

The Decidability of the First-order Theory of Knuth-Bendix Order

Ting Zhang

Microsoft Research Asia

and

Henny B. Sipma, Zohar Manna

Stanford University

Two kinds of orderings are widely used in term rewriting and theorem proving, namely *recursive path ordering* (RPO) and *Knuth-Bendix ordering* (KBO). They provide powerful tools to prove the termination of rewriting systems. They are also applied in ordered resolution to prune the search space without compromising refutational completeness. Solving ordering constraints is therefore essential to the successful application of ordered rewriting and ordered resolution. Besides the needs for decision procedures for quantifier-free theories, situations arise in constrained deduction where the truth value of quantified formulas must be decided. Unfortunately, the full first-order theory of lexicographic path ordering (LPO), the most popular form of RPO, is undecidable. This leaves an open question whether the first-order theory of KBO is decidable. In this paper, we give a positive answer to this question using quantifier elimination.

Categories and Subject Descriptors: F.4.1 [Theory of Computation]: Mathematical Logic and Formal Languages—*Mathematical Logic*

General Terms: Theory, Algorithms

Additional Key Words and Phrases: Decidability, Knuth-Bendix order, term algebras, Presburger arithmetic, quantifier elimination

1. INTRODUCTION

Two kinds of orderings are widely used in term rewriting and theorem proving. One is *recursive path ordering* (RPO) which is based on syntactic precedence [Dershowitz 1982]. The other is *Knuth-Bendix ordering* (KBO) which is of hybrid nature; it relies on numerical values assigned to symbols as well as syntactic precedence [Knuth and Bendix 1970]. In ordered term rewriting, a strategy built on ordering constraints can dynamically orient an equation, at the time of instantiation, even if the equation is not uniformly orientable. This provides a powerful tool to prove the termination of rewriting systems [Comon and Treinen 1994]. In ordered resolution and paramodulation, ordering constraints are used to select maximal literals to perform resolution. It also serves as enabling conditions for inference rules and such conditions can be inherited from previous inferences at each deduction step. This helps to prune redundancy of the search space without compromising refutational completeness [Nieuwenhuis and Rubio 1995].

Solving ordering constraints is therefore essential to the successful application of ordered rewriting and ordered resolution. The decision procedures for quantifier-free constraints of both types of orderings have been well-studied [Comon 1990; Jouannaud and Okada 1991; Nieuwenhuis 1993; Narendran et al. 1999; Nieuwenhuis and Rivero 1999; Korovin and Voronkov 2000; 2001]. However, situations arise where we need to decide the truth values of quantified formulas on those orderings, especially in the $\forall^* \exists^*$ fragment. Examples include checking the soundness

of simplification rules in constrained deduction. Consider a “total simplification scheme” given in [Kirchner et al. 1990; Comon and Treinen 1997],

$$\frac{s \rightarrow t \mid c}{s[v]_p \rightarrow t \mid (c \wedge c' \wedge s|_p = u)} \quad (u \rightarrow v \mid c') ,$$

where $s|_p$ denotes the subterm occurring at position p in s and $s[v]_p$ denotes the term obtained from s by substituting v for $s|_p$. It states that $s \rightarrow t \mid c$ is simplified to $s[v]_p \rightarrow t \mid (c \wedge c' \wedge s|_p = u)$ by $u \rightarrow v \mid c'$ provided *for all* assignments for variables in s that satisfy c , *there exists* an assignment for variables in u that satisfies c' and $s|_p = u$. The soundness of this rule is formally expressed as

$$\text{TA} \models \forall \mathcal{V}(s) \exists \mathcal{V}(u) \left[c \rightarrow (c' \wedge s|_p = u) \right] ,$$

which necessarily involves quantifier alternation. To determine the soundness of such simplification rules, we need to be able to reason in the $\forall^* \exists^*$ fragment.

Unfortunately, the full first-order theory of lexicographic path ordering (LPO), the most popular form of RPO, is undecidable [Treinen 1992; Comon and Treinen 1997] except for the special case where the language only has unary functions and the precedence order is total [Narendran and Rusinowitch 2000]. Until now it has been an open question whether the first-order theory of Knuth-Bendix order is decidable (RTA open problem #99). Here we answer this question affirmatively by showing that an extended theory of term algebras with Knuth-Bendix order admits quantifier elimination.

The basic framework is the combination of term algebras with Presburger arithmetic [Zhang et al. 2006]. The extended language has two sorts; the integer sort \mathbb{Z} and the term sort TA. Intuitively, the language is the set-theoretic union of the language of term algebras and the language of Presburger arithmetic. Formulas are formed from *term literals* and *integer literals* using logical connectives and quantifications. The combination presented in this paper is more tightly coupled than $\text{TA}_{\mathbb{Z}}$ presented in [Zhang et al. 2006]: not only do we have a *weight function* mapping terms to integers, but we also have various *boundary functions* mapping integers to terms. In addition, the Knuth-Bendix order is expanded in two directions. First, the order is decomposed into three disjoint suborders depending on which of three conditions is used in the definition. Secondly, all orders (including the suborders) are extended to gap orders, which assert the least number of distinct objects between two terms. Moreover, as Knuth-Bendix order is recursively defined on a lexicographic extension of itself, gap orders are extended to tuples of terms. Thus we actually establish the decidability of a richer theory.

The result was first published, without proofs, in [Zhang et al. 2005]. This paper provides an extended presentation and includes all the proofs.

1.1 Related Work and Comparison.

Presburger arithmetic (PA) was first shown to be decidable in 1929 by the quantifier elimination method [Enderton 2001]. Efficient algorithms were later discovered by Cooper [Cooper 1972] and further improved in [Reddy and Loveland 1978].

The decidability of the first-order theory of term algebras was first shown by Mal'cev using quantifier elimination [Mal'cev 1971]. This result was proved again

later in different settings [Maher 1988; Comon and Lescanne 1989; Hodges 1993; Comon and Delor 1994; Backofen 1995; Rybina and Voronkov 2001; Kuncak and Rinard 2003a; 2003b; Zhang et al. 2004a; 2004b].

Quantifier elimination has been used to obtain decidability results for various extensions of term algebras. [Maher 1988] shows the decidability of the theory of infinite and rational trees. [Comon and Delor 1994] presents an elimination procedure for term algebras with membership predicate in the regular tree language. [Backofen 1995] presents an elimination procedure for structures of feature trees with arity constraints. [Rybina and Voronkov 2001] shows the decidability of term algebras with queues. [Kuncak and Rinard 2003b] shows the decidability of term powers, which are term algebras augmented with coordinatewise-defined predicates. [Zhang et al. 2004a] extends the quantifier elimination procedure in [Hodges 1993] for term algebras with constant weight function.

The decidability of the theory of RPO has been well-studied. Comon proves the decidability of the quantifier-free theory of total lexicographic path ordering (LPO, a variant of RPO) [Comon 1990]. A similar result holds for RPO [Jouannaud and Okada 1991]. Nieuwenhuis establishes the NP-completeness for the quantifier-free theory of LPO [Nieuwenhuis 1993]. Narendran, Rusinowitch and Verma obtain a similar result for RPO [Narendran et al. 1999]. A more efficient algorithm for the quantifier-free theory of RPO is given by Nieuwenhuis and Rivero [Nieuwenhuis and Rivero 1999]. Comon and Treinen show the undecidability of the first-order theory of LPO and the undecidability of the first-order theory of RPO in case of partial precedence [Treinen 1992; Comon and Treinen 1997]. The decidability of the first-order theory of RPO (LPO) in case of unary signature and total precedence is due to Narendran and Rusinowitch [Narendran and Rusinowitch 2000]. The decidability of the first-order theory of RPO in case of total precedence remains open.

Recently some partial decidability results for the theory of KBO have been obtained. Korovin and Voronkov show the decidability of the quantifier-free theory of term algebras with KBO [Korovin and Voronkov 2000]. They later improve the algorithm and show that the quantifier-free theory of KBO is NP-complete [Korovin and Voronkov 2001]. Analogous to [Narendran and Rusinowitch 2000], they also show the decidability of the first-order theory of KBO in the case where all functions are unary [Korovin and Voronkov 2002].

In this paper, we show the general decidability result for an extended theory of KBO with arbitrary function symbols and weight functions. Our method combines the extraction of integer constraints from term constraints with a reduction of quantifiers on term variables to quantifiers on integer variables. The quantifier elimination procedure makes extensive use of Presburger arithmetic in the reduction.

Reduction of constraints on data structures to constraints on integers has long been a powerful tool to derive decidability results on different domains. This line of investigation goes back to Skolem who showed the decidability of the first-order theory of Boolean algebras by reducing constraints on sets to constraints on the cardinality of sets [Skolem 1970]. It readily follows from the reduction technique that the first order theory of sets with cardinality constraints in Presburger arithmetic

is decidable [Feferman and Vaught 1959]. Recently, Revesz [Revesz 2004], and Kuncak and Rinard [Kuncak and Rinard 2005] independently presented decision procedures for this theory. A combination of Presburger arithmetic and term algebras was used by Korovin and Voronkov to show that the quantifier-free theory of term algebras with Knuth-Bendix order is NP-complete [Korovin and Voronkov 2000; 2001].

1.2 Paper Organization.

Section 3 defines term algebras. Section 4 introduces the theory of term algebras with Knuth-Bendix ordering and presents the technical machinery for eliminating quantifiers. Section 5 presents the main contribution of this paper: it expands the elimination procedure in [Zhang et al. 2004a] for the extended theory of KBO and proves its correctness. Section 6 explains how to adapt the elimination procedure to the special case where the language contains a unary function of weight 0. Section 7 concludes with some ideas for future work. Most proofs are given in Appendix A. The proofs are supported by integer predicates and functions defined in Appendix B and reductions of gap order literals listed in Appendix C.

2. PRELIMINARIES

We assume the first-order syntactic notions of variables, parameters and quantifiers, and semantic notions of structures, satisfiability and validity as in [Enderton 2001].

Syntax

A *signature* Σ is a set of *function symbols* and *predicate symbols* each of which is associated with an arity. The function symbols with *arity* 0 are also called *constants*. The set of Σ -*terms* $\mathcal{T}(\Sigma, \mathcal{X})$ is recursively defined by: (i) every constant $c \in \Sigma$ or variable $x \in \mathcal{X}$ is a term, and (ii) if $f \in \Sigma$ is an n -place function symbol and t_1, \dots, t_n are terms, then $f(t_1, \dots, t_n)$ is a term. We write $\mathcal{T}(\Sigma)$ for $\mathcal{T}(\Sigma, \emptyset)$. Equality = is always included as a binary predicate symbol. If φ is a formula, we use $\mathcal{T}(\varphi)$ to denote the set of terms occurring in φ , and $\mathcal{V}(\varphi)$ to denote the set of variables in φ .

An *atomic formula* is a formula of the form $P(t_1, \dots, t_n)$ where P is an n -place predicate symbol and t_1, \dots, t_n are terms. A *literal* is an atomic formula or its negation. A variable occurs *free* in a formula if it is not in the scope of a quantifier. A formula without quantifiers is called *quantifier-free*. A *ground formula* is a formula with no variables. A *sentence* is a formula in which no variable occurs free. Every quantifier-free formula can be put into *disjunctive normal form*, that is, a disjunction of conjunctions of literals.

We use \vec{x} to denote a sequence of variables, say, x_1, \dots, x_n , and $\exists \vec{x}$ (resp. $\forall \vec{x}$) as an abbreviation of $\exists x_1, \dots, \exists x_n$ (resp. $\forall x_1, \dots, \forall x_n$). We write $\varphi(\vec{x})$ to indicate \vec{x} occur freely in φ , without exclusion of other free variables that are not of interest in the context. A formula $\psi(\vec{x})$ can be put into *prenex form* $Q_1 y_1, \dots, Q_n y_n \varphi(\vec{x}, y_1, \dots, y_n)$, where Q_i 's are either \exists or \forall and $\varphi(\vec{x}, y_1, \dots, y_n)$ is quantifier-free, called the *matrix* of ψ . In case \vec{x} contain all free variables in $\varphi(\vec{x})$, $(\exists \vec{x})\varphi(\vec{x})$ and $(\forall \vec{x})\varphi(\vec{x})$ are called *existential closure* and *universal closure* of $\varphi(\vec{x})$, respectively. If, in addition, $\varphi(\vec{x})$ is quantifier-free, then $(\exists \vec{x})\varphi(\vec{x})$ is called \exists_1 and $(\forall \vec{x})\varphi(\vec{x})$ is called \forall_1 . The quantifier-free (resp. \exists_1, \forall_1) fragment of a language is the subclass of quantifier-free (resp.

\exists_1, \forall_1) sentences in the language.

An *expression* E is either a formula or a term. We use $E[e_1, \dots, e_n]$ to denote that e_1, \dots, e_n are subexpressions of E .

We use \equiv for syntactic equality. We use interval notation for integer sets. For example, the closed interval $[m, n]$ means $\{i \mid m \leq i \leq n\}$. Similarly for open intervals (m, n) and half-open intervals $[m, n)$ and $(m, n]$. We use $(s_i)_{i \in I}$ to denote a sequence indexed by I . For example, by $(e_i)_{i < n}$ we mean e_0, \dots, e_n . We use the standard notations for partial orders. Suppose \triangleleft is a partial order, then by $x \trianglelefteq y$ we mean $x = y \vee x \triangleleft y$. If \triangleleft is transitive, by $x_1 \triangleleft x_2 \triangleleft \dots \triangleleft x_n$ we mean $\bigwedge_{0 < i < n} x_i \triangleleft x_{i+1}$.

Semantics

A Σ -*structure* \mathfrak{A} is a tuple $\langle A, I \rangle$ where A is a non-empty *domain* and I is a function that associates each n -place function symbol f (resp. predicate symbol P) with an n -place function $f^{\mathfrak{A}}$ (resp. relation $P^{\mathfrak{A}}$) on A . We use Gothic letters (like \mathfrak{A}) for structures and Roman letters (like A) for the underlying domain. We usually denote \mathfrak{A} by $\langle A; \Sigma \rangle$ which is called the *signature* of \mathfrak{A} . A *variable assignment* σ (in \mathfrak{A}) is a function that assigns each variable an element of A . We use $\llbracket x \rrbracket_{\sigma}$ to denote the assigned value of x under σ and $\llbracket \varphi \rrbracket_{\sigma}$ for the truth value of φ under σ . $\mathfrak{A} \models \llbracket \varphi \rrbracket_{\sigma}$ means φ is true under σ . A formula φ is *satisfiable* (in \mathfrak{A}), denoted by $\mathfrak{A} \models_{\exists} \varphi$, if $\mathfrak{A} \models \llbracket \varphi \rrbracket_{\sigma}$ for some σ ; is *unsatisfiable* (in \mathfrak{A}), denoted by $\mathfrak{A} \not\models_{\exists} \varphi$, if $\mathfrak{A} \models \llbracket \varphi \rrbracket_{\sigma}$ for no σ ; is *valid* (in \mathfrak{A}), denoted by $\mathfrak{A} \models \varphi$, if $\mathfrak{A} \models \llbracket \varphi \rrbracket_{\sigma}$ for any σ . A formula φ is valid if and only if $\neg \varphi$ is unsatisfiable.

\mathfrak{A} is a *model* of a set T of sentences if every sentence in T is true in \mathfrak{A} . A sentence φ is (*logically*) *implied* by T (or *T -valid*), written $T \models \varphi$, if φ is true in every model of T . Similarly, we say that φ is *T -satisfiable* if φ is true in some model of T and it is *T -unsatisfiable* otherwise. The notions of (T -)validity and (T -)satisfiability naturally extend to a set of formulas. A *theory* T is a set of sentences that is closed under logical implication, that is, if $T \models \varphi$, then $\varphi \in T$. The theory of \mathfrak{A} , written $\text{Th}(\mathfrak{A})$, is the set of all true sentences in \mathfrak{A} . By a *quantifier-free theory* of \mathfrak{A} , written $\text{Th}^{\forall}(\mathfrak{A})$, we mean the quantifier-free subclass of $\text{Th}(\mathfrak{A})$.

By satisfiability and validity of a quantifier-free formula, we actually mean the validity of the corresponding \exists_1 and \forall_1 formulas, respectively. Similarly, the satisfiability problem and the validity problem of a quantifier-free theory (or more precisely, the quantifier-free fragment), respectively, refer to the validity problem of the corresponding \exists_1 fragment and \forall_1 fragment.

Multisorted Logic

All above notions naturally generalize to multi-sorted logic. In a multi-sorted logic, we have a non-empty set S of *sorts*. Variables, constants, equality symbols and quantifiers are indexed by $s \in S$. An n -ary function symbol f is associated with an $n+1$ -tuple $\langle s_1, \dots, s_{n+1} \rangle$, called the *type* of f . Similarly, the type of an n -ary predicate symbol p is an n -tuple $\langle s_1, \dots, s_n \rangle$. For $0 < i \leq n$, we say that the i^{th} -place of f (or p) has sort s_i . We require the type of the equality symbol of sort s be $\langle s, s \rangle$. A term t is of sort s if (i) t is a variable or a constant of sort s , or (ii) t is of the form $f(t_1, \dots, t_n)$ such that the type of f is $\langle s_1, \dots, s_n, s \rangle$, and t_1, \dots, t_n are of sorts s_1, \dots, s_n , respectively. A formula is well-formed if in addition it is *well-typed* in the sense that (i) a term of sort s only occurs in a place of sort s in a function or predicate

symbol, and (ii) variables of sort s are only quantified by \forall_s and \exists_s .

A multi-sorted structure \mathfrak{A} is a tuple $\langle \{A\}_S, S, I \rangle$ where S is the set of sorts, $\{A\}_S$ are mutually disjoint sets (domains) indexed by S , and I is an interpretation such that (i) each n -ary function symbol f with type $\langle s_1, \dots, s_{n+1} \rangle$ is assigned a function $F : A_{s_1} \times \dots \times A_{s_n} \rightarrow A_{s_{n+1}}$; (ii) each n -ary predicate symbol p with type $\langle s_1, \dots, s_n \rangle$ is assigned a relation $P \subseteq A_{s_1} \times \dots \times A_{s_n}$. A variable assignment assigns a variable of sort s an element in A_s . Satisfiability, unsatisfiability and validity are defined as above with \forall_s (resp. \exists_s) being interpreted as “for all (resp. some) elements in the domain A_s ”.

Quantifier Elimination

A theory T is said to *admit quantifier elimination* if any formula can be equivalently (modulo T) and effectively transformed into a quantifier-free formula. If a theory admits quantifier elimination, then the truth value of any sentence is reducible to the truth value of a ground formula.

It is well-known that eliminating arbitrary quantifiers reduces to eliminating existential quantifiers from formulas in the form

$$(\exists \vec{x}) \left[A_1(\vec{x}, \vec{y}) \wedge \dots \wedge A_n(\vec{x}, \vec{y}) \right], \quad (1)$$

where $A_i(\vec{x}, \vec{y})$ ($0 < i \leq n$) are literals [Hodges 1993]. In discussions related to quantifier elimination, a quantified formula always refers to a formula of the form (1). We also assume that the literals A_i are not of the form $x = t$ when x does not appear in t . For $\exists x(x = t \wedge \Phi(x, \vec{y}))$ simplifies to $\Phi(t, \vec{y})$.

Nondeterminism

We present algorithms in a nondeterministic manner; whenever we say “guess ϕ ”, we mean to add a valid (w.r.t. the context) disjunction $\bigvee_i \phi_i$ (where ϕ is one of the disjuncts) to the matrix of (1). When we replace ϕ by $\bigvee_i \phi_i$ or directly introduce $\bigvee_i \phi_i$, it should be understood that an implicit disjunctive splitting is carried out and we work on each resultant disjunct of the form (1) “simultaneously”. We call the target formula *redex* and the resulting formula *reduct*.

Presburger Arithmetic

Presburger arithmetic is the first-order theory of addition in the arithmetic of integers. The corresponding structure is denoted by $\text{PA} = \langle \mathbb{Z}; 0, +, < \rangle$. We use $\mathcal{L}_{\mathbb{Z}}$ to denote the formal language of PA.

3. TERM ALGEBRAS

We present a general language and structure of term algebras. For simplicity, we do not distinguish syntactic terms in the language from semantic terms in the corresponding structure. The meaning should be clear from the context.

Definition 3.1 Term Algebras. A *term algebra* $\text{TA} : \langle \mathbb{T}; C, \mathcal{A}, S, \mathcal{T} \rangle$ consists of

- (1) \mathbb{T} : The *term domain*, which exclusively consists of terms recursively built up from constants by applying non-nullary constructors. Objects in \mathbb{T} are called *TA-terms*. The type of a term t , denoted by $\text{type}(t)$, is the outermost constructor symbol of t . We say that t is α -typed (or is an α -term) if $\text{type}(t) = \alpha$.

- (2) C : A finite set of constructors: $\alpha, \beta, \gamma, \dots$. The arity of α is denoted by $\text{ar}(\alpha)$.
- (3) \mathcal{A} : A finite set of constants: a, b, c, \dots . We require $\mathcal{A} \neq \emptyset$ and $\mathcal{A} \subseteq C$. For $a \in \mathcal{A}$, $\text{ar}(a) = 0$ and $\text{type}(a) = a$.
- (4) \mathcal{S} : A finite set of selectors. For a constructor α with arity $k > 0$, there are k selectors $s_1^\alpha, \dots, s_k^\alpha$ in \mathcal{S} . We call s_i^α ($0 < i \leq k$) the i^{th} α -selector. For a term x , $s_i^\alpha(x)$ returns the i^{th} component of x if x is an α -term and x itself otherwise.
- (5) \mathcal{T} : A finite set of testers. For each constructor α there is a corresponding tester Is_α . For a term x , $\text{Is}_\alpha(x)$ is true if and only if x is an α -term. For a constant a , $\text{Is}_a(x)$ is just $x = a$. In addition there is a special tester Is_A such that $\text{Is}_A(x)$ is true if and only if x is a constant.

We use \mathcal{L}_T to denote the language of term algebras.

A term algebra has two essential properties: (i) the domain \mathbb{T} is *exclusively* generated by recursively applying constructors; (ii) each object in \mathbb{T} is *uniquely* constructed. Note that the term domain \mathbb{T} is the *ground* Herbrand domain $\mathcal{T}(C)$ in the language consisting of only constructors (i. e., $\Sigma = C$). Selectors and testers are introduced into the *formal* language for our study.

EXAMPLE 3.2 LISP LISTS. *Consider the LISP list structure*

$$\text{List} = \langle \text{list}; \{\text{cons}, \text{nil}\}, \{\text{nil}\}, \{\text{car}, \text{cdr}\}, \{\text{Is}_{\text{cons}}, \text{Is}_{\text{nil}}, \text{Is}_A\} \rangle$$

where *list* denotes the domain, *nil* denotes the empty list, *cons* is a binary constructor (pairing function) and *car* and *cdr* are the corresponding left and right selectors (projectors) respectively. It is not difficult to verify that *List* is an instance of term algebras.

The theory of term algebras is axiomatizable. Let \bar{z}_α denote $z_1, \dots, z_{\text{ar}(\alpha)}$. The following formula schemes, in which variables are implicitly universally quantified over \mathbb{T} , axiomatize $\text{Th}(\text{TA})$.

- A1. $t(x) \neq x$, if t is built solely by constructors and t properly contains x .
- A2. $\alpha(x_1, \dots, x_{\text{ar}(\alpha)}) \neq \beta(y_1, \dots, y_{\text{ar}(\beta)})$, if $\alpha, \beta \in C$ and $\alpha \neq \beta$.
- A3. $\alpha(x_1, \dots, x_{\text{ar}(\alpha)}) = \alpha(y_1, \dots, y_{\text{ar}(\alpha)}) \rightarrow \bigwedge_{1 \leq i \leq \text{ar}(\alpha)} x_i = y_i$.
- A4. $\text{Is}_\alpha(x) \leftrightarrow \exists \bar{z}_\alpha \alpha(\bar{z}_\alpha) = x$ for $\alpha \in C$. In particular, $\text{Is}_a(x) \leftrightarrow x = a$ for $a \in \mathcal{A}$.
- A5. $\text{Is}_A(x) \leftrightarrow \bigvee_{a \in \mathcal{A}} \text{Is}_a(x)$.
- A6. $s_i^\alpha(x) = y \leftrightarrow \exists \bar{z}_\alpha (\alpha(\bar{z}_\alpha) = x \wedge y = z_i) \vee (\forall \bar{z}_\alpha (\alpha(\bar{z}_\alpha) \neq x) \wedge x = y)$.

This set of axioms is a variant of the axiomatization given in [Hodges 1993]. In general, selectors and testers can be defined by constructors and vice versa. One direction has been shown by (A4), (A5) and (A6), which are purely definitional axioms. The other direction follows from the equivalence of $\bigwedge_{i=1}^k s_i^\alpha(x) = x_i \wedge \text{Is}_\alpha(x)$ and $x = \alpha(x_1, \dots, x_k)$.

We write $\alpha = (s_1^\alpha, \dots, s_k^\alpha)$ to indicate that α is a non-nullary constructor with $\text{ar}(\alpha) = k$ and $s_1^\alpha, \dots, s_k^\alpha$ are the corresponding selectors of α . A term t is called a *constructor term* if t is a variable or the outermost function symbol of t is a constructor. A constructor term not containing selectors is called *pure*. For example, constants are pure constructor terms. A term t is called a *selector term* if either t is a variable or the outermost function symbol of t is a selector. Note that variables

are both constructor terms and selector terms. We assume that no constructors appear immediately inside selectors as simplification can always be done. For example, $s_i^\alpha(\alpha(x_1, \dots, x_k))$ simplifies to x_i ($0 < i \leq k$) and $s_j^\beta(\alpha(x_1, \dots, x_k))$ simplifies to $\alpha(x_1, \dots, x_k)$ for $\alpha \neq \beta$. As a consequence, a selector term has the form $s_1(\dots(s_n(x)\dots))$ for $n \geq 0$. We use L, F, G, H, \dots to denote (possibly empty) selector sequences. So $s_1(\dots(s_n(x)\dots))$ can be abbreviated as Lx for $L = s_1, \dots, s_n$. The *depth* of x in Lx is $|L|$, the length of L . The depth of x in a formula φ is the maximum depth of x in the selector terms in φ , denoted by $\text{depth}^\varphi(x)$. We use $\text{Is}_\alpha(t_1, \dots, t_n)$ as an abbreviation for $\bigwedge_{0 < i \leq n} \text{Is}_\alpha(t_i)$. We say a selector term $s_i^\alpha(t)$ is *proper* in a formula φ if $\text{Is}_\alpha(t)$ is a conjunct of φ . We can make selector terms proper with type information.

Definition 3.3 Type Completeness. A conjunction of literals Φ is *type complete* if for any selector term t occurring in Φ , exactly one type of tester predicate $\text{Is}_\alpha(t)$ ($\alpha \in C \setminus \mathcal{A} \cup \{A\}$) is a conjunct of Φ .

For a type complete Φ containing a term t , we write $\text{type}(t) = \alpha$ to indicate that $\text{Is}_\alpha(t)$ is a conjunct of Φ . A type complete Φ can be simplified so that any selector term is proper.

EXAMPLE 3.4 TYPE COMPLETENESS. *Let us consider in List the type complete constraint*

$$y \neq \text{cons}(x, \text{car}(x)) \wedge \text{Is}_A(x, \text{car}(x)) \wedge \text{Is}_{\text{cons}}(y) . \quad (2)$$

Thanks to the type information, it can be simplified to

$$y \neq \text{cons}(x, x) \wedge \text{Is}_A(x) \wedge \text{Is}_{\text{cons}}(y) . \quad (3)$$

We could have defined the notion of type completeness only for terms that occur inside selectors. In this way, a type complete formula may contain terms of unspecified types; they are either variables or selector terms that are not embedded inside selectors. This will lead to more efficient algorithms in practice. We choose the above definition, however, because it simplifies descriptions of algorithms presented in the following sections, and in addition, it does not affect the worst-case complexity for those algorithms.

Given a quantifier-free formula φ , we can obtain a type complete φ' from φ by adding exactly one tester predicate $\text{Is}_\alpha(t)$ ($\alpha \in C$) for each term t occurring in φ . We call φ' so obtained a *type completion* of φ .

EXAMPLE 3.5 TYPE COMPLETION. *Let us revisit Example 3.4. It is easily seen that (2) is a type completion of $y \neq \text{cons}(x, \text{car}(x))$.*

EXAMPLE 3.6 TYPE COMPLETION AND SIMPLIFICATION. *Let $\alpha, \beta \in C$, $\alpha \neq \beta$ and $\alpha = (s_1^\alpha)$. A possible type completion for $y = s_1^\alpha(x)$ is*

$$y = s_1^\alpha(x) \wedge \text{Is}_\beta(x, s_1^\alpha(x), y) , \quad (4)$$

which, by Axioms (A4) and (A6), simplifies to $y = x \wedge \text{Is}_\beta(x, y)$. Another type completion is

$$y = s_1^\alpha(x) \wedge \text{Is}_\alpha(x) \wedge \text{Is}_\beta(s_1^\alpha(x), y) , \quad (5)$$

in which the selector term $s_1^\alpha(x)$ is proper and no simplification is possible. As a third possibility, we can have the type completion

$$y = s_1^\alpha(x) \wedge \text{Is}_\alpha(x, s_1^\alpha(x)) \wedge \text{Is}_\beta(y) , \quad (6)$$

which simplifies to false because of type conflicts.

As shown by the above example, a type completion of a satisfiable formula may be contradictory due to type conflicts. A type completion φ' of φ is *compatible* with φ if the satisfiability of φ implies the satisfiability of φ' . Obviously, φ is satisfiable if and only if it has a satisfiable compatible type completion.

In the following sections, we present nondeterministic algorithms that rely on the successful guess of a satisfiable compatible type completion. Unless stated otherwise, we assume that any quantifier-free formula is type complete, and all occurring selector terms are simplified to proper ones. We assume that equalities (resp. disequalities) between terms with conflicting types are simplified to false (resp. true). For example, in List the appearance of $\text{car}(x) \neq y$ should be read as $\text{car}(x) \neq y \wedge \text{Is}_{\text{cons}}(x)$. We also assume that formulas do not have conflicting type literals. For example, we never encounter formulas containing subformulas of the form

$$x \neq \alpha(t_1, \dots, t_{\text{ar}(\alpha)}) \wedge x \neq \beta(t'_1, \dots, t'_{\text{ar}(\beta)})$$

for $\alpha \neq \beta$, because at least one conjunct would have been simplified to true. But to save notation, we omit test literals and treat them as implicit side conditions.

4. TERM ALGEBRAS WITH KNUTH-BENDIX ORDER

In this section we introduce the theory of term algebras with KBO and present the technical machinery needed in the quantifier elimination procedure.

Let Σ be a finite signature in the constructor language (i. e., $\Sigma = C$ in Definition 3.1) and $w : \Sigma \rightarrow \mathbb{N}$ a weight function. We expand $\text{dom}(w)$ to the ground term domain \mathbb{T} by recursively defining $w(\alpha(t_1, \dots, t_k)) = w(\alpha) + \sum_{i=1}^k w(t_i)$. Let $<^\Sigma$ be a linear precedence order on symbols in Σ . We enumerate all symbols in the decreasing $<^\Sigma$ -order such that $\alpha_1 >^\Sigma \alpha_2 >^\Sigma \dots >^\Sigma \alpha_{|\Sigma|}$.

Definition 4.1 Knuth-Bendix Order [Knuth and Bendix 1970]. A Knuth-Bendix order (KBO) $<^{\text{kb}}$ (parameterized with a weight function w and a precedence order $<^\Sigma$) is defined recursively on \mathbb{T} such that for $u, v \in \mathbb{T}$, $u <^{\text{kb}} v$ if and only if one of the following conditions holds:

- (1) $w(u) < w(v)$;
- (2) $w(u) = w(v)$ and $\text{type}(u) <^\Sigma \text{type}(v)$;
- (3) $w(u) = w(v)$, $u \equiv \alpha(u_1, \dots, u_k)$, $v \equiv \alpha(v_1, \dots, v_k)$ and

$$(\exists i) \left[0 < i \leq k \wedge u_i <^{\text{kb}} v_i \wedge \forall j (0 < j < i \rightarrow u_j = v_j) \right] . \quad (7)$$

The KBO $<^{\text{kb}}$ is a well-founded total order on \mathbb{T} [Knuth and Bendix 1970; Baader and Nipkow 1999]. To guarantee well-foundedness, two *compatibility conditions* for w and $<^\Sigma$ are required:

- (i) $w(a) > 0$ for any constant a , and

(ii) a unary function of weight 0, if present, should be the maximum in $<^{\Sigma}$.

Let us denote by \perp the smallest term with respect to $<^{\text{kb}}$. It follows from (i) and (ii) that \perp must be a constant and so it can be determined when w and $<^{\Sigma}$ are given. By (ii) if a unary function of weight 0 exists, it must be unique. For presentation simplicity, we assume that $w(\alpha_1) > 0$. However, the existence of such function in fact considerably simplifies our decision procedure. We defer the discussion to Section 6.

Definition 4.2 Term Algebras with Knuth-Bendix Order. The structure of term algebras with KBO is $\text{TA}_{\text{kb}} = \langle \text{TA}; <^{\text{kb}} \rangle$. Let \mathcal{L}_{kb} denote the language of TA_{kb} .

In the rest of this paper we assume our formal language does not have constructors (except constants). This does not compromise expressiveness as constructors can be defined by selectors and testers. Moreover, this simplification reduces chances of confusion by separating semantic objects (ground constructor terms) in \mathbb{T} from syntactic objects (selector terms) in \mathcal{L}_{kb} .

4.1 Proof Plan

We shall show the decidability of $\text{Th}(\text{TA}_{\text{kb}})$ by quantifier elimination. The procedure relies on the following two ideas: *solved form* and *depth reduction*.

(1) *Solved Form.* A quantifier-free formula $\varphi(x, \vec{y})$ is *solved* in x if it is in the form

$$\bigwedge_{i \leq m} u_i <^{\text{kb}} x \wedge \bigwedge_{j \leq n} x <^{\text{kb}} v_j \wedge \varphi'(\vec{y}) , \quad (8)$$

where x does not appear in u_i, v_j and φ' . It is not hard to argue that $(\exists x) \varphi(x, \vec{y})$ where $\varphi(x, \vec{y})$ is solved in x simplifies to

$$\bigwedge_{i \leq m, j \leq n} u_i <_2^{\text{kb}} v_j \wedge \varphi'(\vec{y}) , \quad (9)$$

where $<_n^{\text{kb}}$, called *gap order*, is an extension of $<^{\text{kb}}$ such that $x <_n^{\text{kb}} y$ states there is an increasing chain from x to y with at least $n - 1$ elements in between [Enderton 2001, page 196]. It is clear that the elimination of $\exists x$, that is, the transformation from (8) to (9), becomes straightforward once the matrix $\varphi(x, \vec{y})$ is solved in x , or equivalently, $\text{depth}^{\varphi}(x) = 0$. This leads us to the notion of *depth reduction*.

