Superposition for Bounded Domains

Thomas Hillenbrand and Christoph Weidenbach

Max-Planck-Institut für Informatik
Stuhlsatzenhausweg 85
D-66123 Saarbrücken
{hillen,weidenbach}@mpi-inf.mpg.de

Abstract. Reasoning about bounded domains in resolution calculi is often painful. For explicit and small domains and formulas with a few variables, grounding can be a successful approach. This approach was in particular shown to be effective by Bill McCune. For larger domains or larger formula sets with many variables, there is not much known. In particular, despite general decidability, superposition implementations that can meanwhile deal with large formula sets typically will not necessarily terminate. We start from the observation that lifting can be done more economically here: A variable does not stand anymore for every ground term, but just for the finitely many domain representatives. Thanks to this observation, the inference rules of superposition can drastically be restricted, and redundancy becomes effective. We present one calculus configuration which constitutes a decision procedure for satisfiability modulo the cardinality bound, and hence decides the Bernays-Schönfinkel class as a simple consequence. Finally, our approach also applies to bounded sorts in combination with arbitrary other, potentially infinite sorts in the framework of soft sorts. This frequent combination – which we recently explored in a combination of Spass and Isabelle – is an important motivation of our study.

1 Introduction

Reasoning about bounded domains in resolution-style calculi is often painful. Despite general decidability, superposition implementations typically will not terminate. Bounded domain means that the domain size is bounded from above: In virtue of a clause like $x \simeq 1 \vee \ldots \vee x \simeq n$, any domain element equals one of some given n "digits", which need not be distinct.

Traditionally, attacking bounded domain problems has been done by so-called finite-domain model generators where the most prominent and influential one is Mace4 [25], developed by Bill McCune. In particular, Mace4 has been successfully applied by mathematicians for reasoning in algebraic structures. In addition to the development of Mace4, Bill McCune was also influential in the development of general-purpose automated theorem provers, in particular Otter [27], which was later on replaced by Prover9 [26]. The first version Mace2 of Bill McCune's Mace program encodes a finite domain problem into a SAT problem and then applies a SAT solver. The recent version Mace4 works directly on the finite domain first-order structure at the advantage of using first-order reductions, such as rewriting. So Mace4 is already a big step from Mace2 towards a

M.P. Bonacina and M.E. Stickel (Eds.): McCune Festschrift, LNAI 7788, pp. 68–100, 2013.

first-order logic reasoning procedure. Actually, Bill McCune was already thinking of integrating MACE4 and PROVER9 more closely and suggested *"MACE4 can be a valuable complement to* PROVER9*, looking for counterexamples before (or at the same time as) using* PROVER9 *to search for a proof."* Although MACE4 and PROVER9 rely on the same coding infrastructure, they actually do not work together. Our contribution is a suggestion of a calculus to bridge the gap between classical first-order reasoning and finite-model-search reasoning in the style of MACE4.

The superposition model operator R (see page 74) serves as a kind of MACE4 component in our calculus. It builds a partial model assumption including function tables for all ground instances, due to the function definition clauses $f(\vec{x}) \simeq 1 \vee \ldots \vee f(\vec{x}) \simeq n$ added by our calculus. Then, as is customary for superposition calculi (Definition 3), inferences can be restricted to a minimal false clause with respect to R. We do not exploit this explicitly in the definition of our calculus, but it is part of the completeness approach. When our calculus terminates by saturation, R is a model of the clause set and R is explicitly given by unit rewrite rules in the saturation, see Section 4.4. So in fact, our calculus combines explicit model building in the style of MACE4 and first-order theorem proving in the style of PROVER9. An approach Bill McCune might have thought of already.

Our approach extends the search of finite-domain model generators, which search for suitable interpretations in domains of increasing order. In our approach such interpretations are implicitly constructed by a (partial) model assumption where the calculus itself operates in a superposition style manner on clauses with variables. The problem of (finite) model computation has gained renewed interest, as witnessed by various recent contributions. For example, new approaches via transformation into certain fragments of logic have been presented in [8] and [14]. A variant tailored to instantiation-based methods is given in [6]. The fruitful interplay of superposition and decision procedures is testified, for example, by [2] and [11].

We start from the observation that lifting can be done more economically here: A variable does not stand anymore for every ground term, but just for the finitely many digits (Sect. 3.1). Conversely, an inference only has to be considered if the range of the pertaining most general unifier does exclusively consist of variables and digits. Secondly, for any non-ground inference one can easily determine those instantiations that satisfy its ordering constraints. Thirdly, redundancy also considers digit instances only, such that stronger simplifications become possible in some situations, but compatibility with the corresponding notion of standard superposition is mostly preserved (Sect. 3.2). In order to obtain this, the above cardinality-bounding clause needs to be exchanged for its functional instances $f(\vec{x}) \simeq 1 \vee \ldots \vee f(\vec{x}) \simeq n$ in order not to lose completeness.

The lifting modification applies to the family of superposition calculi. Soundness and refutational completeness are preserved. We demonstrate this for a domain-specific calculus configuration in which non-Horn clauses are dealt with not by equality factoring, but by aggressive splitting. We give a termination result based on the detection of particular loops, and another one based on ordered rewriting with some instantiation (Sect. 4). Both decision procedures for satisfiability modulo the cardinality bound naturally also cope with the Bernays-Schönfinkel class (Sect. 4.5) as a special case. This solves yet another classical decidability problem by superposition. Finally, the lifting modification is also applicable to bounded sorts in combination with arbitrary other, potentially infinite sorts (Sect. 5) in the framework of dynamic sort theories. This frequent combination – think for example of finite enumeration types in programming languages, or any verification problem that involves a component with finite state space – is an important motivation of our study. Two application scenarios are discussed in Sect. 6. Tedious proofs have been omitted here, but can be found in a technical report [21].

Compared to instantiation-based methods for finite-domain problems with explicit instantiation such as Mace [25], Paradox [13], Finder [34], SEM [39], and related calculi such as hyper-tableaux [7], our calculus does not instantiate variables a priori, but exploits the boundedness of the domain on the level of non-ground clauses. In particular, this offers advantages if the problem has structure that can be employed by inference and reduction rules [32]. As a first simple example, not a single inference is possible between the two unit clauses $P(x_1, \ldots, x_k, x_1)$ and $\neg P(a, y_1, \ldots, y_{k-1}, b)$, but instantiation-based methods will generate more than n^k clauses for domain size n. In general, a superposition inference or simplification that involves variables simulates up to exponentially many ground steps. Likewise, proving one inference redundant may save an exponential amount of work. As a second example, consider an equation $f(x) \simeq x$ and an atom $P(f(g(x)))$, which standard rewriting would simplify to $P(g(x))$. After instantiation with digits this reduction is no longer possible, as any term $g(\ldots)$ is not a digit. For examples of this form, inferring and simplifying at the non-ground level has the potential to exponentially shorten proofs and model representations. In Sect. 6 we elaborate two real-world examples of this form.

Transformation-based methods [24,8] translate a given clause set into a form on which standard inference mechanisms like hyperresolution search for a model in a bottom-up way. This work is orthogonal to ours, because the problem is transformed, whereas we exploit the boundedness of the domain truly at the calculus level. However, neither the instantiation-based nor the transformation-based approach currently support the combination with general first-order theories, in contrast to our calculus.

Starting with a simple Bernays-Schönfinkel style setting, where all function symbols are constants, we prove that a cardinality-bounding clause $x \simeq 1 \vee \ldots \vee x \simeq n$ is not needed and can be dropped. Nevertheless, superposition is not a decision procedure for this class. It may generate arbitrary long clauses with an unbounded number of variables. Consider for example a clause expressing a confluence property, such as

$$\neg P(x, y) \vee \neg P(x, z) \vee P(y, z).$$

All occurrences of P literals are incomparable by the reduction ordering underlying superposition, so in particular superposition self inferences with this clause produce arbitrary long clauses with an unbounded number of variables. Our solution here is to extend superposition inferences by lazy instantiation with digits such that the literals triggering the inference become (strictly) greatest in the ordering. For the above example, a potential inference with a clause $\neg P(1,1)$ is not possible, because whatever digit is substituted for x after unifying $P(y,z)$ with $P(1,1)$ a negative literal will become greatest, assume the natural ordering on the digits $1 \prec 2 \ldots \prec n$ (see Definition 3). Together with a splitting rule this style of reasoning basically already guarantees termination on this fragment. For a bounded fragment with non-constant function symbols, the situation gets more involved. In particular, on such a fragment we do want to perform rewriting as much as possible in the standard first-order style. We achieve this goal by exchanging the cardinality-bounding clause $x \simeq 1 \vee \ldots \vee x \simeq n$ for its functional instances $f(\vec{x}) \simeq 1 \vee \ldots \vee f(\vec{x}) \simeq n$. Still we limit unifiers in inferences to digits or variables, but support almost unlimited rewriting with arbitrary terms and matchers. Basically, these two ingredients lift our approach from the Bernays-Schönfinkel class to full first-order clause sets with a cardinality-bounding clause.

Finally, a sort discipline supports combination of finite domain sorts with others as it naturally occurs in real-world application. Think for example of a network model where the single bits 0 and 1 for building bit vectors representing network addresses must not be confused with other terms. For example, a cardinality bounding clause

$$\neg \mathit{Bit}(x) \vee x \simeq 0 \vee x \simeq 1.$$

should not be involved in any inference at any position with a clause talking about bit vectors, such as performing a logical and operation for IP-addresses:

$$\mathit{IP}(x_1 \circ y_1, \ldots) \simeq \mathit{ipand}(\mathit{IP}(x_1, \ldots), \mathit{IP}(y_1, \ldots))$$

where $\circ$ represents bitwise "and". This property is supported by our bounded domain calculus introduced in Sect. 5.

2 Getting Started

For most logical notions and notations, we refer to [29]. In particular we work in a logic with built-in equality. We stipulate a single-sorted signature Σ that contains the constant symbols 1 through n, which we name *digits,* besides arbitrary other function symbols, possibly including constants. So equality is the only predicate symbol, but free predicate symbols will briefly be discussed in Sect. 4.5. Moving on, a set $\mathscr{V}$ provides an infinite supply of variables. For a term t we denote by $\mathrm{var}(t)$ the set of variables that occur in t; the set $\mathrm{var}(C)$ is defined correspondingly for every clause C. If σ is a substitution, then $\mathrm{dom}\,\sigma$ is the set of all variables for which $x\sigma \not\equiv x$, $\mathrm{ran}\,\sigma$ is the image of $\mathrm{dom}\,\sigma$ under σ, and $\mathrm{cdom}\,\sigma$ is the set of variables occurring in $\mathrm{ran}\,\sigma$. We say that a substitution σ can be *refined into* a substitution ρ if $\rho = \sigma\tau$ for some substitution τ, and that σ is *more general than* ρ if it can be refined into ρ, but not vice versa. For simplicity of notation the equality symbol $\simeq$ is supposed to be symmetric. A literal $s \bowtie t$ is either an equation $s \simeq t$ or a disequation $s \not\simeq t$. A clause is a disjunction of literals; a Horn clause is a clause with at most one positive literal;

the empty clause is denoted by $\bot$. We assume that a reduction ordering $\succ$ is given which is total on ground terms. To every literal, we assign a *complexity* according to $s \simeq t \mapsto \{s, t\}$ and $s \not\simeq t \mapsto \{s, s, t, t\}$. Literals are compared in the multiset extension of $\succ$ on their complexities, and clauses in the two-fold multiset extension on the multisets of the respective literal complexities. If C is a clause and M a clause set, then $M^{\prec C}$ holds all elements of M smaller than C, and $\mathrm{gnd}(C)$ consists of all ground instances of C.

We study the theory $\mathcal{T}$ given by the formula

$$\forall x.\, x \simeq 1 \vee \ldots \vee x \simeq n$$

and will introduce a superposition-based calculus to tackle the $\mathcal{T}$-satisfiability of clause sets over Σ. Note that this also covers the case that the domain size is exactly n, since one can add clauses $i \not\simeq j$ for any distinct $i, j \in [1; n]$.

The calculus will be described by rule patterns of three different types in a fraction-like notation. Clauses occurring in the numerator are generally called *premises*, and in the denominator *conclusions*. As usually, premises are assumed to share no variables. Finite clause sequences $C_1, \ldots, C_m$ where $m \geq 0$ are abbreviated as $\vec{C}$, and $\bigwedge \vec{C}$ is the conjunction of all C_i. If C denotes a clause and M a clause set, then M, C is shorthand notation for $M \cup \{C\}$.

(i) *Inference rules:* $\mathcal{I}\,\dfrac{\vec{C}}{D}$ if *condition*

denotes any transition from a clause set $M, \vec{C}$ to $M, \vec{C}, D$ provided *condition* is fulfilled. Occasionally the rightmost of the premises is named *main premise*, and the remaining ones are the *side premises.*

(ii) *Reduction rules:* $\mathcal{R}\,\dfrac{C}{\vec{C'}}\,\vec{D}$ if *condition*

stands for any transition from a clause set $M, C, \vec{D}$ to a clause set $M, \vec{C'}, \vec{D}$ whenever *condition* holds. In essence, the clause C is replaced by the clauses $\vec{C'}$, the sequence of which may be empty.

(iii) *Split rules:* $\mathcal{S}\,\dfrac{C}{D \mid D'}$ if *condition*

describes any transition from a clause set M, C to the pair of clause sets $(M, C, D \mid M, C, D')$ constrained by *condition.* Note that the premise is part of each of the descending clause sets.

In the *condition* part of inference rules, frequently some terms, say s and t, are required to have a most general unifier σ. We stipulate that σ satisfies $\mathrm{dom}\,\sigma \cup \mathrm{cdom}\,\sigma \subseteq \mathrm{var}(s, t)$, which for syntactic unification is without loss of generality. Furthermore, occurrences of terms or literals may be restricted to maximal ones. In the former case this refers to the enclosing literal, and in the latter to the enclosing clause. Maximality means that no other occurrence is greater, and is strict if none is greater or equal. Correspondingly we will speak of greatest occurrences, which are greater than or equal to the remaining ones, and of strictly greater ones, that are greater than all the rest. There is no difference between being greatest or maximal in case the underlying ordering is total, as happens in the case of ground clauses and a reduction ordering total on ground terms.

An application of one of the above rules is called an *inference,* a *reduction* or a *split*, respectively. Given an inference with premises $\vec{C}$ and conclusion D, then an *instance* of this inference is every inference with premises $\vec{C}\sigma$ and conclusion $D\sigma$.

