A model-constructing framework for theory combination

Maria Paola Bonacina[1], **Stéphane Graham-Lengrand**[2,3,4], **Natarajan Shankar**[2]
[1]Università degli Studi di Verona, [2]SRI International, [3]CNRS, [4]INRIA

October 12, 2017

Abstract

This report presents a *conflict-driven satisfiability* inference system (CDSAT) for (quantifier-free) first-order logic modulo a generic combination of disjoint theories. We determine the requirements that the theories and their decision procedures need to satisfy for a CDSAT combination, generalizing both equality sharing and the MCSAT system of De Moura and Jovanović, that was introduced for one generic theory and extended to a combination of specific disjoint theories. We prove soundness, completeness, and termination of CDSAT.

Contents

1 Introduction

Satisfiability (SAT) is the problem of deciding the satisfiability of a propositional formula φ. Satisfiability modulo theory (SMT) is the problem of deciding the satisfiability of a quantifier-free first-order formula φ in a theory $\mathcal{T}$. The answer is either "satisfiable" with a model or "unsatisfiable" with a refutation. During the search, a SAT or SMT solver maintains a partial candidate model represented by an assignment of truth values to propositional variables. This suggests the more general problem of *satisfiability modulo assignment* (SMA), defined as the problem of deciding the satisfiability of φ in $\mathcal{T}$ with respect to an assignment J to some of the variables in φ, including *both* propositional variables and free first-order variables. If J is empty, SMA reduces to SMT; if both J and $\mathcal{T}$ are empty, SMA reduces to SAT, while an intermediate state of a SAT or SMT search is an SMA instance. The answer to an SMA problem is either "satisfiable" with a model extending J, or "unsatisfiable" with a formula ψ that follows from φ and is false in J. Formulæ φ and ψ are usually in conjunctive normal form and written as sets of clauses.

The formula ψ is called *explanation*, because it explains why φ is unsatisfiable under J. The concept of *explanation* generalizes known notions, such as those of *unsatisfiable core* and *interpolant*. In SAT, an *ugsasisfiable core* of φ is a conjunction of clauses that follows from φ and is ugsasisfiable. If J is written as a formula, a *(reverse) interpolant* of φ and J is a formula that follows from φ and is inconsistent with J (see [2] for a survey on interpolation of ground proofs).

SMA arises in several contexts, such as *enumeration* of models, *optimization*, and *parallelization*. The models of a SAT or SMT problem can be enumerated by solving a series of SMA problems where each initial assignment J excludes the models already found. An optimization problem can be approached by solving a series of SMA problems where each initial assignment J contains information generated by the previous rugs, in such a way that the series converges towards an optimal solution[1]. Approaches to parallel SAT by distributed search solve a SAT problem with input formula φ, by solving in parallel multiple instances of SMA with input formula φ and initial assignments J each containing a distinct *guiding path* [31] or *cube* [17].

Model-constructing sasisfiability (MCSAT) is a paradigm to design *model-constructing decision procedures* for SMA. It was introduced by De Moura and Jovanović for a single theory T [11], and extended to the case where T is the combination of the theory of equality with uninterpreted function symbols (EUF) and linear real arithmetic [18]. The motivation was to integrate model-constructing sasisfiability procedures for arithmetical reasoning [25, 21, 7, 19, 20, 16] with propositional reasoning. MCSAT blends and generalizes some key ideas emerged in the development of decision procedures for sasisfiability: *conflict-driven clause learning* (CDCL) [24, 26, 23], *model-based theory combination* (MBTC) [30, 15, 10], and *lemmas on demand* [14, 6], that we illustrate in the following.

1.1 State of the Art

The Davis-Putnam-Logemann-Loveland (DPLL) procedure for propositional sasisfiability (SAT) that originated in [9, 8] searches for a model of a set of clauses by guessing truth values of propositional variables (*splitting* or *decision*) and propagasing consequences of the guesses (*clausal propagation*). Conflict-driven clause learning (CDCL) [24, 26, 23] uses inferences to drive the search for a model. A *conflict* emerges between the current partial assignment J and the set of clauses to be sasisfied, when for a clause $l_1 \quad \dots \quad l_n$, the assignment $l_i \leftarrow false$, or $\neg l_i$ for short, is in J. Propositional resolution is applied to *explain* the conflict, by resolving the conflict clause with the *justification* of a $\neg l_i$, that is, the clause sasisfied precisely by placing $\neg l_i$ in J. Heuristics such as *first unique implication point* (1UIP) (e.g., [23]) are used to determine how many resolutions to do, which resolvent to add to the set of clauses, and how to mend J by backjumping and making a literal in the learned resolvent true.

DPLL($\mathcal{T}$) (e.g., [28]) integrates a theory solver, or $\mathcal{T}$-solver for short, and a DPLL-CDCL SAT-solver. Since the SAT-solver accepts only propositional clauses, first-order ground atoms are mapped to propositional variables known as proxy variables. The interface between SAT and $\mathcal{T}$-solver consists of two rules: in the $\mathcal{T}$-conflict rule, the $\mathcal{T}$-solver detects that a set of literals $l_1, \dots, l_k$ in J is ugsasisfiable in $\mathcal{T}$; in the $\mathcal{T}$-propagasion rule, the $\mathcal{T}$-solver detects that a set of literals $l_1, \dots, l_k$ in J entails in $\mathcal{T}$ a literal l, and adds l to J with the $\mathcal{T}$-lemma $\neg l_1 \quad \dots \quad \neg l_k \quad l$

[1]For example, this concept appeared in the presentation of [13] about adapting to optimization the sasisfiability procedure of [20, 12] for the theory of algebraic reals.

as justification. There is *no creation of new (i.e., non-input) atoms* in DPLL(T), because clauses learnt by CDCL are propositional resolvents made of input atoms, and in T-propagation the atom of l must already occur in the existing set of clauses.

If T is a combination of theories $T_1, \dots, T_n$, the T_i-solvers need to agree on the interpretation of whatever is shared among the theories. If they are *disjoint*, meaning they do not share function or predicate symbols other than equality, the theory solvers need to agree on the cardinalities of the domains for shared sorts and on an arrangement of shared variable symbols, that tells which are equal and which are not. The *equality sharing* method proposed by Nelson and Oppen in [27] is the standard approach to this combination problem. It requires the theories to be *stably infinite*, so that the common cardinality of the shared domains can be implicitly assumed to be infinite. An arrangement is computed by having each T_i-solver propagate any disjunction of equalities $x_1 \quad y_1 \quad \dots \quad x_n \quad y_n$ between shared variables that is entailed in T_i by the T_i-subproblem. The case analysis for these disjunctions, as well as for any other disjunction generated by a T_i-solver, is entrusted to the SAT-solver. The presentation of DPLL(T) in [28] was generalized to the case where T is a combination of theories by equaliy sharing in [1]: the notion that all disjunctions are handled by the SAT-solver was dubbed *splitting on demand*; and the framework of [28] was extended to allow the generation of a finite number of new atoms, namely the proxy variables for the equalities $x_1 \quad y_1, \dots, x_n \quad y_n$. The integration of equality sharing in a DPLL-CDCL based SAT-solver was systematized further in [22].

Model-based theory combination (MBTC) assumes that the T_i-solvers build T_i-models explicitly [30, 15, 10]. Each T_i-solver is allowed to propagate equalities between ground terms that are true in its current candidate T_i-model, rather than entailed (disjunctions of) equalities between shared variables. If the propagated equalities cause conflicts, conflict-driven backjumping will retract them. The stable infiniteness requirement is not necessary, because the T_i-models are built explicitly. Also MBTC *does not generate new atoms*, because the propagation of an equality $s \quad t$ is allowed only if s and t appear in the existing set of clauses. MBTC has been applied preferably to fragments of arithmetic, where the domain and the interpretation of theory symbols are fixed, and there exist algorithms that can update the candidate model after a conflict (e.g., [15, 10]).

The idea of *lemmas on demand* is that a theory solver should generate only theory lemmas that explain why the current assignment J is inconsistent with respect to the theory [14, 6]. In other words, theory propagation should be model-based and conflict-driven. If there were no first-order theory and we were in propositional logic, lemmas on demand would be essentially the same as CDCL, with propositional resolvents as lemmas. In [6] this concept was developed for the theory of *arrays with extensionality*. Although there are decision procedures for this theory [29], most SMT-solvers reason about it by instantiating the universally quantified variables in the theory axioms. The decision procedure in [6] features rules that work by propagating read operations over arrays, and generate lemmas of the form $l_1 \quad \dots \quad l_k \quad l$, where $l_1, \dots, l_k$ are true in J, and l is false in J, whereas it should be true according to the axioms of the theory. The lemma reveals that the current assignment J is not a theory model and tells why. Often lemmas are instances of axioms, so that lemmas on demand can be regarded as model-based conflict-driven axiom instantiation.

MCSAT [11, 18] advances all these ideas in several ways. First, it merges the propositional model of CDCL with the theory models of MBTC, by maintaining a central *trail* J that includes both literals assigned true, and assignments of values to free first-order variables. Second, it generalizes CDCL to any theory that can be equipped with clausal inference rules to *explain* theory conflicts. These inference rules generate clauses that may contain *new* ground atoms in the signature of the theory, beyond what is allowed by DPLL($\mathcal{T}$) with splitting on demand. Assignments to first-order variables and new atoms are involved in decisions, propagations, conflict detections, and explanations, on a par with Boolean assignments and input literals. For termination, the method requires that new atoms come from a *finite basis*. A procedure that applies systematically the inference rules to enumerate all atoms in the finite basis would be too inefficient. The key point is that the inference rules are applied only to explain conflicts and amend the current partial assignment, so that the generation of new atoms is model-based and conflict-driven. In this sense, MCSAT is a faithful lifting of CDCL to SMT and SMA, with first-order inferences for theory explanation, beyond explanation by propositional resolution that DPLL-CDCL, DPLL($\mathcal{T}$), DPLL($\mathcal{T}$) with splitting on demand, and their integration with superposition in DPLL(Γ+$\mathcal{T}$) [3], all have in common.

The big picture sees various approaches to generalize CDCL to first-order reasoning. From the SMT side, the process started with generalizations of CDCL to specific theories, such as linear real arithmetic (LRA) [25, 21, 7], linear integer arithmetic (LIA) [19], non-linear arithmetic [20], and floating-point binary arithmetic [16]. By being generic with respect to the theory, and integrating theory reasoning and propositional reasoning in all aspects of deduction and search, MCSAT encompasses these predecessors. From the theorem proving side, *semantically-guided goal-sensitive* (SGGS) reasoning lifts CDCL to a method for first-order logic that is refutationally complete and model-complete in the limit [4, 5]. The future may witness further convergence.

1.2 Contributions

The motivation of MCSAT was to make the integration of theory solvers such as those in [19, 20] with a CDCL-based SAT solver possible. Thus, the combination of theories, involving at least propositional logic, equality, and arithmetic, was an objective of the method since its inception. The goal of this paper is to extend MCSAT to *any generic combination of disjoint theories*. This involves:

- Clarifying the requirements that the theories and their solvers need to fulfill;
- Devising deduction mechanisms for the explanation of conflicts across such a generic combination of theories; and
- Extending the soundness, completeness, and termination results given in [11] for the single theory case to our general combination setting.

This leads to a theory-modular reasoning system, called CDSAT for *Conflict-Driven Satisfiability*, which generalizes both MCSAT and the equality-sharing scheme. In this way we advance both the development of a model-constructing approach to theory combination and the generalization of the CDCL paradigm to first-order reasoning.

2 Preliminary definitions

We assume the basic definitions in automated reasoning, and define those needed for features of CDSAT or especially important in the sequel. The formulæ given as input to CDSAT are quantifier-free formulæ where all variable occurrences are free. In this report, quantifiers may appear only in the axioms of a theory. Axioms are sentences that are formulae where all variables are quantified. Thus, we use *formula* for quantifier-free formula, like those appearing in an input problem, and *sentence* for the axiomatization of a theory.

In SAT and SMT, we are used to writing $l \leftarrow \mathsf{true}$ to say that a propositional atom l is assigned true. In SMT, l can be a proxy for a first-order atom. MCSAT [11, 18] and CDSAT also use assignments to first-order variables that can be proxies for terms. CDSAT generalizes the notion of assignment, allowing assignments to first-order variables, terms, atoms, literals, and even formulæ, in a uniform way. Thus, we choose basic definitions that blur the distinction between function symbol and predicate symbol, and the distinction between term, atom, literal, and formula, regarding all these expressions as terms.

Definition 1 (Signature) A *signature* $\Sigma = (S, F)$ consists of a set S of *sorts* that includes a special sort prop and a set F of *symbols* that includes equality symbols $\quad_s : (s \times s) \to \mathsf{prop}$ for every $s \quad S$. ※

For a symbol $f \quad F$, the notation $f : (s_1 \times \cdots \times s_m) \to s$ says that f has *arity* m, *input sorts* $s_1, \ldots, s_m$ $(m \geq 0)$ and *output sort* s. Symbols can be constant symbols $(m = 0)$, function symbols, and predicate symbols that have prop as output sort. We may write $\quad_S$ for $\{ \quad_s : s \times s \to \mathsf{prop} \mid s \quad S\}$, and $\quad$ for $\quad_s$ when sort s is clear from context. The connectives $\quad$, $\quad$, and $\neg$, if present, are seen as symbols whose input and output sorts are prop. Given a set of sorts S, we use $V = (V^s)_{s \in S}$ for a family of pairwise disjoint sets of *variables*, where V^s is the set of variables of sort s. With a slight abuse of the notation, if S_1 and S_2 are two sets of sorts with their families of sets of variables V_1 and V_2, we write $V_1 \quad V_2$, if for all $s \quad S_1$, we have $s \quad S_2$ and $V_1^s \quad V_2^s$.

Definition 2 ($\Sigma[V]$-term) Given $\Sigma = (S, F)$ and $V = (V^s)_{s \in S}$, for all $s \quad S$, every variable $x \quad V^s$ is a $\Sigma[V]$-term of sort s; and for all symbols $f : (s_1 \times \cdots \times s_m) \to s$ in F, if $t_1, \ldots, t_m$ are $\Sigma[V]$-terms of sorts $s_1, \ldots, s_m$, then $f(t_1, \ldots, t_m)$ is a $\Sigma[V]$-term of sort s. ※

The *free variables* $\mathsf{fv}^s(t)$ of sort s of a $\Sigma[V]$-term t are defined as usual, with $\mathsf{fv}(t)$ denoting the family $(\mathsf{fv}^s(t))_{s \in S}$. We call $\Sigma[V]$-*formulae* the $\Sigma[V]$-terms of sort prop, and use infix notation for equality. We use l for formulæ and t and u for terms of any sort. The standard formulæ of multi-sorted first-order logic can be defined as the closure of our formulæ under quantifiers and Boolean connectives; those with no free variables are called *sentences*, or Σ-*sentences* if we want to specify the signature.

Definition 3 ($\Sigma[V]$-interpretation) Given $\Sigma = (S, F)$ and $V = (V^s)_{s \in S}$, a $\Sigma[V]$-*interpretation* $\mathcal{M}$ consists of:

- For each sort $s \quad S$, a non-empty *domain* $s^{\mathcal{M}}$, with the proviso that $\mathsf{prop}^{\mathcal{M}} = \{\mathsf{true}, \mathsf{false}\}$;
- For each symbol $f : (s_1 \times \cdots \times s_m) \to s$ in F, a function $f^{\mathcal{M}}$ from $s_1^{\mathcal{M}} \times \cdots \times s_m^{\mathcal{M}}$ to $s^{\mathcal{M}}$, with the

proviso that for each sort $s \quad S$, ${}^{\mathcal{M}}_{s}$ is the function from $s^{\mathcal{M}} \times s^{\mathcal{M}}$ to $\{\mathsf{true}, \mathsf{false}\}$ that returns true if and only if its two arguments are the same element; and

- For each variable $v \quad V^s$, an element $v^{\mathcal{M}}$ in $s^{\mathcal{M}}$.

※

Given a $\Sigma[V]$-interpretation M, the interpretation $M(t)$ of a $\Sigma[V]$-term t is defined as usual. So is defined the interpretation of any formula of multi-sorted first-order logic, with or without quantifiers, whose free variables are in V. A Σ-*structure* is a $\Sigma[\vec{\ }]$-interpretation.

A *theory* T on signature Σ is defined axiomatically as a pair (Σ, A), where A is a set of Σ-sentences that are the axioms of T, and model-theoretically as a class of Σ-structures, called models of T or T*-models*. The T-models are those Σ-structures that satisfy the axioms in A. A $T[V]$*-model* is any $\Sigma[V]$-interpretation that is a T-model when the interpretation of variables is forgotten. Two signatures are *disjoint* if they do not share symbols other than equality, and two theories are *disjoint* if their signatures are.

Let $T_1, \ldots, T_n$ be pairwise disjoint theories with signatures $\Sigma_1, \ldots, \Sigma_n$, where $\Sigma_k = (S_k, F_k)$ for $1 \leq k \leq n$. Let T_∞ be their union, with signature $\Sigma_\infty = (S_\infty, F_\infty)$, where $S_\infty = \bigcup_{k=1}^{n} S_k$ and $F_\infty = \bigcup_{k=1}^{n} F_k$, and collection of variables $V_\infty = (V_\infty^s)_{s \in S_\infty}$. From now on, we use *variable* for variables in V_∞. If $T_1, \ldots, T_n$ are defined axiomatically as (Σ_k, A_k), for $1 \leq k \leq n$, the axiomatization of T_∞ is given by $\bigcup_{k=1}^{n} A_k$. In this report we sometimes use Σ and T for anyone of $\Sigma_1, \ldots, \Sigma_n$ and $T_1, \ldots, T_n$, respectively. We also use *term* for $\Sigma_\infty[V_\infty]$-term.

Example 1 Consider the following input problem for CDSAT:

$$P = \{f(\mathsf{select}(\mathsf{store}(a, i, v), j)) \quad w,\ f(u) \quad w - 2,\ i \quad j,\ u \quad v\}.$$

This problem can be understood in the combination of the theory T_1 of linear rational arithmetic, the theory T_2 of equality with the uninterpreted function symbol f, and the theory T_3 of arrays, with the following signatures:

$$\begin{array}{lll}
\Sigma_1 = (\ \{\mathsf{prop}, \mathsf{Q}\} & , & {}_{\{\mathsf{prop},\mathsf{Q}\}} \quad \{(0, 1 : \mathsf{Q}), (+ : (\mathsf{Q} \times \mathsf{Q}) \to \mathsf{Q})\} \quad \{(c\cdot : \mathsf{Q} \to \mathsf{Q}) \mid c \quad \mathbb{Q}\}\ ) \\
\Sigma_2 = (\ \{\mathsf{prop}, \mathsf{Q}, V\} & , & {}_{\{\mathsf{prop},\mathsf{Q},V\}} \quad \{f : V \to \mathsf{Q}\}\ ) \\
\Sigma_3 = (\ \{\mathsf{prop}, V, I, (I \quad V)\}, & & {}_{\{\mathsf{prop},V,I,(I \Rightarrow V)\}} \\
 & & \{\mathsf{select} : (I \quad V) \times I \to V,\ \mathsf{store} : (I \quad V) \times I \times V \to (I \quad V))\}\ )
\end{array}$$

where Q is the sort of the rationals, $\mathbb{Q}$ is the set of the rationals, $w - 2$ is an abbreviation for $w + ((-2)\cdot 1)$, and $(I \quad V)$, I, and V are the sorts of arrays, array indices, and array values, respectively.

※

The language of terms is a common language for communication among the theories to be combined. However, each theory has a partial understanding of a term, as if the theory looked at the term with its own "color filter". Given a term t, a theory T whose signature $\Sigma = (S, F)$ does not include the whole of Σ_∞ sees a subterm of t whose root symbol is not in F as a free variable. We call such a variable Σ-*foreign*, or simply *foreign* if Σ is clear from context. Foreign variables correspond to those terms that would be replaced by *new variables* during *purification*, a process also known as *variable abstraction* or *separation*. Following [18], we use *generalized variables* for free variables including foreign variables. In Fig. 1 we define the set $\mathsf{fv}_\Sigma^s(t)$ of generalized *free* Σ-*variables* of sort s in term t for any $s \quad S$. A variable in $\mathsf{fv}_\Sigma^s(t)$ is Σ-*foreign* if it is not in

$\mathsf{fv}^s_\Sigma(x)$	$\{x\}$	if $x \quad V^s_\infty$
$\mathsf{fv}^s_\Sigma(x)$		if $x \quad V^s_\infty$
$\mathsf{fv}^s_\Sigma(f(t_1,\dots,t_n))$	$\bigcup_{i=1}^n \mathsf{fv}^s_\Sigma(t_i)$	if $f \quad F$
$\mathsf{fv}^s_\Sigma(f(t_1,\dots,t_n))$	$\{f(t_1,\dots,t_n)\}$	if $f \quad F$ and $f(t_1,\dots,t_n)$ is a term of sort s
$\mathsf{fv}^s_\Sigma(f(t_1,\dots,t_n))$		if $f \quad F$ and $f(t_1,\dots,t_n)$ is a term not of sort s

Figure 1: Generalized free Σ-variables for $\Sigma = (S, F)$

V_∞ (fourth line in Fig. 1). Then we use $\mathsf{fv}_\Sigma(t)$ for the family $(\mathsf{fv}^s_\Sigma(t))_{s\in S}$, and we adopt the abbreviations $\mathsf{fv}^s(t)$ and $\mathsf{fv}(t)$ when Σ is Σ_∞. These notations extend as expected to sets of terms or families of sets of terms. Note that a $\Sigma[V]$-interpretation M can interpret term t as $M(t)$ as soon as $\mathsf{fv}_\Sigma(t) \quad V$.

