A Graph-Based Approach to build Extended Canonizers

Silvio Ranise[1] and Christelle Scharff[2⋆]

[1] LORIA-INRIA-Lorraine, 615 Rue du Jardin Botanique, BP 101, 54 602 Villers-les-Nancy, France and Università di Milano, Italy, Email: ranise@loria.fr
[2] Department of Computer Science, Pace University, One Pace Plaza, New York, NY 11106, USA, Email: cscharff@pace.edu

Abstract. We adapt and combine work on rewriting-based decision procedures for T-satisfiability and SER graphs, a graph-based method defined for abstract congruence closure, to build graph-based decision procedures and compute extended canonizers for the theory of equality and the theory of lists *à la* Shostak. Based on graphs our approach addresses implementation issues that were lacking in previous rewriting-based decision procedure approaches and which are important to argue its viability.

1 Introduction

In [9] we have introduced the concept of *extended canonizer* as the basic modularity concept to design a combination schema featuring the efficiency of Shostak method [11] and the modularity of the one by Nelson and Oppen [7]. An extended canonizer *ecan* is a module which allows us to canonize terms with respect to a given theory T and a given T-satisfiable set of equalities Γ, so that the uniform word problem for T, i.e. $T \models \Gamma \Rightarrow s = t$, reduces to the problem of checking the identity $ecan(\Gamma)(s) = ecan(\Gamma)(t)$, where $ecan(\Gamma)(s)$ and $ecan(\Gamma)(t)$ are the "extended canonical forms" of s and t respectively. More specifically, let T be a Σ-theory with decidable uniform word problem and Γ be a conjunction of Σ-equalities. Given any T-satisfiable Γ, an *extended canonizer* for T is a function $ecan(\Gamma) : T(\Sigma, X) \rightarrow T(\Sigma \cup \mathcal{K}, X)$, such that, for any terms s, t, we have $T \models \Gamma \Rightarrow s = t$ iff $ecan(\Gamma)(s) = ecan(\Gamma)(t)$, where $\mathcal{K}$ is a set of fresh constant symbols such that $\Sigma \cap \mathcal{K} = \emptyset$. Each of these constants represents a distinct sub-terms in Γ and permit us to obtain a *flat* set of equalities. Note that this transformation obviously preserves satisfiability. We assume $\succ$ to be a simplification ordering (see e.g. [2]), which is total on ground terms and such that $t \succ c$ for each non-constant term $t \in \mathcal{T}(\Sigma \cup \mathcal{K})$ and constant c of $\mathcal{K}$. Then, by the exhaustive application of the rules of the superposition calculus (called saturation) [8], we obtain a canonical rewrite system, which is confluent and terminating and permits us to define our extended canonizer $ecan(\Gamma)(s)$ for any ground term s of $\mathcal{T}(\Sigma)$ (see [9] for details). Notice that if a theory T admitting an extended canonizer *ecan* is also convex, then it is always possible to build a satisfiability procedure for T by recalling that $\Gamma \wedge \neg e_1 \wedge \cdots \wedge \neg e_n$ is T-unsatisfiable if and only if there exists some $i \in \{1, \ldots, n\}$ such that $\Gamma \wedge \neg e_i$ is T-unsatisfiable or equivalently $T \models \Gamma \Rightarrow e_i$.

In [9] we have briefly described how to build extended canonizers for some important theories like the theory of equality and the theory containing a commutative function symbol, by extending the rewriting approach to satisfiability procedure described in [1]. However, for lack of space, the discussion in [9] is sketchy and does not address complexity and implementation issues which are important to argue the viability of the proposed concept.

In this paper, we overcome this shortcoming and we describe how to build extended canonizers for the theory of equality and the theory of lists *à la* Shostak[3]. To this end, we adapt and generalize the concept of SER graphs. SER graphs [10], a specialized version

⋆ This work is supported by the National Science Foundation under grant ITR-0326540.