A *derivation* from a (not necessarily finite) clause set M with respect to a calculus specified that way is a finitely branching tree such that (i) the nodes are sets of clauses, (ii) the root is M, and (iii) if a node N has the immediate descendants $N_1, \ldots, N_k$, respectively, then there is a transition from N to $N_1, \ldots, N_k$ in the calculus. Infinite inputs could, for example, arise from the instantion of finite sets, or from the enumeration of some theory. A *complete path* $N_1, N_2, \ldots$ in a derivation tree starts from the root, ends in a leaf in case the path is finite, and has the *limit* $N_\infty = \bigcup_i \bigcap_{j \geq i} N_j$. Note that the term "complete" has been chosen simply to stress that the path, if finite, indeed reaches a leaf. Given a redundancy notion for inferences and clauses, a derivation is said to be *fair* if for every complete path $N_1, N_2, \ldots$ the following applies to the transitions from N_∞: (i) Every inference is redundant in some N_i, and (ii) in case a split rule is present in the calculus, then for every split, one of its conclusion is in some N_i or redundant with respect to it. A clause set M is *saturated* if (i) every inference with premises in M is redundant with respect to M, and (ii) for every split, one of its conclusion is in M or redundant with respect to it. In the context of full first-order logic without a bounded domain, fairness and completeness is more involved [16].

3 A Calculus for $\mathcal{T}$-unsatisfiability

3.1 Calculus Rules

Let us first recapitulate a standard variant of superposition. For the sake of simplicity, selection of negative equations is not taken into account yet.

Definition 1. The *standard superposition calculus* $\mathcal{S}$ consists of the rules

Negative superposition
$$\mathcal{I}\,\frac{C \vee \underline{\underline{s} \simeq t} \quad \underline{\underline{u[s']} \not\simeq v} \vee D}{(C \vee u[t] \not\simeq v \vee D)\sigma}$$

Equality resolution
$$\mathcal{I}\,\frac{C \vee \underline{s \not\simeq s'}}{C\sigma}$$

Positive superposition
$$\mathcal{I}\,\frac{C \vee \underline{\underline{s} \simeq t} \quad \underline{\underline{u[s']} \simeq v} \vee D}{(C \vee u[t] \simeq v \vee D)\sigma}$$

Equality factoring
$$\mathcal{I}\,\frac{C \vee \underline{\underline{s} \simeq t} \vee \underline{s'} \simeq u}{(C \vee t \not\simeq u \vee s \simeq u)\sigma}$$

subject to the *restrictions*

(i) $\sigma = \mathrm{mgu}(s, s')$, and $s' \notin \mathcal{V}$ in case of the superposition rules
(ii) under σ, the underlined occurrences are maximal or strictly maximal
(iii) the main premise is strictly maximal under σ

and augmented with a notion of *redundancy* with respect to a clause set M:

(a) for a clause C, if $\mathrm{gnd}(M)^{\prec C\sigma} \models C\sigma$ for every ground substitution σ
(b) for an inference, if for every ground instance with maximal premise $C\sigma$ and conclusion $D\sigma$ we have $\mathrm{gnd}(M)^{\prec C\sigma} \models D\sigma$

Let us remark that condition (iii) shows up, for example, in [4, Sect. 3], but not in all presentations of superposition. It excludes, for instance, the superposition of $s \simeq t$ into $s \simeq u$ if $s \succ t \succ u$, and may facilitate termination proofs.

The calculus $\mathcal{S}$ is sound and refutationally complete in the sense that $M \models \bot$ and $\bot \in M$ coincide for every saturated set M, and the limit of every fair derivation is saturated and equivalent to the input. The completeness proof relies on a model functor that associates with M a convergent ground rewrite system R. For saturated M free of $\bot$, the model is the quotient R^* of the free ground term algebra modulo the congruence generated by R. The rewrite system is given in terms of sequences $\mathrm{Gen}(C)$ and R_C which are defined by mutual recursion. For every ground clause C let $\mathrm{Gen}(C) = \{s \to t\}$ if (i) $C \equiv C' \vee \underline{\underline{s} \simeq t} \in \mathrm{gnd}(M)$, (ii) $R_C^* \not\models C$, (iii) $R_C^* \not\models t \simeq u$ for all literals $s \simeq u$ in C, (iv) s is R_C-irreducible; and let $\mathrm{Gen}(C) = \{\}$ otherwise. Furthermore R_C is $\bigcup_{D \prec C} \mathrm{Gen}(D)$, and finally R is $\bigcup_D \mathrm{Gen}(D)$.

As already mentioned, the rationale of our calculus refinement is that a variable will just stand for the digits, not for every ground term. Let us introduce some notions to make this precise: For a substitution τ we say that it *numbers* if $\mathrm{ran}\,\tau \subseteq [1;n]$. Note that τ is more general than another numbering substitution τ if and only if $\tau \subset \tau'$, in the set-theoretic sense. So we say that τ *minimally numbers* with respect to a set of conditions if these are satisfied by τ and by no other numbering τ' more general than τ. Furthermore τ *ground numbers* a clause C if τ numbers and $C\tau$ is ground. The set of all ground instances of C under such substitutions is denoted by $\Omega(C)$, and its elements are called the *Ω-instances* of C.

Alas, if we apply the new interpretation of variables to the clause $x \simeq 1 \vee \ldots \vee x \simeq n$ which defines our theory, then each of its instances in terms of digits is a tautology. Hence we exchange $\mathcal{T}$ for the set $\mathcal{T}'$ of its functional instances, which consists of these clauses:

$$f(\vec{x}) \simeq 1 \vee \ldots \vee f(\vec{x}) \simeq n \qquad \text{for any } f \in \Sigma \setminus [1;n]$$

$\mathcal{T}'$ is weaker than $\mathcal{T}$ in the sense that the upper cardinality bound is only applied to function values; but it satisfies the same universal formulae. There is an increase in the initial number of clauses, but this will be outshined by the fact that no inferences with complex unifiers are necessary. Interestingly, within the Bernays-Schönfinkel class the set $\mathcal{T}'$ is empty, as we will demonstrate in Sect. 4.5.

The following equivalence makes our consideration precise:

Proposition 2. A clause set M is $\mathcal{T}$-satisfiable iff $\Omega(M \cup \mathcal{T}')$ is satisfiable.

How to exploit this proposition? Assume M is a non-ground clause set, and we are interested in its $\mathcal{T}$-satisfiability. In standard superposition, if N is a saturated presentation of $M \cup \mathcal{T}$, then $\mathrm{gnd}(N)$ is saturated as well. However, it is sufficient that $\Omega(N)$ is saturated, provided the saturation started from $M \cup \mathcal{T}'$. The benefit is that lifting can be made more economically: An inference only has to be considered if the range of the pertaining most general unifier consists of variables and digits only. Secondly, for any non-ground inference one can, via partial instantiation with digits, determine those instantiations that satisfy its ordering constraints. Thirdly, redundancy also considers digit instances only and becomes effective. We formulate the following refinement of standard superposition:

Definition 3. *Superposition for bounded domains $\mathcal{S}_B$* refines $\mathcal{S}$ as follows:

$$\begin{array}{llll}
\textit{Negative super-position} & \mathcal{I}\,\dfrac{C \lor \underline{\underline{s} \simeq t} \quad \underline{\underline{u[s']} \not\simeq v} \lor D}{(C \lor u[t] \not\simeq v \lor D)\sigma} & \textit{Equality resolution} & \mathcal{I}\,\dfrac{C \lor \underline{s \not\simeq s'}}{C\sigma} \\
\textit{Positive super-position} & \mathcal{I}\,\dfrac{C \lor \underline{\underline{s} \simeq t} \quad \underline{\underline{u[s']} \simeq v} \lor D}{(C \lor u[t] \simeq v \lor D)\sigma} & \textit{Equality factoring} & \mathcal{I}\,\dfrac{C \lor \underline{\underline{s} \simeq t} \lor \underline{s'} \simeq u}{(C \lor t \not\simeq u \lor s \simeq u)\sigma}
\end{array}$$

under the *restrictions*

(i) $\sigma = \mathrm{mgu}(s, s')$, and $s' \notin \mathscr{V}$ in case of the superposition rules

(ii) $\operatorname{ran}\sigma \subseteq \mathscr{V} \cup [1;n]$

(iii) there is a minimally numbering substitution τ such that under $\sigma\tau$
 – the underlined occurrences are greatest or strictly greatest
 – the main premise is strictly greatest

where *redundancy* with respect to a clause set M is given

(a) for a clause C if $\Omega(M)^{\prec C\rho} \models C\rho$ for every ground numbering ρ

(b) for an inference with main premise C, most general unifier σ, minimally numbering substitution τ and conclusion D if for every ground numbering ρ we have $\Omega(M)^{\prec C\sigma\tau\rho} \models D\rho$.

Based on the notion of redundancy, *simplification,* in its general form, is making a clause redundant by adding (zero or more) entailed smaller clauses. Here it is already enough if these conditions hold on the Ω-instances.

$$\mathcal{R}\,\frac{C}{\vec{C'}}\,\vec{D} \qquad \text{if} \quad \begin{array}{l} \cdot\ C \text{ is redundant w.r.t. } \vec{C'}, \vec{D} \\ \cdot\ \Omega(C, \vec{D}) \models \Omega(\bigwedge \vec{C'}) \\ \cdot\ \Omega(C) \succ \Omega(\vec{C'}) \end{array}$$

Testing the existence of a substitution τ in (iii) is effective for every decidable reduction ordering. Actually, instead of just testing, one could alternatively enumerate *all* such minimally numbering substitutions τ for which the mentioned maximality conditions hold, and for each τ add the inference conclusion instantiated by τ. The number of these substitutions is always finite, but it may become large. Just to give an example, equality resolution would become the following:

$$\mathcal{I}\,\frac{C \lor s \not\simeq s'}{C\sigma\tau} \ \text{if} \ \begin{array}{l} \cdot\ \sigma = \mathrm{mgu}(s, s') \text{ and } \operatorname{ran}\sigma \subseteq \mathscr{V} \cup [1;n] \\ \cdot\ \tau \text{ minimally numbers such that} \\ \quad s \not\simeq s' \text{ is greatest under } \sigma\tau \end{array}$$

Let us stress that condition (ii) – absence of complex unifiers – is easy to test and should exclude many of the inferences drawn in the standard calculus. For example, with the lexicographic path ordering [22] induced by the precedence $+ \succ s$, from the two clauses $(x+y)+z \simeq x+(y+z)$ and $u+s(v) \simeq s(u+v)$ one would normally obtain every $s^i(x+y)+z \simeq x+(s^i(y)+z)$. But since y needs to be bound to $s(v)$, not a single inference is drawn in the calculus $\mathcal{S}_B$.

We stipulate that from now on the smallest ground terms are the digits from $[1;n]$, say such that $n \succ \ldots \succ 1$. Then the calculus $\mathcal{S}_B$ is sound and refutationally complete in the sense that a clause set M is $\mathcal{T}$-unsatisfiable if and only if every fair $\mathcal{S}_B$-derivation from $M \cup \mathcal{T}'$ eventually produces the empty clause. Notably the minimality of the digits is indispensable: Assume that $\succ$ is the lexicographic path ordering induced by the precedence $n \succ \ldots \succ 1 \succ f \succ c$. Then from the unsatisfiable clause set $\{f(x) \simeq 1,\ 1 \simeq c,\ 1 \not\simeq f(c)\}$ nothing but the clause $f(c) \not\simeq c$ is inferable. When lifting ground-level inferences to the non-ground level, this minimality is needed to show that variable overlaps are non-critical; and indeed the variable overlap from $1 \simeq c$ into $f(x) \simeq 1$ would produce $f(c) \simeq 1$ and eventually lead to the empty clause.

3.2 Redundancy in $\mathcal{S}_B$ and in $\mathcal{S}$

In the calculus $\mathcal{S}_B$, redundancy on the general level is defined via redundancy of Ω-instances on the ground level, whereas in standard superposition one goes back to redundancy of all ground instances. Let us compare under which conditions a clause C is redundant with respect to a clause set M. In the calculus $\mathcal{S}_B$ we require $\Omega(M)^{\prec C\rho} \models C\rho$ for every ground numbering ρ. The condition in standard superposition is $\mathrm{gnd}(M)^{\prec C\sigma} \models C\sigma$ for every ground substitution σ. So for redundancy in the sense of $\mathcal{S}_B$ fewer instances need to be shown redundant, but on the other hand there are fewer premises for doing so. For example, $f(g(1)) \simeq 1$ is not redundant with respect to $f(x) \simeq 1$, since it is not entailed from $f(1) \simeq 1, \ldots, f(n) \simeq 1$. Fortunately, in $\mathcal{S}_B$-derivations the set M with respect to which redundancy is studied always contains the clauses of $\mathcal{T}'$, possibly simplified. Therefore we additionally have $g(1) \simeq 1 \vee \ldots \vee g(1) \simeq n$ at hand, with which $f(g(1)) \simeq 1$ does become redundant.

This subsection contains two results that generalize this observation. Firstly, if every digit instance $C\rho$ is entailed from smaller ground instances of M except some problematic ones, then C is redundant in the sense of $\mathcal{S}_B$. Secondly, if every $C\rho$ follows from arbitrary smaller ground instances, but C is not of a particular form, then C is also redundant. These results permit us to adapt concrete simplification techniques like rewriting or subsumption to our calculus. The subsection ends with a demonstration that $\mathcal{S}_B$ should not be mixed with the standard notion of redundancy.

We reserve the identifier f for non-digit function symbols, whereas i, j, k denote digits and $\vec{\imath}$ a vector thereof. For any term t, let $\mathrm{Dig}(t)$ denote the clause $t \simeq 1 \vee \ldots \vee t \simeq n$.

Given a clause C with ground substitution σ, we call the pair C, σ *problematic* if $x\sigma \equiv f(\vec{\imath})$ for some $x \in \mathrm{var}(C)$ and $C\sigma \preceq \mathrm{Dig}(f(\vec{\imath}))$. Otherwise the pair is called *unproblematic*. Furthermore, let $\mathring{\mathrm{gnd}}(C)$ denote the set of all ground instances $C\sigma$ for which C, σ is unproblematic, and let $\mathring{\mathrm{gnd}}$ extend to clause sets in the usual way. Here are two necessary and quite restrictive conditions for C, σ to be problematic: Firstly, some variable $x \in \mathrm{var}(C)$ may occur only in literals of the form $x \simeq i$ and $x \simeq y$. Secondly, the greatest literal of $C\sigma$ must have the form $f(\vec{\imath}) \simeq j$.