(2) *Depth Reduction.* Let us first consider the simple case where x is α -typed for a proper constructor α and all occurrences of x have depth greater than 0. By introducing new variables $x_1, \dots, x_{\text{ar}(\alpha)}$ (called the *descendants* of x) to represent x , we can rewrite $\exists x \varphi(x, \vec{y})$ to

$$\exists x_1, \dots, \exists x_{\text{ar}(\alpha)} \varphi'(x_1, \dots, x_{\text{ar}(\alpha)}, \vec{y}) , \quad (10)$$

where $\varphi'(x_1, \dots, x_{\text{ar}(\alpha)}, \vec{y})$ is obtained from $\varphi(x, \vec{y})$ by substituting x_i for $s_i^{\alpha} x$ ($0 < i \leq \text{ar}(\alpha)$). It is clear that $\text{depth}^{\varphi'}(x_i) < \text{depth}^{\varphi}(x)$. If all occurrences of x have the same depth, then by repeating the process we can generate a formula solved in \vec{x} where \vec{x} are descendants of x . A difficulty arises when not all occurrences of x have equal depth. So eventually we meet the situation

where some occurrences of x have depth 0 and some do not. Here we have to represent all occurrences of x of depth 0 in terms of $s_1^\alpha(x), \dots, s_{\text{ar}(\alpha)}^\alpha(x)$. This amounts to reducing literals of the form $x \prec_n^{\text{kb}} t$ and literals of the form $t \prec_n^{\text{kb}} x$ to quantifier-free formulas using $s_1^\alpha(x), \dots, s_{\text{ar}(\alpha)}^\alpha(x)$. After that we can introduce new variables and do quantifier manipulation just as in the simple case to bring $\exists x \varphi(x, \vec{y})$ into the form of (10). Therefore by the depth reduction of x , we actually mean reducing the depths of the descendants of x , and this essentially depends on the reduction of $x \prec_n^{\text{kb}} t$ and $t \prec_n^{\text{kb}} x$. To carry out the reduction we need to extend the language as follows.

- (a) We decompose \prec^{kb} into three disjoint suborders \prec^w , \prec^p and \prec^l , each of which is also extended to gap orders.
- (b) We introduce Presburger arithmetic explicitly in order to define *counting constraints* to count how many distinct terms there are at certain weight, and define *boundary functions* to *delineate* gap orders.
- (c) The reduction of literals like $x \prec_n^{\text{kb}} t$ or $t \prec_n^{\text{kb}} x$ eventually comes down to resolving relations between two terms of the same weight and of the same type. So we need to extend all aforementioned notions to tuples of terms of the same total weight.

In the rest of this section we define these extensions.

4.2 Decomposition of Knuth-Bendix Order

Definition 4.3 Decomposition of KBO. A Knuth-Bendix order \prec^{kb} can be decomposed into three *disjoint* orders, a *weight order* \prec^w , a *precedence order* \prec^p , and a *lexicographic order* \prec^l , as follows:

$$\begin{aligned} u \prec^w v &\leftrightarrow w(u) < w(v) , \\ u \prec^p v &\leftrightarrow w(u) = w(v) \wedge \text{type}(u) \prec^\Sigma \text{type}(v) , \\ u \prec^l v &\leftrightarrow w(u) = w(v) \wedge \text{type}(u) = \text{type}(v) \wedge u \prec^{\text{kb}} v . \end{aligned}$$

It is clear that $u \prec^{\text{kb}} v$ is equivalent to $u \prec^w v \vee u \prec^p v \vee u \prec^l v$. We write $u \prec^{\text{pl}} v$ for $u \prec^p v \vee u \prec^l v$ and $u \preceq^\# v$ for $u \prec^\# v \vee u = v$ ($\# \in \{\text{kb}, w, p, l, \text{pl}\}$). We say that w has the highest priority, followed by p and l , and define the maximum and minimum of a set of suborders accordingly. For example, $\max\{w, p, l\} = w$ and $\min\{w, p, l\} = l$.

4.3 Gap Orders

To express formulas of the form $\exists x(u \prec^\# x \prec^\# v)$ ($\# \in \{\text{kb}, w, p, l, \text{pl}\}$), in a quantifier-free language we need to extend all aforementioned orders to *gap orders*.

Definition 4.4 Gap Orders. Define \prec_n^{kb} ($n \geq 0$) such that \prec_0^{kb} is $=$ and for $n > 0$

$$u \prec_n^{\text{kb}} v \leftrightarrow (\exists u_1, \dots, \exists u_n) [u \prec^{\text{kb}} u_1 \prec^{\text{kb}} \dots \prec^{\text{kb}} u_n \preceq^{\text{kb}} v] .$$

Let $\# \in \{w, p, l, \text{pl}\}$. Define two orders $\prec_n^\#$ and $\preceq_n^\#$ ($n \geq 0$) such that $\prec_0^\#$ is $\preceq^\#$ and $\preceq_0^\#$ is $=$, and for $n > 0$

$$\begin{aligned} u \prec_n^\# v &\leftrightarrow u \prec_n^{\text{kb}} v \wedge u \prec^\# v , \\ u \preceq_n^\# v &\leftrightarrow u \prec_n^{\text{kb}} v \wedge \neg (u \prec_{n+1}^\# v) . \end{aligned}$$

Note that $<_1^\sharp$ is just $<^\sharp$, and for $n \geq 0$, we have

$$u <_n^\sharp v \leftrightarrow u <_{n+1}^\sharp v \vee u \leq_n^\sharp v .$$

A gap order $u <_n^\sharp v$ ($n > 0$) states that “ u is less than v w.r.t. $<^\sharp$, and there are at least $n - 1$ elements in between.” Similarly, $u \leq_n^\sharp v$ ($n > 0$) states that “ u is less than v w.r.t. $<^\sharp$, and there are exactly $n - 1$ elements in between”. We call $<_n^\sharp$ and \leq_n^\sharp *stretchable* gap orders and *rigid* gap orders, respectively. Accordingly, $u <_n^\sharp v$ are *stretchable* gap order literals and $u \leq_n^\sharp v$ are *rigid* gap order literals.

EXAMPLE 4.5 ELIMINATION IN LINEAR ORDER. *The formula $\exists x(u <^l x <^l v)$ reduces to $u <_2^l v$ if u, v do not contain x .*

4.4 Boundary Functions

Consider the formula $u \leq_1^w v$. Intuitively it states “ $w(u) < w(v)$ and there are no terms z such that $u <^{kb} z <^{kb} v$, that is, u is the largest term of weight $w(u)$ and v is the smallest term of weight $w(v)$ ”. To express this we introduce *boundary functions*.

Definition 4.6 *Boundary Functions*. Let $n, p > 0$. The following functions are called boundary functions:

- (1) $0^w : \mathbb{N} \rightarrow \mathbb{T}$ such that $0^w(n)$ is the smallest term (w.r.t. $<^{kb}$) of weight n ,
- (2) $0^p : \mathbb{N}^2 \rightarrow \mathbb{T}$ such that $0^p(n, p)$ is the smallest term (w.r.t. $<^{kb}$) of weight n and type α_p ,
- (3) $1^w : \mathbb{N} \rightarrow \mathbb{T}$ such that $1^w(n)$ is the largest term (w.r.t. $<^{kb}$) of weight n ,
- (4) $1^p : \mathbb{N}^2 \rightarrow \mathbb{T}$ such that $1^p(n, p)$ is the largest term (w.r.t. $<^{kb}$) of weight n and type α_p ,

where, for all of the above, $f(n) = \perp$ and $f(n, p) = \perp$, if no such term exists.

We write $0_{(\dots)}^\sharp$ for $0^\sharp(\dots)$ and $1_{(\dots)}^\sharp$ for $1^\sharp(\dots)$. We call $0_{(\dots)}^\sharp$ *lower boundary functions* and $1_{(\dots)}^\sharp$ *upper boundary functions*. TA-terms having one of these functions as root symbol are called *boundary terms*, distinguishing them from ordinary TA-terms. Correspondingly, we have *lower boundary terms* and *upper boundary terms*. A literal of the form $u \star v$, where \star is either a term equality or a gap order, is *open* if both u and v are ordinary TA-terms, *closed* if both u and v are boundary terms, and *half-open* otherwise. Although by definition boundary terms are syntactic objects, with little chance of confusion, we also call the corresponding semantic terms in \mathbb{T} (ground) boundary terms.

4.5 Integer Extension of Term Algebras

To be able to express the boundary terms in the formal language, we extend term algebras with Presburger arithmetic.

Definition 4.7. The structure of term algebras with integers is

$$\text{TA}_{\mathbb{Z}} = \langle \text{TA}; \text{PA}; (\cdot)^w \rangle ,$$

where PA is Presburger arithmetic and $(\cdot)^w$ denotes the weight function. We use $\mathcal{L}_{\mathbb{T}}^{\mathbb{Z}}$ to denote the language of $\text{TA}_{\mathbb{Z}}$.

We use subscripts \mathbb{T} , \mathbb{Z} (or prefixes TA-, PA-) to denote notions related to term sort and integer sort, respectively. For example, $\Phi_{\mathbb{T}}$ denotes a formula in $\mathcal{L}_{\mathbb{T}}$, the language of pure term algebras, $\Phi_{\mathbb{Z}}$ denotes a formula in $\mathcal{L}_{\mathbb{Z}}$, the language of Presburger arithmetic, $\mathcal{V}_{\mathbb{T}}$ denotes the collection of variables in $\mathcal{L}_{\mathbb{T}}$ and $\mathcal{V}_{\mathbb{Z}}$ denotes the collection of variables in $\mathcal{L}_{\mathbb{Z}}$. Although $\mathcal{L}_{\mathbb{T}}^{\mathbb{Z}}$ contains two equality predicates: *term equality* $=_{\mathbb{T}}$ in $\mathcal{L}_{\mathbb{T}}$ and *integer equality* $=_{\mathbb{Z}}$ in $\mathcal{L}_{\mathbb{Z}}$, we use $=$ in both cases unless there is a chance of confusion. Recall that we do not distinguish syntactic TA-terms in $\mathcal{L}_{\mathbb{T}}$ and semantic TA-terms in \mathbb{T} . We also use “term” for “TA-” when there is no confusion. For example, by *term variables* we mean TA-variables and by *term quantifiers* we mean quantifiers on term variables.

A TA-term can occur inside the weight function. Such occurrence is called an *integer occurrence* to be distinguished from the normal *term occurrence*. From now on, we freely use integer terms t^w to form Presburger formulas. For example, $\text{car}(x)$ is a TA-term and $(\text{car}(x))^w$ is a PA-term. The first occurrence of $\text{car}(x)$ is a term occurrence and the second one is an integer occurrence.

4.6 Extension of Knuth-Bendix Order

Definition 4.8 Extension of Knuth-Bendix Order. The structure of term algebras with KBO, extended with gap orders, boundary functions and Presburger arithmetic, is

$$\text{TA}_{\text{kb}^+}^{\mathbb{Z}} = \langle \text{TA}_{\text{kb}}; \text{TA}_{\mathbb{Z}}; <_{n'}^{\#} \leq_n^{\#}, \# \in \{w, p, l, pl\}, n \geq 0; 0_{(\dots)}^*, 1_{(\dots)}^*, * \in \{w, p\} \rangle .$$

We denote by $\mathcal{L}_{\text{kb}^+}$ the language extending \mathcal{L}_{kb} with gap orders and boundary terms. The complete language is denoted by $\mathcal{L}_{\text{kb}^+}^{\mathbb{Z}}$.

EXAMPLE 4.9 EXTENSION OF KNUTH-BENDIX ORDER. *The formula*

$$(\exists x : \mathbb{T}) \left[0_{(x^w)}^w <^{pl} x \wedge x <^{pl} 1_{(x^w)}^w \right]$$

states that there exists a term $x \in \mathbb{T}$ such that there are at least three elements with the same weight as x (including x itself). Note that the first and the third occurrences of x are integral while the second one is an ordinary term.

The truth value of the formula in Example 4.9 relies on the number of distinct TA-terms of a certain weight. This is the essential use of Presburger arithmetic.

Definition 4.10 Counting Constraint. A *counting constraint* is a predicate $\text{CNT}_n^\alpha(z)$ that states there are at least $n+1$ different α -terms of weight z . $\text{CNT}_n(z)$ is similarly defined with α -terms replaced by TA-terms. We write Tree^α (resp. Tree) for CNT_0^α (resp. CNT_0).

Counting constraints play a central role in our elimination procedure; they help reduce term quantifiers to integer quantifiers. It is easily seen that the formula from Example 4.9 is reduced to $(\exists z : \mathbb{Z}) \text{CNT}_2(z)$. It was proved in [Korovin and Voronkov 2000; Zhang et al. 2004a] that counting constraints can be expressed in PA.

EXAMPLE 4.11. *Consider $\text{List}_{\mathbb{Z}} = (\text{List}; \text{PA}; (\cdot)^w)$ where List is as in Example 3.2, and $(\cdot)^w$ is a constant weight function equal to 1. We showed in [Zhang et al. 2004b] that $\text{CNT}_n^{\text{cons}}(x)$ is $x \geq 2m - 1 \wedge 2 \nmid m$ where m is the least number such that the m -th Catalan number $C_m = \frac{1}{m} \binom{2m-2}{m-1}$ is greater than n . This is not surprising as C_m gives the number of binary trees with m leaves (that tree has $2m - 1$ nodes).*

4.7 Tuples of Terms

The extensions for tuples of terms are defined as follows:

Definition 4.12 KBO on Tuples. Let $\vec{u} = \langle u_1, \dots, u_k \rangle, \vec{v} = \langle v_1, \dots, v_k \rangle$ such that $\sum_{i=1}^k w(u_i) = \sum_{i=1}^k w(v_i)$. The lexicographic extension $<^{k;kb}$ ($k \geq 1$) of $<^{kb}$ on k -tuples of the same weight is defined such that $\vec{u} <^{k;kb} \vec{v}$ if and only if (7) holds.

Definition 4.13 Suborders on Tuples. Let $\vec{u} = \langle u_1, \dots, u_k \rangle, \vec{v} = \langle v_1, \dots, v_k \rangle \in \mathbb{T}^k$, $\sharp \in \{w, p, l, pl\}$. We define those composite orders on tuples as follows.

$$\vec{u} <^{k;\sharp} \vec{v} \leftrightarrow u_1 <^\sharp v_1 \vee (u_1 = v_1 \wedge \langle u_2, \dots, u_k \rangle <^{k-1;kb} \langle v_2, \dots, v_k \rangle) .$$

We say that $<^{k;\sharp}$ is a tuple order of length k . We identify tuple orders of length 1 with term orders. We say that $\vec{u} <^{k;\sharp} \vec{v}$ is *proper* if $u_1 <^\sharp v_1$ and we have

$$\vec{u} <^{k;kb} \vec{v} \leftrightarrow \vec{u} <^{k;w} \vec{v} \vee \vec{u} <^{k;p} \vec{v} \vee \vec{u} <^{k;l} \vec{v} .$$

We write $\vec{u} <^{k;pl} \vec{v}$ for $\vec{u} <^{k;p} \vec{v} \vee \vec{u} <^{k;l} \vec{v}$ and $\vec{u} \leq^{k;\sharp} \vec{v}$ for $\vec{u} = \vec{v} \vee \vec{u} <^{k;\sharp} \vec{v}$ ($\sharp \in \{kb, w, p, l, pl\}$).

Definition 4.14 Tuple Gap Orders. We define $<_n^{k;kb}$ ($k > 0; n \geq 0$) such that $<_0^{k;kb}$ is $=$ and for $n > 0$

$$\vec{u} <_n^{k;kb} \vec{v} \leftrightarrow (\exists \vec{u}_1, \dots, \exists \vec{u}_n : \mathbb{T}^k) [\vec{u} <^{k;kb} \vec{u}_1 <^{k;kb} \dots <^{k;kb} \vec{u}_n \leq^{k;kb} \vec{v}] .$$

Let $\sharp \in \{w, p, l, pl\}$. Define two orders $<_n^{k;\sharp}$ and $\leq_n^{k;\sharp}$ ($k \geq 1; n \geq 0$) such that $<_0^{k;\sharp}$ is $\leq^{k;\sharp}$, $\leq_0^{k;\sharp}$ is $=$, and for $n > 0$

$$\begin{aligned} \vec{u} <_n^{k;\sharp} \vec{v} &\leftrightarrow \vec{u} <_n^{k;kb} \vec{v} \wedge \vec{u} <^{k;\sharp} \vec{v} , \\ \vec{u} \leq_n^{k;\sharp} \vec{v} &\leftrightarrow \vec{u} <_n^{k;\sharp} \vec{v} \wedge \neg (\vec{u} <_{n+1}^{k;\sharp} \vec{v}) . \end{aligned}$$

Similar as term gap orders, $<_1^{k;\sharp}$ is just $<^{k;\sharp}$, and we have

$$\vec{u} <_n^{k;\sharp} \vec{v} \leftrightarrow \vec{u} <_{n+1}^{k;\sharp} \vec{v} \vee \vec{u} \leq_n^{k;\sharp} \vec{v} .$$

As before we call $<_n^{k;\sharp}$ and $\leq_n^{k;\sharp}$ *stretchable* tuple gap orders and *rigid* tuple gap orders, respectively. Accordingly, $u <_n^{k;\sharp} v$ are *stretchable* tuple gap order literals and $u \leq_n^{k;\sharp} v$ are *rigid* tuple gap order literals.

Definition 4.15 Tuple Boundary Functions. Let $k, n, m, p > 0$. Define partial functions:

- (1) $\bar{0}^{k;kb} : \mathbb{N} \rightarrow \mathbb{T}^k$ ($k \geq 1$) such that $\bar{0}^{k;kb}(n)$ is the smallest k -tuple (w.r.t. $<^{k;kb}$) of weight n .
- (2) $\bar{0}^{k;w} : \mathbb{N}^2 \rightarrow \mathbb{T}^k$ ($k \geq 1$) such that $\bar{0}^{k;w}(n, m)$ is the smallest k -tuple (w.r.t. $<^{k;kb}$) of weight n and its first component has weight m .
- (3) $\bar{0}^{k;p} : \mathbb{N}^3 \rightarrow \mathbb{T}^k$ ($k \geq 1$) such that $\bar{0}^{k;p}(n, m, p)$ is the smallest k -tuple (w.r.t. $<^{k;kb}$) of weight n and its first component has weight m and type α_p .
- (4) $\bar{1}^{k;kb} : \mathbb{N} \rightarrow \mathbb{T}^k$ ($k \geq 1$) such that $\bar{1}^{k;kb}(n)$ is the largest k -tuple (w.r.t. $<^{k;kb}$) of weight n .

- (5) $\bar{\Gamma}^{k,w} : \mathbb{N}^2 \rightarrow \mathbb{T}^k$ ($k \geq 1$) such that $\bar{O}^{k,w}(n, m)$ is the largest k -tuple (w.r.t. $<^{k, kb}$) of weight n and its first component has weight m .
- (6) $\bar{\Gamma}^{k,p} : \mathbb{N}^3 \rightarrow \mathbb{T}^k$ ($k \geq 1$) such that $\bar{O}^{k,p}(n, m, p)$ is the largest k -tuple (w.r.t. $<^{k, kb}$) of weight n and its first component has weight m and type α_p .

As before these functions are made total by assigning $\langle \perp, \dots, \perp \rangle$ to undefined values. We write $\bar{O}_{(\dots)}^{k,\#}$ for $\bar{O}^{k,\#}(\dots)$ and $\bar{\Gamma}_{(\dots)}^{k,\#}$ for $\bar{\Gamma}^{k,\#}(\dots)$. We call $\bar{O}_{(\dots)}^{k,\#}$ *lower tuple boundary functions* and $\bar{\Gamma}_{(\dots)}^{k,\#}$ *upper tuple boundary functions*. Terms having one of these functions as root symbol are called *boundary tuples*. Correspondingly, we have *lower boundary tuples* and *upper boundary tuples*. As before we call a *tuple literal* $\vec{u} \star \vec{v}$ *open* if both \vec{u} and \vec{v} are ordinary tuples, *closed* if both \vec{u} and \vec{v} are boundary tuples, and *half-open* otherwise.

To avoid unnecessary complications, we choose to treat tuples (including boundary tuples) as “syntactic sugar”; they are only used in the intermediate steps of the reduction. Lemma 4.27 shows that literals containing tuples can be reduced to formulas in $\mathcal{L}_{kb^+}^Z$.

4.8 Delineated Gap Order Completion

Revisiting the transformation from (8) to (9), we see that the number of gap orders in (9) is quadratic in the number of gap orders in (8). This complicates the termination proof for the elimination procedure. Nevertheless, we can avoid this difficulty by postulating the relative positions of parameters. This leads to the notion of *order completion*.

Definition 4.16 Gap Order Completion. A *gap order completion* (GOC) of a set of terms $\{t_1, \dots, t_n\}$ is the chain

$$t_{f(1)} \preceq_1 t_{f(2)} \preceq_2 \cdots \preceq_{n-1} t_{f(n)} ,$$

where f is a permutation function on $\{1, \dots, n\}$ and \preceq_i ($0 < i \leq n$) stands for $=$, $\preceq_m^\#$ or $<_m^\#$ ($\# \in \{w, p, l, pl\}$, $m > 0$). By a GOC of a formula φ , we mean a GOC of $\mathcal{T}(\varphi)$, the set of terms in φ .

EXAMPLE 4.17 GAP ORDER COMPLETION. A *possible GOC* of

$$\varphi(x, y, z) : x <_9^w y \wedge x <^{pl} z \wedge z <^w y$$

is $x <_5^{pl} z <_4^w y$.

However, gap order completions are not sufficient. Clearly $(\exists x : \mathbb{T})[u <^w x <^p v]$ implies $u <_2^w v$. But for the converse to hold, $v \neq 0_{(vw)}^w$ is required. As another example, $(\exists x : \mathbb{T})[u <^p x <^p v]$ implies $u <_2^p v$, but not vice versa, because there may exist only two p -intervals at weight $w(u)$ while $(\exists x : \mathbb{T})[u <^p x <^p v]$ requires that u , x and v all be in different p -intervals. In order to preserve equivalence, intuitively, we need to “delineate” a GOC to make sure ordinary terms in different *intervals* (a notion to be defined precisely below) are not related in any gap orders.

EXAMPLE 4.18 ORDER ARRANGEMENT. *Figure 1 shows an order arrangement of the linear order*

$$x_1 <_{n_1}^w x_2 <_{n_2}^p x_3 <_{n_3}^l x_4 .$$

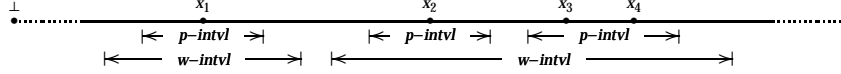


Fig. 1. An order arrangement of $x_1 \prec_{n_1}^w x_2 \prec_{n_2}^p x_3 \prec_{n_3}^l x_4$.

The weight of x_1 is strictly lower than that of x_2 , x_3 , and x_4 . The weight of x_2 , x_3 , and x_4 is the same, but the precedence of x_2 is lower than that of x_3 and x_4 . Finally, x_3 is smaller than x_4 in the lexicographic order. We call a maximal list of elements with the same weight a w -interval, and similarly a maximal list of elements with the same weight and precedence order a p -interval. Thus, the second w -interval above has two inner p -intervals.

We want to avoid relating ordinary elements at different levels in different intervals. Therefore we augment the gap order completion with boundary terms, called a delineated gap order completion.

Definition 4.19 Delineated Gap Order Completion. A delineated gap order completion (DGOC) of a set of terms S is a GOC of S in which if there occurs the following pattern $v_1 \triangleleft_{n_1}^{\sharp} u \triangleleft_{n_2}^{\natural} v_2$, where $n_1, n_2 > 0$, \triangleleft stands for either $<$ or \leq , $\sharp, \natural \in \{w, p, l, pl\}$, and u is an ordinary term in \mathcal{L}_{kb} , then either $\sharp \equiv \natural \equiv pl$ or $\sharp \equiv \natural \equiv l$. I.e., ordinary terms do not delineate two intervals unless they are asserted equal to boundary terms. By a DGOC of a formula φ , we mean a DGOC of $\mathcal{T}(\varphi)$, the set of terms in φ .

We assume in a DGOC that both the smallest element and the largest element are boundary terms.

EXAMPLE 4.20 DELINEATED GAP ORDER COMPLETION. Revisit Example 4.17. A possible DGOC of $\varphi(x, y, z)$ is

$$\varphi'(x, y, z) : \underbrace{0_{(x^w)}^w \prec_1^{pl} x \prec_5^{pl} z \prec_2^{pl} 1_{(x^w)}^w}_{w\text{-interval}} \prec_1^w \underbrace{0_{(y^w)}^w \prec_1^{pl} y \prec_1^{pl} 1_{(y^w)}^w}_{w\text{-interval}} .$$

We have

$$(\exists z : \mathbb{T}) \varphi'(x, y, z) \leftrightarrow 0_{(x^w)}^w \prec_1^{pl} x \prec_7^{pl} 1_{(x^w)}^w \prec_1^w 0_{(y^w)}^w \prec_1^{pl} y \prec_1^{pl} 1_{(y^w)}^w .$$

4.9 Lemmas

In this section we state a sequence of lemmas that justify the elimination procedure given in the next section. These lemmas share the following common features.

- (1) They state the soundness of symbolic transformations for formulas in *primitive form*, a special prenex form where the prefix only consists of existential quantifiers and the matrix is a conjunction of literals;
- (2) A formula φ is transformed into a finite disjunction $\bigvee_i \varphi_i$ where for any i , φ_i is in primitive form and the matrix of φ_i , a conjunction of literals, contains no more open gap order literals than that of φ does. In the following whenever we say that a reduction does not introduce more open gap order literals, we always mean that it does not introduce more in each resulting conjunction of literals.

To save space, however, we omit these conditions in the description of each lemma.

The following lemma allows us to assume that a formula in $\mathcal{L}_{\text{kb}^+}$ only contains *positive* order literals, *positive* tester literals and term equality literals.

LEMMA 4.21 ELIMINATION OF NEGATIVE LITERALS. *Assume formulas are type complete.*

- (1) *Any negative tester literal is equivalent to a disjunction of positive tester literals.*
- (2) *Any negative term equality literal (i. e., disequality between TA-terms) is equivalent to a disjunction of positive order literals.*
- (3) *Any negative order literal is equivalent to a disjunction of integer literals, tester literals and positive order literals.*

The following lemma allows us to assume that the matrix in a primitive formula is a delineated gap order completion.

LEMMA 4.22 DELINEATED GAP ORDER COMPLETION. *Any conjunction of positive order literals in $\mathcal{L}_{\text{kb}^+}$ can be effectively transformed to an equivalent finite disjunction of delineated gap order completions.*

In principle, boundary terms can appear in the weight function or in selectors, selector terms can occur in the weight function, and the weight function can be used to construct boundary terms. Repeating this process we can build more and more complex terms. The following lemma eliminates this superficial complication.

LEMMA 4.23 NO EMBEDDING OF BOUNDARY TERMS. *Any formula in $\mathcal{L}_{\text{kb}^+}^Z$ can be effectively reduced to an equivalent formula in which no boundary terms appear inside selectors or the weight function.*

From now on, we assume that boundary terms are not properly embedded in other terms. The following lemma states that term gap order literals between a variable x of non-constant type α_p and another TA-term can be expressed using x 's immediate descendants $s_i^{\alpha_p} x$.

LEMMA 4.24 REDUCTION OF TERM GAP ORDER LITERALS. *Let*

$$\star \in \{ \prec_n^{\text{kb}}, \prec_n^{\text{w}}, \prec_{n'}^{\text{p}}, \prec_{n'}^{\text{l}}, \prec_n^{\text{pl}}, \preceq_n^{\text{kb}}, \preceq_n^{\text{w}}, \preceq_{n'}^{\text{p}}, \preceq_{n'}^{\text{l}}, \preceq_n^{\text{pl}} \} \quad (n > 0) .$$

If x is a TA-variable of type α_p with $\alpha_p = (s_1^{\alpha_p}, \dots, s_k^{\alpha_p})$ and t is an arbitrary TA-term, then $x \star t$ ($t \star x$) can be effectively reduced to an equivalent quantifier-free formula $\varphi(s_1^{\alpha_p} x, \dots, s_k^{\alpha_p} x)$ (in $\mathcal{L}_{\text{kb}^+}^Z$) where x only occurs in $s_i^{\alpha_p} x$ ($0 < i \leq k$).

By this lemma we can always assume that all *term occurrences* of a TA-variable have the same depth, and hence we are able to reduce them all together to depth 0. As we mentioned before, this is the main battlefield of quantifier elimination. To streamline the proof, we introduce the following three lemmas. The first one takes care of closed term literals generated during the reduction.

LEMMA 4.25 REDUCTION OF CLOSED TERM LITERALS. *Let*

$$\star \in \{ =, \prec_n^{\text{kb}}, \prec_n^{\text{w}}, \prec_{n'}^{\text{p}}, \prec_{n'}^{\text{l}}, \prec_n^{\text{pl}}, \preceq_n^{\text{kb}}, \preceq_n^{\text{w}}, \preceq_{n'}^{\text{p}}, \preceq_{n'}^{\text{l}}, \preceq_n^{\text{pl}} \} \quad (n > 0) .$$

If $u \star v$ is closed, i. e., both u and v are boundary terms, then it can be effectively reduced to an equivalent Presburger formula.

The second one solves the case where no term occurrence of x appears in the other side of the gap order literal.

LEMMA 4.26 REDUCTION OF NON-CLOSED TERM GAP ORDER LITERALS. *Let*

$$\star \in \{ \prec_n^{kb}, \prec_n^w, \prec_n^p, \prec_n^l, \prec_n^{pl}, \preceq_n^{kb}, \preceq_n^w, \preceq_n^p, \preceq_n^l, \preceq_n^{pl} \} \quad (n > 0) .$$

If x is a TA-variable of type α_p with $\alpha_p = (S_1^{\alpha_p}, \dots, S_k^{\alpha_p})$ and t is either a boundary term or an ordinary TA-term not containing x , then $x \star t$ ($t \star x$) can be effectively reduced to an equivalent quantifier-free formula $\varphi(S_1^{\alpha_p}x, \dots, S_k^{\alpha_p}x)$ where x only occurs in $S_i^{\alpha_p}x$ ($0 < i \leq k$).

This lemma in fact deals with the most sophisticated part of the reduction that eventually comes down to the success of reducing relations between tuples of the same weight, as is stated by the third lemma.

LEMMA 4.27 REDUCTION OF TUPLE LITERALS. *Let U, V be k -tuples of the same weight, and*

$$\star \in \{ =, \prec_n^{kbb}, \prec_n^{kw}, \prec_n^{kp}, \prec_n^{kl}, \prec_n^{kpl}, \preceq_n^{kbb}, \preceq_n^{kw}, \preceq_n^{kp}, \preceq_n^{kl}, \preceq_n^{kpl} \} \quad (k > 0, n > 0) .$$

- (1) *If $U = \langle u_1, \dots, u_k \rangle$ is an ordinary tuple, then $U \star V$ ($V \star U$) can be effectively reduced to an equivalent quantifier-free formula $\varphi(u_1, \dots, u_k)$ (in $\mathcal{L}_{kb^+}^Z$) in which u_i ($0 < i \leq k$) does not occur inside selectors.*
- (2) *If $U \star V$ ($V \star U$) is a closed tuple, i. e., both U and V are boundary tuples, then it can be effectively reduced to an equivalent Presburger formula.*

We never need to reduce open or half-open term equality literals, because, as shown below, we can carry out elimination on $(\exists x)[x = t \wedge \varphi(x)]$ even if t contains x .

LEMMA 4.28 ELIMINATION OF TERM QUANTIFIERS. *Let x be a term variable, $\varphi_{kb^+}(x)$ a conjunction of literals in \mathcal{L}_{kb^+} with $\text{depth}^{\varphi_{kb^+}}(x) = 0$, and $\varphi_Z(x)$ a Presburger formula in which x occurs inside the weight function. Then*

$$(\exists x : \mathbb{T}) \left[\varphi_{kb^+}(x) \wedge \varphi_Z(x) \right]$$

can be effectively reduced to $\varphi'_{kb^+} \wedge \varphi'_Z$ in which x does not occur and φ'_{kb^+} is quantifier-free.

In fact term quantifiers are reduced to integer quantifiers that can be eliminated according to the following lemma.

LEMMA 4.29 ELIMINATION OF INTEGER QUANTIFIERS. *Let z be an integer variable, $\varphi_{kb^+}(z)$ a conjunction of literals in \mathcal{L}_{kb^+} where z occurs inside boundary terms, and $\varphi_Z(z)$ a Presburger formula. Then*

$$(\exists z : \mathbb{Z}) \left[\varphi_{kb^+}(z) \wedge \varphi_Z(z) \right]$$

can be effectively reduced to $\varphi'_{kb^+} \wedge \varphi'_Z$ where no z occurs and φ'_{kb^+} is quantifier-free.

5. QUANTIFIER ELIMINATION FOR $\text{Th}(\text{TA}_{KB^+}^Z)$

In this section we present a quantifier elimination procedure for $\text{Th}(\text{TA}_{kb^+}^Z)$ which is an extension of the elimination procedure for $\text{Th}(\text{TA}_Z)$ [Zhang et al. 2006]. We first recall the following notation conventions.

- (1) *Primitive Form*. All transformations are carried out on formulas of the form (1). Each step of the transformations manipulates (1) to produce a version of the same form (or multiple versions of the same form in case disjunctions are introduced).
- (2) *Positive Literals*. All order literals have positive occurrence. This is guaranteed by Lemma 4.21.
- (3) *Nondeterminism*. Whenever we say “guess ψ ”, we mean split on a finite disjunction $\bigvee_i \varphi_i$, which is valid in the context and contains ψ as a disjunct.
- (4) *Type Completeness*. All formulas are type-complete. But we omit listing tester literals unless they are needed for the correctness proof.

5.1 The Elimination Procedure

The elimination procedure consists of the following two algorithms:

ALGORITHM 5.1 ELIMINATION OF INTEGER QUANTIFIERS IN $\text{TA}_{\text{kb}^+}^{\mathbb{Z}}$. *We assume that formulas with quantifiers on integer variables are in the form*

$$(\exists \vec{z} : \mathbb{Z}) \left[\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z}) \wedge \varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z}) \right], \quad (11)$$

where \vec{y}, \vec{z} are integer variables, \vec{x} are term variables. Note that \vec{x} may occur inside the weight function in $\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z})$ and \vec{y}, \vec{z} may appear inside boundary terms in $\varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z})$. Repeatedly apply the following steps (1) and (2) to (11) until $\vec{z} = \emptyset$.

- (1) If none of \vec{z} appears inside any boundary terms, then $\varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z})$ is just $\varphi_{\text{kb}^+}(\vec{x}, \vec{y})$, which can be moved out of $\exists \vec{z}$. We then obtain

$$(\exists \vec{z} : \mathbb{Z}) \left[\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z}) \right] \wedge \varphi_{\text{kb}^+}(\vec{x}, \vec{y}).$$

Since $(\exists \vec{z} : \mathbb{Z})[\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z})]$ is in $\mathcal{L}_{\mathbb{Z}}$, we can proceed to remove the block of existential quantifiers using Cooper’s method [Cooper 1972; Reddy and Loveland 1978]. Similar to the elimination of integer quantifiers in $\text{TA}_{\mathbb{Z}}$ [Zhang et al. 2006], we can defer the actual elimination on $(\exists \vec{z} : \mathbb{Z})[\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z})]$ until all term quantifiers have been eliminated. The reason is the same: the elimination of term quantifiers does not require the integer constraint in (12) to be quantifier-free.

