Cooperation of Background Reasoners in Theory Reasoning by Residue Sharing

Cesare Tinelli

Department of Computer Science
University of Iowa
Iowa City, IA 52242 – USA
http://www.cs.uiowa.edu/~tinelli

Abstract. We propose a general way of combining background reasoners in theory reasoning. Using a restricted version of the Craig Interpolation Lemma, we show that background reasoner cooperation can be achieved as a form of constraint propagation, much in the spirit of existing combination methods for decision procedures. In this case, constraint information is propagated across reasoners by exchanging *residues* over a common signature. As an application of our approach, we describe a multi-theory version of the semantic tableau calculus.

1 Introduction

Theory reasoning is a powerful deduction paradigm in which a general-purpose reasoner is complemented by a specialized procedure, the *background* reasoner, which (semi-)decides formula satisfiability with respect to a certain background theory. After the pioneering work of Stickel who devised a theory version of resolution [8], nearly all existing calculi for automatic reasoning have been extended to theory reasoning (see [2] for a survey). Essentially all of them however consider the integration of just one background reasoner into the main one.

The usual explanation for such a limitation is that, although integrating several background reasoners would be desirable for modularity and scalability purposes, it is hard to get them to cooperate in a sound and complete way.

We argue in this paper that the cooperation of background reasoners in theory reasoning is instead conceptually simple and can be easily described in terms of partial theory reasoning in the sense of [8]. We will appeal to a variant of a well-known interpolation result, the Craig Interpolation Lemma, to show that background reasoner cooperation can be achieved as a form of constraint propagation, much in the spirit of existing combination methods for decision procedures [5]. The main idea in this case will be to propagate information between reasoners by exchanging quantifier-free *residues* (see later) over a common signature.

The Craig Interpolation Lemma states that whenever two theories $\mathcal{T}_1$ and $\mathcal{T}_2$ are jointly unsatisfiable they have an *interpolant*, a sentence φ made only of symbols shared by $\mathcal{T}_1$ and $\mathcal{T}_2$, which is entailed by one theory and unsatisfiable with the other. Now, although the lemma is in principle enough for the type of reasoner cooperation we suggest, it is not too useful in its general formulation because it provides no information on the syntactical form of interpolants. This is unfortunate because most theory reasoning calculi effectively work only with certain types of formulas (clauses, usually).

We provide a restricted version of the lemma which shows that in the context of theory reasoning all needed interpolants are, in essence, disjunctions of *ground*

literals. Thanks to this result, background reasoner cooperation via constraint propagation becomes a viable option, as we will try to demonstrate. To do that we will describe a multi-theory extension of semantic tableaux, for which the soundness and completeness arguments are particularly straightforward.

It will be clear then that, as in all cases of partial theory reasoning, the real challenge lies in identifying specific situations in which the generation of residues can be implemented in a reasonably efficient and complete way. But we will leave that out of the scope of this paper.

2 Preliminaries

A *signature* Σ consists of a set Σ^{P} of *relation symbols* and a set Σ^{F} of *function symbols*, each with an associated *arity*, an integer $n \geq 0$. A *constant* symbol is a function symbol of zero arity.

We use the standard notions of (Σ-)formula, clause, literal, free and bound variable, substitution, structure (aka model) and so on. Where φ is a formula, we denote by $\mathcal{V}ar(\varphi)$ the set of φ's free variables (with $\mathcal{V}ar(\Phi)$ extending the notation to sets Φ of formulas as obvious). A *sentence* is a formula with no free variables. A *ground* formula is a formula with no variables. A *theory* is a set of sentences—we do not insist that the set be consistent. We denote by $\bot$ the universally false formula and assume that it is a (ground) Σ-literal for every signature Σ.

