Available online at www.sciencedirect.com
SCIENCE DIRECT®

ELSEVIER

Electronic Notes in Theoretical Computer Science 125 (2005) 69–85

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Justifying Equality[⋆]

Leonardo de Moura, Harald Rueß, and Natarajan Shankar

SRI International, Computer Science Laboratory,
333 Ravenswood Avenue, Menlo Park, CA 94025, USA
{*demoura,ruess,shankar*}*@csl.sri.com*

Abstract

We consider the problem of finding irredundant bases for inconsistent sets of equalities and disequalities. These are subsets of inconsistent sets which do not contain any literals which do not contribute to the unsatisfiability in an essential way, and can therefore be discarded. The approach we are pursuing here is to decorate derivations with proofs and to extract irredundant sets of assumptions from these proofs. This requires specialized operators on proofs, but the basic inference systems are otherwise left unchanged. In particular, we include justifying inference systems for union-find structures and abstract congruence closure, but our constructions can also be applied to other inference systems such as Gaussian elimination.

Keywords: Decision Procedures, Satisfiability, Union-Find, Congruence Closure, Proof Theory, Minimal Proofs.

1 Introduction

Constraint solving has many applications, including the discovery of abstraction predicates in protocol and software verification [4] and for lazy combinations for planning [13] and formal verification [13,2,6,5]. The effectiveness of these constraint solving problems depends on identifying “small” inconsistent subsets of constraints.

We therefore consider the problem of finding an *irredundant basis* for an inconsistent set Γ of equalities and disequalities. These are subsets of Γ which do not contain any redundant literal, that is, literals which do not contribute to the unsatisfiability of Γ in an essential way, and can therefore be discarded.

[⋆] Funded by SRI International, by NSF Grants CCR-0082560 and CCR-ITR-0325808, DARPA/AFRL Contract F33615-00-C-3043, and NASA Contract NAS1-20334

doi:10.1016/j.entcs.2004.06.068

This notion of irredundant basis is unrelated to the commonly used proof-theoretic measure which counts the number of inference steps. Also, irredundant bases are not necessarily minimal among all inconsistent subsets of Γ. For example, $\{x = z,\ y = z,\ x = y,\ x \neq y\}$ is inconsistent, $\{x = z,\ y = z,\ x \neq y\}$ is an irredundant basis for this inconsistency, but obviously it is not minimal as $\{x = y,\ x \neq y\}$ is also inconsistent. Indeed, the computation of minimally inconsistent bases is usually harder than the problem of computing irredundant bases. For example, an irredundant basis for an inconsistent conjunction of variable equalities and disequalities can be computed in $O(n\,log(n))$ time—with n the number of variables—whereas standard algorithms based on Boolean matrix multiplication for producing shortest deduction paths between two variables take $O(n^3)$ time.[1] Moreover, the problem of minimal bases for equality over uninterpreted functions is NP-hard [12].

Our starting point is the union-find structure used for deciding equality. A canonizer constructs canonical representatives for given terms with respect to the given equalities so that equality can be decided by syntactically comparing canonical forms. Such canonizer-based inference systems are attractive from an algorithmic point of view as equalities are applied in a directed way and term universes in combination procedures based on canonization are usually much smaller than the corresponding term universes in combination methods without such a canonizer. However, reduction to canonical forms accumulates many redundant literals into the assumptions of corresponding proofs.

Consider, for example, the inconsistent set $\{x = x',\ x' = z,\ y = x',\ u = f(x),\ u \neq v,\ v = f(y)\}$, where f is an uninterpreted function symbol and all variables are existentially-quantified (constants). In processing these literals from left to right, abstract congruence closure (ACC) [8,1,9], builds up a set of directed equalities $\{x \to z,\ x' \to z,\ y \to z\}$, with the left hand sides assumed to be larger than the right hand sides according to some given variable ordering. The variable arguments of the uninterpreted terms $f(x)$ and $f(y)$ are both replaced with their canonical representative z, and application of congruence yields the inconsistency. Since all input literals are used in this proof of unsatisfiability, simply tracking dependencies or collecting assumptions from an explicitly generated proof object is not sufficient for generating irredundant bases. The algorithm in [6] for computing irredundant bases by successively eliminating redundant literals has proven to be too costly in practice.

