On Ordering Constraints for Deduction with Built-In Abelian Semigroups, Monoids and Groups*

Guillem Godoy and Robert Nieuwenhuis

Technical University of Catalonia, Dept. LSI,
Jordi Girona 1, 08034 Barcelona, Spain
Phone/fax: +34-934017330/7014
E-mail: { ggodoy,roberto }@lsi.upc.es

Abstract. It is crucial for the performance of ordered resolution or paramodulation-based deduction systems that they incorporate specialized techniques to work efficiently with standard algebraic theories E.
Essential ingredients for this purpose are term orderings that are E-*compatible*, for the given E, and algorithms deciding constraint satisfiability for such orderings.
Here we introduce a uniform technique providing the first such algorithms for some orderings for abelian semigroups, abelian monoids and abelian groups, which we believe will lead to reasonably efficient techniques for practice.
The algorithms are optimal since we show that, for *any* well-founded E-compatible ordering for these E, the constraint satisfiability problem is NP-hard even for conjunctions of inequations, and our algorithms are in NP.

Keywords: symbolic constraints, term orderings, automated deduction.

* Both authors are partially supported by the ESPRIT Basic Research Action CCL-II, ref. WG # 22457. and the Spanish CICYT project HEMOSS ref. TIC98-0949-C02-01. The first author is supported by Departament d'Universitats, Recerca i Societat de la Informació de la Generalitat de Catalunya.

1 Introduction

It is crucial for the performance of ordered resolution or paramodulation-based deduction systems that they incorporate specialized techniques to work efficiently with standard algebraic theories E, like abelian semigroups (AC, for associative and commutative) abelian monoids (AC0), or abelian groups (AG).

Essential ingredients for this purpose are *reduction* (i.e., well-founded and monotonic) orderings $\succ$ on ground terms that are *E-compatible* for the given E, i.e., $s =_E s' \succ t' =_E t$ implies $s \succ t$, and algorithms deciding the satisfiability of *ordering constraints* for such orderings. Such ordering constraints are used to express ordered strategies in automated deduction at the formula level [KKR90]. This allows one to reduce the search space by inheriting the ordering restrictions while keeping completeness [NR95,NR00].

An ordering constraint is a quantifier-free first-order formula built over terms in $T(\mathcal{F}, \mathcal{X})$ and over the binary predicate symbols '$=$' and '$>$'. These constraints are interpreted over the domain of ground terms, where $=$ and $>$ are interpreted, respectively, as a congruence $\approx$ and a reduction ordering $\succ$ such that $\succ$ is total up to $\approx$, i.e., for all ground terms s and t either $s \succ t$ or $t \succ s$ or $t \approx s$. Hence a *solution* of a constraint C is a substitution σ with range $T(\mathcal{F})$ and whose domain is the set of variables of C such that $C\sigma$ evaluates to true when interpreting $=$ as $\approx$ and $>$ as $\succ$. Then we say that σ *satisfies* C.

The first practical applications of ordering constraints gave rise to the distinction between *fixed signature* semantics (solutions are built over a given signature $\mathcal{F}$), and *extended signature* semantics (new symbols are allowed to appear in solutions). The latter semantics is in some cases easier to check, and is used in applications like the computation of saturated sets of ordering constrained clauses that can be used for deduction with other clauses containing arbitrary new (e.g., Skolem) symbols, but it is less restrictive and hence less powerful for refutational theorem proving. The satisfiability problem for ordering constraints was first shown decidable for the well-known *recursive path orderings* (RPO) introduced by N. Dershowitz [Der82], for fixed signatures [Com90,JO91] and extended ones [NR95,Nie93]. NP algorithms (fixed and extended signatures) were given in [Nie93,NRV99]. For the Knuth-Bendix ordering (KBO) this result has only recently been obtained (for fixed signatures) in [KV00].

Ordered strategies and ordering constraint inheritance can be used without loosing completeness with built-in algebraic theories E, like AC [NR97,Vig94] or AG [GN00]. An additional advantage of constraints in this context is that in each inference only one conclusion is generated, instead of one conclusion for each E-unifier. This can have dramatic consequences. For example, there are more than a million unifiers in $mgu_{AC}(f(x,x,x), f(y_1,y_2,y_3,y_4))$. But, probably due to the lack of adequate orderings and constraint solving algorithms, these ideas have not been put into practice yet. For example, McCune found his well-known AC-paramodulation proof of the Robbins conjecture [McC97] by still computing complete sets of AC-unifiers, and adding one new equation for each one of them (although heuristics were used to discard some of the unifiers).

Indeed, of the many, rather complex, AC-compatible reduction orderings that have been defined in the literature, only for the AC-RPO ordering of [RN95] a constraint solving algorithm exists [CNR95]. But, unfortunately, this algorithm is far from practical due to its conceptual and computational complexity, and moreover, it only deals with extended signature semantics.

However, in many practical cases one has to deal with only one single associative and commutative symbol, and then a simple version of the RPO on *flattened* terms, which we will call FRPO, fulfills all requirements. The same FRPO can be used as an ingredient for an AG-compatible reduction ordering AG-RPO that satisfies all requirements of [GN00], by using it to compare AG-normal forms of ground terms. Finally, it turns out that an AC0-compatible ordering AC0-RPO is obtained in a similar way by considering normal forms w.r.t. the rule $x + 0 \to x$.

Here we introduce a uniform technique providing the first constraint solving algorithms for fixed signature semantics for AC compatible orderings. More precisely, we give NP algorithms for FRPO-based orderings for abelian semigroups, abelian monoids and abelian groups. We believe that the new techniques will lead to reasonably efficient practical algorithms for these orderings, and give new insights for the development of constraint solving methods over fixed signatures for other E-compatible orderings.

This paper is structured as follows. After the basic definitions of Section 2, in Section 3 we deal with FRPO constraints. For explanation purposes, we start with constraints built with a single unary symbol f, a constant symbol 0 and the AC symbol $+$, and later extend it to arbitrary signatures. After explaining the relatively simple extension to AC0-RPO in Section 4, in Section 5 we deal with the hardest part of the paper, namely the techniques for AG-RPO.

It is obvious that the satisfiability problems we deal with are NP-hard, because as subcases they include the AC, AC0 and AG-unifiability problems which are all NP-hard. As a consequence, since our algorithms are in NP, they are optimal, and the problems are NP-complete. But one may wonder whether there exists any ordering at all for these E such that at least the satisfiability problem for positive conjunctions of inequations (by which one cannot always encode unification) is in P. In Section 6, we answer this question negatively: we show that for *any* well-founded total E-compatible ordering for each one of these E, the problem is NP-hard even for conjunctions of positive inequations.

Finally, in Section 7 we give some conclusions and directions for further work.

2 Basic Definitions

We use the standard notation and terminology for terms and constraints of [DJ90] and [NR99]. The rewrite system R_{AG} consists of the following five rules:

$$\begin{aligned} x + 0 &\to x \\ -x + x &\to 0 \\ -(-x) &\to x \\ -0 &\to 0 \\ -(x+y) &\to (-x) + (-y) \end{aligned}$$

By AG we denote the set of seven equations consisting of these five rules (seen as equations) plus AC, the associativity and commutativity axioms for $+$. By AC0 we mean AC $\cup$ $R_0 = \{x + 0 \to x\}$. By $=_E$ we denote the congruence on terms generated by a set of equations E. In this paper, rewriting with a set of rules R is always considered *modulo* AC, that is, when writing $\to_{R_{AG}}$, we mean the (convergent) relation $=_{\mathrm{AC}} \to_{R_{AG}} =_{\mathrm{AC}}$, and terms will always be considered in *flattened form* w.r.t. AC: we consider e.g. $+(a, b, c)$ instead of $+(a, +(b, c))$. Furthermore, $+$ is written in infix notation: $a + b + c$.

