Linear-Time Algorithms for Graphs of Bounded Rankwidth: A Fresh Look Using Game Theory (Extended Abstract) [⋆]

Alexander Langer, Peter Rossmanith, and Somnath Sikdar

RWTH Aachen University, 52074 Aachen, Germany.

Abstract. We present an alternative proof of a theorem by Courcelle, Makowski and Rotics [6] which states that problems expressible in MSO_1 are solvable in linear time for graphs of bounded rankwidth. Our proof uses a game-theoretic approach and has the advantage of being self-contained. In particular, our presentation does not assume any background in logic or automata theory. Moreover our approach can be generalized to prove other results of a similar flavor, for example, that of Courcelle's Theorem for treewidth [3, 19].

1 Introduction

In this paper we give an alternate proof of the theorem by Courcelle, Makowski and Rotics [6]: *Every decision or optimization problem expressible in* MSO_1 *is linear time solvable on graphs of bounded cliquewidth.* We prove the same theorem for graphs of bounded rankwidth. Since rankwidth and cliquewidth are equivalent width measures in the sense that a graph has bounded rankwidth iff it has bounded cliquewidth, it does not matter which of these width measures is used to state the theorem [21].

The proof by Courcelle et al. [6, 7] makes use of the Feferman-Vaught Theorem [10] adapted to MSO (cf. [14, 15]) and MSO transductions (cf., [4]). Understanding this proof requires a reasonable background in logic and as such this proof is out of reach of many practicing algorithmists. An alternative proof of this theorem has been recently published by Ganian and Hliněný [11] who use an automata-theoretic approach to prove the theorem. Our approach to proving this theorem is game-theoretic, an outline of which follows.

It is known that any graph of rankwidth t can be represented by a t-labeled parse tree [11]. Given any integer q, one can define an equivalence relation on the class of all t-labeled graphs as follows: t-labeled graphs G_1 and G_2 are equivalent, denoted $G_1 \equiv_q^{\mathrm{MSO}} G_2$, iff for every MSO_1-formula of quantifier rank at most q $G_1 \models \varphi$ iff $G_2 \models \varphi$, i.e., no formula with at most q nested quantifiers can distinguish them. The number of equivalence classes depends on the quantifier rank q and the number of labels t and each equivalence class can be represented

⋆ This work is supported by the Deutsche Forschungsgemeinschaft (DFG) under grant RO 927/8. A full version is available at `http://arxiv.org/abs/1102.0908`.

by a tree-like structure of size $f(q,t)$, where f is a computable function of q and t only.

This tree-like representative of an equivalence class, called a *reduced characteristic tree of depth* q and denoted by $\mathrm{RC}_q(G)$, captures all model-checking games that can be played on graphs in that equivalence class and formulas of quantifier rank at most q (see Section 3). One can construct a reduced characteristic tree of depth q given a t-labeled parse tree of an n-vertex graph in time $O(f'(q,t) \cdot n)$ (Section 4). Finally to decide whether $G \models \varphi$, for some MSO_1-formula φ of quantifier rank at most q, we simply simulate the model checking game on φ and G using $\mathrm{RC}_q(G)$. This takes an additional $O(f(q,t))$ time and shows that one can decide whether $G \models \varphi$ in time $O(f''(q,t) \cdot n)$ proving the main theorem:

The Main Theorem ([6, 11]). *Let φ be an MSO_1-formula with $\mathrm{qr}(\varphi) \leq q$. There is an algorithm that takes as input a t-labeled parse tree decomposition T of a graph G and decides whether $G \models \varphi$ in time $O(f(q,t) \cdot |T|)$, where f is some computable function and $|T|$ is the number of nodes in T.*

The notions of q-equivalence $\equiv_q^{\mathrm{MSO}}$ and related two-player pebble games (such as the Ehrenfeucht-Fraïssé game) are fundamental to finite model theory and can be found in any book on the subject (cf. [9]). However for understanding this paper, one does not need any prior knowledge of these concepts.

2 Preliminaries

Rankwidth was originally defined by Oum and Seymour in terms of *branchwidth* [22]. However this definition is not very useful from an algorithmic point-of-view and this prompted Courcelle and Kanté [5] to introduce the notion of bilinear products of multi-colored graphs and algebraic expressions over these products as an equivalent description of rankwidth. Ganian and Hliněný [11] formulated the same ideas in terms of labeling joins and parse trees which we briefly describe here.

t-labeled graphs. A *t-labeling* of a graph G is a mapping $lab\colon V(G) \to 2^{[t]}$ which assigns to each vertex of G a subset of $[t] = \{1, \ldots, t\}$. A *t-labeled graph* is a pair (G, lab), where lab is a labeling of G and is denoted by $\bar{G}$. Since a t-labeling function may assign the empty label to each vertex, an unlabeled graph is considered to be a t-labeled graph for all $t \geq 1$. A t-labeling of G may also be interpreted as a mapping from $V(G)$ to the t-dimensional binary vector space $\mathrm{GF}(2^t)$ by associating the subset $X \subseteq [t]$ with the t-bit vector $\mathbf{x} = x_1 \ldots x_t$, where $x_i = 1$ if and only if $i \in X$. Thus one can represent a t-labeling lab of an n-vertex graph as an $n \times t$ binary matrix.

A *t-relabeling* is a mapping $f\colon [t] \to 2^{[t]}$. One can also view a t-relabeling as a linear transformation from the space $\mathrm{GF}(2^t)$ to itself and one can therefore represent a t-relabeling by a $t \times t$ binary matrix T_f. For a t-labeled graph $\bar{G} = (G, lab)$, we define $f(\bar{G})$ to be the t-labeled graph $(G, f \circ lab)$, where $(f \circ lab)(v)$ is the vector in $\mathrm{GF}(2^t)$ obtained by applying the linear transformation f to the vector $lab(v)$. It is easy to see that the labeling $lab' = f \circ lab$ is the matrix product $lab \times T_f$. Informally, to calculate $(f \circ lab)(v)$, apply the map f to each element of $lab(v)$ and "sum the elements modulo 2".

We now define three operators on t-labeled graphs that will be used to define parse tree decompositions of t-labeled graphs. These operators were first described by Ganian and Hliněný in [11]. The first operator is denoted $\odot$ and represents a nullary operator that creates a new graph vertex with the label 1. The second operator is the t-labeled join and is defined as follows. Let $\bar{G}_1 = (G_1, lab_1)$ and $\bar{G}_2 = (G_2, lab_2)$ be t-labeled graphs. The *t-labeled join* of $\bar{G}_1$ and $\bar{G}_2$, denoted $\bar{G}_1 \otimes \bar{G}_2$, is defined as taking the disjoint union of G_1 and G_2 and adding all edges between vertices $u \in V(G_1)$ and $v \in V(G_2)$ such that $|lab_1(u) \cap lab_2(v)|$ is odd. The resulting graph is unlabeled.

