A Unified Approach to Boundedness Properties in MSO*

Łukasz Kaiser†1, Martin Lang2, Simon Leßenich3, and Christof Löding2

1 LIAFA, CNRS & Université Paris Diderot – Paris 7, France
2 Lehrstuhl für Informatik 7, RWTH Aachen University, Germany
3 Mathematische Grundlagen der Informatik, RWTH Aachen University, Germany

Abstract

In the past years, extensions of monadic second-order logic (MSO) that can specify boundedness properties by the use of operators referring to the sizes of sets have been considered. In particular, the logics costMSO introduced by T. Colcombet and MSO+U by M. Bojańczyk were analyzed and connections to automaton models have been established to obtain decision procedures for these logics. In this work, we propose the logic *quantitative counting MSO* (qcMSO for short), which combines aspects from both costMSO and MSO+U. We show that both logics can be embedded into qcMSO in a natural way. Moreover, we provide decidability proofs for the theory of its weak variant (quantification only over finite sets) for the natural numbers with order and the infinite binary tree. These decidability results are obtained using a regular cost function extension of automatic structures called resource-automatic structures.

1998 ACM Subject Classification F.1.1 Models of Computation, F.4.1 Mathematical Logic, F.4.3 Formal Languages

Keywords and phrases quantitative logics, monadic second order logic, boundedness, automatic structures, tree automata

Digital Object Identifier 10.4230/LIPIcs.CSL.2015.441

1 Introduction

The tight connection of monadic second-order logic (MSO), which is the extension of first-order logic by quantification over sets of elements, and finite automata over word and tree structures has led to a rich theory and many applications and decision procedures in verification and synthesis (see [20] for an introduction and overview).

In the past years, quantitative variants or extensions of MSO with a method to refer to the size of set variables have been introduced. Most prominently, there are costMSO, proposed by T. Colcombet (cf. [11]), and MSO+U, by M. Bojańczyk (cf. [3]). The logic costMSO extends standard MSO by a new atomic formula of the form $|X| \leq N$ (for a set variable X, and a parameter N that is interpreted as natural number) that is only allowed to appear positively in formulas. The N is a global parameter shared among all occurrences of this operator. The logic has a quantitative semantics: In a structure $\mathfrak{S}$, we assign a formula φ the minimal value for $N \in \mathbb{N} \cup \{\infty\}$ such that the formula is satisfied as a normal MSO

* This work was partially supported by the DFG Research Training Group 1298 (AlgoSyn) and the DFG research grant *Automatentheoretische Verifikationsprobleme mit Ressourcenschranken.*
† Currently at Google Inc.

formula with the additional cardinality bounds. If it cannot be satisfied for any N, the value is ∞ (and if it is satisfiable but no subformula of the form $|X| \leq N$ appears, the value is 0). The logic has proved to be useful for studying boundedness questions, in which the precise values of a formula are not of interest, but rather the question whether the values are bounded on sets of structures (for example on sets of words or trees). This intended semantics of (un)boundedness is directly encoded in MSO+U. It extends standard MSO by a new quantifier $UX.\varphi(X)$. Such a formula is satisfied if there are arbitrarily large finite sets X that satisfy $\varphi(X)$. Correspondingly, the logic MSO+U has a boolean semantics and does not forbid the use of negation.

The algorithmic properties of both logics and equivalent automaton models have been studied intensively. The logic costMSO is closely connected to *regular cost functions* (cf. [10]) and has an automaton model called B-automata. This automaton equivalence proved useful to obtain algorithmic methods for decision procedures of costMSO on finite words and infinite words (cf. [17]) and was also extended to the *weak* variant costWMSO, which only allows set quantification ranging over finite sets, on the infinite binary tree (cf. [21]). Similar questions were investigated for MSO+U. It turned out that the weak variant WMSO+U has equivalent automaton models on infinite words (lookahead limsup automata) (cf. [3]) and on the infinite binary tree (nested limsup automata) (cf. [7]). These yield decision procedures for the theory of WMSO+U on the respective structures. However, very recently M. Bojańczyk, P. Parys and S. Toruńczyk were able to show that the MSO+U-theory of the natural numbers with order is undecidable (cf. [6]).

We aim at constructing an MSO variant that combines the two aspects of costMSO and MSO+U and identified two key ingredients: First, a mechanism to measure the size of sets that satisfy a definable property as provided by $|X| \leq N$ in costMSO. Secondly, the possibility to test for these sizes within the logic similar to the quantifier U in MSO+U. It is clear that one loses decidability very soon if these mechanisms are too precise. Thus, we want to concentrate on the problem of boundedness. We propose the logic *quantitative counting MSO* (for short qcMSO) as a logic with quantitative semantics over the domain $\mathbb{N} \cup \{\infty\}$. Its basic syntax is similar to standard MSO without negation. We add the operator $|X|$ for set variables and the operator $\varphi = \infty$ for formulas. The definition of the semantics is inspired by the quantitative μ-calculus (cf. [14]). We associate `true` with ∞ and `false` with 0. Accordingly, the boolean connectives $\wedge$, $\vee$ are evaluated by min and max. The quantifiers $\exists$, $\forall$ are are evaluated by sup and inf. The formula $|X|$ evaluates to the number of elements of X and $\varphi = \infty$ has the value `true` (∞) if φ evaluates to ∞ and `false` (0) otherwise.

We show that there is a natural translation of costMSO and MSO+U into equivalent qcMSO formulas. Moreover, we show that the questions of *boundedness* and partly also *dominance* that are considered in connection with costMSO formulas can be expressed in qcMSO. These connections also hold for the respective weak variants of the logics.

The main contribution of this work is a decision procedure for qcWMSO sentences over the naturals with linear order and the infinite binary tree. More precisely, we prove the following main theorem.

▶ **Theorem 1.**

(a) *Given a sentence $\varphi \in$ qcWMSO, the evaluation $[\![\varphi]\!]^{(\omega,<)}$ of φ on the natural numbers with order is effectively computable.*

(b) *Given a sentence $\varphi \in$ qcWMSO, it is decidable whether $[\![\varphi]\!]^{\mathfrak{T}_2} = \infty$.*

The theorem is based on a quantitative extension of *automatic structures* called *resource-automatic structures*, which was presented in [18]. This framework introduces structures with quantitative relations – called *resource structures* – and provides a general method to compute the cost of first-order queries on structures whose relations are representable by B-automata. We show how such first-order queries can be extended with an ∞-comparison operator such that we can translate qcWMSO into such queries. We use the fact that the finite powerset structure of the naturals with a set size relation is resource-automatic. Furthermore, we provide an extension of the framework to finite trees and show that the finite powerset structure of the infinite binary tree with (an approximation of) the set size relation is *resource-tree-automatic*. The obtained decidability result can be seen as the best we could have hoped for as full qcMSO inherits the undecidability of full MSO+U already on the naturals. However, the connection between *weak* qcMSO and resource-automatic structures nicely resembles the known correspondence between WMSO and standard automatic structures.

The remainder of this work is structured as follows: First, we fix some notations and introduce the formal basics of regular cost functions, resource-automatic structures and the logics costMSO and MSO+U. In Section 3, we introduce qcMSO and explain the embedding of costMSO and MSO+U. Section 4 extends the theory of resource-automatic structures such that qcWMSO formulas over the naturals can be expressed in this framework. Moreover, we provide an extension to finite trees that enables us to decide boundedness of qcWMSO even on the infinite binary tree.