The approach we are pursuing here is to decorate derivations with proofs and to extract an irredundant set of assumptions from these proofs. This requires specialized operators on proofs, but the basic inference procedures

[1] But sub-cubic algorithms are possible for solving the related problem of finding successor vertices of shortest paths [7].

$$\frac{}{\vdash\ e_{s,t}\ :\ s = t} \qquad \frac{}{\vdash\ d_{s,t}\ :\ s \neq t}$$

$$\frac{}{\vdash\ \mathbf{1}_t\ :\ t = t} \qquad \frac{\vdash\ \rho\ :\ s = t}{\vdash\ \rho^{-1}\ :\ t = s} \qquad \frac{\vdash\ \tau\ :\ s = t,\ \vdash\ \sigma\ :\ t = r}{\vdash\ \tau;\sigma\ :\ s = r}$$

$$\frac{\vdash\ \rho_1\ :\ s_1 = t_1, \ldots, \vdash\ \rho_n\ :\ s_n = t_n}{\vdash\ cg_f(\rho_1, \ldots, \rho_n)\ :\ f(s_1, \ldots, s_n) = f(t_1, \ldots, t_n)}$$

$$\frac{\vdash\ \rho\ :\ t \neq t}{\vdash\ 0_t(\rho)\ :\ \bot}$$

Fig. 1. Proof theory for $\mathcal{U}$.

are left unchanged. Our main contribution is a *join operator* on directed equality proofs for $x = z$ and $y = z$ with x, y larger than z according to some given variable ordering. Obviously, taking the union of the assumptions of these two proofs leads to imprecision as there might be a join z' of x, y which is greater than z. However, the symmetric difference of assumptions yields irredundant bases for these kinds of "valley" proofs. We extend this basic insight to produce proofs with small sets of justifications for congruence closure.

2 Background

Given a signature Σ, a Σ-structure M maps each n-ary function symbol f in Σ to an n-ary map $M(f)$ over a suitable domain. We assume that M also maps free variables x to domain elements $M(x)$. The interpretation $M[\![t]\!]$ of a Σ-term t in a Σ-structure M is defined so that $M[\![x]\!] = M(x)$, and $M[\![f(t_1, \ldots, t_n)]\!] = M(f)(M[\![t_1]\!], \ldots, M[\![t_n]\!])$. A Σ-literal is either a Σ-equality or a Σ-disequality, and a Σ-equality $s = t$ (Σ-disequality $s \neq t$) is satisfied in Σ-structure M iff $M[\![s]\!] = M[\![t]\!]$ ($M[\![s]\!] \neq M[\![t]\!]$). The interpretation of the propositional connectives and quantifiers is standard [10]. When Σ-structure M satisfies a Σ-formula φ, we write $M \models \varphi$. A Σ-*theory* $\mathcal{T}$ is a class of Σ-structures—the *models* of the theory—closed under isomorphism. A set of Σ-literals L is $\mathcal{T}$-unsatisfiable if there is no Σ-structure M in $\mathcal{T}$ such that $M \models l$ for all $l \in L$. Literals L are $\mathcal{T}$-valid if for all Σ-structures M, $M \models l$ for all $l \in L$.

The proof theory for the theory $\mathcal{U}$ of equality over uninterpreted functions is included in Figure 1. Judgements are of the form $\vdash p : \varphi$ with p a proof of literal φ. The *assumptions* of proofs are obtained as follows.

$$
\begin{array}{ll}
Ax(e_{s,t}) = \{e_{s,t}\} & Ax(d_{s,t}) = \{d_{s,t}\} \\
Ax(\mathbf{1}_s) = \emptyset & Ax(\rho^{-1}) = Ax(\rho) \\
Ax(\tau;\, \sigma) = Ax(\tau) \cup Ax(\sigma) & Ax(cg_f(\rho_1, \ldots, \rho_n)) = Ax(\rho_1) \cup \ldots \cup Ax(\rho_n)
\end{array}
$$

The theory of *equality over variables* is the $\mathcal{U}$ theory for the empty signature Σ. A set Γ of Σ literals is $\mathcal{U}$-unsatisfiable if, and only if, there is a proof ρ of $\bot$ with $Ax(\rho) \subseteq \Gamma$. Proofs built up from reflexivity ($\mathbf{1}_t$), symmetry (ρ^{-1}), and transitivity ($\rho;\, \sigma$) are referred to as *equality-chaining proofs*. We also make use of a number of identities on proofs such as $(\mathbf{1}_t;\, \rho) = \rho$ and $cg_f(\mathbf{1}_{t_1}, \ldots, \mathbf{1}_{t_n}) = \mathbf{1}_{f(t_1,\ldots,t_n)}$.