Additionally, a clause C is called *critical* if it has an Ω-instance $C\rho$ with greatest term $f(\vec{\imath})$ such that $C\rho \preceq \mathrm{Dig}(f(\vec{\imath}))$. Otherwise C is called *noncritical.* Note that this notion refers to Ω-instances, whilst in a problematic pair C, σ, the second element is an arbitrary ground substitution,

Lemma 4. Consider a path in an $\mathcal{S}_B$-derivation from $M \cup \mathcal{T}'$ to N and a clause C. Then C is redundant with respect to N if one of the following conditions holds, where ρ ranges over all ground numbering substitutions:

(i) $\mathrm{g\mathring{n}d}(N)^{\prec C\rho} \models C\rho$ for all ρ,

(ii) C is noncritical and $\mathrm{gnd}(N)^{\prec C\rho} \models C\rho$ for all ρ.

The difference between our redundancy notion and the one of standard superposition may show up in practice: Assume $n = 2$ and some input M which via $\mathcal{S}_B$ eventually leads to the clause set $N = \{x \simeq 1,\ f(1) \simeq 2,\ f(2) \simeq 2,\ f(1) \not\simeq 1\}$. Now the clause $x \simeq 1$ has the ground instances $2 \simeq 1$ and $f(1) \simeq 1$ which make the second and the third clause redundant in the standard sense. Since $f(1) \simeq 1$ is not an Ω-instance of $x \simeq 1$, these clauses are not redundant in the sense of $\mathcal{S}_B$. Note also that $x \simeq 1, \{x \mapsto f(1)\}$ is problematic and that both $f(1) \simeq 2$ and $f(2) \simeq 2$ are critical, such that Lem. 4 does not apply.

Going further, the example shows that combining $\mathcal{S}_B$ with standard redundancy is problematic: If $f(1) \simeq 2$ and $f(2) \simeq 2$ were deleted from N, then the rest $\{x \simeq 1,\ f(1) \not\simeq 1\}$ would be $\mathcal{S}_B$-saturated, despite the apparent unsatisfiability. Summing up, refutational completeness would be lost. However, because of Lem. 4 only in rare cases is standard redundancy stronger than redundancy in the sense of $\mathcal{S}_B$.

Notably the opposite relation can be observed as well: Let $n = 2$, $C \equiv x \simeq y \vee f(1) \simeq y$ and $N = \{f(1) \simeq 1 \vee f(1) \simeq 2,\ f(2) \simeq 1 \vee f(2) \simeq 2,\ 1 \simeq 2,\ C\} \supseteq \mathcal{T}'$. The clause C is redundant in the sense of $\mathcal{S}_B$ because $C\rho$ is a tautology if $x\rho \equiv y\rho$, and because otherwise $C\rho$ is subsumed by $1 \simeq 2$. However C is not redundant in the standard sense: Consider the ground instance $C\sigma \equiv f(1) \simeq 1 \vee f(1) \simeq 1$. We obtain $\mathrm{gnd}(N)^{\prec C\sigma} = \{1 \simeq 2,\ 1 \simeq 1 \vee f(1) \simeq 1,\ 2 \simeq 1 \vee f(1) \simeq 1\}$, which is equivalent to $\{1 \simeq 2\}$. Clearly, this does not entail $f(1) \simeq 1$. One cannot hold the exchange of $\mathcal{T}'$ for $\mathcal{T}$ responsible for this phenomenon, since it also occurs in case of $N' = \{x \simeq 1 \vee x \simeq 2,\ 1 \simeq 2,\ C\}$.

3.3 Application to Unit Rewriting

For a set E of unit equations, the *ordered rewrite relation* $\rightarrow_E$ is commonly defined as the smallest relation on terms such that $u[s\sigma] \rightarrow_E u[t\sigma]$ whenever $s \simeq t \in E$ and $s\sigma \succ t\sigma$, where σ is a substitution such that $s\sigma$ occurs as subterm u. If t contains variables that do not show up in s, as in $f(x) \simeq f(y)$, then one has to guess an instantiation of these in order to achieve decreasingness. However, in case a solution exists, then binding to the minimal constant works as well. So we stipulate that additionally $(\mathrm{var}(t) \setminus \mathrm{var}(s))\sigma \equiv \{1\}$ holds. As usual, the reflexive-transitive closure of $\rightarrow_E$ is denoted by $\rightarrow_E^*$.

We extend the ordered rewrite relation $\rightarrow_E$ from terms to clauses in the obvious way. As such, it is a simplification in the sense of our calculus only if the clause to be simplified is above the simplifying equation instances. For example,

$$f(3) \simeq 1 \ \rightarrow_{\{f(3) \simeq 2\}} \ 2 \simeq 1$$

is a rewrite step, but not a simplification, because the clause to be rewritten is smaller than the one used for rewriting. In order to capture this, let $\rightarrow_E^{\succ}$ denote the smallest relation on clauses such that $C[s\sigma] \rightarrow_E^{\succ} C[t\sigma]$ whenever $s \simeq t \in E$, $s\sigma \rightarrow_E t\sigma$ and $C\rho \succ (s \simeq t)\sigma\rho$ for all ground numbering ρ. A further condition is necessary to ensure that the rewritten clause is redundant according to Lem. 4: Rewriting $C \rightarrow_E^* D$ is called *Ω-admissible* for any noncritical C. If C is critical, however, then it contains literals of the shape $f(\vec{s}) \simeq t$ where t and every s_i is a digit or a variable, such that with a suitable ground numbering substitution ρ the term $f(\vec{s})\rho$ is the greatest of $C\rho$. Then $C \rightarrow_E^* D$ is *Ω-admissible* only if rewrite steps on the left-hand side of such literals $f(\vec{s}) \simeq t$ with equations $x \simeq i \in E$ or $x \simeq y \in E$ only take place below f. So the following is an instance of simplification in the calculus $\mathcal{C}$:

Ordered unit rewriting

$$\mathcal{R}\,\frac{C}{D}\,E \qquad \text{if} \quad \begin{array}{l} \cdot\ E \text{ is a set of unit equations} \\ \cdot\ C \rightarrow_E^{\succ} \circ \rightarrow_E^* D \\ \cdot\ C \rightarrow_E^* D \text{ is } \Omega\text{-admissible} \end{array}$$

4 Obtaining a Decision Procedure

4.1 Calculus Rules

Refutational completeness of the calculus $\mathcal{S}_B$ means that if M is $\mathcal{T}$-unsatisfiable, then in every fair derivation eventually the empty clause will show up, even for infinite M. If M is $\mathcal{T}$-satisfiable, however, then derivations without suitable simplification steps may become infinite. We will present a calculus configuration that enforces termination. To make this effective, naturally the input clause set M must be finite, as well as the signature Σ; and the ordering $\succ$ must be decidable.

Going back to standard superposition $\mathcal{S}$, without simplifications this calculus does not decide the satisfiability of finite ground clause sets: If $\succ$ is a lexicographic path ordering induced by the precedence $a \succ f \succ b$, then from the equations $f(a) \simeq a$ and $a \simeq f(b)$ one obtains an infinite series $f(f(b)) \simeq a$, $f(f(f(b))) \simeq a$, ... by positive superposition. However, if all clauses are units, then every inference conclusion makes its main premise redundant and hence can be turned into a simplification. The clause set decreases in the multiset extension of the clause ordering, which guarantees termination.

The satisfiability of finite ground Horn clause sets can be decided the same way if in every clause with negative literals, at least one of them shall be *selected.* This eager selection leads to a positive unit literal strategy [15], where the side premise of superposition inferences is always a positive unit clause. We denote this ground-level calculus variant by $\mathcal{G}$. Via splitting of non-Horn clauses into Horn clauses, decidability extends to the non-Horn case. In [20], we have encoded Sudoku puzzles as ground satisfiability problems for the SPASS theorem prover, which proceeding that way succeeded within a blink of an eye. Therefore, we decided to choose $\mathcal{G}$ as a basis for the formulation of a decision procedure.

The resulting calculus $\mathcal{C}$ for arbitrary clauses is an instance of $\mathcal{S}_B$ where in every Horn clause with negative literals at least one of them shall be selected. Besides, equality factoring is exchanged for an aggressive splitting rule. If a clause contains positive literals with shared variables, then the digit instances of this clause are split. The number of split conclusions can become large, but remains finite, as opposed to the general case without theory $\mathcal{T}$. In order to simplify the treatment, a clause should always be split at the same position. Hence we assume that for every non-Horn clause an arbitray partitioning into two subclauses is *designated* where each subclause has strictly fewer positive literals. Now, the calculus rules are the following:

Negative superposition

$$\mathcal{I}\,\frac{l \simeq r \quad s[l'] \not\simeq t \vee C}{(s[r] \not\simeq t \vee C)\sigma\tau}$$

if
- $\cdot$ $l' \notin \mathscr{V}$ and $\sigma = \mathrm{mgu}(l, l')$
- $\cdot$ $\operatorname{ran}\sigma \subseteq \mathscr{V} \cup [1;n]$
- $\cdot$ τ minimally numbers such that l and s are strictly greatest under $\sigma\tau$
- $\cdot$ $s \not\simeq t$ is selected
- $\cdot$ C is Horn

Positive superposition

$$\mathcal{I}\,\frac{l \simeq r \quad s[l'] \simeq t}{(s[r] \simeq t)\sigma\tau}$$

if
- $\cdot$ $l' \notin \mathscr{V}$ and $\sigma = \mathrm{mgu}(l, l')$
- $\cdot$ $\operatorname{ran}\sigma \subseteq \mathscr{V} \cup [1;n]$
- $\cdot$ τ minimally numbers such that l and s are strictly greatest under $\sigma\tau$ and $(l \simeq r)\sigma\tau \prec (s \simeq t)\sigma\tau$

Equality resolution

$$\mathcal{I}\,\frac{C \vee t \not\simeq t'}{C\sigma}$$

if
- $\cdot$ $\sigma = \mathrm{mgu}(t, t')$
- $\cdot$ $\operatorname{ran}\sigma \subseteq \mathscr{V} \cup [1;n]$
- $\cdot$ $t \not\simeq t'$ is selected
- $\cdot$ C is Horn

Split

$$\mathcal{S}\,\frac{C \vee s \simeq t \vee l \simeq r \vee D}{(C \vee s \simeq t)\tau \mid (l \simeq r \vee D)\tau}$$

if
- $\cdot$ the partitioning is designated
- $\cdot$ τ minimally numbers such that the conclusions share no variables

In the two superposition rules, applying the numbering substitution τ in the conclusion guarantees that the number of variables in the latter is not higher than in the main premise, which is exploited in one of our termination proofs. Conversely, if there is no increase in the number of variables before applying τ, then one can *avoid enumerating* all such substitutions and just add the *single* conclusion instance where τ is the identity.

4.2 Soundness and Refutational Completeness

Next we give a formulation of lifting, which is at the heart of our approach. We reduce completeness of our calculus, on the non-ground level, to that of calculus

$\mathcal{G}$ on the ground level. Therefore, no dedicated model functor will be necessary for proving $\mathcal{C}$ complete. For any clause set M, let $\widehat{M}$ denote its Ω-instances that are Horn clauses.

Proposition 5. If a clause set M is $\mathcal{C}$-saturated, then $\widehat{M}$ is $\mathcal{G}$-saturated.

Proof. We adapt the usual lifting arguments (see for example [29, p. 393]) to our calculus, inspecting $\mathcal{G}$-inferences with premises from $\widehat{M}$. If a clause $D \in \widehat{M}$ contains negative literals, then let the literal selection be inherited from one arbitrary $C \in M$ that instantiates into D.

- Ground positive superposition: Given two clauses $l \simeq r$ and $s \simeq t$ from M with ground numbering substitution ρ, consider the $\mathcal{G}$-inference with premises $l\rho \simeq r\rho$ and $s\rho[l\rho]_p \simeq t\rho$, and conclusion $s\rho[r\rho]_p \simeq t\rho$. The position p is not introduced by ρ because the range of ρ consists of digits only. This $\mathcal{G}$-inference corresponds to a variable overlap if $s|_p \equiv x \in \mathscr{V}$, and to a non-variable overlap otherwise.

  In the former case we have $x\rho \equiv l\rho$, such that $l\rho$ is a digit. Because $l\rho \succ r\rho$ and the digits are the smallest ground terms, the term $r\rho$ must be a digit as well. Let ρ' denote the substitution identical to ρ except that $x\rho' \equiv r\rho$. Then $(s \simeq t)\rho'$ is contained in $\Omega(M)$ and makes the inference redundant.

  Now we come to non-variable overlaps. Let $l' \equiv s|_p$, furthermore $\sigma = \mathrm{mgu}(l, l')$ with $\mathrm{dom}\,\sigma \subseteq \mathrm{var}(l, l')$, and $\rho = \sigma\sigma'$. Because ρ is a ground numbering substitution, we know that $x\rho$ is a digit for every $x \in \mathrm{dom}\,\sigma$. Given $\rho = \sigma\sigma'$, every $x\sigma$ is either a digit or a variable, because the range of σ contains only digits and variables.

  The substitution σ' numbers the clauses $s\sigma \simeq t\sigma$ and $l\sigma \simeq r\sigma$ in such a way that the literals $l\sigma$ and $s\sigma$ are greatest under σ', respectively, and that $(l \simeq r)\sigma\sigma' \prec (s \simeq t)\sigma\sigma'$. If τ is a more general such substitution, then it satisfies $\mathrm{dom}\,\tau \subseteq \mathrm{dom}\,\sigma'$ and $x\tau \equiv x\sigma'$ for every $x \in \mathrm{dom}\,\tau$, which implies $\tau \subseteq \sigma'$. There exists a $\subset$-minimal such τ because all descending $\subset$-chains are finite. Summing up: $l \simeq r,\ s[l'] \simeq t \vdash (s[r] \simeq t)\sigma\tau$ is a $\mathcal{C}$-inference with premises from M, and is redundant with respect to M because M is saturated. If $\sigma' = \tau\tau'$, then the inference instance under τ' is redundant with respect to $\Omega(M)$.
- Ground equality resolution: Consider a Horn clause $C \vee t \not\simeq t' \in M$ with ground numbering substitution ρ such that $C\rho \vee t\rho \not\simeq t'\rho \vdash C\rho$ is a $\mathcal{G}$-inference. We may assume that $t \not\simeq t'$ is selected in $C \vee t \not\simeq t'$. As usual, t and t' have a most general unifier σ, which specializes into ρ say via σ'. We obtain $\mathrm{ran}\,\sigma \subseteq \mathscr{V} \cup [1;n]$ like for ground positive superposition. So $C \vee t \not\simeq t' \vdash C\sigma$ is a $\mathcal{C}$-inference with premises from M; and its redundancy carries over to that of the above instance.
- Ground negative superposition: similar to ground positive superposition, but taking selectedness into account like for ground equality resolution.

□

The calculus $\mathcal{C}$ is sound and refutationally complete:

Lemma 6. For every clause set M, the following are equivalent:

(i) M is $\mathcal{T}$-satisfiable.