2 Preliminaries

We write Σ for a finite alphabet and Σ^* for the set of finite sequences (words) of letters from Σ. To evaluate MSO and its extensions on words, we consider words as relational structures: a word $w = w_1 \dots w_n \in \Sigma^*$ corresponds to the structure $(\{1,\dots,n\},<,(P_a)_{a\in\Sigma})$, where the P_a are monadic predicates such that $i \in P_a$ if and only if $w_i = a$. For infinite words, that is, sequences from Σ^ω, we use the set ω of natural numbers as the universe. Additionally, we consider finite and infinite binary trees over the alphabet Σ. A tree t is a mapping $\mathrm{dom}(t) \to \Sigma$ where $\mathrm{dom}(t) \subseteq \{0,1\}^*$ is a prefix-closed set that describes the nodes of the tree in such a way that ε represents the root node, and for a node $u \in \{0,1\}^*$, $u0$ is the left and $u1$ the right successor. Since we only consider binary trees, for every $u \in \mathrm{dom}(t)$ we either have $u0, u1 \in \mathrm{dom}(t)$ or $u0, u1 \notin \mathrm{dom}(t)$. A tree is finite if $\mathrm{dom}(t)$ is finite. The set of all finite trees over the alphabet Σ is denoted by $\mathcal{T}_\Sigma$. When we talk about infinite binary trees, we usually mean trees with $\mathrm{dom}(t) = \{0,1\}^*$.

2.1 Regular Cost Functions and the logic costMSO

In [9], regular cost functions were introduced based on two dual variants of *cost automata*. A cost automaton is a normal NFA with an additional finite set Γ of counters. These counters support three kinds of atomic operations: First, a counter can be incremented by one ($\mathtt{i}$). Secondly, a counter can be reset to zero ($\mathtt{r}$) and lastly, a counter can be *checked* ($\mathtt{c}$). The counters are driven by the transitions. Correspondingly, for a cost automaton $(Q,\Sigma,q_0,\Delta,F,\Gamma)$ the transition relation Δ is a subset of $Q \times \Sigma \times Q \times (\{\mathtt{i},\mathtt{r},\mathtt{c}\}^*)^\Gamma$. A run ρ of such an automaton is identified with a sequence of states and transitions and is, as usual, called accepting if it starts in q_0 and ends in a state of F. Along the run, we simulate the values of the counters (starting with 0) according to the operations on the transitions, and whenever a counter is checked, its current value is stored for later evaluation. We denote the set of checked counter values (over all counters) by $C(\rho)$. The semantics of cost automata are functions $\Sigma^* \to \mathbb{N} \cup \{\infty\}$, and come in two (dual) flavors: A *B-automaton* has only counter operations of the forms $\{\varepsilon, \mathtt{ic}, \mathtt{r}\}$, a run ρ has the value $\sup C(\rho)$ and a word $w \in \Sigma^*$ is assigned the infimum over all accepting runs ρ on this word. Dually, an *S-automaton* has only counter operations of the forms $\{\varepsilon, \mathtt{i}, \mathtt{cr}, \mathtt{r}\}$, a run ρ has the value $\inf C(\rho)$ and a word $w \in \Sigma^*$ is assigned the supremum over all accepting runs ρ on this word. For a B-/S-cost automaton $\mathcal{A}$, we write $[\![\mathcal{A}]\!]_B$ and $[\![\mathcal{A}]\!]_S$ to refer to the respective semantics.

Regular cost functions are the functions definable by B- or S-automata up to a certain equivalence relation $\approx$. The equivalence is based on the notion of *correction functions.* A correction function $\alpha : \mathbb{N} \cup \{\infty\} \to \mathbb{N} \cup \{\infty\}$ is a monotone mapping that maps ∞ and only ∞ to ∞. Let $f, g : A \to \mathbb{N} \cup \{\infty\}$ be two functions. We say f is α-dominated by g and write $f \preceq_\alpha g$ if for all $a \in A$: $f(a) \leq \alpha(g(a))$. We call f and g α-equivalent and write $f \approx_\alpha g$ if $f \preceq_\alpha g$ and $g \preceq_\alpha f$. Additionally, we just write $\approx$ to indicate that there exists an α such that the relation holds. The $\approx$ relation is the same as saying that two cost functions are bounded on the same subsets of their domain. Formally, we have $f \approx g$ iff for all $B \subseteq A$: $\sup_{x \in B} f(x) < \infty \Leftrightarrow \sup_{x \in B} g(x) < \infty$. A proof can be found in [9].

Regular cost functions possess closure properties comparable to those of regular languages (cf. [10]). They are closed under min and max of two functions, which extends the classical union and intersection closure. Moreover, they are closed under inf- and sup-projections. These projections extend classical alphabet projection in the following way: Let Σ be an alphabet, $\pi : \Sigma^{k+1} \to \Sigma^k$ a projection that removes the last component and let π^* be the letterwise extension of π to words. The (π)-inf-projection of a function $f : (\Sigma^{k+1})^* \to \mathbb{N} \cup \{\infty\}$ is defined by $w \mapsto \inf_{u \in \pi^{*-1}(w)} f(u)$, and respectively for sup.

It is well-known that regular languages correspond to languages definable in MSO over word models. This equivalence was lifted to regular cost functions with the logic costMSO. The syntax of costMSO is standard MSO syntax extended with a new predicate that is only allowed to appear positively in the formula: $|X| \leq N$ for all set variables X. costMSO is evaluated over standard relational structures by an inductively defined semantics. For the sake of a uniform presentation, we assume (w.l.o.g.) that only set variables are used. Additionally, set inclusion $\subseteq$ is added as a relation. For a relational structure $\mathfrak{S} = (S, R_1, \ldots, R_n)$ and a valuation $\beta : X \to 2^S$ of the free variables the semantics can be defined as follows (see [11]):

$$[\![R_i X_1 \ldots X_{k_i}]\!]^{\mathfrak{S},\beta} := \begin{cases} 0, & (a_1, \ldots, a_{k_i}) \in R_i^{\mathfrak{S}}, \beta(X_i) = \{a_i\} \\ \infty, & \text{otherwise} \end{cases}$$

$$[\![\neg R_i X_1 \ldots X_{k_i}]\!]^{\mathfrak{S},\beta} := \begin{cases} \infty, & (a_1, \ldots, a_{k_i}) \in R_i^{\mathfrak{S}}, \beta(X_i) = \{a_i\} \\ 0, & \text{otherwise} \end{cases}$$

$$[\![|X| \leq N]\!]^{\mathfrak{S},\beta} := |\beta(X)|$$

$$[\![\varphi \wedge \psi]\!]^{\mathfrak{S},\beta} := \max([\![\varphi]\!]^{\mathfrak{S},\beta}, [\![\psi]\!]^{\mathfrak{S},\beta}) \qquad [\![\varphi \vee \psi]\!]^{\mathfrak{S},\beta} := \min([\![\varphi]\!]^{\mathfrak{S},\beta}, [\![\psi]\!]^{\mathfrak{S},\beta})$$

$$[\![\exists X \varphi(X)]\!]^{\mathfrak{S},\beta} := \inf_{S' \subseteq S} [\![\varphi(X)]\!]^{\mathfrak{S},\beta[X \mapsto S']} \qquad [\![\forall X \varphi(X)]\!]^{\mathfrak{S},\beta} := \sup_{S' \subseteq S} [\![\varphi(X)]\!]^{\mathfrak{S},\beta[X \mapsto S']}$$

This semantics assigns each sentence φ a function $\Sigma^* \to \mathbb{N} \cup \{\infty\}$ over word models. The main equivalence theorem for costMSO states that these functions are exactly the regular cost functions (cf. [9]). The central decision problems for costMSO are boundedness and dominance: A formula φ is bounded over a domain $\mathcal{D}$, if there exists a bound $B \in \mathbb{N}$ such that $[\![\varphi]\!]^{\mathfrak{S}} < B$ for all $\mathfrak{S} \in \mathcal{D}$. A formula φ dominates a formula ψ on a domain $\mathcal{D}$, if for all subsets $\mathcal{C} \subseteq \mathcal{D}$ it holds that whenever φ is bounded, ψ is bounded as well.

2.2 MSO+U

Another approach to introduce a method to express boundedness or unboundedness problems in MSO is with the help of the unbounding quantifier. Unlike costMSO, this leads to a qualitative extension. In MSO+U, a new, third set quantifier U is added to MSO. This quantifier evaluates to true if there are arbitrarily large finite sets that satisfy a formula, so formally, $\mathfrak{A} \vDash_{\text{MSO+U}} UX.\varphi(X)$ if and only if $\mathfrak{A} \vDash_{\text{MSO}} \exists X(|X| > n \wedge |X| < \infty \wedge \varphi(X))$ for every $n \in \mathbb{N}$.