- (2) If for some $z \in \vec{z}$, z occurs inside some boundary terms, we set

$$\begin{aligned} \varphi'_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z} \setminus z) \wedge \varphi'_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z} \setminus z) &:= \\ \text{ELIMINATE-INT} \left((\exists z : \mathbb{Z}) \left[\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z}) \wedge \varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z}) \right] \right), & \\ (\exists \vec{z} : \mathbb{Z}) \left[\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z}) \wedge \varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z}) \right] &:= \\ (\exists (\vec{z} \setminus z) : \mathbb{Z}) \left[\varphi'_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z} \setminus z) \wedge \varphi'_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z} \setminus z) \right]. & \end{aligned}$$

The existence of **ELIMINATE-INT** is guaranteed by Lemma 4.29.

ALGORITHM 5.2 ELIMINATION OF TERM QUANTIFIERS IN $\text{TA}_{\text{kb}^+}^{\mathbb{Z}}$. *We assume that formulas with quantifiers on term variables are in the form*

$$(\exists \vec{x} : \mathbb{T}) \left[\varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z}) \right], \quad (12)$$

where \vec{x}, \vec{y} are term variables, \vec{z} are integer variables. Note that \vec{z} may occur inside boundary terms in $\varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z})$, and \vec{x}, \vec{y} may occur inside the weight function in $\varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z})$.

Repeatedly apply the following steps (1) and (2) to (12) until $\vec{x} = \emptyset$.

(1) **Depth Reduction.** If $(\forall x \in \vec{x}) \text{depth}^{\varphi_{\text{kb}^+}}(x) > 0$.

(a) **VARIABLE SELECTION.** Select an α -typed variable $x \in \vec{x}$ for some $\alpha = (s_1^\alpha, \dots, s_{\text{ar}(\alpha)}^\alpha)$.

This selection is always possible as $\text{depth}^{\varphi_{\text{kb}^+}}(x) > 0$. We require that in the next run of (1a) we choose one of the variables generated by this run of (1b). I.e., the variable selection is done in depth-first manner. This is crucial to guarantee that a run eventually leaves (1). Let $\vec{x}' \equiv \vec{x} \setminus x$.

(b) **DECOMPOSITION.** Rewrite (12) to:

$$\left(\exists \vec{x}', x_1, \dots, x_{\text{ar}(\alpha)}, x : \mathbb{T} \right) \left[\text{Is}_\alpha(x) \wedge \bigwedge_{0 < i \leq \text{ar}(\alpha)} s_i^\alpha(x) = x_i \right. \\ \left. \wedge \varphi_{\text{kb}^+}(\vec{x}', \vec{y}, \vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{x}', \vec{y}, \vec{z}) \right]. \quad (13)$$

(c) **SIMPLIFICATION.** Exhaustively apply the following simplification rules to φ_{kb^+} and $\varphi_{\mathbb{Z}}$ in (13):

i. replace $s_i^\alpha(x)$ by x_i ($0 < i \leq \text{ar}(\alpha)$);

ii. replace x^w by $w(\alpha) + \sum_{i=1}^{\text{ar}(\alpha)} x_i^w$;

iii. replace $x <_n^\# t$ by **DEPTH-REDUCTION** ($x <_n^\# t$); similar for $t <_n^\# x$, $x \leq_n^\# t$ and $t \leq_n^\# x$.

The existence of **DEPTH-REDUCTION** follows from Lemma 4.24. Let the resulting formula be

$$\left(\exists \vec{x}', x_1, \dots, x_{\text{ar}(\alpha)}, x : \mathbb{T} \right) \left[\text{Is}_\alpha(x) \wedge \bigwedge_{0 < i \leq \text{ar}(\alpha)} s_i^\alpha(x) = x_i \right. \\ \left. \varphi'_{\text{kb}^+}(\vec{x}', s_1^\alpha(x), \dots, s_{\text{ar}(\alpha)}^\alpha(x), \vec{y}, \vec{z}) \wedge \varphi'_{\mathbb{Z}}(\vec{x}', s_1^\alpha(x), \dots, s_{\text{ar}(\alpha)}^\alpha(x), \vec{y}, \vec{z}) \right]. \quad (14)$$

It is now clear that if x occurs in φ'_{kb^+} and $\varphi'_{\mathbb{Z}}$ it occurs inside some of $s_i^\alpha(x)$ ($i \in [1.. \text{ar}(\alpha)]$). Since

$$\left(\forall x_1, \dots, x_{\text{ar}(\alpha)} : \mathbb{T} \right) \left(\exists x : \mathbb{T} \right) \left[\text{Is}_\alpha(x) \wedge \bigwedge_{0 < i \leq \text{ar}(\alpha)} s_i^\alpha(x) = x_i \right]$$

is valid in TA, we can replace in (14), $s_1^\alpha(x), \dots, s_{\text{ar}(\alpha)}^\alpha(x)$, respectively, by $x_1, \dots, x_{\text{ar}(\alpha)}$, and hence remove $\bigwedge_{0 < i \leq \text{ar}(\alpha)} s_i^\alpha(x) = x_i$, $\text{Is}_\alpha(x)$ together with $\exists x$, obtaining

$$\left(\exists \vec{x}', x_1, \dots, x_{\text{ar}(\alpha)} : \mathbb{T} \right) \\ \left[\varphi'_{\text{kb}^+}(\vec{x}', x_1, \dots, x_{\text{ar}(\alpha)}, \vec{y}, \vec{z}) \wedge \varphi'_{\mathbb{Z}}(\vec{x}', x_1, \dots, x_{\text{ar}(\alpha)}, \vec{y}, \vec{z}) \right]. \quad (15)$$

(2) **Elimination.** If $(\exists x \in \vec{x}) \text{depth}^{\varphi_{\text{kb}^+}}(x) = 0$.

Take the x as in the guard condition, guess a DGOC for all terms related with x in gap

```

Input:  $(\exists \vec{x} : \mathbb{T}) [\varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z})]$ .
while  $\vec{x} \neq \emptyset$  do
  if  $(\forall x \in \vec{x}) \text{depth}^{\varphi_{\text{kb}^+}}(x) > 0$  then
    (1) Depth Reduction
      (1a) VARIABLE SELECTION
      (1b) DECOMPOSITION
      (1c) SIMPLIFICATION
  else  $(\exists x \in \vec{x}) \text{depth}^{\varphi_{\text{kb}^+}}(x) = 0$ 
    (2) Elimination
  end if
end while

```

Fig. 2. Quantifier Elimination in $\text{TA}_{\text{kb}^+}^{\mathbb{Z}}$.

order literals (by Lemma 4.22) and then eliminate x by Lemma 4.28. Formally we set

$$\begin{aligned}
\varphi'_{\text{kb}^+}(\vec{x} \setminus x, \vec{y}, \vec{z}) \wedge \varphi'_{\mathbb{Z}}(\vec{x} \setminus x, \vec{y}, \vec{z}) &:= \\
&\text{ELIMINATE-TERM} \left((\exists x : \mathbb{T}) [\varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z})] \right), \\
(\exists \vec{x} : \mathbb{T}) [\varphi_{\text{kb}^+}(\vec{x}, \vec{y}, \vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{x}, \vec{y}, \vec{z})] &:= \\
&(\exists (\vec{x} \setminus x) : \mathbb{T}) [\varphi'_{\text{kb}^+}(\vec{x} \setminus x, \vec{y}, \vec{z}) \wedge \varphi'_{\mathbb{Z}}(\vec{x} \setminus x, \vec{y}, \vec{z})].
\end{aligned}$$

For the ease of understanding Algorithm 5.2, we show its high-level control-flow in Figure 2. It is now easily seen that Algorithm 5.2 is a greedy algorithm in the sense that it tries to do *Elimination* (step (2)) as soon as the *elimination condition* holds, that is, all term occurrences of this variable are of depth 0. Otherwise, the algorithm tries to create the elimination condition using *Depth Reduction* (step (1)) which includes three sequential sub-procedures: *VARIABLE SELECTION* (step (1a)), *DECOMPOSITION* (step (1b)) and *SIMPLIFICATION* (step (1c)). We require that *VARIABLE SELECTION* be done in depth-first manner. As all depths are finite, this guarantees that a run eventually leaves *Depth Reduction* and enter *Elimination*.

LEMMA 5.3 SOUNDNESS. *Each transformation step in Algorithms 5.1 and 5.2 preserves equivalence.*

PROOF. Equivalence preservation has been assured by the lemmas in Section 4.9. More precisely, Lemma 4.29 justifies the soundness of Algorithm 5.1. For Algorithm 5.2, Axioms (1)-(6) in Section 3 justify step (1b); Lemma 4.24, with the help of Lemmas 4.25, 4.26 and 4.27, justifies step (1c); Lemma 4.28 justifies step (2). \square

LEMMA 5.4 TERMINATION. *Both Algorithm 5.1 and Algorithm 5.2 terminate.*

THEOREM 5.5 DECIDABILITY. $\text{Th}(\text{TA}_{\text{kb}^+}^{\mathbb{Z}})$ is decidable, and hence so is $\text{Th}(\text{TA}_{\text{kb}})$.

PROOF. By Lemmas 5.3 and 5.4. \square

5.2 An Example

EXAMPLE 5.6 QUANTIFIER ELIMINATION IN $\text{List}_{\text{kb}^+}^{\mathbb{Z}}$. *Let us go through an example with emphasis on the depth reduction. We only show one simple trace of the reduction. Consider*

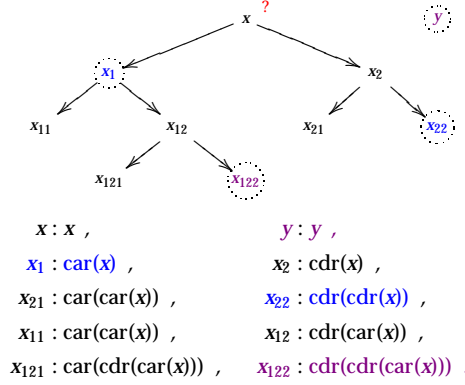


Fig. 3. Quantifier Elimination on $(\exists x) [\text{car}(x) <_2^1 \text{cdr}(\text{cdr}(x)) \wedge \text{cdr}(\text{cdr}(\text{car}(x))) <_3^1 y]$.

in $\text{List}_{\text{kb}^+}^{\mathbb{Z}}$ the following formula

$$(\exists x) \left[\text{car}(x) <_2^1 \text{cdr}(\text{cdr}(x)) \wedge \text{cdr}(\text{cdr}(\text{car}(x))) <_3^1 y \right] , \quad (16)$$

where $\text{depth}^{(16)}(x) = 3$. Figure 3 shows the tree representation of (16) which is sibling complete, that is, sibling nodes coexist. Formula (16) states that there exists a tree x whose descendants x_1, x_{22}, x_{122} and a parameter y satisfy certain order relations.

At the first run of step (1), we introduce fresh variables x_1 and x_2 to replace $\text{car}(x)$ and $\text{cdr}(x)$, respectively. By a standard quantifier manipulation we obtain

$$(\exists x_1 \exists x_2) \left[x_1 <_2^1 \text{cdr}(x_2) \wedge \text{cdr}(\text{cdr}(x_1)) <_3^1 y \right] , \quad (17)$$

where $\text{depth}^{(17)}(x_1) = 2$ and $\text{depth}^{(17)}(x_2) = 1$, both less than $\text{depth}^{(16)}(x)$. In the second run of step (1), we pick x_1 and replace $x_1 <_2^1 \text{cdr}(x_2)$ by $\text{car}(x_1) = \text{car}(\text{cdr}(x_2)) \wedge \text{cdr}(x_1) <_2^1 \text{cdr}(\text{cdr}(x_2))$ (which is one of several choices). We obtain

$$(\exists x_2 \exists x_{11} \exists x_{12}) \left[x_{11} = \text{car}(\text{cdr}(x_2)) \wedge x_{12} <_2^1 \text{cdr}(\text{cdr}(x_2)) \wedge \text{cdr}(x_{12}) <_3^1 y \right] . \quad (18)$$

At this point we have $\text{depth}^{(18)}(x_{11}) = 0$ and the run enters step (2). In this case we can immediately remove $\exists x_{11}$, obtaining

$$(\exists x_2 \exists x_{12}) \left[x_{12} <_2^1 \text{cdr}(\text{cdr}(x_2)) \wedge \text{cdr}(x_{12}) <_3^1 y \right] , \quad (19)$$

where $\text{depth}^{(19)}(x_{12}) = 1$ and $\text{depth}^{(19)}(x_2) = 2$. At the third run of step (1), we select x_{12} . The run could give us

$$(\exists x_2 \exists x_{121} \exists x_{122}) \left[x_{121} = \text{car}(\text{cdr}(\text{cdr}(x_2))) \wedge x_{122} <_2^1 \text{cdr}(\text{cdr}(x_2)) \wedge x_{122} <_3^1 y \right] ,$$

which as before by step (2) simplifies to

$$(\exists x_2 \exists x_{122}) \left[x_{122} <_2^1 \text{cdr}(\text{cdr}(x_2)) \wedge x_{122} <_3^1 y \right] . \quad (20)$$

Still we have $\text{depth}^{(20)}(x_{122}) = 0$ which justifies another run of step (2). Let us take a gap order completion $x_{122} <_2^1 \text{cdr}(\text{cdr}(x_2)) <_1^1 y$ (which again is just one of many choices) and

rewrite (20) to

$$(\exists x_2 \exists x_{122}) \left[x_{122} <_2^1 \text{cdr}(\text{cdr}(x_2)) <_1^1 y \right]. \quad (21)$$

With the help of boundary functions, (21) reduces to

$$(\exists x_2) \left[0_{((\text{cdr}(\text{cdr}(x_2)))^w)}^w <_2^1 \text{cdr}(\text{cdr}(x_2)) <_1^1 y \right]. \quad (22)$$

The fourth and the fifth runs of step (1) (with the same trick of quantifier manipulation) give us

$$(\exists x_{222}) \left[0_{(x_{222}^w)}^w <_2^1 x_{222} <_1^1 y \right]. \quad (23)$$

After that the run returns to step (2) as $\text{depth}^{(23)}(x_{222}) = 0$. Here we have to reduce term quantifiers to integer quantifiers because x_{222} also appears in boundary terms. By Lemma 4.28, (23) is equivalent to

$$(\exists z) \left[0_{(z)}^w <_3^1 y \wedge \text{Tree}^{\text{cons}}(z) \right], \quad (24)$$

which simplifies to $0_{(y^w)}^w <_3^1 y \wedge \text{Tree}^{\text{cons}}(y^w)$, and in turn to

$$0_{(y^w)}^w <_3^1 y, \quad (25)$$

as $0_{(y^w)}^w <_3^1 y$ implies $\text{Tree}^{\text{cons}}(y^w)$. It is not hard to verify that (25) implies (16) as desired. (We do not have equivalence because this is just one trace of the reduction.) In fact, the above elimination procedure tells us how to obtain a solution to (16) provided (25) holds. Let us show the construction in bottom-up fashion. By $x_\#$, we refer to labels in Figure 3. First, we set x_{122} to $0_{(y^w)}^w$ to take care of $x_{122} <_3^1 y$. Second, we find x_{11} , x_{121} and x_{22} to satisfy $x_1 <_2^1 x_{22}$. This can always be done as there are no constraints on x_{11} , x_{121} and x_{22} . Third, we set x_{21} to an arbitrary value. In this way we obtain a value for x that is the solution to (16).

We note that the depth reduction of a variable is at the expense of increasing the depth of a term on the other side of a gap order predicate. This happens when φ_{kb^+} contains $x \star t$ (or $t \star x$) and $\text{depth}^{\varphi_{\text{kb}^+}}(x) > 0$. For example, from (17) to (18), the depth of x_2 increases by 1. Moreover, the depth reduction in general introduces more existential quantifiers and more equalities in at least one resulting formula (e.g., this also happens in the reduction from (17) to (18)). In every step of the transformation, however, the number of open gap order literals in each resulting primitive formula is no more than that in the original formula. Moreover, the final elimination procedure removes at least one open gap order literal if the eliminated variable occurs in such literals (e.g., from (21) to (22) and from (23) to (24)). When all open gap order literals are gone, the depths of terms will be strictly decreasing. This forces the run of Algorithm 5.2 to eventually leave step (1) and from then on to stay in step (2) until all existential quantifiers are removed.

6. PRESENCE OF A 0-WEIGHT UNARY FUNCTION

Up to now we have assumed that the language does not contain a unary function f of weight 0. However, the presence of such a function considerably simplifies the reduction procedure. First we note that there exists no maximum ground TA-term

$1_{(m)}^w$ of weight m for any $m > 0$ (when we do not consider the artificial fix using the sink value \perp), because otherwise we will have a contradiction as $1_{(m)}^w <^{kb} f(1_{(m)}^w)$ and $w(1_{(m)}^w) = w(f(1_{(m)}^w))$. As a consequence, $<_n^w$ is *dense* in the sense that if $u <_n^w v$ ($n > 0$), there are infinitely many ground TA-terms in between u and v with respect to $<^{kb}$. Similarly, there exists no maximum ground TA-term $1_{(m,p)}^p$ of weight m and type α_p except when α_p is a constant and $m = w(\alpha_p)$, and hence $<_n^p$ is *dense* except when the left operand is a constant. Moreover, $<_n^l$ is *dense* except when the order is resolved between the corresponding rightmost immediate subterms of its operands. To make the above statements precise, we present reductions for open gap order literals involving $<_n^w$, $<_n^p$, $<_n^l$, \leq_n^w , \leq_n^p , and \leq_n^l ($n > 0$).

6.1 Reduction of $u <_n^w v$ and $u \leq_n^w v$

Clearly, $u <_n^w v$ implies $u <^w v$ and assuming $u <^w v$, we have an infinite chain

$$u <^{pl} f(u) <^{pl} f^2(u) <^{pl} \dots <^{pl} f^n(u) <^{pl} \dots <^w v .$$

Therefore $u <_n^w v$ simplifies to $u <^w v$, which further simplifies to the integer literal $w(u) < w(v)$. Due to the existence of the infinite chain from u to v , $u \leq_n^w v$ simplifies to false.

6.2 Reduction of $u <_n^p v$ and $u \leq_n^p v$

Reduction of $u <_n^p v$ and $u \leq_n^p v$ when u is not a constant. Let t be a proper subterm of u and we denote u by $u[t]$ to emphasize this occurrence of t . Let $u[t']$ be the term obtained from u by substituting t' for t . Assuming $u <^p v$, we have an infinite increasing chain

$$u[t] <^l u[f(t)] <^l u[f^2(t)] <^l \dots <^l u[f^n(t)] <^l \dots <^p v .$$

Therefore, $u <_n^p v$ simplifies to $u <^p v$, which is just $q < p$ assuming $\alpha_p = \text{type}(u)$ and $\alpha_q = \text{type}(v)$. For the same reason as before, $u \leq_n^p v$ simplifies to false.

Reduction of $u <_n^p v$ and $u \leq_n^p v$ when u is not a constant. Let $\text{type}(v) = \alpha_q$. We first check if there exists a non-constant ground TA-term $t[s]$ (where s is a proper subterm) such that $w(t[s]) = w(u)$ and $\text{type}(t[s]) = \alpha_p$, and $u <^\Sigma \alpha_p <^\Sigma \alpha_q$. This condition can be formally expressed as

$$\bigvee_{q < p < u, \alpha_p \notin \mathcal{A}} \text{Tree}^{\alpha_p}(u^w) , \quad (26)$$

whose truth value can be determined statically given a fixed signature. If (26) holds, we have an infinite chain

$$u <^p t[s] <^l t[f(s)] <^l t[f^2(s)] <^l \dots <^l t[f^n(s)] <^l \dots <^p v . \quad (27)$$

Therefore $u <_n^p v$ simplifies to true and $u \leq_n^p v$ simplifies to false. If (26) does not hold, then all ground TA-terms in between u and $O_{(v^w, q)}^p$ must be constants. Let us assume there are $m - 1$ constants in between u and $O_{(v^w, q)}^p$. If $m > n$, then $u <_n^p v$ simplifies to true while $u \leq_n^p v$ simplifies to false. If $m = n$, then both $u <_n^p v$ and $u \leq_n^p v$ simplify to true. If $m < n$, then $u <_n^p v$ simplifies to $O_{(v^w, q)}^p <_{n-m}^l v$ while

$u \leq_n^p v$ simplifies to $0_{(v^w, q)}^p \leq_{n-m}^1 v$. We need further consider three subcases under the assumption that $m < n$.

- (1) v is a constant. Then $0_{(v^w, q)}^p = v$, and so both $u <_n^p v$ and $u \leq_n^p v$ simplify to false.
- (2) v is of type f . Since (26) does not hold, $0_{(v^w, q)}^p$ must be $f(c_1)$ where c_1 is the smallest constant of weight $w(v)$. Suppose that there are r constants of weight $w(v)$ and they are ordered as

$$c_1 <^{kb} c_2 <^{kb} \dots <^{kb} c_r .$$

Then v must be of the form $f^l(c_r)$ ($0 < l \leq r$), where f^l denotes l applications of f . We have the following *tight* chain

$$\begin{aligned} & f(c_1) <^{kb} \dots <^{kb} f(c_{r-1}) <^{kb} f(c_r) \\ & <^{kb} f^2(c_1) <^{kb} \dots <^{kb} f^2(c_{r-1}) <^{kb} f^2(c_r) \\ & <^{kb} \dots \dots \\ & <^{kb} f^l(c_1) <^{kb} \dots <^{kb} f^l(c_{r-1}) <^{kb} f^l(c_r) . \end{aligned}$$

So $0_{(v^w, q)}^p <_{n-m}^1 v$ is true if and only if $(l-1)r + (r-1) \geq n-m$. The reduct can be formally expressed as

$$\bigvee_{1 \leq r' \leq r} \text{Is}_f(v) \wedge \text{Is}_f(s^f v) \wedge \dots \wedge \text{Is}_f((s^f)^{l(r')-1} v) \wedge c_{r'} = (s^f)^{l(r')} v , \quad (28)$$

where $l(r')$ is the minimum integer such that $(l(r')-1)r + (r'-1) \geq n-m$. A similar simplification holds for $0_{(v^w, q)}^p \leq_{n-m}^1 v$ except that we require that $(l(r')-1)r + (r'-1) = n-m$.

- (3) v is not a constant and not of type f . Let us assume $\text{ar}(\alpha_q) = k > 0$. Let $\langle l_1, \dots, l_k \rangle$ be the smallest k -integer tuple of weight $w(v) - w(\alpha_q)$ (with respect to the lexicographic extension of the integer ordering) such that $\text{Tree}(l_i)$ holds for all $i \in [1, k]$. We first assume that for some $k' \in [1, k]$,

$$\bigwedge_{1 \leq i \leq k'-1} (s_i^{\alpha_q} v)^w = l_i \wedge (s_{k'}^{\alpha_q} v)^w \neq l_{k'} . \quad (29)$$

In fact we must have $(s_{k'}^{\alpha_q} v)^w > l_{k'}$ because $\langle l_1, \dots, l_k \rangle$ is the smallest such k -tuple. Then we have $0_{(l_{k'})}^w <^w s_{k'}^{\alpha_q} v$, and hence $f^n(0_{(l_{k'})}^w) <^w s_{k'}^{\alpha_q} v$ for any $n > 0$. Therefore we have an infinite chain

$$\begin{aligned} 0_{(v^w, q)}^p & \leq^1 \alpha_q(0_{(i_1)}^w, \dots, 0_{(i_{k'})}^w, \dots, 0_{(i_k)}^w) \\ & <^1 \alpha_q(0_{(i_1)}^w, \dots, f(0_{(i_{k'})}^w), \dots, 0_{(i_k)}^w) \\ & <^1 \alpha_q(0_{(i_1)}^w, \dots, f^2(0_{(i_{k'})}^w), \dots, 0_{(i_k)}^w) \\ & <^1 \dots \dots \\ & <^1 \alpha_q(s_1^{\alpha_q} v, \dots, s_{k'}^{\alpha_q} v, \dots, s_k^{\alpha_q} v) = v . \end{aligned}$$

Hence under condition (29), $0_{(v^w, q)}^p <_{n-m}^l v$ simplifies to true and $0_{(v^w, q)}^p \leq_{n-m}^l v$ simplifies to false. Next we assume

$$\bigwedge_{0 < i \leq k} (s_i^{\alpha_q} v)^w = I_i \wedge \bigwedge_{0 < i \leq k} s_i^{\alpha_q} v = 0_{(I_i)}^w . \quad (30)$$

Under this condition, $0_{(v^w, q)}^p = v$, and hence both $0_{(v^w, q)}^p <_{n-m}^l v$ and $0_{(v^w, q)}^p \leq_{n-m}^l v$ simplify to false. If we assume that for some $k' \in [1, k-1]$,

$$\bigwedge_{0 < i \leq k} (s_i^{\alpha_q} v)^w = I_i \wedge s_{k'}^{\alpha_q} v \neq 0_{(I_{k'})}^w , \quad (31)$$

then we must have $0_{(I_{k'})}^w <^{pl} s_{k'}^{\alpha_q} v$. So we still have an infinite chain

$$\begin{aligned} 0_{(v^w, q)}^p &\leq^l \alpha_q(0_{(I_1)}^w, \dots, 0_{(I_{k'})}^w, \dots, 0_{(I_k)}^w) \\ &<^l \alpha_q(0_{(I_1)}^w, \dots, 0_{(I_{k'})}^w, \dots, f(0_{(I_k)}^w)) \\ &<^l \alpha_q(0_{(I_1)}^w, \dots, 0_{(I_{k'})}^w, \dots, f^2(0_{(I_k)}^w)) \\ &<^l \dots \\ &<^l \alpha_q(s_1^{\alpha_q} v, \dots, s_{k'}^{\alpha_q} v, \dots, s_k^{\alpha_q} v) = v . \end{aligned}$$

Therefore under condition (31), $0_{(v^w, q)}^p <_{n-m}^l v$ simplifies to true and $0_{(v^w, q)}^p \leq_{n-m}^l v$ simplifies to false. The last case is that

$$\bigwedge_{0 < i \leq k} (s_i^{\alpha_q} v)^w = I_i \wedge \bigwedge_{1 \leq i \leq k-1} s_i^{\alpha_q} v = 0_{(I_i)}^w \wedge s_k^{\alpha_q} v \neq 0_{(I_k)}^w . \quad (32)$$

Then $0_{(v^w, q)}^p <_{n-m}^l v$ reduces to $0_{(I_k)}^w <_{n-m}^{pl} s_k^{\alpha_q} v$ and $0_{(v^w, q)}^p \leq_{n-m}^l v$ reduces to $0_{(I_k)}^w \leq_{n-m}^{pl} s_k^{\alpha_q} v$. Now we have to reapply the above reduction to $0_{(I_k)}^w <_{n-m}^{pl} s_k^{\alpha_q} v$ and to $0_{(I_k)}^w \leq_{n-m}^{pl} s_k^{\alpha_q} v$. But since $I_k < w(v)$, this process eventually terminates.

Note that the lower boundary terms we used in the reduction are all defined on known constant weights, and hence they are fixed ground TA-terms which can be expressed using selector terms and constants.

6.3 Reduction of $u <_n^l v$ and $u \leq_n^l v$

If either u or v is a constant, then both $u <_n^l v$ and $u \leq_n^l v$ simplify to false. Now assume $\text{type}(u) = \text{type}(v) = \alpha_q$ and $\text{ar}(\alpha_q) = k > 0$. So

$$u = \alpha_q(s_1^{\alpha_q} u, \dots, s_k^{\alpha_q} u) , \quad v = \alpha_q(s_1^{\alpha_q} v, \dots, s_k^{\alpha_q} v) .$$

If we assume that

$$\bigwedge_{1 \leq i \leq k-1} s_i^{\alpha_q} u = s_i^{\alpha_q} v \wedge s_{k'}^{\alpha_q} u <^{kb} s_{k'}^{\alpha_q} v \quad (33)$$

for some $k' \in [1, k-1]$, then we have an infinite chain

$$\begin{aligned}
u &= \alpha_q(s_1^{\alpha_q} u, \dots, s_{k'}^{\alpha_q} u \dots, s_k^{\alpha_q} u) \\
&<^l \alpha_q(s_1^{\alpha_q} u, \dots, s_{k'}^{\alpha_q} u \dots, f(s_k^{\alpha_q} u)) \\
&<^l \alpha_q(s_1^{\alpha_q} u, \dots, s_{k'}^{\alpha_q} u \dots, f^2(s_k^{\alpha_q} u)) \\
&<^l \dots \dots \\
&<^l \alpha_q(s_1^{\alpha_q} v, \dots, s_{k'}^{\alpha_q} v \dots, s_k^{\alpha_q} v) = v .
\end{aligned}$$

Therefore under condition (33), $u <_n^l v$ simplifies to true and $u \cong_n^l v$ simplifies to false. The only possibility left is

$$\bigwedge_{1 \leq i \leq k-1} s_i^{\alpha_q} u = s_i^{\alpha_q} v \wedge s_k^{\alpha_q} u <^{kb} s_k^{\alpha_q} v . \quad (34)$$

Under this condition, $u <_n^l v$ simplifies to $s_k^{\alpha_q} u <_n^{kb} s_k^{\alpha_q} v$, and $u \cong_n^l v$ simplifies to $s_k^{\alpha_q} u \cong_n^{kb} s_k^{\alpha_q} v$.

This concludes all reductions we need to consider. Although we carried out the reductions at the semantic level by case distinction, all preconditions and the corresponding reducts are expressible in our formal language, and hence so are the reductions themselves. For example, The reductions of $u <_n^l v$ and $u \cong_n^l v$, respectively, are

$$\begin{aligned}
u <_n^l v &\rightarrow s_k^{\alpha_q} u <_n^{kb} s_k^{\alpha_q} v \vee \bigvee_{0 < i < k} s_i^{\alpha_q} u <^{kb} s_i^{\alpha_q} v , \\
u \cong_n^l v &\rightarrow s_k^{\alpha_q} u <_n^{kb} s_k^{\alpha_q} v \wedge \bigwedge_{0 < i < k} s_i^{\alpha_q} u = s_i^{\alpha_q} v .
\end{aligned}$$

Note that that we introduced boundary functions to delineate $<_n^w$, $<_n^p$, \cong_n^w and \cong_n^p , and to decompose $<_n^l$ and \cong_n^l . In the above reductions, however, not only are upper boundary functions undefined, but also the lower boundary functions have no use in these reductions (except in the intermediate steps). As a result, we do not need to introduce any boundary functions into $\mathcal{L}_{kb^+}^Z$ at all. Also note that we introduced tuple gap orders and tuple boundary functions to decompose $<_n^l$ and \cong_n^l ; also these, however, are not necessary in the above reductions. Hence we do not need to introduce tuple gap orders and tuple boundary functions into $\mathcal{L}_{kb^+}^Z$ either. Thus, the presence of a 0-weight unary function significantly simplifies the depth reduction procedure. In fact, we have already presented the proof for Lemmas 4.24. All other lemmas related to tuples and boundary functions in Section 4.9 and the corresponding proofs in Section A are no longer needed. Neither is Section B nor Section C therefore.

7. CONCLUSION

We showed the decidability of the first-order theory of term algebras with Knuth-Bendix order by quantifier elimination. Our method combines the extraction of integer constraints from term constraints with the reduction of quantifiers on term variables to quantifiers on integer variables. In fact, we established the decidability of a much more expressive theory.

Two problems related to practical complexity need further investigation. First, as a rule of thumb, more expressive power means higher complexity. Even if the theoretical complexity bound is the same, in practice the efficiency will be compromised. It is worthwhile to search for the smallest extension of KBO that admits quantifier elimination. Second, the elimination is intrinsically limited to processing quantified variables one at a time. We plan to extend the method in [Zhang et al. 2006] to eliminate a block of quantifiers of the same kind in one step. We believe this will be a significant improvement in pragmatic terms, since in most applications the quantifier alternation depth is small.

We also plan to investigate the decidability issue of the first-order theory of KBO in a non-ground term domain [Knuth and Bendix 1970], or with a partial precedence order on the signature [Baader and Nipkow 1999].

APPENDIX

A. PROOFS

In this section we present the proofs not included in the main text of this paper. These proofs rely on definitions about integer predicates and functions defined in Appendix B and reductions of equalities and gap order literals presented in Appendix C. In this and the following sections, unless stated otherwise, we assume that our language does not contain a 0-weight unary function. For ease of reference we restate the lemmas and theorems.

Lemma 4.21 (Elimination of Negative Literals). Assume formulas are type complete.

- (1) Any negative tester literal is equivalent to a disjunction of positive tester literals.
- (2) Any negative term equality literal (i. e., disequality between TA-terms) is equivalent to a disjunction of positive order literals.
- (3) Any negative order literal is equivalent to a disjunction of integer literals, tester literals and positive order literals.

PROOF. Negative tester literals can be eliminated using the following two equivalences.

$$\neg \text{Is}_{\alpha_p}(x) \leftrightarrow \bigvee_{\alpha_q \in C, \alpha_p \neq \alpha_q} \text{Is}_{\alpha_q}(x) , \quad \neg \text{Is}_A(x) \leftrightarrow \bigvee_{\alpha_p \in C \setminus \mathcal{A}} \text{Is}_{\alpha_p}(x) .$$

Term disequalities can be eliminated according to

$$u \neq v \leftrightarrow u <^w v \vee u <^p v \vee u <^l v \vee v <^w u \vee v <^p u \vee v <^l u .$$

Now let $\sharp \in \{w, p, l\}$ and $n > 0$. It is easily seen that the following equivalences

hold.

$$\begin{aligned}
 u <_n^\# v &\leftrightarrow u <_{n+1}^\# v \vee u \leq_n^\# v, \\
 \neg(u <_n^\# v) &\leftrightarrow \neg(u <_{n+1}^\# v) \wedge \neg(u \leq_n^\# v), \\
 u \leq_n^\# v &\leftrightarrow (u <_n^\# v) \wedge \neg(u <_{n+1}^\# v), \\
 \neg(u \leq_n^\# v) &\leftrightarrow \neg(u <_n^\# v) \vee u <_{n+1}^\# v.
 \end{aligned}$$

Therefore, for $n > 0$, we have the following equivalences.

$$\begin{aligned}
 \neg(u <_n^w v) &\leftrightarrow v^w \leq u^w \vee \bigvee_{i < n} u \leq_i^w v, \\
 \neg(u \leq_n^w v) &\leftrightarrow v^w \leq u^w \vee \bigvee_{i < n} u \leq_i^w v \vee u <_{n+1}^w v, \\
 \neg(u <_n^p v) &\leftrightarrow u^w \neq v^w \vee \left(\bigvee_{0 < p \leq |\Sigma|} \text{Is}_{\alpha_p}(u) \leftrightarrow \text{Is}_{\alpha_p}(v) \right) \vee v <^p u \vee \bigvee_{i < n} u \leq_i^p v, \\
 \neg(u \leq_n^p v) &\leftrightarrow u^w \neq v^w \vee \left(\bigvee_{0 < p \leq |\Sigma|} \text{Is}_{\alpha_p}(u) \leftrightarrow \text{Is}_{\alpha_p}(v) \right) \\
 &\quad \vee v <^p u \vee \bigvee_{i < n} u \leq_i^p v \vee u <_{n+1}^p v, \\
 \neg(u <_n^l v) &\leftrightarrow u^w \neq v^w \vee \left(\bigwedge_{0 < p \leq |\Sigma|} \text{Is}_{\alpha_p}(u) \leftrightarrow \neg \text{Is}_{\alpha_p}(v) \right) \vee v <^l u \vee \bigvee_{i < n} u \leq_i^l v, \\
 \neg(u \leq_n^l v) &\leftrightarrow u^w \neq v^w \vee \left(\bigwedge_{0 < p \leq |\Sigma|} \text{Is}_{\alpha_p}(u) \leftrightarrow \neg \text{Is}_{\alpha_p}(v) \right) \\
 &\quad \vee v <^l u \vee \bigvee_{i < n} u \leq_i^l v \vee u <_{n+1}^l v.
 \end{aligned}$$

Note that $\neg(u <_0^\# v)$ is $u \neq v \wedge \neg(u <^\# v)$ and $\neg(u \leq_0^\# v)$ is just $u \neq v$. Under the assumption of type completeness, the truth values of tester literals are predetermined, and hence no conjunction is introduced in the above reductions. And obviously, none of the above reductions introduces more open gap order literals in any of the reducts. \square

Lemma 4.22 (Delineated Gap Order Completion). Any conjunction of positive order literals in $\mathcal{L}_{\text{kb}^+}$ can be effectively transformed to an equivalent finite disjunction of delineated gap order completions.