Definition 2.1 Let Γ be a $\mathcal{T}$-unsatisfiable set of Σ-literals.

(i) If $\Delta \subseteq \Gamma$ and Δ is $\mathcal{T}$-unsatisfiable, then Δ is a $\mathcal{T}$-*basis* for Γ.

(ii) A $\mathcal{T}$-basis Δ of Γ is *minimal* if there is no $\mathcal{T}$-basis Δ' of Γ such that $|\Delta'| < |\Delta|$, where $|.|$ denotes the set-theoretic cardinality.

(iii) A $\mathcal{T}$-basis Δ of Γ is *irredundant* if no strict subset of Δ is a $\mathcal{T}$-basis for Γ.

Since $\Gamma \to \varphi$ is $\mathcal{T}$-valid if, and only if, $\Gamma \wedge \neg\varphi$ is $\mathcal{T}$-unsatisfiable, we also say that $\Theta \subseteq \Gamma$ is a $\mathcal{T}$-basis for φ if $\Theta \to \varphi$ if $\mathcal{T}$-valid. Moreover, ρ is an *irredundant proof* for φ if $Ax(\rho)$ is an irredundant basis for φ.

3 Equality over Variables

A set Γ of equalities and disequalities over variables is inconsistent if, and only if, there is a disequality $x \neq y$ in Γ such that x and y are in the same equivalence class of the equivalence closure of the equalities in Γ. Using the union-find algorithm, a partitioning of the variables in Γ is maintained in an incremental manner (see, for example, [3]). The operation $union(x = y)$ merges the two equivalence classes for x and y, and $find(x)$ returns the canonical representative of the equivalence class containing the variable x. A sequence of m *union* and *find* operations can be performed in worst-case time $O(m\,\alpha(n))$ with n the number of variables in Γ and $\alpha(n)$ the inverse of the Ackermann function [11].

We extend the union-find algorithm with an operator $explain(x = y)$ which returns an irredundant basis for the implied equality $x = y$, and analyze its complexity.

Definition 3.1 [Union-find-explain structure] Let $(\mathcal{V}, \prec)$ be a pair consisting of a nonempty, finite set $\mathcal{V}$ of variables with a total ordering $\prec$ on $\mathcal{V}$, and let E be the set of equalities over $\mathcal{V}$. Then, a *union-find-explain structure* is a pair of functions $(\phi : \mathcal{V} \to \mathcal{V}, \pi : \mathcal{V} \to 2^E)$ such that, for all $x \in \mathcal{V}$, $\phi(x) \prec x$ or $\phi(x) \equiv x$, and $\pi(x)$ is a basis for $x = \phi(x)$.

For a union-find-explain structure (ϕ, π), $\phi^*(x)$ denotes the *canonical* representative of the equivalence class for x, and $\pi^*(x)$ is a basis for $x = \phi^*(x)$.

Definition 3.2 Let (ϕ, π) be a union-find-explain structure; then:

$$\phi^*(x) := \begin{cases} x & , \phi(x) \equiv x \\ \phi^*(\phi(x)) & , \mathit{otherwise} \end{cases} \qquad \pi^*(x) := \begin{cases} \emptyset & , \phi(x) \equiv x \\ \pi(x) \uplus \pi^*(\phi(x)) & , \mathit{otherwise} \end{cases}$$

where $s_1 \uplus s_2 := (s_1 \cup s_2) \backslash (s_1 \cap s_2)$ denotes the symmetric difference of two sets.

Variable equalities are added incrementally to a union-find-explain structure by the *union* operation.