We also use the usual notions of satisfiability and entailment but extended to the case of formulas, as opposed to sentences, as done in the mathematical logic literature. Specifically, a set Φ of formulas is *satisfiable* in a structure $\mathcal{A}$ if there is a valuation of $\mathcal{V}ar(\Phi)$ into elements of $\mathcal{A}$ that makes every formula in Φ true in $\mathcal{A}$. A formula φ is *satisfiable* in a structure $\mathcal{A}$ if $\{\varphi\}$ is satisfiable in $\mathcal{A}$. A set of formulas (resp. a formula) is *satisfiable* if it is satisfiable in some structure $\mathcal{A}$, and *unsatisfiable* otherwise. The set Φ *entails* the formula φ, in symbols $\Phi \models \varphi$, if every structure and valuation that satisfy Φ also satisfy φ; equivalently, if the set $\Phi \cup \{\neg\varphi\}$ is unsatisfiable. Notice that Φ is unsatisfiable if and only if $\Phi \models \bot$.

The reader unfamiliar with this notion of satisfiability/entailment for formulas should observe that in it free variables essentially behave as free constant symbols. This differs from the practice in automated reasoning of treating free variables as implicitly universally quantified.[1] The distinction here is important and should be kept in mind because, for example, when we talk about the satisfiability of a clause we are quantifying its variables universally, whereas when we talk about the satisfiability of a quantifier-free formula we are not. This means for instance that when $\{p(x), \neg p(y)\}$ denotes a set of *clauses*, it is unsatisfiable; when it denotes instead a set of *literals*, it is satisfiable.

In theory reasoning, satisfiability and entailment are also given with respect to a certain theory. The definitions below subsume the various, slightly different ones in the literature.

Definition 1. *Let $\mathcal{T}$ be any theory. A set Φ of formulas is* $\mathcal{T}$-satisfiable *iff $\mathcal{T} \cup \Phi$ is satisfiable; otherwise it is* $\mathcal{T}$-unsatisfiable*. A formula φ is a* $\mathcal{T}$-consequence of Φ *in symbols $\Phi \models_{\mathcal{T}} \varphi$, iff $\mathcal{T} \cup \Phi \models \varphi$.*

[1] Perhaps because of the fact that clauses are written without their universal quantifier prefix.

A formula is *universal* if it is in prefix normal form and its (possibly empty) quantifier prefix contains only universal quantifiers. A theory is universal if it is axiomatized by a set of universal sentences.[2] Theory reasoning considers only universal theories as background theories because they are the only ones that can be "safely" built-in into a deduction calculus. The reason is that the Herbrand theorem extends immediately to $\mathcal{T}$-satisfiability in a universal theory.

Proposition 1 (Herbrand Theorem for Theory Reasoning). *Let $\varphi_1, \ldots, \varphi_n$ be $n > 0$ quantifier-free formulas and $\mathcal{T}$ a universal theory. Then the set $\{\tilde{\forall}\varphi_1, \ldots, \tilde{\forall}\varphi_n\}$ (where $\tilde{\forall}\,\varphi$ denotes the universal closure of φ) is $\mathcal{T}$-unsatisfiable iff there is a finite set of ground instances of formulae from $\{\varphi_1, \ldots, \varphi_n\}$ that is $\mathcal{T}$-unsatisfiable.*

3 Theory Reasoning over Multiple Theories

3.1 Partial Theory Reasoning

Because of its generality, theory reasoning encompasses a vast array of seemingly different reasoning frameworks. Here we will focus on what [2] calls *literal level theory reasoning*.[3] The basic operation in traditional refutation-based calculi is the detection of pairs of complementary literals; in other words, the detection of an unsatisfiable set Φ made of two quantifier-free formulas of a specific kind. Literal level theory reasoning generalizes this operation in two directions: the type and number of quantifier-free formulas in Φ, and the notion of (un)satisfiability, defined with respect to a certain *background theory* $\mathcal{T}$ in the sense of Def. 1.