Note that, for every fixed n, $|X| > n$ is expressible in classical MSO. For completeness reasons, the dual quantifier B is also added, with the semantics that $\mathfrak{A} \vDash BX.\varphi$ if and only if there is a (finite) bound on the size of sets that satisfy φ. For obvious reasons, MSO+U is only studied over infinite structures, in particular infinite words and trees.

Unlike classical MSO and costMSO, there is most likely no automaton model for MSO+U with full second-order quantification for topological reasons [15], and furthermore, the MSO+U-theory of the natural numbers with order is already undecidable [6]. However, the weak variant is decidable over infinite words [4] and infinite trees [8].

2.3 (Resource-) Automatic Structures and FO+RR

The theory of automatic structures provides a formalism to obtain logical structures with a decidable first-order theory by automata representations of the structures (for a comprehensive introduction see, e.g., [16]). A relational structure $\mathfrak{S} = (S, R_1, \dots, R_n)$ is called *automatic* if there are a representation of the universe S in form of a regular language and representations of the relations R_1 up to R_n in the form of *synchronous transducers*. A synchronous transducer for a j-ary relation over $S \subseteq \Sigma^*$ can be seen as a (normal) automaton operating over the vector alphabet $(\Sigma \uplus \{\$\})^j$. It reads all j words letter-by-letter in parallel. Shorter words are padded to the length of the longest word with a newly introduced padding symbol \$. This transformation from a tuple of words to a (padded) word over a vector alphabet is called *convolution*.

The main result for automatic structures is that they always have a decidable first-order theory (see [16]). This is obtained by inductively translating logical operations into operations for automata. Boolean connectives can be represented by union and intersection of regular languages and existential quantification corresponds to alphabet projection for vector alphabets. The original result has been extended to ω-words and finite and infinite trees (see [1, 2]).

Motivated by quantitative verification questions, the idea of automatic structures has been lifted to *resource structures* in [18]. Resource structures are a quantitative extension of relational structures. The quantitative aspect is introduced via the relations: A tuple of elements $\bar{a}$ is not just in some relation R or not, but being in relation may *cost* some value from $\mathbb{N} \cup \{\infty\}$. The verification-driven question was how expensive it is to satisfy a first-order definable property in such a structure. The logic FO+RR (first-order over resource relations) was designed to provide a formalism for this question. Its syntax is identical to normal FO without negation. For a resource structure $\mathfrak{S} = (S, R_1, \dots, R_n)$ and a variable interpretation $\beta : X \to S$, the semantics is inductively defined as follows:

$$[\![R_i x_1 \dots x_{k_i}]\!]^{\mathfrak{S},\beta} := R_i^{\mathfrak{S}}(\beta(x_1), \dots, \beta(x_{k_i}))$$

$$[\![x = y]\!]^{\mathfrak{S},\beta} := \begin{cases} 0, & \beta(x) = \beta(y) \\ \infty, & \text{otherwise} \end{cases} \qquad [\![x \neq y]\!]^{\mathfrak{S},\beta} := \begin{cases} \infty, & \beta(x) = \beta(y) \\ 0, & \text{otherwise} \end{cases}$$

$$[\![\varphi \wedge \psi]\!]^{\mathfrak{S},\beta} := \max([\![\varphi]\!]^{\mathfrak{S},\beta}, [\![\psi]\!]^{\mathfrak{S},\beta}) \qquad [\![\varphi \vee \psi]\!]^{\mathfrak{S},\beta} := \min([\![\varphi]\!]^{\mathfrak{S},\beta}, [\![\psi]\!]^{\mathfrak{S},\beta})$$

$$[\![\exists x \varphi(x)]\!]^{\mathfrak{S},\beta} := \inf_{s \in S} [\![\varphi(x)]\!]^{\mathfrak{S},\beta[x \mapsto s]} \qquad [\![\forall x \varphi(x)]\!]^{\mathfrak{S},\beta} := \sup_{s \in S} [\![\varphi(x)]\!]^{\mathfrak{S},\beta[x \mapsto s]}$$

This semantics answers the intuitive question in the following way: Let φ be an FO+RR-formula with $[\![\varphi]\!]^{\mathfrak{S}} = n < \infty$. If we consider φ as a normal FO-formula, it is satisfied if we allow all those tuples to be in relation (in the classical sense) that cost at most n. This relation also explains the absence of negation. In order to preserve this intuitive semantics, monotonicity is necessary: if we allow a higher resource usage, more formulas should become true.

Resource-automatic structures extend the idea of automatic structures to resource structures and FO+RR. A structure is called resource-automatic if its universe can be represented by a regular language and the semantics of the relations can be specified via *synchronous cost transducers*. For the sake of simplicity it suffices to see such a transducer as a B-automaton operating on a vector alphabet in the same way as for standard synchronous transducers. The details can be found in [18]. The main result that we use here is that the value of FO+RR formulas can be computed over resource-automatic structures.

2.4 Cost Tree Automata and Cost Games

In [12], the theory of regular cost functions was lifted to finite trees. Regular cost functions on trees are defined by B- or S-tree automata. These automata can be seen as an extension of B-/S-automata on words and nondeterministic top-down tree automata (see, e.g., [13] for an introduction to regular tree languages and tree automata). For the sake of a simpler presentation, we move the counter actions to the states of the automaton and tailor the definition to our setting of binary trees.

A cost tree automaton is a tuple $\mathcal{A} = (Q, \Sigma, Q_I, \Delta, F, \Gamma, \gamma)$ where the components have the following meaning: Q is a finite set of states, Σ is the input alphabet, $Q_I \subseteq Q$ is a set of initial states, $\Delta \subseteq Q \times \Sigma \times Q \times Q$ is the transition relation, $F \subseteq \Sigma \times Q$ the set of final letter/state combinations, Γ the finite set of counters and $\gamma : Q \to (\Gamma \to \{\mathtt{i}, \mathtt{r}, \mathtt{cr}\}^*)$ the counter actions. A run ρ of $\mathcal{A}$ on a tree t is also a tree with $\mathrm{dom}(\rho) = \mathrm{dom}(t)$, but with a labeling from Q that is consistent with the transition relation Δ. We call ρ accepting if $\rho(\varepsilon) \in Q_I$ and there are matching letter/states pairs for the leaves in F. Formally, for all $u \in \mathrm{leaves}(t) : (t(u), \rho(u)) \in F$. In the same way as for cost automata on words, we define B- and S-tree automata. They inherit the restrictions on the counter operations from word automata and their quantitative semantics is defined in the same spirit. However, we now compute the value along all paths from the root to a leaf in the tree as for word automata and take the maximum of the values in B-automata and the minimum in S-automata. Again, we write $[\![\mathcal{A}]\!]_B$ and $[\![\mathcal{A}]\!]_S$ to refer to this semantics.

For an analysis of tree automata, it is often helpful to take a game-theoretic viewpoint to the membership problem. To this end, we define *cost games* following the idea of [12]. For our purpose, it is helpful to view a cost game as a standard two-player reachability game on a finite graph that is extended with a finite set Γ of counters and counter actions γ that map every position in the game to the counter actions as for cost tree automata. As usual, the game positions are partitioned into two sets: One that belongs to the first player – called Eve – and one that belongs to the second player – called Adam. A play is formed similar to standard games: The play starts in some initial position. Then, the player that controls the respective game position chooses a next position according to the edges of the graph. Additionally, we simulate the counters along with the play and write $C(\tau)$ for the set of checked counter values in a play analogously to cost automata. For an introduction to games and their connection to tree automata see, e.g., [20].