Example 2 Continuing Example 1, the free Σ-variables of P, when Σ is anyone of Σ_1, Σ_2, and Σ_3 are as follows:

$$\begin{aligned}
\mathsf{fv}_{\Sigma_1}(P) &= \{\ f(\mathsf{select}(\mathsf{store}(a,i,v),j)),\ w,\ f(u),\ i \quad j,\ u \quad v\ \}\\
\mathsf{fv}_{\Sigma_2}(P) &= \{\ \mathsf{select}(\mathsf{store}(a,i,v),j),\ w,\ u,\ w-2,\ i \quad j,\ v\ \}\\
\mathsf{fv}_{\Sigma_3}(P) &= \{\ f(\mathsf{select}(\mathsf{store}(a,i,v)),j) \quad w,\ f(u) \quad w-2,\ i,\ j,\ u,\ v\ \}
\end{aligned}$$

※

3 Theory assignments and models

The notion of *assignment* is central to CDSAT. First, CDSAT reads any input problem as an assignment: a SAT problem $\{l_1,\dots,l_m\}$, where $l_1,\dots,l_m$ are propositional clauses, is read as $\{l_1 \leftarrow \mathsf{true},\dots,l_m \leftarrow \mathsf{true}\}$; an SMT problem $\{l_1,\dots,l_m\}$, where $l_1,\dots,l_m$ are literals as in Example 1, is read as $\{l_1 \leftarrow \mathsf{true},\dots,l_m \leftarrow \mathsf{true}\}$; an SMA problem $\{l_1,\dots,l_m\}$ with input assignment $\{x \leftarrow \sqrt{2},\ j \leftarrow 0\}$ is read as $\{l_1 \leftarrow \mathsf{true},\dots,l_m \leftarrow \mathsf{true},\ x \leftarrow \sqrt{2},\ j \leftarrow 0\}$. Then, the goal of CDSAT is to determine whether the input is satisfiable. CDSAT uses assignments to terms of different sorts to represent a candidate model of the input problem and reason about it. Therefore, we need to define (1) a sufficiently general notion of *assignment*, and (2) what it means that an assignment is satisfied, or, equivalently, when the current assignment does indeed capture a model. The latter is the objective of the notion of *endorsement*, or when a model endorses an assignment. For a Boolean assignment it suffices that assignment and model agree: whatever is assigned a truth value in the assignment has that truth value in the model. When other sorts are involved, as in $x \leftarrow \sqrt{2}$, a preliminary step is required, because a value such as $\sqrt{2}$ is not necessarily part of the signature of any theory involved, and therefore is not necessarily interpreted by any of their models. The preliminary step is to extend the signatures of the theories $T_1,\dots,T_k$ with *new constant symbols* to name whichever values may be necessary to assign in order to establish satisfiability. In this section we introduce first theory extensions and then assignments and endorsements, considering first one theory and then many.

3.1 Assignments and models for one theory

An extension adds constant symbols to the signature. Clearly, we are interested only in extensions that are *conservative*:

Definition 4 (Conservative theory extension) Given a theory $\mathcal{T}$ with signature $\Sigma = (S, F)$, a *conservative extension* of $\mathcal{T}$ is a theory $\mathcal{T}^+$ with signature $\Sigma^+ = (S, F^+)$ such that F^+ extends F with new constant symbols and every set of $\Sigma[\mathcal{V}]$-formulæ that is $\mathcal{T}^+$-unsat is $\mathcal{T}$-unsat. ※

Conservativity ensures that reasoning in the extension does not change the problem: if CDSAT discovers $\mathcal{T}_k^+$-unsatisfiability, the problem is $\mathcal{T}_k$-unsatisfiable; if the problem is $\mathcal{T}_k$-satisfiable, there is a $\mathcal{T}_k^+$-model that CDSAT can build.

Let $\mathcal{T}$ be a theory and $\mathcal{T}^+$ a conservative extension. $\mathcal{D}_s$ denotes the set of added constants of sort $s \quad S$, called *$\mathcal{T}^+$-values* of sort s. A sort $s \quad S$ with a non-empty $\mathcal{D}_s$ is called *$\mathcal{T}^+$-public*, because there are $\mathcal{T}^+$-values that can be assigned to terms of sort s in a CDSAT derivation.

Since all models interpret formulæ as true or false, we assume without loss of generality that prop is a $\mathcal{T}^+$-public sort with $\mathcal{D}_{\mathsf{prop}} = \{\mathsf{true}, \mathsf{false}\}$. In other words, true and false are simultaneously the two Boolean values (cf. Definition 3) and two constants that name them. Furthermore, since Boolean constants and more generally Boolean terms are formulae, true and false also need to be interpreted: we assume that true and false are respectively valid and unsatisfiable in $\mathcal{T}^+$.

The *trivial extension* of a theory $\mathcal{T}$ is the extension that only adds $\{\mathsf{true}, \mathsf{false}\}$ as new constants and true and $\neg\mathsf{false}$ as new axioms.

Example 3 Let RA be the theory of real arithmetic on signature $\Sigma_{\mathsf{RA}} = (\{\mathsf{R}, \mathsf{prop}\}, F)$, with $F = \{(0, 1 : \mathsf{R}), (+, -, \times : (\mathsf{R}\times\mathsf{R}) \to \mathsf{R})\} \quad {}_{\{\mathsf{R},\mathsf{prop}\}}$. An extension RA^+ may add a new constant for every real number that is algebraic. In this manner the signature remains countable. The sort R is a RA^+-public sort and the RA^+-values of sort R are the algebraic reals. The axioms of RA^+ are the formulæ that hold in the standard model of the reals that interprets every RA^+-value as itself. ※

Extending the signature with names to denote all individuals in the domain(s) of a $\mathcal{T}$-model, as in the example above, is a standard move in logic. In such cases a $\mathcal{T}^+$-value is both the model element and the corresponding constant symbol that names it. We will do this when $\mathcal{T}$ has a clear "intended model" as for the integers or the reals. For theories without an "intended model", we may take $\mathcal{T}^+$ to be the trivial extension of $\mathcal{T}$. The theory of Equality with Uninterpreted Function symbols (EUF) can be treated in this way. Alternatives for this theory will be considered in the sequel. $\mathcal{T}^+$-values are the values that may appear in *$\mathcal{T}^+$-assignments*:

Definition 5 ($\mathcal{T}^+$-Assignment) A *$\mathcal{T}^+$-assignment* is a set of pairs $t \leftarrow \mathfrak{c}$ where t is a term of a $\mathcal{T}^+$-public sort s and $\mathfrak{c} \quad \mathcal{D}_s$. Term t and all its subterms are said to *occur* in the assignment. The assignment is *plausible* if for no formula l it contains both $l \leftarrow \mathsf{true}$ and $l \leftarrow \mathsf{false}$. ※

For example, $\{x \leftarrow \sqrt{2},\ x + y \leftarrow \sqrt{3}\}$ and $\{f(x) \leftarrow \sqrt{2},\ (1 \times x \quad x) \leftarrow \mathsf{true}\}$ are RA-assignments, with x, y and $x + y$ occurring in the former, and x, $f(x)$, $1 \times x$ and $(1 \times x \quad x)$ occurring in the latter. A $\mathcal{T}^+$-assignment whose pairs all assign values to formulæ is a Boolean assignment. A singleton $\mathcal{T}^+$-assignment is often written $t \leftarrow \mathfrak{c}$ instead of $\{t \leftarrow \mathfrak{c}\}$. A *first-order $\mathcal{T}^+$-assignment* is a singleton $\mathcal{T}^+$-assignment that is not Boolean. We use A and L for singleton $\mathcal{T}^+$-assignments, reserving L for Boolean ones, and J for generic $\mathcal{T}^+$-assignments. The ***flip*** $\overline{L}$ of a singleton Boolean assignment L assigns to the same formula the opposite Boolean value. When there is no ambiguity, we abbreviate $l \leftarrow \mathsf{true}$ as l and $l \leftarrow \mathsf{false}$ as $\bar{l}$, except in the case of equality where $t \simeq_s u \leftarrow \mathsf{false}$ is abbreviated as $t \not\simeq_s u$.

Example 4 Building on Example 3, assume theory $\mathcal{T}_{\mathsf{RA}}$ is combined with some other theory, whose signature features a symbol $f : \mathsf{R} \to \mathsf{R}$. Then $x \leftarrow \sqrt{2}$, $f(0) \leftarrow \sqrt{2}$, $\{x \leftarrow \sqrt{2},\ f(0) \leftarrow \sqrt{2}\}$ and $\{x \leftarrow \sqrt{2},\ f(0) \leftarrow \sqrt{2},\ (1 \times f(0) \simeq f(0))\}$ are all $\mathcal{T}^+_{\mathsf{RA}}$-assignments, and $1 \times f(0) \simeq f(0)$ is a singleton Boolean assignment. ※

We proceed next to define what it means that a model ***endorses*** a $\mathcal{T}^+$-assignment.

Definition 6 (Endorsement) A $\mathcal{T}^+[\mathcal{V}]$-model $\mathcal{M}$ ***endorses*** a $\mathcal{T}^+$-assignment J with $\mathsf{fv}_\Sigma(J) \subseteq \mathcal{V}$, if for all $t \leftarrow \mathfrak{c}$ in J, we have $\mathcal{M}(t) = \mathfrak{c}^{\mathcal{M}}$. ※

In the special case of a Boolean assignment, this simply means that $\mathcal{M}$ interprets the formulæ of J with the correct truth values. The extended signature Σ^+ allows us to predicate endorsement on structures that can make sense of values such as $\sqrt{2}$. Predicating it on $\mathcal{T}^+$-models, we further assume that these structures interpret values in an "intended way", in this case ensuring that $\sqrt{2} \times \sqrt{2} = 2$ holds.

3.2 Combining theories

From now on, we assume that each theory $\mathcal{T}_i$, $1 \leq i \leq n$, has a conservative extension $\mathcal{T}_i^+$ with signature $\Sigma_i^+ = (S_i, F_i^+)$ and set $\mathcal{D}_s^i$ of $\mathcal{T}_i^+$-values of sort s for all $s \in S_i$. As expected, the union of $\mathcal{T}_1^+, \ldots, \mathcal{T}_n^+$ is an extension $\mathcal{T}_\infty^+$ of $\mathcal{T}_\infty$ with signature $\Sigma_\infty^+ = (S_\infty, \bigcup_{k=1}^n F_k^+)$. If $\mathcal{T}_1^+, \ldots, \mathcal{T}_n^+$ are defined axiomatically with sets of axioms $\mathcal{A}_1^+, \ldots, \mathcal{A}_n^+$, the axiomatization of $\mathcal{T}_\infty^+$ is given by $\bigcup_{k=1}^n \mathcal{A}_k^+$. We assume without loss of generality that every non-Boolean $\mathcal{T}_\infty^+$-value unambiguously comes from a unique $\mathcal{T}_k^+$.

$\mathcal{T}_\infty^+$-assignments are simply called ***assignments***, and denoted H. A sort s may be both $\mathcal{T}_i^+$-public and $\mathcal{T}_j^+$-public for $i \neq j$, and therefore an assignment H may contain $t \leftarrow \mathfrak{c}_i$ and $t \leftarrow \mathfrak{c}_j$ for $i \neq j$ for a term t of a sort s, a $\mathcal{T}_i^+$-value $\mathfrak{c}_i \in \mathcal{D}_s^i$, and a $\mathcal{T}_j^+$-value $\mathfrak{c}_j \in \mathcal{D}_s^j$. Unless s is prop, $\mathfrak{c}_i$ and $\mathfrak{c}_j$ live in the different worlds described by $\mathcal{T}_i^+$ and $\mathcal{T}_j^+$, but they will be identified as the same element when constructing a $\mathcal{T}_\infty^+$-model endorsing H from a $\mathcal{T}_i^+$-model and a $\mathcal{T}_j^+$-model both endorsing H.

In order to extend the notion of endorsement (cf. Definition 6) to problems involving many theories we need the notion of ***theory view***:

Definition 7 (Theory view) Given theory $\mathcal{T}$ with signature Σ and extension $\mathcal{T}^+$ with signature $\Sigma^+ = (S, F^+)$, where $S \subseteq S_\infty$ and $F^+ \subseteq F_\infty^+$, the ***theory view*** for $\mathcal{T}$, or ***$\mathcal{T}$-view***, of an assignment H is the $\mathcal{T}^+$-assignment $H_\mathcal{T} =$

$$\begin{array}{ll} & \{\ t \leftarrow \mathfrak{c} \quad \mid\ t \leftarrow \mathfrak{c} \text{ is a } \mathcal{T}\text{-assignment in } H\ \} \\ \cup\ \bigcup_{k=1}^n & \{\ t_1 \simeq_s t_2 \ \mid\ t_1 \leftarrow \mathfrak{c}, t_2 \leftarrow \mathfrak{c} \text{ are } \mathcal{T}_k\text{-assignments in } H,\ s \in S \setminus \{\mathsf{prop}\}\ \} \\ \cup\ \bigcup_{k=1}^n & \{\ t_1 \not\simeq_s t_2 \ \mid\ t_1 \leftarrow \mathfrak{c}_1, t_2 \leftarrow \mathfrak{c}_2 \text{ are } \mathcal{T}_k\text{-assignments in } H,\ \mathfrak{c}_1 \neq \mathfrak{c}_2,\ s \in S \setminus \{\mathsf{prop}\}\ \}. \end{array}$$

※

The first part of $H_{\mathcal{T}}$ is the part of H that theory T^+ can understand; the second part adds all the equalities entailed by assignments of identical values; and the third part adds all the disequalities entailed by assignments of distinct values. Typically either T_∞-views or T_k-views, for some k, $1 \le k \le n$, are considered.

A CDSAT input is a T_∞^+-assignment H and the problem is to determine whether there exists a $T_\infty^+[\mathsf{fv}(H)]$-model M that endorses its T_∞^+-view:

Definition 8 (View endorsement) Given theory T with signature Σ and extension T^+ with signature $\Sigma^+ = (S, F^+)$, where S S_∞ and F^+ F_∞^+, a $T^+[V]$-model M *view-endorses* an assignment H with $\mathsf{fv}_\Sigma(H)$ V, if it endorses the T-view $H_{\mathcal{T}}$. ※

Note that the disequalities in the definition of $H_{\mathcal{T}}$ impose that any $T^+[V]$-model endorsing $H_{\mathcal{T}}$ *distinguishes* the distinct T_k^+-values that appear in H for all k. If H is a Boolean assignment, view endorsement collapses to endorsement. CDSAT solves an SMA problem by searching for a $T_\infty^+[\vec{V_\infty}]$-model that endorses the input assignment.

4 Theory modules

In this section we identify a notion of *theory module*, which is an abstraction of the theory-specific decision procedures, implemented in theory solvers or theory plugins [18]. Like all conflict-driven procedures, CDSAT allows the *explicit* construction of a (partial) model. In practice, it lets the theory-specific procedures expand the input assignment, finding values for subterms of the input problem as well as other terms introduced during the derivation. Some of the assignments are guesses, while others are consequences of such guesses according to *inferences*. Therefore, a module I for theory T with extension T^+, or T-module, is given by two components:

1. A set of I*-inference rules*, modeling reasoning steps in theory T^+; and
2. A function basis_k, called *local basis*, that maps any finite set X of terms to a *finite* set of terms $\mathsf{basis}_k(X)$ representing the terms that inferences can introduce during a derivation from an input problem whose set of terms is included in X.

A T-module is required to satisfy a completeness property, that will be defined in such a way to ensure that if all T-modules are complete then their CDSAT combination is complete. Section 4.1 formalizes theory modules. Section 4.2 introduces the concepts of acceptability and relevance that will be used to define the CDSAT system (Section 6). Section 4.3 defines the completeness requirement for a theory module.

4.1 Theory inference system and basis

Let T be one of the theories $T_1, \ldots, T_n$ to be combined, with signature Σ, and let T^+ be its extension. The first component of a T-module is an inference system I to reason in theory T^+. Due to the centrality of assignments in CDSAT, the theory inference systems work on assignments, and inference rules derive Boolean assignments from generic assignments. An I*-inference* J L derives a singleton Boolean assignment L from a T^+-assignment J. An inference system I is *sound* if for all its inferences J L whenever $\mathsf{fv}_\Sigma(J, L)$ V, every $T^+[V]$-model that view-endorses J

$$\begin{array}{rl}
t_1 \leftarrow \mathfrak{c}_1, t_2 \leftarrow \mathfrak{c}_2 \quad t_1 \quad {}_s\, t_2 & \text{if } \mathfrak{c}_1 \text{ and } \mathfrak{c}_2 \text{ are the same } \mathcal{T}^+\text{-value of sort } s \\
t_1 \leftarrow \mathfrak{c}_1, t_2 \leftarrow \mathfrak{c}_2 \quad t_1 \quad {}_s\, t_2 & \text{if } \mathfrak{c}_1 \text{ and } \mathfrak{c}_2 \text{ are distinct } \mathcal{T}^+\text{-values of sort } s \\
t_1 \quad {}_s\, t_1 & \text{reflexivity} \\
t_1 \quad {}_s\, t_2 \quad t_2 \quad {}_s\, t_1 & \text{symmetry} \\
t_1 \quad {}_s\, t_2, t_2 \quad {}_s\, t_3 \quad t_1 \quad {}_s\, t_3 & \text{transitivity}
\end{array}$$

where t_1, t_2, and t_3 are terms of sort s.

Figure 2: Equality inference rules

endorses L. Since all theories include equality, every theory inference system includes the ***equality inference rules*** of Fig. 2.

Example 5 Following Example 4, these are all RA-inferences:

$$\begin{array}{c}
\{\, x \leftarrow \sqrt{2},\ f(0) \leftarrow \sqrt{2}\ \} \quad_{\mathsf{RA}}\ (x \times f(0) \quad 1+1) \\
(x \leftarrow \sqrt{2}) \quad_{\mathsf{RA}}\ (x \times x \quad 1+1) \\
\{(f(0) \leftarrow \sqrt{2}), (x \leftarrow \sqrt{2})\} \quad_{\mathsf{RA}}\ (f(0) \quad x) \\
\{(f(0) \leftarrow \sqrt{2}), (x \leftarrow \sqrt{3})\} \quad_{\mathsf{RA}}\ (f(0) \quad x)
\end{array}$$

※

The second component of a theory module $\mathcal{I}$ is a ***local basis*** for signature Σ. This is a function that must be provided when describing a theory module, but it does not need to be implemented: it only plays a role in the proof of termination of CDSAT derivations using this theory module. To motivate this notion, notice that if an $\mathcal{I}$-inference $J \quad L$ is used to infer L from J, assignment L may introduce new terms that were not in J nor even in the input problem. This is in line with MCSAT calculi [11, 18] where, unlike DPLL($\mathcal{T}$), theory solvers can introduce terms and formulæ that were not present in the original problem, jeopardising termination. Termination is ensured by requiring that the new terms introduced by theory solvers be drawn from a finite set called ***global basis***. So the second component of our notion of theory module is a theory-local version of this global basis, used to limit to a finite number the range of terms that the theory module may introduce in a derivation for an input problem. This will help ensuring termination of the CDSAT transition system.

To define the notion of local basis, we introduce the following terminology: A ***closed set*** is a finite set of terms that is closed under the subterm relation and equalities on a sort different from prop: if t is a subterm of u and u is in the set, then so is t; and if t and u are in the set, of a sort s different from prop, then so is $t \quad_s u$.

Definition 9 (Local basis) A function basis mapping any closed set X to a closed set basis(X) is said to be a ***local basis*** for signature Σ if the following properties hold:

- Original terms: $X \quad$ basis(X);
- Finiteness: basis(X) is finite;
- Monotonicity: If $X \quad Y$ then basis$(X) \quad$ basis(Y);
- Idempotence: basis(basis(X)) $=$ basis(X);
- No introduction of foreign variables: Every foreign Σ-variable of basis(X) is in X.

※

Examples of local bases for various theories are given in Section 5. Section 9 shows how the properties required of a local basis contribute to the termination and progress properties of the CDSAT system, i.e. the fact that it systematically reduces the input problem to an interesting normal form. This will also rely, however, on the existence of a *global basis* for the combination of theories, and Section 9.3 shows an example of sufficient condition that entails its existence.

For any finite set X of terms, we write $\ X$ for the smallest closed set containing X. Note that the closure operation $X \to \ X$ is monotonic and idempotent. Given a local basis, we generalise the notation $\mathsf{basis}(X)$ to the cases where X is not a closed set, meaning $\mathsf{basis}(\ X)$.

In brief, a module for theory $\mathcal{T}$ with extension $\mathcal{T}^+$ is a pair (, basis) as described above.

A further requirement that we impose on a theory module is the *completeness* property that we describe in Section 4.3. But it is not needed for the definition of the CDSAT system *per se* (given in Section 6), which does require, on the other hand, a couple of concepts and notations.

4.2 Acceptability and Relevance

CDSAT builds incrementally a (partial) model by extending the existing assignment with values for unassigned terms. When adding an assignment, the system checks that it does not cause a *one-step violation*, that is, an inconsistency that one inference is sufficient to expose:

Definition 10 (One-step violation) Given a $\mathcal{T}^+$-assignment J, a first-order $\mathcal{T}^+$-assignment A *violates* J in one $\mathcal{I}$-step, if there exists an $\mathcal{I}$-inference $(J', A \ _{\mathcal{I}} L)$ with $J', \overline{L} \ J$. ※

Definition 11 (Acceptability) A singleton $\mathcal{T}^+$-assignment $(t \leftarrow \mathfrak{c})$ is *acceptable for J and $\mathcal{I}$* if (i) J does not already assign a value to t and (ii) either $(t \leftarrow \mathfrak{c})$ is Boolean or it does not violate J in one $\mathcal{I}$-step. ※

When adding $t \leftarrow \mathfrak{c}$ to J, acceptability prevents repetitions (cf. Condition (i)) and contradictions: if $t \leftarrow \mathfrak{c}$ is Boolean, its flip should not be in J, preserving plausibility (cf. Condition (i) in Definition 11 and Definition 5); if $t \leftarrow \mathfrak{c}$ is first-order, and therefore has no flip, so that plausibility does not apply, acceptability ensures that none of the consequences one inference step away has its flip in J (cf. Condition (ii)).

The following notion of *relevance* organizes the division of labor among modules. $\mathcal{T}^+$-relevant terms are those that the $\mathcal{T}^+$-module should consider for assignment in order to advance the model building process:

Definition 12 ($\mathcal{T}^+$-relevant terms) A term is *$\mathcal{T}^+$-relevant* for an assignment H, if either (i) it occurs in H and has a $\mathcal{T}^+$-public sort, or (ii) it is an equality $t_1 \ _s t_2$ whose terms t_1 and t_2 occur in H and whose sort $s \ S$ is not $\mathcal{T}^+$-public. ※

For instance in the assignment $\{x \leftarrow \sqrt{5}, f(x) \leftarrow \sqrt{2}, f(y) \leftarrow \sqrt{3}\}$, x and y, both of sort R, are RA-relevant, not EUF-relevant, assuming R is not EUF-public, while $x \ _{\mathsf{R}} y$ is EUF-relevant, not RA-relevant. Each theory needs to have a mechanism to fix and communicate equalities between terms of a known sort, such as x and y: EUF can do it by deciding the truth-value of $x \ _{\mathsf{R}} y$, while RA can do it by assigning values, either the same or different, to x and y.