Let us first recall the definition of RPO, which allows for variadic symbols (hence we can cope with flattened terms). We assume given a *precedence* $>$ on $\mathcal{F}$, and, for each $f \in \mathcal{F}$, a *status* which is either *multiset* or *lexicographic*. In the following, a symbol will have the multiset status if, and only if, it is variadic. Below, the relation $=^{\mathrm{mul}}$ has to be understood modulo permutations of the direct subterms of any symbol whose status is multiset. More precisely, for every permutation π, if status(f) = multiset, then, for all terms $t_1, \ldots, t_n$, $f(t_1, \ldots, t_n) =^{\mathrm{mul}} f(t_{\pi(1)}, \ldots t_{\pi(n)})$. Then RPO is defined as follows: $s = f(s_1, \ldots, s_n) \succ_{rpo} g(t_1, \ldots, t_m) = t$ iff

1. $\exists\, i \in \{1, \ldots, n\}$ $s_i \succ_{rpo} t$ or $s_i =^{\mathrm{mul}} t$, or
2. $f > g$ and $s \succ_{rpo} t_i$ for all $i = 1, \ldots, m$ or
3. $f = g$ and status(f) = multiset, and $\{s_1, \ldots, s_n\} \succ_{\mathrm{rpo}}^{\mathrm{mul}} \{t_1, \ldots, t_m\}$ where $\succ_{\mathrm{rpo}}^{\mathrm{mul}}$ is the multiset extension of $\succ_{rpo}$ or
4. $f = g$ and status(f) = lexicographic, and $(s_1, \ldots, s_n) \succ_{\mathrm{rpo}}^{\mathrm{lex}} (t_1, \ldots, t_n)$ where $\succ_{\mathrm{rpo}}^{\mathrm{lex}}$ is the lexicographic extension of $\succ_{rpo}$.

In the following, we call the RPO on flattened terms FRPO: we define $s \succ_{frpo} t$ if *flat*$(s) \succ_{rpo}$ *flat*(t). FRPO is not monotonic in general:

Example 1. If $+ > a > b$ then $b + b \succ_{frpo} a$ but $a + a \succ_{frpo} b + b + a$. Also, if $a > + > f$ then $f(a) + f(a) \succ_{frpo} f(f(a))$ but $f(a) + f(f(a)) \succ_{frpo} f(a) + f(a) + f(a)$. Similar non-monotonicities occur if there is more than one AC symbol. □

However, we have the following (*folk?*) result:

Lemma 1. *If $+$ is the only AC symbol and either $+$ is the smallest symbol in the precedence, or else only the smallest constant is smaller than $+$, then FRPO is an AC-compatible reduction (i.e., monotonic and well-founded) ordering on ground terms that is total up to $=_{\mathrm{AC}}$.*

Let us now define the AC0-RPO and AG-RPO orderings. Given two ground terms s and t, we define

$s \succ_{ac0-rpo} t$ if $s \to_{R_0} s' \succ_{frpo} t' \leftarrow_{R_0} t$,
and
$s \succ_{ag-rpo} t$ if $s \to_{R_{AG}} s' \succ_{frpo} t' \leftarrow_{R_{AG}} t$.

The following is not difficult to prove (see also [GN00]):

Lemma 2. *AC0-RPO (AG-RPO) is an AC0-compatible (AG-compatible) reduction ordering on ground terms that is total up to* $=_{\mathrm{AC}}$ *(*$=_{AG}$*) if* $+$ *is the only AC symbol and the precedence is of the form* $\ldots > + > 0$ *(*$\ldots > - > + > 0$*).*

In the following, we will consider these precedences.

3 FRPO Constraint Solving

For explanation purposes, we present here the simple subcase where the signature contains only $+$, 0, and a unary function symbol f, with the precedence $f > + > 0$.

Let C be an ordering constraint built over f, $+$ and 0, and let T_C be the set of all (sub)terms of C that are: variables, sides of relations $>$ or $=$ in C, terms headed with f, or terms t such that $f(t)$ occurs in C. A *linear constraint* for C is a constraint S of the form

$$t_{1,1} = \ldots = t_{1,k_1} > \ldots > t_{n,1} = \ldots = t_{n,k_n}$$

where all $t_{i,j}$ are distinct and $\{t_{1,1}, \ldots, t_{1,k_1}, \ldots, t_{n,1}, \ldots, t_{n,k_n}\} = T_C \cup \{0\}$. We denote by $=_S$ the equivalence relation generated by the equalities in S and by $>_S$ the smallest strict ordering relation on $T(\mathcal{F}, \mathcal{X})$ compatible with $=_S$ and containing the inequalities of S.

Each constraint C can be expressed as an equivalent (i.e., with the same solutions) finite disjunction of linear constraints S for C (see below); similarly, in what follows we will also make the following assumptions:

A1. W.l.o.g. we can assume S to be of the form

$$x_1 = t_{1,1} = \ldots = t_{1,k_1} > \ldots > x_n = t_{n,1} = \ldots = t_{n,k_n}$$

where $\{x_1, \ldots, x_n\} = \mathrm{vars}(S)$ and all $t_{i,j}$ are distinct non-variable terms. Indeed it is sufficient to insert a new (existentially quantified) variable in each equivalence class without any variables, or to merge two equal variables into one if necessary (merging of equal variables, which will be done more often in this paper, can be recorded separately if one wants to reconstruct a solution for the original constraint rather than to decide its satisfiability).

A2. W.l.o.g. we may assume that each $t_{i,j}$ is either: a sum of variables, or the term 0, or of the form $f(x)$ where x is a variable. This is accomplished by replacing non-variable arguments t by the variable x such that $x =_S t$.

A3. W.l.o.g. we may also assume that in each equivalence class $x_i = t_{i,1} = \ldots = t_{i,k_i}$ either all $t_{i,j}$ are headed by $+$ or else the class is simply $x_i = f(x)$ or $x_i = 0$ or x_i. This is the case since equalities between terms headed with different top symbols are trivially unsatisfiable, and linear constraints (to which the previous transformations have been applied) containing equalities $f(x) = f(y)$ are satisfiable only if x and y are the same variable. The rightmost equivalence class can be assumed to be $x_n = 0$: otherwise S is trivially unsatisfiable.

A4. Again w.l.o.g., for comodity of explanations, S can be assumed to be of the form $x = f(z) > \ldots$. A constraint $x_1 = t_{1,1} = \ldots = t_{1,k_1} > \ldots$ can be transformed by adding an additional leftmost equivalence class into $x_0 = f(x_1) > x_1 = t_{1,1} = \ldots = t_{1,k_1} > \ldots$.

A5. Every variable x occurring as a proper subterm in S can w.l.o.g. be assumed to have another occurrence to the right of it in S at top level (i.e., not as a proper subterm of another term). Otherwise, S is trivially unsatisfiable.

A6. One may assume that if $f(x) >_S f(y)$, then also $x >_S y$. Otherwise, S is again trivially unsatisfiable.

A7. If we have $y_1 + \ldots + y_k \geq_S f(y)$, then, for some i in $1 \ldots k$ we have $y_i \geq_S f(y)$. Otherwise, S is again trivially unsatisfiable.

Example 2. Let the constraint C be $f(x+z) > y \ \wedge \ z > f(x)$. One of its linear constraints is $y = f(x+z) > f(x) > x+z > x = z = 0$. Enforcing the assumptions, it becomes $y = f(w_2) > w_1 = f(x) > w_2 = x + x > x = 0$ by adding new variables w_1 and w_2 for the classes of $f(x)$ and $x+z$ respectively, and merging x and z. However, it is in contradiction with our initial constraint C. Another linear constraint is $f(x+z) > x+z > z = y > f(x) > x = 0$, which becomes $w_1 = f(w_2) > w_2 = x+y > y > w_3 = f(x) > x = 0$. This linear system satisfies all our assumptions and it is not in contradiction with C. □

Lemma 3 ([Com90,Nie93]). *Each constraint C can be expressed as an equivalent finite disjunction of linear constraints satisfying the previous assumptions.*

Proof. By totality of $\succ$ up to $\approx$, for all solutions σ of C and all pairs of terms s and t in T_C either $s\sigma \succ t\sigma$ or $t\sigma \succ s\sigma$ or $s\sigma \approx t\sigma$. Each relation $s > t$ or $s = t$ of C is either included in such a linear constraint S or in contradiction with S. So if $\mathcal{S}$ is the (finite) set of all linear constraints for C, then C is equivalent to the subset of $\mathcal{S}$ containing those S that are compatible in this sense with C. The additional assumptions can be imposed as explained for each one of them. □

3.1 Segments and the splitting transformation

A term u is a *summand* if it is headed with a symbol different from $+$. It is a *top-level summand* of a term t if t is of the form u or $u + t'$. A *segment* T of a linear constraint S is a subsequence of S of the form

$$x_0 = f(s) \ > \ x_1 = t_{1,1} = \ldots = t_{1,k_1} \ > \ \ldots \ > \ x_i = t_{i,1} = \ldots = t_{i,k_i} \ > \ x_{i+1} = t$$

where t is 0 or headed with f and all $t_{i,j}$ are sums of variables. The variables $x_1,\ldots,x_{i+1}$ are said to be the *defined variables* of T, and their occurrences as single variables in their equivalence classes are their *definitions*.