We also define B- and S-games with the restrictions on the counter actions as for automata. In both types of games, Eve aims to reach the goal positions F of the reachability game. In B-games she additionally wants to minimize the largest checked counter value. In S-games she wants to maximize the smallest checked counter values. This is analogous to automata. Correspondingly, an infinite play that never reaches F has value ∞ in B-games and value 0 in S-games. We define the *value* of a game based on the best values the players can enforce based on their *strategies*. For this purpose, we consider strategies as in standard two-player games on graphs. If Eve and Adam fix their strategies σ_E and σ_A, the resulting play is fixed and we denote it by τ_{σ_E,σ_A}. Cost games are determined (see [12]). That is, $\inf_{\sigma_E}\sup_{\sigma_A}\mathrm{val}_B(\tau_{\sigma_E,\sigma_A}) = \sup_{\sigma_A}\inf_{\sigma_E}\mathrm{val}_B(\tau_{\sigma_E,\sigma_A})$ and $\sup_{\sigma_E}\inf_{\sigma_A}\mathrm{val}_S(\tau_{\sigma_E,\sigma_A}) = \inf_{\sigma_A}\sup_{\sigma_E}\mathrm{val}_S(\tau_{\sigma_E,\sigma_A})$ and we write $\mathrm{val}_B(\mathcal{G})$ and $\mathrm{val}_S(\mathcal{G})$ for this value of the game $\mathcal{G}$.

It is helpful for our analysis to view the value computation of a tree t on a cost automaton as a cost game. The idea is essentially identical to the *membership game* (see [20]) for tree automata. The two players partly construct a run on one path of the tree from the root to a leaf. It starts in the root with some state from Q_I. In every position, Eve selects the transition from Δ that should be used and Adam chooses whether he wants to continue in the left or right child of the current node. The game continues in this node with the state given by the chosen transition. The counters and counter actions are just copied from the respective states in the automaton. The final positions are determined by the final letter/state pairs. We call this game the *cost membership game* of $\mathcal{A}$ on t and write $\mathcal{M}_{\mathcal{A},t}$. We have that $\mathrm{val}_B(\mathcal{M}_{\mathcal{A},t}) = [\![\mathcal{A}]\!]_B(t)$ and $\mathrm{val}_S(\mathcal{M}_{\mathcal{A},t}) = [\![\mathcal{A}]\!]_S(t)$.

3 Quantitative Counting MSO

In this section we introduce a quantitative extension of MSO with a focus on counting. We do so following the approach used for the quantitative μ-calculus [14]. Thus, we keep the standard syntax, and also use the traditional interpretations of the operators. However, instead of interpreting these over $\{0,1\}$, we evaluate minimums and maximums over the natural numbers with infinity.

To be more precise, we start with MSO without first-order variables and without negation. We then add a new atom to count the sizes of sets to the syntax, and allow formulas to be compared to infinity. Fixing a relational signature $\tau = \{R_1,\dots,R_n\}$ and a set $\mathcal{V} = \{X_1,\dots,X_m\}$ of second-order variables, formulas of qcMSO are defined inductively.

- Atomic formulas are of the form $RX_1\dots X_r$, $X_i = \varnothing$, $X_i \in X_j$, or $|X_i|$, for variables X_k and relation symbols $R \in \tau$ of arity r.
- If φ,ψ are formulas, then $\varphi\wedge\psi$, $\varphi\vee\psi$, $\varphi<\infty$, and $\varphi=\infty$ are formulas.
- If φ is a formula and X a variable, then $\exists X\varphi$ and $\forall X\varphi$ are formulas.

In contrast to the logics defined before, we view ∞ as true and 0 as false, which allows us to adapt the classical semantics. Furthermore, all atomic formulas but formulas $|X|$ are boolean. Given a τ-structure $\mathfrak{A}$ and an interpretation $\beta\colon X\to\mathcal{P}(A)$, the semantics $[\![\cdot]\!]^{\mathfrak{A},\beta}\colon \mathrm{qcMSO}(\tau)\to\mathbb{N}\cup\{\infty\}$ is defined as follows, where we omit $\mathfrak{A},\beta$ if clear from the context for better readability.

$$[\![|X|]\!] = |\beta(X)| \qquad [\![RX_1\dots X_r]\!] = \begin{cases}\infty, & (a_1,\dots,a_r)\in R, \beta(X_i)=\{a_i\}\\ 0, & \text{otherwise}\end{cases}$$

$$[\![X=\varnothing]\!] = \begin{cases}\infty, & \beta(X)=\varnothing\\ 0, & \text{otherwise}\end{cases} \qquad [\![X\in Y]\!] = \begin{cases}\infty, & \beta(X)=\{a\}, a\in\beta(Y)\\ 0, & \text{otherwise}\end{cases}$$

$$[\![\varphi\vee\psi]\!] = \max([\![\varphi]\!],[\![\psi]\!]) \qquad [\![\varphi\wedge\psi]\!] = \min([\![\varphi]\!],[\![\psi]\!])$$

$$[\![\varphi<\infty]\!] = \begin{cases}\infty, & [\![\varphi]\!]<\infty\\ 0, & \text{otherwise}\end{cases} \qquad [\![\varphi=\infty]\!] = \begin{cases}\infty, & [\![\varphi]\!]=\infty\\ 0, & \text{otherwise}\end{cases}$$

$$[\![\exists X\varphi]\!] = \sup_{A'\subseteq A}[\![\varphi]\!]^{\beta[X\mapsto A']} \qquad [\![\forall X\varphi]\!] = \inf_{A'\subseteq A}[\![\varphi]\!]^{\beta[X\mapsto A']}$$

As a convention, formulas whose evaluation can only be 0 or ∞, that is, `true` or `false`, are called boolean formulas. For example, formulas without any occurrence of $|X|$ are boolean, and so are formulas $\varphi = \infty$.

As atomic formulas which are also formulas of MSO always evaluate to ∞ and 0, qcMSO clearly extends MSO: a formula can be translated by replacing every occurrence of a negation with a comparison $< \infty$. What is more, it is easily expressible that a set is finite. As supremums and infimums over sets that satisfy a certain property can be encoded in a straightforward manner, we can thus also express the quantifier U:

$$UX.\varphi \equiv (\exists X(|X| \wedge |X| < \infty \wedge \varphi)) = \infty.$$

It is an easy consequence that qcMSO subsumes MSO+U.

▶ **Lemma 2.** *For every formula $\varphi \in$ (W)MSO+U there is a formula $\varphi' \in$ qc(W)MSO such that $\mathfrak{A} \models \varphi$ if and only if $[\![\varphi']\!]^{\mathfrak{A}} = \infty$.*

One easily obtains the following lemma by exchanging $\wedge$ and $\vee$, swapping the quantifiers, replacing positive boolean atomic formulas φ by $\varphi < \infty$ and negated atoms $\neg\varphi$ by φ to take the inverted semantics between costMSO and qc(W)MSO into account. Additionally, we need to rewrite $|X| < N$ to $|X|$ because of the different syntax of the operators.

▶ **Lemma 3.** *For every cost(W)MSO-formula φ there exists a qc(W)MSO-formula φ' such that $[\![\varphi]\!]^{\mathfrak{A}} = [\![\varphi']\!]^{\mathfrak{A}}$.*

Furthermore, the boundedness property and the dominance relation for costMSO on finite words and trees are expressible in qcMSO on $(\omega, <)$ and the infinite binary tree $\mathfrak{T}_2 = (\{0,1\}^*, S_0, S_1)$, respectively, as stated by the following lemma.

▶ **Lemma 4.**
1. *Given a costMSO-formula φ over finite words, one can effectively construct a qcWMSO-formula φ_b such that $[\![\varphi_b]\!]^{(\omega,<)} = \infty$ (`true`) if and only if φ is bounded over finite words.*
2. *Given two costMSO-formulas φ, ψ over finite words, one can effectively construct a qcMSO-formula ϑ_d such that $[\![\vartheta_d]\!]^{\mathfrak{T}_2} = \infty$ if and only if φ dominates ψ on finite words.*

Proof. Regarding 1., a finite word over $\Sigma = \{0,1\}$ can be represented by two sets X, X_1 such that $X \subseteq \omega$ is a finite initial subset of the natural numbers indicating the length of the word and $X_1 \subseteq X$ (also finite) contains the positions labeled with 1. Clearly, such sets can be defined in WMSO and thus qcWMSO by a formula $\psi(X, X_1)$. As costMSO is subsumed by qcMSO, there exists a qcMSO-formula φ' with the same evaluation. Let φ'_r be this formula where quantification is relativized to (finite) X and $P_1 Y$ is replaced by $Y \in X_1$. Then, boundedness is expressed by $(\exists X \exists X_1(\psi(X, X_1) \wedge \varphi'_r)) < \infty$.