In theory reasoning systems, the $\mathcal{T}$-satisfiability test is not performed by the main reasoner, the *foreground reasoner*, but is delegated instead to a specialized subsystem, the *background reasoner* for $\mathcal{T}$. Roughly, we speak of *total theory reasoning* if the background reasoner gets a set Φ of formulas from the foreground one and simply confirms whether Φ is $\mathcal{T}$-unsatisfiable or not. We speak of *partial theory reasoning* if, whenever Φ is not $\mathcal{T}$-unsatisfiable, the background reasoner returns a *residue* for it, that is, a quantifier-free formula whose negation, if added to Φ, would make it $\mathcal{T}$-unsatisfiable.

The precise definition of residue varies depending on the author and the partial theory reasoning calculus in question. But they are all instances of the one below.

Definition 2 (Residue). *Let Φ be a set of quantifier-free formulas, called a* key set*. Let ψ be a quantifier free formula and σ a substitution. The pair $\langle \sigma, \psi \rangle$ is a* $\mathcal{T}$-residue *of Φ iff the set $\Phi\sigma \cup \{\neg\psi\}$ is $\mathcal{T}$-unsatisfiable or, equivalently, iff $\Phi\sigma \models_{\mathcal{T}} \psi$.*

According to the definition above, the pair $\langle \sigma, \bot \rangle$ a is $\mathcal{T}$-residue of the key set Φ if and only if $\Phi\sigma$ is $\mathcal{T}$-unsatisfiable. More precisely then, we talk about total theory reasoning when the background reasoner computes only residues of the form $\langle \sigma, \bot \rangle$, if any, and partial theory reasoning otherwise.

In the following, we will simply say *residue* instead of $\mathcal{T}$-residue whenever $\mathcal{T}$ is clear from context. Abusing the terminology, we will also call residue the second component of a residue $\langle \sigma, \psi \rangle$, especially when σ is the empty substitution.

[2] Some authors refer to universal theories as *open* or *quantifier-free theories*, again because of the common practice of writing their axioms without the quantifiers.

[3] The other forms of theory reasoning can be recast as essentially special cases of the literal level form.

3.2 Combining Background Reasoners

Suppose we are interested in a background theory $\mathcal{T}$ obtained as the union of $n > 1$ theories $\mathcal{T}_1, \ldots, \mathcal{T}_n$. Also suppose that we do not have a background reasoner for $\mathcal{T}$ but we do have one for each $\mathcal{T}_i$. From a practical standpoint, instead of implementing a reasoner for $\mathcal{T}$ anew, it would be interesting to integrate the reasoners for the various $\mathcal{T}_i$ directly into a foreground reasoner and have them work together to detect the $\mathcal{T}$-unsatisfiability of formulas. The question then is how to make the reasoners cooperate in a sound and complete way.

In this section, we provide some interpolation results which suggest that background reasoners can cooperate by exchanging residues over a common quantifier-free language. In the next section, we will embed this kind of cooperation into a specific theory reasoning calculus and argue that the resulting calculus is sound and complete. For simplicity, we will consider the case of just two background theories. From what follows, however, it should be clear that all the given results lift by iteration to the case of more than two theories.

We will impose no model-theoretic restrictions on the two theories other than universality. Also, we will make no assumptions on whether the theories share no, some or all predicate symbols. We will assume, however, that their signatures have *exactly* the same set of function symbols. For our purposes such a restriction is not as stringent as it sounds. In fact, background reasoners used in theory reasoning typically accept input formulas containing *free* function symbols, that is, symbols not appearing in their theory.[4] In that case, the signature of each background theory can be always expanded with free symbols until all the signatures get to have the same set of function symbols.

3.3 The Interpolation Lemma

The main theoretical result of the paper is provided by the following restricted version of the Craig Interpolation Lemma.