In such a segment T, every variable occurring in some $t_{i,j}$ is defined either in T itself or in some other segment to the right of T. Now our aim is to transform S in such a way that the latter kind of variables are removed from T, while preserving satisfiability. On the other hand, as a result of this transformation, terms $f(v)$ where v is a sum of variables may appear in S.

The idea is as follows. Let σ be some arbitrary solution of S, let x be a variable defined in T, and let y be the variable defined in the equivalence class immediately below x, that is, x is x_j with $1 \leq j \leq i$, and y is x_{j+1}. Then $x\sigma \succ y\sigma \succeq t\sigma$. Therefore, for at least one of the top-level summands u of $x\sigma$ we have $u \succeq t\sigma$. Hence, if U_x is the sum of all top-level summands u of $x\sigma$ with $u \succeq t\sigma$, and u_x is the (possibly empty) sum of the smaller ones, then $x\sigma$ is of the form $U_x + u_x$ or of the form U_x. Similarly, $y\sigma$ can be of the form $U_y + u_y$ or U_y. Furthermore, either (i) $U_x \succ U_y$, or else, if u_x is non-empty, (ii) $U_x = U_y$ and u_y is empty or $u_x \succ u_y$. In the former case, we say that $x > y$ due to the "large" sumands, and in the latter case due to the "small" summands.

According to these ideas, S will be transformed by the following *splitting transformation*, treating one whole segment T at the same time, segment by segment from left to right, except for the rightmost segment, that does not need any treatment. One can assume that in segments T' to the left of T, all variables not below f are defined in T'. Let T be:

$$x_0 = f(s) \; > \; x_1 = t_{1,1} = \ldots = t_{1,k_1} \; > \; \ldots \; > \; x_i = t_{i,1} = \ldots = t_{i,k_i} \; > \; x_{i+1} = t$$

1. Guess a subset of *split* variables of $\{x_1 \ldots x_i\}$ such that whenever $x =_S y_1 + \ldots + y_k$, then x is split if, and only if, at least one of the y_i is split or defined in a segment to the right of T (intuitively, x is split if it is guessed to have at least one "small" summand).
2. If x is a split variable, then introduce two new variables X and x', and everywhere in S replace x by $X + x'$. In this case we say that x is *split* into $X + x'$ (intuitively, the X is for the large summands and the x' for the small ones). If x is a non-split variable of $\{x_1 \ldots x_{i+1}\}$, replace x by a new variable X.
3. After this, the equivalence classes e in the segment are either of the form $V_1 + v_1 = \ldots = V_k + v_k$ or of the form $V_1 = \ldots = V_k$, where the V_i are sums of upper case variables and the v_i are sums of lower case variables and variables defined in segments to the right of T. If e is such an equivalence class, we denote by E the equivalence class $V_1 = \ldots = V_k$ and by e' the class $v_1 = \ldots = v_k$ (if it exists for e). Then we can write T as $x_0 = f(s) > e_1 > \ldots > e_{i+1}$ and we can guess, for each relation $e_j > e_{j+1}$ whether (i) it is due to the large summands or (ii) to the small ones (note that case (ii) applies only if e'_j is non-empty). Accordingly, replace T by the new segment T':
   $$x_0 = f(s) \; > \; E_1 \,\#\, \ldots \,\#\, E_{i+1}$$

Furthermore, insert each e'_j in a segment to the right of T, adding it to an existing equivalence class or creating a new one, in such a way that, whenever $E_j =_{T'} E_{j+1}$, either $e'_j > e'_{j+1}$ or e'_{j+1} does not exist.

This transformation does not increase the number of segments of S and only a polynomial number of variables are split: each variable can only lead to k splittings, where k is the number of segments.

Example 3. (Example 2 continued) Let us apply the splitting transformation to the result $w_1 = f(w_2) > w_2 = x + y > y > w_3 = f(x) > x = 0$ of Example 2. First we treat the leftmost segment $w_1 = f(w_2) > w_2 = x + y > y > w_3 = f(x)$. The possible variables to be split are w_2 and y. We guess to split only w_2 into $W_2 + w'_2$, obtaining $w_1 = f(W_2 + w'_2) > W_2 + w'_2 = x + y > y > w_3 = f(x)$. Now, for the relation $W_2 + w'_2 > y$ we guess $W_2 = y$. After removing w'_2 from this segment and inserting it, for example, in the equivalence class of 0, we obtain $w_1 = f(y + x) > y > w_3 = f(x) > x = 0$. For the segment $w_3 = f(x) > x = 0$ no splitting is needed. □

If $x >_S y$ and S' is obtained from S by the splitting transformation, then the occurrences of $f(x)$ and $f(y)$ in S become $f(X + X' + X'' + \ldots)$ and $f(Y + Y' + Y'' + \ldots)$ in S', respectively, where $X, X', X'', \ldots$ are in different segments, with $X >_{S'} X' >_{S'} X'' >_{S'} \ldots$ and the same happens for the $Y, Y', Y'', \ldots$ Furthermore, we have either (i) $X >_{S'} Y$, or $X =_{S'} Y$ and $X' >_{S'} Y'$, or $X =_{S'} Y$ and $X' =_{S'} Y'$ and $X'' >_{S'} Y''$, etc.; or, otherwise, we have (ii) $X >_{S'} Y$ and x becomes $X + X' + \ldots$ and y becomes Y, or $X =_{S'} Y$ and $X' =_{S'} Y'$ and x becomes $X + X' + X'' + \ldots$ and y becomes Y or $Y + Y'$, etc. Whenever for two sums of variables $X + X' + X'' + \ldots$ and $Y + Y' + Y'' + \ldots$ this situation applies, we say that they are *compared by segments in S'*.

3.2 Diophantine systems

Assume S is the result of applying the splitting transformation to a linear system. Now we can define a system of diophantine equations and inequations D_S for S as follows. For each segment T in S of the form

$x_0 = f(s) \; > \; x_1 = t_{1,1} = \ldots = t_{1,k_1} \; > \; \ldots \; > \; x_i = t_{i,1} = \ldots = t_{i,k_i} \; > \; x_{i+1} = t$

the system D_S contains the equations and inequations:

1. $x_1 > x_2,\ x_2 > x_3,\ \ldots,\ x_i > x_{i+1}$
2. $x_j = t_{j,k}$, for all j in $\{1 \ldots i\}$, and all k in $\{1 \ldots k_j\}$
3. the equation $x_{i+1} = 1$.

Example 4. (Example 3 continued) The system of diophantine equations for $w_1 = f(y + x) > y > w_3 = f(x) > x = 0$ is

$$w_1 = 1 \quad y > w_3 \quad w_3 = 1 \quad x = 1$$

We obtain a solution θ for it by defining $y\theta = 2$. Below we will see that from each such a θ one can build a solution σ for the linear constraint from right to left.

We have $x\sigma = 0$ and hence $w_3\sigma = f(0)$. Now for each variable v with $v\theta = n$, we define $v\sigma = t + \ldots^{n)} + t$, where t is the summand at the lower end of its segment; e.g., we define $y\sigma$ to be $f(0)+f(0)$. Finally, we have $w_1\sigma = f(f(0)+f(0)+0)$. If one desires to reconstruct the solution for the original constraint of Example 2: $w_2'\sigma$ is 0, and $z\sigma$ is $f(0)+f(0)$. □

The following simple result will be used below when solving ordering constraints on multisets of several elements as multisets over a single element:

Lemma 4. *Let C be a set $\{e_n, \ldots, e_0\}$ with an ordering $\succ$ where $e_n \succ \ldots \succ e_0$. Then for any decreasing sequence of finite multisets over C*

$$M_0 \ggcurly \ldots \ggcurly M_m$$

there exists a weighting function $f : C \to \mathcal{N}$ with $f(e_0) = 1$ such that

$$F(M_0) > \ldots > F(M_m)$$

where the extension to multisets F of f is defined $F(\{a_1 \ldots, a_k\}) = f(a_1) + \ldots + f(a_k)$.

Proof. Let k be $n_0 + \ldots + n_m$. Then, for instance, the function $f(e_i) = k^i$ fulfills the requirements. □

Lemma 5. *Let $S_1 \ldots S_m$ be the resulting systems of applying the splitting transformation to a linear constraint S over the signature $f > + > 0$. Then S is satisfiable for FRPO if, and only if, some D_{S_i} is satisfiable in the positive natural numbers.*

Proof. $\Longleftarrow$: Assume $D_{S'}$ is satisfiable for some S' in $\{S_1 \ldots S_m\}$. Let θ be a solution for $D_{S'}$. We can build a solution σ for S' as follows. For each segment T in S of the form

$$x_0 = f(s) \;>\; x_1 = t_{1,1} = \ldots = t_{1,k_1} \;>\; \ldots \;>\; x_i = t_{i,1} = \ldots = t_{i,k_i} \;>\; x_{i+1} = t$$

assume a (partial) solution σ has already been defined for all segments to the right of T. Then, for the variables x_j defined in this segment we define $x_j\sigma$ to be $t\sigma + \ldots^{n)} + t\sigma$ where $n = x_j\theta$ (note that if T is the rightmost segment, then t is 0). Clearly, σ satisfies all equality relations in S', that is, $u\sigma =_{\mathrm{AC}} v\sigma$ for all u and v with $u =_{S'} v$. Furthermore, it also satisfies the relations $x_j\sigma \succ x_{j+1}\sigma$ with j in $\{1 \ldots i\}$ for such segments T.