Regarding 2., we identify a word $w \in \{0,1\}^*$ with the respective position in the tree. For MSO, it is easy to define, given a sentence φ over finite words, a formula $\varphi'(x)$ over the infinite binary tree such that $w \models \varphi$ if and only if $\mathfrak{T}_2 \models \varphi'(w)$. This directly extends to costMSO, and accordingly, we also obtain corresponding translations $\widehat{\varphi}(X), \widehat{\psi}(X) \in$ qcMSO for φ, ψ. Dominance is then expressed by $\forall X[(\exists Y(Y \in X \wedge \widehat{\varphi}(Y))) = \infty \vee (\exists Y(Y \in X \wedge \widehat{\psi}(Y))) < \infty]$. We remark that it is important here that $\forall X$ also quantifies over infinite sets to match the definition of dominance. ◀

By adapting the construction for the boundedness formula, it follows straightforwardly that boundedness on infinite words and finite trees can also be expressed (in the respective structures).

▶ **Corollary 5.**

- *The boundedness problem for costMSO on infinite words can be expressed in qcMSO on* $(\omega, <)$.
- *The boundedness problem for costMSO on finite trees can be expressed in qcWMSO on the infinite binary tree.*

As qcMSO extends both costMSO and MSO+U, negative results for either of these transfer. In fact, the conditional undecidability result for MSO+U on the infinite binary tree from [5] was recently improved: it was shown in [6] that the MSO+U-theory of $(\omega, <)$ is undecidable. Thus, the undecidability of the model-checking problem for qcMSO follows:

▶ **Corollary 6.** *Given a qcMSO-sentence* φ, *it is undecidable whether* $[\![\varphi]\!]^{(\omega,<)} = \infty$.

4 Resource-Automatic Structures and FO+RR$^{=\infty}$

Towards the decidability of the qcWMSO-theories of the natural numbers with order and the infinite binary tree, we follow an approach used in [19]: Instead of working with second-order quantification directly, we consider the first-order problem on the powerset structure restricted to finite sets. We prove that the respective structures are resource-automatic and resource-tree-automatic, respectively, and that deciding the theory reduces to deciding the theory of an extension of FO+RR with infinity comparisons.

As a first step, we extend syntax and semantics of FO+RR introduced earlier by the new operators $\varphi = \infty$ and $\varphi < \infty$ for formulas φ such that $\varphi = \infty$ evaluates to 0 if $[\![\varphi]\!] = \infty$ and to ∞ otherwise, and dually for $\varphi < \infty$. We use the decidability of FO+RR established in [18], which itself relies on the closure properties for regular cost functions described in [9]. To prove that this extension FO+RR$^{=\infty}$ is still decidable on resource-automatic structures, it suffices to provide an automata-theoretical construction that transforms a B-automaton $\mathcal{A}$ for φ into a B-automaton $\mathcal{A}_\infty$ for $\varphi = \infty$.

▶ **Lemma 7.** *Let* $\mathcal{A}$ *be a* B*-automaton. One can effectively construct a* B*-automaton* $\mathcal{A}_\infty$ *such that* $[\![\mathcal{A}_\infty]\!](w) = 0$ *if* $[\![\mathcal{A}]\!](w) = \infty$ *and* $[\![\mathcal{A}_\infty]\!](w) = \infty$ *otherwise*

Proof. As we consider automata over finite words, observe that $[\![\mathcal{A}]\!](w) = \infty$ only if there are no accepting runs of $\mathcal{A}$ on w. Hence, to construct $\mathcal{A}_\infty$, we first view $\mathcal{A}$ as an NFA by removing all counters, then complement it, and reintroduce a dummy counter that is never checked. By construction, this automaton $\mathcal{A}_\infty$ maps a word to 0 if and only if $\mathcal{A}$ as an NFA rejects the word, and all other words are mapped to ∞. ◀

▶ **Corollary 8.** *Given an FO+RR*$^{=\infty}$*-sentence* φ *and a resource-automatic structure* $\mathfrak{A}$, *it is decidable whether* $[\![\varphi]\!]^{\mathfrak{A}} = \infty$.

To prove that the qcWMSO-theory of $(\omega, <)$ is decidable, consider the structure $\mathfrak{F} = (\mathrm{FinPot}(\mathbb{N}), <, |\cdot|, \in, = \varnothing)$, where

- $\mathrm{FinPot}(\mathbb{N}) = \{a \subseteq \mathbb{N} \mid |a| < \infty\}$,
- $a < b = \infty$ if $a = \{a'\}, b = \{b'\}$ are singleton sets such that $a' < b'$, and $a < b = 0$ otherwise,
- $|a|$ evaluates to the size of a,
- $a \in b = \infty$ if $a = \{a'\}$ is a singleton and $a' \in b$, and $a \in b = 0$ otherwise,
- and $a = \varnothing$ evaluates to ∞ if a is indeed empty and to 0 otherwise.

It is not difficult to see that this structure is resource-automatic, using the regular language $\{0\} \cup \{0,1\}^*1$ where a word corresponds to the set that consists of the indices where the word is 1. For $|\cdot|$, a single counter to count the occurrences of 1s suffices. The other relations correspond to the complements of the relations in the classical automatic structures setting.

Following the approach used to embed costMSO into qcMSO, by exchanging conjunctions and disjunctions, replacing $\exists X$ by $\forall x$ and $\forall X$ by $\exists x$ and swapping $= \infty$ and $< \infty$, one obtains the following lemma.

▶ **Lemma 9.** *For every qcWMSO-sentence φ, one can effectively construct an FO+RR$^{=\infty}$-sentence φ' such that $[\![\varphi]\!]^{(\omega,<)} = [\![\varphi']\!]^{\mathfrak{F}}$.*

▶ **Corollary 10.** *Given a qcWMSO-sentence φ, it is decidable whether $[\![\varphi]\!]^{(\omega,<)} = \infty$.*

It was argued in [18] that exact evaluations of FO+RR-sentences on resource-automatic structures can be computed once it is known that the evaluation is bounded. This can, for example, be achieved by successively trying parameters $n = 0, 1, \dots$ with standard first-order evaluation on the structure where a resource relation R is replaced by the relation of tuples of R of cost at most n. As the automata $\mathcal{A}_\infty$ have boolean evaluations, thus are essentially NFAs, this approach can be lifted to FO+RR$^{=\infty}$-sentences, by substituting only relations outside the scope of ∞-comparisons. Accordingly, exact evaluations can be computed also for qcWMSO on the ordered natural numbers.

▶ **Theorem 1.**

(a) *Given a sentence $\varphi \in$ qcWMSO, the evaluation $[\![\varphi]\!]^{(\omega,<)}$ of φ on the natural numbers with order is effectively computable.*

4.1 Resource-Tree-Automatic Structures

We aim at generalizing the idea of resource-automatic structures to universes that are representable as a regular tree language. This extends the well-understood idea of tree-automatic structures (see, e.g., [1]). Moreover, it allows us to reuse the idea presented previously to obtain an algorithm for qcWMSO on the infinite binary tree, because finite trees can be used to represent all finite subsets of the infinite binary tree.

The general approach to resource-tree-automatic structures and the presentation of the result follow the ideas for resource-automatic structures as presented in [18]. First, we fix some additional notation. Secondly, we describe an inductive translation strategy from FO+RR$^{=\infty}$-formulas to cost tree automata. While the cases of the boolean connectives can be directly transferred, the translation of the quantifiers needs some more insight into cost games. Correspondingly, we analyze strategies in S-games and use the results to complete the inductive translation, which provides an algorithmic method to compute the (approximate) value of FO+RR$^{=\infty}$-formulas.