From now on, we assume that each theory T_k, $1 \leq k \leq n$

allows us to include theories that are not stably infinite. We only assume that one of theories, named $\mathcal{T}_0$ and with signature $\Sigma_0 = (S_\infty, F_0)$, has information about the cardinality constraints from all the theories. An assignment J then satisfies the criterion for $\mathcal{T}^+$, defined next as $\mathcal{T}_0$-*compatibility*, if any model of $\mathcal{T}_0$ view-endorsing J can be turned into a model of $\mathcal{T}^+$.

Definition 16 ($\mathcal{T}_0$-Compatibility) Given a family of terms $\mathcal{V} = (\mathcal{V}^s)_{s \in S_\infty}$ and a $\mathcal{T}^+$-assignment J, we say that J is *$\mathcal{T}_0$-compatible with $\mathcal{T}^+$ sharing $\mathcal{V}$* if for all $\mathcal{T}_0[\mathsf{fv}_{\Sigma_0}(J \quad \mathcal{V})]$-model $\mathcal{M}_0$ that view-endorses J, there exists a $\mathcal{T}^+[\mathsf{fv}_\Sigma(J \quad \mathcal{V})]$-model $\mathcal{M}$ that view-endorses J, such that for all $s \quad S$, $\left|s^{\mathcal{M}}\right| = \left|s^{\mathcal{M}_0}\right|$, and for all t and t' in $\mathcal{V}^s$, $\mathcal{M}(t) = \mathcal{M}(t')$ if and only if $\mathcal{M}_0(t) = \mathcal{M}_0(t')$. ※

Compatibility will allow us "to glue" together in one model the models provided by the n theories by showing that an assignment that is consistent with $\mathcal{T}_0$, and $\mathcal{T}_0$-compatible with all theories $\mathcal{T}_1^+, \ldots, \mathcal{T}_n^+$, is view-endorsed by a common $\mathcal{T}_\infty^+$-model (cf. Section 8.3). We can finally express the requirement on theory modules as follows:

Definition 17 ($\mathcal{T}_0$-Completeness) A $\mathcal{T}$-module $\mathcal{I}$ is *$\mathcal{T}_0$-complete* if for all plausible $\mathcal{T}^+$-assignments J,

- Either J is $\mathcal{T}_0$-compatible with $\mathcal{T}^+$ sharing all subterms in J;
- Or $\mathcal{I}$ can extend J.

※

An immediate corollary is that, by combining a complete theory module for $\mathcal{T}_0$ and $\mathcal{T}_0$-complete theory modules for all theory extensions $\mathcal{T}_1^+, \ldots, \mathcal{T}_n^+$, CDSAT satisfies *model-soundness*: if the system ever produces an assignment that no theory module can extend, then the input problem is satisfiable in $\mathcal{T}_\infty^+$. Together with termination and progress, this immediately entails *refutational completeness*: if the problem is unsatisfiable in $\mathcal{T}_\infty^+$, then the system returns unsat.

5 Examples of theory modules

In this section we give examples of theories and theory modules, and we describe how decision procedures for Nelson-Oppen theories [27] can be treated as theory modules and accommodated in the CDSAT framework. As we shall see in the examples, it will be convenient, in theory-specific inferences, to use an unsatisfiable Boolean assignment . For this we can use an arbitrary variable x of sort prop, and introduce to stand for $(x \quad_{\mathsf{prop}} x) \leftarrow \mathsf{true}$ and for $(x \quad_{\mathsf{prop}} x) \leftarrow \mathsf{false}$. No interpretation can ever endorse and the equality inference can be regarded as trivially available. However this poses the issue as to whether such an inference should be regarded as extending an assignment (cf. Definition 13). The answer is positive under the assumption that is in the local basis. Then, if the theory module cannot extend J, it means that is in J and is not.

The following lemma will be useful to prove completeness properties of the example theory modules:

Lemma 1 Assume a $\mathcal{T}$-module $\mathcal{I}$ cannot extend a plausible $\mathcal{T}^+$-assignment J. Then:

1. For all sorts $s \quad S \backslash \{\mathsf{prop}\}$, the binary relation $(\_ \quad_s \_) \quad J$ over terms occurring in J of sort

s is an equivalence relation, and if $(t_1 \leftarrow \mathfrak{c}_1) \quad J$ and $(t_2 \leftarrow \mathfrak{c}_2) \quad J$, then $\mathfrak{c}_1$ is identical to $\mathfrak{c}_2$ if and only if $(t_1 \quad {}_s\, t_2) \quad J$;
2. For all sorts $s \quad S$ that is not $\mathcal{T}^+$-public, and all terms t_1 and t_2 of sort s occurring in J, $t_1 \quad {}_s\, t_2$ is assigned a value in J. All formulae that occur in J are assigned values in J.
3. For all $\mathcal{T}^+$-public sorts $s \quad S$ such that
   - The only $\mathcal{I}$-inferences of the form $J', (t \leftarrow \mathfrak{c}) \quad_{\mathcal{I}} L$, with t of sort s, are equality inferences, and
   - There are countably many $\mathcal{T}^+$-values,

   every term of sort s that occurs in J is assigned a value in J.

※

Proof:
1. First, notice that $\mathsf{basis}_{\mathcal{I}}(J)$ is closed and therefore contains all equalities between terms occurring in J of sort $s = \mathsf{prop}$. If $\mathcal{I}$ cannot extend J, then it means that the Boolean assignments inferred by the equality inferences for reflexivity, symmetry, and transitivity are already present in J. This makes this binary relation an equivalence one.

   Second, if $\mathfrak{c}_1$ is identical to $\mathfrak{c}_2$, an equality inference can infer $t_1 \quad {}_s\, t_2$, so if $\mathcal{I}$ cannot extend J, $t_1 \quad {}_s\, t_2$ must already be present in J. Conversely if $\mathfrak{c}_1$ is different from $\mathfrak{c}_2$, an equality inference can infer $t_1 \quad {}_s\, t_2$, so if $\mathcal{I}$ cannot extend J, $t_1 \quad {}_s\, t_2$ must already be present in J, and therefore plausibility of J entails that $t_1 \quad {}_s\, t_2$ is not in J.
2. Direct consequence of the fact that $\mathcal{I}$ cannot extend J.
3. Point 2 already subsumes the case when $s = \mathsf{prop}$. Otherwise assume by contradiction that a term t of sort s that occur in J is not assigned a value. As $\mathcal{I}$ cannot extend J, in order to derive a contradiction it suffices to find an assignment that is acceptable for J and $\mathcal{I}$. Consider the equivalence class of t for the binary relation $(\_ \quad {}_s\, \_) \quad J$ (which Point 1 proves to be an equivalence one). If none of the terms in that equivalence class are assigned a value in J, then for a fresh value $\mathfrak{c}$ of sort s (i.e. never used in J), $(t \leftarrow \mathfrak{c})$ is an assignment acceptable for J and $\mathcal{I}$. If one of them, say t_1, is assigned some value $\mathfrak{c}_1$ in J, then again $(t \leftarrow \mathfrak{c}_1)$ is an assignment acceptable for J and $\mathcal{I}$: indeed, if $(t \quad {}_s\, t_2)$ or $(t_2 \quad {}_s\, t)$ is in J with $(t_2 \leftarrow \mathfrak{c}_2) \quad J$, then Point 1 entails that $(t_1 \quad {}_s\, t_2) \quad J$ and therefore $\mathfrak{c}_1$ and $\mathfrak{c}_2$ are the same; and if $(t \quad {}_s\, t_2)$ or $(t_2 \quad {}_s\, t)$ is in J with $(t_2 \leftarrow \mathfrak{c}_2) \quad J$, then Point 1 entails that $(t_1 \quad {}_s\, t_2)\ /\ J$, and therefore $\mathfrak{c}_1$ and $\mathfrak{c}_2$ are different.

□

5.1 A Module for Propositional Logic

The signature Σ_{Bool} for propositional logic (Bool) is:

$$(\{\mathsf{prop}\} \ , \quad {}_{\{\mathsf{prop}\}} \quad \{(\quad , \quad : (\mathsf{prop} \times \mathsf{prop}) \to \mathsf{prop}), (\neg : \mathsf{prop} \to \mathsf{prop})\} \ )$$

We take as extension Bool^+ the trivial one, so that the signature Σ^+_{Bool} only adds $\{\mathsf{true}, \mathsf{false}\}$ as Bool^+-values. The simplest module we can take for Bool, call it $\mathcal{I}_{\mathsf{Bool}_{\mathsf{eval}}}$, only has the following *evaluation inferences*:

$$l_1 \leftarrow \mathfrak{b}_1, \ldots, l_m \leftarrow \mathfrak{b}_m \quad {}_{\mathsf{Bool}_{\mathsf{eval}}}\ l \leftarrow \mathfrak{b}$$

where $l_1, \ldots, l_m$ are formulæ, l is in the closure of $l_1, \ldots, l_m$ under the Σ_{Bool}-constructs, and $\mathfrak{b}$

is its evaluation, fully determined by $\mathfrak{b}_1, \ldots, \mathfrak{b}_m$ and the truth tables for the connectives in the signature. As local basis we take the identity function so that $\mathsf{basis}_{\mathsf{Bool}}(X) = X$ for all X.

Lemma 2 (Completeness) For any theory T_0, the Bool-module $I_{\mathsf{Bool}_{\mathsf{eval}}}$ is T_0-complete. ※

Proof: Let J be a plausible Bool^+-assignment. If $I_{\mathsf{Bool}_{\mathsf{eval}}}$ cannot extend J, if means in particular that all formulae that occur in J are assigned values in J. It also means that the $\Sigma^+_{\mathsf{Bool}}[\mathsf{fv}_{\Sigma_{\mathsf{Bool}}}(J)]$-interpretation $\mathcal{M}$ that interprets every $t \quad \mathsf{fv}_{\Sigma_{\mathsf{Bool}}}(J)$ as indicated by J (and interprets every Boolean connective as indicated by the usual truth tables) view-endorses J: Indeed, if $(l \leftarrow \mathfrak{b}) \quad J$ and $\mathcal{M}(l)$ were different from $\mathfrak{b}$, then there would be an $I_{\mathsf{Bool}_{\mathsf{eval}}}$-inference concluding $\overline{l \leftarrow \mathfrak{b}}$, and thus $I_{\mathsf{Bool}_{\mathsf{eval}}}$ could extend J.

Let Σ_0 be T_0's signature. Given any $T_0[\mathsf{fv}_{\Sigma_0}(J)]$-interpretation $\mathcal{M}_0$ that view-endorses J, $\left|\mathsf{prop}^{\mathcal{M}}\right| = \left|\mathsf{prop}^{\mathcal{M}_0}\right| = 2$, and for all terms t and t' of sort prop occurring in J, $\mathcal{M}(t) = \mathcal{M}(t')$ if and only if $\mathcal{M}_0(t) = \mathcal{M}_0(t')$, since this happens if and only if t and t' are assigned the same value in J. □

Although they are not needed for completeness, it is convenient to add the following inference rules, comprising, from left to right, two rules for negation, two rules for conjunction elimination, and two rules for unit propagation:

$$\begin{array}{llllll}
\neg l & {}_{\mathsf{Bool}}\ \bar{l} & \overline{l_1 \quad \cdots \quad l_m} & {}_{\mathsf{Bool}}\ \overline{l_i} & l_1 \quad \cdots \quad l_m, \{\overline{l_j} \mid j = i\} & {}_{\mathsf{Bool}}\ l_i \\
\overline{\neg l} & {}_{\mathsf{Bool}}\ l & l_1 \quad \cdots \quad l_m & {}_{\mathsf{Bool}}\ l_i & \overline{l_1 \quad \cdots \quad l_m}, \{l_j \mid j = i\} & {}_{\mathsf{Bool}}\ \overline{l_i}
\end{array}$$

where $1 \leq j, i \leq m$. We call $I_{\mathsf{Bool}} = (\quad_{\mathsf{Bool}}, \mathsf{basis}_{\mathsf{Bool}})$ the resulting module, which is of course still T_0-complete for any T_0, given that every time $I_{\mathsf{Bool}_{\mathsf{eval}}}$ can extend a Bool^+-assignment, so can I_{Bool}.

5.2 Linear Rational Arithmetic (LRA)

Theory LRA can be decided by a simplex algorithm, which could be integrated to our framework as described above. Below, we present a theory module for LRA that more closely follows its MCSAT treatment [18]. The signature Σ_{LRA} of theory LRA is $(\{\mathsf{prop}, \mathsf{Q}\} , F_{\mathsf{LRA}})$ where F_{LRA} is

$$_{\mathsf{prop},\mathsf{Q}} \qquad \{(0, 1 : \mathsf{Q}), (+ : (\mathsf{Q} \times \mathsf{Q}) \to \mathsf{Q}), (<, \leq : (\mathsf{Q} \times \mathsf{Q}) \to \mathsf{prop})\} \qquad \{(c \cdot : \mathsf{Q} \to \mathsf{Q}) \mid c \quad \mathbb{Q}\}$$

We take as extension LRA^+ the theory whose signature Σ^+_{LRA} adds to F_{LRA} one constant for each rational number, namely:

$$(\{\mathsf{prop}, \mathsf{Q}\} , F_{\mathsf{LRA}} \quad \{(\tilde{q} : \mathsf{Q}) \mid q \quad \mathbb{Q}\}),$$

and that adds to LRA, as axioms, the equalities $\tilde{q} \quad_{\mathsf{Q}} q \cdot 1$ for all rational numbers q.

In order to define the module I_{LRA} with a local basis, we need to fix an arbitrary total order between all Σ_{LRA}-variables of sort Q. A term t is *maximal in a term* t' if it is the greatest element in $\mathsf{fv}^{\mathsf{Q}}_{\Sigma_{\mathsf{LRA}}}(t')$ according to . We then define the module I_{LRA} as $(\quad_{\mathsf{LRA}}, \mathsf{basis}_{\mathsf{LRA}})$ as follows. The I_{LRA}-inferences are those of the following forms:

- Evaluations:

$$t_1 \leftarrow \tilde{q}_1, \ldots, t_m \leftarrow \tilde{q_m} \quad_{\mathsf{LRA}}\ l \leftarrow \mathfrak{b}$$

where $t_1, \ldots, t_m$ are terms of sort Q, l is a formula in the closure of $t_1, \ldots, t_m$ under the symbols in F_{LRA}, and $\mathfrak{b}$ is the evaluation for it as defined through this closure.

- Positivization:
$$\overline{t_1 < t_2} \quad {}_{\mathsf{LRA}}\, t_2 \leq t_1$$
$$\overline{t_1 \leq t_2} \quad {}_{\mathsf{LRA}}\, t_2 < t_1$$
- Elimination of equality:
$$t_1 \quad {}_{\mathsf{Q}}\, t_2 \quad {}_{\mathsf{LRA}}\, t_i \leq t_j \qquad \text{with } \{i, j\} = \{1, 2\}$$
- Elimination of disequality:
$$(e_1 \leq t), (t \leq e_2), (e_1 \quad {}_{\mathsf{Q}}\, e_0), (e_2 \quad {}_{\mathsf{Q}}\, e_0), (t \quad {}_{\mathsf{Q}}\, e_0) \quad {}_{\mathsf{LRA}}$$
where t is a free Σ_{LRA}-variable of $(e_1 \leq t), (t \leq e_2)$, $(t \quad {}_{\mathsf{Q}}\, e_0)$, but is not free in e_0, e_1 or e_2. The expressions $(e_1 \leq t)$, $(t \leq e_2)$, and $(t \quad {}_{\mathsf{Q}}\, e_0)$ range over all Boolean assignments that can be normalised to that form (by the usual normalisation of rational expressions).
- Fourier-Motzkin resolutions [18]:
$$(e_1 \lessdot_1 t), (t \lessdot_2 e_2) \quad {}_{\mathsf{LRA}}\, (e_1 \lessdot_3 e_2)$$
where $\lessdot_1, \lessdot_2, \lessdot_3$ are all in $\{<, \leq\}$ and $\lessdot_3$ is $<$ if and only if either $\lessdot_1$ or $\lessdot_2$ is $<$,
and t is maximal in $(e_1 \lessdot_1 t)$ and maximal in $(t \lessdot_2 e_2)$, but is not in $\mathsf{fv}^{\mathsf{Q}}_{\Sigma_{\mathsf{LRA}}}(e_1, e_2)$.

  The expressions $(e_1 \lessdot_1 t)$ and $(t \lessdot_2 e_2)$ range over all Boolean assignments that can be normalised to that form.
- Elimination of empty solution spaces:
$$t_1 \leftarrow \tilde{q_1}, \ldots, t_m \leftarrow \tilde{q_m}, E \quad {}_{\mathsf{LRA}}$$
where $t_1, \ldots, t_m$ are Σ_{LRA}-variables of sort Q, E is a (not necessarily single) Boolean assignment such that for all x in $\mathsf{fv}^{\mathsf{Q}}_{\Sigma_{\mathsf{LRA}}}(E)$, $x \quad t_i$ or $x = t_i$ for some $1 \leq i \leq m$, and $t_1 \quad {}_{\mathsf{Q}}\, \tilde{q_1}, \ldots, t_m \quad {}_{\mathsf{Q}}\, \tilde{q_m}, E$ is not LRA^+-satisfiable.

The local basis is simply the closure of a closed set under the LRA inferences: Given a closed set X, let $\mathsf{basis}_{\mathsf{LRA}}(X)$ be the smallest closed set containing X and closed under the rules

$$- \qquad \frac{t < t'}{t' \leq t} \qquad \frac{t \leq t'}{t' < t} \qquad \frac{(e_1 \lessdot_1 t) \quad (t \lessdot_2 e_2)}{(e_1 \lessdot_3 e_2)}$$

where $\lessdot_3$ is $<$ if and only if either $\lessdot_1$ or $\lessdot_2$ is $<$, t is maximal in $(e_1 \lessdot_1 t)$ and in $(t \lessdot_2 e_2)$, and t / $\mathsf{fv}^{\mathsf{Q}}_{\Sigma_{\mathsf{LRA}}}(e_1, e_2)$, and the expressions $(e_1 \lessdot_1 t)$ and $(t \lessdot_2 e_2)$ range over terms that can be normalised to that form. Remark that $\mathsf{basis}_{\mathsf{LRA}}(X)$ is finite.

The role of the precedence relation can be illustrated by the following example:

$$\begin{aligned} l_0 :&\quad -2{\cdot}x - y < 0 \\ l_1 :&\quad x + y < 0 \\ l_2 :&\quad x < -1 \end{aligned}$$

Note that the assignment l_0, l_1, l_2 is LRA-unsatisfiable. If the Fourier-Motzkin resolution inference were not restricted to eliminate the maximal variable only, it could be used to generate the following infinite chain:

$$\begin{array}{lll} l_3: & -y < -2 & \text{from } l_0 \text{ and } l_2 \\ l_4: & x < -2 & \text{from } l_1 \text{ and } l_3 \\ l_5: & -y < -4 & \text{from } l_0 \text{ and } l_4 \\ l_6: & x < -4 & \text{from } l_1 \text{ and } l_5 \\ l_7: & -y < -8 & \text{from } l_0 \text{ and } l_6 \\ \ldots & & \end{array}$$

So the introduction of the precedence and the restriction of the Fourier-Motzkin resolution inference to only eliminate the maximal variable is used to prevent this chain: if for instance $y \quad x$, then l_3 can be derived, but not l_4.

This restriction has a price with regards to completeness: Clearly, a smart strategy for LRA would try assigning values to Σ_{LRA}-variables according to the precedence order, starting from the lowest Σ_{LRA}-variable, which in our example would be y. But the completeness requirement cannot assume that this strategy is being employed, and module I_{LRA} must be able to extend any unsatisfiable assignment J such as

$$l_0, l_1, l_2, l_3, (x \leftarrow 0), \quad .$$

Clearly, there is no acceptable assignment for y, so we must find an inference that infers a Boolean assignment L that is not already present in J. This is where the elimination of empty solution spaces comes in: taking E to be l_0, l_1, we have the inference

$$(-2{\cdot}x - y < 0), (x + y < 0), (x \leftarrow 0) \quad {}_{\mathsf{LRA}}$$

Now this kind of inference is only useful for those cases, as above, where Σ_{LRA}-variables were not assigned according to the precedence order. Indeed, assume the premiss

$$J' \quad = \quad t_1 \leftarrow \tilde{q_1}, \ldots, t_m \leftarrow \tilde{q_m}, E$$

of an inference eliminating empty solution spaces is included in an assignment J, and assume J satisfies the property that if it assigns a value to a Σ_{LRA}-variable, then it also assigns a value to every lower Σ_{LRA}-variable according to . In that case, every Σ_{LRA}-variable of E is assigned a value in J, so E can be fully evaluated. Since J' is not satisfiable, at least one of the single Boolean assignment in E must be violated by this evaluation. So an evaluation inference can be used to extend J.

Lemma 3 If I_{LRA} cannot extend a plausible LRA^+-assignment J, then every term occurring in J of sort Q is assigned a value in J. ※

Proof: We first show that all Σ_{LRA}-variables in J of sort Q are assigned a value.
Let $t_1 \leftarrow \tilde{q_1}, \ldots, t_m \leftarrow \tilde{q_m}$ be the assignments in J for Σ_{LRA}-variables of sort Q, with $t_1 \quad \cdots \quad t_m$. Assume, by contradiction, that t is the smallest unassigned Σ_{LRA}-variable in J, according to .

- If $t \quad t_m$:
  Let E_J be the biggest Boolean assignment included in J such that any Σ_{LRA}-variable in $\mathsf{fv}^{\mathsf{Q}}_{\Sigma_{\mathsf{LRA}}}(E_J)$ is smaller than or equal to one of $t_1, \ldots, t_m$ according to . The assignment $t_1 \leftarrow \tilde{q_1}, \ldots, t_m \leftarrow \tilde{q_m}, E_J$ is LRA^+-satisfiable, otherwise I_{LRA} could extend J with an inference eliminating an empty solution space. Let q be the value of t in a model endorsing this assignment. We prove that $t \leftarrow \tilde{q}$ is acceptable for J: First, it cannot violate J with an inference eliminating empty solution spaces, because the Boolean assignment E involved in

such an inference is included in E_J, and q was chosen so that $t_1 \quad {}_{\mathsf{Q}}\, \tilde{q}_1, \ldots, t_m \quad {}_{\mathsf{Q}}\, \tilde{q_m}, t \quad {}_{\mathsf{Q}}\, \tilde{q}, E_J$ is LRA^+-satisfiable. Second, it cannot violate J with an evaluation inference, because the Boolean assignment L involved in such an inference is again in E_J. So $t \leftarrow \tilde{q}$ is acceptable for J and therefore I_{LRA} can extend J.