Hence it only remains to be checked that σ satisfies $f(s)\sigma \succ x_1\sigma$. Since $x_1\sigma$ is of the form $t\sigma + \ldots + t\sigma$ and $f > +$, it suffices to check that $f(s)\sigma \succ t\sigma$, where t is headed with f (the case where t is 0 is trivial). Then $f(s)$ is of the form $f(X + X' + X'' + \ldots)$ and t is of the form $f(Y + Y' + Y'' + \ldots)$, as a result of the splitting transformation applied to terms $f(x)$ and $f(y)$.

But by assumption A6, if $f(x) >_S f(y)$, then also $x >_S y$. Therefore, our result follows: after the splitting transformation, the sums $X, X', X'', \ldots$ and $Y, Y', Y'', \ldots$ are compared by segments in S', and σ assigns one different summand to each segment, and in the correct order.

Once we have this solution σ for S', it can be extended to a solution for S by recursively defining $x\sigma$ to be $X\sigma + x'\sigma$, for each splitting of a variable x into $X + x'$.

$\Longrightarrow$: Assume S is satisfiable. Now we prove that $D_{S'}$ is satisfiable as well for some S' in $\{S_1 \ldots S_m\}$. Let σ be a solution of S. Let S' be the system obtained by applying the splitting transformation according to σ, that is, if x is defined in a segment T of S of the form

$x_0 = f(s) \ > \ x_1 = t_{1,1} = \ldots = t_{1,k_1} \ > \ \ldots \ > \ x_i = t_{i,1} = \ldots = t_{i,k_i} \ > \ x_{i+1} = t,$

then x is split into $X + x'$ if $x\sigma$ contains any summands smaller than $t\sigma$; we proceeed similarly for the other guessings, and σ is extended conveniently for the new variables. The extended substitution σ is a solution for S'. Moreover, in a segment of S' like the previous one, for all j in $\{1 \ldots i+1\}$ we have that $x_j\sigma$ contains only top-level summands greater than or equal to $t\sigma$.

Now let $C = \{u_0, \ldots, u_n\}$ be all the different top-level summands of these variables, where $u_n \succ u_{n-1} \succ \ldots \succ u_0$ and u_0 is $t\sigma$. Every $x_j\sigma$ and $t_{j,l}\sigma$ can be seen as a multiset on these summands (the multiset of its top-level summands). By Lemma 4 there exists a function $f : C \to \mathcal{N}$ such that its extension F to multisets satisfies $F(x_1\sigma) \ggcurly \ldots \ggcurly F(x_{i+1}\sigma)$, and $F(x_{i+1}\sigma) = f(u_0) = 1$. Moreover, since $x_j\sigma$ and $t_{j,l}\sigma$ are the same multiset, if $t_{j,l}$ is of the form $x_{j_1} + \ldots + x_{j_l}$, then $F(x_j\sigma) = F(t_{j,l}\sigma) = F(x_{j_1}\sigma) + \ldots + F(x_{j_l}\sigma)$. Therefore, the assignment $x_j = F(x_j\sigma)$ satisfies the equations of $D_{S'}$ corresponding to T. $\square$

Theorem 1. *The satisfiability problem for FRPO constraints over the signature* $f > + > 0$ *is in NP.*

Proof. Generating one of the linear constraints S of the disjunction equivalent to C consists of a polynomial number of guessings of the relations between all the subterms in C, and the size of S is polynomial w.r.t. the size of C. The splitting transformation consists of a polynomial number of guessings. By Lemma 5 S is satisfiable if and only if there exists a sequence of guessings, in the splitting transformation, giving a linear constraint S', such that $D_{S'}$ is satisfiable. Checking whether $D_{S'}$ is satisfiable is again in NP [Sch87]. $\square$

3.3 More function symbols

We consider now the case where the signature contains any finite number of function symbols with arbitrary arities. The precedence is now of the form $\ldots > + > 0$.

W.l.o.g., the following additional assumptions w.r.t. the linear constraint generated from the initial constraint may be assumed (otherwise the linear constraint is again trivially unsatisfiable):

A8. If $f(x_1, \ldots, x_n) >_S g(y_1, \ldots, y_m)$ and $g > f$, then $x_i \geq_S g(y_1, \ldots, y_m)$ for some i in $1 \ldots n$.

A9. If $f(x_1, \ldots, x_n) >_S f(y_1, \ldots, y_n)$ then either $x_i \geq_S f(y_1, \ldots, y_n)$ for some i in $1 \ldots n$ or else $(x_1, \ldots, x_n) >_S^{lex} (y_1, \ldots, y_n)$.

Segments are defined as before, except that now the function symbols at the begining and at the end of it may be different: a segment T of a linear constraint S is a subsequence of S of the form

$$x_0 = s \ > \ x_1 = t_{1,1} = \ldots = t_{1,k_1} \ > \ \ldots \ > \ x_i = t_{i,1} = \ldots = t_{i,k_i} \ > \ x_{i+1} = t$$

where s and t are not headed with $+$ and all $t_{i,j}$ are sums of variables. The splitting transformation and the diophantine system are defined exactly as before.

Lemma 6. *Let $S_1 \ldots S_m$ be the resulting systems of applying the splitting transformation to a linear constraint S. Then S is satisfiable if, and only if, some D_{S_i} is satisfiable.*

Proof. The left-to-right implication is identical to the one in the proof of Lemma 5. For the right-to-left implication, the substitution σ is also created from the diophantine solution θ in the same way; the only difference is the justification that $f(s_1, \ldots, s_n)\sigma \succ t\sigma$, where $f(s_1, \ldots, s_n)$ is the leftmost term in the segment and t is the rightmost one. One shows that $f(s_1, \ldots, s_n)\sigma \succ t'\sigma$, where t' is t or any other term to the right of t. This is done inductively from right to left. By induction hypothesis, we can also assume that σ is a solution for all relations between terms to the right of $f(s_1, \ldots, s_n)$.

The result is trivial if t' is 0. Otherwise, let t' be of the form $g(t_1, \ldots, t_m)$, and let $f(x_1, \ldots, x_n)$ and $g(y_1, \ldots, y_m)$ be the terms in S corresponding to the terms $f(s_1, \ldots, s_n)$ and $g(t_1, \ldots, t_m)$ in S'. Note that the argument terms s_i and t_i are sums of variables.

1. Assume $f > g$. Then, the result follows by definition of RPO since $f > g > +$ and by induction hypothesis, $f(s_1, \ldots, s_n)\sigma \succ y\sigma$ for all variables y occurring in $g(t_1, \ldots, t_m)$.
2. Assume $g > f$. Then, by assumption A8, $x_i \geq_S g(y_1, \ldots, y_m)$ for some i in $1 \ldots n$. Moreover, this relation is preserved for σ in the part of S' for which σ is a solution by induction hypothesis. Therefore $f(s_1, \ldots, s_n)\sigma \succ t'\sigma$.
3. Finally we consider the case $f = g$. By assumption A9 we have that either $x_i \geq_S g(y_1, \ldots, y_n)$ for some i in $1 \ldots n$ or else $(x_1, \ldots, x_n) >_S^{lex} (y_1, \ldots, y_n)$. In first case we conclude in a similar way as for the case $g > f$. In the seccond case, some x_i satisfies that $x_i >_S y_i$ and $x_j = y_j$ for all j such that $j < i$. Therefore s_i and t_i are of the form $X + X' + \ldots$ and $Y + Y' + \ldots$ respectively, and this sums are compared by segments in S'. Therefore we obtain $s_i\sigma \succ t_i\sigma$. Moreover, by induction hypothesis we have that $f(s_1, \ldots, s_n)\sigma \succ t_j\sigma$ for all j in $1 \ldots n$. □

Theorem 2. *The satisfiability problem for FRPO constraints is in NP.*

4 AC0-RPO Constraints

In this section we consider AC0-RPO constraints over arbitrary signatures of the form $\ldots > f > + > 0$. Observe that all terms of the form $0 + \ldots + 0$ are equivalent to 0 in this setting and that hence the second smallest term w.r.t. the ordering $\succ$ is $f(0, \ldots, 0)$. Therefore we can add, w.l.o.g., an aditional assumption to our linear constraints:

A10 All linear constraints S are of the form $S' > x = f(y, \ldots, y) > y = 0$ and no term of the form $t + y$ occurs in S.