First, we extend the definition of convolution and transducers to trees. For a finite alphabet Σ, let $\Sigma^{\otimes m} = (\Sigma \cup \{\$\})^m$ and let $t_1 \in \mathcal{T}_{\Sigma^{\otimes m}}, t_2 \in \mathcal{T}_\Sigma$ be two trees. We define the *convolution* by $t := t_1 \otimes t_2 \in \mathcal{T}_{\Sigma^{\otimes m+1}}$ with $\mathrm{dom}(t) = \mathrm{dom}(t_1) \cup \mathrm{dom}(t_2)$ and $t(u)_i = t_1(u)_i$ if $u \in \mathrm{dom}(t_1)$, $t(u)_i = \$$ otherwise for $i \leq m$ and $t(u)_{m+1} = t_2(u)$ if $u \in \mathrm{dom}(t_2)$, $t(u)_{m+1} = \$$ otherwise. A tree $t \in \mathcal{T}_{\Sigma^{\otimes m}}$ is *correctly padded* if there are $t_1, \dots, t_m \in \mathcal{T}_\Sigma$ such that $t = t_1 \otimes \dots \otimes t_m$. This means correctly padded trees have have no \$ if there are normal letters in the same component of a descendant. Moreover, they have no positions labeled completely with padding symbols. We write $\square := \$^m$ as a short-hand for a vector of appropriate dimensionality containing only padding symbols. An m-dimensional synchronous B-/S-tree transducer $\mathcal{A}$ is a cost tree automaton operating over the alphabet $\Sigma^{\otimes m}$. We define its semantics by the cost automaton semantics over the convolution: $[\![\mathcal{A}]\!] : (\mathcal{T}_\Sigma)^m \to \mathbb{N} \cup \{\infty\}$, $(t_1, \dots, t_m) \mapsto [\![\mathcal{A}]\!]_{B/S}(t_1 \otimes \dots \otimes t_m)$. With these preparations, we can define resource-tree-automatic structures.

▶ **Definition 11.** Let $\mathfrak{S} = (S, R_1, \dots, R_m)$ be a resource structure. We call $\mathfrak{S}$ resource-tree-automatic if S is representable as a regular language of finite trees and there are synchronous B-/S-tree transducers $\mathcal{A}_{R_i}$ such that $R_i^{\mathfrak{S}} = [\![\mathcal{A}_{R_i}]\!]$.

For the purpose of a clearer proof presentation we will assume that $S = \mathcal{T}_\Sigma$. This is no restriction of the general case since we can introduce a new automatic predicate $P_S \subseteq \mathcal{T}_\Sigma$ that contains exactly the elements of S and relativize all quantifications w.r.t. P_S for every regular tree language S.

4.1.1 Translating FO+RR$^{=\infty}$ to transducers

We now provide the necessary ingredients for an inductive translation of FO+RR$^{=\infty}$-formulas into synchronous cost transducers. For a given formula φ with k free variables, we construct a k-dimensional synchronous cost transducer $\mathcal{A}_\varphi$ such that $[\![\varphi]\!] = [\![\mathcal{A}_\varphi]\!]$. The cases of atomic formulas are simple. For a relation R_i, we are given a cost transducer $\mathcal{A}_{R_i}$ by definition. The operators $=$ and $\neq$ can be implemented with simple cost tree automata that just check whether in all positions in the tree all letters in the alphabet vector match or that there is a mismatch somewhere, respectively. The semantics of the boolean connectives is min and max. Correspondingly, we directly use the closure of cost tree automata under min and max, which was already established in [12]. The $=\infty$ operator can be translated in the same way as for finite words: By the definition of the semantics of B-tree automata, the value of a tree is ∞ if and only if the tree is rejected in the classical sense, i.e., there is no strategy of Eve in $\mathcal{M}_{\mathcal{A},t}$ to always reach a final state in every play. Thus, we can obtain an automaton for $=\infty$ by ignoring the counters and constructing a classical complement automaton for the given one. When interpreted as a B-tree automaton, it will output 0 for trees that were previously not accepted (that is, had value ∞) and ∞ otherwise.

It remains to consider existential and universal quantification. In the classical setting, universal quantification can be expressed by negation and existential quantification. In our setting, we have to consider both existential and universal quantification since FO+RR$^{=\infty}$ has no negation. We deal with these quantifications by inf-projection and sup-projection for cost automata. However, on the automaton level, one has to deal with the padding symbol. After projecting away one of the components, the automaton may contain transitions only labeled with padding symbols. Let us illustrate the problem by a simple example over $\Sigma = \{a, b\}$: Let Rxy be a binary relation symbol that evaluates to the length of the longest path in y. Formally, $R(t_1, t_2) := \max_{u \in \mathrm{dom}(t_2)} |u|$. A B-tree automaton $\mathcal{A}$ that just increments its counter in every step on the second tree implements this relation. Now, consider the tree $t_0 = a$ that consists only of a root node. For this tree, we have $[\![\forall y R t_0 y]\!] = \infty$ but if we look at the sup-projection of $\mathcal{A}$ to the first component denoted by $\mathcal{A}^{\sup}$ we obtain

$$[\![\mathcal{A}^{\sup}]\!](a) = \max\{[\![\mathcal{A}]\!]((a,a)), [\![\mathcal{A}]\!]((a,b)), [\![\mathcal{A}]\!]((a,\$))\} = 1$$

To compute the supremum over all trees, we have to consider t_0 extended by arbitrarily long sequences of padding symbols ($\square$).

In the classical setting (for existential quantification and standard projection), this can be handled by treating such pure padding transitions as ε-transitions and then eliminating them. However, in cost automata the pure padding transitions are much more difficult to eliminate (see [18] where this problem is treated for cost automata on words). As a solution, we split the computation of sup and inf into the respective projection operation and an additional sup/inf over arbitrarily long padding sequences. Formally, for a tree $t \in \mathcal{T}_{\Sigma^{\otimes m}}$, let

$$\mathrm{padext}\,(t) := \left\{ s \in \mathcal{T}_{\Sigma^{\otimes m}} \mid \mathrm{dom}(s) \supseteq \mathrm{dom}(t), s(u) = \begin{cases} t(u), & u \in \mathrm{dom}(t) \\ \square, & \text{otherwise} \end{cases} \right\}$$

Consider the case of existential quantification, and assume that we have applied the inf-projection and obtained a B-automaton $\mathcal{A}$ (which might still contain pure padding transitions). For a tree $t \in \mathcal{T}_{\Sigma^{\otimes m}}$ we are interested in the value $\inf_{s \in \mathrm{padext}(t)} [\![\mathcal{A}]\!]_B(s)$. If we modify $\mathcal{A}$ such that it can simulate on t all runs of $\mathcal{A}$ on trees from padext (t), then we obtain the correct value for t because the semantics of B-automata takes the infimum over all runs. Similarly, we use S-automata for the sup-projection to obtain $\sup_{s \in \mathrm{padext}(t)} [\![\mathcal{A}]\!]_S(s)$.

Thus, for a given tree t and a cost tree automaton $\mathcal{A}$, we aim at modifying $\mathcal{A}$ such that it can simulate all runs on trees from padext (t) that contribute to the overall value. For this purpose, we consider the membership game on the infinite tree $t_\square$ that is labeled with $\square$ at all positions. Since all positions in this tree look the same, there is no need to keep track of the position in the tree. Moreover, Eve decides in each round whether the current node is treated as an inner node or as a leaf (since Eve has to reach the goal set, she has to decide for a leaf at some point). We call this the *padding game* for $\mathcal{A}$ and denote it by $\mathcal{M}_{\mathcal{A},\square}$. We now investigate the “concatenation” of the cost membership game followed by the padding game.

Let $(\mathcal{M}_{\mathcal{A},t} \triangleright \mathcal{M}_{\mathcal{A},\square})$ consist of the union of $\mathcal{M}_{\mathcal{A},t}$ and $\mathcal{M}_{\mathcal{A},\square}$ with the following additional connecting edges: from a position (q,u) of $\mathcal{M}_{\mathcal{A},t}$ with a leaf u of t, Eve can decide to stay inside $\mathcal{M}_{\mathcal{A},t}$ and finish the game as usual, or to treat u as an inner node. In the latter case, the play would reach a position (q', ui) for a node ui not in the domain of t. Instead, the play jumps to $\mathcal{M}_{\mathcal{A},\square}$ in state q'.

