pas, and G. Rozenberg, Context-free text grammars, *Acta Informatica* 31 (1994) 161–206.
11. J. Engelfriet, A characterization of context-free NCE graph languages by monadic second-order logic on trees, *Lecture Notes in Computer Science* 532 (1991) 311–327.
12. J. Engelfriet, A regular characterization of graph languages definable in monadic second-order, *Theoret. Comput. Sci.* 88 (1991) 139–150.
13. J. Engelfriet, T. Harju, A. Proskurowski, and G. Rozenberg, Survey on graphs as 2-structures and parallel complexity of decomposition, Technical Report 93-06 (1993), Dept. of Comp. Sci, Leiden University.
14. F. Gecseg and M. Steinby, *Tree Automata*, Akademiai Kiado, Budapest, 1984.
15. H.J. Hoogeboom and P. ten Pas, Text languages in an algebraic framework, *Fundamenta Informaticae*, to appear.
16. J. Mezei and J.B. Wright, Algebraic automata and context-free sets, *Information and Control*, 11 (1967) 3–29.
17. A. Potthoff and W. Thomas, Regular tree languages without unary symbols are star-free, *Lecture Notes in Computer Science* 710 (1993) 396–405.
18. J.W. Thatcher, J.B. Wright, Generalized finite automata theory with an application to a decision problem of second-order logic, *Math. Syst. Th.* 2 (1968) 57–82.