[3] The theory of lists *à la* Shostak is defined by the axioms: $cdr(cons(X, Y)) \approx X$, $car(cons(X, Y)) \approx Y$, $cons(car(X), cdr(X)) \approx X$.

of the SOUR graphs [6] that were developed for general completion, present an efficient graph-based method that combines the key ideas of completion [5] and abstract congruence closure [3] to provide a graph-based decision procedure for the word problem of the underlying (ground) equational theory.The corresponding disadvantage is that it permits us to obtain a convergent rewrite system over the original signature only by further transforming the graph. Directed (SER) graphs that support full structure sharing are used to represent terms and equalities. Each vertex v is labeled by (i) a function symbol of Σ denoted by $Symbol(v)$, and (ii) a constant of $\mathcal{K}$ denoted by $Constant(v)$. The vertices labeled by constants of $\mathcal{K}$ represent terms or more generally equivalence classes of terms. Edges carry information about subterm relationships between terms (S), rewrite rules (R) and unordered equalities (E). We write $u -_E v$ and $u \rightarrow_R v$ to denote equality and rewrite edges (between vertices u and v), respectively. Subterm edges are also labeled by an index, and we write $u \rightarrow^i_S v$. Informally, this subterm edge indicates that v represents the i-th subterm of the term represented by u. This graph structure provides a suitable basis for computing (abstract) congruence closures as graph transformation rules as described in [10]. The efficiency of SER graphs crucially depends on the use of a simple ordering (that needs to be defined only on $\mathcal{K}$), rather than a full term ordering.

The advantage of SER graphs is that they allow us to easily reuse ideas to build decision procedures developed in the rewriting approach [1] in a direct and natural way. They permit us to compute normal forms with respect to both an extended and the original signature by a suitable post-processing phase (by using compression and selection rules [3]). Besides we believe that they provide us with refined characterization of the computational complexity of the developed algorithms, which is not the case for [1] because of the abstract notion of computation (i.e. a calculus) used.

In this paper we assume the reader is familiar with standard terminology of equational logic and rewriting (see e.g. [2]). In the following let Σ be a set of function symbols and $\mathcal{K}$ be a set of constants such that $\Sigma \cap \mathcal{K} = \emptyset$. We call Σ the (basic) *signature*, and $\Sigma \cup \mathcal{K}$ the *extended signature*. This paper is organized in the following way. In section 2 we show how to build decision procedures and compute extended canonizers for the theory of equality and the theory of lists *à la* Shostak in our framework and we conclude in section 3.

2 SER Graphs for Extended Canonizers

In this section we describe how to combine and adapt work from [9, 10] to build decision procedures and compute extended canonizers for the theory of equality and the theory of lists *à la* Shostak. In particular we show how SER graphs represent the state of the procedures whose computations are described by a suitable set of transition rules ST (applied on an initial SER graph) and how to compute extended canonizers from saturated SER graphs.

2.1 From SER Graphs to Extended Canonizers

Initial SER Graph An *initial* SER graph, $DAG(\Gamma)$, represents a set of equalities Γ as well as the subterm structure of terms in Γ. It is characterized by the following conditions: (i) If $Symbol(v)$ is a constant, then v has no outgoing subterm edges; and (ii) if $Symbol(v)$ is a function symbol of arity n, then there is exactly one edge of the form $v \rightarrow^i_S v_i$, for each i with $1 \leq i \leq n$.

The term $Term(v)$ *represented* by a vertex v is recursively defined as follows: If $Symbol(v)$ is a constant, then $Term(v) = Symbol(v)$; if $Symbol(v)$ is a function symbol of arity n, then $Term(v) = Symbol(v)(Term(v_1), \ldots, Term(v_n))$, where $v \rightarrow^i_S v_i$, for $1 \leq i \leq n$. Evidently, $Term(v)$ is a term over signature Σ. We require that distinct vertices of $DAG(\Gamma)$ represent different terms. Moreover, we insist that $DAG(\Gamma)$ contain no rewrite edges and that each

equality edge $u -_E v$ correspond to an equality $s \approx t$ of Γ (with u and v representing s and t, respectively), and vice versa. The vertices of the graph $DAG(\Gamma)$ also represent flat terms over the extended signature $\Sigma \cup \mathcal{K}$. More specifically, if $Symbol(v)$ is a constant, then $ExtTerm(v) = Constant(v)$, and if $Symbol(v)$ is a function symbol of arity n, then $ExtTerm(v) = Symbol(v)(Constant(v_1), \ldots, Constant(v_n))$, where $v \rightarrow_S^i v_i$, for $1 \leq i \leq n$.