▶ **Lemma 12.** *Let $\mathcal{A}$ be a nondeterministic cost tree automaton. We have:*

- $\mathrm{val}_B((\mathcal{M}_{\mathcal{A},t} \triangleright \mathcal{M}_{\mathcal{A},\square})) = \inf_{s \in \mathrm{padext}(t)} [\![\mathcal{A}]\!]_B(s)$
- $\mathrm{val}_S((\mathcal{M}_{\mathcal{A},t} \triangleright \mathcal{M}_{\mathcal{A},\square})) = \sup_{s \in \mathrm{padext}(t)} [\![\mathcal{A}]\!]_S(s)$

With this knowledge and the observation that $\mathcal{M}_{\mathcal{A},\square}$ does not depend on the input tree t, we develop methods to precompute information on $\mathcal{M}_{\mathcal{A},\square}$ with the goal of providing a modified automaton $\mathcal{A}'$ whose membership game $\mathcal{M}_{\mathcal{A}',t}$ approximates (in the sense of $\approx$ equivalence) the combined game $(\mathcal{M}_{\mathcal{A},t} \triangleright \mathcal{M}_{\mathcal{A},\square})$.

First, we consider the case of inf. This case is easier due to the inherent asymmetry in our setting. We claim that it is sufficient to know from which positions $(q, \square)$ Eve can win $\mathcal{M}_{\mathcal{A},\square}$ when we interpret this as a simple reachability game with goal set F. This has the following justification: In B-games every counter is always checked after an increment. Thus, the value of a play can only increase and Eve should just reach a final position as fast as possible. If she just plays the normal reachability strategy she reaches F in at most as many steps as the size of $\mathcal{M}_{\mathcal{A},\square}$ (denoted by $|\mathcal{M}_{\mathcal{A},\square}|$). In the worst case, every of these steps increments the counter. However, even if she could have avoided some of these increments, the error w.r.t. an optimal strategy is at most $|\mathcal{M}_{\mathcal{A},\square}|$. Thus, we obtain:

▶ **Lemma 13.** *Let $\mathcal{A}$ be a synchronous B-tree transducer over $\Sigma^{\otimes m}$. One can construct a synchronous B-tree transducer $\mathcal{A}'$ such that: $[\![\mathcal{A}']\!]_B(t) \approx \inf_{s \in \mathrm{padext}(t)} [\![\mathcal{A}]\!]_B(s)$.*

The sup-case requires more sophisticated methods for two major reasons: First, the independent use of increment and check prevent an argument as before. Secondly, there is no single strategy witnessing $\mathrm{val}_S(\mathcal{G}) = \infty$. Moreover, we recognize that it does not suffice to compute the value of $\mathcal{M}_{\mathcal{A},\square}$ to capture the behavior of $(\mathcal{M}_{\mathcal{A},t} \triangleright \mathcal{M}_{\mathcal{A},\square})$ in an automaton. In the combined game, the counters are not initialized with 0 at the beginning of the $\mathcal{M}_{\mathcal{A},\square}$-part but inherit the current counter values from $\mathcal{M}_{\mathcal{A},t}$. So a direct $\mathtt{cr}$ at the beginning of $\mathcal{M}_{\mathcal{A},\square}$ ensures that Adam can always achieve value 0 if $\mathcal{M}_{\mathcal{A},\square}$ is considered individually. But in the combined game the counter may have a large value from $\mathcal{M}_{\mathcal{A},t}$. If he could first reset the counter before checking it, this would be much better.

We approach this problem by a more detailed analysis of strategies of Adam in S-cost games. Due to space restrictions, we can only provide the cornerstones of the proof strategy here. Let σ be a strategy of Adam in an S-cost game $\mathcal{G}$. We remind the reader that Adam wants to check counters with *small* values or avoid F. Since we can handle the reachability of F individually before, we concentrate on the counter values. We measure the success of σ in the following three categories per counter:

1. $\mathtt{cr}$: The strategy σ can enforce a check of the counter after a bounded number of steps.
2. $\mathtt{rcr}$: The strategy σ can enforce a reset before any check and subsequently a check after a bounded number of increments
3. $\bot$: The strategy σ provides no guarantees on the counter.

Formally, such a profile p is a mapping from the set of counters Γ to $\{\mathtt{cr}, \mathtt{rcr}, \bot\}$ and we say that a strategy σ guarantees p if all plays that are played according to σ satisfy the conditions stated in p. We call a profile p *simple* and write, e.g., $[c_1 \mapsto \mathtt{cr}]$ if it provides only a guarantee on one counter – the other counters are mapped to $\bot$. In order to incorporate the choices of Eve in the play, we need combinations of several such profiles to describe a strategy σ. For example, Eve may choose whether she wants to check a counter c_1 or c_2. We express this as a disjunction of profiles (called generalized profile) and write in this example $[c_1 \mapsto \mathtt{cr}] \vee [c_2 \mapsto \mathtt{cr}]$.

Computing all possible generalized profiles for strategies of Adam in an S-cost game is a sufficient precomputation to approximate the complete game $(\mathcal{M}_{\mathcal{A},t} \triangleright \mathcal{M}_{\mathcal{A},\square})$ at the position of a leaf node in $\mathcal{M}_{\mathcal{A},t}$. There are only finitely many generalized profiles. We want to compute for every such generalized profile whether Adam has a strategy that guarantees it. We then want to simulate the behavior of $(\mathcal{M}_{\mathcal{A},t} \triangleright \mathcal{M}_{\mathcal{A},\square})$ in an automaton. To implement this, we use the fact that one can extend nondeterministic cost tree automata to alternating cost tree automata in a similar way as standard alternating tree automata (see [12]). The alternation allows us to represent Adam's choice among his possible generalized profiles followed by Eve's choice among the profiles in the disjunction in the automaton. In the target state, the automaton checks counters that have guarantee $\mathtt{cr}$ and resets and then checks counters that have guarantee $\mathtt{rcr}$. A detailed analysis of S-cost games shows that the bound on the number of increments in the guarantees $\mathtt{cr}$ and $\mathtt{rcr}$ only depends on the size of the game. Since the size of $\mathcal{M}_{\mathcal{A},\square}$ only depends on the size of $\mathcal{A}$, we obtain the following:

▶ **Lemma 14.** *Let $\mathcal{A}$ be a synchronous S-tree transducer over $\Sigma^{\otimes m}$. One can construct a synchronous S-tree transducer $\mathcal{A}'$ such that: $[\![\mathcal{A}']\!]_S(t) \approx \sup_{s \in \mathrm{padext}(t)} [\![\mathcal{A}]\!]_S(s)$.*

The algorithm to compute the possible generalized profiles employs methods from the theory of tree automata on infinite trees. We unfold plays on a finite game graph as an infinite tree and notice that we can approximate strategies that are good for Adam (as they guarantee $\mathtt{cr}$ or $\mathtt{rcr}$) in MSO logic. From the fact that MSO-formulas always have a regular tree as model, we can deduce a bound on the number of increments.

In a last step, we combine all the previous observations and results from the theory of regular cost functions over trees. We saw how to inductively transform an FO+RR$^{=\infty}$-formula φ over a resource-tree-automatic structure into a synchronous cost tree transducer that computes the semantics up to $\approx$. The changes from S- to B-automata (or vice versa) and from alternating to nondeterministic can also be computed effectively up to $\approx$ (cf. [12]). In total, we obtain a transducer $\mathcal{A}$ such that $[\![\mathcal{A}]\!] \approx [\![\varphi]\!]$. If the computed value is ∞, this is even exact. In the other case, we know that $[\![\varphi]\!]$ has a finite value. As described earlier for the case of resource-automatic structures on words, we can similarly compute exact values by a reduction to standard tree-automatic structures for formulas outside of ∞-comparisons.