Note that $|lab_1(u) \cap lab_2(v)|$ is odd if and only if the scalar product $lab_1(u) \bullet lab_2(v) = 1$, that is, the vectors $lab_1(u)$ and $lab_2(v)$ are *not orthogonal* in the space $\mathrm{GF}(2^t)$. For $X \subseteq V(G_1)$, the set of vectors $\gamma(\bar{G}_1, X) = \{\, lab_1(u) \mid u \in X \,\}$ generates a subspace $\langle\gamma(\bar{G}_1, X)\rangle$ of $\mathrm{GF}(2^t)$. The following result shows which pair of vertex subsets do not generate edges in a t-labeled join operation.

Proposition 1 ([12]) *Let $X \subseteq V(G_1)$ and $Y \subseteq V(G_2)$ be arbitrary nonempty subsets of t-labeled graphs $\bar{G}_1$ and $\bar{G}_2$. In the join graph $\bar{G}_1 \otimes \bar{G}_2$ there is no edge between any vertex of X and a vertex of Y if and only if the subspaces $\langle\gamma(\bar{G}_1, X)\rangle$ and $\langle\gamma(\bar{G}_2, Y)\rangle$ are orthogonal in the vector space $\mathrm{GF}(2^t)$.*

The third operator is called the t-labeled composition and is defined using the t-labeled join and t-relabelings. Given three t-relabelings $g, f_1, f_2 \colon [t] \to 2^{[t]}$, the *$t$-labeled composition* $\otimes[g|f_1, f_2]$ is defined on a pair of t-labeled graphs $\bar{G}_1 = (G_1, lab_1)$ and $\bar{G}_2 = (G_2, lab_2)$ as follows:

$$\bar{G}_1 \otimes[g|f_1, f_2]\, \bar{G}_2 := \bar{H} = (\bar{G}_1 \otimes g(\bar{G}_2), lab),$$

where $lab(v) = f_i \circ lab_i(v)$ for $v \in V(G_i)$ and $i \in \{1, 2\}$. Thus the t-labeled composition first performs a t-labeling join of $\bar{G}_1$ and $g(\bar{G}_2)$ and then relabels the vertices of G_1 using f_1 and the vertices of G_2 with f_2. Note that a t-labeling composition is not commutative and that $\{u, v\}$ is an edge of $\bar{H}$ if and only if $lab_1(u) \bullet (lab_2(v) \times T_g) = 1$, where T_g is the matrix representing the linear transformation g.

Definition 1 (t-labeled Parse Trees). A *t-labeled parse tree* T is a finite, ordered, rooted subcubic tree (with the root of degree at most two) such that

1. all leaves of T are labeled with the $\odot$ symbol, and
2. all internal nodes of T are labeled with a t-labeled composition symbol.

A parse tree T *generates* the graph G that is obtained by the successive leaves-to-root application of the operators that label the nodes of T.

It is known that rankwidth can be defined using t-labeled parse trees.

Theorem 1 (The Rankwidth Parsing Theorem [5, 11]). *A graph G has rankwidth at most t if and only if some labeling of G can be generated by a t-labeled parse tree. Moreover, a width-k rank-decomposition of an n-vertex graph can be transformed into a t-labeled parse tree on $\Theta(n)$ nodes in time $O(t^2 \cdot n^2)$.*

Monadic second-order logic (MSO) is an extension of first-order logic which allows quantification over sets of objects. We briefly fix notation, for details please refer to [9]. A *vocabulary* τ is a finite set of relation symbols $P, Q, R, \ldots$ each associated with a natural number known as its *arity*. A *τ-structure* $\mathscr{A}$ consists of a set A called the *universe* of $\mathscr{A}$ and a p-ary relation $R^{\mathscr{A}} \subseteq A^p$ for every p-ary relation symbol R in τ. Graphs can be expressed in a natural way as relational structures with universe the vertex set and a vocabulary consisting of a single binary (edge) relation symbol. To express a t-labeled graph G, we may use a vocabulary τ consisting of the binary relation symbol E (representing, as usual, the edge relation) and t unary relation symbols $L_1, \ldots, L_t$, where L_i represents the set of vertices labeled i.

The *quantifier rank* $\mathrm{qr}(\varphi)$ of a *formula* φ is the maximum number of nested quantifiers occurring in it. A variable in a formula is *free* if it is not within the scope of a quantifier. By $\mathrm{free}(\varphi)$ we denote the set of free variables of φ. A formula without free variables is called a *sentence.*

An *assignment* in $\mathscr{A}$ is a function α that assigns *values* to the free variables of φ. For a variable x and an assignment α, we let $\alpha[x/a]$ denote an assignment that agrees with α except that it assigns the value a to x. We write $\mathscr{A} \models \varphi[\alpha]$ if φ *holds* in $\mathscr{A}$, when the free variables of φ have been assigned the values given by α.

3 The $\equiv_q^{\mathrm{MSO}}$-Relation and its Characterization

Given a vocabulary τ and a natural number q, one can define an equivalence relation on the class of τ-structures as follows. For τ-structures $\mathscr{A}$ and $\mathscr{B}$ and $q \in \mathbf{N}$, define $\mathscr{A} \equiv_q^{\mathrm{MSO}} \mathscr{B}$ (q-equivalence) if and only if $\mathscr{A} \models \varphi \iff \mathscr{B} \models \varphi$ for all MSO sentences φ of quantifier rank at most q. In other words, two structures are q-equivalent if and only if no sentence of quantifier rank at most q can distinguish them. We provide a characterization of the relation $\equiv_q^{\mathrm{MSO}}$ using objects called characteristic trees of depth q. We show that two τ-structures $\mathscr{A}$ and $\mathscr{B}$ have identical characteristic trees of depth q if and only if $\mathscr{A} \equiv_q^{\mathrm{MSO}} \mathscr{B}$. We shall see that characteristic trees are specially useful because their size is "small" and for graphs of bounded rankwidth can be constructed efficiently given their parse tree decomposition. However before we can do that, we need a few definitions.