- If $t_m \quad t$:
  We first show that for any value q, if $(t \leftarrow \tilde{q})$ violates J in one I_{LRA}-step, then it does so with an evaluation inference: Indeed, if it did so with an inference eliminating an empty solution space, the Boolean assignment E involved in this inference is such that $t_1 \quad {}_{\mathsf{Q}}\, \tilde{q}_1, \ldots, t_m \quad {}_{\mathsf{Q}}\, \tilde{q_m}, t \quad {}_{\mathsf{Q}}\, \tilde{q}, E$ is LRA^+-unsatisfiable, and by minimality of t, all Σ_{LRA}-variables of E are assigned values in J; so there is an assignment L in E with an evaluation inference $t_1 \leftarrow \tilde{q_1}, \ldots, t_m \leftarrow \tilde{q_m}, t \leftarrow \tilde{q} \quad {}_{\mathsf{LRA}}\, \overline{L}$. In other words, if an assignment $(t \leftarrow \tilde{q})$ does not violate J in one evaluation step, then it is acceptable for J. In order to find such a value q, it suffices to look at all single Boolean assignments in J whose only free but unassigned Σ_{LRA}-variable is t, otherwise known as ***unit constraints*** on t. Moreover, as I_{LRA} cannot extend J, the conclusions of positivization and elimination of equalities inferences are already in J. So any Boolean assignment in J that is not of the form $e_1 \leq e_2$, $e_1 < e_2$, or $e_1 \quad {}_s\, e_2$, is redundant with the assignments of these forms. The space of solutions for a collection of unit constraints of these forms is an interval from which a finite range of points are excluded. We show that this space cannot be empty: It is empty if either the lower bound is greater than the upper bound, or the two bounds are equal but one of them is strict, or the two bounds are equal and large but a disequality removes the only possible point. In the first two cases, the conclusion L of the applicable Fourier-Motzkin resolution inference must already be present in J, while its flip $\overline{L}$ must also be present as the conclusion of the evaluation inference $J \quad {}_{\mathsf{LRA}}\, \overline{L}$. The third case is also ruled out as it would allow I_{LRA} to extend J with an elimination of disequality inference.

Finally, assume that a term t occurring in J is not a Σ_{LRA}-variable. Its Σ_{LRA}-variables are all assigned in J, so t has a clear value q according to these assignments. If $t \leftarrow \tilde{q}$ were not acceptable for J, then it would violate J with an evaluation inference. Another evaluation inference using the assigned Σ_{LRA}-variables of t instead of $t \leftarrow \tilde{q}$ itself would already violate J, so module I_{LRA} to extend J with that inference. □

Lemma 4 (Completeness) For any theory T_0 whose models interpret Q as an infinite set, module I_{LRA} is T_0-complete. ※

Proof: Let J be a plausible LRA^+-assignment and assume I^2 cannot extend J. As the previous lemma shows, this means that all terms occurring in J of sort Q are assigned values in J (and of course this is also the case for sort prop). Writing $J^{\downarrow}$ for the set of terms occurring in J of sort Q or prop, we build the following $\mathsf{LRA}^+[\mathsf{fv}_{\Sigma_{\mathsf{LRA}}}(J^{\downarrow})]$-model M: Let M's interpretation of Σ^+_{LRA} be the standard one, and let M interpret every Σ_{LRA}-variable in J as indicated by J. We show that M view-endorses J, which is here the same thing as simply endorsing J (since I_{LRA} cannot extend J and therefore J is closed under all equality inferences): Let $t \leftarrow \mathfrak{c}$ be an arbitrary assignment in J. If t is a Σ_{LRA}-variable, then $M(t) = \mathfrak{c}^{\mathcal{M}}$ by definition. Otherwise if t is a formula such that $M(t) = \mathfrak{c}^{\mathcal{M}}$, I_{LRA} would be able to extend J with an evaluation inference deriving $\overline{t \leftarrow \mathfrak{c}}$. And finally if t is a non-variable term of sort Q, I_{LRA} would be able to extend J with an evaluation inference deriving $t \quad_{\mathsf{Q}} t$.

Now let Σ_0 be $\mathcal{T}_0$'s signature and let $\mathcal{M}_0$ be a $\mathcal{T}_0[\mathsf{fv}_{\Sigma_0}(J)]$-interpretation that view-endorses J. Since $\mathsf{Q}^{\mathcal{M}_0}$ is infinite and Σ^+_{LRA} is countable, by Löwenheim-Skolem theorem, $\mathcal{M}$ can be transformed so that $|\mathsf{Q}^{\mathcal{M}}| = |\mathsf{Q}^{\mathcal{M}_0}|$, and for all t and t' in $J^{\downarrow}$, $\mathcal{M}(t) = \mathcal{M}(t')$ if and only if $(t \quad_s t') \quad J$ if and only if $\mathcal{M}_0(t) = \mathcal{M}_0(t')$. □

5.3 Equality with Uninterpreted Function symbols (EUF)

Theory EUF can be decided by a congruence closure procedure, which can be integrated to our framework in one of the two ways described in Section 5.5.

Alternatively, one can restrict EUF-inferences to only apply to basic forms of unsatifiability: Given an arbitrary signature $\Sigma_{\mathsf{EUF}} = (S, \quad_S \quad F)$ for EUF, we can use as EUF-inferences

$$(t_i \quad u_i)_{i=1\dots m}, (f(t_1, \dots, t_m) \quad f(u_1, \dots, u_m)) \quad_{\mathsf{EUF}}$$

for all symbols $f \quad F$. For the local basis, we let $\mathsf{basis}_{\mathsf{EUF}}(X)$ extends X with $\quad$, as well as with any equality $l \quad_{\mathsf{prop}} l'$ such that either l and l' are two formulæ in X with the same symbol $f \quad F$ at their root, or there are in X two terms $f(t_1, \dots, t_m, l, u_1, \dots, u_m)$ and $f(t'_1, \dots, t'_m, l', u'_1, \dots, u'_m)$ with $f \quad F$.

This describes a lazy module $\mathcal{I}_{\mathsf{EUF}}$ for EUF, similar to that used in [18], which will not propagate anything before equalities between existing terms are determined to be in contradiction with the congruence axiom.

As in Section 5.5, we may or may not decide to give ourselves countably many values for each sort in $S \backslash \{\mathsf{prop}\}$. In both cases if module $\mathcal{I}_{\mathsf{EUF}}$ cannot extend an assignment, then all equalities are determined. The proof below is for the extension EUF^+ with the extra values.

Lemma 5 (Completeness) For any theory $\mathcal{T}_0$, module $\mathcal{I}_{\mathsf{EUF}}$ is $\mathcal{T}_0$-complete. ※

Proof: Let J be a plausible EUF^+-assignment and assume $\mathcal{I}_{\mathsf{EUF}}$ cannot extend J. As shown in Lemma 1, all terms occurring in J are assigned values. Now let Σ_0 be $\mathcal{T}_0$'s signature and let $\mathcal{M}_0$ be a $\mathcal{T}_0[\mathsf{fv}_{\Sigma_0}(J)]$-interpretation that view-endorses J. We build the following $\mathsf{EUF}^+[\mathsf{fv}_{\Sigma}(J)]$-interpretation $\mathcal{M}$: it interprets the sorts in S as in $\mathcal{M}_0$; it interprets any Σ-variable t in J as $\mathcal{M}_0(t)$; it interprets any EUF-value v assigned in J to some term t as $\mathcal{M}_0(t)$;[2] it interprets any other EUF-value as an arbitrary element; and finally it interprets every symbol $f : (s_1 \times \dots \times s_m) \to s$ in F as follows: given elements $e_1 \quad s_1^{\mathcal{M}_0}, \dots, e_m \quad s_m^{\mathcal{M}_0}$, if there is in J a term $f(t_1, \dots, t_m)$ with $\mathcal{M}_0(t_1) = e_1, \dots, \mathcal{M}_0(t_m) = e_m$, then define $f^{\mathcal{M}}(e_1, \dots, e_m)$ as $\mathcal{M}_0(f(t_1, \dots, t_m))$,[3] otherwise define it as an arbitrary element of $s^{\mathcal{M}_0}$. A straightforward induction on t shows that, whenever $(t \leftarrow \mathfrak{c}) \quad J$, we have $\mathcal{M}(t) = \mathcal{M}_0(t) = \mathfrak{c}^{\mathcal{M}}$, which concludes the proof. □

The same property holds for the trivial extension of EUF, which has no extra values but the Boolean ones. The proof is almost identical, except that non-Boolean terms occurring in J are

[2]If there are two such terms, $\mathcal{M}_0$ must interpret them identically.

[3]If there are two such terms $f(t_1, \dots, t_m)$ and $f(u_1, \dots, u_m)$, then by Lemma 1, J must contain

$$(t_1 \simeq u_1), \dots, (t_m \simeq u_m), f(t_1, \dots, t_m) \simeq f(u_1, \dots, u_m)$$

(otherwise $\mathcal{I}_{\mathsf{EUF}}$ could extend J with an inference), so $\mathcal{M}_0(f(t_1, \dots, t_m)) = \mathcal{M}_0(f(u_1, \dots, u_m))$.

not assigned values, but as for any two terms t and u of sort $s \quad S\backslash\{\mathsf{prop}\}$ we still have a value for $t \quad_s u$ in J, so the argument can be easily adapted.

We could also add some more eager EUF-inferences, without having to change the completeness argument:[4]

$$\begin{array}{rcl} (t_i \quad u_i)_{i=1\ldots n} & \mathsf{EUF} & (f(t_1,\ldots,t_m) \quad f(u_1,\ldots,u_m)) \\ (t_i \quad u_i)_{i=1\ldots m, i\neq i_0}, f(t_1,\ldots,t_m) \quad f(u_1,\ldots,u_m) & \mathsf{EUF} & t_{i_0} \quad u_{i_0} \end{array}$$

for all symbols $f \quad F$.

5.4 Arrays (Arr)

The theory of arrays Arr can be decided by an algorithm that can be integrated to our framework in one of the two ways described in Section 5.5. Alternatively, one can restrict Arr-inferences to only apply to basic forms of unsatifiability, leading to a theory module for Arr that is similar to the EUF module(s) presented above.

Consider a signature $\Sigma_{\mathsf{Arr}} = (S, F)$, where S is the free closure of a set S_{basic} of basic sorts (that includes prop) under the binary array sort constructor, which from an ***index sort*** I and a ***value sort*** V builds the ***array sort*** $I \quad V$, and where F is

$$\begin{array}{lll} {}_S & \{(\mathsf{select}_{I\Rightarrow V} : (I \quad V) \times I \to V) & \mid (I \quad V) \quad S\} \\ & \{(\mathsf{store}_{I\Rightarrow V} : (I \quad V) \times I \times V \to (I \quad V))) & \mid (I \quad V) \quad S\} \\ & \{(\mathsf{di}\square_{I\Rightarrow V} : (I \quad V) \times (I \quad V) \to I) & \mid (I \quad V) \quad S\} \end{array}$$

When sorts can be inferred we sometimes omit writing them as subscripts of the symbols in F. We further abbreviate $\mathsf{store}(a,i,v)$ as $a[i{:=}v]$ and $\mathsf{select}(a,i)$ as $a[i]$.

As with EUF and with the approach described in Section 5.5, we can take as extension Arr^+ of Arr either the trivial one or the one that adds an infinite countable set of values to each sort in S. We formalise the module with the latter option, but the former one could be given as easily.

For this module I_{Arr} we take as inferences:

$$\begin{array}{rcl} (t \quad t'), (i \quad i'), (t[i] \quad t'[i']) & \mathsf{Arr} & \\ (t \quad t'), (i \quad i'), (u \quad u'), (t[i{:=}u] \quad t'[i'{:=}u']) & \mathsf{Arr} & \\ (t \quad t'), (u \quad u'), (\mathsf{di}\square(t,u) \quad \mathsf{di}\square(t',u')) & \mathsf{Arr} & \\ (t' \quad t[i{:=}u]), (i \quad j), (u \quad t'[j]) & \mathsf{Arr} & \\ (t' \quad t[i{:=}u]), (i \quad j), (j \quad j'), (t[j] \quad t'[j']) & \mathsf{Arr} & \\ (t \quad u) & \mathsf{Arr} & (t[\mathsf{di}\square(t,u)] \quad u[\mathsf{di}\square(t,u)]) \end{array}$$

The first three inference rules simply express the congruence property of the signature symbols. The fourth and fifth are simply the expression of the traditional axioms as inference rules that can handle equalities. The last axiom is the only one that can produce new terms, and is the expression as an inference rule of the (Skolemized) extensionality axiom.

For the local basis we define $\mathsf{basis}_{\mathsf{Arr}}(X)$ as the smallest closed set Y containing $X \quad \{ \quad \}$ and satisfying the following closure properties:

[4] In fact, this can be done for any theory treated as in Section 5.5, the main question being how to detect the applicability of such inferences.

- If t and u are in Y then so are $t[\mathsf{di}□(t,u)]$ and $u[\mathsf{di}□(t,u)]$;
- If l_1 and l_2 are subterms of sort prop of some terms in Y whose root symbol is select, store, or $\mathsf{di}□$, then $l_1 \quad_{\mathsf{prop}} l_2$ is in Y.

Note that this set is finite (in particular, $\mathsf{di}□$ only produces terms whose sorts are structurally smaller than that of its arguments).

In order to express the completeness property, we identify the following notion: an ***updatable function set*** from U to V is a subset of the set of functions from U to V that is stable under finite modifications of their graphs.

Lemma 6 (Completeness) For any theory T_0 whose models M_0 are such that $(I \quad V)^{\mathcal{M}_0}$ has the cardinality of an updatable function set from $I^{\mathcal{M}_0}$ to $V^{\mathcal{M}_0}$, module I_{Arr} is T_0-complete. ※

Proof: Let J be a plausible Arr^+-assignment and assume I_{Arr} cannot extend J. As shown in Lemma 1, all terms occurring in J are assigned values. Now let Σ_0 be T_0's signature and let M_0 be a $T_0[\mathsf{fv}_{\Sigma_0}(J)]$-interpretation that view-endorses J.

Let $I \quad V$ be an array sort, let X be an updatable set of functions from $I^{\mathcal{M}_0}$ to $V^{\mathcal{M}_0}$ that is in bijection with $(I \quad V)^{\mathcal{M}_0}$, and let $f_0 \quad X$. We start by defining a bijective function ϕ from $(I \quad V)^{\mathcal{M}_0}$ to X.

For this we first define the restriction ϕ_Y of ϕ on the (finite) subset Y of $(I \quad V)^{\mathcal{M}_0}$ consisting of those elements a such that at least one term occurring in J is interpreted as a by M_0: For such an element a, we consider the binary relation $R_a \quad I^{\mathcal{M}_0} \times V^{\mathcal{M}_0}$ defined as follows:

$$\{(M_0(i), M_0(t[i])) \mid t[i] \text{ occurring in } J \text{ with } M_0(t) = a\}$$
$$\{(M_0(i), M_0(u)) \mid t[i{:=}u] \text{ occurring in } J \text{ with } M_0(t[i{:=}u]) = a\}$$
$$\{(M_0(i), M_0(t[i])) \mid t[j{:=}u] \text{ occurring in } J \text{ with } M_0(t[j{:=}u]) = a \text{ and } M_0(i) = M_0(j)\}$$

This relation is (finite and) functional (otherwise I_{Arr} could extend J), and is therefore a (finite and) partial function from $I^{\mathcal{M}_0}$ to $V^{\mathcal{M}_0}$; we extend it into a total function $\phi_Y(a)$ by mapping any remaining element c in $I^{\mathcal{M}_0}$ to the element $f_0(c) \quad V^{\mathcal{M}_0}$. As $\phi_Y(a)$ differs from f_0 by only finitely many modifications, $\phi_Y(a)$ is in X. Moreover, ϕ_Y is injective: For any two elements $a = M_0(t)$ and $a' = M_0(t')$ with $a = a'$, J must contain $t \quad t'$ and therefore $t[\mathsf{di}□(t,t')] \quad t'[\mathsf{di}□(t,t')]$ (otherwise I_{Arr} could extend J); so $\phi_Y(a)(M_0(\mathsf{di}□(t,t'))) = M_0(t[\mathsf{di}□(t,t')])$ is different from $\phi_Y(a')(M_0(\mathsf{di}□(t,t'))) = M_0(t'[\mathsf{di}□(t,t')])$ and therefore $\phi_Y(a) = \phi_Y(a')$. Given that the ϕ_Y is injective and that $(I \quad V)^{\mathcal{M}_0}$ is in bijection with X, we can extend ϕ_Y into a bijection ϕ from $(I \quad V)^{\mathcal{M}_0}$ to X.

Now we build an $\mathsf{Arr}^+[\mathsf{fv}_\Sigma(J)]$-interpretation M as follows: M interprets every sort s like M_0 does, it interprets any Σ-variable t in J as $M_0(t)$; it interprets any Arr-value $\mathfrak{c}$ assigned in J to some term t as $M_0(t)$;[5] and it interprets any other Arr-value as an arbitrary element. We are left with the interpretations of the three kinds of symbols select, store and $\mathsf{di}□$: Given any array sort $I \quad V$, we define

- $\mathsf{select}^{\mathcal{M}}_{I\Rightarrow V}$ as the function mapping any $(a,c) \quad (I \quad V^{\mathcal{M}}) \times I^{\mathcal{M}}$ to $\phi(a)(c) \quad V^{\mathcal{M}}$;
- $\mathsf{store}^{\mathcal{M}}_{I\Rightarrow V}$ as the function mapping any $(a,c,d) \quad (I \quad V^{\mathcal{M}}) \times I^{\mathcal{M}} \times V^{\mathcal{M}}$ to $\phi^{-1}(f) \quad (I \quad V^{\mathcal{M}})$, where f is the function that maps c to d and any other $c' \quad I^{\mathcal{M}}$ to $\phi(a)(c') \quad V^{\mathcal{M}}$;

[5]If there are two such terms, $\mathcal{M}_0$ must interpret them identically.

- $\mathsf{di}□^{\mathcal{M}}_{I\Rightarrow V}$ as the function mapping any (a, a') $(I \quad V^{\mathcal{M}}) \times (I \quad V^{\mathcal{M}})$ with $a = a'$ to an element $c \quad I^{\mathcal{M}}$ such that $\phi(a)(c) = \phi(a')(c)$, and mapping any pair (a, a) to an arbitrary element of $I^{\mathcal{M}}$.

Clearly by construction, M satisfies the axioms of the array theory, and is therefore a $\mathsf{Arr}^{+}[\mathsf{fv}_{\Sigma}(J)]$-interpretation. A straightforward induction on t shows that, whenever $(t \leftarrow \mathfrak{c}) \quad J$, we have $M(t) = M_0(t) = \mathfrak{c}^{\mathcal{M}}$, which concludes the proof. □

Again, the same property holds for the trivial extension of Arr, which has no extra values but the Boolean ones. The proof is almost identical, except that non-Boolean terms occurring in J are not assigned values, but as for any two terms t and u of sort $s \quad S \setminus \{\mathsf{prop}\}$ we still have a value for $t \quad_s u$ in J, so the argument can be easily adapted.

5.5 Theories with procedures suited for Nelson-Oppen combination

Assume T is a stably infinite theory on signature $\Sigma = (S, F)$, with a procedure deciding the satisfiability of conjunctions of literals. A first way to accommodate such a theory and procedure into our framework is to take as extension T^{+1} the trivial one, whose signature we denote Σ^{+1}, and define a module $I^{1} = (\quad_{\mathcal{T}}, \mathsf{basis}_{\mathcal{T}})$ for theory T with extension T^{+1} as follows. The local basis $\mathsf{basis}_{\mathcal{T}}$ only adds the formula (i.e. , $\mathsf{basis}_{\mathcal{T}}(X) = X \quad \{ \quad \}$ for all X). The I^{1}-inference system features a single inference rule

$$l_1 \leftarrow \mathfrak{b}_1, \ldots, l_m \leftarrow \mathfrak{b}_m \quad_{\mathcal{T}}$$

where $l_1, \ldots, l_m$ are formulæ, and the conjunction of the literals corresponding to the Boolean assignments $l_1 \leftarrow \mathfrak{b}_1, \ldots, l_m \leftarrow \mathfrak{b}_m$ is found T-unsatisfiable by the decision procedure.

Lemma 7 (Completeness) For any theory T_0 whose models interpret each sort in $S \setminus \{\mathsf{prop}\}$ as an infinite countable set, the T-module I^{1} is T_0-complete. ※

Proof: Let J be a plausible T^{+1}-assignment. If I^{1} cannot extend J, Lemma 1 concludes that all formulae that occur in J are assigned values in J, and for any two terms t and t' occurring in J of sort s in $S \setminus \{\mathsf{prop}\}$, the equality $t \quad_s t'$ is assigned a value in J. Also, the conjunction of the literals corresponding to the Boolean assignments in J is T-satisfiable: otherwise I^{1} could extend J with inference $J \quad_{\mathcal{T}}$. By interpreting the Boolean values as themselves we get a $T^{+1}[\mathsf{fv}_{\Sigma}(J)]$-interpretation M that view-endorses J. Since T is stably infinite, we can assume w.l.o.g. that it interprets every sort $S \setminus \{\mathsf{prop}\}$ as an infinite countable set. Let Σ_0 be T_0's signature. Given any $T_0[\mathsf{fv}_{\Sigma_0}(J)]$-interpretation M_0 that view-endorses J, we are left to check that for all terms t and t' occurring in J of some sort $s \quad S$, $M(t) = M(t')$ if and only if $M_0(t) = M_0(t')$. This is the case, since either s is prop and each of $M(t) = M(t')$ and $M_0(t) = M_0(t')$ occur if and only if t and t' are assigned the same value in J, or s is not prop and each of $M(t) = M(t')$ and $M_0(t) = M_0(t')$ occur if and only if $(t \quad_s t' \leftarrow \mathsf{true})$ is in J. □

Note that the above would also work if we only took as I^{1}-inferences those inferences $J \quad_{\mathcal{T}}$ where J is an *unsatisfiable core* (removing any single assignment from J would result in a T-satisfiable conjunction of literals).