Proposition 2 (Ground Interpolation Lemma). *Let Σ_1, Σ_2 be two signatures such that ${\Sigma_1}^{\mathrm{F}} = {\Sigma_2}^{\mathrm{F}}$ and let $\Sigma := \Sigma_1 \cap \Sigma_2$. For $i = 1, 2$ let $\mathcal{T}_i$ be a universal theory of signature Σ_i. The universal theory $\mathcal{T}_1 \cup \mathcal{T}_2$ is unsatisfiable iff there is a disjunction ψ of ground Σ-literals such that $\mathcal{T}_1 \models \psi$ and $\mathcal{T}_2 \models \neg\psi$.*

The proof of this proposition is fairly routine by model theory standards.[5] It is based on three main facts: 1) every substructure of a model of a universal theory is in turn a model of the theory; 2) a set of generators of a structure does not change if one adds or removes predicate symbols from its signature; 3) a model $\mathcal{A}_1$ of $\mathcal{T}_1$ and a model $\mathcal{A}_2$ of $\mathcal{T}_2$ above can be amalgamated into a model of $\mathcal{T}_1 \cup \mathcal{T}_2$ whenever $\mathcal{A}_1$ and $\mathcal{A}_2$ are isomorphic over the shared signature Σ.

For our purposes, the following corollary of the proposition will be more useful.

[4] Free function symbols occur naturally in refutation-based theory reasoning calculi as a result of Skolemization of existential variables.

[5] Variation on the theme are found as exercises in many mathematical logic or model theory textbooks.

Proposition 3. *Let Σ_1, Σ_2 be two signatures such that ${\Sigma_1}^{\mathrm{F}} = {\Sigma_2}^{\mathrm{F}}$ and let $\Sigma := \Sigma_1 \cap \Sigma_2$. For $i = 1, 2$ let $\mathcal{T}_i$ be a universal Σ_i-theory and Φ_i a set of quantifier-free Σ_i-formulas. Then, the following are equivalent:*

1. *$\Phi_1 \cup \Phi_2$ is $(\mathcal{T}_1 \cup \mathcal{T}_2)$-unsatisfiable;*
2. *there is a disjunction ψ of Σ-literals with $\mathcal{V}ar(\psi) \subseteq \mathcal{V}ar(\Phi_1 \cup \Phi_2)$ such that*

$$\Phi_1 \models_{\mathcal{T}_1} \psi \quad \text{and} \quad \Phi_2 \cup \{\psi\} \models_{\mathcal{T}_2} \bot \ .$$

The existence of an interpolant like ψ above is already guaranteed by the Craig Interpolation Lemma. The contribution of Prop. 3 is to specify that, under the given assumptions, Φ_1 and Φ_2 have an interpolant which is a disjunction of literals with no new variables.

Prop. 3 holds in the exact same formulation in both flavors of first-order logic: the one without equality, the traditional logic of automated reasoning, which we use here, and the one with equality, which treats the equality symbol as a logical constant and always interprets it as the identity relation. The only difference is that, whereas in the second flavor interpolants may in general contain equations, they do so in the first flavor only if equality is explicitly defined in *both* theories $\mathcal{T}_1$ and $\mathcal{T}_2$. In particular, this entails that when the theories share no predicate symbols at all the only possible interpolant of Φ_1 and Φ_2 in Prop. 3 is either $\bot$ or its negation.[6]

4 A Multi-Theory Tableau Calculus

We will now show how the interpolation results of the previous section can be used to integrate multiple background reasoners into a theory reasoning calculus. To be specific we will define a multi-reasoner extension of the partial theory version of (free variable) semantic tableaux. Our exposition of the calculus will follow closely the one given in [3]. There, a tableau is defined as a multiset of tableau branches, where a branch is a multiset of first-order formulas. As usual, branches are interpreted logically as the conjunction of their elements, and tableaux as the disjunction of their branches.

Tableaux are manipulated according to the derivation rules listed in Fig. 1 which operate on the types of formulas defined by the tables in the same figure. The main idea of the theory tableau calculus is to *close* a tableau branch by detecting its unsatisfiability with respect to a given background theory $\mathcal{T}$. Operationally, this is done by choosing a subset of the formulas in the branch and asking a specialized reasoner about their $\mathcal{T}$-unsatisfiability.