Definition 2 (Induced Structure and Sequence). Let $\mathscr{A}$ a τ-structure with universe A and let $\bar{c} = c_1, \ldots, c_m \in A^m$. The structure $\mathscr{A}' = \mathscr{A}[\bar{c}] = \mathscr{A}[\{c_1, \ldots, c_m\}]$ induced by $\bar{c}$ is a τ-structure with universe $A' = \{c_1, \ldots, c_m\}$ and interpretations $P^{\mathscr{A}'} := P^{\mathscr{A}} \cap \{c_1, \ldots, c_m\}^r$ for every relation symbol $P \in \tau$ of arity r. For an arbitrary sequence of objects $\bar{c} = c_1, \ldots, c_m$ and a set U, we let $\bar{c}[U]$ be the subsequence of $\bar{c}$ that contains only objects in U. For a sequence of sets $\bar{C} = C_1, \ldots, C_p$ we let $\bar{C} \cap A$ denote the sequence $C_1 \cap U, \ldots, C_p \cap U$ and write $\bar{C} \cap \bar{c}$ for $C_1 \cap \{c_1, \ldots, c_m\}, \ldots, C_p \cap \{c_1, \ldots, c_m\}$.

Definition 3 (Partial Isomorphism). Let $\mathscr{A}$ and $\mathscr{B}$ be structures over the vocabulary τ with universes A and B, respectively, and let π be a map such that domain$(\pi) \subseteq A$ and range$(\pi) \subseteq B$. The map π is said to be a *partial isomorphism* from $\mathscr{A}$ to $\mathscr{B}$ if (1) π is one-to-one and onto; and (2) for every p-ary relation symbol $R \in \tau$ and all $a_1, \ldots, a_p \in$ domain(π), $R^{\mathscr{A}} a_1, \ldots, a_p$ iff $R^{\mathscr{B}} \pi(a_1), \ldots, \pi(a_p)$. If domain$(\pi) = A$ and range$(\pi) = B$, then π is an *isomorphism* between $\mathscr{A}$ and $\mathscr{B}$ and $\mathscr{A}$ and $\mathscr{B}$ are *isomorphic*.

Let $(\mathscr{A}, \bar{A})$ and $(\mathscr{B}, \bar{B})$ be tuples, where $\bar{A} = A_1, \ldots, A_s$ and $\bar{B} = B_1, \ldots, B_s$, $s \geq 0$, such that for all $1 \leq i \leq s$, we have $A_i \subseteq A$ and $B_i \subseteq B$. We say that π is a partial isomorphism between $(\mathscr{A}, \bar{A})$ and $(\mathscr{B}, \bar{B})$ if (1) π is a partial isomorphism between $\mathscr{A}$ and $\mathscr{B}$, and (2) for each $a \in$ domain(π) and all $1 \leq i \leq s$, it holds that $a \in A_i$ iff $\pi(a) \in B_i$. The tuples $(\mathscr{A}, \bar{A})$ and $(\mathscr{B}, \bar{B})$ are *isomorphic* if π is an isomorphism between $\mathscr{A}$ and $\mathscr{B}$.

In Definition 2 of an induced structure we ignore the order of the elements in $\bar{c}$. For the purposes in this paper, the order in which the elements are chosen is important because it is used to map variables in the formula to elements in the structure. Moreover, elements could repeat in the vector $\bar{c}$ and this fact is lost when we consider the induced structure $\mathscr{A}[\bar{c}]$. To capture both the order and the multiplicity of the elements in vector $\bar{c}$ in the structure $\mathscr{A}[\bar{c}]$, we introduce the notion of an *ordered induced structure*.

Let U be a set and $\equiv$ be an equivalence relation on U. For $u \in U$, we let $[u]_{\equiv} = \{\, u' \in U \mid u \equiv u' \,\}$ be the *equivalence class* of u under $\equiv$, and $U/{\equiv} = \{\, [u]_{\equiv} \mid u \in U \}$ be the *quotient space* of U under $\equiv$. A vector $\bar{c} = c_1, \ldots, c_m \in A^m$ defines a natural equivalence relation $\equiv_{\bar{c}}$ on the set $[m] = \{1, \ldots, m\}$: for $i, j \in [m]$, we have $i \equiv_{\bar{c}} j$ if and only if $c_i = c_j$. For simplicity, we write $[i]_{\bar{c}}$ for $[i]_{\equiv_{\bar{c}}}$.

Definition 4 (Ordered Induced Structure). Let $\mathscr{A}$ be a τ-structure and $\bar{c} = c_1, \ldots, c_m \in A^m$. The *ordered structure induced by* $\bar{c}$ is the τ-structure $\mathscr{H} = \text{Ord}(\mathscr{A}, \bar{c})$ with universe $H = [m]/{\equiv_{\bar{c}}}$ such that the map $h\colon c_i \mapsto [i]_{\bar{c}}$, $1 \leq i \leq m$, is an isomorphism between $\mathscr{A}[\bar{c}]$ and $\mathscr{H}$. Thus $\text{Ord}(\mathscr{A}, \bar{c})$ is simply the structure $\mathscr{A}[\bar{c}]$ with element c_i being called $[i]_{\bar{c}}$. Let $\bar{C} = C_1, \ldots, C_p$ with $C_i \subseteq A$, $1 \leq i \leq p$. Then we let $\text{Ord}(\mathscr{A}, \bar{c}, \bar{C}) := \big(\text{Ord}(\mathscr{A}, \bar{c}), \bar{h}, h(\bar{C} \cap \bar{c})\big)$, where $h\colon c_i \mapsto [i]_{\bar{c}}$, $1 \leq i \leq m$, $\bar{h} = h(1), \ldots, h(m)$ and $h(\bar{C} \cap \bar{c}) = h(C_1 \cap \bar{c}), \ldots, h(C_p \cap \bar{c})$. See Figure 1.

3.1 Model Checking Games and Characteristic Trees

Testing whether a non-empty structure models a formula can be specified by a *model checking game* (also known as *Hintikka game*, see [16, 13]). Let $\mathscr{A}$ be a τ-structure with universe A. Let φ be a formula and α be an assignment to the free variables of φ. The game is played between two players called the *verifier* and the *falsifier*. The verifier tries to prove that $\mathscr{A} \models \varphi[\alpha]$ whereas the falsifier

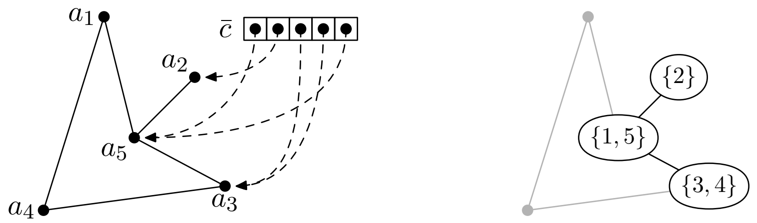

Fig. 1. The vector $\bar{c} = a_5a_2a_3a_3a_5$ lists vertices in the graph $\mathscr{G}$ on the left. The resulting ordered induced structure $\mathrm{Ord}(\mathscr{G}, \bar{c})$ is depicted in black on the right.