The second way to accommodate such a theory and procedure into our framework is to take as extension T^{+2} the theory whose signature Σ^{+2} adds to Σ an infinite countable collection of T^{+2}-values $\mathfrak{c}_0^s, \ldots, \mathfrak{c}_i^s, \ldots$ for each sort s in $S\backslash\{\mathsf{prop}\}$, but that does not assume anything about these new values. We then define the module I^2 (for theory T with extension T^{+2}) as ($_{\mathcal{T}}$, $\mathsf{basis}_{\mathcal{T}}$) again, i.e. with the exact same inferences and local basis as in I^1. In other words, the equality inferences are the only inferences that make use of first-order T^{+2}-assignments, inferring equalities and disequalities between terms that are assigned identical or distinct T^{+2}-values. In effect, T^{+2}-values are thus used to label the equivalence classes of terms: For instance the T^{+2}-assignment

$$t_1 \leftarrow \mathfrak{c}, t_2 \leftarrow \mathfrak{c}, t_3 \leftarrow \mathfrak{c}_3, t_4 \leftarrow \mathfrak{c}_4, t_5 \leftarrow \mathfrak{c}_5$$

encodes the four equivalence classes given by the literals

$$t_1 \quad t_2, t_1 \quad t_3, t_1 \quad t_4, t_1 \quad t_5, t_3 \quad t_4, t_3 \quad t_5, t_4 \quad t_5$$

Lemma 8 (Completeness) For any theory T_0 whose models interpret each sort in $S\backslash\{\mathsf{prop}\}$ as an infinite countable set, the T-module I^2 is T_0-complete. ※

Proof: Let J be a plausible T^{+2}-assignment and assume I^2 cannot extend J. As shown by Lemma 1, this means that all terms occurring in J of a sort in S are assigned values in J. It also means that for any two terms t and t' occurring in J of sort s in $S\backslash\{\mathsf{prop}\}$, the equality $t \quad_s t'$ is assigned a value in J: Indeed, $(t \quad_s t')$ belongs to the closed set $\mathsf{basis}_{\mathcal{T}}(J)$, and since t and t' are both assigned values in J, I^2 could extend J with an equality inference concluding $t \quad_s t'$ or $t \quad_s t'$, unless $t \quad_s t'$ is already assigned a value in J. Finally, as for I^1, we also conclude that the conjunction of the literals corresponding to the Boolean assignments in J is T-satisfiable. So there exists a T-model of those literals, which we turn into a $T^{+2}[\mathsf{fv}_\Sigma(J)]$-interpretation M that view-endorses J as follows: M interprets the sorts in S, the symbols in F, and the Σ-variables as in the T-model; the Boolean values are interpreted as themselves; and for a non-Boolean T^{+2}-value v of sort s, either it is never used in J, in which case we let M interpret it arbitrarily in $s^{\mathcal{M}}$, or there is some assignment $t \leftarrow \mathfrak{c}$ in J, in which case we define $v^{\mathcal{M}}$ as the interpretation of term t in the T-model.[6] That makes M view-endorse J. The rest of the argument is the same as for I^1. □

As for I^1, the above would also work if we only took as I^2-inferences those inferences $J \quad_{\mathcal{T}}$ where J is an *unsatisfiable core*.

The above shows that the Nelson-Oppen combinations of theories form a particular case of our framework: The triviality of the basis corresponds to the fact that the procedures in a Nelson-Oppen combination do not introduce new literals (literals that were not present in the original problem), and their interaction does not introduce any new literals either, save for the equalities between shared variables that an *arrangement* determines. The unability for one theory to extend an assignment J entails that the theory, on its own, has a model, but also entails that every equality is determined by J, which is thereby defining a particular *arrangement* between shared variables. If the other theories cannot extend J either, then a model for the union of the theories exists, a property that is a consequence of our completeness requirement (see Section 8.3).

Module I^2 is a slight optimisation of module I^1 that allows the representation of determined

[6] Note that if there are two such assignments in J, say $t_1 \leftarrow \mathfrak{c}$ and $t_2 \leftarrow \mathfrak{c}$, we know that $(t_1 \simeq_s t_2) \in J$ and therefore the $\mathcal{T}$-model interprets t_1 and t_2 as the same value.

equalities without explicitly listing the literals $t_1 \quad {}_s t_2$ or $t_1 \quad {}_s t_2$ (in the worst case, quadratic in the number of terms), but using CDSAT values as identifiers for equivalence classes of terms.

This is one possible use of CDSAT values. The next examples of theories are theories for which a decision procedure could fit into the framework as described above, but for which a more interesting theory module could be defined, with a more interesting way of using values.

6 Abstract calculus

In this section we present our calculus. We start with the data structures that it uses:

Definition 18 (Trail) A *trail* is

- a finite DAG whose vertices are singleton assignments $t \leftarrow \mathfrak{c}$ whose non-root vertices are all Boolean assignments, and such that if $t \leftarrow \mathfrak{c}_1$ and $t \leftarrow \mathfrak{c}_2$ are two vertices then $\mathfrak{c}_1$ and $\mathfrak{c}_1$ are non-Boolean $\mathcal{T}_i^+$- and $\mathcal{T}_j^+$-values, respectively, for distinct i and j;
- a collection of roots marked as *decisions* and equipped with a total order denoted $<$.

※

The total order allows the numbering of decisions from smallest to greatest, assigning to each decision a number traditionally called its *level*,[7] as we do in Section 9.

The calculus uses the following concepts and notations about trails. A trail Γ can always be seen as an assignment (its set of vertices, forgetting about edges), so we can freely use with Γ every notation defined for assignments. Given a non-decision vertex A in Γ, we write $\mathsf{expl}_\Gamma(A)$ for the predecessors of A in Γ. Given a set of vertices J, the set of decisions that have a path to some vertex in J is denoted $\mathsf{dec}_\Gamma(J)$, and the greatest decision in it (when it is non-empty) is denoted $\mathsf{dec}_\Gamma^{\mathsf{sup}}(J)$. An assignment J *is in* Γ if every single assignment in J is a vertex in Γ. In this case, and for any single Boolean assignment L such that neither L nor $\overline{L}$ is a vertex of Γ, we write $\Gamma, (J \quad L)$ for the trail extending Γ with a new vertex for L and a new edge from each single assignment of J to L. By extension, we say that a $\mathcal{T}$-inference $(J \quad {}_{\mathcal{T}} L)$ *builds on* Γ if J is in Γ and L is not in Γ. The trail Γ, A is the extension of Γ with a new root A marked as a decision, and greater than any decision in Γ. If D is a set of decision roots in Γ, the trail denoted $\Gamma \setminus D$ is obtained from Γ by removing the decisions in D and every vertex that is reachable from one of these decisions. If A is a root of Γ and A' is a new assignment, the trail denoted $\Gamma \setminus A \curvearrowleft A'$ is obtained by extending $\Gamma \setminus A$ with a new root A' marked as a decision, and inserted in the order at the same position as A.

The calculus is a state transition system:

Definition 19 (Search state, conflict state) *States* are of two kinds:

- *search states*, which are simply trails
- *conflict states*, which are of the form $\ \Gamma; E; A\ $, where Γ is a trail, E is a subset of vertices in Γ, and $A = \mathsf{dec}_\Gamma^{\mathsf{sup}}(E)$.

※

[7] We use an order rather than numbers, because the calculus will allow the removal of a decision without necessarily removing the greater decisions, thus triggering a re-numbering, should numbers be used.

Search rules

Decide

$\Gamma \quad \dashrightarrow_{\mathcal{B}} \quad \Gamma, A$ if A is a T_k^+-assignment for a T_k^+-relevant term of Γ that is acceptable for $\Gamma_{\mathcal{T}_k}$ and I_k, with $1 \leq k \leq n$

In the next three rules we assume an inference $J \quad_{\mathcal{I}_k} L$ builds on Γ, with L being an assignment for a formula in B and $1 \leq k \leq n$.

Propagate

$\Gamma \quad \dashrightarrow_{\mathcal{B}} \quad \Gamma, (J \quad L)$ if $\overline{L}$ not in Γ

Conflict

$\Gamma \quad \dashrightarrow_{\mathcal{B}} \quad \Gamma'$ if $\overline{L}$ in Γ, $A = \mathsf{dec}^{\mathsf{sup}}_{\Gamma}(J, \overline{L})$, and $\Gamma; J, \overline{L}; A \quad = \quad ^* \Gamma'$

Fail

$\Gamma \quad \dashrightarrow_{\mathcal{B}} \quad \mathsf{unsat}$ if $\overline{L}$ in Γ, and $\mathsf{dec}_{\Gamma}(J, \overline{L}) =$

Conflict analysis rules

Resolve

$\Gamma; E, L; A \quad = \quad \Gamma; E \quad \mathsf{expl}_{\Gamma}(L); A$ if L is not a decision, $A \quad \mathsf{dec}_{\Gamma}(L)$, and either A is Boolean or A / $\mathsf{expl}_{\Gamma}(L)$

UIPBackjump

$\Gamma; E, L; A \quad = \quad (\Gamma \setminus D), (E \quad \overline{L})$ if D is a set of decisions with $A \quad D$ and $A' \quad \mathsf{dec}_{\Gamma}(E)$, $A'' \quad D, A' < A''$

SemSplit

$\Gamma; E, L; A \quad = \quad \Gamma \setminus A \curvearrowleft \overline{L}$ if A is a non-Boolean assignment in $\mathsf{dec}_{\Gamma}(E) \cap \mathsf{expl}_{\Gamma}(L)$

Undo

$\Gamma; E, A; A \quad = \quad \Gamma \setminus A$ if A is a non-Boolean assignment not in $\mathsf{dec}_{\Gamma}(E)$

Figure 3: The CDSAT transition system

The intuition of a conflict state $\Gamma; E; A$ is that E is a set of conflicting assignments in Γ, and A is the greatest decision that contributes to the conflict:[8] we shall undo (at least) that decision before switching back to a search state.

Definition 20 (Calculus) An *input problem* is an assignment.

The *initial state* corresponding to a given input problem $X = \{A_1, \ldots, A_m\}$ is the search state whose vertices are the assignments in X, all of which are non-decision roots.

The MCSAT system, presented in Fig. 3, is a transition system between search states, parameterised by a set B of terms called *basis*. Such transitions are denoted e.g. $\Gamma \dashrightarrow_{\mathcal{B}} \Gamma'$ and called

[8] A is always determined by Γ and E, but keeping it explicitly in the state simplifies the rules.

B-transitions. The system also uses an auxiliary transition system for ***conflict analysis***, whose single steps are denoted with $=$ and multi-steps are denoted $=\ ^*$. ※

We say that an input problem is Boolean, or an SMT-problem, if it is a Boolean assignment. Otherwise it is more generally an SMA problem.

Notice that the side-condition for rule UIPBackjump implies in particular that A has no path to any vertex in E (A / $\mathsf{dec}_\Gamma(E)$). In this system we have not included rules for learning, forgetting or restarting. We address learning in Section 10, mostly to show that it can be included without breaking our termination argument, while we leave the forgetting and restarting features out of our abstract description. Finally, we introduce the notion of ***level***, which will be used in Sections 8.1 and 9.1 about soundness and termination.

Definition 21 (Level i assignment in a trail) Let Γ be a trail, with $A_1, \ldots, A_m$ being its decisions unambiguously enumerated from smallest to greatest. For all i in $1, \ldots, m$, the set L_i^Γ of ***level*** i ***assignments in*** Γ is defined as the set of all assignments A in Γ such that $A_i = \mathsf{dec}_\Gamma^{\mathsf{sup}}(\{A\})$ (i.e. A_i is the greatest decision with a path to A). The set L_0^Γ of ***level*** 0 ***assignments in*** Γ is the set of all assignments A in Γ such that $\mathsf{dec}_\Gamma(\{A\}) =$ (i.e. no decision has a path to A). ※

We write $L_{\leq i}^\Gamma$ for $\bigcup_{j=0}^{i} L_j^\Gamma$.

7 Example with arithmetic, EUF, and arrays

In this section, we illustrate how CDSAT works in the case of Example 2. Fig. 4 shows a CDSAT derivation, where assignments in sort Q are LRA$^+$-assignments, and assignments in sort V are T_V-assignments, where T_V can be taken to be Arr$^+$, EUF, or another theory altogether.

We start with the four non-decision roots on line 0 (l0). Theory LRA then makes the decision $(w \leftarrow 0)$ (any other value was possible there) (l1). Three decisions follow on lines 2,3,4: From the point of view of LRA, these are really consequences of the decision $(w \leftarrow 0)$ (and, for the last two, of the input equalities), but that fact is private to the theory: from the point of view of the main calculus, they are all decisions. Theory T_V then makes the decision $(u \leftarrow \mathfrak{c}_1)$, where $\mathfrak{c}_1$ is a random T_V-value of sort V, followed by $(v \leftarrow \mathfrak{c}_1)$ (as in arithmetic, this is a consequence for T_V of $(u \leftarrow \mathfrak{c}_1)$ and the equality $u \quad v$, but is seen as a decision by the main calculus). Theory T_V then makes the decision $(a[i{:=}v][j] \leftarrow \mathfrak{c}_2)$, with another random value different from $\mathfrak{c}_1$ (there is no reason why the same one should be picked) (l7). Theory Arr picks up on this, spotting an incoming conflict: the equalities $(v \quad a[i{:=}v][j])$ and $(a[i{:=}v] \quad a[i{:=}v])$ are inferred from the assignments (l8,l9), and the calculus switches to a conflict state (l10) because of the Arr-inference $(i \quad j), (v \quad a[i{:=}v][j]), (a[i{:=}v] \quad a[i{:=}v]) \quad_{\mathsf{Arr}}$.[9] Rule Resolve applies on $a[i{:=}v] \quad a[i{:=}v]$, which removes it from the conflict as it has no predecessors (l11). Then only rule UIPBackjump applies, yielding the search state Γ_3 (l12). Theory T_V then makes the decision $(a[i{:=}v][j] \leftarrow \mathfrak{c}_1)$, the only acceptable one according to the trail (l13). Theory EUF picks up on this, spotting an incoming conflict, and the assignments $f(u) \quad f(a[i{:=}v][j])$ and $u \quad a[i{:=}v][j]$ are inferred, yielding the trail Γ_5. The calculus switches to a conflict state (l16), because of the

[9]We assume for simplicity that $\top$ is in the trail, otherwise it can be introduced at that point.

0 $(f(a[i{:=}v][j]) \quad w) \quad , \quad (f(u) \quad w-2) \quad , \quad (i \quad j) \quad , \quad (u \quad v) \qquad =: \Gamma_0$
1 $\Gamma_0, (w \leftarrow 0)$
2 $\Gamma_0, (w \leftarrow 0), (w-2 \leftarrow -2)$
3 $\Gamma_0, (w \leftarrow 0), (w-2 \leftarrow -2), (f(u) \leftarrow -2)$
4 $\Gamma_0, (w \leftarrow 0), (w-2 \leftarrow -2), (f(u) \leftarrow -2), (f(a[i{:=}v][j]) \leftarrow 0) \qquad =: \Gamma_1$
5 $\Gamma_1, (u \leftarrow \mathfrak{c}_1)$
6 $\Gamma_1, (u \leftarrow \mathfrak{c}_1), (v \leftarrow \mathfrak{c}_1)$
7 $\Gamma_1, (u \leftarrow \mathfrak{c}_1), (v \leftarrow \mathfrak{c}_1), (a[i{:=}v][j] \leftarrow \mathfrak{c}_2)$
8 $\Gamma_1, (u \leftarrow \mathfrak{c}_1), (v \leftarrow \mathfrak{c}_1), (a[i{:=}v][j] \leftarrow \mathfrak{c}_2), (v, a[i{:=}v][j] \quad v \quad a[i{:=}v][j])$
9 $\Gamma_1, (u \leftarrow \mathfrak{c}_1), (v \leftarrow \mathfrak{c}_1), (a[i{:=}v][j] \leftarrow \mathfrak{c}_2), (v, a[i{:=}v][j] \quad v \quad a[i{:=}v][j]), (\quad a[i{:=}v] \quad a[i{:=}v]) \qquad =: \Gamma_2$
10 $\Gamma_2; \quad i \quad j,\ v \quad a[i{:=}v][j],\ a[i{:=}v] \quad a[i{:=}v] \quad ; \quad a[i{:=}v][j] \leftarrow \mathfrak{c}_2$
11 $\Gamma_2; \quad i \quad j,\ v \quad a[i{:=}v][j] \quad ; \quad a[i{:=}v][j] \leftarrow \mathfrak{c}_2$
12 $\Gamma_1, (u \leftarrow \mathfrak{c}_1), (v \leftarrow \mathfrak{c}_1), (i \quad j \quad v \quad a[i{:=}v][j]) \qquad =: \Gamma_3$
13 $\Gamma_3, (a[i{:=}v][j] \leftarrow \mathfrak{c}_1) \qquad =: \Gamma_4$
14 $\Gamma_4, (f(u), f(a[i{:=}v][j]) \quad f(u) \quad f(a[i{:=}v][j]))$
15 $\Gamma_4, (f(u), f(a[i{:=}v][j]) \quad f(u) \quad f(a[i{:=}v][j])), (u, a[i{:=}v][j] \quad u \quad a[i{:=}v][j]) \qquad =: \Gamma_5$
16 $\Gamma_5; \quad f(u) \quad f(a[i{:=}v][j]),\ u \quad a[i{:=}v][j] \quad ; \quad a[i{:=}v][j] \leftarrow \mathfrak{c}_1$
17 $\Gamma_3, (f(u), f(a[i{:=}v][j]) \quad f(u) \quad f(a[i{:=}v][j])), (f(u) \quad f(a[i{:=}v][j]) \quad u \quad a[i{:=}v][j]) \qquad =: \Gamma_6$
18 $\Gamma_6; \quad u \quad v,\ v \quad a[i{:=}v][j],\ u \quad a[i{:=}v][j] \quad ; \quad f(a[i{:=}v][j]) \leftarrow 0$
19 $\Gamma_6; \quad u \quad v,\ v \quad a[i{:=}v][j],\ f(u) \quad f(a[i{:=}v][j]) \quad ; \quad f(a[i{:=}v][j]) \leftarrow 0$
20 $\Gamma_0, (w \leftarrow 0), (w-2 \leftarrow -2), (f(u) \leftarrow -2), (u \leftarrow \mathfrak{c}_1), (v \leftarrow \mathfrak{c}_1), (i \quad j \quad v \quad a[i{:=}v][j]), \ldots$
21 $\ldots (u \quad v, v \quad a[i{:=}v][j] \quad f(u) \quad f(a[i{:=}v][j])) \qquad =: \Gamma_7$
22 $\Gamma_7, (f(a[i{:=}v][j]) \quad w, f(u) \quad f(a[i{:=}v][j]) \quad f(u) \quad w) \qquad =: \Gamma_8$
23 $\Gamma_8; \quad w \leftarrow 0,\ f(u) \leftarrow -2,\ f(u) \quad w \quad ; \quad f(u) \leftarrow -2$
24 $\Gamma_0, (w \leftarrow 0), (w-2 \leftarrow -2), (u \leftarrow \mathfrak{c}_1), (v \leftarrow \mathfrak{c}_1), (i \quad j \quad v \quad a[i{:=}v][j]), \ldots$
25 $\ldots (u \quad v, v \quad a[i{:=}v][j] \quad f(u) \quad f(a[i{:=}v][j])), \ldots$
26 $\ldots (f(a[i{:=}v][j]) \quad w, f(u) \quad f(a[i{:=}v][j]) \quad f(u) \quad w) \qquad =: \Gamma_9$
27 $\Gamma_9, (f(u) \quad w-2, f(u) \quad w \quad w \quad w-2)$
28 unsat

Figure 4: Example

EUF-inference $(u \quad a[i{:=}v][j]), (f(u) \quad f(a[i{:=}v][j])) \quad _{\mathsf{EUF}}$. Again, only rule Resolve applies, yielding the search state Γ_6. At this point, Theory T_V cannot pick a value for $a[i{:=}v][j]$: the explanation is to be found in the equality inference $(u \quad v), (v \quad a[i{:=}v][j]) \quad _{=} u \quad a[i{:=}v][j]$ which contradicts the trail. Making progress towards the latest decision contributing to the conflict ($f(a[i{:=}v][j] \leftarrow 0$), we use Resolve (l19), and then UIPBackjump, to get back the search state Γ_7 spreading over lines 20 and 21. Theory LRA is now unable to pick a value for $f(a[i{:=}v][j])$: this is explained by inferring $f(u) \quad w$ from the current trail (l22), while the current assignments for w and $f(u)$ rather entail $f(u) \quad w$, triggering a conflict (l23). Applying Undo yields the search state Γ_9, spreading over lines 24, 25, and 26. Theory LRA is then unable to correct the assignment for $f(u)$: this is explained by the inferences $w \quad w-2$ from the trail, and $\quad w \quad w-2$; applying the former puts us in a position to apply the Fail rule.

8 Soundness

In this section, we focus on two special kinds of states that CDSAT can reach. Section 9 will show that one of these states will necessarily be reached in finitely many steps, but this section focuses on the consequences of reaching these states from an input problem: The first kind is the state unsat itself, and Section 8.1 shows that, if it is reached from an input problem, then this problem is unsatisfiable, a property known as *refutational soundness*. The second kind is that of *mute states* which, roughly speaking, are states that no theory module can extend; Section 8.3 shows the *model-soundness* property, namely that reaching a mute state does indeed imply that a T_∞^+-model of the input problem can be extracted from the state, assuming appropriate completeness conditions on the theory modules, which we introduced in Section 4.3 and whose purpose we illustrate in Section 8.2 by a motivating example.

8.1 Refutational soundness

Intuitively, the refutational soundness property comes from the fact that every propagation that was made came from a sound inference of one of the theories, or from a conflict which itself results from the analysis of prior propagations. By recursively analysing the history of how trails and conflicts were formed, we show that such trails and conflicts satisfy a property of "soundness" with respect to the union of the theories, formalised by the next definition. This property will entail that models of the input problem will be models of every level 0 assignments, so if a contradiction was found at level 0, then there can be no model of the input problem.

Definition 22 (Justified trail, justified conflict state) A non-decision vertex A of a trail Γ is *justified in* Γ (or simply *justified*, if Γ is clear) if, whenever $\mathsf{fv}(\mathsf{expl}_\Gamma(A), A) \quad V$,

1. every $T_\infty^+[V]$-model that view-endorses $\mathsf{expl}_\Gamma(A)$ view-endorses A, and,
2. should both A and $\mathsf{expl}_\Gamma(A)$ be Boolean assignments, if also every $T_\infty[V]$-model that endorses $\mathsf{expl}_\Gamma(A)$ endorses A.