(ii) Every fair derivation from $M \cup \mathcal{T}'$ contains a complete path $N_1, N_2, \ldots$ such that the empty clause is not in N_∞.

Proof sketch. A series of propositions rewrites characterization (i) into (ii). First, the clause set M is $\mathcal{T}$-satisfiable iff in every derivation from $M \cup \mathcal{T}'$ there exists a complete path $N_1, N_2, \ldots$ such that every $\Omega(N_i)$ is satisfiable. Second, every $\Omega(N_i)$ in this path is satisfiable iff $\Omega(N_\infty)$ is. Third, we note that N_∞ is $\mathcal{C}$-saturated provided the derivation is fair. Fourth, $\Omega(N_\infty)$ and $\widehat{N_\infty}$ are equivalent because in case of saturated sets, split conclusions are redundant or contained. Fifth, $\widehat{N_\infty}$ is saturated with respect to $\mathcal{G}$ according to Prop. 5. Finally, since $\mathcal{G}$ is sound and complete, the satisfiability of $\widehat{N_\infty}$ is equivalent to $\bot \notin \widehat{N_\infty}$, which is the same as $\bot \notin N_\infty$. □

4.3 Termination by Loop Detection

In this subsection, we will pinpoint where the non-terminating behaviour in $\mathcal{C}$-derivations arises from, and then look for a remedy. So we study here fair $\mathcal{C}$-derivations that start from some finite input $M \cup \mathcal{T}'$ and exclusively consist of inferences, splits and simplifications. In order to avoid trivial loops, no inference or split shall be repeated while the parent clauses persist. Furthermore, simplifications shall not increase the number of variables in a clause, a condition that inferences and splits satisfy:

Proposition 7. Inference and split conclusions do not have more variables than one of the premises.

Compared to standard superposition, the calculus $\mathcal{C}$ is far more restrictive: There are no inferences with complex unifiers; both inferences and splits do not increase the number of variables; and in each satisfiable path, every ground term can eventually be rewritten into a digit. A further observation is the following:

Proposition 8. Consider an inference or a split or a simplification

$$\mathcal{I}\,\frac{C_1\ \ldots\ C_m}{D_1} \quad \text{or} \quad \mathcal{S}\,\frac{C_m}{D_1 \mid D_2} \quad \text{or} \quad \mathcal{R}\,\frac{C_m}{\vec{D}}\,\vec{C}'$$

in the calculus $\mathcal{C}$. In each case we have $\Omega(C_m) \succ \Omega(D_i)$, for every i.

By König's lemma, a derivation is infinite if and only if it contains an infinite path. Such a path is only possible with infinitely many inference steps. The clauses that occur in a path can be arranged in a forest with the input clauses as root nodes, and with each inference conclusion or split clause or reduct being attached to its corresponding parent clause C_m of Prop. 8. Inductively it is clear that whenever a clause does not persist in a path of a derivation, but is generated again later, then the two occurrences produce distinct nodes. Therefore, a path in a derivation is infinite if and only if the corresponding forest is. Because of the decreasingness result in Prop. 8, infinite paths in the forest are impossible. By construction of the calculus, a node in the forest with infinitely many children can only arise from binary inferences, more precisely, from superposition inferences with the same main premise. Since the number of possible substitutions is essentially finite, by the bounded number of variables, we obtain:

Lemma 9. A $\mathcal{C}$-derivation is infinite if and only if it contains a path with

– an infinite sequence of equations $l \simeq r_i$, up to variable renaming,

– a persistent clause $C[l']$, and

– infinitely many superposition steps $\mathcal{S}\,\dfrac{l \simeq r_i \qquad C[l']}{C[r_i]\sigma\tau}$ with σ, τ fixed,

provided it starts from finite $M \cup \mathcal{T}$.

Note that negative superposition steps cannot be excluded from the characterization: If there is an inference into a positive main clause $s[l'] \simeq t$, then one may also construct one into $s[l'] \not\simeq x \vee x \simeq t$ where x is fresh, and simplify the resolvents $(s[r_i] \not\simeq k \vee k \simeq t)\sigma\tau$ to $(s[r_i] \simeq t)\sigma\tau$.

A first example of an infinite derivation, without any rewriting, was given in the beginning of Sect. 4, on page 78. As in all examples to come, a lexicographic path ordering is employed. The inducing precedence here is $a \succ f \succ b$:

$$\mathcal{I}\,\frac{a \simeq f(b) \quad f(a) \simeq a}{f(f(b)) \simeq a} \qquad \mathcal{I}\,\frac{a \simeq f(f(b)) \quad f(a) \simeq a}{f(f(f(b))) \simeq a} \qquad \cdots$$

As already said, all inference steps could be carried out as simplifications, namely by unit rewriting. We would like to know whethe unit rewriting prevents nontermination. Inferencing is not rewriting in the next example, where the precedence is $f \succ h \succ g \succ a$:

$$\mathcal{I}\,\frac{f(a) \simeq g(x) \quad h(f(y),z) \simeq f(y)}{h(g(1),z) \simeq f(a)} \qquad \mathcal{I}\,\frac{f(a) \simeq h(g(1),x) \quad h(f(y),z) \simeq f(y)}{h(h(g(1),1),z) \simeq f(a)} \qquad \cdots$$

However, the second side premise can be rewritten by the first, which is not possible in the following example. We use a signature with the binary function symbols $\cdot$, $+$, and $\_^{-}$ for exponentiation, and with the unary symbol f. In order to avoid overly many parentheses, the symbol $\_^{-}$ shall bind tightest, followed by $\cdot$ and $+$, which shall associate to the left. The precedence is $f \succ \cdot \succ \_^{-} \succ + \succ n \succ \ldots \succ 1$.

$$\mathcal{I}\,\frac{x^x \cdot z \;\simeq\; 1 \cdot x^x + z \cdot x^x \qquad x^y \cdot 1 + f(x^x \cdot z') \;\simeq\; x^x \cdot z'}{1 \cdot x^x + 1 \cdot x^x + f(x^x \cdot z') \simeq x^x \cdot z'}$$

$${\scriptstyle \{x_1+x_2+f(x_3\cdot x_4)\simeq x_1+x_2+x_4\cdot x_3\}}\Big\downarrow_{\Upsilon}$$

$$1 \cdot x^x + 1 \cdot x^x + z' \cdot x^x \simeq x^x \cdot z'$$

$$\mathcal{I}\,\frac{x^x \cdot z \;\simeq\; 1 \cdot x^x + 1 \cdot x^x + z \cdot x^x \qquad x^y \cdot 1 + f(x^x \cdot z') \;\simeq\; x^x \cdot z'}{1 \cdot x^x + 1 \cdot x^x + 1 \cdot x^x + f(x^x \cdot z') \simeq x^x \cdot z'}$$

$${\scriptstyle \{x_1+x_2+f(x_3\cdot x_4)\simeq x_1+x_2+x_4\cdot x_3\}}\Big\downarrow_{\Upsilon}$$

$$1 \cdot x^x + 1 \cdot x^x + 1 \cdot x^x + z' \cdot x^x \simeq x^x \cdot z'$$

$$\vdots$$

Still unit rewriting enforces termination: From the premise of the first inference into that of the second, there is a superposition inference producing $1{\cdot}x^x{+}1{\cdot}x^x \simeq 1 \cdot x^x + 1 \cdot x^x + 1 \cdot x^x$, which simplifies all higher-index right-hand sides r_i.

This example leads to a general property: Under the conditions given in Lem. 9, there exist superposition inferences from $l \simeq r_1$ and $l \simeq r_2$, and one easily calculates that with respect to these and to $C[r_1]\sigma\tau$, the inference producing $C[r_2]\sigma\tau$ is redundant, though unit rewriting need not always simplify the conclusion. We call a $\mathcal{C}$-derivation *loop-free* if it contains no such inference steps, and satisfies the conditions given in the beginning of this subsection: no repetition of inferences or splits from persisting parent clauses, and no increase in the number of variables when simplifying a clause. Whether an individual inference satisfies these conditions or not can be read off the derivation history, or memoized suitably.

Theorem 10. Loop-free $\mathcal{C}$-derivations decide $\mathcal{T}$-satisfiability of finite clause sets.

This termination result is built on the insight in Prop. 8 that only the decreasing inferences be drawn. In the calculus $\mathcal{C}$, this is achieved with the numbering substitution τ, which is not present in the calculus $\mathcal{S}_B$. Alternatively, one could attach constraints to the clauses and thereby restrict the inference conclusions to the decreasing digit instances.

4.4 Termination by Rewriting

The calculus $\mathcal{C}$ is constructed such that if a clause set is saturated, then the associated model can be read off the set of remaining unit equations, which is ground confluent and reduces every ground term to a digit. Therefore, we set out here a decision procedure with unit rewriting as the major simplification device.

Given a clause set N with unit equations $E \subseteq N$, we say that N *reduces to digits* if $f(\vec{\imath}) \rightarrow_E^* j$ for every digit vector $\vec{\imath}$. Inductively every ground term can then be rewritten to a digit as well. Furthermore, a clause is called $[1;n]$*-shallow* if non-digit function symbols occur only at the top-level of positive literals.

One may want to test explicitly whether a given N_k reduces to digits already (and if so, perhaps test immediately whether E_k describes a $\mathcal{T}$-model of M). Notably the property is not always inherited from N_k to N_{k+1}. Consider for example the following simplification steps in the sense of the calculus $\mathcal{C}$:

$$\mathcal{R}\,\frac{f(3) \simeq f(1)}{1 \simeq f(1)}\; 1 \not\simeq 1 \vee f(3) \simeq 1 \qquad\qquad \mathcal{R}\,\frac{f(3) \simeq 1}{f(2) \simeq 1}\; \begin{array}{l} f(1) \simeq 3 \\ f(1) \simeq 2 \end{array}$$

The term $f(3)$ is E_k-reducible, but not necessarily E_{k+1}-reducible. As the second example shows, this may even occur if unit equations are simplified with respect to E_k only. In case this is not desired, one has to restrict the simplification of unit equations. For example, ordered unit rewriting, instance rewriting, subsumption and tautology elimination are compatible.

Alas, even when a given N_k reduces to digits, such that every term $f(\vec{\imath})$, and every ground term $f(\vec{t})$, is reducible, then unit rewriting on the non-ground level can be inapplicable although it would be possible on every Ω-instance: If $n = 2$ and $N = \{f(1) \simeq 2,\ f(2) \simeq 1,\ f(f(x)) \simeq x\}$, then the third equation cannot be rewritten, but its Ω-instances could be turned into the tautologies $1 \simeq 1$ or $2 \simeq 2$, respectively. Hence we need to combine instantiation and rewriting in that situation. If C is a clause and Γ a set of numbering substitutions with $\operatorname{dom}\tau \subseteq \operatorname{var}(C)$ for every $\tau \in \Gamma$, then we say that Γ *covers* C if every ρ that ground numbers C can be obtained as specialization of some $\tau \in \Gamma$.

Instance rewriting

$$\mathcal{R}\,\frac{C}{\{D_\rho : \rho \in \Gamma\}}\,E \qquad \text{if} \quad \begin{array}{l} \cdot\ E \text{ is a set of unit equations} \\ \cdot\ \Gamma \text{ covers } C \\ \cdot\ \text{for every } \rho \in \Gamma\text{: } C\rho \rightarrow_E^{\succ} \circ \rightarrow_E^{*} D\rho \\ \quad\ \text{and } C\rho \rightarrow_E^{*} D\rho \text{ is } \Omega\text{-admissible} \end{array}$$

Instance rewriting allows clauses to be replaced eventually by their $[1;n]$-shallow equivalents:

Proposition 11. Consider a complete path $N_1, N_2, \ldots$ in a fair derivation from $M \cup \mathcal{T}'$, where M is finite.

(i) For some index κ, all $N_{\kappa+i}$ contain $\bot$; or they all reduce to digits.

(ii) If $C \in N_{\kappa+i}$ is not $[1;n]$-shallow, then C can effectively be simplified into a finite set of $[1;n]$-shallow clauses.

We will require instance rewriting only on newly generated clauses once the clause set reduces to digits. Note also that simplifying a $[1;n]$-shallow clause with respect to other such clauses can arbitrarily increase the number of variables and need not preserve $[1;n]$-shallowness, as for example witnessed by

$$\mathcal{R}\,\frac{f(x) \simeq 2}{g(1) \not\simeq g(1) \vee y_1 \not\simeq y_1 \vee \ldots \vee y_m \not\simeq y_m \vee f(x) \simeq 1}\,2 \simeq 1$$

if $f(1) \succ g(1)$. Clearly this counteracts our efforts towards termination; so a strategy is needed that guides the execution of calculus steps. We say that a $\mathcal{C}$-derivation is a *$\mathcal{C}_\kappa$-derivation* from a clause set M if (i) it is fair, (ii) the root node is $M \cup \mathcal{T}'$, and in every path eventually (iii) simplifications do not increase the number of variables, (iv) $[1;n]$-shallowness is preserved under simplifications, (v) inferences and splits are not repeated, (vi) every fresh inference conclusion which is not $[1;n]$-shallow, is immediately simplified into a set of $[1;n]$-shallow clauses, (vii) no duplicate literals occur in $[1;n]$-shallow clauses, and (viii) $[1;n]$-shallow clauses equal up to variable renaming are identified. Indeed such derivations exist for every finite M: The crucial item (vi) can be satisfied because of Prop. 11.

Theorem 12. $\mathcal{C}_\kappa$-derivations decide $\mathcal{T}$-satisfiability of finite clause sets.

4.5 Extensions

Let us have a short look at a many-sorted setting where $\mathcal{T}$ consists of size restrictions for every sort, each built over an individual set of digits. One has to employ the usual typing constraints for equations, terms and substitutions. Then the calculus $\mathcal{C}$, and the results obtained for it so far, straightforwardly extends to this situation.

Up to now, our calculus did not deal with predicates. Of course one could extend $\mathcal{C}$ with an ordered resolution rule, and consider predicate atoms in the superposition and split rules. Alternatively, we can introduce a two-element sort Bool, say over the digits I and II, and provide a clause $\mathrm{I} \not\simeq \mathrm{II}$. As usually we can now encode predicate atoms $P(\vec{t})$ of any other sort as equations $P(\vec{t}) \simeq \mathrm{I}$. Notably $\mathcal{T}'$ need not contain an axiom $P(\vec{x}) \simeq \mathrm{I} \vee P(\vec{x}) \simeq \mathrm{II}$: Given an algebra $\mathcal{A}$ such that at some point $P^{\mathcal{A}}$ does not map into $\{\mathrm{I}^{\mathcal{A}}, \mathrm{II}^{\mathcal{A}}\}$, let the algebra $\mathcal{B}$ coincide with $\mathcal{A}$ except that $P^{\mathcal{B}}$ maps all such points onto $\mathrm{II}^{\mathcal{B}}$. Then $\mathcal{A}$ and $\mathcal{B}$ satisfy the same encoded atoms $P(\vec{t}) \simeq \mathrm{I}$.