tries to disprove this. We assume without loss of generality that φ is in negation normal form, i.e., negations in φ appear only at the atomic level. This can always be achieved by applying simple rewriting rules such as $\neg\forall x\varphi(x) \rightsquigarrow \exists x\neg\varphi(x)$. The model checking game $\mathcal{MC}(\mathscr{A}, \varphi, \alpha)$ is positional with positions (ψ, β), where ψ is a subformula of φ and β is an assignment to the free variables of ψ. The game starts at position (φ, α). At a position $(\forall X\psi(X), \beta)$, the falsifier chooses a subset $D \subseteq A$, and the game continues at position $(\psi, \beta[X/D])$. Similarly, at a position $(\forall x\psi(x), \beta)$ or $(\psi_1 \wedge \psi_2, \beta)$, the falsifier chooses an element $d \in A$ or some $\psi := \psi_i$ for some $1 \leq i \leq 2$ and the game then continues at position $(\psi, \beta[x/d])$ or (ψ, β), respectively. The verifier moves analogously at existential formulas. If an element is chosen then the move is called a *point move*; if a set is chosen then the move is a *set move*. The game ends once a position (ψ, β) is reached, such that ψ is an atomic or negated formula. The verifier *wins* if and only if $\mathscr{A} \models \psi[\beta]$. We say that the verifier has a *winning strategy* if they win every play of the game irrespective of the choices made by the falsifier. It is well known that the model checking game characterizes the satisfaction relation $\models$. The following lemma can easily be shown by induction over the structure of φ.

Lemma 1 (cf., [13]). *Let $\mathscr{A}$ be a τ-structure, let φ be an* MSO *formula, and let α be an assignment to the free variables of φ. Then $\mathscr{A} \models \varphi[\alpha]$ if and only if the verifier has a winning strategy on the model checking game on $\mathscr{A}$, φ, and α.*

A model checking game on a τ-structure $\mathscr{A}$ and a formula φ with quantifier rank q can be represented by a tree of depth q in which the nodes represent positions in the game and the edges represent point and set moves made by the players. Such a tree is called a *game tree* and is used in combinatorial game theory for analyzing games (see [2], for instance). For our purposes, we define a notion related to game trees called *full characteristic trees* which are finite rooted trees, where the nodes represent positions and edges represent moves of the game. A node is a tuple that represents the sets and elements that have been chosen thus far. The node can be thought of as a succinct representation of the state of the game played till the position represented by that node. However, note that a full characteristic tree depends on the quantifier rank q and *not* on a particular formula.

Definition 5 (Full Characteristic Trees). Let $\mathscr{A}$ be a τ-structure with universe A and let $q \in \mathbf{N}$. For elements $\bar{c} = c_1, \dots, c_m \in A^m$, sets $\bar{C} = C_1, \dots, C_p$ with $C_i \subseteq A$, $1 \le i \le p$, let $T = \mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C})$ be a finite rooted tree such that (1) $\mathrm{root}(T) = (\mathscr{A}[\bar{c}], \bar{c}, \bar{C} \cap \bar{c})$, and (2) if $m+p+1 \le q$ then the subtrees of the root of $\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C})$ is the set $\left\{ \mathrm{FC}_q(\mathscr{A}, \bar{c}d, \bar{C}) \;\middle|\; d \in A \right\} \cup \left\{ \mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C}D) \;\middle|\; D \subseteq A \right\}$. The *full characteristic tree of depth* q for $\mathscr{A}$, denoted by $\mathrm{FC}_q(\mathscr{A})$, is defined as $\mathrm{FC}_q(\mathscr{A}, \varepsilon, \varepsilon)$, where ε is the empty sequence.

Let $T = (V, E)$ be a rooted tree. We let $\mathrm{root}(T)$ be the root of T and for $u \in V$ we let $\mathrm{children}_T(u) = \{ v \in V \mid (u, v) \in E \}$ and $\mathrm{subtree}_T(u)$ be a subtree of T rooted at u, and $\mathrm{subtrees}(T) = \{ \mathrm{subtree}_T(u) \mid u \in \mathrm{children}_T(\mathrm{root}(T)) \}$.

We now define a model checking game $\mathcal{MC}(F, \varphi, \bar{x}, \bar{X})$ on full characteristic trees $F = \mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C})$ and formulas φ with $\mathrm{qr}(\varphi) \le q$, where $\bar{x} = x_1, \dots, x_m$ are the free object variables of φ, $\bar{X} = X_1, \dots, X_p$ are the free set variables of φ, $\bar{c} = c_1, \dots, c_m \in A^m$, and $\bar{C} = C_1, \dots, C_p$ with $C_i \subseteq A$, $1 \le i \le p$. The rules are similar to the classical model checking game $\mathcal{MC}(\mathscr{A}, \varphi, \alpha)$. The game is positional and played by two players called the *verifier* and the *falsifier* and is defined over subformulas ψ of φ. However instead of choosing sets and elements explicitly, the tree F is traversed top-down. At the same time, we "collect" the list of variables the players encountered, such that we can make the assignment explicit once the game ends. The game starts at the position $(\varphi, \bar{x}, \bar{X}, \mathrm{root}(F))$. Let $(\psi, \bar{y}, \bar{Y}, v)$ be the position at which the game is being played, where $v = (\mathscr{H}, \bar{d}, \bar{D})$ is a node of $\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C})$, and ψ is a subformula of φ with $\mathrm{free}(\psi) = \bar{y} \cup \bar{Y}$. At a position $(\forall X \vartheta(X), \bar{y}, \bar{Y}, v)$ the falsifier chooses a child $u = (\mathscr{H}, \bar{d}, \bar{D}D)$ of v, where $D \subseteq A$, and the game continues at position $(\vartheta, \bar{y}, \bar{Y}X, u)$. Similarly, at a position $(\forall x \vartheta(x), \bar{y}, \bar{Y}, v)$ the falsifier chooses a child $u = (\mathscr{H}', \bar{d}d, \bar{D})$, where $d \in A$, and the game continues in $(\vartheta, \bar{y}x, \bar{Y}, u)$, and at a position $(\vartheta_1 \wedge \vartheta_2, \bar{y}, \bar{Y}, v)$, the falsifier chooses some $1 \le i \le 2$, and the game continues at position $(\vartheta_i, \bar{y}, \bar{Y}, v)$. The verifier moves analogously at existential formulas.