With this additional assumption, it is easy to see that the whole rest of the steps described in the previous section directly suffice for AC0-RPO constraints. Minor details are that, during the splitting process, the new assumption A10 has to be preserved, and then, no small variables resulting from a splitting can be inserted in the rightmost segment. Moreover, in the diophantine system it is not necessary to create the equations corresponding to the rightmost segment.

Observe that the basic idea of the splitting process is that solutions for the linear constraint are transformed into new solutions where, at every segment, the variables that appear in it contain only top-level summands of this segment. Therefore, 0 does not appear in segments that are not the rightmost one, and hence everything behaves like in the FRPO case, again solving the diophantine equations over the positive natural numbers. This gives us the following result.

Theorem 3. *The satisfiability problem for AC0-RPO constraints is in NP.*

5 AG-RPO Constraints

In this section we consider AG-RPO constraints over arbitrary signatures of the form $\ldots > - > + > 0$. In this context summands are terms headed with some symbol different from 0, $+$ or $-$.

Let us first consider some examples over the signature $f > a > - > + > 0$ where f is unary and a is a constant.

Example 5. Then the smallest terms over this signature in increasing order w.r.t. $\succ$ are:

$0,\ a,\ a{+}a,\ a{+}a{+}a,\ \ldots,\ {-a},\ {-a}{-}a,\ {-a}{-}a{-}a,\ \ldots,\ f(0),\ f(0){+}a$
$f(0){+}a{+}a,\ \ldots,\ f(0){-}a,\ f(0){-}a{-}a,\ \ldots,\ f(0){+}f(0),\ f(0){+}f(0){+}a,\ \ldots,\ {-f(0)}$
where $-a$ is the smallest limit ordinal ω, $f(0)$ is 2ω, $f(0) - a$ is 3ω, $f(0) + f(0)$ is 4ω, $-f(0)$ is ω^2, and $f(a)$ is $2\omega^2$. □

Example 6. We have $f(f(a)) \succ f(a - f(0) + f(a - a))$ since
$nf_{R_{AG}}(f(f(a))) = f(f(a)) \succ_{FRPO} f(a) = nf_{R_{AG}}(f(a - f(0) + f(a - a)))$. □

Example 7. Terms can be smaller than their subterms: $\sigma \models x > f(x - f(a))$ if $x\sigma = f(a)$, since $nf_{R_{AG}}(f(a)) = f(a) \succ_{FRPO} f(0) = nf_{R_{AG}}(f(f(a) - f(a)))$. □

Since, as we have seen in the previous example, a linear constraint such that x appears to the right of the segment where it is defined may be satisfiable, assumption A5 will not be made in this section. Similarly, the following example shows us that terms headed with f may become equal to terms headed with $+$ or $-$. Hence assumption A3 is also dropped in this section:

Example 8. $\sigma \models x - y = f(z)$ if we have $x\sigma = f(a) + f(a)$, $y\sigma = f(a)$, $z\sigma = a$. □

An other difficulty to be taken into account is that, after the splitting transformation, contrarily to what happened in the previous sections, a solution for a linear constraint may need more than one different top-level summand for some segments:

Example 9. Suppose that we have a signature of the form $f > - > + > 0$ where f is unary. Then the smallest terms are ordered like:

$0, \quad f(0), \quad f(0)+f(0), \quad f(0)+f(0)+f(0), \quad \ldots,$

$-f(0), \quad -f(0)-f(0), \quad -f(0)-f(0)-f(0), \quad \ldots, \quad f(f(0)).$

The linear constraint $f(f(0)) > -z > z > y > -y > f(0)$ is unsatisfiable: since we need to satisfy $y > -y$, necessarily $y\sigma$ is a sum of negative $f(0)$'s. Therefore $z\sigma$ is of the form $-f(0) - \ldots - f(0)$, with some more negative $f(0)$'s. But then $-z > z$ is not satisfied by σ.

However, the linear constraint $f(f(f(0))) > -z > z > y > -y > f(0)$ has the solution σ where $y\sigma = -f(0) - f(0)$ and $z\sigma = f(f(0)) + f(f(0))$. It has no solution where $y\sigma$ and $z\sigma$ are built from one single summand. □

5.1 Only unary symbols

For explanation purposes, in this subsection we first assume that all the non-constant function symbols have arity one. Our signature is of the form $\ldots > h > c_1 > \ldots > c_l > - > + > 0$, where h is the smallest non-constant function symbol, i.e., all the c_i are constants.

Then we have the following ordering on summands (from which the ordering on ground terms is easily derived). If $l = 0$ then the smallest summands are, in increasing order: $h(0)$, $h(h(0))$, $h(h(h(0)))$,... If $l \neq 0$ then the smallest summands are, in increasing order: c_l, $\ldots$, c_1, $h(0)$, $h(c_l)$, $h(c_l + c_l)$, $h(c_l + c_l + c_l)$,.... These summands will be denoted by $sum_1, sum_2, sum_3, \ldots$

Note that the successor summand of a summand of the form $h(s)$ is $h(s + sum_1)$ if s is not of the form $s' - sum_1$, and $h(s - sum_1)$ otherwise. The successor summand of a summand $f(s)$ with $f > h$ is always $h(f(s))$. We write $succsum_k(u)$ to denote the k-th successor summand of u.

Conditions for the linear constraints. As before, we generate a disjunction of linear constraints, and apart from the assumptions $A1 - A9$, except, as said, A3 and A5, we need:

- A11. W.l.o.g. one can assume that all the constants c_i and the terms sum_1, sum_2 and $h(0)$ appear in S, and in the correct order. We will refer to the segment between sum_2 and sum_1 as the *base segment.*
- A12. If a segment starts with $x = h(y)$ and ends with $f(z)$ with $f > h$ then y is defined in this segment.
- A13. All linear constraints are of the form $S' > x = sum_1 > y = t_1 = \ldots = t_k = 0$ and no term of the form $t + y$ occurs in S.

Condition A13 is a modification of condition A10. Now, in the class of 0 sums of variables defined to the left of it may appear. In a solution for the constraint, these variables will contain summands that cancel each other out.

In this setting, a sum of variables is, in fact, a sum of positive and negative variables, and all assumptions have to be interpreted accordingly. For example, condition A7 implies that no term of the form $-x$ is in a segment to the left of the segment where x is defined.

The splitting transformation. The splitting transformation is essentially as before, with some differences. For example, when we guess that some relation is due to the small summands, the small terms cannot be inserted in the class of 0. Therefore, it makes no sense to do any splitting of variables in the base segment. Another difference with the previous cases is that after splitting and removing small variables from a segment T, some variables defined in T may appear to the right of T. For this reason, we need to introduce the so-called *associated* equations, a set of equations asociated to each segment, but that is not inserted in the linear constraint. During the splitting transformation, just after removing the small variables of a segment T, equations are associated to T as follows. Let s be a term in an equivalence class to the right of T, and let s be is of the form $M+m$ or $f(M+m)$, where M is a sum of positive and negative variables defined in T, and m does not contain any of these variables. Then clearly in any solution σ the term $M\sigma$ must be equivalent to 0. Therefore, for each such s, the part M is removed from s, and $M=0$ becomes an associated equation of T (if the part m of s is empty, then M is replaced by x, the variable of the class of 0).

Finally, for explanation purposes, we want the rightmost class of each T to be of the form $x=t$, for some term t not headed by $+$. This can be accomplished as follows. Assume after splitting, this class is of the form $x = T_i + t'_1 = \ldots = T_l + t'_l = t$, where the T_i are the "large" sums, i.e., the sums of the positive and negative variables defined in T. Then the class $t'_1 = \ldots = t'_l$ necessarily has to be inserted in the class of 0. Furthermore, the T_i's are removed as well, and the equations $x - T_i = 0$ are added as additional associated equations of T.

By processing the segments in this manner, from left to right, when we arrive to the segment containing the class of 0, it is of the form $x = sum_1 > x = 0$, since the rest of variables cannot appear in this segment, at this point.