As an application, consider the validity problem for a formula $\phi \equiv \forall x_1 \ldots \forall x_n \exists y_1 \ldots \exists y_m \phi'$ where ϕ' is quantifier-free and contains no function symbols. This problem was proven decidable by Bernays and Schönfinkel [9]. Now, ϕ is valid iff $\psi \equiv \forall y_1 \ldots \forall y_m \neg\phi'\{x_1 \mapsto 1, \ldots, x_n \mapsto n\}$ is unsatisfiable iff ψ is $\mathcal{T}$-unsatisfiable. Since no function symbols are present, the set $\mathcal{T}'$ is empty. Notably, no instance rewriting steps are needed in such derivations because all clauses are shallow.

Corollary 13. Both $\mathcal{C}_\kappa$-derivations and loop-free $\mathcal{C}$-derivations decide the Bernays-Schönfinkel class.

Finally, it is often desired to assume the bounded domain digits $1, \ldots, n$ to be different. This can be expressed by n^2 disequations $i \not\simeq j$, where $i \neq j$ and $1 \leq i, j \leq n$. For larger n this is not a desirable solution. Then an additional inference rules that removes equations between digits [33] is a better solution. This approach was already successfully tested for Bernays-Schönfinkel problems over a large number of constants [35].

5 Combinations with Unbounded First-Order Theories

So far, we have only considered the case where the entire Herbrand domain of a formula is finite. The interesting question is whether the techniques developed in the previous sections can be generalized to a setting where the overall Herbrand domain may be infinite, but bounded subsets of the domain are specified. The answer we give in the section is affirmative; the combination can exploit the advanced technology: In every inference, variables over any bounded subset only need to be instantiated to variables and to the finitely many domain representatives. Furthermore no inferences with the axiom expressing boundedness are needed.

The overall approach is to code bounded subsets via monadic predicates, which we also call *soft sorts* [19,36]. In contrast to the use of sorts in algebraic specifications, sorts are represented in the clause set by their monadic relativization predicates enjoying the standard first-order semantics. Sort theories show up in the form of Horn clauses in these monadic predicates and can be dynamically used for simplification. Therefore, soft sorts may be empty, there are no restrictions on the language, sorts are not a priori disjoint, elements of sorts are not necessarily different and sorts may of course also be defined via general clauses. For example, the clause $\neg R(x, f(x)) \vee S(x)$ defines x to be contained in the sort S if the relation $R(x, f(x))$ holds, and the clause $\neg S(x) \vee \neg T(x)$ states that the sorts S and T are disjoint.

Provided a clause C contains a negative literal $\neg S(x)$, we say that x *is of sort* S *in* C. To give an example, any model $\mathcal{A}$ of the clauses $S(1)$, $S(2)$, $\neg S(x) \vee x \simeq 1 \vee x \simeq 2$ must satisfy $1 \leq |S^{\mathcal{A}}| \leq 2$. If we add the clause $1 \not\simeq 2$, then any model $\mathcal{A}$ fulfills $|S^{\mathcal{A}}| = 2$, whereas the alternative extension with $1 \simeq 2$ leads to $|S^{\mathcal{A}}| = 1$. Concerning functions, the clause $\neg S(x) \vee S(f(x))$ declares f to map elements from S into S.

In this section, we study the bounded-domain theory $\mathcal{T}$ for one sort S of cardinality up to n defined by

$$\mathcal{T} = \{S(1),\ \ S(2),\ \ \ldots,\ \ S(n),\ \ \neg S(x) \vee x \simeq 1 \vee \ldots \vee x \simeq n\}$$

which is a clausal presentation of the formula $\forall x.\, S(x) \leftrightarrow x \simeq 1 \vee \ldots \vee x \simeq n$. The results can be extended to several bounded-domain sorts in the obvious way.

Similarly to the restricted case of Sect. 3, we would like to instantiate variables of sort S with digits only. All such instances of the clause $\neg S(x) \vee x \simeq 1 \vee \ldots \vee x \simeq n \in \mathcal{T}$ are tautologies. To compensate for this, we introduce an operator $\_^{\circ}$ to be applied to input clauses that replaces every positive literal $S(t)$ by the disjunction $t \simeq 1 \vee \ldots \vee t \simeq n$. Furthermore let in this section

$$\mathcal{T}' = \{S(1), \ldots, S(n)\}.$$

Finally $\Omega_S(C)$ shall denote the set of all clauses obtained from C via instantiation of all variables of sort S in C with digits. In this sense Ω_S is the restriction of Ω to variables of sort S. The following lemma is the analogue of Prop. 2:

Lemma 14. A clause set N is $\mathcal{T}$-satisfiable iff $\Omega_S(N^{\circ})$ is $\mathcal{T}'$-satisfiable.

Proof. "$\Rightarrow$" Let $\mathcal{A}$ be a model for $N \cup \mathcal{T}$, i.e., $\mathcal{A} \models N \cup \mathcal{T}$ and so $\mathcal{A} \models \mathcal{T}'$. Since in particular $\mathcal{A} \models \mathcal{T}$ we know $S^{\mathcal{A}} = \{1^{\mathcal{A}}, \ldots, n^{\mathcal{A}}\}$ and hence $\mathcal{A} \models C$ iff $\mathcal{A} \models \Omega_S(C)$ for any clause C. We show $\mathcal{A} \models C$ implies $\mathcal{A} \models C^{\circ}$ for all $C \in N$. We distinguish the following cases: (i) C does not contain a positive literal $S(t)$. Then $C \equiv C^{\circ}$ and we are done. (ii) Let $C \equiv S(t_1) \vee \ldots \vee S(t_m) \vee D$ and D does not contain a positive literal $S(t)$, $m > 0$. Let σ be any valuation[1] for all variables in all t_i. Then $\mathcal{A}, \sigma \models S(t_i)$ iff $(t_i\sigma)^{\mathcal{A}} \in S^{\mathcal{A}}$ iff $(t_i\sigma)^{\mathcal{A}} = k^{\mathcal{A}}$ for some digit $1 \leq k \leq n$ iff $\mathcal{A}, \sigma \models t_i \simeq k$. Hence if $\mathcal{A} \models C$ so $\mathcal{A} \models C^{\circ}$.

"$\Leftarrow$" Let $\mathcal{A} \models \Omega_S(N^{\circ}) \cup \mathcal{T}'$ and let $\mathcal{A}'$ be identical to $\mathcal{A}$, except that $S^{\mathcal{A}'} = \{1^{\mathcal{A}}, \ldots, n^{\mathcal{A}}\}$. Obviously, $\mathcal{A}' \models \mathcal{T}'$ and $\mathcal{A}' \models \Omega_S(N^{\circ})$ because $\{1^{\mathcal{A}}, \ldots, n^{\mathcal{A}}\} \subseteq S^{\mathcal{A}}$. We need to show $\mathcal{A}' \models N$ and $\mathcal{A}' \models \mathcal{T}$ where the latter holds by construction of $\mathcal{A}'$. By construction $\mathcal{A}', \sigma \models S(t_i)$ iff $\mathcal{A}', \sigma \models t_i \simeq k$ for some digit $1 \leq k \leq n$ and any valuation σ in the variables of t_i. Now assume there is a clause $C \in N$ with $\mathcal{A}', \sigma \not\models C\sigma$ for some valuation σ. Thus, if there is some $\neg S(x)$ in C, then

[1] We confuse here substitutions and valuations in the usual way.

$x\sigma \in S^{\mathcal{A}}$ implying $C^{\circ}\sigma \in \Omega_S(N^{\circ})$. Now, since $\mathcal{A}', \sigma \models S(t_i)$ iff $\mathcal{A}', \sigma \models t_i \simeq k$ we have $\mathcal{A}', \sigma \not\models C^{\circ}\sigma$, a contradiction.

Note that the four clauses $N = \{S(1), S(2), \neg S(x) \vee x \simeq 1 \vee x \simeq 2, f^3(x) \not\simeq f(x)\}$ are satisfiable as neither the input, nor the output of f is specified to be of sort S. Adding the declaration $N' = N \cup \{\neg S(x) \vee S(f(x))\}$ lets f map from S into S and hence causes unsatisfiability. For the latter clause, the transformation of Lem. 14 applies. We get $(\neg S(x) \vee S(f(x)))^{\circ} = \neg S(x) \vee f(x) \simeq 1 \vee f(x) \simeq 2$ and we obtain the set

$$\begin{aligned}\Omega_S((N')^{\circ}) = \{&S(1), S(2), f^3(x) \not\simeq f(x),\\ &\neg S(1) \vee f(1) \simeq 1 \vee f(1) \simeq 2,\\ &\neg S(2) \vee f(2) \simeq 1 \vee f(2) \simeq 2\}\end{aligned}$$

which is unsatisfiable.

Now by the lifting theorem for standard superposition, we know because of Lem. 14 that $N \cup \mathcal{T}$ has a superposition refutation iff $N^{\circ} \cup \mathcal{T}'$ has one. The open question is how we can exploit the fact that we considered solely numbering substitutions for variables of sort S. Note that although S has a bounded domain, the overall domain of N may be infinite. Hence we cannot take the approach of Sect. 3 where we used the numbering substitution available for all variables to require that inferences are only performed on strictly greatest terms and literals. Furthermore, the abstract superposition redundancy notion is no longer effective and satisfiability is of course not decidable anymore. Therefore, the idea is to restrict the range of substitutions for variables of sort S to $\mathcal{V} \cup [1; n]$, and to require that (strict) maximality is preserved under any numbering substitution for the bounded sort S. The superposition calculus including this refinement consists of the standard rules positive and negative superposition, equality resolution and factoring, instantiated by the additional restrictions.

Positive superposition

$$\mathcal{I}\,\frac{C \vee l \simeq r \quad s[l'] \simeq t \vee D}{(C \vee s[r] \simeq t \vee D)\sigma} \quad \text{if}$$

- $\cdot$ $l' \notin \mathcal{V}$ and $\sigma = \mathrm{mgu}(l, l')$
- $\cdot$ $\operatorname{ran}\sigma|_S \subseteq \mathcal{V} \cup [1; n]$
- $\cdot$ there exists a minimally numbering τ of sort S such that l, $l \simeq r$, s, $s \simeq t$ are strictly maximal under $\sigma\tau$ and $(C \vee l \simeq r)\sigma\tau \not\succeq (s \simeq t \vee D)\sigma\tau$

Negative superposition

$$\mathcal{I}\,\frac{C \vee l \simeq r \quad s[l'] \not\simeq t \vee D}{(C \vee s[r] \not\simeq t \vee D)\sigma} \quad \text{if}$$

- $\cdot$ $l' \notin \mathcal{V}$ and $\sigma = \mathrm{mgu}(l, l')$
- $\cdot$ $\operatorname{ran}\sigma|_S \subseteq \mathcal{V} \cup [1; n]$
- $\cdot$ there exists a minimally numbering τ of sort S such that l, $l \simeq r$, s are strictly maximal under $\sigma\tau$, $s \not\simeq t$ is maximal under $\sigma\tau$ or selected

Equality resolution

$$\mathcal{I}\,\frac{C \vee t \not\simeq t'}{C\sigma} \quad \text{if} \quad \begin{array}{l} \cdot\ \sigma = \mathrm{mgu}(t, t') \\ \cdot\ \mathrm{ran}\,\sigma|_S \subseteq \mathscr{V} \cup [1;n] \\ \cdot\ \text{there exists a minimally numbering } \tau \\ \quad \text{of sort } S \text{ such that} \\ \quad t \not\simeq t' \text{ is maximal under } \sigma\tau \text{ or selected} \end{array}$$

Equality factoring

$$\mathcal{I}\,\frac{C \vee s \simeq t \vee s' \simeq u}{(C \vee t \not\simeq u \vee s \simeq u)\sigma} \quad \text{if} \quad \begin{array}{l} \cdot\ \sigma = \mathrm{mgu}(s, s') \\ \cdot\ \mathrm{ran}\,\sigma|_S \subseteq \mathscr{V} \cup [1;n] \\ \cdot\ \text{there exists a minimally numbering } \tau \\ \quad \text{of sort } S \text{ such that } s,\ s' \\ \quad \text{are strictly maximal under } \sigma\tau, \\ \quad s \simeq t \text{ is maximal under } \sigma\tau \end{array}$$

For the general combination of a bounded sort with arbitrary formulae over potentially infinite domains, we cannot aim at a decision procedure, but only at refutational completeness. Hence, the above rule delays instantiations of bounded-sort variables as long as possible, but applies the underlying restrictions.

We stipulate that the digits $1, \ldots, n$ are minimal in the ordering $\succ$. Furthermore, in every clause $\neg S(t) \vee C$ with a negative sort literal, the argument of S shall always be a digit or a variable. If t is neither of these, then one can apply variable abstraction and obtain $\neg S(x) \vee x \not\simeq t \vee C$. Notably the new variable x needs to be instantiated with digits only, and hence cannot become maximal. Our considerations give rise to the following completeness result:

Theorem 15. A clause set N is $\mathcal{T}$-unsatisfiable iff there is a derivation of the empty clause from $N^\circ \cup \mathcal{T}'$ by the superposition calculus defined above.

Proof. By Lem. 14 $N \cup \mathcal{T}$ is unsatisfiable iff $\Omega_S(N') \cup \mathcal{T}'$ is unsatisfiable. The set $\Omega_S(N') \cup \mathcal{T}'$ is unsatisfiable iff there is a derivation of the empty clause by the standard superposition calculus. As the digits are minimal in the ordering, they might only be replaced by each other. For any clause $\neg S(x) \vee C \in N'$, all instances of $\neg S(x)$ in the proof are generated by substitutions from x into $[1;n]$. Hence, all steps can be lifted to steps of the above refined superposition calculus on N'.

Here is an example for the refined maximality condition. Let $\succ$ denote the lexicographic path ordering induced by the precedence $f \succ g \succ n \succ \ldots \succ 1$. Then in the clause $\neg S(x) \vee g(x,y) \simeq y \vee f(y) \simeq y$, the literal $g(x,y) \simeq y$ is not maximal, because $f(y) \succ g(i,y)$ holds for every $i \in [1;n]$.

6 Three Application Scenarios

6.1 Combination with Theories

There is currently a great interest in combining general purpose reasoning procedures for propositional or first-order logic with theories, such as arithmetic or theories modeling data structures. A combination of this kind has great potential in program analysis and verification. It is also mandatory in the sense that, e.g., the theory of arithmetic is not representable in first-order logic whereas the control flow of a program can hardly be represented by an arithmetic theory.