The game stops once an atomic or negated formula has been reached. Suppose that a particular play of the game ends at a position $(\psi, \bar{y}, \bar{Y}, v)$, where ψ is a negated atomic or atomic formula with $\mathrm{free}(\psi) = \{y_1, \dots, y_s, Y_1, \dots, Y_t\}$ and $v = (\mathscr{H}, \bar{d}, \bar{D})$ some node of F, where $\bar{d} = d_1, \dots, d_s$ and $\bar{D} = D_1, \dots, D_t$. Let α be an assignment to the free variables of φ, such that $\alpha(y_i) = d_i$, $1 \le i \le s$, and $\alpha(Y_i) = D_i$, $1 \le i \le t$. The verifier *wins* the game if and only if $\mathscr{H} \models \psi[\alpha]$. The verifier has a *winning strategy* if and only if they can win every play of the game irrespective of the choices made by the falsifier. In what follows, we identify a position $(\psi, \bar{y}, \bar{Y}, v)$ of the game $\mathcal{MC}(\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C}), \varphi, \bar{x}, \bar{X})$, where $v = (\mathscr{H}, \bar{d}, \bar{D})$, with the game $\mathcal{MC}(\mathrm{FC}_q(\mathscr{A}, \bar{d}, \bar{D}), \psi, \bar{y}, \bar{Y})$.

Lemma 2. *Let $\mathscr{A}$ be a τ-structure and let φ be an* MSO *formula with* $\mathrm{qr}(\varphi) \le q$ *and free variables $\{x_1, \dots, x_m, X_1, \dots, X_m\}$. Let α be an assignment to the free variables of φ. Then the verifier has a winning strategy in the model checking game $\mathcal{MC}(\mathscr{A}, \varphi, \alpha)$ if and only if the verifier has a winning strategy in the model checking game $\mathcal{MC}(\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C}), \varphi, \bar{x}, \bar{X})$, where $\bar{c} = \alpha(x_1), \dots, \alpha(x_m)$ and $\bar{C} = \alpha(X_1), \dots, \alpha(X_p)$.*

Lemma 2 is shown by simulating each play of the model checking game $\mathcal{MC}(\mathscr{A}, \varphi, \alpha)$ in $\mathcal{MC}(\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C}), \varphi, \bar{x}, \bar{X})$ and vice versa. Therefore, a full characteristic tree of depth q for a structure $\mathscr{A}$ can be used to simulate the model checking game on $\mathscr{A}$ and any formula φ of quantifier rank at most q. However the size of such a tree is of the order $(2^n + n)^q$, where n is the number of elements in the universe of $\mathscr{A}$. We now show that one can "collapse" equivalent branches of a full characteristic tree to obtain a much smaller labeled tree (called a reduced characteristic tree) that is in some sense equivalent to the original (full) tree. We will then show that for a graph G of rankwidth at most t, the reduced characteristic tree of G is efficiently computable given a t-labeled parse tree decomposition of G. We achieve this collapse by replacing the induced structures $\mathscr{A}[\bar{c}]$ in the full characteristic tree by a more generic, implicit representation — that of their ordered induced substructures $\mathrm{Ord}(\mathscr{A}, \bar{c})$.

Definition 6 (Reduced Characteristic Trees). Let $\mathscr{A}$ be a τ-structure and let $q \in \mathbf{N}$. For elements $\bar{c} = c_1, \ldots, c_m \in A^m$, sets $\bar{C} = C_1, \ldots, C_p$ with $C_i \subseteq A$, $1 \le i \le p$, we let $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ be a finite rooted tree such that (1) the root of $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C}))$ is $\mathrm{Ord}(\mathscr{A}, \bar{c}, \bar{C})$, and (2) if $m + p + 1 \le q$ then the subtrees of the root of $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ is the set $\{\, \mathrm{RC}_q(\mathscr{A}, \bar{c}d, \bar{C}) \mid d \in A \,\} \cup \{\, \mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C}D) \mid D \subseteq A \,\}$. The *reduced characteristic tree of depth q* for the structure $\mathscr{A}$, denoted by $\mathrm{RC}_q(\mathscr{A})$, is defined to be $\mathrm{RC}_q(\mathscr{A}, \varepsilon, \varepsilon)$, where ε is the empty sequence.

One can define the model checking game $\mathcal{MC}(R, \varphi, \bar{x}, \bar{X})$ on a tree $R = \mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ in exactly the same manner as $\mathcal{MC}(\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C}), \varphi, \bar{x}, \bar{X})$. As mentioned before, our interest in $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ lies in that: (1) they are equivalent to $\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C})$; (2) they are "small"; and, (3) they are efficiently computable if $\mathscr{A}$ is a graph of rankwidth at most t. We first show that $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ is equivalent to its full counterpart $\mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C})$.

Lemma 3. *Let $\mathscr{A}$ be a τ-structure and let $q \in \mathbf{N}$. Let $\bar{c} = c_1, \ldots, c_m \in A^m$ and $\bar{C} = C_1, \ldots, C_p$ with $C_i \subseteq A$, $1 \le i \le p$. Let $F = \mathrm{FC}_q(\mathscr{A}, \bar{c}, \bar{C})$ and $R = \mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$. Then the verifier has a winning strategy in the model checking game $\mathcal{MC}(F, \varphi, \bar{x}, \bar{X})$ if and only if the verifier has a winning strategy in the game $\mathcal{MC}(R, \varphi, \bar{x}, \bar{X})$, where $\varphi \in \mathrm{MSO}(\tau)$ with $\mathrm{qr}(\varphi) \le q$ with free object variables $\bar{x} = x_1, \ldots, x_m$ and free set variables $\bar{X} = X_1, \ldots, X_p$.*

From Lemmas 1, 2, and 3, we obtain the important fact that reduced characteristic trees are in fact equivalent to their full counterparts and characterize the equivalence relation $\equiv_q^{\mathrm{MSO}}$.

Corollary 1. *Let $\mathscr{A}$ and $\mathscr{B}$ be τ-structures and $q \in \mathbf{N}$. Then $\mathrm{RC}_q(\mathscr{A}) = \mathrm{RC}_q(\mathscr{B})$ iff $\mathscr{A} \equiv_q^{\mathrm{MSO}} \mathscr{B}$.*

The next lemma shows that reduced characteristic trees have small size. For $i \in \mathbf{N}$, we define $\exp^{(i)}(\cdot)$ as: $\exp^{(0)}(x) = x$, $\exp^{(1)}(x) = 2^x$ and $\exp^{(i)}(x) = 2^{2\exp^{(i-1)}(x)}$ for $i \ge 2$.