PROOF. Let φ_{kb^+} be a conjunction of positive order literals in $\mathcal{L}_{\text{kb}^+}$. Without loss of generality we assume that φ_{kb^+} only contains literals like $u \leq_n^\# v$ and $u <_n^\# v$ ($\# \in \{w, p, l\}$) for $n > 0$. Let $\mathcal{T}(\varphi_{\text{kb}^+})$ be the set of all TA-terms in φ_{kb^+} . In the following we describe a nondeterministic algorithm to compute a disjunction of DGOs equivalent to φ_{kb^+} .

We view φ_{kb^+} as a labeled directed graph G_{kb^+} in which TA-terms in $\mathcal{T}(\varphi_{\text{kb}^+})$ are vertex labels and $u \leq_n^\# v$ (or $u <_n^\# v$) represents a directed *edge* from u to v with number label n and color label $\#$. An edge corresponding to a stretchable gap order

literal $u \prec_n^\# v$ (resp. a rigid gap order literal $u \preceq_n^\# v$) is called *stretchable* (resp. *rigid*). A path is *rigid* if every edge on the path is rigid; it is *stretchable* otherwise. The *length of a path* is the sum of all number labels on the path.

First we guess an equality partition for $\mathcal{T}(\varphi_{kb^+})$. All vertexes in the same equivalence class are merged by consolidating corresponding outgoing and incoming edges. We choose an arbitrary representative for each equivalence class as the vertex label.

Second we guess a linear order with respect to $\preceq_1^\#$ and $\prec_1^\#$ for $\mathcal{T}(\varphi_{kb^+})$. It amounts to adding to G_{kb^+} a Hamiltonian path: a path between two distinct vertexes of G_{kb^+} that visits each vertex exactly once. We call this path the *primary path* and the edges on this path *primary edges*. The primary path corresponds to a gap order completion and it is a prototype of the delineated gap order completion we are about to construct. At this phase, a primary edge may be labeled by any color in $\{w, p, l\}$, but all number labels are 1.

Third we insert boundary terms to delineate the primary path if necessary.

Up to now there may exist two types of inconsistency.

- (1) G_{kb^+} contains a loop.
- (2) The color of a non-primary edge is incompatible with colors on the corresponding primary path. Let order $\{w, p, l\}$ be $l \prec^c p \prec^c w$. We say that the color c of a non-primary edge is incompatible with colors $\{c_i \mid 0 < i \leq k\}$ on the corresponding primary path if $c \neq \max\{c_i \mid 0 < i \leq k\}$ with respect to \prec^c . For example, $u \prec_{n_1}^l v \prec_{n_2}^p w$ is incompatible with neither $u \prec_{n_3}^w w$ nor $u \prec_{n_4}^l w$ while it is compatible with $u \prec_{n_5}^p w$.

If an inconsistency is detected, the constraint simplifies to false.

To reduce φ_{kb^+} to a DGOC, we need to remove all non-primary edges. A non-primary edge from u to v is called an *over-edge* (resp. *under-edge*, *para-edge*) if its length is greater than (resp. less than, equal to) the length of the primary path from u to v . We say that a non-primary edge from u to v *covers* all primary edges on the path from u to v .

We remove all *redundant edges* which either are para-edges or stretchable under-edges. If there exists a rigid under-edge, then the constraint simplifies to false. Now we assume that all under-edges are removed.

The existence of an over-edge means the current gap order completion “over-approximates” φ_{kb^+} . To discharge an over-edge, we distribute the extra length over the number labels on primary stretchable edges that is covered by this over-edge. For example consider the following primary path from u to v to w .

$$u \xrightarrow{p_1} v \xrightarrow{p_2} w, \quad (35)$$

where both edges are stretchable. Suppose that there exists an over-edge from u to w with the number label $p > p_1 + p_2$. We replace the subgraph (35) by one of the following.

$$u \xrightarrow{p_1+n_1} v \xrightarrow{p_2+n_2} w, \quad (36)$$

where $n_1, n_2 \geq 0$ and $n_1 + n_2 = p - p_1 - p_2$. Note that we do not add extra length to rigid primary edges, and if the corresponding primary path is not stretchable, then

φ_{kb^+} simplifies to false. After the distribution, we make each stretchable primary edge rigid if the corresponding over-edge that covers it is rigid. Now the original over-edge becomes redundant and we remove it. One by one, we can remove all over-edges and obtain a graph that only has primary edges (false can be viewed as $x <^w x$). \square

The next lemma is used in the proof of Lemma 4.28.

LEMMA A.1 OPEN GAP ORDER LITERALS IN DGOCs. *Formulas of the form*

$$\bigwedge_{i=1}^m u_i \leq_{p_i}^{\sharp_i} x \wedge \bigwedge_{j=1}^n x \leq_{q_j}^{\flat_j} v_j, \quad (37)$$

where x is an ordinary TA-term, \leq stands for $<$ or \leq , $\sharp_i, \flat_j \in \{w, p, l\}$, and $p_i, q_j > 0$, can be effectively transformed into an equivalent finite disjunction of DGOCs such that the number of open gap order literals in each DGOC is no more than that in the original formula. Moreover, if both the predecessor and the successor of x in a DGOC are boundary terms, then the DGOC has one fewer open gap order literals than the original formula provided the original formula contains at least one open gap order literal.

PROOF. By Lemma 4.22 it suffices to show that in each DGOC the number of open gap order literals is no more than that in the original formula. Let us assume there are $m_1 \leq m$ ordinary TA-terms that are less than x and $n_1 \leq n$ ordinary TA-terms that are greater than x . So in the original formula there are $m_1 + n_1 + 1$ ordinary terms and $m_1 + n_1$ open gap order literals. In each obtained DGOC, the number of ordinary TA-terms can not exceed $m_1 + n_1 + 1$ as no new ordinary TA-terms are introduced. It is easily seen that in a linear order containing $k > 0$ ordinary TA-terms, the number of open gap order literals is at most $k - 1$. So there are at most $m_1 + n_1$ open gap order literals at each resulting DGOC. The introduction of boundary terms (to delineate a gap order) and the merge of two vertexes (to equalize two terms) can only decrease the number of open gap order literals. Moreover, suppose the original formula contains at least one open gap order literal. Without loss of generality, let us assume it is $u_1 \leq_1 x$, and so $m_1 > 0$. Since both the predecessor and the successor of x in the DGOC are boundary terms, x is separated from other ordinary TA-terms, and so there are at most $m_1 - 1$ open gap order literals at the left-hand side of x in the DGOC. Also there are at most n_1 open gap order literals at the right-hand side of x . Therefore, the total number of open gap order literals is $m_1 + n_1 - 1$, one less than that in the original formula. \square

Lemma 4.23 (No Embedding of Boundary Terms). Any formula in $\mathcal{L}_{\text{kb}^+}^Z$ can be effectively reduced to an equivalent formula in which no boundary terms appear inside selectors or the weight function.

PROOF. By the following equivalences we can eliminate all integer terms formed

by applying the weight function to boundary terms.

$$\begin{aligned}
(O_{(m)}^w)^w &\leftrightarrow (WD_{0^w}(m) \wedge m) \vee (\neg WD_{0^w}(m) \wedge w(\perp)) , \\
(O_{(m,p)}^p)^w &\leftrightarrow (WD_{0^w}(m,p) \wedge m) \vee (\neg WD_{0^w}(m,p) \wedge w(\perp)) , \\
(1_{(m)}^w)^w &\leftrightarrow (WD_{1^w}(m) \wedge m) \vee (\neg WD_{1^w}(m) \wedge w(\perp)) , \\
(1_{(m,p)}^p)^w &\leftrightarrow (WD_{1^w}(m,p) \wedge m) \vee (\neg WD_{1^w}(m,p) \wedge w(\perp)) .
\end{aligned}$$

Next we show how to eliminate selectors in front of boundary terms. For a selector term $s_i^{\alpha_p} O_{(m)}^w$ (where $\text{ar}(\alpha_p) = k$), we consider four cases.

- (1) $O_{(m)}^w$ is not well-defined. In this case $O_{(m)}^w = \perp$, and hence $s_i^{\alpha_p} O_{(m)}^w$ simplifies to \perp .
- (2) $O_{(m)}^w$ is well-defined, but $\text{Tree}^{(\alpha_p)}(m)$ is false. Since $\text{Tree}^{(\alpha_p)}(m)$ does not hold, $O_{(m)}^w$ can not be of type α_p , and so $s_i^{\alpha_p} O_{(m)}^w$ simplifies to $O_{(m)}^w$.
- (3) $\text{Tree}^{(\alpha_p)}(m)$ holds but there exists p' such that $p < p' \leq |\Sigma|$ and $\text{Tree}^{(\alpha_{p'})}(m)$ holds. Since $\alpha_{p'} <^\Sigma \alpha_p$, $O_{(m)}^w$ can not be of type α_p , and hence $s_i^{\alpha_p} O_{(m)}^w$ simplifies to $O_{(m)}^w$.
- (4) $\text{Tree}^{(\alpha_p)}(m)$ is true and for any p' such that $p < p' \leq |\Sigma|$, $\text{Tree}^{(\alpha_{p'})}(m)$ does not hold. The condition says that $O_{(m)}^w$ is of type α_p , i. e., $O_{(m)}^w = \alpha(t_1, \dots, t_{\text{ar}(\alpha_p)})$. It is not hard to argue that $\langle t_1, \dots, t_{\text{ar}(\alpha_p)} \rangle$ should be the smallest k -tuple of weight n . In Appendix B.2(4) it is shown that the weight of the i^{th} component of a k -tuple which is the smallest (with respect to $<^{k,\text{kb}}$) k -tuple of weight n , denoted by $\text{CWS}_i^k(n)$, can be defined in Presburger arithmetic. Therefore, we can simplify $s_i^{\alpha_p} O_{(m)}^w$ to $O_{(m)}^w$ with selector $(\text{CWS}_i^k(m-w(\alpha_p)))$.

The above four preconditions can be respectively expressed as

$$\begin{aligned}
&\neg WD_{0^w}(m) , \\
&WD_{0^w}(m) \wedge \neg \text{Tree}^{(\alpha_p)}(m) , \\
&\text{Tree}^{(\alpha_p)}(m) \wedge \bigvee_{p < p' \leq |\Sigma|} \text{Tree}^{(\alpha_{p'})}(m) , \\
&\text{Tree}^{(\alpha_p)}(m) \wedge \bigwedge_{p < p' \leq |\Sigma|} \neg \text{Tree}^{(\alpha_{p'})}(m) .
\end{aligned}$$

Similarly we can handle terms of the form $s_j^{\alpha_p} 1_{(m)}^w$.

For a selector term $s_j^{\alpha_p} O_{(m,p')}^p$ (where $\text{ar}(\alpha_p) = k$), we consider three cases.

- (1) $O_{(m,p')}^p$ is not well-defined. Then $s_j^{\alpha_p} O_{(m,p')}^p$ simplifies to \perp .
- (2) $O_{(m,p')}^p$ is well-defined, but $p \neq p'$. Then $s_j^{\alpha_p} O_{(m,p')}^p$ simplifies to $O_{(m,p')}^p$.
- (3) $O_{(m,p')}^p$ is well-defined, and $p = p'$. The condition says that $O_{(m,p)}^p = \alpha(t_1, \dots, t_{\text{ar}(\alpha_p)})$ and $\langle t_1, \dots, t_{\text{ar}(\alpha_p)} \rangle$ is the smallest k -tuple of weight n . As case (4) in the reduction of $s_i^{\alpha_p} O_{(m)}^w$, $s_j^{\alpha_p} O_{(m,p')}^p$ simplifies to $O_{(m,p')}^p$ with selector $(\text{CWS}_j^k(m-w(\alpha_p)))$.

The above three preconditions can be respectively expressed as

$$\begin{aligned} & \neg \text{WD}_{0^p}(m, p') , \\ & \text{WD}_{0^p}(m, p') \wedge p \neq p' , \\ & \text{WD}_{0^p}(m, p') \wedge p = p' . \end{aligned}$$

Similarly we can handle terms of the form $s_j^{\alpha_p} 1_{(m, p')^p}$.

Repeating this transformation, eventually all selectors in front of boundary terms are removed and we obtain a formula in which no boundary terms occur inside selectors.

We note that the above reduction is purely syntactic as for each case, both the precondition and the corresponding reduct are expressible in our formal language. In general, let $\varphi[t]$ be a formula in which term t occurs. Suppose there are n disjoint cases in total for the reduction of t and for each case i ($0 < i \leq n$), t is reduced to t_i under the precondition θ_i . Then $\varphi[t]$ can be equivalently rewritten as

$$\bigvee_{0 < i \leq n} (\theta_i \wedge \varphi[t_i]) . \quad (38)$$

It is easily seen from (38) that the transformation involves disjunctive splittings and since θ_i are Presburger formulas, the transformation does not increase the number of open gap order literals in any of the resulting conjunctions of literals. \square

Lemma 4.24 (Reduction of Term Gap Order Literals). Let

$$\star \in \{ <_n^{\text{kb}}, <_n^w, <_{n'}^p, <_{n'}^l, <_{n'}^{\text{pl}}, \leq_n^{\text{kb}}, \leq_{n'}^w, \leq_{n'}^p, \leq_{n'}^l, \leq_n^{\text{pl}} \} \quad (n > 0) .$$

If x is a TA-variable of type α_p with $\alpha_p = (s_1^{\alpha_p}, \dots, s_k^{\alpha_p})$ and t is an arbitrary TA-term, then $x \star t$ ($t \star x$) can be effectively reduced to an equivalent quantifier-free formula $\varphi(s_1^{\alpha_p} x, \dots, s_k^{\alpha_p} x)$ (in $\mathcal{L}_{\text{kb}^+}^Z$) where x only occurs in $s_i^{\alpha_p} x$ ($0 < i \leq k$).

PROOF. It suffices to only consider $\star \in \{ <_n^w, <_{n'}^p, <_{n'}^l, \leq_n^w, \leq_{n'}^p, \leq_{n'}^l \}$ for $n > 0$. If t is a TA-term not containing x , then the result follows directly from Lemma 4.26. Suppose otherwise. Then t must be in the form Lx where L is a non-empty block of selectors. We have two cases depending on whether Lx is at the left-hand side or the right-hand side of \star .

Case 1. $x \star Lx$. Since L is non-empty and constraints are type-complete, Lx is a proper subterm of x . If $x \star Lx$, then $x <^{\text{kb}} Lx$, violating the subterm property of KBO [Baader and Nipkow 1999]. Therefore, $x \star Lx$ simplifies to false.

Case 2. $Lx \star x$. If \star is $<_n^w$ (resp. $<_n^p$), reduction 201 (resp. 202) (see Appendix C) puts x and Lx in separate literals, and hence the problem goes away. Similarly if \star is \leq_n^w or \leq_n^p . The last two cases are $Lx <_n^l x$ and $Lx \leq_n^l x$, which are only possible if the language contains the unary function f of weight 0. To save notation, we use s_f to denote the corresponding selector of f . Now $Lx <_n^l x$ and $Lx \leq_n^l x$, respectively, are in the forms

$$s_f^i(x) <_n^l x \quad \text{and} \quad s_f^i(x) \leq_n^l x ,$$

where $i > 0$ and s_f^i denote i applications of s_f . Note that both $s_f^i(x) <_n^l x$ and $s_f^i(x) \leq_n^l x$ imply $\text{Is}_f(s_f^i(x))$. In the rest of the proof, we view x as a fixed ground TA-term unless it appears in a formula in the formal language. Let us consider two subcases.

A. There is a non-constant ground TA-term whose weight is $w(x)$ and type is not f .

Let $g(t_1, \dots, t_j)$ be such a term; i. e., $g \neq f$ and $w(g(t_1, \dots, t_j)) = w(x)$.

(a) If $s_f^i(x)$ is $f^k(a)$ (where $k > 0$ and a is a constant), then x is $f^{i+k}(a)$. We have either $a <^p g(t_1, \dots, t_j)$ or $g(t_1, \dots, t_j) <^p a$, as $w(g(t_1, \dots, t_j)) = w(x) = w(a)$.

i. $a <^p g(t_1, \dots, t_j)$. Then we have

$$f^k(a) <^l f^k(g(t_1, \dots, t_j)) <^l \dots <^l f^k(g(f^l(t_1), \dots, t_j)) <^l \dots <^l f^{i+k}(a) .$$

ii. $g(t_1, \dots, t_j) <^p a$. Then we have

$$f^k(a) <^l f^{i+k}(g(t_1, \dots, t_j)) <^l \dots <^l f^{i+k}(g(f^l(t_1), \dots, t_j)) <^l \dots <^l f^{i+k}(a) .$$

(b) If $s_f^i(x)$ is $f^k(h(t_1, \dots, t_j))$ (where $k > 0$ and $h \neq f$), then x is $f^{k+i}(h(t_1, \dots, t_j))$.

Then we have

$$f^k(h(t_1, \dots, t_j)) <^l \dots <^l f^k(h(f^l(t_1), \dots, t_j)) <^l \dots <^l f^{k+i}(a) .$$

In all cases, there are infinitely many terms in between $s_f^i(x)$ and x . Hence $s_f^i(x) <_n^l x$ simplifies to $\text{Is}_f(s_f^i(x))$ and $s_f^i(x) \leq_n^l x$ simplifies to false.

B. Any term of weight $w(x)$ is either an f -term (f -type term) or a constant. The condition says that $s_f^i(x)$ must be in the form $f^k(c)$ (for some constant c and $k > 0$) and then x is in the form $f^{k+i}(c)$. In this case, $f^k(c) \leq_r^l f^{k+i}(c)$ where r is i times the number of constants of weight $w(x)$. More precisely, let us assume that there are m constants of weight $w(x)$ ordered as follows:

$$c_1 <^{\text{kb}} \dots <^{\text{kb}} c_j <^{\text{kb}} \dots <^{\text{kb}} c_m ,$$

where $c \equiv c_j$. Then we have i tight chains each of which has length m as follows.

$$\begin{aligned} f^k(c_j) <^{\text{kb}} \dots <^{\text{kb}} f^k(c_m) <^{\text{kb}} f^{k+1}(c_1) <^{\text{kb}} \dots <^{\text{kb}} f^{k+1}(c_j) , \\ \dots \dots \dots \\ f^{k+i-1}(c_j) <^{\text{kb}} \dots <^{\text{kb}} f^k(c_m) <^{\text{kb}} f^{k+i}(c_1) <^{\text{kb}} \dots <^{\text{kb}} f^{k+i}(c_j) . \end{aligned}$$

Therefore, $s_f^i(x) <_n^l x$ simplifies to

$$\text{Is}_f(s_f^i(x)) \wedge \text{CNT}_m(x^w) ,$$

where $m = \lceil n/i \rceil - 1$ and $\text{CNT}_m(x^w)$ is defined in the language with f removed, and $s_f^i(x) \leq_n^l x$ simplifies to

$$\text{Is}_f(s_f^i(x)) \wedge \text{CNT}_m(x^w) \wedge \neg \text{CNT}_{m+1}(x^w) \wedge i \mid n ,$$

where $m = \lceil n/i \rceil - 1$ and $\text{CNT}_m(x^w)$ is defined the same.

Finally we note that each precondition can be expressed in Presburger arithmetic. Condition (A) is

$$\bigvee_{\alpha \in \mathcal{C} \setminus \mathcal{A} \setminus \{f\}} \text{Tree}^\alpha(\mathbf{x}^w), \quad (39)$$

and condition (B) is just the negation of (39). \square

Lemma 4.25 (Reduction of Closed Term Literals). Let

$$\star \in \{ =, <_n^{\text{kb}}, <_n^{\text{w}}, <_{n'}^{\text{p}}, <_{n'}^{\text{l}}, <_n^{\text{pl}}, \leq_n^{\text{kb}}, \leq_n^{\text{w}}, \leq_{n'}^{\text{p}}, \leq_{n'}^{\text{l}}, \leq_n^{\text{pl}} \} \quad (n > 0).$$

If $u \star v$ is closed, i. e., both u and v are boundary terms, then it can be effectively reduced to an equivalent Presburger formula.

PROOF. There are many combinations of closed term literals. Here we only highlight the key components in the proof and leave the details to Sections C.1, C.2 and C.3.

Counting Constraints. A closed term literal states that a certain number of ground TA-terms exist in a continuous segment formed by concatenating adjacent w -intervals or p -intervals. The idea of the reduction is to count how many ground TA-terms are in each interval. For example, $1_{(m)}^{\text{w}} = 0_{(m')}^{\text{w}}$ means that $m = m'$ and there is exactly one ground TA-term of weight m , which can be formally expressed as

$$m = m' \wedge \text{Tree}(m) \wedge \neg \text{CNT}_1(m).$$

As another example, $1_{(m)}^{\text{w}} <_n^{\text{w}} 0_{(m')}^{\text{w}}$ ($n > 0$) reduces to (by reduction (97) in Appendix C, copied below)

$$\left(\begin{array}{c} (n = 1 \rightarrow m < m') \\ \wedge \\ n > 1 \rightarrow \bigvee_{0 < r < n} (\exists z_1 \dots \exists z_r) \left(\begin{array}{c} m < z_1 < \dots < z_r < m' \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_1, \dots, n_r > 0}} \bigwedge_{0 < i \leq r} \text{CNT}_{n_i-1}(z_i) \end{array} \right) \end{array} \right). \quad (97)$$

We briefly explain the meaning of (97). Since $1_{(m)}^{\text{w}}$ is the largest TA-term of weight m and $0_{(m')}^{\text{w}}$ is the smallest TA-term of weight of m' , $1_{(m)}^{\text{w}} <_n^{\text{w}} 0_{(m')}^{\text{w}}$ means that there are at least $n - 1$ TA-terms whose weights are in (m, m') . If $n = 1$, then $1_{(m)}^{\text{w}} <_n^{\text{w}} 0_{(m')}^{\text{w}}$ obviously reduces to $m < m'$. In case $n > 1$, we assume the $n - 1$ TA-terms in between $1_{(m)}^{\text{w}}$ and $0_{(m')}^{\text{w}}$ are distributed in r ($0 < r < n$) different levels measured by weights, which is formally represented by

$$\bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_1, \dots, n_r > 0}} \bigwedge_{0 < i \leq r} \text{CNT}_{n_i-1}(z_i).$$

Since r ranges from 1 to $n - 1$ which is a constant, (97) is an abbreviation for a first-order formula.

The reduction of $1_{(m)}^w \leq_n^w 0_{(m')}^w$ is similarly obtained as (reduction (145), copied below)

$$\left(\begin{array}{c} n = 1 \rightarrow (m < m' \wedge \forall z (m < z < m' \rightarrow \neg \text{Tree}(z))) \\ \wedge \\ n > 1 \rightarrow \bigvee_{0 < r < n} (\exists z_1 \dots \exists z_r) \left(\begin{array}{c} m < z_1 < \dots < z_r < m' \\ \wedge \\ \neg (\exists z_1 \dots \exists z_{r+1}) (m < z_1 < \dots < z_{r+1} < m') \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_1, \dots, n_r > 0}} \bigwedge_{0 < i \leq r} \widehat{\text{CNT}}_{n_i-1}(z_i) \end{array} \right) \end{array} \right) .$$

Here we need to add two more conjuncts. When $n = 1$ we need

$$\forall z (m < z < m' \rightarrow \neg \text{Tree}(z))$$

to make sure that there exists no term whose weight is in (m, m') . When $n > 1$, we need

$$\neg (\exists z_1 \dots \exists z_r) (m < z_1 < \dots < z_{r+1} < m')$$

to guarantee that for a fixed $r \in (0, n)$ there are exactly r legitimate weights in (m, m') .

Well-definedness of Boundary Terms. The reductions listed in Sections C.1-C.5 implicitly assume well-definedness of boundary terms. The final reduction results should incorporate well-definedness predicates (in short, WD predicates, see Section B). For example, for $n > 1$, $1_{(m)}^w <_n^w 0_{(m')}^w$ is equivalent to

$$\begin{aligned} & \left(\text{WD}_{1^w}(m) \wedge \text{WD}_{0^w}(m') \wedge (97) \right) \\ & \vee \left(\neg \text{WD}_{1^w}(m) \wedge \text{WD}_{0^w}(m') \wedge \perp <_n^w 0_{(m')}^w \right) \\ & \vee \left(\text{WD}_{1^w}(m) \wedge \neg \text{WD}_{0^w}(m') \wedge 1_{(m)}^w <_n^w \perp \right) \\ & \vee \left(\neg \text{WD}_{1^w}(m) \wedge \neg \text{WD}_{0^w}(m') \wedge \perp <_n^w \perp \right) . \end{aligned}$$

However, we do not need to go through this detour if the parameters in boundary terms come from weights of TA-terms, because WD predicates are valid. For example, for $n > 1$, the final reduction result of $1_{(x^w)}^w <_n^w 0_{(y^w)}^w$ is just

$$\left(\begin{array}{c} (n = 1 \rightarrow x^w < y^w) \\ \wedge \\ n > 1 \rightarrow \bigvee_{0 < r < n} (\exists z_1 \dots \exists z_r) \left(\begin{array}{c} x^w < z_1 < \dots < z_r < y^w \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_1, \dots, n_r > 0}} \bigwedge_{0 < i \leq r} \text{CNT}_{n_i-1}(z_i) \end{array} \right) \end{array} \right) .$$

Note that we use \perp as the “sink value” to make boundary functions total. In general, the choice of a sink value affects the truth values of formulas in $\mathcal{L}_{\text{kb}^+}^{\mathbb{Z}}$; i. e., $\text{Th}(\text{TA}_{\text{kb}^+}^{\mathbb{Z}})$ varies with the sink value. However, $\text{Th}(\text{TA}_{\text{kb}})$ remains the same thanks to the incorporation of WD predicates.

Reduction of Relations Involving Constants. There is no separate list of reductions involving constants, as constants are just boundary terms. For example, $0_{(w(\alpha_p), p)}^p$ is another name for a constant α_p . So is $1_{(w(\alpha_p), p)}^p$.

Summary. Appendix C provides a full list of the relevant reductions as follows:

- C.1. closed term equalities;
- C.2. closed stretchable term gap order literals;
- C.3. closed rigid term gap order literals.

□

Lemma 4.26 (Reduction of Non-closed Term Gap Order Literals). Let

$$\star \in \{ \prec_n^{\text{kb}}, \prec_n^w, \prec_n^p, \prec_n^l, \prec_n^{\text{pl}}, \preceq_n^{\text{kb}}, \preceq_n^w, \preceq_n^p, \preceq_n^l, \preceq_n^{\text{pl}} \} \quad (n > 0) .$$

If x is a TA-variable of type α_p with $\alpha_p = (s_1^{\alpha_p}, \dots, s_k^{\alpha_p})$ and t is either a boundary term or an ordinary TA-term *not* containing x , then $x \star t (t \star x)$ can be effectively reduced to an equivalent quantifier-free formula $\varphi(s_1^{\alpha_p} x, \dots, s_k^{\alpha_p} x)$ where x only occurs in $s_i^{\alpha_p} x$ ($0 < i \leq k$).

PROOF. It suffices to only consider $\star \in \{ \prec_n^w, \prec_n^p, \prec_n^l, \preceq_n^w, \preceq_n^p, \preceq_n^l \}$ for $n > 0$. As before, we leave the detailed reductions to Sections C.4-C.7, and only highlight the key components different from those in the proof of Lemma 4.25.

Gap Order Delineation. The general reduction rule is to delineate a gap order to “indecomposable” intervals by inserting boundary terms. Let us first consider open gap order literals. Suppose both u and v are ordinary TA-terms, $u \prec_n^w v$ ($n > 1$) reduces to

$$\bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} u \prec_{n_1}^{\text{pl}} 1_{(u^w)}^w \prec_{n_2}^w 0_{(v^w)}^w \prec_{n_3}^{\text{pl}} v . \quad (40)$$

Here is the intuition behind this reduction. $u \prec_n^w v$ states that $u <^w v$ and there is an increasing chain of length n from u to v . On this chain from u to v we have to go across two boundaries, $1_{(u^w)}^w$ first and $0_{(v^w)}^w$ second. The two boundaries delineate the chain into three segments and the sum of the lengths of the three segments should be greater than or equal to n . Since there are infinitely many such combinations, a naive representation would require infinitely many disjunctions of conjunctions of literals. However, we can find a finite *cover* (an equivalent finite subset) by requiring that the sum of the lengths of the three segments is just equal to n , because for any $k > 0$,

$$\forall n_1, \dots, \forall n_k \left(\sum_{i=1}^k n_i \geq n \rightarrow (\exists n'_1 \leq n_1), \dots, (\exists n'_k \leq n_k) \sum_{i=1}^k n'_i = n \right)$$

is valid in \mathbb{N} , and because for any $n, n' \in \mathbb{N}$, $n < n'$ implies that

$$\forall x, y \left(x <_n^\# y \leftrightarrow (x <_n^\# y \vee x <_{n'}^\# y) \right),$$

where $\# \in \{w, p, l\}$, is valid in $\mathcal{L}_{kb^+}^Z$.

The reduction for rigid gap order literals is similar. For example, $u \leq_n^w v$ ($n > 1$) reduces to

$$\bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} u \leq_{n_1}^{pl} 1_{(u^w)}^w \leq_{n_2}^w 0_{(v^w)}^w \leq_{n_3}^{pl} v.$$

Recall that $x <^{pl} y \equiv x <^p y \vee x <^l y$. Suppose we choose the branch

$$u <_{n_1}^p 1_{(u^w)}^w <_{n_2}^w 0_{(v^w)}^w <_{n_3}^p v. \quad (41)$$

Assuming u is of type α_p , v is of type α_q , we can further delineate (41) to

$$\bigvee_{\substack{n_1=n_{11}+n_{12} \\ n_3=n_{31}+n_{32}}} u <_{n_{11}}^l 1_{(u^w,p)}^p <_{n_{12}}^p 1_{(u^w)}^w <_{n_2}^w 0_{(v^w)}^w <_{n_{31}}^p 0_{(v^w,q)}^p <_{n_{32}}^{pl} v. \quad (42)$$

In general we delineate $u <_n^\# v$ by inserting m boundary terms t_1, \dots, t_m in between u and v satisfying

$$u \leq_{n_0}^\# t_1 \leq_{n_1}^\# t_2 \leq_{n_2}^\# \dots \leq_{n_{m-1}}^\# t_m \leq_{n_m}^\# v, \quad (43)$$

where $\#_i \in \{w, p, l\}$ ($i \leq m$) and $\max\{\#_i \mid i \leq m\} = \#$. Recall that $l <^c p <^c w$. For example, $\max\{w, p\} = w$ and $\min\{p, l\} = l$. It is not hard to argue that $u <_n^\# v$ is equivalent to

$$\bigvee_{n_0+\dots+n_m=n} u <_{n_0}^\# t_1 <_{n_1}^\# t_2 <_{n_2}^\# \dots <_{n_{m-1}}^\# t_m <_{n_m}^\# v. \quad (44)$$

In such a delineation, $<_0^\#$ can be read as $=$. This saves unnecessary splittings for $\leq^\#$. However, it is not essential as the termination of our reductions does not rely on the decrease of gap counts.

Similarly, $u \leq_n^\# v$ can be delineated as

$$\bigvee_{n_0+\dots+n_m=n} u \leq_{n_0}^\# t_1 \leq_{n_1}^\# t_2 \leq_{n_2}^\# \dots \leq_{n_{m-1}}^\# t_m \leq_{n_m}^\# v. \quad (45)$$

The delineation of half open gap order literals is similar.

Termination. The termination of these reductions does not rely on the decrease of gap counts. Rather, it relies on the fact that a gap order is eventually delineated to “indecomposable” p-intervals. A closed equality literal or a closed gap order literal can be reduced to a Presburger formula (Sections C.1- C.3). So eventually half open gap order literals and open gap order literals only involve TA-terms in the same p-interval; i. e., they are in the forms $x <_n^l t$, $t <_n^l x$, $x \leq_n^l t$ and $t \leq_n^l x$ ($n > 0$) where x and t are of type α_p for some $\alpha_p = (s_1^{\alpha_p}, \dots, s_k^{\alpha_p})$.

Let us first consider literals like $x <_n^l t$. Suppose that t is an ordinary term. Then $x = \alpha_p(s_1^{\alpha_p} x, \dots, s_k^{\alpha_p} x)$, $t = \alpha_p(s_1^{\alpha_p} t, \dots, s_k^{\alpha_p} t)$, and $x <_n^l t$ is equivalent to

$$\langle s_1^{\alpha_p} x, \dots, s_k^{\alpha_p} x \rangle <_n^{k;kb} \langle s_1^{\alpha_p} t, \dots, s_k^{\alpha_p} t \rangle ,$$

which, by Lemma 4.27, further reduces to a formula in $\mathcal{L}_{kb^+}^Z$ in terms of $s_i^{\alpha_p} x$ and $s_i^{\alpha_p} t$ ($0 < i \leq k$).