Example 10. Let us consider the signature $h > - > + > 0$. Suppose during the splitting transformation just after splitting the variables of the leftmost segment we obtain:

$$z = h(x_3) > x_3 > x_2 > x_1 > x_0 = x_3 - x_2 - x_1 + y_2 - y_1 - y_1 = h(y_1) >$$
$$y_3 = x_2 - x_1 - x_0 + y_2 + y_1 > y_2 > y_1 > h(w) > w = 0.$$

At this point, if we assume that this splitting of variables has been done according to a solution σ, then, all the $x_i\sigma$ contain top-level summands bigger than or equal to $h(y_1)\sigma$, and all the $y_i\sigma$ contain top-level summands smaller than $h(y_1)\sigma$. Since $(x_3 - x_2 - x_1 + y_2 - y_1 - y_1)\sigma$ must coincide with $h(y_1)\sigma$, the summands below the $y_i\sigma$'s must cancel each other, i.e. $(y_2 - y_1 - y_1)\sigma$ must be 0. Therefore, we remove

$y_2-y_1-y_1$ from the sum $x_3-x_2-x_1+y_2-y_1-y_1$, and add it to the class of 0, obtaining:

$z = h(x_3) > x_3 > x_2 > x_1 > x_0 = x_3-x_2-x_1 = h(y_1) >$
$y_3 = x_2-x_1-x_0+y_2+y_1 > y_2 > y_1 > h(w) > w = y_2-y_1-y_1 = 0$

Now, in order to leave the treated segment in a normalized form $x_0 = h(y_1)$, we remove the $x_3-x_2-x_1$ and we add $x_0-x_3+x_2+x_1 = 0$ to the set of associated equations of this segment.

Finally, since the term $x_2-x_1-x_0+y_2+y_1$ is to the right of $h(y_1)$, and hence it must contain only summands smaller than $h(y_1)\sigma$, we have to force the x_i's to cancel each other. We remove $x_2-x_1-x_0$ and we add $x_2-x_1-x_0 = 0$ to the associated equations of the leftmost segment. Note that this is a different treatment with respect to what was done with $y_2-y_1-y_1$ before. But remember that the aim is to remove variables of the treated segment from the other segments to the right of it. In fact, this $y_2-y_1-y_1$ added to the class of 0 will be removed from this class when we will treat the next segment, since none of the y_i's is defined in the rightmost segment.

Just after finishing the treatment of the leftmost segment we obtain:

$z=h(x_3) > x_3 > x_2 > x_1 > x_0=h(y_1) >$
$y_3=y_2+y_1 > y_2 > y_1 > h(w) > w=y_2-y_1-y_1=0$

where the leftmost segment contain the associated equations $x_0-x_3+x_2+x_1 = 0$ and $x_2-x_1-x_0 = 0$. □

Diophantine equations. Example 9 shows that now in solutions more than one summand may be needed in a single segment. But only a certain number of summands play an important role in the comparisons.

Example 11. If $a > b > c$, in the inequation $a+a+a+b+b+c \succ a+a+a-c-c-c$ the summand b will be called the *decisive* summand, since it is the greatest sumand that appears in both terms with a different number of occurrences. □

Let s be a term and u a summand. The *number of occurrences* of u in s (notation $\#(u, s)$) is the integer n such that $s =_{AG} nu+s'$, where u is not a top-level summand of s'. For instance $\#(a, f(a+b)-a-a) = -2$. Let s and t be two ground terms such that $s \succ t$. The *decisive summand* of the inequation $s \succ t$ is the top-level summand u such that for all summands $v \succ u$, $\#(v, s) = \#(v, t)$, and either (i) $\#(u, s) > \#(u, t) \geq 0$ or (ii) $\#(u, s) < \#(u, t)$ and $\#(u, s) < 0$.

Once the splitting transformation has been applied to S, we can define a system of diophantine equations and inequations D_S for S as follows. For each segment T in S of the form

$x_0=s \;>\; x_1=t_{1,1}=\ldots=t_{1,k_1} \;>\; \ldots \;>\; x_i=t_{i,1}=\ldots=t_{i,k_i} \;>\; x_{i+1}=t$

with associated equations $q_1 = 0, \ldots, q_l = 0$, several guessings are necessary. First, the number *ndec* of the segment is guessed. Intuitively, for a given σ, the number *ndec* is the cardinality of $dec(T\sigma)\cup\{t\sigma\}$, where $dec(T\sigma)$ is the number of different decisive summands in the inequations $x_j\sigma \succ x_{j+1}\sigma$ with $j > 0$. Hence one can guess *ndec* to be between 1 and $i+1$. There are some cases where it must be exactly 1, which is when we know that for all σ we have $s\sigma = succsum_1(t\sigma)$:

- s is some c_j and t is c_{j+1}, or
- t is c_1 and s is $h(0)$, or
- t is sum_1 and s is sum_2, or
- t is headed with some f with $f > h$ and s is $h(x_{i+1})$.

In the following, the elements of $dec(T\sigma) \cup \{t\sigma\}$ are denoted (and ordered) by $u_{ndec} \succ \ldots \succ u_1$. Note that always $t\sigma$ is u_1 (if the splitting has been done according to σ).

Now, for every variable x_j with $1 \leq j \leq i+1$ we create $ndec$ integer variables $x_{j,1}, \ldots, x_{j,ndec}$, representing the number of occurrences of each decisive summand in x_j. For the segments where $ndec$ is 1 (as for the base segment) we preserve the same variable name x_j for the corresponding integer variable.

Example 12. Consider $f > h > - > + > 0$ and suppose that after the splitting transformation we have:

$$z_4 = h(w_1+x_2) > w_6 = -w_5 > w_5 > w_4 = -w_3 > w_3 > w_2 = -w_1 > w_1 = f(z_3) >$$
$$z_3 = h(x_3) > y_4 = -y_3 > y_3 > y_2 = -y_1 > y_1 = h(x_2) >$$
$$z_2 = h(x_1) > x_3 > x_2 > x_1 = h(z_1) > z_1 = 0$$

Now, we want to find a solution σ such that for every variable it contains summands greater than or equal to the rightmost term of the segment where it is defined. We may guess that the number of decisive summands for the leftmost segment is 3. Therefore, we need to guarantee that at least two summands between $f(z_3)\sigma$ and $h(w_1+x_2)\sigma$ exist exist. Observe that the successor summand of $f(z_3)\sigma$ is $h(f(z_3))\sigma$ and the next one is $h(f(z_3)+h(0))\sigma$. Since x_2 is a variable in the base segment, we need $x_2\sigma$ to be at least $h(0)+h(0)$. Here appears the need of adding, to the diophantine system, either an equation of the form $x_2 \geq 2$ or one of the form $x_2 < 0$, since $-f(0)$ is greater than any sum of positive $f(0)$'s.

Later on, we may decide that the number of decisive summands for the segment $z_3 = h(x_3) > y_4 = -y_3 > y_3 > y_2 = -y_1 > y_1 = h(x_2)$ is 2. We need to guarantee that there exists at least one summand between $h(x_3)\sigma$ and $h(x_2)\sigma$. Observe that x_3 and x_2 are defined in the base segment. If we guess $x_2\sigma$ to be $h(0)+\ldots+h(0)$, then either $x_3\sigma$ is also of the form $h(0)+\ldots+h(0)$ with at least two more $h(0)$'s than $x_2\sigma$, or $x_3\sigma$ is of the form $-h(0)-\ldots-h(0)$. If we guess that $x_2\sigma$ is $-h(0)-\ldots-h(0)$, then $x_3\sigma$ also has to be $-h(0)-\ldots-h(0)$, but with at least two more $-h(0)$'s than $x_2\sigma$. □

We now impose some more diophantine equations ensuring that there will be enough space for the decisive summands between $s\sigma$ and $t\sigma$, when $ndec > 1$. Assume $ndec > 1$ and let y and z be variables defined in the base segment:

1. If s is of the form $h(y+s')$ and t is of the form $h(z+s')$, it has to be guessed whether one adds either the equations (i) $y \geq z+ndec$ and $z \geq 0$, or the equations (ii) $y \leq z-ndec$ and $z < 0$, or the equations (iii) $y < 0$ and $z \geq 0$.
2. If s is of the form $h(y+s')$ and t is of the form $h(s')$, there is another choice between the equation (i) $y \geq ndec$, and the equation (ii) $y < 0$.
3. If s is of the form $h(x_{i+1}+y)$ and t is of the form $f(t')$, either the equation (i) $y \geq ndec-1$ or (ii) $y < 0$ is added.