In our extension of the calculus we consider background theories $\mathcal{T}$ that are in fact the union of a universal Σ_1-theory $\mathcal{T}_1$ and a universal Σ_2-theory $\mathcal{T}_2$. Although one would certainly want $\mathcal{T}_1$ and $\mathcal{T}_2$ to be such that $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$ is consistent, technically, do not need to assume that. We do assume though that the signatures of $\mathcal{T}_1$ and $\mathcal{T}_2$ have exactly the same function symbols, as discussed in the previous section.

We also assume that for each $\mathcal{T}_i$ we have a background reasoner that accepts a Σ_i-key set (cf. Def. 2) as input and is capable of enumerating all of its $\mathcal{T}_i$-residues.

Now let $\Sigma := \Sigma_1 \cap \Sigma_2$. The main idea of our extension is to collect into a key set only formulas in a branch that have the same signature, either Σ_1 or Σ_2. The assembled Σ_i-key set ($i = 1, 2$) is given to the corresponding background reasoner,

[6] Notice that $\neg\bot$ is the interpolant of Φ_1 and Φ_2 exactly when $\bot$ is the interpolant of Φ_2 and Φ_1.

which returns a residue for it. The residue may contain Σ-formulas, which can be later included into a Σ_j-key set (with $j \neq i$) by virtue of the fact that $\Sigma \subseteq \Sigma_j$. In practice, this means that the two background reasoners share information about the satisfiability of a branch by exchanging residues over the language of quantifier-free Σ-formulas, understood by both. We will see that, by the interpolation results of the previous section, this is in principle all they need to do to detect the $\mathcal{T}$-unsatisfiability of a branch.

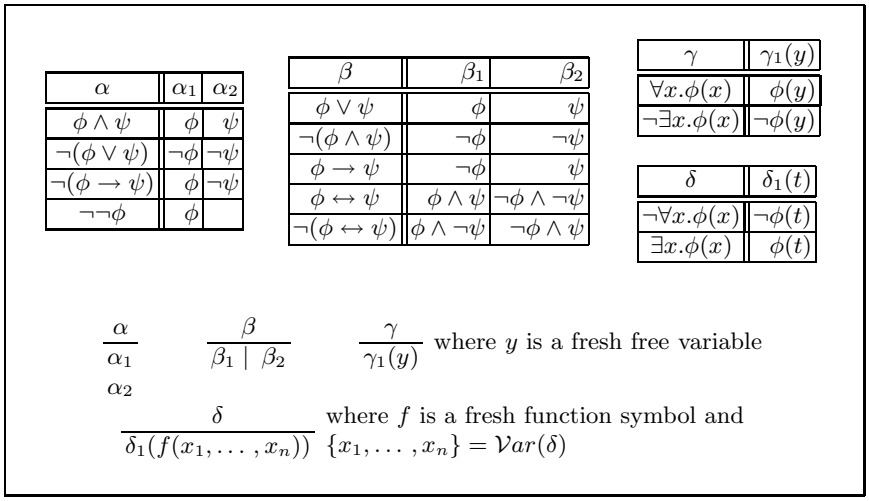

Fig. 1. Formula Types and Tableau Derivation Rules.

The following definition extends the usual notion of tableau derivation to our multi-theory framework.

Definition 3 (Derivation). *A sequence $T_0, T_1, \ldots, T_n$ of $n \geq 0$ tableaux is a* derivation *of T_n from T_0 iff for all $j > 0$, T_j is obtained from T_{j-1}*

1. *by applying one of the derivation rules in Fig. 1 to a branch $B \in T_{j-1}$, i.e., by defining T_j as follows for some $\varphi \in B$:*

$$T_j := \begin{cases} (T_{j-1} \setminus \{B\}) \cup \{(B \setminus \{\varphi\}) \cup \{\alpha_1, \alpha_2\}\} & \text{if } \varphi \text{ is of type } \alpha \\ (T_{j-1} \setminus \{B\}) \cup \{(B \setminus \{\varphi\}) \cup \{\beta_1\}, (B \setminus \{\varphi\}) \cup \{\beta_2\}\} & \text{if } \varphi \text{ is of type } \beta \\ (T_{j-1} \setminus \{B\}) \cup \{B \cup \{\gamma_1\}\} & \text{if } \varphi \text{ is of type } \gamma \\ (T_{j-1} \setminus \{B\}) \cup \{(B \setminus \{\varphi\}) \cup \{\delta_1\}\} & \text{if } \varphi \text{ is of type } \delta \end{cases}$$