SER Graph Transformations Rules The SER graph transformation rules are formally defined as pairs of tuples of the form $(E_s,\ E_e,\ E_r,\ V,\ \mathcal{K},\ KC,\ C) \rightarrow (E'_s,\ E'_e,\ E'_r,\ V',\ \mathcal{K}',\ KC',\ C')$, where the individual components specify a graph, an extended signature, and an ordering on constants, before and after rule application. Specifically,

- the first three components describe the sets of subterm, equality, and rewrite edges, respectively;
- the fourth component describes the set of vertices[4];
- the fifth component describes the extension of the original signature Σ;
- the sixth component describes the (partial) ordering on constants. Specifically, KC is a set of "ordering constraints" of the form $\{c_i \succ c_j \mid c_i,\ c_j \in \mathcal{K}\}$. (A set of such constraints is considered *satisfiable* if there is an irreflexive, transitive relation on $\mathcal{K}$ that meets all of them); and
- the last component describes the function C that associates a constant of $\mathcal{K}$ and a constant of Σ to a vertex of the graph. The signature of C is $V \rightarrow \mathcal{K} \times \Sigma$. Let $Dom(f)$ be the domain of a function f and $update(f, i, e)$ be a function f' which is identical to f for every value in $Dom(f)$ except for i for which $f'(i) = e$. Indeed, $dom(update(f, i, e)) = dom(f) \cup \{i\}$.

Correctness Exhaustive application of the graph transformation rules is *sound* in that the equational theory represented over Σ-terms does not change. It *terminates*; this can be proved by assigning a suitable weight to graphs that decreases with each application of a transformation rule. It is *complete* in that the (extended) rewrite system that can be extracted from the final graph is convergent.

Extended Canonizers Computing an extended canonizer $ecan(\Gamma)(s)$ of a term $s \in \mathcal{T}(\Sigma)$ considering a set of ground equalities Γ and a theory T is performed by saturating the initial SER Graph $DAG(\Gamma)$ w.r.t. a set of transformation rules ST to obtain a graph G' and by (recursively) integrating s to this saturated graph.

- If s is a constant c of Σ and c labels a vertex v of G', v represents s.
- If s is a constant c of Σ that is not present on G', a new vertex v labeled by c and a new constant c_{new} are added to G' to represent s.
- If $s = f(s_1, \ldots, s_n)$, a new vertex v labeled by f and a new constant c_{new} are added to G' to represent s. The terms s_i representing v_i are recursively integrated to the graph such that there is exactly one S edge of the form $v \rightarrow_S^i v_i$, for each i with $1 \leq i \leq n$.

A marker # is added to point on the vertex v representing s. The graph is saturated w.r.t. the same set of transformation rules ST. $ecan(\Gamma)(s)$ is computed by following the marker #:

- If the marker points on a vertex v with an outgoing R edge to a vertex w, $ecan(\Gamma)(s) = Constant(w)$.
- Otherwise, $ecan(\Gamma)(s) = ExtTerm(v)$.

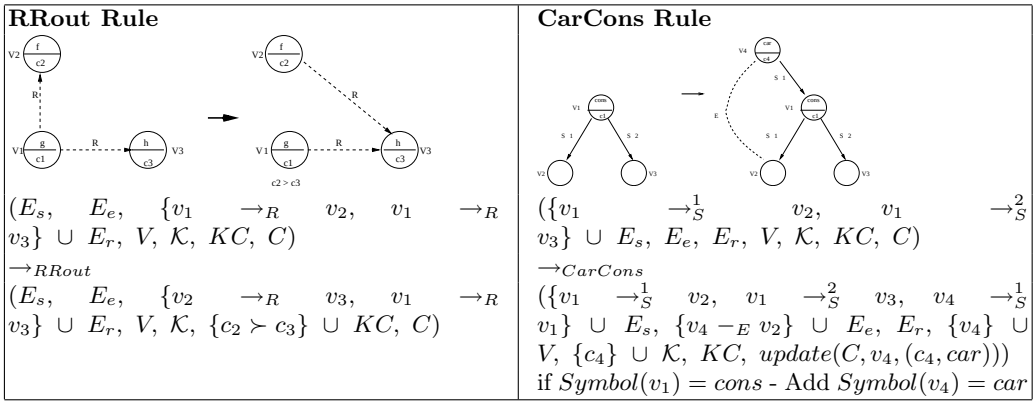
[4] Note that, in the theory of equality, vertices are deleted only. In the theory of lists *à la* Shostak vertices are added and deleted.