Now suppose that t is a boundary term, say $t \equiv 0_{(m)}^w$. We can rewrite t as $\alpha_p(s_1^{\alpha_p} 0_{(m)}^w, \dots, s_k^{\alpha_p} 0_{(m)}^w)$. So $x <_n^l t$ is equivalent to

$$\langle s_1^{\alpha_p} x, \dots, s_k^{\alpha_p} x \rangle <_n^{k;kb} \langle s_1^{\alpha_p} 0_{(m)}^w, \dots, s_k^{\alpha_p} 0_{(m)}^w \rangle ,$$

which, by Lemma 4.27, further reduces to a formula in $\mathcal{L}_{kb^+}^Z$ in terms of $s_i^{\alpha_p} x$ and $s_i^{\alpha_p} 0_{(m)}^w$ ($0 < i \leq k$). Each $s_i^{\alpha_p} 0_{(m)}^w$ must be the smallest (w.r.t. $<^{kb}$) of weight $(s_i^{\alpha_p} 0_{(m)}^w)^w$. By Lemma 4.23, $s_1^{\alpha_p} 0_{(m)}^w, \dots, s_k^{\alpha_p} 0_{(m)}^w$ are simplified to boundary terms not occurring inside selectors, say $0_{(f_1(m))}^w, \dots, 0_{(f_k(m))}^w$ where f_1, \dots, f_k are integer functions definable in Presburger arithmetic (Section B).

We have similar results for $t <_n^l x$, $x \leq_n^l t$ and $t \leq_n^l x$.

Summary. Appendix C provides a full list of the relevant reductions as follows:

- C.4. half-open stretchable gap order literals;
- C.5. half-open rigid gap order literals;
- C.6. open stretchable gap literals;
- C.7. open rigid gap literals.

□

Lemma 4.27 (Reduction of Tuple Literals). Let U, V be k -tuples of the same weight, and

$$\star \in \{ =, <_n^{k;kb}, <_n^{k;w}, <_n^{k;p}, <_n^{k;l}, <_n^{k;pl}, \leq_n^{k;kb}, \leq_n^{k;w}, \leq_n^{k;p}, \leq_n^{k;l}, \leq_n^{k;pl} \} \quad (k > 0, n > 0) .$$

- (1) If $U = \langle u_1, \dots, u_k \rangle$ is an ordinary tuple, then $U \star V$ ($V \star U$) can be effectively reduced to an equivalent quantifier-free formula $\varphi(u_1, \dots, u_k)$ (in $\mathcal{L}_{kb^+}^Z$) in which u_i ($0 < i \leq k$) does not occur inside selectors.
- (2) If $U \star V$ ($V \star U$) is a closed tuple, i. e., both U and V are boundary tuples, then it can be effectively reduced to an equivalent Presburger formula.

PROOF. There are many types of tuple literals, especially the closed tuple literals. However, as tuples are only used in the intermediate steps in the reduction, we only encounter a small portion of the combinations. The detailed reductions are given in Sections C.8-C.15. Here we only highlight the key components of the proof.

Notations. First recall that we define tuple relations between tuples of the same weight. So tuple relations are not only parameterized by k , the tuple length, but are also parameterized by m , the total weight.

Second recall that we define suborders on tuples $\langle u_1, \dots, u_k \rangle \prec_n^{k;\#} \langle v_1, \dots, v_k \rangle$ ($\# \in \{w, p, l\}$) as

$$u_1 \prec_n^{k;\#} v_1 \vee \left(u_1 = v_1 \wedge \langle u_2, \dots, u_k \rangle \prec_n^{k;\#} \langle v_2, \dots, v_k \rangle \right), \quad (46)$$

instead of as

$$\exists i(0 < i \leq k) \left[u_i \prec_n^{k;\#} v_i \wedge \forall j(1 \leq j < i) u_j = v_j \right]. \quad (47)$$

As we shall see soon, this technical choice gives reducts (reduced formulas) a uniform appearance.

Third recall that $\langle u_1, \dots, u_k \rangle \prec_n^{k;\#} \langle v_1, \dots, v_k \rangle$ is *proper* if $u_1 \prec_n^{k;\#} v_1$. We claim that improper orders can be decomposed to a Boolean combination of proper orders between k -tuples and proper orders between tuples of shorter length. We first show that we can always write $U \prec_n^{k;\#} V$ into the form

$$\langle u_1, \dots, u_k \rangle \prec_n^{k;\#} \langle v_1, \dots, v_k \rangle.$$

This trivially holds if both U and V are ordinary tuples. Now suppose U is a boundary tuple. We claim that $U = \langle v_1, \dots, v_k \rangle$ where u_1, \dots, u_k are boundary terms. In fact we have

$$\begin{aligned} \bar{1}_{(sum)}^{k;kb} &= \langle 1_{(CWL_1^k(sum))}^w, 1_{(CWL_2^k(sum))}^w, \dots, 1_{(CWL_k^k(sum))}^w \rangle, \\ \bar{1}_{(sum,m)}^{k;w} &= \langle 1_{(m)}^w, 1_{(CWL_1^{k-1}(sum-m))}^w, \dots, 1_{(CWL_{k-1}^{k-1}(sum-m))}^w \rangle, \\ \bar{1}_{(sum,m,p)}^{k;p} &= \langle 1_{(m,p)}^w, 1_{(CWL_1^{k-1}(sum-m))}^w, \dots, 1_{(CWL_{k-1}^{k-1}(sum-m))}^w \rangle, \\ \bar{0}_{(sum)}^{k;kb} &= \langle 0_{(CWS_1^k(sum))}^w, 0_{(CWS_2^k(sum))}^w, \dots, 0_{(CWS_k^k(sum))}^w \rangle, \\ \bar{0}_{(sum,m)}^{k;w} &= \langle 0_{(m)}^w, 0_{(CWS_1^{k-1}(sum-m))}^w, \dots, 0_{(CWS_{k-1}^{k-1}(sum-m))}^w \rangle, \\ \bar{0}_{(sum,m,p)}^{k;p} &= \langle 0_{(m,p)}^w, 0_{(CWS_1^{k-1}(sum-m))}^w, \dots, 0_{(CWS_{k-1}^{k-1}(sum-m))}^w \rangle, \end{aligned}$$

where $CWL_i^k(n)$ (resp. $CWS_i^k(n)$) gives the weight of the i^{th} component of the largest (resp. the smallest) k -tuple (w.r.t. $\prec_n^{k;kb}$) of weight n (Appendix B.2(4-5)). The case that V is a boundary tuple is similar.

Therefore, $U \prec_n^{k;\#} V$ reduces to

$$\begin{aligned} & \left((u_1 = v_1) \wedge \langle u_2, \dots, u_k \rangle \prec_n^{k;kb} \langle v_2, \dots, v_k \rangle \right) \\ & \vee \left((u_1 \prec_n^{k;\#} v_1) \wedge \langle u_1, \dots, u_k \rangle \prec_n^{k;\#} \langle v_1, \dots, v_k \rangle \right), \end{aligned}$$

which contains two disjuncts; the first is a tuple order on tuples of a shorter length and the second is a proper tuple order on tuples of the same length. Iteratively applying the process, an improper k -tuple order can be decomposed to a Boolean combination of proper k' -tuple orders for $k' \leq k$. Therefore, it suffices to show *proper reductions*, the reductions of proper orders on tuples of the same weight, which are given in Sections C.8-C.15. All reduction rules therefore contain an implicit side condition $u_1 \prec_n^{k;\#} v_1$ or $u_1 \preceq_n^{k;\#} v_1$ in the redex (target formula).

Gap Order Delineation and Tuple Length Reduction. The general reduction rule is the same as in the proof of Lemma 4.26, namely, using boundary tuples to delineate a tuple order to “indecomposable” intervals. For example, Let $U = \langle u_1, \dots, u_k \rangle$, $V = \langle v_1, \dots, v_k \rangle$ be ordinary tuples, and $sum = \sum_{i=1}^k u_i^w = \sum_{i=1}^k v_i^w$. $U \leq_n^{k:w} V$ ($n > 0$) reduces to (reduction (343), copied below)

$$\bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \langle u_1, \dots, u_k \rangle \leq_{n_1}^{k:pl} \bar{1}_{(sum, u_1^w)}^{k:w} \leq_{n_2}^{k:w} \bar{0}_{(sum, v_1^w)}^{k:w} \leq_{n_3}^{k:pl} \langle v_1, \dots, v_k \rangle . \quad (343)$$

Now it is clear why we define $\langle u_1, \dots, u_k \rangle \leq_n^{k:\#} \langle v_1, \dots, v_k \rangle$ by (46), and not by (47). Suppose we did it in the other way. Then in the reduction of (343), we are not able to have $\langle u_1, \dots, u_k \rangle \leq_{n_1}^{k:pl} \bar{1}_{(sum, u_1^w)}^{k:w}$, because the order between $\langle u_1, \dots, u_k \rangle$ and $\bar{1}_{(sum, u_1^w)}^{k:w}$ may be determined by the weight of the i^{th} components for some $i > 1$.

Besides gap order delineation, we also need to decompose a tuple gap order using tuple gap orders of shorter lengths and term gap orders. For example, $\langle u_1, \dots, u_k \rangle \leq_{n_1}^{k:p} \bar{1}_{(sum, u_1^w)}^{k:w}$, which is a conjunct in a disjunct of (343), reduces to (reduction (326), copied below)

$$\bigvee_{\substack{n_{11}(n_{12}+1)+n_{13} \geq n_1, \\ n_{11}, n_{12}, n_{13} \leq n_1}} u_1 \leq_{n_{11}}^p 1_{(u_1^w)}^w \wedge \bar{0}_{(sum-u_1^w)}^{k-1:kb} \leq_{n_{12}}^{k-1:kb} \bar{1}_{(sum-u_1^w)}^{k-1:kb} \wedge \langle u_2, \dots, u_k \rangle \leq_{n_{13}}^{k-1:kb} \bar{1}_{(sum-u_1^w)}^{k-1:kb} . \quad (326)$$

We briefly explain the reason behind (326). The tuples in between $\langle u_1, \dots, u_k \rangle$ and $\bar{1}_{(sum, u_1^w)}^{k:w}$ can be divided into $n_{11} + 1$ segments where $n_{11} + 1$ is the number of ground TA-terms in between u_1 and $1_{(u_1^w)}^w$ (including u_1 and $1_{(u_1^w)}^w$). Each of the first n_{11} segments contains $n_{12} + 1$ k -tuples, where $n_{12} + 1$ is the number of $(k-1)$ -tuples of weight $sum - u_1^w$. The last segment contains n_{13} k -tuples because n_{13} is the number of $(k-1)$ -tuples greater than or equal to $\langle u_2, \dots, u_k \rangle$ but less than $\bar{1}_{(sum-u_1^w)}^{k-1:kb}$. We require $n_{11}(n_{12} + 1) + n_{13} \geq n_1$ and $n_{11}, n_{12}, n_{13} \leq n_1$ to have finitely many disjunctions. It suffices because for any $k > 0$ and for any polynomial $P(x_1, \dots, x_k)$ only containing positive coefficients, the following formula is valid in \mathbb{N} .

$$\forall m_1, \dots, \forall m_k \left[P(m_1, \dots, m_k) \geq m \rightarrow \exists m'_1 \leq m, \dots, \exists m'_k \leq m \left(\bigwedge_{i=1}^k m'_i \leq m_i \wedge P(m'_1, \dots, m'_k) \geq m \right) \right] .$$

The reduction of $U \leq_n^{k:\#} V$ is similarly defined with stretchable gap orders replaced by corresponding rigid gap orders. For example, $\langle u_1, \dots, u_k \rangle \leq_{n_1}^{k:p} \bar{1}_{(sum, u_1^w)}^{k:w}$, reduces

to (reduction (338), copied below)

$$\bigvee_{\substack{n_{11}(n_{12}+1)+n_{13}=n_1, \\ n_{11}, n_{12}, n_{13} \leq n_1}} \mathbf{u}_1 \cong_{n_{11}}^p \mathbf{1}_{(u_1^w)}^w \wedge \bar{\mathbf{O}}_{(sum-u_1^w)}^{k-1;kb} \cong_{n_{12}}^{k-1;kb} \bar{\mathbf{1}}_{(sum-u_1^w)}^{k-1;kb} \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \cong_{n_{13}}^{k-1;kb} \bar{\mathbf{1}}_{(sum-u_1^w)}^{k-1;kb}. \quad (338)$$

Here we require $n_{11}(n_{12} + 1) + n_{13} = n_1$ instead of $n_{11}(n_{12} + 1) + n_{13} \geq n_1$.

As an example for closed relations, we consider $\bar{\mathbf{1}}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{\mathbf{O}}_{(sum,m')}^{k:w}$ which reduces to (reduction (247), copied below)

$$\left(\begin{array}{c} (n = 1 \rightarrow m < m') \\ \wedge \\ n > 1 \rightarrow \\ \bigvee_{0 < r < n} \left(\begin{array}{c} m < CW_1^k(sum, m, m') < \dots < CW_r^k(sum, m, m') < m' \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_i > 0}} \bigwedge_{0 < i \leq r} \bar{\mathbf{O}}_{(sum, CW_i^k(sum, m, m'))}^{k:w} \prec_{n_i-1}^{k:w} \bar{\mathbf{1}}_{(sum, CW_i^k(sum, m, m'))}^{k:w} \end{array} \right) \end{array} \right). \quad (247)$$

We explain the meaning of (247). Since $\bar{\mathbf{1}}_{(sum,m)}^{k:w}$ is the largest k -tuple of weight sum with the first component having weight m and $\bar{\mathbf{O}}_{(m')}^w$ is the smallest tuple of weight of sum with the first component having weight m' , $\bar{\mathbf{1}}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{\mathbf{O}}_{(sum,m')}^{k:w}$ means that there are at least $n - 1$ k -tuples of weight sum in between $\bar{\mathbf{1}}_{(sum,m)}^{k:w}$ and $\bar{\mathbf{O}}_{(sum,m')}^{k:w}$ (but including neither $\bar{\mathbf{1}}_{(sum,m)}^{k:w}$ nor $\bar{\mathbf{O}}_{(sum,m')}^{k:w}$) and in each of these k -tuples the weight of the first component is greater than m but smaller than m' . If $n = 1$, then obviously $\bar{\mathbf{1}}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{\mathbf{O}}_{(sum,m')}^{k:w}$ reduces to $m < m'$. If $n > 1$, then for some r ($0 < r < n$), the $n - 1$ k -tuples distribute into r different levels according to their weights, which is formally represented as

$$\bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_i \geq 1}} \bigwedge_{0 < i \leq r} \bar{\mathbf{O}}_{(sum, CW_i^k(sum, m, m'))}^{k:w} \prec_{n_i-1}^{k:w} \bar{\mathbf{1}}_{(sum, CW_i^k(sum, m, m'))}^{k:w},$$

where $CW_i^k(n, m, m')$ gives the i^{th} smallest integer in (m, m') which is the weight of a component in a k -tuple of weight n (Appendix B.2(3)), and

$$\bar{\mathbf{O}}_{(sum, CW_i^k(sum, m, m'))}^{k:w} \prec_{n_i-1}^{k:w} \bar{\mathbf{1}}_{(sum, CW_i^k(sum, m, m'))}^{k:w}$$

states that there are n_i k -tuples of weight sum with the first component of weight $CW_i^k(sum, m, m')$. We require $\sum_{i=1}^r n_i = n - 1$ and $n_i \geq 1$ for $1 \leq i \leq r$.

The reduction of $\bar{\mathbf{1}}_{(sum,m)}^{k:w} \cong_n^{k:w} \bar{\mathbf{O}}_{(sum,m')}^{k:w}$ is similarly obtained as (reduction (295)),

copied below)

$$\left(\begin{array}{l} n = 1 \rightarrow (m < m' \wedge \forall z(m < z < m' \rightarrow \neg \text{IsTW}^k(m))) \\ \wedge \\ n > 1 \rightarrow \\ \bigvee_{0 < r < n} \left(\begin{array}{l} m < \text{CW}_1^k(\text{sum}, m, m') < \dots < \text{CW}_r^k(\text{sum}, m, m') < m' \\ \wedge \\ \neg (m < \text{CW}_1^k(\text{sum}, m, m') < \dots < \text{CW}_{r+1}^k(\text{sum}, m, m') < m') \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_i > 0}} \bigwedge_{0 < i \leq r} \bar{0}_{(\text{sum}, \text{CW}_i^k(\text{sum}, m, m'))}^{\leq k; w} \bar{1}_{(\text{sum}, \text{CW}_i^k(\text{sum}, m, m'))}^{\leq k; w} \end{array} \right) \end{array} \right) \quad . \quad (295)$$

Here we need to add two more conjuncts. When $n = 1$ we need

$$\forall z(m < z < m' \rightarrow \neg \text{IsTW}^k(m))$$

to make sure that there exists no k -tuple whose weight is in (m, m') . When $n > 1$, we need

$$\neg (m < \text{CW}_1^k(\text{sum}, m, m') < \dots < \text{CW}_{r+1}^k(\text{sum}, m, m') < m')$$

to guarantee that for a fixed $r \in (0, n)$ there are exactly r legitimate (tuple) weights in (m, m') .

Termination. As for Lemma 4.26, the termination of these reductions does not rely on the decrease of gap counts. Rather, a tuple gap order is first delineated to “indecomposable” intervals which is in turn reduced to tuple gap orders of shorter lengths. Recall that tuple gap orders of length 1 are essentially term gap orders. Repeatedly applying the reduction, eventually only term gap orders $<_n^\#$ and $\leq_n^\#$ ($\# \in \{w, p, l\}$) appear, i. e., a tuple gap order reduces to a formula in $\mathcal{L}_{\text{kb}^+}^{\mathbb{Z}}$, and hence the reduction terminates. In particular, a closed tuple gap order or a closed tuple equality literal reduces to a Presburger formula.

Well-definedness of Boundary Tuples. Reductions listed in Sections C.8-C.15 implicitly rely on the assumption that boundary tuples are well-defined. This assumption holds because parameters of boundary tuples come from weights of well-defined TA-terms.

Reduction of Literals Involving $\bar{0}_{(\text{sum})}^{k; \text{kb}}$ or $\bar{1}_{(\text{sum})}^{k; \text{kb}}$. There is no separate list for reductions of tuple gap orders involving $\bar{0}_{(\text{sum})}^{k; \text{kb}}$ or $\bar{1}_{(\text{sum})}^{k; \text{kb}}$, as they can be expressed, respectively, by

$$\bar{0}_{(\text{sum})}^{k; \text{kb}} = \bar{0}_{(\text{sum}, \text{MinCW}^k(\text{sum}))}^{k; w} \quad , \quad \bar{1}_{(\text{sum})}^{k; \text{kb}} = \bar{1}_{(\text{sum}, \text{MaxCW}^k(\text{sum}))}^{k; w} \quad '$$

where $\text{MinCW}^k(\text{sum})$ (resp. $\text{MaxCW}^k(\text{sum})$) gives the smallest (resp. the largest) integer in $(0, \text{sum} + 1)$ that is the weight of a component in a k -tuple having weight sum (Appendix B.2(1,2)).

The Number of Open Gap Order Literals. As we mentioned before, tuple gap orders reduce to term gap orders. The reduction does not increase the number of open term gap order literals. The only reductions that generate new open term gap order literals are that of $\langle u_1, \dots, u_k \rangle \prec_n^{k;l} \langle v_1, \dots, v_k \rangle$ (reduction (345)) and that of $\langle u_1, \dots, u_k \rangle \preceq_n^{k;l} \langle v_1, \dots, v_k \rangle$ (reduction (348)). Consider reduction (345) (copied below)

$$\bigvee_{\substack{n_1+n_2+n_3=n \\ n_3>0}} \left(\begin{array}{c} \langle u_2, \dots, u_k \rangle \prec_{n_1}^{k-1;kb} \bar{\uparrow}_{(rem)}^{k-1;kb} \\ \wedge \\ \bar{O}_{(rem)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \langle v_2, \dots, v_k \rangle \\ \wedge \\ \bigvee_{\substack{(m_1+1)(m_2-1) \geq (n_3-1) \\ m_2>0, m_1, m_2 < n_3}} \left(\begin{array}{c} \bar{O}_{(rem)}^{k-1;kb} \prec_{m_1}^{k-1;kb} \bar{\uparrow}_{(rem)}^{k-1;kb} \\ \wedge \\ u_1 \prec_{m_2}^l v_1 \end{array} \right) \end{array} \right), \quad (345)$$

where $rem = \sum_{i=2}^k u_i^w = \sum_{i=2}^k v_i^w$. It is clear that in (345) each disjunct (conjunction of literals) only contains one open gap order literal, namely $u_1 \prec_{m_2}^l v_1$. Since the reduction is triggered by an open gap order literal, there is no increase in the number of open gap order literals. Although $u_1 \prec^l v_1$ is an implicit open gap order literal in the redex, it is not included in any reducts, as it has been replaced by the more precise relation $u_1 \prec_{m_2}^l v_1$. The case for reduction (348) is similar.

Summary. Appendix C provides a full list of the relevant reductions as follows:

- C.8. closed tuple equality literals;
- C.9. half-open tuple equality literals;
- C.10. closed stretchable tuple gap order literals;
- C.11. closed rigid tuple gap order literals;
- C.12. half-open stretchable tuple gap order literals;
- C.13. half-open rigid tuple gap order literals;
- C.14. open stretchable tuple gap order literals;
- C.15. open rigid tuple gap order literals.

□

Lemma 4.28 (Elimination of Term Quantifiers). Let x be a term variable, $\varphi_{kb^+}(x)$ a conjunction of literals in \mathcal{L}_{kb^+} with $\text{depth}^{\varphi_{kb^+}}(x) = 0$, and $\varphi_{\mathbb{Z}}(x)$ a Presburger formula in which x occurs inside the weight function. Then

$$(\exists x : \mathbb{T}) \left[\varphi_{kb^+}(x) \wedge \varphi_{\mathbb{Z}}(x) \right]$$

can be effectively reduced to $\varphi'_{kb^+} \wedge \varphi'_{\mathbb{Z}}$ in which x does not occur and φ'_{kb^+} is quantifier-free.

PROOF. The antecedent of the lemma only requires that all *term occurrences* of x have depth 0 and so x may have arbitrarily many integer occurrences of the form

$(Lx)^w$ where L is a (possibly empty) selector sequence. We divide the proof into two parts.

Part 1. $\varphi_{\text{kb}^+}(x)$ contains $x = t$ where t may contain x . We rewrite $(\exists x : \mathbb{T})[\varphi_{\text{kb}^+}(x) \wedge \varphi_{\mathbb{Z}}(x)]$ as

$$(\exists x : \mathbb{T}) \left[x = t \wedge \varphi(x) \right], \quad (48)$$

where $\varphi(x)$ is a conjunction of literals in $\mathcal{L}_{\text{kb}^+}^{\mathbb{Z}}$. We consider four cases.

- (1) t does not contain x . Then (48) simplifies to $\varphi(t)$.
- (2) $t \equiv x$. Then (48) simplifies to $(\exists x)\varphi(x)$.
- (3) $t \equiv Lx$ where L is a non-empty selector block. Then (48) simplifies to false, since we assume that terms like Lx are proper (Definition 3.3).
- (4) t is a boundary term in which $(L_i x)^w$ ($i \leq n$) occur in an integer function. In this case we assume that (48) has the form

$$(\exists x : \mathbb{T}) \left[x = t[f((L_0 x)^w, \dots, (L_n x)^w)] \wedge \varphi(x) \right], \quad (49)$$

where $L_0 x \equiv x$ (i. e., $L_0 \equiv \emptyset$), $L_0 x, \dots, L_n x$ enumerate all selector terms containing x (including improper selector terms inside selectors), and $f((L_0 x)^w, \dots, (L_n x)^w)$ denotes the maximum integer term in t . We assume $t[f((L_0 x)^w, \dots, (L_n x)^w)]$ is a well-defined boundary term (Section B). Otherwise $t = \perp$ and hence (48) reduces to $\varphi(\perp)$. The set of terms $\{L_i x \mid i \leq n\}$ corresponds to a tree with root x . We assume that this tree is sibling complete. For example, a formula in List is sibling complete if for any term t it either contains both $\text{car}(t)$ and $\text{cdr}(t)$ or contains neither. In general, if we need $L'x$ to appear, we can just set

$$f((L_0 x)^w, \dots, (L_n x)^w, (L'x)^w) := f((L_0 x)^w, \dots, (L_n x)^w, x^w) + (L'x)^w - (L'x)^w.$$

It is easily seen that (49) is equivalent to

$$(\exists x : \mathbb{T}) \left[x = t[f((L_0 x)^w, \dots, (L_n x)^w)] \wedge \varphi(t[f((L_0 x)^w, \dots, (L_n x)^w)] \right], \quad (50)$$

where x only has one term occurrence, namely, as the left-hand side of

$$x = t[f((L_0 x)^w, \dots, (L_n x)^w)]. \quad (51)$$

Note that the substitution that we used to transform (49) to (50) may put a boundary term inside a selector term. In fact, it is only this type of substitution that makes a boundary term appear inside a selector. However, Lemma 4.23 eliminates this superficial complication. We claim that (50) is equivalent to

$$(\exists (L_0 x)^w, \dots, (L_n x)^w : \mathbb{Z}) \left[\varphi(t[f((L_0 x)^w, \dots, (L_n x)^w)]) \wedge \varphi_{\Delta}((L_0 x)^w, \dots, (L_n x)^w) \right], \quad (52)$$

where $(L_i x)^w$ are pseudo integer variables, and φ_{Δ} is the integer constraint on weights of subterms of x satisfying the following conditions.

- (a) φ_{Δ} requires that x be a legitimate term.
 - i. For each leaf node $L_i x$, φ_{Δ} includes $\text{Tree}^{\alpha_p}((L_i x)^w)$, where $\text{type}(L_i x) = \alpha_p$. This states that each leaf has a well-defined weight.

ii. For each non-leaf node $L_i x$, φ_Δ includes

$$(L_i x)^w = w(\alpha_p) + \sum_{j=1}^k (s_j^{\alpha_p} L_i x)^w ,$$

where $\text{type}(L_i x) = \alpha_p = (s_1^{\alpha_p}, \dots, s_k^{\alpha_p})$ and $s_1^{\alpha_p} L_i x, \dots, s_k^{\alpha_p} L_i x$ are immediate children of $L_i x$. This states that the weight of a tree is the sum of the weight of the root and the weights of the immediate children of the root.

(b) φ_Δ requires that x be a legitimate boundary term. For example, suppose

$$t \equiv 0_{(f((L_0 x)^w, \dots, (L_n x)^w))}^w . \quad (53)$$

- i. φ_Δ includes $x^w = f((L_0 x)^w, \dots, (L_n x)^w)$.
- ii. Each node $L_i x$ must be the smallest (w.r.t. $<^{kb}$) of weight $(L_i x)^w$. If $\text{type}(L_i x) = \alpha_p$, then there exists no p' such that $p < p' \leq |\Sigma|$ and $\text{Tree}^{\alpha_{p'}}((L_i x)^w)$ holds. This constraint is formally expressed as

$$\bigwedge_{p < p' \leq |\Sigma|} \neg \text{Tree}^{\alpha_{p'}}((L_i x)^w) .$$

- iii. Each non-leaf node must have the smallest tuple of children (w.r.t. $<^{k, kb}$). If $L_i x$ has k immediate siblings $s_1^{\alpha_p} L_i x, \dots, s_k^{\alpha_p} L_i x$ (in the order from left to right), we require that $\langle s_1^{\alpha_p} L_i x, \dots, s_k^{\alpha_p} L_i x \rangle$ be the smallest k -tuple (w.r.t. $<^{k, kb}$) of weight $n = \sum_{j=1}^k (s_j^{\alpha_p} L_i x)^w$. This constraint is formally expressed as

$$\bigwedge_{0 < j \leq k} (s_j^{\alpha_p} L_i x)^w = \text{CWS}_j^k(n) ,$$

where $\text{CWS}_j^k(n)$ gives the weight of the j^{th} component of the smallest k -tuple (w.r.t. $<^{k, kb}$) of weight n (Section B).

Similarly we can obtain constraints for

$$t \equiv 0_{(f((L_0 x)^w, \dots, (L_n x)^w), p)}^p , \quad t \equiv 1_{(f((L_0 x)^w, \dots, (L_n x)^w))}^w , \quad t \equiv 1_{(f((L_0 x)^w, \dots, (L_n x)^w), p)}^p .$$

The intuition behind this reduction is that when a term x occurs inside the weight function, all its term properties (that are expressible in the formal language) can be fully characterized by integer properties that are expressible in \mathcal{L}_Z . We therefore view $(L_i x)^w$ as syntactical integer variables. By variable renaming we transform (52) to

$$(\exists z_0, \dots, z_n : \mathbb{Z}) \left[\varphi(t[f(z_0, \dots, z_n)]) \wedge \varphi_\Delta(z_0, \dots, z_n) \right] . \quad (54)$$

We show equivalence of (50) and (54). Suppose x is a ground boundary term satisfying (50), then $w(L_0 x), \dots, w(L_n x)$ clearly satisfy $\varphi_\Delta((L_0 x)^w, \dots, (L_n x)^w)$, which describes integer constraints on $L_0 x, \dots, L_n x$. Therefore, (50) implies (54). For the reverse direction, let z_0, \dots, z_n be integers satisfying (54). Without loss of generality we assume t to be as in (53). We obtain an x by setting

$L_i x := O^p(z_i, p)$ for each leaf node $L_i x$ with type α_p . Such x is clearly a well-defined ground boundary term $O_{(f(z_0, \dots, z_n))}^w$ because $\varphi_\Delta(z_0, \dots, z_n)$ is a complete integer constraint on the weights of all subterms of $O_{(f(z_0, \dots, z_n))}^w$. Therefore, (50) is satisfied.

Part 2. $\varphi_{\text{kb}^+}(x)$ does not contain equalities like $x = t$. Since x is the only term occurrence of x in φ_{kb^+} , we can move other literals not containing x out of $(\exists x : \mathbb{T})$, and hence we assume that φ_{kb^+} is in the form

$$\bigwedge_{0 < i \leq l} u_i \preceq_i x \wedge \bigwedge_{0 < j \leq m} x \preceq_{l+j} v_j, \quad (55)$$

where \preceq_i ($0 < i \leq l + m$) are gap orders. By Lemma A.1, φ_{kb^+} is equivalent to a disjunction of delineated gap order completions, each of which contains no more open gap order literals than $\varphi_{\text{kb}^+}(x)$ does. Now we assume $\varphi_{\text{kb}^+}(x)$ is a DGOC, denoted by

$$\text{DGOC}(x, (L_0 x)^w, \dots, (L_n x)^w),$$

where (as in Part 1) $L_0 \equiv x, L_0 x, \dots, L_n x$ enumerate *all* selector terms containing x and form a sibling-complete tree with x being the root, and $(L_0 x)^w, \dots, (L_n x)^w$ enumerate all integer occurrences of x that may occur in $\varphi_{\mathbb{Z}}(x)$ as well as inside boundary terms in $\varphi_{\text{kb}^+}(x)$. Let $\{x\}^w$ abbreviate $(L_0 x)^w, \dots, (L_n x)^w$. Since $\text{DGOC}(x, \{x\}^w)$ is delineated, we assume that $\text{DGOC}(x, \{x\}^w)$ has the form

$$\underbrace{t_1 \preceq_1 \cdots \preceq_{i-2} t_{i-1}}_{\text{Head}(\{x\}^w)} \preceq_{i-1} x \preceq_i \underbrace{t_{i+1} \preceq_{i+1} \cdots \preceq_{n-1} t_n}_{\text{Tail}(\{x\}^w)},$$

where $\preceq_{i-1} \in \{<_{n_1}^\#, \preceq_{n_1}^\#\}$, $\preceq_i \in \{<_{n_2}^\natural, \preceq_{n_2}^\natural\}$, $\# \equiv \natural \in \{l, \text{pl}\}$, and $\text{Head}(\{x\}^w)$ (resp. $\text{Tail}(\{x\}^w)$) denote the linear order before x (resp. after x) in the above sequence. Without loss of generality, we further assume \preceq_{i-1} is $<_{n_1}^l$ and \preceq_i is $<_{n_2}^l$ for $n_1, n_2 > 0$. Accordingly, $\text{DGOC}(x, \{x\}^w)$ has the form

$$\text{Head}(\{x\}^w) <_{n_1}^l x <_{n_2}^l \text{Tail}(\{x\}^w).$$

We write $\text{Head}(\{x\}^w) <_{n_1+n_2}^\# \text{Tail}(\{x\}^w)$ to denote

$$\underbrace{t_1 \preceq_1 \cdots \preceq_{i-2} t_{i-1}}_{\text{Head}(\{x\}^w)} <_{n_1+n_2}^l \underbrace{t_{i+1} \preceq_{i+1} \cdots \preceq_{n-1} t_n}_{\text{Tail}(\{x\}^w)},$$

the ordering obtained from $\text{DGOC}(x, \{x\}^w)$ by removing the *term occurrence* of x and accordingly joining two gap orders before and after such occurrence. We claim that

$$(\exists x : \mathbb{T}) \left[\text{Head}(\{x\}^w) <_{n_1}^l x <_{n_2}^l \text{Tail}(\{x\}^w) \wedge \varphi_{\mathbb{Z}}(\{x\}^w) \right] \quad (56)$$

is equivalent to

$$(\exists \{x\}^w : \mathbb{Z}) \left[\text{Head}(\{x\}^w) <_{n_1+n_2}^l \text{Tail}(\{x\}^w) \wedge \varphi_\Delta(\{x\}^w) \wedge \varphi_{\mathbb{Z}}(\{x\}^w) \right], \quad (57)$$

where $\{x\}^w$ are pseudo integer variables as in Part 1, $\varphi_\Delta(\{x\}^w)$ consists of $x^w = (t_{i-1})^w$, and integer constraints requiring that $L_0 x, \dots, L_n x$ form a legitimate tree

(condition (4a) for $\varphi_{\Delta}(\{x\}^w)$ in Part 1). By variable renaming we rewrite (57) as

$$(\exists \vec{z} : \mathbb{Z}) \left[\text{Head}(\vec{z}) \prec_{n_1+n_2}^1 \text{Tail}(\vec{z}) \wedge \varphi_{\Delta}(\vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{z}) \right], \quad (58)$$

where $\vec{z} = \{z_0, \dots, z_n\}$.

We show equivalence of (56) and (58). It is clear that (56) is equivalent to

$$(\exists x : \mathbb{T})(\exists \vec{z} : \mathbb{Z}) \left[\varphi_{\Delta}(\{x\}^w) \wedge \varphi_{\mathbb{Z}}(\{x\}^w) \wedge \text{Head}(\{x\}^w) \prec_{n_1}^1 x \prec_{n_2}^1 \text{Tail}(\{x\}^w) \wedge \vec{z} = \{x\}^w \right], \quad (59)$$

where $\vec{z} = \{x\}^w$ stands for $\bigwedge_{i=0}^n z_i = (L_i x)^w$. We can rewrite (59) to

$$(\exists \vec{z} : \mathbb{Z})(\exists x : \mathbb{T}) \left[\varphi_{\Delta}(\vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{z}) \wedge \text{Head}(\vec{z}) \prec_{n_1}^1 x \prec_{n_2}^1 \text{Tail}(\vec{z}) \wedge \vec{z} = \{x\}^w \right],$$

which is equivalent to (58) after eliminating x using the fact that weight properties about x and its subterms are completely characterized by $\varphi_{\mathbb{Z}}(\{x\}^w)$. Now by Lemma 4.29 we eliminate $\exists x$ from (58), obtaining $\varphi'_{\text{kb}^+} \wedge \varphi'_{\mathbb{Z}}$ in which x does not occur and φ'_{kb^+} is quantifier-free, as desired.