2. *or by adding a $\mathcal{T}_i$-residue (for $i = 1$ or $i = 2$) to a branch $B \in T_{j-1}$, i.e., by defining T_j as follows, where $\langle \sigma, \psi \rangle$ is a $\mathcal{T}_i$-residue of some Σ_i-key set $\Phi \subseteq B$:*

$$T_j := (T_{j-1} \setminus \{B\})\sigma \cup \{B\sigma \cup \{\psi\}\}$$

3. *or by closing a branch $B \in T_{j-1}$, i.e., by defining T_j as $T_{j-1} \setminus \{B\}$ if $\bot \in B$.*

The tableau calculus defined by the derivations in Def. 3 is sound and complete in the following sense.

Theorem 1 (Soundness and Completeness). *A $(\Sigma_1 \cup \Sigma_2)$-sentence φ is entailed by $\mathcal{T} = \mathcal{T}_1 \cup \mathcal{T}_2$ iff there is a derivation of the empty tableau from the tableau $\{\{\neg\varphi\}\}$.*

It is easy to prove that every derivation preserves the $\mathcal{T}$-satisfiability of a tableau, which immediately shows the soundness of the calculus. To prove its completeness one can proceed as in the case of just one background theory by showing that, if a sentence is $\mathcal{T}$-unsatisfiable, it is possible to derive from it a tableau all of whose branches can be closed. We sketch the proof below by considering for simplicity just the ground case.

Let us say that a formula is *pure* if it is either a Σ_1-formula or a Σ_2-formula. Let $T_0 := \{\{\neg\varphi\}\}$ be the initial tableau and assume that the ground $(\Sigma_1 \cup \Sigma_2)$-formula $\neg\varphi$ is $\mathcal{T}$-unsatisfiable. Applying a proper sequence of derivation rules from Fig. 1, it is possible to derive from T_0 a tableau T all of whose branches are $\mathcal{T}$-unsatisfiable and made of pure ground formulas only. Let B be any of these branches.

Since B is $\mathcal{T}$-unsatisfiable it has a subset Φ_i of Σ_i-formulas, for $i = 1, 2$, such that $\Phi_1 \cup \Phi_2$ is $(\mathcal{T}_1 \cup \mathcal{T}_2)$-unsatisfiable. By Prop. 3, Φ_1 and Φ_2 have a ground Σ-interpolant ψ. Now, ψ is a residue of the Σ_1-key set Φ_1 by definition, hence it can be added to B by Point 2 of Def. 3. Moreover, $\Phi_2 \cup \psi$ is a $\mathcal{T}_2$-unsatisfiable Σ_2-key set, hence its residue $\bot$ can be added to B by the same point. But then B can be closed by Point 3 of Def. 3. This shows that the calculus is complete for ground sentences. Thanks to Prop. 1, the completeness proof can then be lifted to the non-ground sentences as usual.

It is typical for a background reasoner to accept only key sets made just of literals (and still produce disjunctions of literals as residues). We leave it to the reader to verify that our extension remains complete even if we only use this restricted kind of key set.

We conclude this section with some general remarks on our combination results.

First, we would like to stress that the cooperation method described here is not limited to tableaux only. For instance, we have readily applied it to theory resolution as well, with a soundness and completeness proof for the resulting calculus that is almost as simple as in the tableau case. We have every reason to believe that the method can also be used to produce a multi-theory extension of all the other existing theory reasoning calculi.