The following equations are added to the system D_S in order to express for which inequation which decisive summand is decisive, and whether it decides positively or negatively:

1. For each j between 1 and i, we guess which index summand k between 1 and $ndec$ is the decisive one for the inequation $x_j > x_{j+1}$. Now, for all $k' > k$ we add the equation $x_{j,k'} = x_{j+1,k'}$. In order to decide in which manner the k-th summand is decisive, we guess adding either (i) $x_{j,k} > x_{j+1,k} > 0$ or (ii) $x_{j,k} < x_{j+1,k}$ and $x_{j,k} < 0$.
2. Let $t_{j,l}^k$ be the result of replacing in $t_{j,l}$ every variable $x_{j'}$ by $x_{j',k}$, the integer variable corresponding the the k-th decisive summand. Now in order to make sure that the number of occurrences of the k-th summand at each side of the equality coincides, add $x_{j,k} = t_{j,l}^k$, for all j in $\{1 \ldots i\}$, and all k in $\{1 \ldots ndec\}$, and all l in $\{1 \ldots k_j\}$. We proceed identically with the associated equations.
3. We add $x_{i+1,1} = 1$, and for all k in $\{2 \ldots ndec\}$ we add $x_{i+1,k} = 0$.

Constraint Solving.

Lemma 7. *Let $D_1 \ldots D_m$ be the diophantine systems generated from all the linear constraints obtained by applying the splitting transformation to a linear constraint S. Then S is satisfiable iff some D_i is satisfiable over the integers.*

Proof. $\Longrightarrow$: As for the AC case, assuming that S is satisfiable by some substitution σ, we find some S' resulting from a splitting transformation of S according to σ, and some extension of σ such that σ is a solution of S'; and, moreover, given a segment T of S' of the form

$$x_0 = s \;>\; x_1 = t_{1,1} = \ldots = t_{1,k_1} \;>\; \ldots \;>\; x_i = t_{i,1} = \ldots = t_{i,k_i} \;>\; x_{i+1} = t,$$

for all j in $\{1 \ldots i+1\}$ we have that all the $x_j\sigma$ contain only summands greater than or equal to $t\sigma$.

Consider the set $C = \{u_1, \ldots, u_n\}$ including $t\sigma$ and all the different top-level summands of these variables that are decisive in some inequation $x_j\sigma \succ x_{j+1}\sigma$, for some j in $\{1, \ldots, i\}$, and suppose that $u_n \succ \ldots \succ u_1$. Let D' be the diophantine system obtained from S' by doing the guessings according to σ: the chosen $ndec$ is precisely the cardinality n of C for such segments T, and the the chosen index for the generated equations from the inequation $x_j > x_{j+1}$ is k if u_k is the decisive summand in $x_j\sigma \succ x_{j+1}\sigma$, etc. Now, if for such segments T we assign $\#(u_k, x_j\sigma)$ to each variable $x_{j,k}$, then we obtain a solution for D'.

$\Longleftarrow$: Assume that some guessed $D_{S'}$ is satisfiable for some resulting S' from a splitting transformation of S. Let θ be a solution for $D_{S'}$. We build a solution σ for S' as follows. For each segment T in S of the form

$$x_0 = s \;>\; x_1 = t_{1,1} = \ldots = t_{1,k_1} \;>\; \ldots \;>\; x_i = t_{i,1} = \ldots = t_{i,k_i} \;>\; x_{i+1} = t$$

assume $ndec$ has been guessed for T and that σ has been defined for all segments to the right of T. Then, $t\sigma$ is already defined. Then, for all j in $\{1, \ldots, i+1\}$ we define $x_j\sigma$ in this manner: for all k in $\{1, \ldots, ndec\}$, we force $\#(succsum_k(t\sigma), x_j\sigma)$ to be $x_{j,k}\theta$, and $\#(u, x_j\sigma) = 0$ for any other summand u. Clearly, σ satisfies all equality relations in S', and all the relations $x_j \succ x_{j+1}$ with j in $\{1,\ldots,i\}$ for such segments T, and it remains to check that σ satisfies $s\sigma \succ x_1\sigma$. This is done by distinguishing some cases:

- For the cases where $ndec$ is necessarily 1, clearly $s\sigma$ is greater than a sum of positive or negative $t\sigma$'s, and then it is greater than $t'\sigma$ for all t' to the right of t.
- Let s and t be of the form $h(s')$ and $h(t')$, respectively, and T is not the base segment. By our assumptions and the way of doing the splitting process, we have that s' and t' are of the form $X+X'+\ldots$ and $Y+Y'+\ldots$ respectively, and these sums are compared by segments in S'. Then, either (a) there exists a pair (X_d, Y_d) that it is the leftmost one of $(X,Y),(X',Y'),\ldots$ such that $X_d >_{S'} Y_d$, or (b) there exist a variable X_d in $X, X', \ldots$ such that it is the leftmost one that is not in a class of one of the $Y, Y', \ldots$. In both cases, if X_d is not defined in the base segment, then $h(s')\sigma \succ succsum_k(h(t')\sigma)$ for all natural number k, and we conclude since $x_1\sigma$ is a sum of positive and negative summands of this form. Note that, since all the $X, X', \ldots$ are defined in different segments, no cancellation of summands happens in $(X+X'+\ldots)\sigma$. Consider now that X_d is defined in the base segment. In case (a), necessarily Y_d is defined in the base segment, and observe that $X =_{S'} Y, X' =_{S'} Y', \ldots$ for all the pairs in $(X,Y),(X',Y'),\ldots$ to the left of (X_d, Y_d). In fact, by our assumptions, these pairs of variables have been unified, and therefore s and t are of the form $h(X_d + s'')$ and $h(Y_d + s'')$ respectively. Furthermore, either the equations (i) $X_d \geq Y_d + ndec$ and $Y_d \geq 0$, or the equations (ii) $X_d \leq Y_d - ndec$ and $Y_d < 0$, or the equations (iii) $X_d < 0$ and $Y_d > 0$ have been added to the diophantine system. In all this cases and by induction hypothesis we obtain $h(s')\sigma \succ succsum_k(h(t')\sigma)$ for all k in $1\ldots ndec-1$, and since $x_1\sigma$ is a sum of positive and negative summands of this form, we conclude again. In case (b), either the equation (i) $X_d \geq ndec$, or the equation (ii) $X_d < 0$ has been added to the diophantine system, and by analogous arguments we conclude again.
- Let s and t be of the form $f(s')$ and $g(t')$, respectively, for $g > f > h$. Then, by our assumptions, and the form of doing the splitting process, exactly one variable in s' is defined in this segment. Therefore $f(s')\sigma \succ g(t')\sigma$, and hence $f(s')\sigma$ is bigger w.r.t. $\succ$ than any sum of positive and negative summands of the form $h(g(t')+sum_1+\ldots+sum_1)\sigma$.
- Let s and t be of the form $h(s')$ and $f(t')$, respectively. By our assumptions and the form of doing the splitting process s' contains one variable x_j defined in this segment. If j is not i+1, we have that $x_j\sigma \succeq f(t')\sigma + f(t')\sigma$. Therefore $s\sigma \succeq h(f(t')\sigma + f(t')\sigma)$, and since $h(f(t')\sigma + f(t')\sigma)$ is greater than any term of the form $h(f(t')+sum_1+\ldots+sum_1)\sigma$ we have that $s\sigma \succ x_1\sigma$. Suppose now that j is i+1. Then s is of the form $h(x_{i+1}+s'')$. If s'' contains some variable not defined in the base segment, then, $s''\sigma$ is bigger than any term of the form $sum_1+\ldots+sum_1$, and therefore $h(x_{i+1}+s'')\sigma$ is bigger than any sum of positive and negative summands of the form $h(x_{i+1}+sum_1+\ldots+sum_1)\sigma$. If s is of the form $h(x_{i+1}+y)$ where y is a variable defined in the base

segment, then, our diophantine equations and induction on the solution σ guarantee that $y\sigma \succeq sum_1 + \ldots^{n\,dec)} + sum_1$. Since $x_1\sigma$ is a sum of positive and negative summands smaller than $h(x_{i+1} + sum_1 \ldots^{n\,dec)} + sum_1)\sigma$, we have that $s\sigma \succ x_1\sigma$.