The elimination procedure needs no essential change for other choices of \preceq_{i-1} and \preceq_i . We only note that if both \preceq_{i-1} and \preceq_i are rigid gap orders, then the order between $\text{Head}(\{x\}^w)$ and $\text{Tail}(\{x\}^w)$ should be rigid too. For example, let \preceq_{i-1} be $\preceq_{n_1}^1$ and \preceq_i be $\preceq_{n_2}^1$. We have

$$(\exists x : \mathbb{T}) \left[\text{Head}(\{x\}^w) \preceq_{n_1}^1 x \preceq_{n_2}^1 \text{Tail}(\{x\}^w) \wedge \varphi_{\mathbb{Z}}(\{x\}^w) \right],$$

which reduces to

$$(\exists \vec{z} : \mathbb{Z}) \left[\text{Head}(\vec{z}) \preceq_{n_1+n_2}^1 \text{Tail}(\vec{z}) \wedge \varphi_{\Delta}(\vec{z}) \wedge \varphi_{\mathbb{Z}}(\vec{z}) \right].$$

Finally note that we carry out the elimination even if x has other integer occurrences with depths greater than 0. We could have strengthened the precondition that requires that *all* occurrences of x has depth 0. To satisfy this precondition, however, the simplification procedure, step (1c) in Algorithm 5.2 has to include a sub-procedure for decomposing literals like $x = t$ using proper selector terms containing x . We choose to hide this complexity in the proof rather than exposing it in the algorithm. \square

Lemma 4.29 (Elimination of Integer Quantifiers). Let z be an integer variable, $\varphi_{\text{kb}^+}(z)$ a conjunction of literals in $\mathcal{L}_{\text{kb}^+}$ where z occurs inside boundary terms, and $\varphi_{\mathbb{Z}}(z)$ a Presburger formula. Then

$$(\exists z : \mathbb{Z}) \left[\varphi_{\text{kb}^+}(z) \wedge \varphi_{\mathbb{Z}}(z) \right]$$

can be effectively reduced to $\varphi'_{\text{kb}^+} \wedge \varphi'_{\mathbb{Z}}$ where no z occurs and φ'_{kb^+} is quantifier-free.

PROOF. Let z , $\varphi_{\text{kb}^+}(z)$ and $\varphi_{\mathbb{Z}}(z)$ be as stated above. Let us assume z has n occurrences in $\varphi_{\text{kb}^+}(z)$, namely in n maximum integer functions enumerated below

$$f_1(z), \quad \dots, \quad f_n(z),$$

where f_i ($0 < i \leq n$) are arbitrary integer functions definable in Presburger arithmetic, and $f_i(z)$ ($0 < i \leq n$) occur inside boundary terms. Note that f_i 's are introduced during reductions in Section C, and they are constructed using those integer functions defined in Section B.

For each $0 < i \leq n$, we proceed as follows. Without loss of generality, we assume $f_i(z)$ *properly* occurs inside a boundary term t , that is, t has one of the following forms:

$$0_{(f_i(z))}^w, \quad 0_{(f_i(z), p)}^p, \quad 1_{(f_i(z))}^w, \quad 1_{(f_i(z), p)}^p.$$

We do case analysis according to the occurrences of t in $\varphi_{\text{kb}^+}(z)$.

(1) t occurs in literals of the forms

$$t <_n^\# t', \quad t \leq_n^\# t', \quad t' <_n^\# t, \quad t' \leq_n^\# t, \quad t = t',$$

where $n > 0$, $\# \in \{w, p, l\}$ and t' is also a boundary term. By Lemma 4.26, we can remove all such closed literals from $\varphi_{\text{kb}^+}(z)$ and add the corresponding equivalent Presburger formulas to $\varphi_{\mathbb{Z}}(z)$.

(2) t occurs in literals of the forms

$$t <_n^* t', \quad t \leq_n^* t', \quad t' <_n^* t, \quad t' \leq_n^* t, \quad t = t',$$

where $n > 0$, $* \in \{p, l\}$ and t' is an ordinary term. Then we replace every occurrence of $f_i(z)$ in $\varphi_{\text{kb}^+}(z)$ by $(t')^w$ and add $f_i(z) = (t')^w$ to $\varphi_{\mathbb{Z}}(z)$. Formally, we set

$$\begin{aligned} \varphi_{\text{kb}^+}(z) &:= \varphi_{\text{kb}^+}(z) \left[f_i(z)/(t')^w \right], \\ \varphi_{\mathbb{Z}}(z) &:= \varphi_{\mathbb{Z}}(z) \wedge f_i(z) = (t')^w. \end{aligned}$$

(3) t occurs in literals of the forms

$$t <_n^w t', \quad t \leq_n^w t', \quad t' <_n^w t, \quad t' \leq_n^w t,$$

where $n > 0$ and t' is an ordinary term. In this case, we introduce new boundary terms to delineate those gap orders. For example, $t <_n^w t'$ is replaced by

$$\bigvee_{n_1+n_2=n} t <_{n_1}^w 0_{((t')^w)}^w <_{n_2}^{pl} t'.$$

Similar transformations apply to $t \leq_n^w t'$, $t' <_n^w t$ and $t' \leq_n^w t$. Then we reduce this case to case 1 and to case 2.

In this way we can remove all $f_i(z)$ from φ_{kb^+} , obtaining

$$(\exists z : \mathbb{Z}) \left[\varphi'_{\text{kb}^+} \wedge \varphi'_{\mathbb{Z}}(z) \right],$$

where φ'_{kb^+} does not contain z . So we can move φ'_{kb^+} out of $\exists z$, obtaining

$$\varphi'_{\text{kb}^+} \wedge (\exists z : \mathbb{Z}) \varphi'_{\mathbb{Z}}(z),$$

as desired. \square

Lemma 5.4 (Termination). Both Algorithm 5.1 and Algorithm 5.2 terminate.

PROOF. The termination of Algorithm 5.1 is straightforward. The termination of Algorithm 5.2 needs a careful examination. The subtlety comes from step (1). It is clear that each run of step (1) terminates as the decomposition (step (1b)) is done in depth-first manner, which eventually produces a variable whose term occurrences are all of depth 0. However, this step may introduce more existentially quantified variables, gap order literals and equality literals and it may also increase the depth of variables that are not the target of the reduction. Consider in $\text{List}_{\text{kb}^+}^Z$. The formula

$$(\exists x) \left[x <_{n_1}^l t \wedge \text{car}(x) <_{n-2}^l t \right],$$

where $\text{depth}(x) = 1$ and $\text{depth}(t) = 0$, reduces to (one of the choices)

$$(\exists x_1)(\exists x_2) \left[x_1 = \text{car}(t) \wedge x_2 <_{n_1}^l \text{cdr}(t) \wedge x_1 <_{n-2}^l t \right],$$

where $\text{depth}(x_1) = \text{depth}(x_2) = 0$ but $\text{depth}(t) = 1$. Here we have one more existentially quantified variable and a new equality literal, and the depth of t is increased by 1. Reduction (201) shows a case where the number of gap order literals increases.

A careful examination of the reductions in Section C reveals that in all transformations the number of open gap order literals is never increasing, which is crucial in this termination proof. We proceed by first establishing five claims.

- (1) The number of open gap order literals in each reduct is no more than that in the corresponding redex.
- (2) The total number of occurrences of newly introduced variables in the reduct is no more than the number of occurrences of the decomposed variable in the redex.
- (3) Without reductions (203) and (206), the sum of the depths of variables strictly decreases in each run of step (1).
- (4) Reductions (203) and (206) are only triggered by step (1) targeting variables occurring in open gap order literals.
- (5) A run of step (1) targeting a variable x that occurs in an open gap order literal produces four types of outcomes.
 - (a) The sum of the depths of variables in the reduct decreases.
 - (b) The reduct contains fewer open gap order literals than the redex.
 - (c) The reduct contains a newly introduced variable x' that occurs in an open gap order literal and is the target for the next run of step (1).
 - (d) The reduct contains a newly introduced variable x' that occurs in an open gap order literal and is the target for the next run of step (2).

Proof of Claim (1). Recall that this statement is included in every lemma regarding equivalent transformation. Here we list the transformation steps and their corresponding justifications.

Depth Reduction	Lemma 4.26
Elimination of Negative Literals	Lemma 4.21
Equality Elimination	Lemma 4.28, Part 1
Term Elimination	Lemma 4.28, Part 2
Delineated Gap Order Completion	Lemma A.1

Proof of Claim (2). In all tuple reductions in Sections C.8-C.15, if $\langle u_1, \dots, u_k \rangle$ appears in a redex, then each u_i appears at most once in each corresponding reduct. The claim is then proved by induction.

Proof of Claim (3). It can be shown by checking reductions in Sections C.6, C.7, C.14 and C.15 that (i) the depth of a term can only increase in reductions (345) and (348) which are triggered by reductions (203) and (206), respectively, or in a substitution due to eliminating x appearing in an open equality literal of the form $x = t$ (Part 1 of the proof of Lemma 4.28); (ii) an open equality literal is only generated in reductions (345) and (348). As a consequence, reductions (203) and (206) are the only sources of increasing depths of variables. In each run of step (1), the depths of newly introduced variables are less than the depth of the decomposed variable in the redex. By claim (2), without reductions (203) and (206), the sum of the depths of variables must be strictly decreasing in each run of step (1).

Proof of Claim (4). A check of the reductions in Section C justifies this claim.

Proof of Claim (5). There are two main cases.

- (1) All occurrences of the targeted variable x have depth greater than 0.

Then step (1) only involves simplification of selector terms (step (1(c)i)) and simplification of integer terms (step (1(c)ii)). The simplification of integer terms clearly does not affect the depths of variables. For the simplification of selector terms, let us consider $LS_i^\alpha x \star L'S_i^{\alpha'} x'$ where $\text{type}(x) = \alpha$, $\text{type}(x') = \alpha'$. If $x \equiv x'$, then $\alpha \equiv \alpha'$, and this literal simplifies to $Lx_i \star L'x'_i$. If $x \not\equiv x'$, then this literal simplifies to $Lx_i \star L'S_i^{\alpha'} x'$. In both cases, $\text{depth}(x_i)$ is less than $\text{depth}(x)$, and the depths of other variables do not increase. Therefore, the sum of the depths of variables decreases. The case for $L'S_i^{\alpha'} x' \star LS_i^\alpha x$ is similar.

- (2) $\text{depth}(x) > 0$ and $x \star t$ (or $t \star x$) occurs.

Then step (1) also involves reductions of gap order literals (step (1(c)iii)). If neither reduction (203) nor reduction (206) is used, then x and t are separated by boundary terms, and hence at least one open gap order literal is removed with no new one generated. If either reduction (203) or reduction (206) is used, then \star must be $<_n^l$ or \leq_n^l . Let us consider $x <_n^l t$ (the cases for $t <_n^l x$, $x \leq_n^l t$ and $t \leq_n^l x$ are similar). Reduction (203) triggers reduction (345), which produces a reduct containing the following conjuncts

$$x_1 = S_1^\alpha t, \quad \dots, \quad x_{i-1} = S_{i-1}^\alpha t, \quad x_i <_{n'}^\# S_i^\alpha t,$$

where $<^\# \in \{w, p, l\}$ and $n' \leq n$. All newly introduced variables x_1, \dots, x_{i-1} will be removed by substitution directly. If $\text{depth}(x_i) > 0$ in the reduct, then x_i becomes the target in the next run of step (1). Otherwise, step (2) is activated with x_i being the elimination target.

Now suppose that Algorithm 5.2 does not terminate. The run must visit step (1) infinitely many times, because only step (1) can generate more existentially quantified variables. We claim that either reduction (203) or reduction (206) runs infinitely often. Suppose otherwise, then from some point on, there are no runs of reductions (203) and (206). Then by claim (3), a run of step (1) always reduces the sum

of the depths of variables. Eventually the depths of all variables are 0 and step (1) is disabled and the run of Algorithm 5.2 stays in step (2) forever, a contradiction.

Now we assume that either reduction (203) or reduction (206) occurs infinitely often. It follows that there must be infinitely many runs of step (1) targeting variables occurring in open gap order literals, because by claim (4) only such runs can trigger reduction (203) or reduction (206). However, this cannot happen even if there are infinitely many such runs of step (1). We analyze all types of outcomes of step (1) targeting variables occurring in open gap order literals. Outcome (5b) clearly does not happen infinitely many times because at least one open gap order literal is removed each time. Outcome (5d) cannot happen infinitely often either, because (thanks to Lemma A.1) each occurrence triggers step (2) that reduces the number of open gap order literals at least by one. Thanks to variable selection (step (1a)) in depth-first manner, outcome (5c) cannot happen continuously infinitely many times as the depths of variables are finite. If outcome (5c) happens, then the next outcome is either (5c) again or (5d). So if outcome (5c) happens infinitely many times, then so does outcome (5d), a contradiction. The only possibility left is that outcome (5a) happens infinitely often. And in fact it will be continuously happening from some point on. This means that from that point on the sum of the depths of variables strictly decreases, and hence eventually step (1) is disabled forever, contradicting our initial assumption.

We have exhausted all cases and all of them lead to contradictions. Hence Algorithm 5.2 terminates. \square

B. DEFINITIONS

In this section, we show that the integer functions and predicates used in the proofs in Appendix A and in the reductions in Appendix C can be defined in Presburger arithmetic.

B.1 Definition of Integer Predicates

- (1) $\widehat{\text{CNT}}_{k,n}^\alpha(x)$ ($k > 0, n \geq 0$): there are *exactly* $n+1$ different α -terms of length x in TA with $|\mathcal{A}| = k$.

$$\widehat{\text{CNT}}_{k,n}^\alpha(x) \stackrel{\text{def}}{=} \text{CNT}_{k,n}^\alpha(x) \wedge \neg \text{CNT}_{k,n+1}^\alpha(x) .$$

$\widehat{\text{CNT}}_{k,n}(x)$ is similarly defined with α -terms replaced by TA-terms.

$$\widehat{\text{CNT}}_{k,n}(x) \stackrel{\text{def}}{=} \text{CNT}_{k,n}(x) \wedge \neg \text{CNT}_{k,n+1}(x) .$$

- (2) $\text{IsTW}^k(n)$: n is the weight of a k -tuple.

$$\text{IsTW}^k(n) \stackrel{\text{def}}{=} (\exists z_1, \dots, \exists z_k > 0) \left(\sum_{i=1}^k z_i = n \wedge \bigwedge_{0 < i \leq k} \text{Tree}(z_i) \right) .$$

- (3) $\text{IsCW}^k(n, m)$: m is the weight of a component in a k -tuple of weight n .

$$\text{IsCW}^k(n, m) \stackrel{\text{def}}{=} \text{IsTW}^k(n) \wedge \left((k > 1 \wedge \text{IsTW}^{k-1}(n-m)) \vee (k = 1 \wedge m = n) \right) .$$

- (4) $\text{IsTW}^k(n, p)$: n is the weight of a k -tuple whose first component is of type α_p .

$$\text{IsTW}^k(n, p) \stackrel{\text{def}}{=} (\exists z_1, \dots, \exists z_k > 0) \left(\sum_{i=1}^k z_i = n \wedge \bigwedge_{0 < i \leq k} \text{Tree}(z_i) \wedge \text{Tree}^{\alpha_p}(z_1) \right) .$$

- (5) $\text{IsCW}^k(n, m, p)$: n is the weight of a k -tuple whose first component has weight m and type α_p .

$$\text{IsCW}^k(n, m, p) \stackrel{\text{def}}{=} \text{IsCW}^k(n, m) \wedge \text{Tree}^{\alpha_p}(m) .$$

- (6) $\text{WD}_\Omega(\Delta)$: $\Omega(\Delta)$ is a well-defined boundary term or boundary tuple.

$$\begin{aligned} \text{WD}_{0^w}(m) &\stackrel{\text{def}}{=} \text{Tree}(m) , & \text{WD}_{1^w}(m) &\stackrel{\text{def}}{=} \text{Tree}(m) , \\ \text{WD}_{0^p}(m, p) &\stackrel{\text{def}}{=} \text{Tree}^{\alpha_p}(m) , & \text{WD}_{1^p}(m, p) &\stackrel{\text{def}}{=} \text{Tree}^{\alpha_p}(m) , \\ \text{WD}_{\bar{0}^{kkb}}(n) &\stackrel{\text{def}}{=} \text{IsTW}^k(n) , & \text{WD}_{\bar{1}^{kkb}}(n) &\stackrel{\text{def}}{=} \text{IsTW}^k(n) , \\ \text{WD}_{\bar{0}^{kw}}(n, m) &\stackrel{\text{def}}{=} \text{IsCW}^k(n, m) , & \text{WD}_{\bar{1}^{kw}}(n, m) &\stackrel{\text{def}}{=} \text{IsCW}^k(n, m) , \\ \text{WD}_{\bar{0}^{kp}}(n, m, p) &\stackrel{\text{def}}{=} \text{IsCW}^k(n, m, p) , & \text{WD}_{\bar{1}^{kp}}(n, m, p) &\stackrel{\text{def}}{=} \text{IsCW}^k(n, m, p) . \end{aligned}$$

B.2 Definition of Integer Functions

- (1) $x = \text{MinCW}^k(n, r, r')$: x is the smallest integer in (r, r') that is the weight of a component in a k -tuple of weight n ; $x = r$ if there is no such integer.

$$\begin{aligned} &\left[(\neg \text{IsTW}^k(n) \vee \forall z(r < z < r' \rightarrow \neg \text{IsCW}^k(n, z)) \right] \wedge x = r \Big] \vee \\ &\left[\text{IsCW}^k(n, x) \wedge r < x < r' \wedge \forall z(r < z < x \rightarrow \neg \text{IsCW}^k(n, z)) \right] . \end{aligned} \quad (60)$$

- (2) $x = \text{MaxCW}^k(n, r, r')$: x is the largest integer in (r, r') that is the weight of a component in a k -tuple of weight n ; $x = r$ if there is no such integer.

$$\begin{aligned} &\left[(\neg \text{IsTW}^k(n) \vee \forall z(r < z < r' \rightarrow \neg \text{IsCW}^k(n, z)) \right] \wedge x = r \Big] \vee \\ &\left[\text{IsCW}^k(n, x) \wedge r < x < r' \wedge \forall z(x < z < r' \rightarrow \neg \text{IsCW}^k(n, z)) \right] . \end{aligned} \quad (61)$$

- (3) $x = \text{CW}_i^k(n, r, r')$ ($i > 0$): x is the i^{th} smallest integer in (r, r') that is the weight of a component in a k -tuple of weight n ; $x = \text{MaxCW}^k(n, r, r')$ if no such integer exists. We give the definition by induction on i .

Case $i = 1$.

$$x = \text{MinCW}^k(n, r, r').$$

Case $i > 1$.

$$\begin{aligned} &\left[x > \text{CW}_{i-1}^k(n, r, r') \wedge r < x < r' \wedge \text{IsCW}^k(n, x) \wedge \right. \\ &\quad \left. \forall z(\text{CW}_{i-1}^k(n, r, r') < z < x \rightarrow \neg \text{IsCW}^k(n, z)) \right] \vee \\ &\left[x = \text{CW}_{i-1}^k(n, r, r') \right. \\ &\quad \left. \wedge \forall z((z > \text{CW}_{i-1}^k(n, r, r') \wedge r < z < r') \rightarrow \neg \text{IsCW}^k(n, z)) \right] . \end{aligned} \quad (62)$$

With no risk of confusion, we use the following abbreviations.

$$\begin{aligned} \text{MinCW}^k(n) &\text{ abbr. } \text{MinCW}^k(n, 0, n+1) , \\ \text{MaxCW}^k(n) &\text{ abbr. } \text{MaxCW}^k(n, 0, n+1) , \\ \text{CW}_i^k(n) &\text{ abbr. } \text{CW}_i^k(n, 0, n+1) . \end{aligned}$$

- (4) $x = \text{CWS}_i^k(n)$ ($0 < i \leq k$) : x is the weight of the i^{th} component of the smallest k -tuple (w.r.t. $\prec^{k, \text{kb}}$) of weight n ; $x = 0$ if no such weight exists. We define it inductively as follows.

Case $i = 1$.

$$x = \text{MinCW}^k(n) .$$

Case $1 < i \leq k$.

$$\left(\neg \text{IsTW}^k(n) \rightarrow x = 0 \right) \wedge \left(\text{IsTW}^k(n) \rightarrow x = \text{MinCW}^{k-i+1} \left(n - \sum_{j=1}^{i-1} \text{CWS}_j^k(n) \right) \right) .$$

- (5) $x = \text{CWL}_i^k(n)$ ($0 < i \leq k$) : x is the weight of the i^{th} component of the largest k -tuple (w.r.t. $\prec^{k, \text{kb}}$) of weight n ; $x = 0$ if no such weight exists. We define it inductively as follows.

Case $i = 1$.

$$x = \text{MaxCW}^k(n) .$$

Case $1 < i \leq k$.

$$\left(\neg \text{IsTW}^k(n) \rightarrow x = 0 \right) \wedge \left(\text{IsTW}^k(n) \rightarrow x = \text{MaxCW}^{k-i+1} \left(n - \sum_{j=1}^{i-1} \text{CWS}_j^k(n) \right) \right) .$$

C. REDUCTIONS

In this section we present all reductions that can occur in Algorithm 5.2. The following outlines the contents in each subsection.

C.1	Closed Term Equalities	C.8	Closed Tuple Equalities
—	(No Half-open Term Equalities)	C.9	Half-open Tuple Equalities
C.2	Closed Stretchable Gap Orders	C.10	Closed Stretchable Tuple Gap Orders
C.3	Closed Rigid Gap Orders	C.11	Closed Rigid Tuple Gap Orders
C.4	Half-open Stretchable Gap Orders	C.12	Half-open Stretchable Tuple Gap Orders
C.5	Half-open Rigid Gap Orders	C.13	Half-open Rigid Tuple Gap Orders
C.6	Open Stretchable Gap Orders	C.14	Open Stretchable Tuple Gap Orders
C.7	Open Rigid Gap Orders	C.15	Open Rigid Tuple Gap Orders

We assume that all gap order counts in redexes are positive numbers. Let $\vec{u} = \langle u_1, \dots, u_k \rangle$, $\vec{v} = \langle v_1, \dots, v_k \rangle$, $u = \alpha_q(u_1, \dots, u_k)$, and $v = \alpha_q(v_1, \dots, v_k)$.

C.1 Reduction of Closed Equalities

$$0_{(m)}^w = 0_{(m')}^w \mapsto m = m' \tag{63}$$

$$0_{(m)}^w = 0_{(m', p')}^p \mapsto m = m' \wedge \bigwedge_{p' < i \leq |\Sigma|} \neg \text{Tree}^{(ai)}(m) \tag{64}$$

$$0_{(m)}^w = 1_{(m')}^w \mapsto m = m' \wedge \neg \text{CNT}_1(m) \quad (65)$$

$$0_{(m)}^w = 1_{(m',p')}^p \mapsto m = m' \wedge \bigwedge_{p' < i \leq |\Sigma|} \neg \text{Tree}^{(\alpha_i)}(m) \wedge \neg \text{CNT}_1^{(\alpha_{p'})}(m) \quad (66)$$

$$0_{(m,p)}^p = 0_{(m',p')}^p \mapsto m = m' \wedge p = p' \quad (67)$$

$$0_{(m,p)}^p = 1_{(m')}^w \mapsto m = m' \wedge \bigwedge_{0 < i < p} \neg \text{Tree}^{(\alpha_i)}(m) \wedge \neg \text{CNT}_1^{(\alpha_p)}(m) \quad (68)$$

$$0_{(m,p)}^p = 1_{(m',p')}^p \mapsto m = m' \wedge p = p' \wedge \neg \text{CNT}_1^{(\alpha_p)}(m) \quad (69)$$

$$1_{(m)}^w = 1_{(m')}^w \mapsto m = m' \quad (70)$$

$$1_{(m)}^w = 1_{(m',p')}^p \mapsto m = m' \wedge \bigwedge_{0 < i < p'} \neg \text{Tree}^{(\alpha_i)}(m) \quad (71)$$

$$1_{(m,p)}^p = 1_{(m',p')}^p \mapsto m = m' \wedge p = p' \quad (72)$$

C.2 Reduction of Closed Stretchable Gap Orders

$$0_{(m)}^w <_n^w 0_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} 0_{(m)}^w <_{n_1}^{p_1} 1_{(m)}^w <_{n_2}^w 0_{(m')}^w \quad (73)$$

$$0_{(m)}^w <_n^p 0_{(m')}^w \mapsto \text{false} \quad (74)$$

$$0_{(m)}^w <_n^l 0_{(m')}^w \mapsto \text{false} \quad (75)$$

$$0_{(m)}^w <_n^w 0_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} 0_{(m)}^w <_{n_1}^{p_1} 1_{(m)}^w <_{n_2}^w 0_{(m')}^w <_{n_3}^w 0_{(m',p')}^p \quad (76)$$

$$0_{(m)}^w <_n^p 0_{(m',p')}^p \mapsto m = m' \wedge \bigvee_{n_{p'+1}+\dots+n_{|\Sigma|}=n} \bigwedge_{p' < i \leq |\Sigma|} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \quad (77)$$

$$0_{(m)}^w <_n^l 0_{(m',p')}^p \mapsto \text{false} \quad (78)$$

$$0_{(m)}^w <_n^w 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} 0_{(m)}^w <_{n_1}^{p_1} 1_{(m)}^w <_{n_2}^w 0_{(m')}^w <_{n_3}^{p_1} 1_{(m')}^w \quad (79)$$

$$0_{(m)}^w <_n^p 1_{(m')}^w \mapsto \left(m = m' \wedge \bigvee_{\substack{p,p' \in \{1,\dots,|\Sigma|\} \\ p \neq p'}} (\text{Tree}^{(\alpha_p)}(m) \wedge \text{Tree}^{(\alpha_{p'})}(m)) \right) \wedge \bigvee_{n_1+\dots+n_{|\Sigma|}=n+1} \bigwedge_{0 < i \leq |\Sigma|} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \quad (80)$$

$$0_{(m)}^w \prec_n^1 1_{(m')}^w \mapsto \left(\begin{array}{l} m = m' \wedge \bigwedge_{\substack{p,p' \in \{1, \dots, |\Sigma|\} \\ p \neq p'}} (\text{Tree}^{(\alpha p)}(m) \rightarrow \neg \text{Tree}^{(\alpha p')}(m)) \\ \wedge \bigvee_{0 < i \leq |\Sigma|} \text{CNT}_n^{(\alpha_i)}(m) \end{array} \right) \quad (81)$$

$$0_{(m)}^w \prec_n^w 1_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} 0_{(m)}^w \prec_{n_1}^{p_1} 1_{(m)}^w \prec_{n_2}^w 0_{(m')}^w \prec_{n_3}^{p_1} 0_{(m',p')}^w \prec_{n_4}^{p_1} 1_{(m',p')}^w \quad (82)$$

$$0_{(m)}^w \prec_n^p 1_{(m',p')}^p \mapsto \left(\begin{array}{l} m = m' \wedge \bigvee_{p' < p \leq |\Sigma|} \text{Tree}^{(\alpha p)}(m) \\ \wedge \bigvee_{n_{p'} + \dots + n_{|\Sigma|} = n+1} \bigwedge_{p' \leq i \leq |\Sigma|} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (83)$$

$$0_{(m)}^w \prec_n^1 1_{(m',p')}^p \mapsto m = m' \wedge \bigwedge_{p' < p \leq |\Sigma|} \neg \text{Tree}^{(\alpha p)}(m) \wedge \text{CNT}_n^{(\alpha_{p'})}(m) \quad (84)$$

$$0_{(m,p)}^p \prec_n^w 0_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} 0_{(m,p)}^p \prec_{n_1}^{p_1} 1_{(m)}^w \prec_{n_2}^w 0_{(m')}^w \quad (85)$$

$$0_{(m,p)}^p \prec_n^p 0_{(m')}^w \mapsto \text{false} \quad (86)$$

$$0_{(m,p)}^p \prec_n^1 0_{(m')}^w \mapsto \text{false} \quad (87)$$

$$0_{(m,p)}^p \prec_n^w 0_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} 0_{(m,p)}^p \prec_{n_1}^{p_1} 1_{(m,p)}^p \prec_{n_2}^{p_1} 1_{(m)}^w \prec_{n_3}^w 0_{(m')}^w \prec_{n_4}^{p_1} 0_{(m',p')}^p \quad (88)$$

$$0_{(m,p)}^p \prec_n^p 0_{(m',p')}^p \mapsto \left(\begin{array}{l} m = m' \wedge p' < p \\ \wedge \\ \bigvee_{n_{p'} + \dots + n_{p-1} = n} \bigwedge_{p' \leq i < p} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (89)$$

$$0_{(m,p)}^p \prec_n^1 0_{(m',p')}^p \mapsto \text{false} \quad (90)$$

$$0_{(m,p)}^p \prec_n^w 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} 0_{(m,p)}^p \prec_{n_1}^{p_1} 1_{(m,p)}^p \prec_{n_2}^{p_1} 1_{(m)}^w \prec_{n_3}^w 0_{(m')}^w \prec_{n_4}^{p_1} 1_{(m')}^w \quad (91)$$

$$0_{(m,p)}^p \prec_n^p 1_{(m')}^w \mapsto \left(\begin{array}{l} m = m' \wedge \bigvee_{0 < i < p} \text{Tree}^{(\alpha_i)}(m) \\ \wedge \bigvee_{n_1 + \dots + n_p = n+1} \bigwedge_{0 < i \leq p} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (92)$$

$$0_{(m,p)}^p \prec_n^1 1_{(m')}^w \mapsto m = m' \wedge \bigwedge_{0 < i < p} \neg \text{Tree}^{(\alpha_i)}(m) \wedge \text{CNT}_n^{(\alpha_p)}(m) \quad (93)$$

$$0_{(m,p)}^p <_n^w 1_{(m',p')}^p \mapsto \bigvee_{\substack{\sum_{i=1}^5 n_i = n \\ n_3 > 0}} 0_{(m,p)}^p <_{n_1}^l 1_{(m,p)}^p <_{n_2}^{pl} 1_{(m)}^w <_{n_3}^w 0_{(m')}^w <_{n_4}^{pl} 0_{(m',p')}^p <_{n_5}^l 1_{(m',p')}^p \quad (94)$$

$$0_{(m,p)}^p <_n^p 1_{(m',p')}^p \mapsto \left(\begin{array}{c} m = m' \wedge p' < p \\ \wedge \\ \bigvee_{n_{p'} + \dots + n_p = n+1} \bigwedge_{p' \leq i \leq p} (n_i > 0 \rightarrow \text{CNT}_{n_{i-1}}^{(\alpha_i)}(m)) \end{array} \right) \quad (95)$$

$$0_{(m,p)}^p <_n^l 1_{(m',p')}^p \mapsto m = m' \wedge p = p' \wedge \text{CNT}_n^{(\alpha_p)}(m) \quad (96)$$

$$1_{(m)}^w <_n^w 0_{(m')}^w \mapsto \left(\begin{array}{c} (n = 1 \rightarrow m < m') \\ \wedge \\ n > 1 \rightarrow \bigvee_{0 < r < n} (\exists z_1 \dots \exists z_r) \left(\begin{array}{c} m < z_1 < \dots < z_r < m' \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_1, \dots, n_r > 0}} \bigwedge_{0 < i \leq r} \text{CNT}_{n_{i-1}}(z_i) \end{array} \right) \end{array} \right) \quad (97)$$

$$1_{(m)}^w <_n^p 0_{(m')}^w \mapsto \text{false} \quad (98)$$

$$1_{(m)}^w <_n^l 0_{(m')}^w \mapsto \text{false} \quad (99)$$

$$1_{(m)}^w <_n^w 0_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1 > 0}} 1_{(m)}^w <_{n_1}^w 0_{(m')}^w <_{n_2}^{pl} 0_{(m',p')}^p \quad (100)$$

$$1_{(m)}^w <_n^p 0_{(m',p')}^p \mapsto \text{false} \quad (101)$$

$$1_{(m)}^w <_n^l 0_{(m',p')}^p \mapsto \text{false} \quad (102)$$

$$1_{(m)}^w <_n^w 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1 > 0}} 1_{(m)}^w <_{n_1}^w 0_{(m')}^w <_{n_2}^{pl} 1_{(m')}^w \quad (103)$$

$$1_{(m)}^w <_n^p 1_{(m')}^w \mapsto \text{false} \quad (104)$$

$$1_{(m)}^w <_n^l 1_{(m')}^w \mapsto \text{false} \quad (105)$$

$$1_{(m)}^w <_n^w 1_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_1 > 0}} 1_{(m)}^w <_{n_1}^w 0_{(m')}^w <_{n_2}^{pl} 0_{(m',p')}^p <_{n_3}^{pl} 1_{(m',p')}^p \quad (106)$$

$$1_{(m)}^w <_n^p 1_{(m',p')}^p \mapsto \text{false} \quad (107)$$

$$1_{(m)}^w <_n^l 1_{(m',p')}^p \mapsto \text{false} \quad (108)$$

$$1_{(m,p)}^p <_n^w 0_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2 > 0}} 1_{(m,p)}^p <_{n_1}^{pl} 1_{(m)}^w <_{n_2}^w 0_{(m')}^w \quad (109)$$

$$1_{(m,p)}^p <_n^p 0_{(m')}^w \mapsto \text{false} \quad (110)$$

$$1_{(m,p)}^p \prec_n^l 0_{(m')}^w \mapsto \text{false} \quad (111)$$

$$1_{(m,p)}^p \prec_n^w 0_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} 1_{(m,p)}^p \prec_{n_1}^{pl} 1_{(m)}^w \prec_{n_2}^w 0_{(m')}^w \prec_{n_3}^w 0_{(m',p')}^p \quad (112)$$

$$1_{(m,p)}^p \prec_n^p 0_{(m',p')}^p \mapsto \left(\begin{array}{c} n = 1 \rightarrow (m = m' \wedge p' < p) \\ \wedge \\ m = m' \wedge p' < p - 1 \\ \wedge \\ n > 1 \rightarrow \left(\bigvee_{n_{p'+1}+\dots+n_{p-1}=n-1} \bigwedge_{p'<i<p} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \right) \end{array} \right) \quad (113)$$

$$1_{(m,p)}^p \prec_n^l 0_{(m',p')}^p \mapsto \text{false} \quad (114)$$

$$1_{(m,p)}^p \prec_n^w 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} 1_{(m,p)}^p \prec_{n_1}^{pl} 1_{(m)}^w \prec_{n_2}^w 0_{(m')}^w \prec_{n_3}^{pl} 1_{(m')}^w \quad (115)$$

$$1_{(m,p)}^p \prec_n^p 1_{(m')}^w \mapsto m = m' \wedge \bigvee_{n_1+\dots+n_{p-1}=n} \bigwedge_{0<i<p} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \quad (116)$$

$$1_{(m,p)}^p \prec_n^l 1_{(m')}^w \mapsto \text{false} \quad (117)$$

$$1_{(m,p)}^p \prec_n^w 1_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} 1_{(m,p)}^p \prec_{n_1}^{pl} 1_{(m)}^w \prec_{n_2}^w 0_{(m')}^w \prec_{n_3}^{pl} 0_{(m',p')}^w \prec_{n_4}^{pl} 1_{(m',p')}^w \quad (118)$$