2.2 Extended Canonizers for the Theory of Equality

The set of transformation rules ST is $\{Orient,\ SR,\ RRout,\ RRin, Merge\}$. *Orient*, replaces an equality edge, $v -_E w$, by a rewrite edge, $v \rightarrow_R w$, provided $Constant(v) \succ Constant(w)$. The ordering $\succ$ needs to be defined on constants in $\mathcal{K}$ not on terms over $\Sigma \cup \mathcal{K}$. The *SR* rule replaces one subterm edge by another one. In logical terms it represents the simplification of a subterm by rewriting, or in fact the simultaneous simplification of all occurrences of a subterm, if the graph presentation encodes full structure sharing for terms. The *RRout* (see figure 1[5]) and *RRin* rules each replace one rewrite edge by another. They correspond to certain equational inferences with the underlying rewrite rules (namely, critical pair computations and compositions, which for ground terms are also simplifications). The *RRin* rule is useful for efficiency reasons, though not mandatory. If the rule is applied exhaustively, the resulting rewrite system will be a right-reduced rewrite system over the extended signature. The *Merge* rule collapses two vertices that represent the same term over the extended signature into a single vertex. It ensures structure full sharing.

2.3 Extended Canonizers for the Theory of Lists *à la* Shostak

The set of transformation rules ST is $\{Orient,\ SR,\ RRout,\ RRin,\ Merge,\ CrdCons,\ CarCons,\ CardCdrCons\}$. *Orient*, *SR*, *RRout* and *RRin* rules are the same as in section 2.2. However, the ordering $\succ$ needs to be defined on $\mathcal{K} \cup \{cons, car, cdr\}$ such for $c \in \mathcal{K}$, $cons, car, cdr \succ c$. Rewriting and computing critical pairs in the presence of the axioms of the theory of lists *à la* Shostak requires extending the equalities of Γ [4]. These extension computations are implemented by the *CrdCons*, *CarCons* (see figure 1) and *CardCdrCons* rules each corresponding to one of the axioms of the theory of lists *à la* Shostak.



$(E_s,\ E_e,\ \{v_1 \rightarrow_R v_2,\ v_1 \rightarrow_R v_3\} \cup E_r,\ V,\ \mathcal{K},\ KC,\ C)$
$\rightarrow_{RRout}$
$(E_s,\ E_e,\ \{v_2 \rightarrow_R v_3,\ v_1 \rightarrow_R v_3\} \cup E_r,\ V,\ \mathcal{K},\ \{c_2 \succ c_3\} \cup KC,\ C)$

$(\{v_1 \rightarrow^1_S v_2,\ v_1 \rightarrow^2_S v_3\} \cup E_s,\ E_e,\ E_r,\ V,\ \mathcal{K},\ KC,\ C)$
$\rightarrow_{CarCons}$
$(\{v_1 \rightarrow^1_S v_2,\ v_1 \rightarrow^2_S v_3,\ v_4 \rightarrow^1_S v_1\} \cup E_s,\ \{v_4 -_E v_2\} \cup E_e,\ E_r,\ \{v_4\} \cup V,\ \{c_4\} \cup \mathcal{K},\ KC,\ update(C, v_4, (c_4, car)))$
if $Symbol(v_1) = cons$ -

Fig. 1. *RRout* and *CarCons* Graph Transformation Rules

2.4 Example

Figure 2a presents the construction of $DAG(\Gamma)$, where $\Gamma = \{car(c) \approx d,\ cdr(c) \approx e,\ cons(i,j) \approx k\}$ and $\mathcal{K} = \{c_1, \ldots, c_9\}$. Considering the ordering $\{c_5 \succ c_6, c_7 \succ c_8, c_{12} \succ c_9, c_{10} \succ c_1, c_{11} \succ c_2, c_4 \succ c_3\}$ we obtain the graph on figure 2b and the equivalent convergent rewrite system over the extended signature: $\{c_{12} \rightarrow c_9,\ c_5 \rightarrow c_6,\ c_7 \rightarrow c_8,\ c_{10} \rightarrow c_1,\ c_{11} \rightarrow c_2,\ c_4 \rightarrow c_3,\ cons(c_6, c_8) \rightarrow c_{12},\ car(c_9) \rightarrow c_5,\ cdr(c_9) \rightarrow c_7,\ car(c_3) \rightarrow c_{10},\ cdr(c_3) \rightarrow c_1,\ cons(c_1, c_2) \rightarrow c_4\}$. Using the graph we have: $ecan(\Gamma)(car(c)) = c_6$.