Lemma 4. *Let $\mathscr{A}$ be a τ-structure with universe A such that each relation symbol in τ has arity at most r, and $q \in \mathbf{N}$. Then the number of reduced characteristic* *trees* $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ *for all possible choices of* $\bar{c}, \bar{C}$ *is at most* $\exp^{(q+1)}(|\tau| \cdot q^r + q \log q + q^2)$*. The size of a reduced characteristic tree* $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ *is at most* $(\exp^{(q)}(|\tau| \cdot q^r + q \log q + q^2))^4$.

4 Constructing Characteristic Trees

In this section, we show how to construct reduced characteristic trees of depth q for a graph G of rankwidth t when given a t-labeled parse tree decomposition of G. A t-labeled graph may be represented as τ-structure where $\tau = \{E, L_1, \ldots, L_t\}$. The symbol E is a binary relation symbol representing the edge relation and L_i for $1 \le i \le t$ is a unary relation symbol representing the set of vertices with label i. In what follows, whenever we talk about a τ-structure $\mathscr{A}$, we mean a graph viewed as a structure over the vocabulary $\{E, L_1, \ldots, L_t\}$.

Lemma 5. *Let* $\mathscr{A}$ *be a* τ*-structure with* $|A| = 1$*. Let* $q \ge 0$ *and* $\bar{c} \in A^m$ *and* $\bar{C} = C_1, \ldots, C_p$ *with* $C_i \subseteq A$*,* $1 \le i \le p$*. Then* $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ *can be constructed in constant time for each fixed* q*.*

In what follows, we let $\mathscr{A}_1, \mathscr{A}_2$ and $\mathscr{A} = \mathscr{A}_1 \otimes \mathscr{A}_2$ be τ-structures, where $\otimes = \otimes[g|f_1, f_2]$ for t-relabelings g, f_1, and f_2. Recall that if $\mathscr{A} = \mathscr{A}_1 \otimes \mathscr{A}_2$, then we assume that A_1 and A_2 (the universes of $\mathscr{A}_1$ and $\mathscr{A}_2$, respectively) are disjoint. Furthermore for a fixed constant $q \ge 0$, let m and p be nonnegative integers such that $m + p \le q$, $\bar{c} = c_1, \ldots, c_m \in (A_1 \cup A_2)^m$ and $\bar{C} = C_1, \ldots, C_p$, where $C_j \subseteq A_1 \cup A_2$, $1 \le j \le p$. For $i \in \{1, 2\}$, we let $\bar{c}_i = c_{i,1}, \ldots, c_{i,m_i} = \bar{c}[A_i]$.

In the remainder of this section, we show how to construct $\mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ given $\mathrm{RC}_q(\mathscr{A}_1, \bar{c}_1, \bar{C} \cap \bar{c}_1)$ and $\mathrm{RC}_q(\mathscr{A}_2, \bar{c}_2, \bar{C} \cap \bar{c}_2)$. For the construction, we need to know the order in which the elements in $\bar{c}_1$ and $\bar{c}_2$ appear in $\bar{c}$. This motivates us to define the notion of an *indicator vector* $\mathrm{ind}(A_1, A_2, \bar{c})$.

Definition 7. The *indicator vector* of $\bar{c} = c_1, \ldots, c_m$, denoted $\mathrm{ind}(A_1, A_2, \bar{c})$, is the vector $\bar{d} = d_1, \ldots, d_m$, such that for $i \in \{1, 2\}$ and all $1 \le j \le m$ it holds that $d_j = (i, k)$ iff $c_j = c_{i,k}$. That is, $d_j = (i, k)$ iff c_j is the kth element in the vector $\bar{c}_i = \bar{c}[A_i]$. If $\bar{d} = d_1, \ldots, d_m$ and $(i, k) \in \{1, 2\} \times [m + 1]$, then we use $\bar{d}(i, k)$ to denote the vector $d_1, \ldots, d_{m+1}$, where $d_{m+1} = (i, k)$.

Constructing $R = \mathrm{RC}_q(\mathscr{A}, \bar{c}, \bar{C})$ when given $R_1 = \mathrm{RC}_q(\mathscr{A}_1, \bar{c}_1, \bar{C} \cap \bar{c}_1)$, $R_2 = \mathrm{RC}_q(\mathscr{A}_2, \bar{c}_2, \bar{C} \cap \bar{c}_2)$, and $\bar{d} = \mathrm{ind}(A_1, A_2, \bar{c})$ consists of the following two steps: First construct the label for $\mathrm{root}(R) = \mathrm{Ord}(\mathscr{A}, \bar{c}, \bar{C})$, and then recursively construct its subtrees. Since $\mathrm{Ord}(\mathscr{A}, \bar{c}) \cong \mathscr{A}[\bar{c}]$ and $\mathscr{A}_i[\bar{c}_i] \cong \mathrm{Ord}(\mathscr{A}_i, \bar{c}_i)$, one easily sees that $\mathrm{Ord}(\mathscr{A}, \bar{c}) \cong \mathrm{Ord}(\mathscr{A}_1, \bar{c}_1) \otimes \mathrm{Ord}(\mathscr{A}_2, \bar{c}_2)$. For the first step, we therefore just need to rename elements in $\mathrm{Ord}(\mathscr{A}_1, \bar{c}_1) \otimes \mathrm{Ord}(\mathscr{A}_2, \bar{c}_2)$ in an appropriate way. The information on how elements are to be renamed is stored in the indicator vector $\bar{d}$ of $\bar{c}$. The formal definition of the renaming operator $\otimes_{\bar{d}}$ and Lemma 6 are technical and may be skipped if the reader believes that one can construct $\mathrm{Ord}(\mathscr{A}, \bar{c})$ from $\mathrm{Ord}(\mathscr{A}_1, \bar{c}_1)$ and $\mathrm{Ord}(\mathscr{A}_2, \bar{c}_2)$ using $\bar{d}$.

Definition 8. For $i \in \{1,2\}$, let $\mathrm{Ord}(A_i, \bar{c}_i, \bar{C} \cap A_i) = (\mathscr{H}_i, \bar{c}'_i, \bar{C}'_i)$. Define a map $f\colon [m] \to H_1 \uplus H_2$ as follows: for all $1 \le j \le m$, let $f(j) = [k]_{\bar{c}_i}$ iff $d_j = (i,k)$. Then we define $\mathrm{Ord}(\mathscr{A}_1, \bar{c}[A_1], \bar{C} \cap A_1) \otimes_{\bar{d}} \mathrm{Ord}(\mathscr{A}_2, \bar{c}[A_2], \bar{C} \cap A_2)$ as $\mathrm{Ord}(\mathscr{H}_1 \otimes \mathscr{H}_2, f(1) \dots f(m), \bar{C}'_1 \cup \bar{C}'_2)$.