Now let Γ be a trail and H be a subset of its non-decision roots.

The trail Γ is H*-justified* if all non-decision vertices A that are not in H are justified in Γ.

A conflict state $\Gamma; E; A$ is H*-justified* if the trail Γ is H-justified and, whenever $\mathsf{fv}(E) \quad V$,

1. there are no $T_\infty^+[V]$-models that view-endorse E, and,
2. should E be Boolean, if there are also no $T_\infty[V]$-models that endorse E.

※

To show the soundness property we also need to project a given T_∞^+-model as a T_i^+-model, a notion that is more precisely given by the following definition.

Definition 23 (Model projection) Given $\Sigma = (S, F)$ and $\Sigma' = (S', F')$ with $S' \quad S$ and $F' \quad F$, and given Σ-variables V and Σ'-variables V' with $\mathsf{fv}_\Sigma(V') \quad V$, the $\Sigma'[V']$-*projection* of a $\Sigma[V]$-interpretation M is the $\Sigma'[V']$-interpretation M' such that $s^{\mathcal{M}'} = s^{\mathcal{M}}$ for all $s \quad S'$, $f^{\mathcal{M}'} = f^{\mathcal{M}}$ for all $f \quad F'$, and $t^{\mathcal{M}'} = M(t)$ for all $t \quad V'$. ※

Lemma 9 (Soundness of inferences with respect to the union of theories)

Let $J \quad _{\mathcal{I}_k} L$ be an I_k-inference for some theory T_k with extension T_k^+ with $1 \leq k \leq n$, and let $\mathsf{fv}(J, L) \quad V$. Any $T_\infty^+[V]$-model that view-endorses J (resp. any $T_\infty[V]$-model endorsing J, should J be Boolean) endorses L. ※

Proof: Let Σ_k (resp. Σ_k^+) be the signature of theory T_k (resp. T_k^+), and let $V' = \mathsf{fv}_{\Sigma_k}(J, L)$. Note that $\mathsf{fv}(V') \quad \mathsf{fv}(J, L) \quad V$.

Assume M is a $T_\infty^+[V]$-model that view-endorses J. The $\Sigma_k^+[V']$-projection of M is a $\Sigma_k^+[V']$-interpretation M' that interprets any $\Sigma_k^+[V']$-term, and any Σ_k^+-sentence, the same way M does. So M' is a $T_k^+[V']$-model view-endorsing J. By definition of T_k-inferences, M' must endorse L, and therefore M must also endorse L.

In the case where J is Boolean, assume M is a $T_\infty[V]$-model that endorses J. Again, the $\Sigma_k[V']$-projection of M is a $\Sigma_k[V']$-interpretation M' that interprets any $\Sigma_k[V']$-term, and any Σ_k-sentence, the same way M does. So M' is a $T_k[V']$-model endorsing J. Then by conservativity of T_k^+ with respect to T_k, every $T_k[V']$-model endorsing J must endorse L. In particular M'. So again M endorses L as well. □

Now we prove that the property is preserved by every step of a CDSAT derivation:

Lemma 10 (Preservation of justifiedness)

1. If $\Gamma; E; A$ is H-justified and $\Gamma; E; A \quad = \quad ^* \Gamma'$, then Γ' is H-justified.
2. If Γ is H-justified and $\Gamma \dashrightarrow_{\mathcal{B}} \Gamma'$, then Γ' is H-justified.

※

Proof:

1. By induction on the derivation of $\Gamma; E; A \quad = \quad ^* \Gamma'$ and case analysis on the first rule used:
   - Resolve The trail does not change, so it is still H-justified. Moreover, assume there is a $T_\infty^+[V]$-model that view-endorses $E \quad \mathsf{expl}_\Gamma(L)$ (resp. $T_\infty[V]$-model endorsing $E \quad \mathsf{expl}_\Gamma(L)$, should $E \quad \mathsf{expl}_\Gamma(L)$ be Boolean). Since the trail is H-justified and $L \,/\, H$ (as H only contains non-decision roots), that model would also be a $T_\infty^+[V]$-model (resp. $T_\infty[V]$-model) of E, L.
   - UIPBackjump Since H only contains non-decision roots, no vertex of H is affected by the removal of vertices. All remaining vertices are still justified, as their predecessors have not changed. Let $E \quad \overline{L}$ be the propagation arising from conflict E, L: Any $T_\infty^+[V]$-model that view-endorses E (resp. any $T_\infty[V]$-model endorsing E, should E be Boolean) cannot be endorsing L as well, so it must endorse $\overline{L}$.
   - SemSplit Again, since H only contains non-decision roots, no vertex of H is affected by the removal of vertices. All remaining vertices are still justified, as their predecessors have not changed. The only new vertex being a decision, the resulting trail is H-justified.
   - Undo Again, since H only contains non-decision roots, no vertex of H is affected by the removal of vertices. All remaining vertices are still justified, as their predecessors have not changed. There are no new vertices, so the resulting trail is H-justified.
2. By case analysis on the rule:

Decide The only change in the trail is the addition of a decision, so the trail remains H-justified.

Propagate There is only one new vertex: L, and by Lemma 9, it is justified.

Conflict Again by Lemma 9, any $T_\infty^+[V]$-model that view-endorses J (resp. $T_\infty[V]$-model endorsing J, should J be Boolean) must endorse L, so clearly there are no $T_\infty^+[V]$-model that view-endorse $J, \overline{L}$ (resp. $T_\infty[V]$-model endorsing $J, \overline{L}$). Therefore the resulting conflict state is H-justified, and by the point 1, the search state resulting from conflict analysis is H-justified.

Fail There is nothing to prove.

□

Lemma 11 (Satisfiability of level 0 assignments)

Let Γ be an H-justified trail and let $\mathsf{fv}(\Gamma) \quad V$.

- Any $T_\infty^+[V]$-model that view-endorses H view-endorses L_0^Γ.
- If H is Boolean, then L_0^Γ is Boolean and any $T_\infty[V]$-model endorsing H endorses L_0^Γ.

※

Proof: In L_0^Γ, there are no decision assignments, so every vertex that is not in H is justified. By induction on the DAG structure of L_0^Γ (i.e. propagating truth values along the paths), the model of H view-endorses every vertex of that level (resp. endorses every vertex of that level, should H be Boolean). □

Theorem 12 (Refutational soundness) Let H be an input problem. If the calculus reaches state unsat, then there is no $T_\infty^+[V]$-model with $\mathsf{fv}(H) \quad V$ that view-endorses H. If H is Boolean, then there is no $T_\infty[V]$-model with $\mathsf{fv}(H) \quad V$ that endorses H. ※

Proof: The initial state is trivially H-justified. By Lemma 10, every trail in the derivation of unsat is H-justified, in particular the last trail Γ preceding unsat. Assume there is a $T_\infty^+[V]$-model M with $\mathsf{fv}(H) \quad V$ that view-endorses H (resp. a $T_\infty[V]$-model M with $\mathsf{fv}(H) \quad V$ that endorses H, should H be Boolean). By picking random domain elements to interpret the variables in $\mathsf{fv}(L_0^\Gamma) \setminus \mathsf{fv}(H)$, we get a model M' that still view-endorses (resp. endorses) H. So by Lemma 11 M' view-endorses (resp. endorses) L_0^Γ, which includes J and $\overline{L}$ used in the transition to unsat. By as $J \quad_{\mathcal{I}_k} L$ for one of the theory modules I_k, Lemma 9 entails that M' should also endorse L. □

8.2 Example and reference theory

We now turn to model-soundness. As already mentioned in Section 4.3, a trail is not a datastructure that can be used by the theory modules for $T_1, \ldots, T_n$ to exchange *complete* information about the shared sorts' cardinalities. Here is an example:

Example 6 Given two sorts s and s', let $T_{s,s'}$ be the theory on signature

$$\Sigma_{s,s'} = (\{\mathsf{prop}, s, s'\}, \quad {}_{\{\mathsf{prop},s,s'\}} \quad \{f_{s,s'} : s \to s'\})$$

whose models are those satisfying the property that, should $|s| \geq 2$, $f_{s,s'}$ is injective but not surjective (this theory can be given by one axiom of multi-sorted first-order logic).

Now let s_1, s_2, s_3 be three sorts, and consider the combination T_∞ of the four disjoint theories $T_{s_1,s_2}, T_{s_2,s_3}, T_{s_3,s_1}$, and T_4, the theory on signature $\Sigma_4 = (\{\mathsf{prop}, s_3\}, \quad {}_{\{\mathsf{prop},s_3\}})$ whose models are those where $|s_3| \leq 1000$ (also axiomatisable by one axiom of multi-sorted first-order logic).

There is exactly one model of $\mathcal{T}_\infty$, namely that where each of the sorts s_1, s_2, s_3 is interpreted as a singleton, and $f_{s_1,s_2}, f_{s_2,s_3}, f_{s_3,s_1}$ are interpreted as the only functions that exist between the three sorts.

Now the input problem $x \quad_{s_1} y$ is unsatisfiable in $\mathcal{T}_\infty$. Note that removing any of the four theories makes it satisfiable. The question is now to understand how CDSAT will produce the unsat answer. None of the four theories can, on its own, detect the problem; none of them even knows about all three sorts (all of which are involved in making the problem unsat). None of the function symbols appear in the input problem, so it is hard to see why the theory modules would start enumerating the input/output pairs (i.e. the graphs) of the functions interpreting the symbols, especially given that there could be infinitely many (e.g. if theory $\mathcal{T}_4$ was not there). ※

We deal with this issue in two stages:

First, we rule out the kind of example above by requiring that the cardinality constraints aggregated from the different theories be ***subsumed*** by one of the theories $\mathcal{T}_1, \ldots, \mathcal{T}_n$, say $\mathcal{T}_k$. This is typically not the case in the above example. Technically, we require that the signature of theory $\mathcal{T}_k$ is of the form (S_∞, F_i) (i.e. it knows about all sorts), and the module for every theory other than $\mathcal{T}_k$ is $\mathcal{T}_k^+$-complete (as well as requiring that the $\mathcal{T}_k$-module is complete). With that requirement, the next section shows model-soundness of CDSAT.

Second, the case of theory combinations $\mathcal{T}_\infty = \mathcal{T}_1 \quad \ldots \quad \mathcal{T}_n$ where none of the involved theories subsumes the aggregated cardinality constraints (as in the example above) can be reduced to the case where there is one, if one can find a ***extra*** theory $\mathcal{T}_0$, with extension $\mathcal{T}_0^+$,

- that subsumes them, meaning that the module for every theory $\mathcal{T}_1, \ldots, \mathcal{T}_n$ is $\mathcal{T}_0^+$-complete;
- such that $\mathcal{T}_0^+ \quad \mathcal{T}_\infty^+$ conservatively extends $\mathcal{T}_\infty^+$, meaning that every set of formulæ that is $(\mathcal{T}_0^+ \quad \mathcal{T}_\infty^+)$-unsat is $\mathcal{T}_\infty^+$-unsat;
- together with a complete theory module for it.

Indeed if such a theory can be found, CDSAT can be run with the theory modules for $\mathcal{T}_0, \mathcal{T}_1, \ldots, \mathcal{T}_n$. In case it concludes unsatisfiability of an input problem, conservativity implies its unsatisfiability in $\mathcal{T}_1 \quad \ldots \quad \mathcal{T}_n$.

In the case of the above example, $\mathcal{T}_0$ can be taken to be the theory on signature $\Sigma_0 = (\{\mathsf{prop}, s_1, s_2, s_3\}, \quad_{\{\mathsf{prop}, s_1, s_2, s_3\}})$ that imposes that $|s_1| = |s_2| = |s_3| = 1$. We can take $\mathcal{T}_0^+$ to be the trivial extension of $\mathcal{T}_0$, and take the $\mathcal{T}_0$-module $\mathcal{I}_0$ to comprise the inferences $t \quad_s u \quad_{\mathcal{I}_0}$ for s in $\{s_1, s_2, s_3\}$. This would be the module that detects the unsatisfiability of the input problem $x \quad_{s_1} y$.

In the case where $\mathcal{T}_1, \ldots, \mathcal{T}_k$ are all stably infinite, $\mathcal{T}_0$ can be the theory of structures where all sorts (but prop) are countably infinite, and the $\mathcal{T}_0$-module with no inference rules is complete for this (since an assignment is only going to mention finitely many terms and values).

8.3 Model-soundness

In this section, theory T_1 plays a particular role, as motivated by the previous section. In particular, we assume that its signature Σ_1 is of the form (S_∞, F_1) (i.e. it knows all sorts).

The core theorem of this section, Theorem 13 below, glues together the different models that the different theories may have for a given assignment. For this we need the notion of *shared terms*: Equalities or disequalities between shared terms will have to be agreed upon by the theories in order to construct a model.

Definition 24 (Shared terms) Given a finite set X of terms, the set $V_{\mathsf{shared}}(X)$ of *shared terms* (for signatures $\Sigma_1, \dots, \Sigma_n$) is the smallest superset Y of X closed under the following rules

$$\frac{u, u' \quad Y \quad t \quad \mathsf{fv}_{\Sigma_k}(u) \quad t \quad \mathsf{fv}_{\Sigma_j}(u') \quad i = j}{t \quad Y} \qquad \frac{u \quad Y \quad t \quad \mathsf{fv}_{\Sigma_k}(u) \quad t \;/\; V_\infty}{t \quad Y}$$ ※

Note that, since X is finite, $V_{\mathsf{shared}}(X)$ is also finite (as it consists of subterms of terms in X). We write $V^s_{\mathsf{shared}}(X)$ for the (finite) set of terms in X of sort s, and we identify $V_{\mathsf{shared}}(X)$ with the family $(V^s_{\mathsf{shared}}(X))_{s \in S_\infty}$. We also use the notations $V^s_{\mathsf{shared}}(H)$ and $V_{\mathsf{shared}}(H)$ when H is an assignment, viewing H as the finite set of terms being assigned values, and even $V^s_{\mathsf{shared}}(\Gamma)$ and $V_{\mathsf{shared}}(\Gamma)$ when Γ is a trail, viewing it as an assignment.

Definition 25 (Model-describing assignment) An assignment H is *model-describing* if $H_{\mathcal{T}_1}$ is consistent with T_1^+ and if, for $2 \leq k \leq n$, $H_{\mathcal{T}_k}$ is T_1^+-compatible with T_k^+ sharing $V_{\mathsf{shared}}(H)$. ※

We now prove the main theorem about the common model construction.

Theorem 13 (Glueing models) If H is a model-describing assignment, there exists a $T_\infty^+[\mathsf{fv}(H)]$-model view-endorsing H. ※

We assume H is a model-describing assignment and split the proof as follows.

Lemma 14 We always have $\mathsf{fv}(H) \quad \bigcup_{k=1}^n \mathsf{fv}_{\Sigma_k}(V_{\mathsf{shared}}(H))$. ※

Proof: Let $x \quad \mathsf{fv}(H)$ and let Y be the (non-empty) subset of $\bigcup_{k=1}^n \mathsf{fv}_{\Sigma_k}(V_{\mathsf{shared}}(H))$ of terms having x as a subterm. Let t be a term in Y such that the distance between t's root position and the position of the leftmost occurrence of x is minimal. If t is not x itself, then its root is a symbol of some Σ_k, and there is $u \quad \mathsf{fv}_{\Sigma_k}(t)$ having the occurrence of x as a subterm. By the second closure rule, $t \quad V_{\mathsf{shared}}(H)$ and therefore $u \quad Y$, with a smaller distance between its root position and the position of the leftmost occurrence of x, which is a contradiction. □

Remark 15 Whenever $1 \leq k \leq n$,

1. $\mathsf{fv}_{\Sigma_k}(H_{\mathcal{T}_k}) = \mathsf{fv}_{\Sigma_k}(H) \quad \mathsf{fv}_{\Sigma_k}(V_{\mathsf{shared}}(H))$, so $\mathsf{fv}_{\Sigma_k}(H_{\mathcal{T}_k} \quad V_{\mathsf{shared}}(H)) = \mathsf{fv}_{\Sigma_k}(V_{\mathsf{shared}}(H))$.
2. view-endorsing $H_{\mathcal{T}_k}$ is the same as view-endorsing H.

※

Definition 26 (Construction of bijections) Since $H_{\mathcal{T}_1}$ is consistent with T_1^+, there exists a $T_1^+[\mathsf{fv}_{\Sigma_1}(H)]$-model that view-endorses H. We extend it into a $T_1^+[\mathsf{fv}_{\Sigma_1}(V_{\mathsf{shared}}(H))]$-model M_1 by picking arbitrary values for terms in $\mathsf{fv}_{\Sigma_1}(V_{\mathsf{shared}}(H)) \setminus \mathsf{fv}_{\Sigma_1}(H)$.

Now whenever $2 \leq k \leq n$, the fact that $H_{\mathcal{T}_k}$ is T_1^+-compatible with T_k^+ implies that there exists a $T_k^+[\mathsf{fv}_{\Sigma_k}(V_{\mathsf{shared}}(H))]$ model M_k that view-endorses H and such that for all $s \quad S_k$,

- $\left|s^{\mathcal{M}_k}\right| = \left|s^{\mathcal{M}_1}\right|$, and
- for all t and t' in $\mathsf{V}^s_{\mathsf{shared}}(H)$, $\mathsf{M}_k(t) = \mathsf{M}_k(t')$ if and only if $\mathsf{M}_1(t) = \mathsf{M}_1(t')$.

We extract from this a bijection ϕ^s_k from $s^{\mathcal{M}_k}$ to $s^{\mathcal{M}_1}$ such that for all $t \quad \mathsf{V}^s_{\mathsf{shared}}(H)$ we have $\phi^s_k(\mathsf{M}_k(t)) = \mathsf{M}_1(t)$. For $k = 1$ and all $s \quad S_\infty$, we take ϕ^s_1 to be the identity from $s^{\mathcal{M}_1}$ to $s^{\mathcal{M}_1}$.

※

We now build a $\mathsf{T}^+_\infty[\mathsf{fv}(H)]$-model M.

Definition 27 (Construction of the model)

- For any sort s, we take $s^{\mathcal{M}} = s^{\mathcal{M}_1}$, and equality symbols are necessarily interpreted as equality of domain elements.
- For any other symbol $f : (s_1 \times \cdots \times s_m) \to s$ in signature Σ_k (the fact that theories are disjoint makes k unique), we take $f^{\mathcal{M}}(a_1, \ldots, a_m) = \phi^s_k(f^{\mathcal{M}_k}((\phi^{s_1}_k)^{-1}(a_1), \ldots, (\phi^{s_m}_k)^{-1}(a_m)))$.
- For a variable $x \quad \mathsf{fv}^s(H)$ with $x \quad \mathsf{V}_{\mathsf{shared}}(H)$: we take $x^{\mathcal{M}} = x^{\mathcal{M}_1}$.
- For a variable $x \quad \mathsf{fv}^s(H)$ with $x \quad \mathsf{V}_{\mathsf{shared}}(H)$: Lemma 14 gives a k with $1 \leq k \leq n$ such that $x \quad \mathsf{fv}_{\Sigma_k}(\mathsf{V}_{\mathsf{shared}}(H))$. Moreover, this k is unique, otherwise the first closure rule of $\mathsf{V}_{\mathsf{shared}}(H)$ would entail $x \quad \mathsf{V}_{\mathsf{shared}}(H)$. So we take $x^{\mathcal{M}} = \phi^s_k(x^{\mathcal{M}_k})$.
- For a non-Boolean T^+_k-value $\mathfrak{c}$ we take $\mathfrak{c}^{\mathcal{M}} = \phi^s_k(\mathfrak{c}^{\mathcal{M}_k})$, and of course $\mathsf{true}^{\mathcal{M}} = \mathsf{true}$ and $\mathsf{false}^{\mathcal{M}} = \mathsf{false}$.

※

Lemma 16 (Coincidence of interpretations)

1. Assume $1 \leq k \leq n$, and let t be a term of sort $s \quad S_k$ such that $\mathsf{fv}_{\Sigma_k}(t) \quad \mathsf{fv}_{\Sigma_k}(H)$.
   Assume for all $s' \quad S_k$ and all $u \quad \mathsf{fv}^{s'}_{\Sigma_k}(t) \cap \mathsf{V}^{s'}_{\mathsf{shared}}(H)$ we have $\mathsf{M}(u) = \phi^{s'}_k(\mathsf{M}_k(u))$.
   then we have $\mathsf{M}(t) = \phi^s_k(\mathsf{M}_k(t))$.
2. For all $t \quad \mathsf{V}_{\mathsf{shared}}(H)$ we have $\mathsf{M}(t) = \mathsf{M}_1(t)$.
3. Assume $1 \leq k \leq n$ and let t be a $\Sigma^+_k[\mathsf{fv}_{\Sigma_k}(\mathsf{V}_{\mathsf{shared}}(H))]$-term of sort $s \quad S_k$.
   We have $\mathsf{M}(t) = \phi^s_k(\mathsf{M}_k(t))$.

※

Proof:

1. By induction on t:
   - If $t \quad \mathsf{fv}_{\Sigma_k}(t) \cap \mathsf{V}_{\mathsf{shared}}(H)$ then we apply the hypothesis.
   - If $t \quad \mathsf{fv}_{\Sigma_k}(t) \backslash \mathsf{V}_{\mathsf{shared}}(H)$ then $t \quad \mathsf{fv}(H)$ and by definition we have $\mathsf{M}(t) = \phi^s_k(\mathsf{M}_k(t))$.
   - If t is of the form $f(t_1, \ldots, t_m)$ with $f : (s_1 \times \cdots \times s_m) \to s$ in signature Σ_k, then $\mathsf{M}(f(t_1, \ldots, t_m)) = \phi^s_k(f^{\mathcal{M}_k}((\phi^{s_1}_k)^{-1}(\mathsf{M}(t_1)), \ldots, (\phi^{s_m}_k)^{-1}(\mathsf{M}(t_m))))$, and by applying the induction hypothesis on each of $t_1, \ldots, t_m$, it is equal to $\phi^s_k(f^{\mathcal{M}_k}(\mathsf{M}_k(t_1)), \ldots, \mathsf{M}_k(t_1)) = \phi^s_k(\mathsf{M}_k(t))$.
2. By induction on t:
   - If $t \quad \mathsf{V}_\infty$ then by definition we have $\mathsf{M}(t) = \mathsf{M}_1(t)$.
   - If t is of the form $f(t_1, \ldots, t_m)$ with $f : (s_1 \times \cdots \times s_m) \to s$ in signature Σ_k for some $1 \leq k \leq n$: Then for all $s' \quad S_k$ and all $u \quad \mathsf{fv}^{s'}_{\Sigma_k}(t) \cap \mathsf{V}^{s'}_{\mathsf{shared}}(H)$, we have $\mathsf{M}(u) = \phi^{s'}_k(\mathsf{M}_k(u))$, since, on the one hand, u is strictly smaller than t and therefore the induction hypothesis

provides $\mathsf{M}(u) = \mathsf{M}_1(u)$, and on the other hand, the fact that $u \quad \mathsf{V}^{s'}_{\mathsf{shared}}(H)$ entails $\mathsf{M}_1(u) = \phi^{s'}_k(\mathsf{M}_k(u))$. We can therefore apply Point 1 and get $\mathsf{M}(t) = \phi^s_k(\mathsf{M}_k(t))$. Since $t \quad \mathsf{V}^s_{\mathsf{shared}}(H)$ we have $\mathsf{M}_1(t) = \phi^s_k(\mathsf{M}_k(t))$ and therefore $\mathsf{M}(t) = \mathsf{M}_1(t)$.