$$1_{(m,p)}^p \prec_n^p 1_{(m',p')}^p \mapsto \left(\begin{array}{c} m = m' \wedge p' < p \\ \wedge \\ \bigvee_{n_{p'+1}+\dots+n_{p-1}=n} \bigwedge_{p' \leq i < p} (n_i > 0 \rightarrow \text{CNT}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (119)$$

$$1_{(m,p)}^p \prec_n^l 1_{(m',p')}^p \mapsto \text{false} \quad (120)$$

C.3 Reduction of Closed Rigid Gap Orders

$$0_{(m)}^w \cong_n^w 0_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} 0_{(m)}^w \cong_{n_1}^{pl} 1_{(m)}^w \cong_{n_2}^w 0_{(m')}^w \quad (121)$$

$$0_{(m)}^w \cong_n^p 0_{(m')}^w \mapsto \text{false} \quad (122)$$

$$0_{(m)}^w \cong_n^l 0_{(m')}^w \mapsto \text{false} \quad (123)$$

$$0_{(m)}^w \cong_n^w 0_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} 0_{(m)}^w \cong_{n_1}^{pl} 1_{(m)}^w \cong_{n_2}^w 0_{(m')}^w \cong_{n_3}^w 0_{(m',p')}^p \quad (124)$$

$$O_{(m)}^w \stackrel{\leq_n^p}{=} O_{(m',p')}^p \mapsto m = m' \wedge \bigvee_{n_{p'+1}+\dots+n_{|\Sigma|}=n} \bigwedge_{p' \leq i \leq |\Sigma|} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{(\alpha_i)}(m)) \quad (125)$$

$$O_{(m)}^w \stackrel{\leq_n^1}{=} O_{(m',p')}^p \mapsto \text{false} \quad (126)$$

$$O_{(m)}^w \stackrel{\leq_n^w}{=} 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} O_{(m)}^w \stackrel{\leq_{n_1}^{p_1}}{=} 1_{(m)}^w \stackrel{\leq_{n_2}^w}{=} O_{(m')}^w \stackrel{\leq_{n_3}^{p_1}}{=} 1_{(m')}^w \quad (127)$$

$$O_{(m)}^w \stackrel{\leq_n^p}{=} 1_{(m')}^w \mapsto \left(\begin{array}{l} m = m' \wedge \bigvee_{\substack{p,p' \in \{1,\dots,|\Sigma|\} \\ p \neq p'}} (\text{Tree}^{(\alpha_p)}(m) \wedge \text{Tree}^{(\alpha_{p'})}(m)) \\ \wedge \bigvee_{n_1+\dots+n_{|\Sigma|}=n+1} \bigwedge_{0 < i \leq |\Sigma|} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (128)$$

$$O_{(m)}^w \stackrel{\leq_n^1}{=} 1_{(m')}^w \mapsto \left(\begin{array}{l} m = m' \wedge \bigwedge_{\substack{p,p' \in \{1,\dots,|\Sigma|\} \\ p \neq p'}} (\text{Tree}^{(\alpha_p)}(m) \rightarrow \neg \text{Tree}^{(\alpha_{p'})}(m)) \\ \wedge \bigvee_{0 < i \leq |\Sigma|} \widehat{\text{CNT}}_n^{(\alpha_i)}(m) \end{array} \right) \quad (129)$$

$$O_{(m)}^w \stackrel{\leq_n^w}{=} 1_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} O_{(m)}^w \stackrel{\leq_{n_1}^{p_1}}{=} 1_{(m)}^w \stackrel{\leq_{n_2}^w}{=} O_{(m')}^w \stackrel{\leq_{n_3}^{p_1}}{=} O_{(m',p')}^w \stackrel{\leq_{n_4}^{p_1}}{=} 1_{(m',p')}^w \quad (130)$$

$$O_{(m)}^w \stackrel{\leq_n^p}{=} 1_{(m',p')}^p \mapsto \left(\begin{array}{l} m = m' \wedge \bigvee_{p' < p \leq |\Sigma|} \text{Tree}^{(\alpha_p)}(m) \\ \wedge \bigvee_{n_{p'}+\dots+n_{|\Sigma|}=n+1} \bigwedge_{p' \leq i \leq |\Sigma|} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (131)$$

$$O_{(m)}^w \stackrel{\leq_n^1}{=} 1_{(m',p')}^p \mapsto m = m' \wedge \bigwedge_{p' < p \leq |\Sigma|} \neg \text{Tree}^{(\alpha_p)}(m) \wedge \widehat{\text{CNT}}_n^{(\alpha_{p'})}(m) \quad (132)$$

$$O_{(m,p)}^p \stackrel{\leq_n^w}{=} O_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} O_{(m,p)}^p \stackrel{\leq_{n_1}^{p_1}}{=} 1_{(m)}^w \stackrel{\leq_{n_2}^w}{=} O_{(m')}^w \quad (133)$$

$$O_{(m,p)}^p \stackrel{\leq_n^p}{=} O_{(m')}^w \mapsto \text{false} \quad (134)$$

$$O_{(m,p)}^p \stackrel{\leq_n^1}{=} O_{(m')}^w \mapsto \text{false} \quad (135)$$

$$O_{(m,p)}^p \stackrel{\leq_n^w}{=} O_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} O_{(m,p)}^p \stackrel{\leq_{n_1}^{p_1}}{=} 1_{(m,p)}^p \stackrel{\leq_{n_2}^{p_1}}{=} 1_{(m)}^w \stackrel{\leq_{n_3}^w}{=} O_{(m')}^w \stackrel{\leq_{n_4}^{p_1}}{=} O_{(m',p')}^p \quad (136)$$

$$O_{(m,p)}^p \stackrel{\leq_n^p}{=} O_{(m',p')}^p \mapsto \left(\begin{array}{l} m = m' \wedge p' < p \\ \wedge \\ \bigvee_{n_{p'}+\dots+n_{p-1}=n} \bigwedge_{p' \leq i < p} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (137)$$

$$0_{(m,p)}^p \stackrel{\leq^1}{\cong}_n 0_{(m',p')}^p \mapsto \text{false} \quad (138)$$

$$0_{(m,p)}^p \stackrel{\leq^w}{\cong}_n 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} 0_{(m,p)}^p \stackrel{\leq^{pl}}{\cong}_{n_1} 1_{(m,p)}^p \stackrel{\leq^{pl}}{\cong}_{n_2} 1_{(m)}^w \stackrel{\leq^w}{\cong}_{n_3} 0_{(m')}^w \stackrel{\leq^{pl}}{\cong}_{n_4} 1_{(m')}^w \quad (139)$$

$$0_{(m,p)}^p \stackrel{\leq^p}{\cong}_n 1_{(m')}^w \mapsto \left(\begin{array}{l} m = m' \wedge \bigvee_{0 < i < p} \text{Tree}^{(\alpha_i)}(m) \\ \wedge \bigvee_{n_1+\dots+n_p=n+1} \bigwedge_{0 < i \leq p} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (140)$$

$$0_{(m,p)}^p \stackrel{\leq^1}{\cong}_n 1_{(m')}^w \mapsto m = m' \wedge \bigwedge_{0 < i < p} \neg \text{Tree}^{(\alpha_i)}(m) \wedge \widehat{\text{CNT}}_n^{(\alpha_p)}(m). \quad (141)$$

$$0_{(m,p)}^p \stackrel{\leq^w}{\cong}_n 1_{(m',p')}^p \mapsto \bigvee_{\substack{\sum_{i=1}^5 n_i = n \\ n_3 > 0}} 0_{(m,p)}^p \stackrel{\leq^1}{\cong}_{n_1} 1_{(m,p)}^p \stackrel{\leq^{pl}}{\cong}_{n_2} 1_{(m)}^w \stackrel{\leq^w}{\cong}_{n_3} 0_{(m')}^w \stackrel{\leq^{pl}}{\cong}_{n_4} 0_{(m',p')}^p \stackrel{\leq^1}{\cong}_{n_5} 1_{(m',p')}^p \quad (142)$$

$$0_{(m,p)}^p \stackrel{\leq^p}{\cong}_n 1_{(m',p')}^p \mapsto \left(\begin{array}{l} m = m' \wedge p' < p \\ \wedge \\ \bigvee_{n_{p'}+\dots+n_p=n+1} \bigwedge_{p' \leq i \leq p} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (143)$$

$$0_{(m,p)}^p \stackrel{\leq^1}{\cong}_n 1_{(m',p')}^p \mapsto m = m' \wedge p = p' \wedge \widehat{\text{CNT}}_n^{(\alpha_p)}(m) \quad (144)$$

$$1_{(m)}^w \stackrel{\leq^w}{\cong}_n 0_{(m')}^w \mapsto \left(\begin{array}{l} n = 1 \rightarrow (m < m' \wedge \forall z (m < z < m' \rightarrow \neg \text{Tree}(z))) \\ \wedge \\ n > 1 \rightarrow \bigvee_{0 < r < n} (\exists z_1 \dots \exists z_r) \left(\begin{array}{l} m < z_1 < \dots < z_r < m' \\ \wedge \\ \neg (\exists z_1 \dots \exists z_{r+1}) \\ (m < z_1 < \dots < z_{r+1} < m') \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_1, \dots, n_r > 0}} \bigwedge_{0 < i \leq r} \widehat{\text{CNT}}_{n_i-1}(z_i) \end{array} \right) \end{array} \right) \quad (145)$$

$$1_{(m)}^w \stackrel{\leq^p}{\cong}_n 0_{(m')}^w \mapsto \text{false} \quad (146)$$

$$1_{(m)}^w \stackrel{\leq^1}{\cong}_n 0_{(m')}^w \mapsto \text{false} \quad (147)$$

$$1_{(m)}^w \stackrel{\leq^w}{\cong}_n 0_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 1_{(m)}^w \stackrel{\leq^w}{\cong}_{n_1} 0_{(m')}^w \stackrel{\leq^{pl}}{\cong}_{n_2} 0_{(m',p')}^p \quad (148)$$

$$1_{(m)}^w \stackrel{\leq^p}{\cong}_n 0_{(m',p')}^p \mapsto \text{false} \quad (149)$$

$$1_{(m)}^w \stackrel{\leq^1}{\cong}_n 0_{(m',p')}^p \mapsto \text{false} \quad (150)$$

$$1_{(m)}^w \leq_n^w 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 1_{(m)}^w \leq_{n_1}^w 0_{(m')}^w \leq_{n_2}^{pl} 1_{(m')}^w \quad (151)$$

$$1_{(m)}^w \leq_n^p 1_{(m')}^w \mapsto \text{false} \quad (152)$$

$$1_{(m)}^w \leq_n^l 1_{(m')}^w \mapsto \text{false} \quad (153)$$

$$1_{(m)}^w \leq_n^w 1_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_1>0}} 1_{(m)}^w \leq_{n_1}^w 0_{(m')}^w \leq_{n_2}^{pl} 0_{(m',p')}^w \leq_{n_3}^{pl} 1_{(m',p')}^w \quad (154)$$

$$1_{(m)}^w \leq_n^p 1_{(m',p')}^p \mapsto \text{false} \quad (155)$$

$$1_{(m)}^w \leq_n^l 1_{(m',p')}^p \mapsto \text{false} \quad (156)$$

$$1_{(m,p)}^p \leq_n^w 0_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} 1_{(m,p)}^p \leq_{n_1}^{pl} 1_{(m)}^w \leq_{n_2}^w 0_{(m')}^w \quad (157)$$

$$1_{(m,p)}^p \leq_n^p 0_{(m')}^w \mapsto \text{false} \quad (158)$$

$$1_{(m,p)}^p \leq_n^l 0_{(m')}^w \mapsto \text{false} \quad (159)$$

$$1_{(m,p)}^p \leq_n^w 0_{(m',p')}^p \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} 1_{(m,p)}^p \leq_{n_1}^{pl} 1_{(m)}^w \leq_{n_2}^w 0_{(m')}^w \leq_{n_3}^w 0_{(m',p')}^p \quad (160)$$

$$1_{(m,p)}^p \leq_n^p 0_{(m',p')}^p \mapsto \left(\begin{array}{l} n = 1 \rightarrow (m = m' \wedge p' < p \wedge \bigwedge_{p' < i < p} \neg \text{Tree}^{\alpha_i}(m)) \\ \wedge \\ n > 1 \rightarrow \left(\begin{array}{l} m = m' \wedge p' < p - 1 \wedge \\ \bigvee_{n_{p'+1}+\dots+n_{p-1}=n-1} \bigwedge_{p' < i < p} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{\alpha_i}(m)) \end{array} \right) \end{array} \right) \quad (161)$$

$$1_{(m,p)}^p \leq_n^l 0_{(m',p')}^p \mapsto \text{false} \quad (162)$$

$$1_{(m,p)}^p \leq_n^w 1_{(m')}^w \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} 1_{(m,p)}^p \leq_{n_1}^{pl} 1_{(m)}^w \leq_{n_2}^w 0_{(m')}^w \leq_{n_3}^{pl} 1_{(m')}^w \quad (163)$$

$$1_{(m,p)}^p \leq_n^p 1_{(m')}^w \mapsto m = m' \wedge \bigvee_{n_1+\dots+n_{p-1}=n} \bigwedge_{0 < i < p} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{\alpha_i}(m)) \quad (164)$$

$$1_{(m,p)}^p \leq_n^l 1_{(m')}^w \mapsto \text{false} \quad (165)$$

$$1_{(m,p)}^p \leq_n^w 1_{(m',p')}^w \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} 1_{(m,p)}^p \leq_{n_1}^{pl} 1_{(m)}^w \leq_{n_2}^w 0_{(m')}^w \leq_{n_3}^{pl} 0_{(m',p')}^w \leq_{n_4}^{pl} 1_{(m',p')}^w \quad (166)$$

$$1_{(m,p)}^p \stackrel{\leq_n^p}{\cong} 1_{(m',p')}^p \mapsto \left(\begin{array}{c} m = m' \wedge p' < p \\ \wedge \\ \bigvee_{n_{p'} + \dots + n_{p-1} = n} \bigwedge_{p' \leq i < p} (n_i > 0 \rightarrow \widehat{\text{CNT}}_{n_i-1}^{(\alpha_i)}(m)) \end{array} \right) \quad (167)$$

$$1_{(m,p)}^p \stackrel{\leq_n^p}{\cong} 1_{(m',p')}^p \mapsto \text{false} \quad (168)$$

C.4 Reduction of Half Open Stretchable Gap Orders

$$0_{(m)}^w <_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m)}^w <_{n_1}^w 0_{(u^w)}^w <_{n_2}^{pl} u \quad (169)$$

$$0_{(m)}^w <_n^p u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m)}^w <_{n_1}^p 0_{(m,q)}^p <_{n_2}^l u \quad (170)$$

$$0_{(m)}^w <_n^l u \mapsto \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \bigwedge_{q < i \leq |\Sigma|} \neg \text{Tree}^{(\alpha_i)}(u^w) \wedge \bar{0}_{(m-w(\alpha_q))}^{k, kb} <_n^{k, kb} \vec{u} \quad (171)$$

$$0_{(m,p)}^p <_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m,p)}^p <_{n_1}^w 0_{(u^w)}^w <_{n_2}^{pl} u \quad (172)$$

$$0_{(m,p)}^p <_n^p u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m,p)}^p <_{n_1}^p 0_{(m,q)}^p <_{n_2}^l u \quad (173)$$

$$0_{(m,p)}^p <_n^l u \mapsto p = q \wedge \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \bar{0}_{(m-w(\alpha_q))}^{k, kb} <_n^{k, kb} \vec{u} \quad (174)$$

$$1_{(m)}^w <_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 1_{(m)}^w <_{n_1}^w 0_{(u^w)}^w <_{n_2}^{pl} u \quad (175)$$

$$1_{(m,p)}^p <_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 1_{(m,p)}^p <_{n_1}^w 0_{(u^w)}^w <_{n_2}^{pl} u \quad (176)$$

$$u <_n^w 1_{(m)}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u <_{n_1}^{pl} 1_{(u^w)}^w <_{n_2}^w 1_{(m)}^w \quad (177)$$

$$u <_n^p 1_{(m)}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u <_{n_1}^l 1_{(m,q)}^p <_{n_2}^p 1_{(m)}^w \quad (178)$$

$$u <_n^l 1_{(m)}^w \mapsto \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \bigwedge_{0 < i < q} \neg \text{Tree}^{(\alpha_i)}(u^w) \wedge \vec{u} <_n^{k, kb} \vec{1}_{(m-w(\alpha_q))}^{k, kb} \quad (179)$$

$$u <_n^w 1_{(m,p)}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u <_{n_1}^{pl} 1_{(u^w)}^w <_{n_2}^w 1_{(m,p)}^p \quad (180)$$

$$u <_n^p 1_{(m,p)}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u <_{n_1}^l 1_{(m,q)}^p <_{n_2}^p 1_{(m,p)}^p \quad (181)$$

$$u <_n^l 1_{(m,p)}^p \mapsto p = q \wedge \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \vec{u} <_n^{k, \text{kb}} \vec{1}_{(m-w(\alpha_q))}^{k, \text{kb}} \quad (182)$$

$$u <_n^w 0_{(m)}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u <_{n_1}^{p1} 1_{(u^w)}^w <_{n_2}^w 0_{(m)}^w \quad (183)$$

$$u <_n^w 0_{(m,p)}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u <_{n_1}^{p1} 1_{(u^w)}^w <_{n_2}^w 0_{(m,p)}^p \quad (184)$$

C.5 Reduction of Half Open Rigid Gap Orders

$$0_{(m)}^w \leq_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m)}^w \leq_{n_1}^w 0_{(u^w)}^w \leq_{n_2}^{p1} u \quad (185)$$

$$0_{(m)}^w \leq_n^p u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m)}^w \leq_{n_1}^p 0_{(m,q)}^p \leq_{n_2}^l u \quad (186)$$

$$0_{(m)}^w \leq_n^l u \mapsto \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \bigwedge_{q < i \leq |\Sigma|} \neg \text{Tree}^{(\alpha_i)}(u^w) \wedge \vec{0}_{(m-w(\alpha_q))}^{k, \text{kb}} \leq_n^{k, \text{kb}} \vec{u} \quad (187)$$

$$0_{(m,p)}^p \leq_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m,p)}^p \leq_{n_1}^w 0_{(u^w)}^w \leq_{n_2}^{p1} u \quad (188)$$

$$0_{(m,p)}^p \leq_n^p u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 0_{(m,p)}^p \leq_{n_1}^p 0_{(m,q)}^p \leq_{n_2}^l u \quad (189)$$

$$0_{(m,p)}^p \leq_n^l u \mapsto p = q \wedge \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \vec{0}_{(m-w(\alpha_q))}^{k, \text{kb}} \leq_n^{k, \text{kb}} \vec{u} \quad (190)$$

$$1_{(m)}^w \leq_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 1_{(m)}^w \leq_{n_1}^w 0_{(u^w)}^w \leq_{n_2}^{p1} u \quad (191)$$

$$1_{(m,p)}^p \leq_n^w u \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} 1_{(m,p)}^p \leq_{n_1}^w 0_{(u^w)}^w \leq_{n_2}^{p1} u \quad (192)$$

$$u \leq_n^w 1_{(m)}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u \leq_{n_1}^{p1} 1_{(u^w)}^w \leq_{n_2}^w 1_{(m)}^w \quad (193)$$

$$u \leq_n^p 1_{(m)}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u \leq_{n_1}^l 1_{(m,q)}^p \leq_{n_2}^p 1_{(m)}^w \quad (194)$$

$$u \leq_n^l 1_{(m)}^w \mapsto \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \bigwedge_{0 < i < q} \neg \text{Tree}^{(\alpha_i)}(u^w) \wedge \vec{u} \leq_n^{k, \text{kb}} \vec{1}_{(m-w(\alpha_q))}^{k, \text{kb}} \quad (195)$$

$$u \cong_n^w 1_{(m,p)}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u \leq_{n_1}^{pl} 1_{(u^w)}^w \leq_{n_2}^w 1_{(m,p)}^p \quad (196)$$

$$u \leq_n^p 1_{(m,p)}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u \leq_{n_1}^l 1_{(m,q)}^p \leq_{n_2}^p 1_{(m,p)}^p \quad (197)$$

$$u \leq_n^l 1_{(m,p)}^p \mapsto p = q \wedge \text{Is}_{\alpha_q}(u) \wedge m = u^w \wedge \vec{u} \leq_n^{k, kb} \vec{1}_{(m-w(\alpha_q))}^{k, kb} \quad (198)$$

$$u \leq_n^w 0_{(m)}^w \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u \leq_{n_1}^{pl} 1_{(u^w)}^w \leq_{n_2}^w 0_{(m)}^w \quad (199)$$

$$u \leq_n^w 0_{(m,p)}^p \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} u \leq_{n_1}^{pl} 1_{(u^w)}^w \leq_{n_2}^w 0_{(m,p)}^p \quad (200)$$

C.6 Reductions of Open Stretchable Gap Orders

$$u <_n^w v \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} u <_{n_1}^{pl} 1_{(u^w)}^w <_{n_2}^w 0_{(v^w)}^w <_{n_3}^{pl} v \quad (201)$$

$$u <_n^p v \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} u <_{n_1}^l 1_{(u^w,p)}^p <_{n_2}^p 0_{(v^w,q)}^p <_{n_3}^l v \quad (202)$$

where $\text{type}(u) = \alpha_p$ and $\text{type}(v) = \alpha_q$

$$u <_n^l v \mapsto \vec{u} <_n^{k, kb} \vec{v} \quad (203)$$

where $u = \alpha(u_1, \dots, u_k)$ and $v = \alpha(v_1, \dots, v_k)$

C.7 Reductions of Open Rigid Gap Orders

$$u \leq_n^w v \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} u \leq_{n_1}^{pl} 1_{(u^w)}^w \leq_{n_2}^w 0_{(v^w)}^w \leq_{n_3}^{pl} v \quad (204)$$

$$u \leq_n^p v \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} u \leq_{n_1}^l 1_{(u^w,p)}^p \leq_{n_2}^p 0_{(v^w,q)}^p \leq_{n_3}^l v \quad (205)$$

where $\text{type}(u) = \alpha_p$ and $\text{type}(v) = \alpha_q$

$$u \leq_n^l v \mapsto \vec{u} \leq_n^{k, kb} \vec{v} \quad (206)$$

where $u = \alpha(u_1, \dots, u_k)$ and $v = \alpha(v_1, \dots, v_k)$

C.8 Reduction of Closed Tuple Equalities

$$\bar{O}_{(sum,m)}^{k,w} = \bar{O}_{(sum',m')}^{k,w} \mapsto sum = sum' \wedge m = m' \quad (207)$$

$$\bar{O}_{(sum,m)}^{k,w} = \bar{O}_{(sum',m',p')}^{k,p} \mapsto sum = sum' \wedge m = m' \wedge \bigwedge_{p' < i \leq |\Sigma|} \neg \text{Tree}^{(\alpha_i)}(m) \quad (208)$$

$$\bar{0}_{(sum,m)}^{k:w} = \bar{1}_{(sum',m')}^{k:w} \mapsto \left(\begin{array}{l} sum = sum' \wedge m = m' \wedge \neg \text{CNT}_1(m) \\ \wedge \bar{0}_{(sum-m)}^{k-1:kb} = \bar{1}_{(sum'-m')}^{k-1:kb} \end{array} \right) \quad (209)$$

$$\bar{0}_{(sum,m)}^{k:w} = \bar{1}_{(sum',m',p')}^{k:p} \mapsto \left(\begin{array}{l} sum = sum' \wedge m = m' \wedge \bigwedge_{p' < i \leq |\Sigma|} \neg \text{Tree}^{(a_i)}(m) \\ \wedge \neg \text{CNT}_1^{(a_{p'})}(m) \wedge \bar{0}_{(sum-m)}^{k-1:kb} = \bar{1}_{(sum'-m')}^{k-1:kb} \end{array} \right) \quad (210)$$

$$\bar{0}_{(sum,m,p)}^{k:p} = \bar{0}_{(sum',m',p')}^{k:p} \mapsto sum = sum' \wedge m = m' \wedge p = p' \quad (211)$$

$$\bar{0}_{(sum,m,p)}^{k:p} = \bar{1}_{(sum',m')}^{k:w} \mapsto \left(\begin{array}{l} sum = sum' \wedge m = m' \wedge \bigwedge_{0 < i < p} \neg \text{Tree}^{(a_i)}(m) \\ \wedge \neg \text{CNT}_1^{(a_p)}(m) \wedge \bar{0}_{(sum-m)}^{k-1:kb} = \bar{1}_{(sum'-m')}^{k-1:kb} \end{array} \right) \quad (212)$$

$$\bar{0}_{(sum,m,p)}^{k:p} = \bar{1}_{(sum',m',p')}^{k:p} \mapsto \left(\begin{array}{l} sum = sum' \wedge m = m' \wedge p = p' \\ \wedge \neg \text{CNT}_1^{(a_p)}(m) \wedge \bar{0}_{(sum-m)}^{k-1:kb} = \bar{1}_{(sum'-m')}^{k-1:kb} \end{array} \right) \quad (213)$$

$$\bar{1}_{(sum,m)}^{k:w} = \bar{1}_{(sum',m')}^{k:w} \mapsto sum = sum' \wedge m = m' \quad (214)$$

$$\bar{1}_{(sum,m)}^{k:w} = \bar{1}_{(sum',m',p')}^{k:p} \mapsto sum = sum' \wedge m = m' \wedge \bigwedge_{0 < i < p'} \neg \text{Tree}^{(a_i)}(m) \quad (215)$$

$$\bar{1}_{(sum,m,p)}^{k:p} = \bar{1}_{(sum',m',p')}^{k:p} \mapsto sum = sum' \wedge m = m' \wedge p = p' \quad (216)$$

C.9 Reduction of Half Open Tuple Equalities.

$$\bar{0}_{(sum)}^{k:kb} = \vec{u} \mapsto \bar{0}_{(sum, \text{MinCW}^k(sum))}^{k:w} = \vec{u} \quad (217)$$

$$\bar{0}_{(sum,m)}^{k:w} = \vec{u} \mapsto 0_{(m)}^w = u_1 \wedge \bar{0}_{(sum-m)}^{k-1:kb} = \langle u_2, \dots, u_k \rangle \quad (218)$$

$$\bar{0}_{(sum,m,p)}^{k:p} = \vec{u} \mapsto 0_{(m,p)}^p = u_1 \wedge \bar{0}_{(sum-m)}^{k-1:kb} = \langle u_2, \dots, u_k \rangle \quad (219)$$

$$\bar{1}_{(sum)}^{k:kb} = \vec{u} \mapsto \bar{1}_{(sum, \text{MaxCW}^k(sum))}^{k:w} = \vec{u} \quad (220)$$

$$\bar{1}_{(sum,m)}^{k:w} = \vec{u} \mapsto 1_{(m)}^w = u_1 \wedge \bar{1}_{(sum-m)}^{k-1:kb} = \langle u_2, \dots, u_k \rangle \quad (221)$$

$$\bar{1}_{(sum,m,p)}^{k:p} = \vec{u} \mapsto 1_{(m,p)}^w = u_1 \wedge \bar{1}_{(sum-m)}^{k-1:kb} = \langle u_2, \dots, u_k \rangle \quad (222)$$

C.10 Reductions of Closed Stretchable Tuple Gap Orders

All reductions shown from this section to Section C.15 are reductions of *proper* gap orders between tuples of the same weight. This amounts to assuming that in each redex (reduction target) we have an implicit side condition $u_1 <^\# v_1$ or $u_1 \leq^\# v_1$ where u_1 (resp. v_1) is the leftmost immediate subterm of the left (resp. the right) operand.

$$\bar{0}_{(sum,m)}^{k:w} <_n^{k:w} \bar{0}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \bar{0}_{(sum,m)}^{k:w} <_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} <_{n_2}^{k:w} \bar{0}_{(sum,m')}^{k:w} \quad (223)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:p} \bar{0}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (224)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:l} \bar{0}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (225)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \bar{0}_{(sum,m)}^{k:w} \prec_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \prec_{n_2}^{k:w} \bar{0}_{(sum,m')}^{k:w} \prec_{n_3}^{k:pl} \bar{0}_{(sum,m',p')}^{k:p} \quad (226)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:p} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1(n_2+1) \geq n \\ n_1 > 0, n_1, n_2 \leq n}} 0_{(m)}^w \prec_{n_1}^p 0_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1:kb} \prec_{n_2}^{k-1:kb} \bar{1}_{(sum-m)}^{k-1:kb} \quad (227)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:l} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \text{false} \quad (228)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{1}_{(sum,m')}^{k:w} \mapsto$$

$$\left(\begin{array}{l} m = m' \rightarrow \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1, n_2 \leq (n+1)}} \left(\begin{array}{l} 0_{(m)}^w \prec_{n_1}^w 1_{(m)}^w \\ \wedge \bar{0}_{sum-m}^{k-1:kb} \prec_{n_2}^{k-1:kb} \bar{1}_{sum-m}^{k-1:kb} \end{array} \right) \\ \wedge \\ m < m' \rightarrow \\ \bigvee_{1 < r \leq n+1} \left(\begin{array}{l} l < CW_1^k(sum, l, l') < \dots < CW_r^k(sum, l, l') < l' \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n+1 \\ n_i > 0}} \bigwedge_{0 < i \leq r} \bar{0}_{(sum, CW_i^k(sum, l, l'))}^{k:w} \prec_{n_i-1}^{k:w} \bar{1}_{(sum, CW_i^k(sum, l, l'))}^{k:w} \end{array} \right) \end{array} \right)$$

$$\text{where } l \equiv m - 1 \text{ and } l' \equiv m' + 1. \quad (229)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:p} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1 > 0, n_1, n_2 \leq n+1}} 0_{(m)}^w \prec_{n_1}^p 1_{(m')}^w \wedge \bar{0}_{(sum-m)}^{k-1:kb} \prec_{n_2}^{k-1:kb} \bar{1}_{(sum-m)}^{k-1:kb} \quad (230)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:l} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1 > 0, n_1, n_2 \leq n+1}} 0_{(m)}^w \prec_{n_1}^l 1_{(m')}^w \wedge \bar{0}_{(sum-m)}^{k-1:kb} \prec_{n_2}^{k-1:kb} \bar{1}_{(sum-m)}^{k-1:kb} \quad (231)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} \left(\begin{array}{l} \bar{0}_{(sum,m)}^{k:w} \prec_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \prec_{n_2}^{k:w} \bar{0}_{(sum,m')}^{k:w} \\ \wedge \prec_{n_3}^{k:pl} \bar{0}_{(sum,m',p')}^{k:p} \prec_{n_4}^{k:pl} \bar{1}_{(sum,m',p')}^{k:p} \end{array} \right) \quad (232)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:p} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1 > 0, n_1, n_2 \leq n+1}} 0_{(m)}^w \prec_{n_1}^p 1_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1:kb} \prec_{n_2}^{k-1:kb} \bar{1}_{(sum-m)}^{k-1:kb} \quad (233)$$

$$\bar{0}_{(sum,m)}^{k:w} \prec_n^{k:l} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1 > 0, n_1, n_2 \leq n+1}} 0_{(m)}^w \prec_{n_1}^l 1_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1:kb} \prec_{n_2}^{k-1:kb} \bar{1}_{(sum-m)}^{k-1:kb} \quad (234)$$

$$\bar{0}_{(sum,m,p)}^{k:p} \prec_n^{k:w} \bar{0}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \bar{0}_{(sum,m,p)}^{k:p} \prec_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \prec_{n_2}^{k:w} \bar{0}_{(sum,m')}^{k:w} \quad (235)$$

$$\bar{0}_{(sum,m,p)}^{k:p} \prec_n^{k:p} \bar{0}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (236)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;l} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (237)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;w} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} \left(\bar{0}_{(sum,m,p)}^{k;p} \prec_{n_1}^{k;pl} \bar{1}_{(sum,m,p)}^{k;p} \prec_{n_2}^{k;pl} \bar{1}_{(sum,m)}^{k;w} \right. \\ \left. \prec_{n_3}^{k;w} \bar{0}_{(sum,m')}^{k;w} \prec_{n_4}^{k;pl} \bar{0}_{(sum,m',p')}^{k;p} \right) \quad (238)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;p} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1(n_2+1) \geq n \\ n_1>0, n_1, n_2 \leq n}} \mathbf{0}_{(m,p)}^p \prec_{n_1}^p \mathbf{0}_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (239)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;l} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (240)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;w} \bar{1}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} \left(\bar{0}_{(sum,m,p)}^{k;p} \prec_{n_1}^{k;pl} \bar{1}_{(sum,m,p)}^{k;p} \prec_{n_2}^{k;w} \bar{1}_{(sum,m)}^{k;w} \right. \\ \left. \prec_{n_3}^{k;pl} \bar{0}_{(sum,m')}^{k;w} \prec_{n_4}^{k;pl} \bar{1}_{(sum,m')}^{k;w} \right) \quad (241)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;p} \bar{1}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1>0, n_1, n_2 \leq n+1}} \mathbf{0}_{(m,p)}^p \prec_{n_1}^p \mathbf{1}_{(m')}^w \wedge \bar{0}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (242)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;l} \bar{1}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1>0, n_1, n_2 \leq n+1}} \mathbf{0}_{(m,p)}^p \prec_{n_1}^l \mathbf{1}_{(m')}^w \wedge \bar{0}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (243)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;w} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4+n_5=n \\ n_3>0}} \left(\bar{0}_{(sum,m,p)}^{k;p} \prec_{n_1}^{k;pl} \bar{1}_{(sum,m,p)}^{k;p} \prec_{n_2}^{k;pl} \bar{1}_{(sum,m,p)}^{k;w} \right. \\ \left. \prec_{n_3}^{k;w} \bar{0}_{(sum,m')}^{k;w} \prec_{n_4}^{k;pl} \bar{0}_{(sum,m',p')}^{k;p} \right. \\ \left. \prec_{n_5}^{k;pl} \bar{1}_{(sum,m',p')}^{k;p} \right) \quad (244)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;p} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1>0, n_1, n_2 \leq n+1}} \mathbf{0}_{(m,p)}^p \prec_{n_1}^p \mathbf{1}_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (245)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \prec_n^{k;l} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1) \geq n+1 \\ n_1>0, n_1, n_2 \leq n+1}} \mathbf{0}_{(m,p)}^p \prec_{n_1}^l \mathbf{1}_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (246)$$

$$\bar{1}_{(sum,m)}^{k;w} \prec_n^{k;w} \bar{0}_{(sum,m')}^{k;w} \mapsto \left(\begin{array}{c} (n = 1 \rightarrow m < m') \\ \wedge \\ n > 1 \rightarrow \\ \bigvee_{0 < r < n} \left(\bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_i > 0}} \bigwedge_{0 < i \leq r} \bar{0}_{(sum, CW_i^k(sum, m, m'))}^{k;w} \prec_{n_i-1}^{k;w} \bar{1}_{(sum, CW_i^k(sum, m, m'))}^{k;w} \right) \end{array} \right) \quad (247)$$

$$\bar{1}_{(sum,m)}^{k;w} \prec_n^{k;p} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (248)$$

$$\bar{1}_{(sum,m)}^{k;w} \prec_n^{k;l} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (249)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_1>0}} \bar{1}_{(sum,m)}^{k:w} \prec_{n_1}^{k:w} \bar{0}_{(sum,m')}^{k:w} \prec_{n_2}^{k:pl} \bar{0}_{(sum,m',p')}^{k:p} \quad (250)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:p} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \text{false} \quad (251)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:l} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \text{false} \quad (252)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} \bar{1}_{(sum,m)}^{k:w} \prec_{n_1}^{k:w} \bar{0}_{(sum,m')}^{k:w} \prec_{n_2}^{k:pl} \bar{1}_{(sum,m')}^{k:w} \quad (253)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:p} \bar{1}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (254)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:l} \bar{1}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (255)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:w} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_1>0}} \bar{1}_{(sum,m)}^{k:w} \prec_{n_1}^{k:w} \bar{0}_{(sum,m')}^{k:w} \prec_{n_2}^{k:pl} \bar{0}_{(sum,m',p')}^{k:p} \prec_{n_3}^{k:pl} \bar{1}_{(sum,m',p')}^{k:p} \quad (256)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:p} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \text{false} \quad (257)$$

$$\bar{1}_{(sum,m)}^{k:w} \prec_n^{k:l} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \text{false} \quad (258)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:w} \bar{0}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \bar{1}_{(sum,m,p)}^{k:p} \prec_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \prec_{n_2}^{k:w} \bar{0}_{(sum,m')}^{k:w} \quad (259)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:p} \bar{0}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (260)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:l} \bar{0}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (261)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:w} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \bar{1}_{(sum,m,p)}^{k:p} \prec_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \prec_{n_2}^{k:w} \bar{0}_{(sum,m')}^{k:w} \prec_{n_3}^{k:pl} \bar{0}_{(sum,m',p')}^{k:p} \quad (262)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:p} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{(n_1-1)(n_2+1) \geq n-1 \\ n_1>0, n_1, n_2 < n}} \mathbf{0}_{(m)}^p \prec_{n_1}^p \mathbf{1}_{(m)}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (263)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:l} \bar{0}_{(sum,m',p')}^{k:p} \mapsto \text{false} \quad (264)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:w} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} \bar{1}_{(sum,m,p)}^{k:p} \prec_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \prec_{n_2}^{k:w} \bar{0}_{(sum,m')}^{k:w} \prec_{n_3}^{k:pl} \bar{1}_{(sum,m')}^{k:w} \quad (265)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:p} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{(n_1)(n_2+1) \geq n \\ n_1>0, n_1, n_2 \leq n}} \mathbf{1}_{(m,p)}^p \prec_{n_1}^p \mathbf{1}_{(m')}^w \wedge \bar{0}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (266)$$

$$\bar{1}_{(sum,m,p)}^{k:p} \prec_n^{k:l} \bar{1}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (267)$$

$$\bar{\uparrow}_{(sum,m,p)}^{k;p} \leq_n^{k;w} \bar{\uparrow}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} \left(\bar{\uparrow}_{(sum,m,p)}^{k;p} \leq_{n_1}^{k;p} \bar{\uparrow}_{(sum,m)}^{k;w} \leq_{n_2}^{k;w} \bar{0}_{(sum,m')}^{k;w} \right. \\ \left. \leq_{n_3}^{k;p} \bar{0}_{(sum,m',p')}^{k;p} \leq_{n_4}^{k;p} \bar{\uparrow}_{(sum,m',p')}^{k;p} \right) \quad (268)$$

$$\bar{\uparrow}_{(sum,m,p)}^{k;p} \leq_n^{k;p} \bar{\uparrow}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1(n_2+1) \geq n \\ n_1>0, n_1, n_2 \leq n}} 1_{(m,p)}^p \leq_{n_1}^p 1_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \quad (269)$$

$$\bar{\uparrow}_{(sum,m,p)}^{k;p} \leq_n^{k;l} \bar{\uparrow}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (270)$$