▶ **Theorem 15.** *The semantics of FO+RR$^{=\infty}$ is effectively computable on resource-tree-automatic structures.*

4.1.2 Deciding the qcWMSO-theory of $\mathfrak{T}_2$

It remains to show that the above result can be applied to establish that the qcWMSO-theory of the infinite binary tree is decidable. We use the same idea as for $(\omega, <)$ to obtain this result. Given the infinite binary tree $\mathfrak{T}_2 = (\{0,1\}^*, S_0, S_1)$, we consider the following variant of the finite powerset structure $\mathfrak{F}_2 = (\mathrm{FinPot}(\{0,1\}^*), S_0, S_1, \rangle\cdot\langle, \in, = \varnothing)$, where the (resource) relations are as follows:

- $(a,b) \in S_i$ evaluates to ∞ if both $a = \{a'\}, b = \{b'\}$ are singletons and $b' = a'i$. Otherwise, it evaluates to 0.
- For a set $a \subseteq \{0,1\}^*$, $\rangle a \langle$ evaluates to an approximation of the cardinality of a that can easily be computed by a cost automaton (see below for a further explanation), and is defined as follows:

$$\rangle a \langle = \sup_{w \in a} \left(|\{ i \mid \exists u \in a : w[1\dots i] \leq u \wedge w[1 \dots i+1] \nleq u \}| \right).$$

- $a \in b$ evaluates to ∞ if $a = \{a'\}$ is a singleton such that $a' \in b$, and to 0 otherwise.
- $a = \varnothing$ evaluates to ∞ if a is empty, and to 0 otherwise.

Note that $\rangle\cdot\langle$ in the above structure is different from the evaluation of $|\cdot|$ in qcWMSO. However, boundedness is preserved because we have $\rangle a \langle \leq [\![|a|]\!]^{\mathfrak{T}_2} \leq 2^{\rangle a \langle}$: The first inequality comes from the fact that each element of a can contribute at most 1 to the value of $\rangle a \langle$. To see the second one, consider the following path w for the supremum: Always proceed to the subtree that contains more than half of the remaining elements of a. This counts one if the non-selected subtree was not empty but loses at most half of the remaining elements.

We now claim that $\mathfrak{F}_2$ is resource-tree-automatic. As universe we consider the set $S \subseteq \mathcal{T}_{\{0,1\}}$ of finite binary trees with the property that every inner node of the tree is either labeled with 1 or has a 1-labeled node as descendant. This property can easily be checked by a tree automaton. A labeled tree t corresponds to the subset of $\mathfrak{T}_2$ of those positions in t that are labeled with 1. The condition on the trees in the universe ensures a unique encoding of every set. Clearly, S_0, S_1 are automatic, and so are $\in$ and $= \varnothing$, as they are tree-automatic in the classical sense. The relation $\rangle\cdot\langle$ can be implemented with an S-automaton that simulates walks as described above by nondeterministically guessing and verifying the positions where the not-selected subtree contains elements of a.

Using the same translation as for ω, we reduce the problem of deciding the theory of qcWMSO to the boundedness problem for FO+RR$^{=\infty}$. This is possible as $\rangle\cdot\langle$ in the above sense preserves boundedness.

▶ **Theorem 1.**
(b) *Let φ be a qcWMSO-sentence. It is decidable whether $[\![\varphi]\!]^{\mathfrak{T}_2} = \infty$.*

Although exact FO+RR$^{=\infty}$-evaluations can be computed on resource-tree-automatic structures, this does not entail that qcWMSO can be evaluated exactly on $\mathfrak{T}_2$. This shortcoming has its origin in the fact that cost tree automata can count only along paths. To have an exact reduction from qcWMSO to FO+RR$^{=\infty}$, counts of different paths would have to be combined to simulate counting the size of an arbitrary subset of $\{0,1\}^*$.

5 Conclusion

We introduced the logic qcMSO as a quantitative MSO variant to specify boundedness properties. The logics costMSO and MSO+U can be embedded into qcMSO in a natural way. Thus, the undecidability of MSO+U on $(\omega, <)$ already shows that the semantics of qcMSO on $(\omega, <)$ and the infinite binary tree $\mathfrak{T}_2$ is not computable. Accordingly, we focused on the weak variant qcWMSO and provide a method to compute the value of qcWMSO sentences on $(\omega, <)$ and approximations of the value on $\mathfrak{T}_2$. This result is achieved by a reduction to a cost-function extension of automatic structures – called resource-automatic structures. Moreover, we lift the known results to resource-tree-automatic structures.

In the future, we would like to see whether there is an automaton model for qcWMSO and whether there are meaningful fragments of full qcMSO with good algorithmic properties.

References

1 A. Blumensath. Automatic Structures. Diploma thesis, RWTH-Aachen, 1999.
2 A. Blumensath and E. Grädel. Automatic Structures. In *LICS 2000*, pages 51–62, 2000.
3 M. Bojanczyk. Weak MSO with the Unbounding Quantifier. In *STACS 09*, volume 3, pages 159–170, 2009.
4 M. Bojańczyk. Weak MSO with the unbounding quantifier. *Theory of Computing Systems*, 48(3):554–576, 2011.
5 M. Bojańczyk, T. Gogacz, H. Michalewski, and M. Skrzypczak. On the decidability of MSO+U on infinite trees. In *Automata, Languages, and Programming*, volume 8573 of *LNCS*, pages 50–61. Springer, 2014.
6 M. Bojańczyk, P. Parys, and S. Toruńczyk. The MSO+U theory of $(\mathbb{N}, <)$ is undecidable. *arXiv:1502.04578 [cs.LO]*, 2015.
7 M. Bojanczyk and S. Torunczyk. Weak MSO+U over infinite trees. In *STACS 2012*, volume 14, pages 648–660, Dagstuhl, Germany, 2012.
8 M. Bojańczyk and S. Toruńczyk. Weak MSO+U over infinite trees. In Christoph Dürr and Thomas Wilke, editors, *STACS 2012*, volume 14, pages 648–660, 2012.
9 T. Colcombet. Regular cost functions over words. *Manuscript available online*, 2009.
10 T. Colcombet. The theory of stabilisation monoids and regular cost functions. In *Automata, Languages and Programming*, volume 5556 of *LNCS*, pages 139–150. Springer, 2009.
11 T. Colcombet. Regular cost functions, part I: logic and algebra over words. *Logical Methods in Computer Science*, 9(3), 2013.
12 T. Colcombet and C. Löding. Regular cost functions over finite trees. In *LICS 2010*, pages 70–79, July 2010.
13 H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available online: `http://www.grappa.univ-lille3.fr/tata`, 2007. release October, 12th 2007.
14 D. Fischer, E. Grädel, and Ł. Kaiser. Model Checking Games for the Quantitative mu-Calculus. *Theory Comput. Syst.*, 47(3):696–719, 2010.
15 S. Hummel and M. Skrzypczak. The topological complexity of MSO+U and related automata models. *Fundamenta Informaticae*, 119(1):87–111, 2012.
16 B. Khoussainov and A. Nerode. Automatic presentations of structures. In *Logic and Computational Complexity*, volume 960 of *LNCS*, pages 367–392. Springer, 1995.
17 D. Kuperberg and M. Vanden Boom. On the expressive power of cost logics over infinite words. In *ICALP 2012*, volume 7392 of *LNCS*, pages 287–298. Springer, 2012.
18 M. Lang and C. Löding. Modeling and verification of infinite systems with resources. *Logical Methods in Computer Science*, 9(4), 2013.

19 M. Lang, C. Löding, and A. Manuel. Definability and transformations for cost logics and automatic structures. In *MFCS 2014*, volume 8634 of *LNCS*, pages 390–401. Springer, 2014.
20 W. Thomas. Languages, automata, and logic. In *Handbook of Formal Language Theory*, volume III, pages 389–455. Springer, 1997.
21 M. Vanden Boom. Weak cost monadic logic over infinite trees. In *MFCS 2011*, volume 6907 of *LNCS*, pages 580–591. Springer, 2011.