3. By induction on t:
   - If $t \quad \mathsf{V}_{\mathsf{shared}}(H)$, then $\mathsf{M}(t) = \mathsf{M}_1(t) = \phi^s_k(\mathsf{M}_k(t))$, using Point 2.
   - If $t \quad \mathsf{fv}_{\Sigma_k}(\mathsf{V}_{\mathsf{shared}}(H))$ with $t \quad \mathsf{V}_{\mathsf{shared}}(H)$, then we must have $t \quad \mathsf{fv}(\mathsf{V}_{\mathsf{shared}}(H))$ and, by construction, $\mathsf{M}(t) = \phi^s_k(\mathsf{M}_k(t))$.
   - If t is a non-Boolean $\mathcal{T}^+_k$-value, then by construction we have $\mathsf{M}(t) = \phi^s_k(\mathsf{M}_k(t))$.
   - If t is of the form $f(t_1, \dots, t_m)$ with $f : (s_1 \times \cdots \times s_m) \to s$ in signature Σ_k, then $\mathsf{M}(f(t_1, \dots, t_m)) = \phi^s_k(f^{\mathcal{M}_k}((\phi^{s_1}_k)^{-1}(\mathsf{M}(t_1)), \dots, (\phi^{s_m}_k)^{-1}(\mathsf{M}(t_m))))$, and by applying the induction hypothesis on each of $t_1, \dots, t_m$, it is equal to $\phi^s_k(f^{\mathcal{M}_k}(\mathsf{M}_k(t_1)), \dots, \mathsf{M}_k(t_1)) = \phi^s_k(\mathsf{M}_k(t))$.

□

We now finish the proof of Theorem 13.

Proof: From the previous Lemma, and whenever $1 \leq k \leq n$, M is a $\mathcal{T}^+_k$-model (forgetting the interpretation of all sorts and symbols that are not in Σ^+_k). We are left to show that if $t \leftarrow \mathfrak{c}$ is in $H_{\mathcal{T}_\infty}$, then $\mathsf{M}(t) = \mathfrak{c}^{\mathcal{M}}$.

If the single assignment $t \leftarrow \mathfrak{c}$ is in H, then $\mathfrak{c}$ is a $\mathcal{T}^+_k$-value for one of the theories $\mathcal{T}^+_k$. If it is not in H, then t is of the form $t_1 \;_{s} t_2$ and for any theory $\mathcal{T}^+_k$ with $s \quad S_k$ we have $t \leftarrow \mathfrak{c}$ in $H_{\mathcal{T}_k}$. In both cases, $\mathsf{M}(t) = \mathsf{M}_k(t) = \mathfrak{c}^{\mathcal{M}_k} = \mathfrak{c}^{\mathcal{M}}$, whether $\mathfrak{c}$ is a Boolean value or a non-Boolean $\mathcal{T}^+_k$-value. □

The second task is to formally define the *mute* kind of states, which is a syntactic notion that will entail, together with completeness assumptions about the theory modules, that such states give a model-describing assignment.

Definition 28 (Mute state) A state is *mute* if it is a search state whose underlying assignment H is such that whenever $1 \leq k \leq n$, $H_{\mathcal{T}_k}$ is a plausible $\mathcal{T}^+_k$-assignment that $\mathcal{I}_k$ cannot extend. ※

Lemma 17 (Mute states are model-describing)

Assume that for $2 \leq k \leq n$, module $\mathcal{I}_k$ is $\mathcal{T}^+_1$-complete, and assume module $\mathcal{I}_1$ is complete.

If a state is mute, its underlying assignment is model-describing. ※

Proof: From the definitions of completeness and $\mathcal{T}^+_1$-completeness. □

Corollary 18 (Model-soundness)

Assume that for $2 \leq k \leq n$, module $\mathcal{I}_k$ is $\mathcal{T}^+_1$-complete, and assume module $\mathcal{I}_1$ is complete.

If CDSAT reaches a mute state Γ, there exists a $\mathcal{T}^+_\infty$-model that view-endorses Γ, and therefore view-endorses the input problem. ※

Proof: Notice that the input problem remains present in every reached state. So if CDSAT reaches a mute state, its underlying assignment H contains the input problem and from the previous lemma it is model describing. Hence by Theorem 13, there exists a $\mathcal{T}^+_\infty$-model that view-endorses H, and therefore view-endorses the input problem. □

Note that CDSAT can reach a state whose underlying assignment is model-describing long before it reaches a mute state. In an implementation, it may therefore be useful to let theory modules notify the main algorithm when the underlying assignment of current trail becomes $T_1{}^{+}$-compatible with their theory.

9 Termination and progress

In this section we prove termination and progress of CDSAT: termination ensures that, starting from any input problem X, applying B-transitions always reaches an irreducible state, in finitely many steps and regardless of the strategy used to apply them; progress ensures that an irreducible state has an interesting shape, namely that it is either unsat or a model-describing state. Termination and progress do require, however, that B be appropriate for the input problem X, namely that it be a *global basis* for it.

Definition 29 (Global basis) Let X be a set of terms.

A closed set B is a *global basis for X with respect to theory modules $I_1, \ldots, I_n$* if

1. Original terms: X B
2. Finiteness: B is finite;
3. Stability: for all $1 \leq k \leq n$, $\mathsf{basis}_k(B)$ B

※

Section 9.1 shows how finiteness ensures termination, and how stability ensures progress, i.e. the fact that the B-transition system does not get stuck on a state that is not unsat and that some theory modules can still extend. Section 9.3 gives a condition on the local bases of individual theory modules that is sufficient to guarantee the existence of a global basis.

9.1 Proofs of termination and progress given a global basis

We now assume that B is a finite set of terms, and prove termination of B-transitions. For this, we encode each trail as a $3|B|$-tuple of integers, and show that every B-transition decreases this encoding according to the lexicographic order on $3|B|$-tuples of integers, which we denote $>_{\mathsf{lex}}$.

Definition 30 (Encoding of trails as tuples) Let Γ be a trail, with $A_1, \ldots, A_m$ being its decisions unambiguously enumerated from smallest to greatest. For i in $1, \ldots, m$, let so_i^Γ be 0 if A_i is a Boolean assignment, and be 1 if not. For i in $0, \ldots, m$, let w_i^Γ denote $|B| - |L_i^\Gamma|$ (the number of terms in B that do not have a level i assignment in Γ), and let bad_i^Γ denote the number of decisions in Γ, of some level greater than i, that are first-order T-assignments, for some theory T with module I, and that violate $L_{\leq i}^\Gamma$ in one I-step. We encode Γ as the $3|B|$-tuple

$$\mathsf{asTuple}(\Gamma) = (w_0^\Gamma, \mathsf{bad}_0^\Gamma, \mathsf{so}_1^\Gamma, w_1^\Gamma, \mathsf{bad}_1^\Gamma, \ldots, \mathsf{so}_m^\Gamma, w_m^\Gamma, \mathsf{bad}_m^\Gamma, 2, \ldots, 2).$$

※

We now prove that conflict analysis decreases the tuples encoding trails, compared in lexicographic order:

Lemma 19 (Conflict analysis decreases the measure)

If $\Gamma; E; A \;=\; {}^{*}\,\Gamma'$ then $\mathsf{asTuple}(\Gamma) >_{\mathsf{lex}} \mathsf{asTuple}(\Gamma')$. ※

Proof: By induction on the derivation of $\Gamma; E; A \;=\; {}^{*}\,\Gamma'$, doing a case analysis on the first rule used:

Resolve $\Gamma; E, L; A \;=\; \quad \Gamma; E \quad \mathsf{expl}_\Gamma(L); A$

There is nothing to prove as Resolve does not change the trail.

UIPBackjump $\Gamma; E, L; A \;=\; \quad (\Gamma \setminus D), (E \quad \overline{L})$

Let i be the level of A. Clearly, the side condition of rule UIPBackjump entails that $\overline{L}$ is in $\mathsf{L}_j^{(\Gamma\setminus D),(E\vdash\overline{L})}$ for some j in $0, \ldots, i-1$. For all j' in $1, \ldots, j$, $\mathsf{so}_{j'}^{(\Gamma\setminus D),(E\vdash\overline{L})} = \mathsf{so}_{j'}^{\Gamma}$. And as we only remove decisions (those in D) that are greater than j, for all j' in $0, \ldots, j-1$, $\mathsf{bad}_{j'}^{(\Gamma\setminus D),(E\vdash\overline{L})} \leq \mathsf{bad}_{j'}^{\Gamma}$. For all j' in $0, \ldots, j-1$, $w_{j'}^{(\Gamma\setminus D),(E\vdash\overline{L})} = w_{j'}^{\Gamma}$. Finally, notice that $w_j^{(\Gamma\setminus D),(E\vdash\overline{L})} < w_j^{\Gamma}$, because $\overline{L}$ has been added to level j.

SemSplit $\Gamma; E, L; A \;=\; \quad \Gamma \setminus A^{\checkmark}\!\frown\!\overline{L}$

Let i be the level of A. For all j in $1, \ldots, i-1$, $\mathsf{so}_j^{\Gamma\setminus A^{\checkmark}\frown\overline{L}} = \mathsf{so}_j^{\Gamma}$. For all j in $0, \ldots, i-1$, $w_j^{\Gamma\setminus A^{\checkmark}\frown\overline{L}} = w_j^{\Gamma}$. And as we only replace decision A by a Boolean decision, for all j in $0, \ldots, i-1$, $\mathsf{bad}_j^{\Gamma\setminus A^{\checkmark}\frown\overline{L}} \leq \mathsf{bad}_j^{\Gamma}$. Now $\mathsf{so}_i^{\Gamma\setminus A^{\checkmark}\frown\overline{L}} = 0$ while $\mathsf{so}_i^{\Gamma} = 1$.

Undo $\Gamma; E, A; A \;=\; \quad \Gamma \setminus A$

Let i be the level of A. Clearly, the side condition of rule Undo entails that A violates L_j^{Γ} in one step, for some j in $0, \ldots, i-1$. Taking the smallest of such j, we have: For all j' in $1, \ldots, j$, $\mathsf{so}_{j'}^{\Gamma\setminus A} = \mathsf{so}_{j'}^{\Gamma}$. For all j' in $0, \ldots, j$, $w_{j'}^{\Gamma\setminus A} = w_{j'}^{\Gamma}$. And as we only remove decision A and j is the smallest level that A violates in one step, for all j' in $0, \ldots, j-1$, $\mathsf{bad}_{j'}^{\Gamma\setminus A} = \mathsf{bad}_{j'}^{\Gamma}$. Finally, notice that $\mathsf{bad}_j^{\Gamma\setminus A} < \mathsf{bad}_j^{\Gamma}$.

□

Theorem 20 (Search rules decrease the measure)

If $\Gamma \dashrightarrow_{\mathcal{B}} \Gamma'$ then $\mathsf{asTuple}(\Gamma) >_{\mathsf{lex}} \mathsf{asTuple}(\Gamma')$. ※

Proof:

Decide $\Gamma \dashrightarrow \quad \Gamma, A$

We have $\mathsf{asTuple}(\Gamma)$ of the form $(\ldots, \mathsf{bad}_m^{\Gamma}, 2, \ldots, 2)$, and $\mathsf{asTuple}(\Gamma, A)$ of the form

$$(\ldots, \mathsf{bad}_m^{\Gamma,A}, \mathsf{so}_{m+1}^{\Gamma,A}, w_{m+1}^{\Gamma,A}, \mathsf{bad}_{m+1}^{\Gamma,A}, 2 \ldots, 2).$$

For all j in $1, \ldots, m$, $\mathsf{so}_j^{\Gamma,A} = \mathsf{so}_j^{\Gamma}$. For all j in $0, \ldots, m$, $w_j^{\Gamma,A} = w_j^{\Gamma}$. And as the side-condition of rule Decide forbids A to violate anything in Γ in one step, for all j in $0, \ldots, m$, $\mathsf{bad}_j^{\Gamma,A} = \mathsf{bad}_j^{\Gamma}$. We conclude with $\mathsf{so}_{m+1}^{\Gamma,A} < 2$.

Propagate $\Gamma \dashrightarrow \quad \Gamma, (J \quad L)$

Let i be the level of L. For all j in $1, \ldots, i$, $\mathsf{so}_j^{\Gamma,(J\vdash L)} = \mathsf{so}_j^{\Gamma}$. For all j in $0, \ldots, i-1$, $w_j^{\Gamma,(J\vdash L)} = w_j^{\Gamma}$, and $\mathsf{bad}_j^{\Gamma,(J\vdash L)} = \mathsf{bad}_j^{\Gamma}$. We conclude with $w_i^{\Gamma,(J\vdash L)} < w_i^{\Gamma}$.

Conflict $\Gamma \dashrightarrow \quad \Gamma'$, given that $\Gamma; J, \overline{L}; A \;=\; {}^{*}\,\Gamma'$.

By the previous lemma, $\mathsf{asTuple}(\Gamma') >_{\mathsf{lex}} \mathsf{asTuple}(\Gamma)$.

Fail There is nothing to prove, the transition reaches an irreducible form.

□

Corollary 21 (Termination) B-transitions terminate. ※

We now prove progress, assuming B is a global basis for an input problem X.

Remark 22 If the trail Γ of a state is such that $\Gamma \quad B$, then after one B-transition, the state has a trail $\Gamma' \quad B$. ※

Proof: Only rules Decide and Propagate may introduce a new formula; In the case of Decide, it is either a relevant term or a relevant equality of $\Gamma \quad B$, so it is still in B. In the case of Propagate, the side-condition enforces that this new formula be in B. □

Therefore, starting from an input problem X, any term ever appearing in the trail of a state reached by B-transitions is a term in B.

Theorem 23 (Progress) From every conflict state a transition rule applies. If a search state Γ with $\Gamma \quad B$ is not mute, then a B-transition rule applies. ※

Proof: We start with the case of a conflict state $\Gamma; E; A$: If A is boolean, then Resolve can be applied unless all vertices of E are decisions, one of which is A, and in that case we can apply UIPBackjump. If A is not boolean, assume that Resolve cannot be applied: then E only contains decisions (which may or may not include A) and vertices towards which A has an edge; if A has an edge to at least one vertex in E, then either UIPBackjump or SemSplit applies (depending on whether A is in E), and if not, A must be in E and Undo applies.

Now if a search state Γ is not mute, then for at least one i among $1, \ldots, n$, either $\Gamma_{\mathcal{T}_i}$ is not plausible or it is and I_i can extend it.

If $\Gamma_{\mathcal{T}_i}$ is not plausible, then it is a particular case of the situation where $\Gamma_{\mathcal{T}_i}$ is not included in Γ. This means there are two T_j^{+}-assignments $t_1 \leftarrow \mathfrak{c}_1$ and $t_2 \leftarrow \mathfrak{c}_2$ in Γ with an inferrable equality assignment $(t_1 \quad_s t_2) \leftarrow \mathfrak{b}$ that is not in Γ. If $\overline{(t_1 \quad_s t_2) \leftarrow \mathfrak{b}}$ is in Γ we can apply Conflict or Fail, and if not we can apply Propagate.

We now assume that $\Gamma_{\mathcal{T}_i}$ is a plausible T_i^{+}-assignment included in Γ, and that I_i can extend it. If it can extend $\Gamma_{\mathcal{T}_i}$ with an inference $J \quad_{\mathcal{I}_i} L$, then $J \quad \Gamma_{\mathcal{T}_i} \quad \Gamma$ and L is a Boolean assignment for a formula in $\mathsf{basis}_i(\Gamma_{\mathcal{T}_i}) \quad \mathsf{basis}_i(\Gamma) \quad \mathsf{basis}_i(B) \quad B$ (by monotonicity of basis_i and stability of B). Again, if $\overline{L}$ is in Γ we can apply Conflict or Fail, and if not we can apply Propagate. If it can extend $\Gamma_{\mathcal{T}_i}$ with a T_i^{+}-assignment $t \leftarrow \mathfrak{c}$ that is acceptable for $\Gamma_{\mathcal{T}_i}$ and I_i, where t is a T_i^{+}-relevant term of $\Gamma_{\mathcal{T}_i}$, then rule Decide can be applied. □

Corollary 24 (Normalisation) Given an input problem, CDSAT always reaches a state that is either unsat or a mute state. If the T_1-module is complete and every other module is T_1^{+}-complete, then CDSAT always reaches a state that is either unsat or a model-describing state. ※

9.2 Completeness results

In this section we assume that the T_1-module is complete and every other module is T_1^{+}-complete.

Corollary 25 (Refutational completeness) If there is no $T_\infty^{+}[V]$-model that view-endorses the input problem, then the calculus reaches state unsat. ※

Proof: Since there are no $T^+_\infty[V]$-model that view-endorse the input problem, Corollary 18 entails that the calculus cannot reach a model-describing state. By Lemma 17 and our completeness assumptions it cannot reach a mute state. Therefore Corollary 24 entails that state unsat is reached. □

Corollary 26 (Model-completeness) If there is a $T^+_\infty[V]$-model that view-endorses the input problem, then the calculus reaches a model-describing state. If the input problem is Boolean and there is a $T_\infty[V]$-model endorsing it, then again the calculus reaches a model-describing state. ※

Proof: Since there is a $T^+_\infty[V]$-model that view-endorses the input problem (resp. a $T_\infty[V]$-model endorsing the input problem, should the latter be Boolean), Theorem 12 entails that the calculus cannot reach state unsat. Therefore Corollary 24 entails that a mute state is reached, which by Lemma 17 and our completeness assumption is model-describing. □

9.3 A sufficient criterion for the existence of a global basis

The argument above for the termination of CDSAT depends on the existence of a global basis, which is not necessarily entailed by the local bases of the theory modules $I_1, \ldots, I_n$ and their basic properties. As mentioned in Section 4.1, the risk is that, from the set X of terms involved in an input problem, a theory module I_k might at some point introduce a term u_0 in $Y_0 = \mathsf{basis}_k(X)$, which another theory module I_j had not anticipated, and which prompts I_j to introduce a term t_1 in $X_1 = \mathsf{basis}_j(\mathsf{basis}_k(X))$, which in turns prompts I_k to introduce a term u_1 in $Y_1 = \mathsf{basis}_k(\mathsf{basis}_j(\mathsf{basis}_k(X)))$, etc. In other words, despite each of those sets being finite, $\bigcup_{m\in\mathbb{N}} X_m$ might be infinite, where $X_0 = X$ and $X_{m+1} = \mathsf{basis}_j(\mathsf{basis}_k(X_m))$. In order to get a finite global basis, we therefore explore how to permute local bases, e.g. how to relate $\mathsf{basis}_j(\mathsf{basis}_k(X))$ and $\mathsf{basis}_k(\mathsf{basis}_j(X))$. This leads to a sufficient global criterion, about the local bases $\mathsf{basis}_1, \ldots, \mathsf{basis}_n$, for the existence of a global basis. For this we introduce the notion that a theory module may *produce* and *consume* a given sort.

Definition 31 (Production and consumption of a sort)

Let $I = (\ _{\mathcal{I}}, \mathsf{basis}_{\mathcal{I}})$ be a module for theory T with signature $\Sigma = (S, F)$, and let s be in S.

I does not produce s if for all closed set X, all terms in $\mathsf{basis}_{\mathcal{I}}(X)$ of sort s are in X.

I does not consume s if for all closed set X and all terms t of sort s, if t is a Σ-variable or an equality and all of its strict subterms are in X, then

$$\mathsf{basis}_{\mathcal{I}}(X \quad \{t\}) \qquad\qquad (\mathsf{basis}_{\mathcal{I}}(X) \quad \{t\}).$$ ※

We use these two notions to define a binary relation between theory modules: $I_k \quad I_j$ if there exists a sort s such that I_k produces s and I_j consumes s. The intuition is that if $I_k \quad I_j$ then $\mathsf{basis}_j(\mathsf{basis}_k(X)) \quad \mathsf{basis}_k(\mathsf{basis}_j(X))$ for any X. This will imply the following result.

Theorem 27 (Existence of a global basis) If is acyclic, then theories can be numbered so that if $I_k \quad I_j$ then $i \le j$, and for every closed set X, the set $\mathsf{basis}_\infty(X) = \mathsf{basis}_n(\ldots \mathsf{basis}_1(X))$ is a global basis for X with respect to theory modules $I_1, \ldots, I_n$. ※

The rest of the section is devoted to proving Theorem 27, for which we can already notice that $\mathsf{basis}_\infty(X)$ contains X and is finite for every closed set S. So we only have to prove the stability property of a global basis. We assume the hypothesis of Theorem 27 and assume the numbering of the theories is such that if $I_k \quad I_j$ then $k \leq j$.

Lemma 28 (Permutability of local bases) Assume $1 \leq i < j \leq k$ and let X be a closed set.

1. For all sets Y of terms that is closed under the subterm relation and with $X \quad Y \quad \mathsf{basis}_j(X)$, we have $\mathsf{basis}_k(Y) \qquad (\mathsf{basis}_k(X) \quad Y)$.
2. We have $\mathsf{basis}_k(\mathsf{basis}_j(X)) \quad \mathsf{basis}_j(\mathsf{basis}_k(X))$.

※

Proof: First, note that $I_j \quad I_k$, so none of the sorts produced by I_j is consumed by I_k.