Lemma 6. *Let $\mathscr{A}_1$ and $\mathscr{A}_2$ be τ-structures and let $\otimes =\otimes[g|f_1,f_2]$ for some t-relabelings g, f_1, f_2. Let $\bar{c} = c_1, \dots, c_m \in (A_1 \cup A_2)^m$ and $\bar{C} = C_1, \dots, C_p$, where $C_j \subseteq A_1 \cup A_2$ for $1 \le j \le p$. Also let $\bar{d} = \mathrm{ind}(A_1, A_2, \bar{c})$. Then $\mathrm{Ord}(\mathscr{A}_1 \otimes \mathscr{A}_2, \bar{c}, \bar{C}) = \mathrm{Ord}(\mathscr{A}_1, \bar{c}[A_1], \bar{C} \cap A_1) \otimes_{\bar{d}} \mathrm{Ord}(\mathscr{A}_2, \bar{c}[A_2], \bar{C} \cap A_2)$.*

We now define the *tree cross product* $R_1 \times(q, \otimes, \bar{d})\ R_2$ of R_1 and R_2 and then show that in fact $R = R_1 \times(q, \otimes, \bar{d})\ R_2$. As motivated before, the root of the tree cross product is simply $\mathrm{root}(R_1) \otimes_{\bar{d}} \mathrm{root}(R_2)$. For the construction of the subtrees, recall that each subtree of R corresponds to either a set move $U \subseteq A$ or a point move $a \in A$. Here, $\{\, U \subseteq A \,\} = \{\, U_1 \uplus U_2 \mid U_1 \subseteq A_1, U_2 \subseteq A_2 \,\}$ and $A = A_1 \uplus A_2$. We can therefore reconstruct the subtrees of R by recursively combining each subtree for a set $U_1 \subseteq A_1$ with a subtree for a set $U_2 \subseteq A_2$ (the set S_2 in the following definition), and by choosing subtrees of R_1 for point moves in A_1, and choosing subtrees of R_2 for point moves in A_2 (the set S_1 in the following definition).

Definition 9 (Tree Cross Product). Let $\mathscr{A}_1$ and $\mathscr{A}_2$ be τ-structures and let $\otimes = \otimes[g|f_1,f_2]$ for some t-relabelings g, f_1, f_2. For a fixed constant $q \ge 0$, let m and p be nonnegative integers such that $m + p \le q$. Let $\bar{c} = c_1, \dots, c_m \in (A_1 \cup A_2)^m$ and $\bar{C} = C_1, \dots, C_p$, where $C_j \subseteq A_1 \cup A_2$, $1 \le j \le p$. For $i \in \{1,2\}$, let $\bar{c}_i = c_{i,1}, \dots, c_{i,m_i} = \bar{c}[A_i]$, $q_i \ge q - m - p$, and $R_i = \mathrm{RC}_{q_i}(\mathscr{A}_i, \bar{c}_i, \bar{C} \cap A_i)$ with $\mathrm{root}(R_i) = (\mathscr{H}_i, \bar{c}'_i, \bar{C}'_i) = \mathrm{Ord}(A_i, \bar{c}_i, \bar{C} \cap A_i)$. We define the *tree cross product* of R_1 and R_2, $R = R_1 \times(q, \otimes, \bar{d})\ R_2$, be a finite, rooted tree such that (1) $\mathrm{root}(R) = \mathrm{root}(R_1) \otimes_{\bar{d}} \mathrm{root}(R_2)$, and (2) if $m + p + 1 \le q$, then $\mathrm{subtrees}(R) = S_1 \cup S_2$, where

$$
\begin{aligned}
S_1 = \big\{\, & \mathrm{subtree}_{R_1}(u_1) \times(q, \otimes, \bar{d}(1, m_1+1))\ R_2 \mid \\
& \qquad u_1 = (\mathscr{H}'_1, \bar{c}'_1 c, \bar{C}'_1) \in \mathrm{children}_{R_1}(\mathrm{root}(R_1)) \,\big\} \cup \\
\big\{\, & R_1 \times(q, \otimes, \bar{d}(2, m_2+1))\ \mathrm{subtree}_{R_2}(u_2) \mid \\
& \qquad u_2 = (\mathscr{H}'_2, \bar{c}'_2 c, \bar{C}'_2) \in \mathrm{children}_{R_2}(\mathrm{root}(R_2)) \,\big\}, \\
S_2 = \big\{\, & \mathrm{subtree}_{R_1}(u_1) \times(q, \otimes, \bar{d})\ \mathrm{subtree}_{R_2}(u_2) \mid \\
& \qquad u_i = (\mathscr{H}'_i, \bar{c}'_i, \bar{C}'_i D_i) \in \mathrm{children}_{R_i}(\mathrm{root}(R_i)), 1 \le i \le 2 \,\big\}.
\end{aligned}
$$

Lemma 7. *Let $\mathscr{A}_1$ and $\mathscr{A}_2$ be τ-structures and let $\otimes = \otimes[g|f_1,f_2]$ for some t-relabelings g, f_1, f_2. For nonnegative integers q, m, p with $m + p \le q$, let $\bar{c} = c_1, \dots, c_m \in (A_1 \cup A_2)^m$ and $\bar{C} = C_1, \dots, C_p$, where $C_j \subseteq A_1 \cup A_2$ for $1 \le j \le p$. Also let $\bar{d} = \mathrm{ind}(A_1, A_2, \bar{c})$ and for $1 \le i \le 2$ let $q_i \ge q - m - p$. Then $\mathrm{RC}_q(\mathscr{A}_1 \otimes \mathscr{A}_2, \bar{c}, \bar{C}) = \mathrm{RC}_{q_1}(\mathscr{A}_1, \bar{c}_1, \bar{C} \cap A_1) \times(q, \otimes, \bar{d})\ \mathrm{RC}_{q_2}(\mathscr{A}_2, \bar{c}_2, \bar{C} \cap A_2)$.*

Lemma 8. *Given R_1 and R_2, the tree cross product $R_1 \times(q, \otimes, \bar{d})\ R_2$ can be computed time poly$(|R_1|, |R_2|)$, where $|R_i|$ denotes the number of nodes in R_i.*

We can now finally prove the Main Theorem.

Proof (Main Theorem). It is no loss of generality to assume that G has at least one vertex. Otherwise deciding whether $G \models \varphi$ takes constant time. By Lemmas 1, 2 and 3, to prove that $G \models \varphi$ it is sufficient to show that the verifier has a winning strategy in the model checking game $\mathcal{MC}(\mathrm{RC}_q(G), \varphi, \epsilon, \epsilon)$. By Lemma 4, the size of the reduced characteristic tree $\mathrm{RC}_q(G)$ of a t-labeled graph is at most $f_1(q,t)$ for some computable function f_1 of q and t alone. By Lemma 8, the time taken to combine two reduced characteristic trees of size $f_1(q,t)$ is $f(q,t) = \mathrm{poly}(f_1(q,t))$.