In addition to the so called SMT (SAT Modulo Theories) approach [30] that combine propositional logic with theories, there are meanwhile also a number of combination approaches between first-order logic and theories available [5,23,1,3,12,18] and (partly) implemented and successfully applied. Following the hierarchic approach [5] if the base theory, e.g. arithmetic [1], is completely separated from the first-order theory, and all first-order sorts are bounded, then our calculus from Sect. 5 yields a decision procedure following the ideas of Sect. 4. However, if theory terms and first-order terms are mixed, then the best we can get is completeness, in general. Already a combination of the Bernays-Schönfinkel fragment with linear arithmetic enables the encoding of the halting problem for two-counter machines [28,17].

Nevertheless it is an interesting question and a starting point for future work whether the inference restrictions introduced here, can in fact be combined with superposition/resolution-based combination ideas starting from first-order logic [3,12].

6.2 Vectors over Finite Domains

We have formalized parts of a LAN infrastructure as a bounded-domain problem, see `http://spass-prover.org/prototypes`. Spass saturates this problem in less than one second. Actually, we have currently not integrated the calculus refinements for bounded domains in Spass. However, by the structure of the clauses in this example, only digit unifiers are considered for inferences. Notably, Spass even succeeds on this problem extended with the router and firewall configurations of both Max Planck institutes at Saarbrücken, which takes about 30 minutes. You may submit this as a challenge to your favorite instantiation-based provers. The specification includes a theory of IP addresses; these are essentially bitvectors. A vector-level conjunction $\mathit{AND}(\_,\_)$ is defined via recurrence to a bit-level conjunction $\_ \cdot \_$ as follows:

$$\mathit{AND}(\mathit{IP}(x_{31},\ldots,x_0),\mathit{IP}(y_{31},\ldots,y_0)) \simeq \mathit{IP}(x_{31}\cdot y_{31},\ldots,x_0\cdot y_0)$$

The bit theory has a bounded domain with digits 0 and 1. The theory extension $\mathcal{T}'$ and $\_^{\circ}$-transformation of the sort declaration for $\cdot$ are (see page 86):

$$\mathit{Bit}(0) \qquad \mathit{Bit}(1) \qquad \forall x,y.\, \mathit{Bit}(x)\wedge \mathit{Bit}(y) \rightarrow x\cdot y \simeq 0 \vee x\cdot y\simeq 1$$

The sort $\mathit{Bit}(\_)$ is needed to prevent confusion with other sorts of the theory, in particular IP addresses. This is a perfect example of our approach to combinations in Sect. 5. Bit-level conjunction is defined by

$$\forall x,y.\, \mathit{Bit}(x)\wedge \mathit{Bit}(y) \rightarrow \begin{array}{c}(x\cdot y\simeq 0 \leftrightarrow x\simeq 0 \vee y\simeq 0)\\ \wedge\ (x\cdot y\simeq 1 \leftrightarrow x\simeq 1 \wedge y\simeq 1)\end{array}$$

Confusion of bits is prevented by the clause $0 \not\simeq 1$, and confusion of IP addresses by the following clauses, where i ranges over all indices:

$$IP(x_{31},\ldots,x_{i+1},0,x_{i-1},\ldots,x_0) \not\simeq IP(x_{31},\ldots,x_{i+1},1,x_{i-1},\ldots,x_0)$$

The clausification of the overall theory can be saturated finitely, for example if $\succ$ is an LPO induced by the precedence $AND \succ \cdot \succ IP$. Note that already the minimal model size for this part of the LAN theory is $2^{32}+2$, and due to classless routing, confusion of IP addresses is not an adequate approach to reduce the size of the model. Now, if we extend this theory to bitvectors of length 64, then the additional effort in saturation is bound by a factor of two, whereas an instantiation-based method has to consider 2^{64} domain elements for the IP addresses. This effect that we already pointed out in [21] was later studied in [32] in a systematic way.

6.3 Proof Obligations from ISABELLE

Recently [10], we have been working on a version of Spass [37] that in particular supports proof obligations out of Sledgehammer [31] invocations from Isabelle [38]. One challenge of the translation of higher-order formulas is that the booleans become explicit, i.e., in almost all obligations there is a clause

$$\neg\mathrm{bool}(x) \lor x \simeq \mathrm{true} \lor x \simeq \mathrm{false}$$

We applied to this clause and the soft sort bool the transformations introduced in Sect. 5 without making use of the extra ordering restrictions. When comparing the runs of the new Spass [10] with and without the transformation, we gained an average speedup factor of 3 on all examples.

7 Conclusion and Future Work

We have presented a light-weight adaptation of superposition calculi to the first-order theory of bounded domains. The achievement is a superposition calculus for bounded domains that restricts the range of inference unifiers to digits or variables, facilitates the precise calculation of ordering restrictions, introduces an effective general semantic redundancy criterion, incorporates a particular splitting rule for non-Horn clauses, can constitute a decision procedure for any bounded-domain problem, is mostly compatible with all the standard superposition redundancy criteria, and can in particular be embedded via a general dynamic sort discipline based on monadic predicates in any general first-order setting.

We have already done some promising experiments on the basis of ground-level formulations for bounded domains [20], and a partial, light weight integration into Spass [10] that does not explore the additional ordering restrictions. To this end, ordering computation, inference computation and simplifications need be refined accordingly.

Future work has already started in getting the superposition partial model operator R effective without the need for explicit instantiation. If this works out, then systems can be developed that actually combine the strengths of explicit model search in the style of Mace and automated theorem proving in the style of Otter. A combination Bill McCune might have already thought of.[2]

Acknowledgements. We are indebted to our reviewers for their valuable and constructive comments that were essential for the eventual quality of the paper.

References

1. Althaus, E., Kruglov, E., Weidenbach, C.: Superposition Modulo Linear Arithmetic SUP(LA). In: Ghilardi, S., Sebastiani, R. (eds.) FroCoS 2009. LNCS, vol. 5749, pp. 84–99. Springer, Heidelberg (2009)
2. Armando, A., Ranise, S., Rusinowitch, M.: Uniform Derivation of Decision Procedures by Superposition. In: Fribourg, L. (ed.) CSL 2001. LNCS, vol. 2142, pp. 513–527. Springer, Heidelberg (2001)
3. Armando, A., Bonacina, M.P., Ranise, S., Schulz, S.: New results on rewrite-based satisfiability procedures. ACM Transactions on Computational Logic 10(1), 4:1–4:51 (2009)
4. Bachmair, L., Ganzinger, H.: Rewrite-based equational theorem proving with selection and simplification. Journal of Logic and Computation 4(3), 217–247 (1994)
5. Bachmair, L., Ganzinger, H., Waldmann, U.: Refutational theorem proving for hierarchic first-order theories. Appl. Algebra Eng. Commun. Comput. 5, 193–212 (1994)
6. Baumgartner, P., Fuchs, A., de Nivelle, H., Tinelli, C.: Computing finite models by reduction to function-free clause logic. In: Ahrendt, W., Baumgartner, P., de Nivelle, H. (eds.) Proceedings of the Third Workshop on Disproving, pp. 82–99 (2006)
7. Baumgartner, P., Furbach, U., Pelzer, B.: The hyper tableaux calculus with equality and an application to finite model computation. Journal of Logic and Computation 20(1), 77–109 (2010)
8. Baumgartner, P., Schmidt, R.A.: Blocking and Other Enhancements for Bottom-Up Model Generation Methods. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 125–139. Springer, Heidelberg (2006)
9. Bernays, P., Schönfinkel, M.: Zum Entscheidungsproblem der mathematischen Logik. Mathematische Annalen 99, 342–372 (1928)
10. Blanchette, J.C., Popescu, A., Wand, D., Weidenbach, C.: More SPASS with Isabelle—Superposition with Hard Sorts and Configurable Simplification. In: Beringer, L., Felty, A. (eds.) ITP 2012. LNCS, vol. 7406, pp. 345–360. Springer, Heidelberg (2012), `http://www4.in.tum.de/~blanchet/more-spass.pdf`
11. Bonacina, M.P., Ghilardi, S., Nicolini, E., Ranise, S., Zucchelli, D.: Decidability and Undecidability Results for Nelson-Oppen and Rewrite-Based Decision Procedures. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 513–527. Springer, Heidelberg (2006)
12. Bonacina, M.P., Lynch, C., Mendonça de Moura, L.: On deciding satisfiability by theorem proving with speculative inferences. Journal of Automated Reasoning 47(2), 161–189 (2011)

[2] Actually, in personal communication, one of the authors discussed this idea with Bill.

13. Claessen, K., Sörensson, N.: New techniques that improve MACE-style finite model finding. In: Baumgartner, P., Fermueller, C. (eds.) Proceedings of the Workshop on Model Computation (2003)
14. de Nivelle, H., Meng, J.: Geometric Resolution: A Proof Procedure Based on Finite Model Search. In: Furbach, U., Shankar, N. (eds.) IJCAR 2006. LNCS (LNAI), vol. 4130, pp. 303–317. Springer, Heidelberg (2006)
15. Dershowitz, N.: A Maximal-Literal Unit Strategy for Horn Clauses. In: Okada, M., Kaplan, S. (eds.) CTRS 1990. LNCS, vol. 516, pp. 14–25. Springer, Heidelberg (1991)
16. Fietzke, A., Weidenbach, C.: Labelled splitting. Annals of Mathematics and Artificial Intellelligence 55(1-2), 3–34 (2009)
17. Fietzke, A., Weidenbach, C.: Superposition as a decision procedure for timed automata. In: Ratschan, S. (ed.) MACIS 2011: Fourth International Conference on Mathematical Aspects of Computer and Information Sciences, pp. 52–62 (2011); Journal version to appear in the Journal of Mathematics in Computer Science
18. Fontaine, P., Merz, S., Weidenbach, C.: Combination of Disjoint Theories: Beyond Decidability. In: Gramlich, B., Miller, D., Sattler, U. (eds.) IJCAR 2012. LNCS, vol. 7364, pp. 256–270. Springer, Heidelberg (2012)
19. Ganzinger, H., Meyer, C., Weidenbach, C.: Soft Typing for Ordered Resolution. In: McCune, W. (ed.) CADE 1997. LNCS, vol. 1249, pp. 321–335. Springer, Heidelberg (1997)
20. Hillenbrand, T., Topic, D., Weidenbach, C.: Sudokus as logical puzzles. In: Ahrendt, W., Baumgartner, P., de Nivelle, H. (eds.) Proceedings of the Third Workshop on Disproving, pp. 2–12 (2006)
21. Hillenbrand, T., Weidenbach, C.: Superposition for finite domains. Research Report MPI-I-2007-RG1-002, Max-Planck-Institut für Informatik, Saarbrücken (2007), `http://www.mpi-inf.mpg.de/~hillen/documents/HW07.ps`
22. Kamin, S., Levy, J.-J.: Attempts for generalizing the recursive path orderings. University of Illinois, Department of Computer Science. Unpublished note (1980), Available electronically from `http://perso.ens-lyon.fr/pierre.lescanne/not_accessible.html`
23. Kirchner, H., Ranise, S., Ringeissen, C., Tran, D.-K.: On Superposition-Based Satisfiability Procedures and Their Combination. In: Van Hung, D., Wirsing, M. (eds.) ICTAC 2005. LNCS, vol. 3722, pp. 594–608. Springer, Heidelberg (2005)
24. Manthey, R., Bry, F.: Satchmo: A Theorem Prover Implemented in Prolog. In: Lusk, E., Overbeek, R. (eds.) CADE 1988. LNCS, vol. 310, pp. 415–434. Springer, Heidelberg (1988)
25. McCune, W.: Mace4 reference manual and guide. Technical Report ANL/MCS-TM-264, Argonne National Laboratory (2003)
26. McCune, W.: Prover9 and mace4 (2005-2010), `http://www.cs.unm.edu/~ccune/prover9/`
27. McCune, W.: Otter 3.3 reference manual. CoRR, cs.SC/0310056 (2003)
28. Minsky, M.L.: Computation: Finite and Infinite Machines. Automatic Computation. Prentice-Hall (1967)
29. Nieuwenhuis, R., Rubio, A.: Paramodulation-based theorem proving. In: Robinson, A., Voronkov, A. (eds.) Handbook of Automated Reasoning, vol. I, ch. 7, pp. 371–443. Elsevier (2001)
30. Nieuwenhuis, R., Oliveras, A., Tinelli, C.: Solving SAT and SAT modulo theories: From an abstract Davis–Putnam–Logemann–Loveland procedure to DPLL(T). Journal of the ACM 53, 937–977 (2006)

31. Paulson, L.C., Blanchette, J.C.: Three years of experience with Sledgehammer, a practical link between automatic and interactive theorem provers. In: Sutcliffe, G., Ternovska, E., Schulz, S. (eds.) Proceedings of the 8th International Workshop on the Implementation of Logics (2010)
32. Navarro, J.A., Voronkov, A.: Proof Systems for Effectively Propositional Logic. In: Armando, A., Baumgartner, P., Dowek, G. (eds.) IJCAR 2008. LNCS (LNAI), vol. 5195, pp. 426–440. Springer, Heidelberg (2008)
33. Schulz, S., Bonacina, M.P.: On Handling Distinct Objects in the Superposition Calculus. In: Konev, B., Schulz, S. (eds.) Proc. of the 5th International Workshop on the Implementation of Logics, Montevideo, Uruguay, pp. 66–77 (2005)
34. Slaney, J.: FINDER: Finite Domain Enumerator. In: Bundy, A. (ed.) CADE 1994. LNCS, vol. 814, pp. 798–801. Springer, Heidelberg (1994)
35. Suda, M., Weidenbach, C., Wischnewski, P.: On the Saturation of YAGO. In: Giesl, J., Hähnle, R. (eds.) IJCAR 2010. LNCS, vol. 6173, pp. 441–456. Springer, Heidelberg (2010)
36. Weidenbach, C.: Combining superposition, sorts and splitting. In: Robinson, A., Voronkov, A. (eds.) Handbook of Automated Reasoning, vol. II, ch. 27, pp. 1965–2012. Elsevier (2001)
37. Weidenbach, C., Dimova, D., Fietzke, A., Kumar, R., Suda, M., Wischnewski, P.: SPASS Version 3.5. In: Schmidt, R.A. (ed.) CADE 2009. LNCS, vol. 5663, pp. 140–145. Springer, Heidelberg (2009)
38. Wenzel, M., Paulson, L.C., Nipkow, T.: The Isabelle Framework. In: Mohamed, O.A., Muñoz, C., Tahar, S. (eds.) TPHOLs 2008. LNCS, vol. 5170, pp. 33–38. Springer, Heidelberg (2008)
39. Zhang, J., Zhang, H.: SEM: a system for enumerating models. In: Proceedings of the 14th International Joint Conference on Artificial Intelligence, vol. 1, pp. 298–303. Morgan Kaufmann (1995)

Appendix: Proofs

Proving Proposition 2

Proposition 2. A clause set M is $\mathcal{T}$-satisfiable iff $\Omega(M \cup \mathcal{T}')$ is satisfiable.