1. By induction on the (finite) size of Y.

   If $Y = X$ we are done. Otherwise let t be a term of greatest size in $Y \setminus X$. We know t is in $\mathsf{basis}_j(X)$. Since theories have disjoint signatures, t is either Σ_j-foreign, Σ_k-foreign, a variable or an equality. It cannot be Σ_j-foreign, otherwise the "no introduction of foreign terms" requirement for $\mathsf{basis}_j(X)$ entails that t is in X. It is therefore either a Σ_k-variable or an equality. Moreover, every strict subterm of t is in $Y \setminus \{t\}$ (since Y is closed under the subterm relation), and therefore in $\quad (Y \setminus \{t\})$. Finally, t must be of a sort s that is produced by I_j (otherwise it would be in X) and that is not consumed by I_k. So we apply the definition of s not being consumed by I_k on the closed set $\quad (Y \setminus \{t\})$, and get

   $$\mathsf{basis}_k(\quad (Y \setminus \{t\}) \quad \{t\}) \qquad\qquad (\mathsf{basis}_k(\quad (Y \setminus \{t\})) \quad \{t\}).$$

   On the other hand, let us make two remarks: First, we still have $X \quad (Y \setminus \{t\})$. Second, $Y \setminus \{t\}$ is still closed under the subterm relation: indeed, if t were a strict subterm of a term u in Y, then either u would be in X in which case t would be in X, or u would be in $Y \setminus X$ in which case t would not be a term of greatest size in there. Therefore, we can apply the induction hypothesis on $Y \setminus \{t\}$ and get

   $$\mathsf{basis}_k(Y \setminus \{t\}) \qquad (\mathsf{basis}_k(X) \quad (Y \setminus \{t\})).$$

   Putting it all together:

   $$\begin{array}{rcll}
   \mathsf{basis}_k(Y) & = & \mathsf{basis}_k((Y \setminus \{t\}) \quad \{t\}) & \\
    & & \mathsf{basis}_k(\quad (Y \setminus \{t\}) \quad \{t\}) & \text{as } \mathsf{basis}_k \text{ is monotonic} \\
    & & \quad (\mathsf{basis}_k(\quad (Y \setminus \{t\})) \quad \{t\}) & \text{as proved above} \\
    & = & \quad (\mathsf{basis}_k(Y \setminus \{t\}) \quad \{t\}) & \\
    & & \quad (\quad (\mathsf{basis}_k(X) \quad (Y \setminus \{t\})) \quad \{t\}) & \text{as proved above} \\
    & & \quad (\mathsf{basis}_k(X) \quad (Y \setminus \{t\}) \quad \{t\}) & \\
    & = & \quad (\mathsf{basis}_k(X) \quad Y) &
   \end{array}$$

2. We can derive

   $$\begin{array}{rcll}
   \mathsf{basis}_k(\mathsf{basis}_j(X)) & & (\mathsf{basis}_k(X) \quad \mathsf{basis}_j(X)) & \text{Point 1 with } Y \text{ being } \mathsf{basis}_j(X) \\
    & & (\mathsf{basis}_j(\mathsf{basis}_k(X)) \quad \mathsf{basis}_j(X)) & \text{as } \mathsf{basis}_k(X) \quad \mathsf{basis}_j(\mathsf{basis}_k(X)) \\
    & = & (\mathsf{basis}_j(\mathsf{basis}_k(X))) & \text{as } X \quad \mathsf{basis}_k(X) \\
    & & & \text{and } \mathsf{basis}_j \text{ is monotonic} \\
    & = & \mathsf{basis}_j(\mathsf{basis}_k(X)) & \text{as } \mathsf{basis}_j(\mathsf{basis}_k(X)) \text{ is closed}
   \end{array}$$

□

Lemma 29 (Stability) Given a closed set X and k with $1 \leq k \leq n$,

$$\mathsf{basis}_k(\mathsf{basis}_\infty(X)) = \mathsf{basis}_\infty(X)$$

※

Proof: We show by induction on j the more general result that if $1 \leq k \leq j \leq n$ we have

$$\mathsf{basis}_k(\mathsf{basis}_j(\ldots \mathsf{basis}_1(X))) = \mathsf{basis}_j(\ldots \mathsf{basis}_1(X))$$

The inclusion $\mathsf{basis}_j(\ldots \mathsf{basis}_1(X)) \quad \mathsf{basis}_k(\mathsf{basis}_j(\ldots \mathsf{basis}_1(X)))$ is a requirement on basis_j. For the other direction, we do a case analysis: If $j = k$ then this is simply the idempotence property of basis_k. Otherwise let us assume the induction hypothesis IH for j. We then have

$$\begin{array}{lll} \mathsf{basis}_k(\mathsf{basis}_{j+1}(\mathsf{basis}_j(\ldots \mathsf{basis}_1(X)))) & \mathsf{basis}_{j+1}(\mathsf{basis}_k(\mathsf{basis}_j(\ldots \mathsf{basis}_1(X)))) & \text{Lemma 28.2} \\ & \mathsf{basis}_{j+1}(\mathsf{basis}_j(\ldots \mathsf{basis}_1(X))) & \text{IH} \end{array}$$

□

This finishes the proof of Theorem 27. We finish this section by mentioning which sorts are produced and consumed by the examples of theory modules given in Section 5.

Remark 30 First, most theory modules will produce sort prop, in particular as soon as they may introduce the assignment .

- Modules $\mathcal{I}_{\mathsf{Bool}_{\mathsf{eval}}}$ and $\mathcal{I}_{\mathsf{Bool}}$ for the Boolean theory do not produce or consume any sorts;
- Modules $\mathcal{I}^1$ and $\mathcal{I}^2$, for an abstract theory with a decision procedure, produce sort prop only and do not consume any sorts;
- Module $\mathcal{I}_{\mathsf{LRA}}$ produces sorts prop and Q and consumes sort Q only;
- Module $\mathcal{I}_{\mathsf{EUF}}$ produces sort prop only and does not consume any sorts;
- Module $\mathcal{I}_{\mathsf{Arr}}$ produces all sorts in its signature and consumes all array sorts (such production and consumption occurs because, when seeing $t \quad_{I \Rightarrow V} u$ in the trail, the module can introduce terms $t[\mathsf{di□}(t,u)]$ and $u[\mathsf{di□}(t,u)]$).

※

So these modules can be combined, for instance $\mathcal{I}_{\mathsf{Bool}}, \mathcal{I}^2, \mathcal{I}_{\mathsf{LRA}}, \mathcal{I}_{\mathsf{EUF}}, \mathcal{I}_{\mathsf{Arr}}$, with for any set X the global basis $\mathsf{basis}_{\mathsf{Bool}}(\mathsf{basis}_{\mathcal{I}^2}(\mathsf{basis}_{\mathsf{EUF}}(\mathsf{basis}_{\mathsf{LRA}}(\mathsf{basis}_{\mathsf{Arr}}(X)))))$.

10 Small extensions that do not break termination

In this section we extend CDSAT with three rules: from the previous sections, it is clear that they are not needed for completeness, but we present them because other calculi of the literature that we want to simulate here need them, and rightly so because these extra rules can provide shortcuts in derivations; moreover, they are "safe" in that they preserve justifiedness (as in Lemma 10) and they pass the termination argument (Lemma 19 and Theorem 20).

The following definition allows CDSAT to internalise a conflict (a set of Boolean assignments) in the form of a formula, more precisely a clause.

Definition 32 (Literal assignment and conflict clause) A *literal assignment* is a Boolean assignment $l \leftarrow \mathfrak{b}$ such that l is a Boolean variable (i.e. $\mathsf{fv}_{\Sigma_{\mathsf{Bool}}}(l) = \{l\}$). A *conflict clause* $\mathsf{CC}(E)$

Decide$^+$			
Γ	$\longrightarrow_{\mathcal{B}}$	Γ, A	if A is a $\mathcal{T}_k^+$-assignment for a term in $\mathcal{B}$ and it is acceptable for $\Gamma_{\mathcal{T}_k}$ and $\mathcal{I}_k$, with $1 \leq k \leq n$
Prune			
$\Gamma; E; A$	$=$	$\Gamma \setminus D; E; A$	if D is a set of decisions that are strictly greater than A
Learn			
$\Gamma; J \quad E; A$	$=$	$\Gamma, (J \quad \mathsf{CC}(E)); J \quad E; A$	if E is a set of literal assignments

Figure 5: Extra rules of CDSAT

of a set E of literal assignments is

$$(\bigvee_{(t \leftarrow \mathsf{true}) \in E} \neg t) \quad (\bigvee_{(t \leftarrow \mathsf{false}) \in E} t)$$ ※

Nothing in the definition imposes that there be a conflict, but as we shall see below, we shall only use this concept when E is in the conflict of a conflict state.

The extra rules of CDSAT are given in Fig. 5.

Rule Decide$^+$ allows CDSAT to make a decision on any term in the global basis, rather than on a $\mathcal{T}_k^+$-relevant term of the current trail.

Rule Prune allows CDSAT to prune anything that belongs to a level higher than that of the decision that definitely needs to be undone (being the greatest one that contributes to the conflict). As most SMT-solvers implement the trail as a stack, it is natural, in order to undo a decision, to pop out of the stack anything of a level higher than that of the decision being undone. This rule allows the present CDSAT system to model this behavior.

Rule Learn turns conflicts into clauses and learns them by adding them to the trail. For this to be useful, a Boolean module such as $\mathcal{I}_{\mathsf{Bool}}$ presented in Section 5 needs to be present, otherwise no theory can make sense of the disjunction and the negation symbols. If it is present though, it will be able to re-use that conflict for e.g. unit propagation, so that, even if many theories have been involved in creating the conflict, it can be re-used without involving those theories again. Learning is again a fundamental feature of SAT- and SMT-solvers, which this extra rule thus models in CDSAT. Note that, as expected, learning is not a feature that is needed for completeness (every time a learnt clause is used, the original reasoning justifying that clause could be replayed), but only to speed up the runs. Note that rule Learn allows CDSAT to derive *any* conflict clause encountered during conflict analysis (not necessarily the last one). Also note that the termination argument needs to be adapted, in that the learnt clauses need to be in the global basis: for instance the Boolean module $\mathcal{I}_{\mathsf{Bool}}$ can include all possible learnt clauses in its local basis: for a closed set X, $\mathsf{basis}_{\mathsf{Bool}}(X)$ adds to X all clauses whose Boolean variables appear in X (these are in finite numbers). This modified module now produces sort prop, in the sense of Definition 31, so in order to apply our sufficient criterion for the existence of a global basis, no other theory should consume sort prop (as it is the case with the examples of Section 5).

Other rules that are commonly used in the litterature, like forgetting (learnt clauses) or restarting, can also be added without difficulty but, while they would also preserve state soundness, they may jeopardise termination, unless a specific strategy is used to control them. We therefore leave them as implementation-oriented features.

11 Conclusion

In this report we have provided a generic calculus called CDSAT to combine abstract theories, each of which comes with a theory module that may or may not explicitly use "semantical assignments" such as $x \leftarrow \sqrt{2}$, following the MCSAT approach [11, 18]. The combination being generic, we have identified specifications that we require of the theories and their modules for the generic calculus to be sound, complete, and terminating.

As it can handle several theories abstractly, CDSAT generalizes the MCSAT calculus for the Boolean theory combined with only one abstract theory [11]. CDSAT also generalizes the MCSAT calculus that specifically combines $\mathsf{Bool}, \mathsf{LRA}, \mathsf{EUF}$ [18], and we have expressed the theory-specific mechanisms involved in this combination (or very similar ones) as theory modules satisfying our specifications.

We have also included the decription of a module for the extensional theory of arrays, a theory that had no published MCSAT treatment.

CDSAT is also able to integrate "regular" theory-specific decision procedures used in Nelson-Oppen combinations [27]. In effect, we have reproved the soundness and completeness properties of Nelson-Oppen combinations as a particular case of our soundness and completeness theorems: that where no theory module ever uses semantical assignments. This also opens the door to designing SMT-provers where "MCSAT-like" procedures and "Nelson-Oppen"-like procedures can collaborate.

We chose to express CDSAT at a rather abstract level, not only to avoid relying on theory-specific features but also to avoid committing to specific strategies (for instance using state transition systems rather than more deterministic algorithms) or committing to specific implementation choices (for instance using a graph-based rather than stack-based presentation of trails). The system, and the theorems proved about it, are all the more general as any strategy and implementation choices can be accommodated.

However, several of the more pragmatic aspects of MCSAT, as described for instance in [18], are left aside by this level of abstraction. One of the main questions is how the actual implementation of a theory module can ***detect*** whether one of its inferences is applicable from the current trail or from the extension of the current trail with a potential decision (typically detecting ***acceptability*** of a decision). Let us mention two instances of this issue:

As in several examples of Section 5, a typical CDSAT module inference is ***evaluation***, where the value of a complex term is inferred from the values of its subterms. Detecting acceptability of a decision with regards to the evaluation inferences that it could trigger suggests to keep track of those complex terms that become ***unit constraints***: only one semantical assignment is missing to determine the value of a complex term. Evaluation plays an important role in [11, 18], as does the tracking of unit constraints. The unit completeness property mentioned in [18] is here part of the completeness requirement of theory modules.

Applicability of the LRA-inference that we call here *elimination of empty solution spaces* would in general require a full LRA-solver run, unless the strategy used to assign values to LRA variables does so by systematically treating LRA variables in one particular precedence order (in that case the inference rule is redundant). The generic calculus and its properties are here independent from such strategy issues, but the strategy would here have an impact on how easy it is to detect applicability of inference rules. This follows the general idea that strategies should matter for speed, but not for the soundness, completeness, and termination of the generic calculus.

Acknowledgments Much of this research was conducted when the first author was an international fellow at the Computer Science Laboratory of SRI International in Menlo Park, whose support is greatly appreciated.

References

[1] Clark Barrett, Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. Splitting on demand in SAT modulo theories. In Miki Hermann and Andrei Voronkov, editors, *Proceedings of the 13th International Conference on Logic for Programming, Artificial Intelligence, and Reasoning (LPAR)*, volume 4246 of *Lecture Notes in Artificial Intelligence*, pages 512–526. Springer, 2006. 4

[2] Maria Paola Bonacina and Moa Johansson. Interpolation systems for ground proofs in automated deduction: a survey. *Journal of Automated Reasoning*, 54(4):353–390, 2015. 3

[3] Maria Paola Bonacina, Christopher A. Lynch, and Leonardo de Moura. On deciding satisfiability by theorem proving with speculative inferences. *Journal of Automated Reasoning*, 47(2):161–189, 2011. 5

[4] Maria Paola Bonacina and David A. Plaisted. Semantically-guided goal-sensitive reasoning: model representation. *Journal of Automated Reasoning*, 56(2):113–141, 2016. 5

[5] Maria Paola Bonacina and David A. Plaisted. Semantically-guided goal-sensitive reasoning: inference system and completeness. *Journal of Automated Reasoning*, 59(2):165–218, 2017. 5

[6] Robert Brummayer and Armin Biere. Lemmas on demand for the extensional theory of arrays. *Journal on Satisfiability, Boolean Modeling and Computation*, 6:165–201, 2009. 3, 4

[7] Scott Cotton. Natural domain SMT: A preliminary assessment. In Krishnendu Chatterjee and Thomas A. Henzinger, editors, *Proceedings of the 8th International Conference on Formal Modeling and Analysis of Timed Systems (FORMATS)*, volume 6246 of *Lecture Notes in Computer Science*, pages 77–91. Springer, 2010. 3, 5

[8] Martin Davis, George Logemann, and Donald Loveland. A machine program for theorem-proving. *Communications of the ACM*, 5(7):394–397, 1962. 3

[9] Martin Davis and Hilary Putnam. A computing procedure for quantification theory. *Journal of the ACM*, 7:201–215, 1960. 3

[10] Leonardo de Moura and Nikolaj Bjørner. Model-based theory combination. In Sava Krstić and Albert Oliveras, editors, *Proceedings of the 5th Workshop on Satisfiability Modulo Theories (SMT 2007)*, volume 198(2) of *ENTCS*, pages 37–49. Elsevier, 2008. 3, 4

[11] Leonardo de Moura and Dejan Jovanović. A model-constructing satisfiability calculus. In Roberto Giacobazzi, Josh Berdine, and Isabella Mastroeni, editors, *Proceedings of the 14th International Conference on Verification, Model Checking and Abstract Interpretation (VMCAI)*, volume 7737 of *Lecture Notes in Computer Science*, pages 1–12. Springer, 2013. 3, 4, 5, 6, 12, 44

[12] Leonardo de Moura and Grant Olney Passmore. Computation over real closed infinitesimal and transcendental extensions of the rationals. In Maria Paola Bonacina, editor, *Proceedings of the 24th International Conference on Automated Deduction (CADE)*, volume 7898 of *Lecture Notes in Artificial Intelligence*, pages 177–191. Springer, 2013. 3

[13] Leonardo de Moura and Grant Olney Passmore. Exact global optimization on demand (presentation only). In Silvio Ghilardi, Viorica Sofronie-Stokkermans, and Ashish Tiwari, editors, *Notes of the 3rd Workshop on Automated Deduction: Decidability, Complexity, Tractability (ADDCT)*, pages 50–50, 2013. 3

[14] Leonardo de Moura and Harald Rueß. Lemmas on demand for satisfiability solvers. In *Proceedings of the 5th International Symposium on the Theory and Application of Satisfiability Testing (SAT)*, Lecture Notes in Computer Science, pages 244–251. Springer, 2002. 3, 4

[15] Bruno Dutertre and Leonardo de Moura. A fast linear arithmetic solver for DPLL(T). In Tom Ball and R. B. Jones, editors, *Proceedings of the 18th International Conference on Computer Aided Verification (CAV)*, volume 4144 of *Lecture Notes in Computer Science*, pages 81–94. Springer, 2006. 3, 4

[16] Leopold Haller, Alberto Griggio, Martin Brain, and Daniel Kroening. Deciding floating-point logic with systematic abstraction. In Gianpiero Cabodi and Satnam Singh, editors, *Proceedings of the 12th International Conference on Formal Methods in Computer Aided Design (FMCAD)*. ACM and IEEE, 2012. 3, 5

[17] Marijn Heule, Oliver Kullmann, Siert Wieringa, and Armin Biere. Cube and conquer: guiding CDCL SAT solvers by lookaheads. In K. Eder, J. Lourenço, and O. Shehory, editors, *Proceedings of the 7th Haifa Verification Conference (HVC)*, volume 7261 of *Lecture Notes in Computer Science*, pages 50–65. Springer, 2012. 3

[18] Dejan Jovanović, Clark Barrett, and Leonardo de Moura. The design and implementation of the model-constructing satisfiability calculus. In Barbara Jobstman and Sandip Ray, editors, *Proceedings of the 13th Conference on Formal Methods in Computer Aided Design (FMCAD)*. ACM and IEEE, 2013. 3, 4, 6, 7, 11, 12, 17, 18, 21, 44

[19] Dejan Jovanović and Leonardo de Moura. Cutting to the chase: solving linear integer arithmetic. In Nikolaj Bjørner and Viorica Sofronie-Stokkermans, editors, *Proceedings of the 23rd International Conference on Automated Deduction (CADE)*, volume 6803 of *Lecture Notes in Artificial Intelligence*, pages 338–353. Springer, 2011. 3, 5

[20] Dejan Jovanović and Leonardo de Moura. Solving non-linear arithmetic. In Bernhard Gramlich, Dale Miller, and Ulrike Sattler, editors, *Proceedings of the 6th International Joint Conference on Automated Reasoning (IJCAR)*, volume 7364 of *Lecture Notes in Artificial Intelligence*, pages 339–354. Springer, 2012. 3, 5

[21] Konstantin Korovin, Nestan Tsiskaridze, and Andrei Voronkov. Conflict resolution. In Ian P. Gent, editor, *Proceedings of the 15th International Conference on Principles and Practice of Constraint Programming (CP)*, volume 5732 of *Lecture Notes in Computer Science*, pages 509–523. Springer, 2009. 3, 5

[22] Sava Krstić and Amit Goel. Architecting solvers for SAT modulo theories: Nelson-Oppen with DPLL. In Frank Wolter, editor, *Proceedings of the 6th International Symposium on Frontiers of Combining Systems (FroCoS)*, volume 4720 of *Lecture Notes in Artificial Intelligence*, pages 1–27. Springer, 2007. 4

[23] João P. Marques Silva, Inês Lynce, and Sharad Malik. Conflict-driven clause learning SAT solvers. In Armin Biere, Marjin Heule, Hans Van Maaren, and Toby Walsh, editors, *Handbook of Satisfiability*, volume 185 of *Frontiers in Artificial Intelligence and Applications*, pages 131–153. IOS Press, 2009. 3

[24] João P. Marques Silva and Karem A. Sakallah. GRASP: A search algorithm for propositional satisfiability. *IEEE Transactions on Computers*, 48(5):506–521, 1999. 3

[25] Kenneth L. McMillan, A. Kuehlmann, and Mooly Sagiv. Generalizing DPLL to richer logics. In Ahmed Bouajjani and Oded Maler, editors, *Proceedings of the 21st International Conference on Computer Aided Verification (CAV)*, volume 5643 of *Lecture Notes in Computer Science*, pages 462–476. Springer, 2009. 3, 5

[26] Matthew W. Moskewicz, Conor F. Madigan, Ying Zhao, Lintao Zhang, and Sharad Malik. Chaff: Engineering an efficient SAT solver. In David Blaauw and Luciano Lavagno, editors, *Proceedings of the 39th Design Automation Conference (DAC)*, pages 530–535, 2001. 3

[27] Greg Nelson and Derek C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. on Programming Languages and Systems*, 1(2):245–257, 1979. 4, 15, 44

[28] Robert Nieuwenhuis, Albert Oliveras, and Cesare Tinelli. Solving SAT and SAT modulo theories: from an abstract Davis-Putnam-Logemann-Loveland procedure to DPLL(T). *Journal of the ACM*, 53(6):937–977, 2006. 3, 4

[29] Aaron Stump, Clark W. Barrett, David L. Dill, and Jeremy Levitt. A decision procedure for an extensional theory of arrays. In Joseph Halpern, editor, *Proceedings of the 16th IEEE Symposium on Logic in Computer Science (LICS)*. IEEE Computer Society Press, 2001. 4

[30] Chao Wang, Franjo Ivančić, Malay Ganai, and Aarti Gupta. Deciding separation logic formulae by SAT and incremental negative cycle elimination. In Geoff Sutcliffe and Andrei Voronkov, editors,

Index