Finally notice that, as pointed out in Subsect. 3.3, when $\mathcal{T}_1$ and $\mathcal{T}_2$ have no predicate symbols in common (including equality) the only possible non-tautological interpolant between the pure parts of a branch is $\bot$. In that case then the two background reasoners are fully decoupled[7] and a total theory reasoning strategy can be adopted separately with each of them. All the foreground reasoner needs to do is to "purify" a branch enough, using the usual tableaux derivation rules, until it can assemble a pure key set that will be found unsatisfiable by the corresponding background reasoner.

This sort of approach was already outlined in [8] for (single-)theory resolution. Stickel argued that key sets never needed to include literals whose predicate symbol did not belong to the background theory. These literals could instead be processed by standard resolution steps. His argument can be given a formal justification in our cooperation framework by looking at *non-theory* reasoning steps as theory reasoning

[7] Strictly speaking, this is true only for ground derivations. In non-ground derivations, the reasoners share the substitutions applied to the free variables of the tableau.

over an empty background theory. Then, one can simply consider the extra predicate symbols as part of a second, empty theory and the usual inference steps identifying complementary literals as a background reasoner returning only $\bot$ residues.

5 Related Work

The only research we are aware of that focuses on the cooperation of background reasoners in theory reasoning is that reported in [4, 9, 1, 7]. Except for [7], all of these works embed a well-known combination method by Nelson and Oppen [5] into a specific theory reasoning calculus: analytic tableaux in [4], a variant of the CLP scheme in [9], and constrained resolution in [1].

Although it is not entirely accurate, we could say that the approach in each of these papers is a specialization of the one presented here. One essential difference is that the signatures of the two background theories share at most the equality and the constant symbols, whereas in our case they must share all function symbols. Another is that the two theories are *stably-infinite* (see, e.g., [6]), which is not required in our case. The net effect of these differences, leading to stronger decidability results, is that for each key set Φ it is enough to consider only the finitely many residues of the form $\langle \varepsilon, \psi \rangle$ where ε is the empty substitution and ψ a disjunction of equations between the variables in Φ.

In [7], certain special types of background theories are integrated into the theory connection calculus. A number of rather specific syntactical restrictions are imposed on the theories, including the disjointness of their sets of predicate symbols, none of which are necessary in our approach. It is presently unclear to us how the approach in [7] compares to ours.

Acknowledgments. Thanks to the anonymous referees for their helpful comments and suggestions.

References

1. J. Van Baalen and S. Roach. Using decision procedures to accelerate domain-specific deductive synthesis systems. In P. Flener, editor, *Proc. of the 8th Int. Workshop on Logic Programming Synthesis and Transformation, LOPSTR'98*, volume 1559 of *Lecture Notes in Computer Science*, pages 61–70. Springer, 1999.
2. P. Baumgartner, U. Furbach, and U. Petermann. A unified approach to theory reasoning. Research Report 15–92, Universität Koblenz-Landau, Germany, 1992.
3. B. Beckert and Ch. Pape. Incremental theory reasoning methods for semantic tableaux. In P. Miglioli *et al*, editors, *Proc. of the 5th Workshop on Theorem Proving with Analytic Tableaux and Related Methods, TABLEAUX'92*, volume 1071 of *Lecture Notes in Computer Science*, pages 93–109. Springer, 1996.
4. T. Käufl and N. Zabel. Cooperation of decision procedures in a tableau-based theorem prover. *Revue d'Intelligence Artificielle*, 4(3):99–126, 1990.
5. G. Nelson and D. C. Oppen. Simplification by cooperating decision procedures. *ACM Trans. on Programming Languages and Systems*, 1(2):245–257, October 1979.
6. D. C. Oppen. Complexity, convexity and combinations of theories. *Theoretical Computer Science*, 12:245–257, 1980.
7. U. Petermann. Connection calculus theorem proving with multiple built-in theories. *Journal of Symbolic Computation*, 29(2):373–392, February 2000.
8. M. E. Stickel. Automated deduction by theory resolution. *Journal of Automated Reasoning*, 1(4):333–355, 1985.
9. C. Tinelli and M. T. Harandi. Constraint logic programming over unions of constraint theories. *The Journal of Functional and Logic Programming*, 1998(6).