We claim that the total time taken to construct $\mathrm{RC}_q(G)$ from its parse tree decomposition T is $O(f(q,t) \cdot |T|)$. This is where we use the fact that the graph G has rankwidth at most t. The proof is by induction on $|T|$. By Lemma 5, the claim holds when $|T| = 1$. Suppose that $\bar{G} = \bar{G}_1 \otimes[g|h_1, h_2]\ \bar{G}_2$, where g, h_1, h_2 are t-relabelings and let T_1 and T_2 be parse trees of $\bar{G}_1$ and $\bar{G}_2$, respectively. Then $|T| = |T_1| + |T_2| + 1$, where T is a parse tree of $\bar{G}$. By induction hypothesis, one can construct the reduced characteristic trees $\mathrm{RC}_q(G_1)$ and $\mathrm{RC}_q(G_2)$ in times $O(f(q,t) \cdot |T_1|)$ and $O(f(q,t) \cdot |T_2|)$, respectively. By Lemma 7, one can indeed construct $\mathrm{RC}_q(G)$ given $\mathrm{RC}_q(G_1)$, $\mathrm{RC}_q(G_2)$ and $\bar{d} = \varepsilon$. By using Lemma 8, the time taken to construct $\mathrm{RC}_q(G)$ is $O(f(q,t) + f(q,t) \cdot |T_1| + f(q,t) \cdot |T_2|) = O(f(q,t) \cdot |T|)$, proving the claim.

In order to check whether the verifier has a winning strategy in the model checking game $\mathcal{MC}(\mathrm{RC}_q(G), \varphi, \epsilon, \epsilon)$, one can use a very simple recursive algorithm (see also [13]). A position $p = (\psi, \bar{x}, \bar{X}, u)$ of the model checking game can be identified with a call of the algorithm with arguments p. If ψ is universal, then the algorithm recursively checks whether the verifier has a winning strategy from all positions u' that are reachable from u in the model checking game. If otherwise ψ is existential, then the algorithm checks whether there is one subsequent position in the game from which the verifier has a winning strategy. This algorithm visits each node of the reduced characteristic tree $\mathrm{RC}_q(G)$ at most once. Therefore the time taken to decide whether $G \models \varphi$ is $O(f_1(q,t) + f(q,t) \cdot |T|) = O(f(q,t) \cdot |T|)$, as claimed. □

5 Discussion and Conclusion

With some additional effort the proof of the Main Theorem can be extended to linear optimization problems expressible in MSO_1 (the *LinMSO*-framework). Moreover the results of this paper naturally extend to directed graphs and birankwidth. This allows us to conclude that any decision or optimization problem on directed graphs expressible in MSO_1 is linear-time solvable on graphs of bounded birankwidth [6, 18]. Finally, the game-theoretic approach has already been used to prove Courcelle's result for treewidth [3, 1, 8] with an emphasis on practical implementability [19].

References

1. S. Arnborg, J. Lagergren, and D. Seese. Easy problems for tree-decomposable graphs. *J. Algorithms*, 12(2):308–340, 1991.
2. E. R. Berlekamp, J. H. Conway, and R. K. Guy. *Winning Ways for Your Mathematical Plays.* A.K. Peters, 1982.
3. B. Courcelle. The monadic second order theory of Graphs I: Recognisable sets of finite graphs. *Information and Computation*, 85:12–75, 1990.
4. B. Courcelle. Monadic second-order definable graph transductions: A survey. *Theor. Comput. Sci.*, 126(1):53–75, 1994.
5. B. Courcelle and M. M. Kante. Graph operations characterizing rank-width and balanced graph expressions. In *Proc. of WG*, number 4769 in Lecture Notes in Computer Science, pages 66–75. Springer, 2007.
6. B. Courcelle, J. A. Makowsky, and U. Rotics. Linear Time Solvable Optimization Problems on Graphs of Bounded Clique Width. *Theory Comput. Syst.*, 33:125–150, 2000.
7. B. Courcelle, J. A. Makowsky, and U. Rotics. On the fixed parameter complexity of graph enumeration problems definable in monadic second-order logic. *Discrete Applied Mathematics*, 108(1-2):23–52, 2001.
8. B. Courcelle and M. Mosbah. Monadic second-order evaluations on tree-decomposable graphs. *Theor. Comput. Sci.*, 109(1-2):49–82, 1993.
9. H.-D. Ebbinghaus and J. Flum. *Finite Model Theory.* Springer, 1999.
10. S. Feferman and R. Vaught. The first order properties of algebraic systems. *Fund. Math*, 47:57–103, 1959.
11. R. Ganian and P. Hliněený. On parse trees and Myhill–Nerode–type tools for handling graphs of bounded rank-width. *Disc. App. Math.*, 158(7):851–867, 2010.
12. R. Ganian, P. Hliněný, and J. Obdržálek. Unified approach to polynomial algorithms on graphs of bounded (bi-)rank-width. Submitted, 2009.
13. E. Grädel. Finite model theory and descriptive complexity. In *Finite Model Theory and Its Applications*, pages 125–230. Springer, 2007.
14. Y. Gurevich. Modest Theory of Short Chains. I. *J. Symb. Log.*, 44(4):481–490, 1979.
15. Y. Gurevich. Monadic second-order theories. In S. F. Jon Barwise, editor, *Model-Theoretic Logics*, pages 479–506. Springer-Verlag, 1985.
16. J. Hintikka. *Logic, Language-Games and Information: Kantian Themes in the Philosophy of Logic.* Clarendon Press, 1973.
17. P. Hliněný and S. Oum. Finding branch-decomposition and rank-decomposition. *SIAM Journal on Computing*, 38:1012–1032, 2008.
18. M. M. Kante. The rankwidth of directed graphs. Preprint. Available at: `http://arxiv.org/abs/0709.1433`, 2007.
19. J. Kneis, A. Langer, and P. Rossmanith. Courcelle's Theorem – a game-theoretic approach, 2010. submitted.
20. S. Oum. *Graphs of Bounded Rankwidth.* PhD thesis, Princeton University, 2005.
21. S. Oum and P. D. Seymour. Approximating clique-width and branch-width. *Journal of Combinatorial Theory Series B*, 96(4):514–528, 2006.
22. L. Øverlier and P. Syverson. Locating hidden servers. In *Proceedings of the 2006 IEEE Symposium on Security and Privacy.* IEEE CS, May 2006.