3 Conclusion and Discussion

We have presented a new graph-based method for building graph-based decision procedures and compute extended canonizers for the theory of equality and the theory of lists

[5] Because of lack of space we present only *RRout* and *CarCons* in this paper on figure 1.

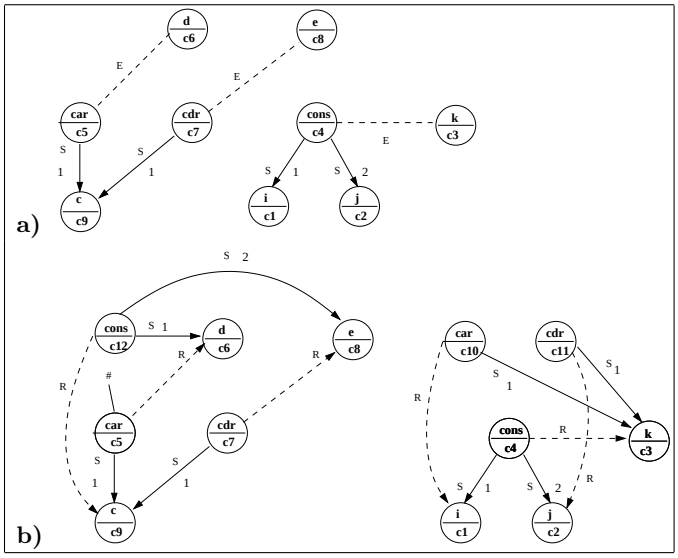


Fig. 2. a) Initial DAG for $\Gamma = \{car(c) \approx d,\ cdr(c) \approx e,\ cons(i,j) \approx k\}$, b) Saturated graph on the extended signature

à la Shostak. The method combines the key ideas of the rewriting-based approach for T-satisfiability and SER graphs. We believe that our approach allows for efficient implementations and a visual presentation that better illuminates the basic ideas underlying the construction of decision procedures for convex theories and the computations of extended canonizers. We plan to provide detailed complexity results in a next version of this paper and are in the process of implementing our method.

References

[1] Armando, A., Ranise, S., Rusinowitch, M.: A rewriting approach to satisfiability procedures. J. of Information and Computation **183(2)** (2003) 140–164.

[2] Baader, F., Nipkow, T.: Term rewriting and all that. Cambridge University Press (1998).

[3] Bachmair, L., Tiwari, A., Vigneron, L.: Abstract congruence closure. J. of Automated Reasoning, Kluwer Academic Publishers **31(2)** (2003) 129–168.

[4] Jouannaud, J. and Kirchner, H.: Completion of a set of rules modulo a set of equations. SIAM J. on Computing, **15(4)** (1986) 1155-1194.

[5] Knuth, D. E., Bendix, P. B.: Simple word problems in universal algebras. Computational Problems in Abstract Algebra, Pergamon Press, Oxford (1970) 263–297.

[6] Lynch, C., Strogova, P.: SOUR graphs for efficient completion. Journal of Discrete Mathematics and Theoretical Computer Science **2(1)** (1998) 1–25.

[7] Nelson, C. G., Oppen, D. C.: Fast Decision Procedures based on Congruence Closure. Journal of the ACM **27(2)** (1980) 356–364.

[8] Rusinowitch, M.: Theorem-proving with resolution and superposition. J. Symb. Comput. **series 11, 1-2** (Mar. 1991), 21-49.

[9] Ranise, S., Ringeissen, C., Tran, D.: Nelson-Oppen, Shostak and the Extended Canonizer: A Family Picture with a Newborn. Proceedings of the First International Colloquium of Theoretical Aspects of Computing. Z. Liu and K. Editors, LNCS, Springer-Verlag **3407** (2005) 372–386.

[10] Scharff, C., Bachmair, L.: On the Combination of Congruence Closure and Completion. Proceedings of the International Artificial Intelligence and Symbolic Computations Conference, B. Buchberger and J. A. Campbell Editors, LNCS, Springer-Verlag **3249** (2004) 103–117.

[11] Shostak, R.E.: Deciding Combinations of Theories. Journal of the ACM **31** (1984) 1–12.