- Let s and t be of the form $f(s')$ and $f(t')$, respectively, for $f > h$. If s' contains some variable defined in T, we proceed as in previous cases. Otherwise, suppose that no variable in s' is defined in T. By our assumptions and induction hypothesis we have that $s'\sigma \succ t'\sigma$. Therefore $f_p(s')\sigma$ is greater than any summand of the form $h(f(t') + sum_1 + \ldots + sum_1)\sigma$, and we conclude again.
- Let s be of the form $f(s')$ and let t be either headed with h or with g such that $f \succ g$. If some variable in s' is defined in this segment, we proceed as in previous cases. Then, suppose that no variable in s' is defined in the segment T. It suffices to show that $s\sigma \succ t\sigma$, since automaticaly we obtain $s\sigma \succ succsum_k(t\sigma)$ for all natural numbers k. What we do is proving that $s\sigma \succ t'\sigma$, where t' is either t or any other term to the right of t, and it is done inductively from right to left. Clearly $s\sigma$ is bigger than 0, all the c_j, and all the $x\sigma$ for variables x in the segments delimited by some c_j. If t' is headed by h or g' such that $f \succ g'$, we obtain $s\sigma \succ t'\sigma$ inductively. If t' is headed by f, we obtain $s\sigma \succ t'\sigma$ as in previous case. If t' is headed by g' such that $g' \succ f$, then, some variable x in s' is defined in a segment to the left of t' in S', but to the right of t, and no other variable in s' is defined in this segment (no cancellation is possible). Therefore we obtain $s\sigma \succ t'\sigma$ inductively. If t' is a sum of variables $y_1 + \ldots + y_{k'} - y_{k'+1} - \ldots - y_k$, then, every $y_j\sigma$ is a sum of positive and negative summands of the form $succsum_l(t''\sigma)$, for some t'' that, by induction hypothesis, satisfies $s\sigma \succ t''\sigma$. Therefore, $s\sigma \succ (y_1 + \ldots + y_k)\sigma$.

This σ can be extended to a solution for S analogously to the AC case. □

Theorem 4. *The satisfiability problem for AG-RPO constraints restricted to signatures with functions of arity* 0 *or* 1 *is in NP.*

5.2 Arbitrary arities

The extension to arbitrary signatures is obtained analogously to the AC case. What has to be taken into account is that $succsum_1(f(s_1, \ldots, s_k))$ is $h(0, \ldots, 0, f(s_1, \ldots, s_k))$, and $succsum_1(h(s_1, \ldots, s_k))$ is $h(s_1, \ldots, s_k + sum_1)$ if s_k is not of the form $s' - sum_1$, and $h(s_1, \ldots, s_k - sum_1)$ otherwise.

Theorem 5. *The satisfiability problem for AG-RPO constraints is in NP.*

6 Hardness

Obviously, the satisfiability problems we deal with are NP-hard, because as subcases they include the AC, AC0 and AG-unifiability problems. But one may wonder whether there exists any ordering at all for these E such that at least the satisfiability problem for positive conjunctions of inequations (by which one cannot always encode unification) is in P. Here we answer this question negatively even if monotonicity of the ordering is not required.

Theorem 6. *Let E be AC, AC0, or AG, and let $\succ$ be any arbitrary well-founded E-compatible ordering on ground terms that is total up to $=_E$. Then the constraint satisfiability problem for $\succ$ and $=_E$ is NP-hard even for constraints that are conjunctions of positive inequations.*

Proof. By reducing 1-in-3-sat with only posive literals. We build a conjunction C_P that is satisfiable if, and only if, the 1-in-3-sat problem P is satisfiable. Let t_3, t_2 and t_1 be the three smallest ground terms w.r.t. $\succ$ such that $t_3 \succ t_2 \succ t_1$. Furthermore, let $min(E)$ and $max(E)$ denote the minimal and maximal term (w.r.t. $\succ$), respectively, of the set $E = \{\ t_1{+}t_1{+}t_1,\ \ t_1{+}t_2{+}t_2,\ \ t_2{+}t_2{+}t_2\ \}$.

Now, for each variable x_i in P, we add an inequation $t_3 > x_i$ to C_P, forcing x_i to be either t_1 or t_2. Furthermore, for each clause in P of the form $x_i \vee x_j \vee x_k$, we force exactly one of its three variables to be t_2, by forcing $x_i{+}x_j{+}x_k$ to be $t_1{+}t_1{+}t_2$:

- if $min(E) \succ t_1{+}t_1{+}t_2$, then add $min(E) > x_i{+}x_j{+}x_k$.
- if $t_1{+}t_1{+}t_2 \succ max(E)$, then add $x_i{+}x_j{+}x_k > max(E)$.
- Otherwise, add $u \ > \ x_i{+}x_j{+}x_k \ > \ v$, where u is the smallest term in E larger than $t_1{+}t_1{+}t_2$, and v is the largest term smaller than $t_1{+}t_1{+}t_2$. $\square$

7 Conclusions and further work

Constraint solving algorithms have been defined for FRPO-based orderings for abelian semigroups, abelian monoids and abelian groups. We believe that the new techniques will lead to reasonably efficient practical algorithms for these orderings. This, as well as building an implementation, is one of the directions for further research in the context of the PhD. Thesis of the first author.

Finally, we expect that the ideas given here will provide new insights (to us or to others) for the development of constraint solving methods over fixed signatures for other E-compatible orderings.

References

[CNR95] Hubert Comon, Robert Nieuwenhuis, and Albert Rubio. Orderings, AC-Theories and Symbolic Constraint Solving. In *10th IEEE Symposium on Logic in Computer Science (LICS)*, pages 375–385, San Diego, USA, June 26–29, 1995.

[Com90] Hubert Comon. Solving symbolic ordering constraints. *International Journal of Foundations of Computer Science*, 1(4):387–411, 1990.

[Der82] Nachum Dershowitz. Orderings for term-rewriting systems. *Theoretical Computer Science*, 17(3):279–301, 1982.

[DJ90] Nachum Dershowitz and Jean-Pierre Jouannaud. Rewrite systems. In Jan van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B: Formal Models and Semantics, chapter 6, pages 244–320. Elsevier Science Publishers B.V., Amsterdam, New York, Oxford, Tokyo, 1990.

[GN00] Guillem Godoy and Robert Nieuwenhuis. Paramodulation with built-in abelian groups. In *15th IEEE Symposium on Logic in Computer Science (LICS)*, pages 413–424, Santa Barbara, USA, June 26–29, 2000. IEEE Computer Society Press.

[JO91] J-P. Jouannaud and M. Okada. Satisfiability of systems of ordinal notations with the subterm property is decidable. In *18th International Colloquium Automata, Languages and Programming (ICALP)*, LNCS 510, pages 455–468, Madrid, Spain, July 16–20 1991. Springer-Verlag.

[KKR90] Claude Kirchner, Hélène Kirchner, and Michaël Rusinowitch. Deduction with symbolic constraints. *Revue Française d'Intelligence Artificielle*, 4(3):9–52, 1990.

[KV00] Konstantin Korovin and Andrei Voronkov. A decision procedure for the existential theory of term algebras with the knuth-bendix ordering. In *15th IEEE Symposium on Logic in Computer Science (LICS)*, pages 291–302, Santa Barbara, USA, June 26–29, 2000. IEEE Computer Society Press.

[McC97] William McCune. Solution of the Robbins problem. *Journal of Automated Reasoning*, 19(3):263–276, December 1997.

[Nie93] Robert Nieuwenhuis. Simple LPO constraint solving methods. *Information Processing Letters*, 47:65–69, August 1993.

[NR95] Robert Nieuwenhuis and Albert Rubio. Theorem Proving with Ordering and Equality Constrained Clauses. *Journal of Symbolic Computation*, 19(4):321–351, April 1995.

[NR97] Robert Nieuwenhuis and Albert Rubio. Paramodulation with Built-in AC-Theories and Symbolic Constraints. *Journal of Symbolic Computation*, 23(1):1–21, May 1997.

[NR99] Robert Nieuwenhuis and Albert Rubio. Paramodulation-based theorem proving. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science Publishers (to appear), 1999.

[NR00] Robert Nieuwenhuis and Albert Rubio. Paramodulation-based theorem proving. In J.A. Robinson and A. Voronkov, editors, *Handbook of Automated Reasoning*. Elsevier Science Publishers (to appear), 2000.

[NRV99] P. Narendran, M. Rusinowitch, and R. Verma. RPO constraint solving is in NP. In G. Gottlob, E. Grandjean, and Katrin Seyr, editors, *12th Int. Conference of the European Association of Computer Science Logic (CSL'98)*, LNCS 1584, pages 385–398, Brno, Czech Republic, August 23–28, 1999. Springer-Verlag.

[RN95] Albert Rubio and Robert Nieuwenhuis. A total AC-compatible ordering based on RPO. *Theoretical Computer Science*, 142(2):209–227, May 15, 1995.

[Sch87] Alexander Schrijver. *Theory of Linear and Integer Programming*. John Wiley and Sons, New York, 1987.

[Vig94] Laurent Vigneron. Associative Commutative Deduction with constraints. In Allan Bundy, editor, *12th International Conference on Automated Deduction (CADE)*, LNAI 814, pages 530–544, Nancy, France, June 1994. Springer-Verlag.