C.11 Reductions of Closed Rigid Tuple Gap Orders

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;w} \bar{0}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \bar{0}_{(sum,m)}^{k;w} \leq_{n_1}^{k;p} \bar{\uparrow}_{(sum,m)}^{k;w} \leq_{n_2}^{k;w} \bar{0}_{(sum,m')}^{k;w} \quad (271)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;p} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (272)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;l} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (273)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;w} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \bar{0}_{(sum,m)}^{k;w} \leq_{n_1}^{k;p} \bar{\uparrow}_{(sum,m)}^{k;w} \leq_{n_2}^{k;w} \bar{0}_{(sum,m')}^{k;w} \leq_{n_3}^{k;p} \bar{0}_{(sum,m',p')}^{k;p} \quad (274)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;p} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1(n_2+1)=n \\ n_1>0, n_1, n_2 \leq n}} 0_{(m)}^w \leq_{n_1}^p 0_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \quad (275)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;l} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (276)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;w} \bar{\uparrow}_{(sum,m')}^{k;w} \mapsto$$

$$\left(\begin{array}{l} m = m' \rightarrow \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1, n_2 \leq (n+1)}} \left(0_{(m)}^w \leq_{n_1}^w 1_{(m)}^w \right. \\ \quad \left. \wedge \bar{0}_{sum-m}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{\uparrow}_{sum-m}^{k-1;kb} \right) \\ \wedge \\ m < m' \rightarrow \\ \bigvee_{1 < r \leq n+1} \left(\begin{array}{l} I < CW_1^k(sum, I, I') < \dots < CW_r^k(sum, I, I') < I' \\ \wedge \\ \neg (I \leq CW_1^k(sum, I, I') < \dots < CW_{r+1}^k(sum, I, I') < I') \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n+1 \\ n_i > 0}} \bigwedge_{0 < i \leq r} \bar{0}_{(sum, CW_i^k(sum, I, I'))}^{k;w} \leq_{n_i-1}^{k;w} \bar{\uparrow}_{(sum, CW_i^k(sum, I, I'))}^{k;w} \end{array} \right) \end{array} \right)$$

$$\text{where } I \equiv m-1 \text{ and } I' \equiv m'+1. \quad (277)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;p} \bar{\uparrow}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_1, n_2 \leq n+1}} 0_{(m)}^w \leq_{n_1}^p 1_{(m')}^w \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \quad (278)$$

$$\bar{O}_{(sum,m)}^{k:w} \leq_n^{k:l} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_1, n_2 \leq n+1}} O_{(m)}^w \leq_{n_1}^l 1_{(m')}^w \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{1}_{(sum-m')}^{k-1:kb} \quad (279)$$

$$\bar{O}_{(sum,m)}^{k:w} \leq_n^{k:w} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} \left(\begin{array}{l} \bar{O}_{(sum,m)}^{k:w} \leq_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \leq_{n_2}^{k:w} \bar{O}_{(sum,m')}^{k:w} \\ \leq_{n_3}^{k:pl} \bar{O}_{(sum,m',p')}^{k:p} \leq_{n_4}^{k:pl} \bar{1}_{(sum,m',p')}^{k:p} \end{array} \right) \quad (280)$$

$$\bar{O}_{(sum,m)}^{k:w} \leq_n^{k:p} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_1, n_2 \leq n+1}} O_{(m)}^w \leq_{n_1}^p 1_{(m',p')}^p \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{1}_{(sum-m')}^{k-1:kb} \quad (281)$$

$$\bar{O}_{(sum,m)}^{k:w} \leq_n^{k:l} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_1, n_2 \leq n+1}} O_{(m)}^w \leq_{n_1}^l 1_{(m',p')}^p \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{1}_{(sum-m')}^{k-1:kb} \quad (282)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:w} \bar{O}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \bar{O}_{(sum,m,p)}^{k:p} \leq_{n_1}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \leq_{n_2}^{k:w} \bar{O}_{(sum,m')}^{k:w} \quad (283)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:p} \bar{O}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (284)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:l} \bar{O}_{(sum,m')}^{k:w} \mapsto \text{false} \quad (285)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:w} \bar{O}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} \left(\begin{array}{l} \bar{O}_{(sum,m,p)}^{k:p} \leq_{n_1}^{k:pl} \bar{1}_{(sum,m,p)}^{k:p} \leq_{n_2}^{k:pl} \bar{1}_{(sum,m)}^{k:w} \\ \leq_{n_3}^{k:w} \bar{O}_{(sum,m')}^{k:w} \leq_{n_4}^{k:pl} \bar{O}_{(sum,m',p')}^{k:p} \end{array} \right) \quad (286)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:p} \bar{O}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1(n_2+1)=n \\ n_1>0, n_1, n_2 \leq n}} O_{(m,p)}^p \leq_{n_1}^p O_{(m',p')}^p \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{1}_{(sum-m')}^{k-1:kb} \quad (287)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:l} \bar{O}_{(sum,m',p')}^{k:p} \mapsto \text{false} \quad (288)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:w} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_3>0}} \left(\begin{array}{l} \bar{O}_{(sum,m,p)}^{k:p} \leq_{n_1}^{k:pl} \bar{1}_{(sum,m,p)}^{k:p} \leq_{n_2}^{k:w} \bar{1}_{(sum,m)}^{k:w} \\ \leq_{n_3}^{k:pl} \bar{O}_{(sum,m')}^{k:w} \leq_{n_4}^{k:pl} \bar{1}_{(sum,m')}^{k:w} \end{array} \right) \quad (289)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:p} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_1, n_2 \leq n+1}} O_{(m,p)}^p \leq_{n_1}^p 1_{(m')}^w \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{1}_{(sum-m')}^{k-1:kb} \quad (290)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:l} \bar{1}_{(sum,m')}^{k:w} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_1, n_2 \leq n+1}} O_{(m,p)}^p \leq_{n_1}^l 1_{(m')}^w \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{1}_{(sum-m')}^{k-1:kb} \quad (291)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:w} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4+n_5=n \\ n_3>0}} \left(\begin{array}{l} \bar{O}_{(sum,m,p)}^{k:p} \leq_{n_1}^{k:pl} \bar{1}_{(sum,m,p)}^{k:p} \leq_{n_2}^{k:pl} \bar{1}_{(sum,m,p)}^{k:w} \\ \leq_{n_3}^{k:w} \bar{O}_{(sum,m')}^{k:w} \leq_{n_4}^{k:pl} \bar{O}_{(sum,m',p')}^{k:p} \\ \leq_{n_5}^{k:pl} \bar{1}_{(sum,m',p')}^{k:p} \end{array} \right) \quad (292)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:p} \bar{1}_{(sum,m',p')}^{k:p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_1, n_2 \leq n+1}} O_{(m,p)}^p \leq_{n_1}^p 1_{(m',p')}^p \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{1}_{(sum-m')}^{k-1:kb} \quad (293)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \leq_n^{k;l} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{(n_1+1)(n_2+1)=n+1 \\ n_1>0, n_2 \leq n+1}} \bar{0}_{(m,p)}^p \leq_{n_1}^l \bar{1}_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (294)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;w} \bar{0}_{(sum,m')}^{k;w} \mapsto \left(\begin{array}{l} n = 1 \rightarrow (m < m' \wedge \forall z (m < z < m' \rightarrow \neg \text{IsTW}^k(m))) \\ \wedge \\ n > 1 \rightarrow \\ \bigvee_{0 < r < n} \left(\begin{array}{l} m < \text{CW}_1^k(sum, m, m') < \dots < \text{CW}_r^k(sum, m, m') < m' \\ \wedge \\ \neg (m < \text{CW}_1^k(sum, m, m') < \dots < \text{CW}_{r+1}^k(sum, m, m') < m') \\ \wedge \\ \bigvee_{\substack{\sum_{i=1}^r n_i = n-1 \\ n_i > 0}} \bigwedge_{0 < i \leq r} \bar{0}_{(sum, \text{CW}_i^k(sum, m, m'))}^{k;w} \leq_{n_i-1}^{k;w} \bar{1}_{(sum, \text{CW}_i^k(sum, m, m'))}^{k;w} \end{array} \right) \end{array} \right) \quad (295)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;p} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (296)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;l} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (297)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;w} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_1>0}} \bar{1}_{(sum,m)}^{k;w} \leq_{n_1}^{k;w} \bar{0}_{(sum,m')}^{k;w} \leq_{n_2}^{k;p;l} \bar{0}_{(sum,m',p')}^{k;p} \quad (298)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;p} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (299)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;l} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (300)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;w} \bar{1}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} \bar{1}_{(sum,m)}^{k;w} \leq_{n_1}^{k;w} \bar{0}_{(sum,m')}^{k;w} \leq_{n_2}^{k;p;l} \bar{1}_{(sum,m')}^{k;w} \quad (301)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;p} \bar{1}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (302)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;l} \bar{1}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (303)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;w} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_1>0}} \bar{1}_{(sum,m)}^{k;w} \leq_{n_1}^{k;w} \bar{0}_{(sum,m')}^{k;w} \leq_{n_2}^{k;p;l} \bar{0}_{(sum,m',p')}^{k;p} \leq_{n_3}^{k;p;l} \bar{1}_{(sum,m',p')}^{k;p} \quad (304)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;p} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (305)$$

$$\bar{1}_{(sum,m)}^{k;w} \leq_n^{k;l} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (306)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;w} \bar{0}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \bar{1}_{(sum,m,p)}^{k;p} \leq_{n_1}^{k;p;l} \bar{1}_{(sum,m)}^{k;w} \leq_{n_2}^{k;w} \bar{0}_{(sum,m')}^{k;w} \quad (307)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;p} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (308)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{kl} \bar{0}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (309)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;w} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \bar{1}_{(sum,m,p)}^{k;p} \leq_{n_1}^{k;pl} \bar{1}_{(sum,m)}^{k;w} \leq_{n_2}^{k;w} \bar{0}_{(sum,m')}^{k;w} \leq_{n_3}^{k;pl} \bar{0}_{(sum,m',p')}^{k;p} \quad (310)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;p} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{(n_1-1)(n_2+1)=n-1 \\ n_1>0, n_1, n_2 < n}} \mathbf{0}_{(m)}^p \leq_{n_1}^p \mathbf{1}_{(m)}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (311)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{kl} \bar{0}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (312)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;w} \bar{1}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} \bar{1}_{(sum,m,p)}^{k;p} \leq_{n_1}^{k;pl} \bar{1}_{(sum,m)}^{k;w} \leq_{n_2}^{k;w} \bar{0}_{(sum,m')}^{k;w} \leq_{n_3}^{k;pl} \bar{1}_{(sum,m')}^{k;w} \quad (313)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;p} \bar{1}_{(sum,m')}^{k;w} \mapsto \bigvee_{\substack{(n_1)(n_2+1)=n \\ n_1>0, n_1, n_2 \leq n}} \mathbf{1}_{(m,p)}^p \leq_{n_1}^p \mathbf{1}_{(m')}^w \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (314)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{kl} \bar{1}_{(sum,m')}^{k;w} \mapsto \text{false} \quad (315)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;w} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2+n_3+n_4=n \\ n_2>0}} \left(\begin{array}{l} \bar{1}_{(sum,m,p)}^{k;p} \leq_{n_1}^{k;pl} \bar{1}_{(sum,m)}^{k;w} \leq_{n_2}^{k;w} \bar{0}_{(sum,m')}^{k;w} \\ \leq_{n_3}^{k;pl} \bar{0}_{(sum,m',p')}^{k;p} \leq_{n_4}^{k;pl} \bar{1}_{(sum,m',p')}^{k;p} \end{array} \right) \quad (316)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{k;p} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \bigvee_{\substack{n_1(n_2+1)=n \\ n_1>0, n_1, n_2 \leq n}} \mathbf{1}_{(m,p)}^p \leq_{n_1}^p \mathbf{1}_{(m',p')}^p \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \quad (317)$$

$$\bar{1}_{(sum,m,p)}^{k;p} \leq_n^{kl} \bar{1}_{(sum,m',p')}^{k;p} \mapsto \text{false} \quad (318)$$

C.12 Reductions of Half Open Stretchable Tuple Gap Orders

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;w} \vec{u} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} \bar{0}_{(sum,m)}^{k;w} \leq_{n_1}^{k;w} \bar{0}_{(sum,u_1^w)}^{k;w} \leq_{n_2}^{k;pl} \vec{u} \quad (319)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{k;p} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{0}_{(m)}^w \leq_{n_1}^p \mathbf{u}_1 \\ \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \\ \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_3}^{k-1;kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (320)$$

$$\bar{0}_{(sum,m)}^{k;w} \leq_n^{kl} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{0}_{(m)}^w \leq_{n_1}^l \mathbf{u}_1 \\ \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{1}_{(sum-m)}^{k-1;kb} \\ \wedge \bar{0}_{(sum-m)}^{k-1;kb} \leq_{n_3}^{k-1;kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (321)$$

$$\bar{0}_{(sum,m,p)}^{k;p} \leq_n^{k;w} \vec{u} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} \bar{0}_{(sum,m,p)}^{k;p} \leq_{n_1}^{k;w} \bar{0}_{(sum,u_1^w)}^{k;w} \leq_{n_2}^{k;pl} \vec{u} \quad (322)$$

$$\bar{O}_{(sum,m,p)}^{k;p} \prec_n^{k;p} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} O_{(m,p)}^p \prec_{n_1}^p \mathbf{u}_1 \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_3}^{k-1;kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (323)$$

$$\bar{O}_{(sum,m,p)}^{k;p} \prec_n^{k;l} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} O_{(m,p)}^p \prec_{n_1}^l \mathbf{u}_1 \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_3}^{k-1;kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (324)$$

$$\vec{u} \prec_n^{k;w} \bar{\uparrow}_{(sum,m)}^{k;w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \vec{u} \prec_{n_1}^{k;p1} \bar{\uparrow}_{(sum,u_1^w)}^{k;w} \prec_{n_2}^{k;w} \bar{\uparrow}_{(sum,m)}^{k;w} \quad (325)$$

$$\vec{u} \prec_n^{k;p} \bar{\uparrow}_{(sum,m)}^{k;w} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \prec_{n_1}^p \bar{\uparrow}_{(m)}^w \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \prec_{n_3}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \end{array} \right) \quad (326)$$

$$\vec{u} \prec_n^{k;l} \bar{\uparrow}_{(sum,m)}^{k;w} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \prec_{n_1}^l \bar{\uparrow}_{(m)}^w \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \prec_{n_3}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \end{array} \right) \quad (327)$$

$$\vec{u} \prec_n^{k;w} \bar{\uparrow}_{(sum,m,p)}^{k;p} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \vec{u} \prec_{n_1}^{k;p1} \bar{\uparrow}_{(sum,u_1^w)}^{k;w} \prec_{n_2}^{k;w} \bar{\uparrow}_{(sum,m,p)}^{k;p} \quad (328)$$

$$\vec{u} \prec_n^{k;p} \bar{\uparrow}_{(sum,m,p)}^{k;p} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \prec_{n_1}^p \bar{\uparrow}_{(m,p)}^p \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \prec_{n_3}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \end{array} \right) \quad (329)$$

$$\vec{u} \prec_n^{k;l} \bar{\uparrow}_{(sum,m,p)}^{k;p} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3 \geq n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \prec_{n_1}^l \bar{\uparrow}_{(m,p)}^p \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \prec_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \prec_{n_3}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \end{array} \right) \quad (330)$$

C.13 Reductions of Half Open Rigid Tuple Gap Orders

$$\bar{O}_{(sum,m)}^{k;w} \leq_n^{k;w} \vec{u} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} \bar{O}_{(sum,m)}^{k;w} \leq_{n_1}^{k;w} \bar{O}_{(sum,u_1^w)}^{k;w} \leq_{n_2}^{k;p1} \vec{u} \quad (331)$$

$$\bar{O}_{(sum,m)}^{k;w} \leq_n^{k;p} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} O_{(m)}^w \leq_{n_1}^p \mathbf{u}_1 \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \leq_{n_2}^{k-1;kb} \bar{\uparrow}_{(sum-m)}^{k-1;kb} \\ \wedge \bar{O}_{(sum-m)}^{k-1;kb} \leq_{n_3}^{k-1;kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (332)$$

$$\bar{O}_{(sum,m)}^{k:w} \leq_n^{k:l} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} O_{(m)}^w \leq_{n_1}^l \mathbf{u}_1 \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_3}^{k-1:kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (333)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:w} \vec{u} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_1>0}} \bar{O}_{(sum,m,p)}^{k:p} \leq_{n_1}^{k:w} \bar{O}_{(sum,u_1^w)}^{k:w} \leq_{n_2}^{k:p} \vec{u} \quad (334)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:p} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} O_{(m,p)}^p \leq_{n_1}^p \mathbf{u}_1 \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_3}^{k-1:kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (335)$$

$$\bar{O}_{(sum,m,p)}^{k:p} \leq_n^{k:l} \vec{u} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} O_{(m,p)}^p \leq_{n_1}^l \mathbf{u}_1 \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_3}^{k-1:kb} \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \end{array} \right) \quad (336)$$

$$\vec{u} \leq_n^{k:w} \bar{\Gamma}_{(sum,m)}^{k:w} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \vec{u} \leq_{n_1}^{k:p} \bar{\Gamma}_{(sum,u_1^w)}^{k:w} \leq_{n_2}^{k:w} \bar{\Gamma}_{(sum,m)}^{k:w} \quad (337)$$

$$\vec{u} \leq_n^{k:p} \bar{\Gamma}_{(sum,m)}^{k:w} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \leq_{n_1}^p \bar{\Gamma}_{(m)}^w \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \leq_{n_3}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \end{array} \right) \quad (338)$$

$$\vec{u} \leq_n^{k:l} \bar{\Gamma}_{(sum,m)}^{k:w} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \leq_{n_1}^l \bar{\Gamma}_{(m)}^w \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \leq_{n_3}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \end{array} \right) \quad (339)$$

$$\vec{u} \leq_n^{k:w} \bar{\Gamma}_{(sum,m,p)}^{k:p} \mapsto \bigvee_{\substack{n_1+n_2=n \\ n_2>0}} \vec{u} \leq_{n_1}^{k:p} \bar{\Gamma}_{(sum,u_1^w)}^{k:w} \leq_{n_2}^{k:w} \bar{\Gamma}_{(sum,m,p)}^{k:p} \quad (340)$$

$$\vec{u} \leq_n^{k:p} \bar{\Gamma}_{(sum,m,p)}^{k:p} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \leq_{n_1}^p \bar{\Gamma}_{(m,p)}^p \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \leq_{n_3}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \end{array} \right) \quad (341)$$

$$\vec{u} \leq_n^{k:l} \bar{\Gamma}_{(sum,m,p)}^{k:p} \mapsto \bigvee_{\substack{n_1(n_2+1)+n_3=n \\ n_1, n_2, n_3 \leq n}} \left(\begin{array}{l} \mathbf{u}_1 \leq_{n_1}^l \bar{\Gamma}_{(m,p)}^p \\ \wedge \bar{O}_{(sum-m)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \\ \wedge \langle \mathbf{u}_2, \dots, \mathbf{u}_k \rangle \leq_{n_3}^{k-1:kb} \bar{\Gamma}_{(sum-m)}^{k-1:kb} \end{array} \right) \quad (342)$$

C.14 Reductions of Open Stretchable Tuple Gap Orders

Let $sum = \sum_{i=1}^k u_i^w = \sum_{i=1}^k v_i^w$ and $rem = \sum_{i=2}^k u_i^w = \sum_{i=2}^k v_i^w$. Let $type(u_1) = \alpha_p$ and $type(v_1) = \alpha_{p'}$.

$$\vec{u} \prec_n^{k:w} \vec{v} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \left(\vec{u} \prec_{n_1}^{k:pl} \bar{\Gamma}_{(sum, u_1^w)}^{k:w} \prec_{n_2}^{k:w} \bar{O}_{(sum, v_1^w)}^{k:w} \prec_{n_3}^{k:pl} \vec{v} \right) \quad (343)$$

$$\vec{u} \prec_n^{k:p} \vec{v} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \left(\vec{u} \prec_{n_1}^{k:l} \bar{\Gamma}_{(sum, u_1^w, p)}^{k:p} \prec_{n_2}^{k:p} \bar{O}_{(sum, v_1^w, p')}^{k:p} \prec_{n_3}^{k:l} \vec{v} \right) \quad (344)$$

$$\vec{u} \prec_n^{k:l} \vec{v} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_3>0}} \left(\begin{array}{c} \langle u_2, \dots, u_k \rangle \prec_{n_1}^{k-1:kb} \bar{\Gamma}_{(rem)}^{k-1:kb} \\ \wedge \\ \bar{O}_{(rem)}^{k-1:kb} \prec_{n_2}^{k-1:kb} \langle v_2, \dots, v_k \rangle \\ \wedge \\ \bar{O}_{(rem)}^{k-1:kb} \prec_{m_1}^{k-1:kb} \bar{\Gamma}_{(rem)}^{k-1:kb} \\ \wedge \\ \bigvee_{\substack{(m_1+1)(m_2-1) \geq (n_3-1) \\ m_2 > 0, m_1, m_2 < n_3}} \left(u_1 \prec_{m_2}^l v_1 \right) \end{array} \right) \quad (345)$$

C.15 Reductions of Open Rigid Tuple Gap Orders

Let $sum = \sum_{i=1}^k u_i^w = \sum_{i=1}^k v_i^w$ and $rem = \sum_{i=2}^k u_i^w = \sum_{i=2}^k v_i^w$. Let $type(u_1) = \alpha_p$ and $type(v_1) = \alpha_{p'}$.

$$\vec{u} \leq_n^{k:w} \vec{v} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \left(\vec{u} \leq_{n_1}^{k:pl} \bar{\Gamma}_{(sum, u_1^w)}^{k:w} \leq_{n_2}^{k:w} \bar{O}_{(sum, v_1^w)}^{k:w} \leq_{n_3}^{k:pl} \vec{v} \right) \quad (346)$$

$$\vec{u} \leq_n^{k:p} \vec{v} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_2>0}} \left(\vec{u} \leq_{n_1}^{k:l} \bar{\Gamma}_{(sum, u_1^w, p)}^{k:p} \leq_{n_2}^{k:p} \bar{O}_{(sum, v_1^w, p')}^{k:p} \leq_{n_3}^{k:l} \vec{v} \right) \quad (347)$$

$$\vec{u} \leq_n^{k:l} \vec{v} \mapsto \bigvee_{\substack{n_1+n_2+n_3=n \\ n_3>0}} \left(\begin{array}{c} \langle u_2, \dots, u_k \rangle \leq_{n_1}^{k-1:kb} \bar{\Gamma}_{(rem)}^{k-1:kb} \\ \wedge \\ \bar{O}_{(rem)}^{k-1:kb} \leq_{n_2}^{k-1:kb} \langle v_2, \dots, v_k \rangle \\ \wedge \\ \bar{O}_{(rem)}^{k-1:kb} \leq_{m_1}^{k-1:kb} \bar{\Gamma}_{(rem)}^{k-1:kb} \\ \wedge \\ \bigvee_{\substack{(m_1+1)(m_2-1) \geq (n_3-1) \\ m_2 > 0, m_1, m_2 < n_3}} \left(u_1 \leq_{m_2}^l v_1 \right) \end{array} \right) \quad (348)$$

D. ACKNOWLEDGMENTS

We thank Aaron Bradley for his comments on an earlier version of this paper. We thank the anonymous referees of CADE'05 for their careful reading and suggestions.

REFERENCES

BAADER, F. AND NIPKOW, T. 1999. *Term Rewriting and All That*. Cambridge University Press, Cambridge, UK.

- BACKOFEN, R. 1995. A complete axiomatization of a theory with feature and arity constraints. *Journal of Logical Programming* 24, 1&2, 37–71.
- COMON, H. 1990. Solving symbolic ordering constraints. *International Journal of Foundations of Computer Science* 1, 4, 387–411.
- COMON, H. AND DELOR, C. 1994. Equational formulae with membership constraints. *Information and Computation* 112, 2, 167–216.
- COMON, H. AND LESCANNE, P. 1989. Equational problems and disunification. *Journal of Symbolic Computation* 7, 371–425.
- COMON, H. AND TREINEN, R. 1994. Ordering constraints on trees. In *Trees in Algebra and Programming - CAAP'94, 19th International Colloquium*, S. Tison, Ed. Lecture Notes in Computer Science, vol. 787. Springer-Verlag, Edinburgh, U.K., 1–14.
- COMON, H. AND TREINEN, R. 1997. The first-order theory of lexicographic path orderings is undecidable. *Theoretical Computer Science* 176, 1-2, 67–87.
- COOPER, D. C. 1972. Theorem proving in arithmetic without multiplication. In *Machine Intelligence*. Vol. 7. American Elsevier, 91–99.
- DERSHOWITZ, N. 1982. Orderings for term-rewriting systems. *Theoretical Computer Science* 7, 279–301.
- ENDERTON, H. B. 2001. *A Mathematical Introduction to Logic*. Academic Press.
- FEFERMAN, S. AND VAUGHT, R. 1959. The first order properties of products of algebraic systems. *Fundamenta Mathematicae* 47, 57–103.
- HODGES, W. 1993. *Model Theory*. Cambridge University Press, Cambridge, UK.
- JOUANNAUD, J.-P. AND OKADA, M. 1991. Satisfiability of systems of ordinal notation with the subterm property is decidable. In *18th International Colloquium on Automata, Languages and Programming*. Lecture Notes in Computer Science, vol. 510. Springer-Verlag, 455–468.
- KIRCHNER, C., KIRCHNER, H., AND RUSINOWITCH, M. 1990. Deduction with symbolic constraints. *Revue Francaise d' Intelligence Artificielle* 4, 3, 9–52. Special issue on automated deduction.
- KNUTH, D. E. AND BENDIX, P. 1970. Simple word problems in universal algebras. In *Computational Problems in Abstract Algebra*. Pergamon Press, 263–297. Reprinted in *Automation of Reasoning, Vol. 2* Jürgen Siekmann and G. Wrightson, editors, pp. 342–376, Springer-Verlag, 1983.
- KOROVIN, K. AND VORONKOV, A. 2000. A decision procedure for the existential theory of term algebras with the Knuth-Bendix ordering. In *Proceedings of 15th IEEE Symposium on Logic in Computer Science* 291 – 302.
- KOROVIN, K. AND VORONKOV, A. 2001. Knuth-Bendix constraint solving is NP-complete. In *Proceedings of 28th International Colloquium on Automata, Languages and Programming (ICALP'01)*. Lecture Notes in Computer Science, vol. 2076. Springer-Verlag, 979–992.
- KOROVIN, K. AND VORONKOV, A. 2002. The decidability of the first-order theory of the Knuth-Bendix order in the case of unary signatures. In *Proceedings of the 22th Conference on Foundations of Software Technology and Theoretical Computer Science, (FSTTCS'02)*. Lecture Notes in Computer Science, vol. 2556. Springer-Verlag, 230–240.
- KUNCAK, V. AND RINARD, M. 2003a. On the theory of structural subtyping. Technical Report MIT-LCS-TR-879, Massachusetts Institute of Technology, Jan.
- KUNCAK, V. AND RINARD, M. 2003b. The structural subtyping of non-recursive types is decidable. In *Proceedings of 18th IEEE Symposium on Logic in Computer Science* IEEE Computer Society Press, 96–107.
- KUNCAK, V. AND RINARD, M. 2005. An algorithm for deciding BAPA: Boolean algebra with Presburger arithmetic. In *the 20th International Conference on Automated Deduction (CADE'05)*. Lecture Notes in Computer Science, vol. 3632. Springer-Verlag, 260–277.
- MAHER, M. J. 1988. Complete axiomatizations of the algebras of finite, rational and infinite tree. In *Proceedings of the 3rd IEEE Symposium on Logic in Computer Science*. IEEE Computer Society Press, 348–357.
- MAL'CEV, A. I. 1971. Axiomatizable classes of locally free algebras of various types. In *The Metamathematics of Algebraic Systems, Collected Papers*. North Holland, Chapter 23, 262–281.
- NARENDHAN, P. AND RUSINOWITCH, M. 2000. The theory of total unary RPO is decidable. In *CL 2000*. Lecture Notes in Artificial Intelligence, vol. 1861. Springer-Verlag, 660–672.

- NARENDRAN, P., RUSINOWITCH, M., AND VERMA, R. M. 1999. RPO constraint solving is in NP. In *Proceedings of the 12th International Workshop on Computer Science Logic (CSL 98)*. Lecture Notes in Computer Science, vol. 1584. Springer-Verlag, 385 – 398.
- NIEUWENHUIS, R. 1993. Simple LPO constraint solving methods. *Information Processing Letters* 47, 2, 65–69.
- NIEUWENHUIS, R. AND RIVERO, J. 1999. Solved forms for path ordering constraints. In *Proceeding of 10th International Conference on Rewriting Techniques and Applications (RTA)*. Lecture Notes in Computer Science, vol. 1631. Springer-Verlag, 1–15.
- NIEUWENHUIS, R. AND RUBIO, A. 1995. Theorem proving with ordering and equality constrained clauses. *Journal of Symbolic Computation* 19, 4, 321–351.
- REDDY, C. R. AND LOVELAND, D. W. 1978. Presburger arithmetic with bounded quantifier alternation. In *Proceedings of the 10th Annual Symposium on Theory of Computing*. ACM Press, 320–325.
- REVESZ, P. 2004. Quantifier-elimination for the first-order theory of boolean algebras with linear cardinality constraints. In *Advances in Databases and Information Systems (ADBIS'04)*. Lecture Notes in Computer Science, vol. 3255. Springer-Verlag, 1–21.
- RYBINA, T. AND VORONKOV, A. 2001. A decision procedure for term algebras with queues. *ACM Transactions on Computational Logic* 2, 2, 155–181.
- SKOLEM, T. 1970. Untersuchungen über die axiome des klassenkalküls und über produktions- und summationsprobleme, welche gewisse klassen von aussagen betreffen. In *Selected works in logic*, J. E. Fenstad, Ed. Universitetsforlaget, 67–101.
- TREINEN, R. 1992. A new method for undecidability proofs of first order theories. *Journal of Symbolic Computation* 14, 437–457.
- ZHANG, T., SIPMA, H. B., AND MANNA, Z. 2004a. Decision procedures for recursive data structures with integer constraints. In *the 2nd International Joint Conference on Automated Reasoning (IJCAR'04)*. Lecture Notes in Computer Science, vol. 3097. Springer-Verlag, 152–167.
- ZHANG, T., SIPMA, H. B., AND MANNA, Z. 2004b. Term algebras with length function and bounded quantifier alternation. In *the 17th International Conference on Theorem Proving in Higher Order Logics (TPHOLs'04)*. Lecture Notes in Computer Science, vol. 3223. Springer-Verlag, 321–336.
- ZHANG, T., SIPMA, H. B., AND MANNA, Z. 2005. The decidability of the first-order theory of term algebras with Knuth-Bendix order. In *the 20th International Conference on Automated Deduction (CADE'05)*, R. Nieuwenhuis, Ed. Lecture Notes in Computer Science, vol. 3632. Springer-Verlag, 131–148.
- ZHANG, T., SIPMA, H. B., AND MANNA, Z. 2006. Decision procedures for term algebras with integer constraints. *Information and Computation* 204, 1526–1574.

Received Month Year; revised Month Year; accepted Month Year