Proof. On the one hand, since $M, \mathcal{T} \models \Omega(M \cup \mathcal{T}')$, every $\mathcal{T}$-model of M is a model of $\Omega(M \cup \mathcal{T}')$ as well. On the other hand, consider any model $\mathcal{A}$ of $\Omega(M \cup \mathcal{T}')$. Its restriction to $\{1^{\mathcal{A}}, \ldots, n^{\mathcal{A}}\}$ is a Σ-algebra because of the range restriction on the functions, and it is a $\mathcal{T}$-model by construction. Finally every clause C is $\mathcal{T}$-equivalent to $\bigwedge \Omega(C)$.

Proving Lemma 6

Proposition 6.1. Let N denote a node in a derivation, with successors $N_1, \ldots, N_k$. If $\Omega(N)$ is satisfiable, so is some $\Omega(N_i)$.

Proof. According to the type of calculus step, we distinguish three cases.

– An inference: Here k equals 1, and N_1 is $N \cup \{C\}$ where C is N-valid. Hence N and N_1 are even equivalent.

- A simplification adhering to the form $\mathcal{R}\frac{C}{\vec{D}}N'$: Again k is 1, but N has a presentation $N = \{C\} \cup N' \cup N''$ such that $N_1 = \{\vec{D}\} \cup N' \cup N''$. The side conditions imply $\Omega(N') \models (\bigwedge \Omega(C)) \leftrightarrow (\bigwedge \Omega(\vec{D}))$, such that the clause sets $\Omega(N)$ and $\Omega(N_1)$ are equivalent.
- A split: In our concrete split rule k equals 2. Let $C' \equiv (C \vee s \simeq t)\tau$ and $D' \equiv (l \simeq r \vee D)\tau$ denote the first and the second conclusion, respectively. Then $C' \vee D'$ is N-valid, and the disjuncts share no variables. If $\mathcal{A}$ is an N-model, then $\mathcal{A}$ satisfies at least one of C' and D', and therefore at least one of $N_1 = N \cup \{C'\}$ and $N_2 = N \cup \{D'\}$.

Proposition 6.2. For every clause set M, the following are equivalent:
(i) M is $\mathcal{T}$-satisfiable.
(ii) Every derivation from $M \cup \mathcal{T}'$ contains a complete path $N_1, N_2, \ldots$ such that every $\Omega(N_i)$ is satisfiable.

Proof. If M is $\mathcal{T}$-satisfiable, then by Prop. 2 the set $\Omega(N_1) = \Omega(M \cup \mathcal{T}')$ is satisfiable, from which we can recursively construct a complete path as required by Prop. 6.1. The converse implication follows from $N_1 = M \cup \mathcal{T}$ by Prop. 2.

If a clause C occurs at some point in a path, then the limit N_∞ entails each of its Ω-instances from smaller or equal Ω-instances. Furthermore satisfiability of N_∞ with respect to Ω-instances is the conjunction of this property over all path elements.

Proposition 6.3. Consider a complete path $N_1, N_2, \ldots$ in some derivation.
(i) If $C \in N_i$ is ground numbered by ρ, then $\Omega(N_\infty)^{\preceq C\rho} \models C\rho$ holds, as well as $\Omega(N_j)^{\preceq C\rho} \models C\rho$ for every $j \geq i$.
(ii) Every $\Omega(N_i)$ is satisfiable iff $\Omega(N_\infty)$ is.
(iii) N_∞ is saturated in case the derivation is fair.

Proof.
(i) The proof is by induction on $C\rho$ with respect to $\succ$. Let j denote ∞ or a natural number greater than or equal to i. If $C \in N_j$ we are done. Otherwise there is an index k between i and j such that C is contained in N_i through N_k, but not in N_{k+1}. By definition of simplification we have $\Omega(\vec{D}, M)^{\prec C\rho} \models C\rho$ for appropriate $\vec{D}, M \subseteq N_{k+1}$. Either $\vec{D}, M$ is empty and $C\rho$ is a tautology, or there is a greatest clause D' in $\Omega(\vec{D}, M)^{\prec C\rho}$. Inductively all elements of $\Omega(\vec{D}, M)^{\prec C\rho}$ are valid in $\Omega(N_j)^{\preceq D'}$, and so is $C\rho$.
(ii) Assume that every $\Omega(N_i)$ is satisfiable. By compactness $\Omega(N_\infty)$ is satisfiable iff each of its finite subsets is. Given one such subset M, for every Ω-instance $C\rho$ within there is an index j such that C is contained in N_j and all successors thereof. Since M is finite, these indices have a finite maximum k. Now $\Omega(N_k)$ comprises M and is satisfiable by assumption.
As to the converse implication, consider an Ω-instance $C\rho$ of a clause $C \in N_i$. Then $\Omega(N_\infty)$ entails $C\rho$ by Prop. 6.3 (i). In other words, any model of $\Omega(N_\infty)$ is a model of $\Omega(N_i)$.

(iii) Firstly we consider an inference with premises $\vec{C}$ from N_∞ and conclusion D with ground numbering substitution ρ. Because of fairness $\Omega(N_i)^{\prec \max\{\vec{C}\rho\}} \models D\rho$ holds for some i, which can be rephrased as $C'_1\rho_1, \ldots, C'_k\rho_k \models D\rho$ for clause instances $C'_j\rho_j$ from $\Omega(N_i)$ below $\max\{\vec{C}\rho\}$. By Prop. 6.3 (i) these clause instances are valid in $\Omega(N_\infty)$ below $\max\{\vec{C}\rho\}$, and so is $D\rho$.
Secondly we study a split from a persistent clause $C \equiv C_1 \vee C_2$ with designated partitioning as indicated and minimally numbering substitution τ. Because of fairness, one split conjunct, say $C_1\tau$, is contained in some N_i or redundant with respect to it. So either $C_1\tau$ is persistent, or $C_1\tau$ is redundant with respect to some N_j where $j \geq i$. In the former case the proof is finished. In the latter we have $\Omega(N_j)^{\prec C_1\rho} \models C_1\rho$ for every ground numbering $\rho = \tau\tau'$, which extends to $\Omega(N_\infty)^{\prec C_1\rho} \models C_1\rho$ with an argument like in the preceding paragraph.

For any clause set M, by $\widehat{M}$ is denoted the set of its Ω-instances which are Horn clauses.

Proposition 6.4. $\Omega(M)$ and $\widehat{M}$ are equivalent for $\mathcal{C}$-saturated clause sets M.

Proof. We show by induction on clause instances that every non-Horn clause $C\rho \in \Omega(M)$ is entailed by $\widehat{M}$. Now, C has a presentation $C \equiv C_1 \vee C_2$ such that the partitioning into C_1 and C_2 is designated. Then ρ numbers the clause C such that the subclauses C_1 and C_2 are variable disjoint. More general such substitutions τ have to satisfy $\tau \subseteq \rho$. There exists a $\subset$-minimal such τ because all descending $\subset$-chains are finite. Then $C \vdash C_1\tau \mid C_2\tau$ is a valid $\mathcal{C}$-split. Because M is saturated, one split conjunct, say $C_1\tau$, is contained in M or redundant with respect to M. In both cases we have $\Omega(M) \models C_1\rho$, and we obtain inductively $\widehat{M} \models C_1\rho$. Finally $C_1\rho$ entails $C\rho$.

The calculus $\mathcal{C}$ is sound and refutationally complete:

Lemma 6. For every clause set M, the following are equivalent:

(i) M is $\mathcal{T}$-satisfiable.

(ii) Every fair derivation from $M \cup \mathcal{T}'$ contains a complete path $N_1, N_2, \ldots$ such that the empty clause is not in N_∞.

Proof. We successively transform the first characterization into the second. By Prop. 6.2 the clause set M is $\mathcal{T}$-satisfiable iff there exists a complete path N_1, N_2, … such that every $\Omega(N_i)$ is satisfiable, or such that $\Omega(N_\infty)$ is, by Prop. 6.3 (ii). Because of Prop. 6.3 (iii) every N_∞ is saturated with respect to $\mathcal{C}$. Hence by Prop. 6.4 the sets $\Omega(N_\infty)$ and $\widehat{N_\infty}$ are equivalent, and the latter is saturated with respect to $\mathcal{G}$ by Prop. 6.4. Since $\mathcal{G}$ is sound and complete, the satisfiability of $\widehat{N_\infty}$ is equivalent to $\bot \notin \widehat{N_\infty}$, which is the same as $\bot \notin N_\infty$.

Proving Lemma 4

We now set out to prove that a ground instance $C\sigma$ of a clause C follows from $\Omega(C, \mathcal{T}')$, and give a criterion when this entailment is from smaller instances.

Proposition 4.5. For every clause C and term t, the following entailment holds: $C\{x \mapsto 1\}, \ldots, C\{x \mapsto n\}, \mathrm{Dig}(t) \models C\{x \mapsto t\}$

Proof. Consider a model $\mathcal{A}$ of the premises. Then there exists a digit i fulfilling $\mathcal{A} \models t \simeq i$. This identity inductively lifts to term contexts, and as equivalence to clause contexts. In particular $\mathcal{A} \models C\{x \mapsto i\}$ implies $\mathcal{A} \models C\{x \mapsto t\}$.

Proposition 4.6. Let C denote a clause with ground substitution $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$. Then $\Omega(C), \mathrm{Dig}(t_1), \ldots, \mathrm{Dig}(t_m) \models C\sigma$ holds.

Proof. The proof is by induction on m. If σ is the identity we are done. Otherwise we decompose σ according to $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\} \cup \{x_{m+1} \mapsto t_{m+1}\} = \sigma_1 \cup \sigma_2$. Since the substitutions are ground we have $\sigma_1 \cup \sigma_2 = \sigma_1 \circ \sigma_2$. Inductively we obtain $\Omega(C\sigma_1), \mathrm{Dig}(t_1), \ldots, \mathrm{Dig}(t_m) \models C\sigma_1$. Proposition 4.5 gives $C\sigma_1, \mathrm{Dig}(t_{m+1}) \models C\sigma_1\sigma_2$.

Proposition 4.7. Ground terms t obey $\Omega(\mathcal{T}') \models \mathrm{Dig}(t)$.

Proof. We induct on the structure of t. In case $t \equiv i$ the clause $\mathrm{Dig}(t)$ is a tautology. In case $t \equiv f(\vec{t})$ the proposition $\Omega(\mathcal{T}') \models \mathrm{Dig}(t_j)$ is inductively true for every j. Furthermore $\mathcal{T}'$ contains $\mathrm{Dig}(f(\vec{x}))$. Let $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$, such that $f(\vec{t}) \equiv f(\vec{x})\sigma$. With Prop. 4.6 we obtain $\Omega(\mathrm{Dig}(f(\vec{x}))), \mathrm{Dig}(t_1), \ldots, \mathrm{Dig}(t_m) \models \mathrm{Dig}(f(\vec{x}))\sigma$.

Proposition 4.8. $\Omega(C, \mathcal{T}') \models C\sigma$ is true for every clause C with ground substitution σ.

Proof. Assume $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$. Then Prop. 4.7 implies that $\Omega(\mathcal{T}') \models \mathrm{Dig}(t_i)$ holds for every i, such that from Prop. 4.6 finally we obtain $\Omega(C), \mathrm{Dig}(t_1), \ldots, \mathrm{Dig}(t_m) \models C\sigma$.

We have seen in Prop. 4.7 that every ground term t is subject to $\Omega(\mathcal{T}') \models \mathrm{Dig}(t)$. In the following we will exploit that usually not all of $\Omega(\mathcal{T}')$ is needed for this entailment. There exist subsets $T \subseteq \Omega(\mathcal{T}')$ such that $T \models \mathrm{Dig}(t)$ holds. By compactness there are finite such T even in case the signature is infinite. Let $\Delta(t)$ denote the smallest of these finite T, with respect to the ordering on clause sets. Let furthermore $\delta(t)$ denote the greatest clause in $\Delta(t) \cup \{\bot\}$, and for ground substitutions σ let $\delta(\sigma)$ stand for the greatest clause in $\delta(\operatorname{ran}\sigma) \cup \{\bot\}$. Actually one can construct $\Delta(t)$ recursively, but this is not necessary for our purposes.

Proposition 4.9. Entailment from $\Omega(\mathcal{T}')$ can be restricted by the bounds $\delta(t)$ and $\delta(\sigma)$:

(i) Every ground term t satisfies $\Omega(\mathcal{T}')^{\preceq\delta(t)} \models \mathrm{Dig}(t)$.

(ii) If σ is a ground substitution for C, then $\Omega(C), \Omega(\mathcal{T}')^{\preceq\delta(\sigma)} \models C\sigma$ holds.

Proof.

(i) By definition we have $\Delta(t) \subseteq \Omega(\mathcal{T}')^{\preceq\delta(t)}$ and $\Delta(t) \models \mathrm{Dig}(t)$.

(ii) Let $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$. Then we obtain $\Omega(\mathcal{T}')^{\preceq \delta(t_i)} \models \mathrm{Dig}(t_i)$ from Prop. 4.9 (i) for every i, and $\Omega(\mathcal{T}')^{\preceq \delta(\sigma)} \models \mathrm{Dig}(t_i)$ by definition of $\delta(\sigma)$. Finally we apply Prop. 4.6 to C and σ.

Proposition 4.10. For ground terms t we have $\delta(t) \equiv \bot$ iff t is a digit.

Proof. In case t is a digit, then $\mathrm{Dig}(t)$ is a tautology and $\Delta(t)$ is empty. Otherwise $\mathrm{Dig}(t)$ is not a tautology.

Proposition 4.11. If t is a ground term and δ a ground substitution, then we can give estimates for $\delta(t)$ and $\delta(\sigma)$ as follows:

(i) $\delta(t) \equiv \mathrm{Dig}(u)$ implies $t \succeq u$.

(ii) $\delta(\sigma) \equiv \mathrm{Dig}(u)$ entails $\max(\mathrm{ran}\,\sigma) \succeq u$.

Proof.

(i) The proof is by induction on the term structure. If t is a digit, then we have $\delta(t) \equiv \bot$ by Prop. 4.10, and there is nothing to show. The case $t \equiv f(\vec{t})$ remains. Let $i_1, \ldots, i_k$ denote exactly the indices for which t_j is not a digit, and let $t' \equiv f(\vec{t})[x_1]_{i_1} \ldots [x_k]_{i_k}$. So t' is obtained from t replacing every non-digit t_j with a fresh variable. Conversely, using $\sigma = \{x_1 \mapsto t_{i_1}, \ldots, x_k \mapsto t_{i_k}\}$ one can instantiate t' back into t again.
In case $k = 0$ the argument vector $\vec{t}$ contains only digits. Choosing $T = \{\mathrm{Dig}(t)\}$ implies $T \subseteq \Omega(\mathcal{T}')$ and $T \models \mathrm{Dig}(t)$. Therefore we have $T \succeq \Delta(t)$ and $\max T \succeq \max \Delta(t) \equiv \delta(t)$, hence $\mathrm{Dig}(t) \succeq \mathrm{Dig}(u)$ and finally $t \succeq u$.
In case $k > 0$ every $\delta(t_{i_j})$ is distinct from $\bot$ by Prop. 4.10, and there exists a ground term v such that $\mathrm{Dig}(v) \equiv \max_j \delta(t_{i_j})$. By induction hypothesis and the subterm property of t we obtain $t \succ v$. Here we choose $T = \Omega(\mathrm{Dig}(t')) \cup \Omega(\mathcal{T}')^{\preceq \mathrm{Dig}(v)}$, which satisfies $T \subseteq \Omega(\mathcal{T}')$. By construction $T \models \mathrm{Dig}(t_{i_j})$ holds for every j. Proposition 4.6 yields $\Omega(\mathrm{Dig}(t')), \mathrm{Dig}(t_{i_1}), \ldots, \mathrm{Dig}(t_{i_k}) \models \mathrm{Dig}(t'\sigma)$. Hence we may conclude that $T \succeq \Delta(t)$ and $\max T \succeq \mathrm{Dig}(u)$. Next we compare T with $\{\mathrm{Dig}(t)\}$. We have $\Omega(\mathrm{Dig}(t')) \prec \{\mathrm{Dig}(t)\}$ by minimality of the digits, and furthermore $\Omega(\mathcal{T}')^{\preceq \mathrm{Dig}(v)} \prec \{\mathrm{Dig}(t)\}$ because of $v \prec t$. Hence we obtain that $\mathrm{Dig}(t) \succ \max T \succeq \mathrm{Dig}(u)$ holds, such that $t \succ u$ is true.

(ii) Let $\sigma = \{x_1 \mapsto t_1, \ldots, x_m \mapsto t_m\}$. Because of $\delta(\sigma) \not\equiv \bot$ we have $\delta(\sigma) \equiv t_i$ for some i. Using Prop. 4.11 (i) we may conclude that $\max_j t_j \succeq t_i \succeq u$ holds.

□

Proposition 4.12. Let C denote a clause with ground substitution σ such that σ is not numbering, and that C, σ is unproblematic. Then $\Omega(C, \mathcal{T}')^{\prec C\sigma} \models C\sigma$ holds.

Proof. We decompose $\sigma = \sigma_1 \cup \sigma_2$ such that the range of σ_1 contains only digits and the range of σ_2 only non-digits. Since the substitutions are ground we have $\sigma = \sigma_1 \circ \sigma_2$. Proposition 4.9 (ii) implies $\Omega(C\sigma_1), \Omega(\mathcal{T}')^{\preceq \delta(\sigma_2)} \models C\sigma_1\sigma_2$. The substitution σ_2 is not empty because σ is not numbering. Hence we have by minimality of the digits $\Omega(C\sigma_1) \prec \{C\sigma_1\sigma_2\}$. We still have to show $\delta(\sigma_2) \prec C\sigma$.

Let t denote the greatest term in $\operatorname{ran}\sigma_2$. By Prop. 4.10 the clause $\delta(t)$ equals $\mathrm{Dig}(f(\vec{\imath}))$ for some term $f(\vec{\imath})$. By Prop. 4.11 (ii) we have $t \succeq f(\vec{\imath})$. If $t \succ f(\vec{\imath})$, then the greatest term of $C\sigma$ is above the greatest of $\delta(\sigma_2)$. Otherwise we obtain $C\sigma \succ \mathrm{Dig}(f(\vec{\imath}))$ from the requirement that C, σ is unproblematic.

Lemma 4. Consider a path in a $\mathcal{C}$-derivation from $M \cup \mathcal{T}'$ to N and a clause C. Then C is redundant with respect to N if one of the following conditions holds, where ρ ranges over all ground numbering substitutions:

(i) $\mathrm{g\mathring{n}d}(N)^{\prec C\rho} \models C\rho$ for all ρ,
(ii) $\mathrm{gnd}(N)^{\prec C\rho} \models C\rho$ for all ρ and C is noncritical.

Proof.

(i) Given an arbitrary ground numbering substitution ρ, there exist clauses $D_1, \ldots, D_m \in N$ and ground substitutions $\sigma_1, \ldots, \sigma_m$ such that every D_i, σ_i is unproblematic and $D_i\sigma_i \prec C\rho$, and that $D_1\sigma_1, \ldots, D_m\sigma_m \models C\rho$. In order to prove $\Omega(N)^{\prec C\rho} \models C\rho$ it suffices to show that $\Omega(N)^{\prec C\rho} \models D_i\sigma_i$ holds for every i. If $D_i\sigma_i$ is a digit instance of D_i, then we have $D_i\sigma_i \in \Omega(N)^{\prec C\rho}$. Otherwise Prop. 4.12 ensures $\Omega(D_i, \mathcal{T}')^{\prec D_i\sigma_i} \models D_i\sigma_i$ because D_i, σ_i is unproblematic. With Prop. 6.3 (i) we get $\Omega(N)^{\preceq D_i\sigma_i} \models D_i\sigma_i$, and therefore $\Omega(N)^{\prec C\rho} \models D_i\sigma_i$.

(ii) Similar to the proof of Lem. 4 (i), for every ground numbering substitution ρ there exist clauses $D_1, \ldots, D_m \in N$ and ground substitutions $\sigma_1, \ldots, \sigma_m$ such that always $D_i\sigma_i \prec C\rho$, and that $D_1\sigma_1, \ldots, D_m\sigma_m \models C\rho$. If $C\rho$ is a tautology we are done. Otherwise we decompose every $\sigma_k = \sigma_k' \cup \sigma_k''$ such that the range of σ_k' contains only digits and the range of σ_k'' only non-digits. Proposition 4.9 (ii) guarantees that $\Omega(D_k\sigma_k'), \Omega(\mathcal{T}')^{\preceq\delta(\sigma_k'')} \models D_k\sigma_k$. By minimality of the digits we obtain $\Omega(D_k\sigma_k') \preceq \{D_k\sigma_k\} \prec \{C\rho\}$.
Next we show that $\delta(\sigma_k'') \prec C\rho$. The clause C is not empty since otherwise $\models \bot$; so $C\rho$ has a greatest term s. Let t denote the greatest term of $D_k\sigma_k$, then we have $s \succeq t$. If $\delta(\sigma_k'') \equiv \bot$ then $\bot \prec C\rho$. Otherwise $\delta(\sigma_k'')$ has the shape $\mathrm{Dig}(f(\vec{\imath}))$. Because of Prop. 4.11 (ii) we have $\max(\operatorname{ran}\sigma_k'') \succeq f(\vec{\imath})$, and because of $t \succeq \max(\operatorname{ran}\sigma_k'')$ we have $s \succeq f(\vec{\imath})$ as well. Now $s \succ f(\vec{\imath})$ directly entails $C\rho \succ \delta(\sigma_k'') \equiv \mathrm{Dig}(f(\vec{\imath}))$. Otherwise s equals $f(\vec{\imath})$, and $C\rho \succ \mathrm{Dig}(f(\vec{\imath}))$ holds because C is noncritical by assumption.
Summing it up, we obtain $\Omega(D_k\sigma_k', \mathcal{T}')^{\prec C\rho} \models D_k\sigma_k$ and therefore as well $\Omega(D_k, \mathcal{T}')^{\prec C\rho} \models D_k\sigma_k$. Via Prop. 6.3 (i) we conclude $\Omega(N)^{\prec C\rho} \models D_k\sigma_k$.

Proving Proposition 7

Proposition 7. Inference and split conclusions do not have more variables than one of the premises.

(i) If $\sigma = \mathrm{mgu}(u, v)$ with $\operatorname{ran}\sigma \subseteq \mathscr{V} \cup [1; n]$ and $\operatorname{dom}\sigma \cup \operatorname{cdom}\sigma \subseteq \operatorname{var}(u, v)$, then there is a variant σ' that additionally satisfies $\operatorname{var}(v\sigma') \subseteq \operatorname{var}(v)$.
(ii) If $C \vdash D$ is a unary inference or a split, then $\operatorname{var}(D) \subseteq \operatorname{var}(C)$ holds.
(iii) If $l \simeq r,\ C \vdash D$ is a binary inference, then $|\operatorname{var}(D)| \leq |\operatorname{var}(C)|$ is true.

Proof.

(i) Let $\mathcal{P}(m)$ hold iff there exists an mgu σ of u and v with $\operatorname{ran}\sigma \subseteq \mathscr{V} \cup [1;n]$, $\operatorname{dom}\sigma \cup \operatorname{cdom}\sigma \subseteq \operatorname{var}(u,v)$, and $|\operatorname{var}(v\sigma) \setminus \operatorname{var}(v)| = m$. By assumption $\mathcal{P}$ holds for some $m \geq 0$. We will now show that $\mathcal{P}(j+1)$ implies $\mathcal{P}(j)$.
Assume σ is a witness for $\mathcal{P}(j+1)$. Because of $j+1 > 0$ there exists a variable y in $\operatorname{var}(v\sigma) \setminus \operatorname{var}(v)$. By the shape of σ, this variable is the σ-image of another variable $x \in \mathit{var}(v)$. Consider now the substitutions $\tau = \{x \mapsto y, y \mapsto x\}$ and $\sigma' = \sigma \circ \tau$. The latter is a unifier of u and v. Because of $\sigma'\tau = \sigma\tau^2 = \sigma$, it is even a most general one. The image of a variable z under σ' is x if $z\sigma \equiv y$, and $z\sigma$ otherwise; in particular $x\sigma' \equiv x$ and $y\sigma' \equiv x$. That is, going from σ to σ', the variable x moves from the dom-part to the cdom-part, and y in the opposite direction, which are all effects in terms of dom and cdom. The identity $\operatorname{var}(v\sigma') = (\operatorname{var}(v\sigma) \cup \{x\}) \setminus \{y\}$ concludes the proof of $\mathcal{P}(j)$.

(ii) In case of an equality resolution step $C \vee t \simeq t' \vdash C\sigma$ we have $\operatorname{cdom}\sigma \subseteq \operatorname{var}(t,t')$. Given a split $C \vee s \simeq t \vee l \simeq r \vee D \vdash (C \vee s \simeq t)\tau \mid (l \simeq r \vee D)\tau$, the substitution τ is numbering, such that $\operatorname{cdom}\tau \subseteq [1;n]$.

(iii) We will prove that $\operatorname{var}(D) \subseteq \operatorname{var}(C)$ holds in case the most general unifier is chosen according to Prop. 7 (i). All mgu's are equal up to variable renaming; and the number of variables in a clause is invariant under such renamings. This yields the estimate stated above.
We jointly treat superposition left and right inferences via the pattern $l \simeq r,\ C[l'] \vdash C[r]\sigma\tau \equiv D$. Because of $l'\sigma\tau \equiv l\sigma\tau \succ r\sigma\tau$ we know that $\operatorname{var}(l'\sigma\tau) \supseteq \operatorname{var}(r\sigma\tau)$ is true, and hence $\operatorname{var}(D) \subseteq \operatorname{var}(C\sigma\tau) \subseteq \operatorname{var}(C\sigma)$. Applying Prop. 7 (i), without loss of generality σ can be chosen such that $\operatorname{var}(l'\sigma) \subseteq \operatorname{var}(l')$. Let σ' denote the restriction of σ to $\operatorname{var}(l')$. By this definition we have $\operatorname{cdom}\sigma' \subseteq \operatorname{var}(l'\sigma) \subseteq \operatorname{var}(l') \subseteq \operatorname{var}(C)$. Since the premises are variable disjoint, we obtain $\operatorname{var}(C\sigma) = \operatorname{var}(C\sigma') \subseteq \operatorname{var}(C) \cup \operatorname{cdom}\sigma' = \operatorname{var}(C)$, which completes the proof of $\operatorname{var}(D) \subseteq \operatorname{var}(C)$.

Proving Theorem 12

Theorem 12. $\mathcal{C}_\kappa$-derivations decide $\mathcal{T}$-satisfiability of finite clause sets.

Proof. Consider a $\mathcal{C}_\kappa$-derivation from a finite clause set M. Then M by Lem. 6 is $\mathcal{T}$-satisfiable if and only if the derivation contains a complete path without the empty clause in the limit. The derivation tree is finitely branching. It remains to show that every path $N_1, N_2, \ldots$ is finite. Let $\|N_i\| = \max\{|\operatorname{var}(C)| \colon C \in N_i\}$.

There exists an index κ such that from N_κ on, the conditions (iii) through (vi) of the definition of $\mathcal{C}_\kappa$-derivation are satisfied. We form a subsequence of $N_1, N_2, \ldots$ that starts from $N'_1 = N_\kappa$. If in N_i a new clause C is inferred and, according to condition (vi), immediately simplified into $[1;n]$-shallow clauses $\vec{D}$ until N_{i+k}, then for $(N'_j)_j$ all sequence elements but N_{i+k} are dropped, and the latter shows up only if $\vec{D}$ is not empty. Assume now $(N_i)_i$ is infinite. Inferences with empty $\vec{D}$ are not repeated because of condition (v), as well as splits; so there must be infinitely many simplifications or inferences with non-empty $\vec{D}$.

Since simplifications are decreasing with respect to Ω-instances, the latter occur infinitely many times; so $(N'_j)_j$ is then infinite as well.

Inductively $\|N'_j\| \leq \|N'_1\|$ holds for all j: If a clause $C \in N'_j$ is simplified to some non-empty $\vec{D}$, then we know that $|\mathrm{var}(D_i)| \leq |\mathrm{var}(C)|$ by condition (iii). In case of a split or an inference, we additionally apply Prop. 7 (ii) and Prop. 7 (iii).

Assume now $(N'_j)_j$ were infinite; then we can argue like above for $(N_i)_i$ and obtain that infinitely many inferences are drawn. The inference conclusions are simplified according to condition (vi), such that they become $[1;n]$-shallow and have no more than $\|N'_1\|$ variables. Because of conditions (vii) and (viii), only finitely many such clauses exist. Moreover the number of clauses that are produced from simplification and splitting alone is finite. Therefore, eventually an inference has to be repeated, but this contradicts condition (v). Hence $(N'_j)_j$ is finite, and so is $(N_i)_i$.