Definition 3.3 Let (ϕ, π) be a union-find-explain structure; then:

$$\mathit{union}_{(\phi,\pi)}(\rho, x, y) := \begin{cases} (\phi, \pi) & , x' \equiv y' \\ (\phi[x' := y'], \pi[x' := \Pi]) & , y' \prec x' \\ (\phi[y' := x'], \pi[y' := \Pi]) & , x' \prec y' \end{cases}$$
$$\text{with } x' \equiv \phi^*(x),\ \ y' \equiv \phi^*(y),\ \ \Pi = \pi^*(x) \cup \pi^*(y) \cup \{\rho\}.$$

We say that (ϕ, π) is a union-find-explain structure for a finite set E of variable equalities if (ϕ, π) is the result of processing the equalities in E, starting with $(\lambda x.\ x,\ \lambda x.\ \emptyset)$. In this case, $\phi^*(x) \equiv \phi^*(y)$ if, and only if, $E \to x = y$ is valid. In other words, a disequality $x \neq y$ is inconsistent with E if, and only if, $\phi^*(x) \equiv \phi^*(y)$. We show that an irredundant basis for implied equalities $x = y$ is obtained by computing the symmetric difference of $\pi^*(x)$ and $\pi^*(y)$.

Theorem 3.4 Let (ϕ, π) be a union-find-explain structure for a finite set of variable equalities; then (for all $x, y \in \mathcal{V}$)

(i) $\pi^*(x)$ is an irredundant basis for $x = \phi^*(x)$, and

(ii) if $\phi^*(x) \equiv \phi^*(y)$, then $\pi^*(x) \uplus \pi^*(y)$ is an irredundant basis for $x = y$.

Proof. By induction on the number of unions. These properties hold initially, since all nodes are distinct and for all x, $\phi(x) \equiv x$, and $\pi(x)$ is empty. For the induction step from n to $n+1$, we assume that these properties hold for (ϕ_n, π_n), $(\phi_{n+1}, \pi_{n+1}) = \mathit{union}_{(\phi_n,\pi_n)}(\rho, x, y)$, and, without loss of generality, $\phi^*(y) \prec \phi^*(x)$. Then, for any $\hat{x}$ in the equivalence class of x,

$$\begin{aligned}\pi^*_{n+1}(\hat{x}) &= \pi^*_n(\hat{x}) \uplus \pi_{n+1}(\phi^*_n(x)) \\ &= \pi^*_n(\hat{x}) \uplus (\pi^*_n(x) \cup \{\rho\} \cup \pi^*_n(y)) \\ &= (\pi^*_n(\hat{x}) \uplus \pi^*_n(x)) \cup \{\rho\} \cup \pi^*_n(y)\ ,\end{aligned}$$

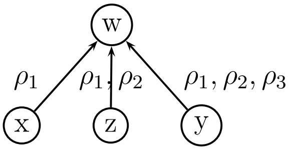
because the equivalence classes of x and y are disjoint, repeated assumptions can only occur within $\pi^*_n(\hat{x}) \uplus \pi^*_n(x)$, but we know by the induction hypothesis that this basis is irredundant.

For the second part, we need to show that for $\hat{x}$ and $\hat{y}$ from the equivalence classes of x and y, respectively, $\pi^*_{n+1}(\hat{x}) \uplus \pi^*_{n+1}(\hat{y})$ is irredundant. As we have already seen,

$$\begin{aligned}&\pi^*_{n+1}(\hat{x}) = (\pi^*_n(\hat{x}) \uplus \pi^*_n(x)) \cup \{\rho\} \cup \pi^*_n(y)\ , \text{ and} \\ &\pi^*_{n+1}(\hat{y}) = \pi^*_n(\hat{y})\ .\end{aligned}$$

Hence $\pi^*_{n+1}(\hat{x}) \uplus \pi^*_{n+1}(\hat{y}) = (\pi^*_n(\hat{x}) \uplus \pi^*_n(x)) \cup \{\rho\} \cup (\pi^*_n(\hat{y}) \uplus \pi^*_n(y))$ which we know is irredundant by the induction hypothesis. □

Example 3.5 Let $\Gamma := \{x \overset{\rho_1}{=} w,\ x \overset{\rho_2}{=} z,\ y \overset{\rho_3}{=} z\}$. Using the variable ordering $w \prec z \prec x \prec y$, processing the equalities from left to right yields the following representation of a union-find-explain structure, where "find" edges are labeled with the elements of the corresponding basis.



Clearly, the equality $\Gamma \rightarrow x = y$ is valid, since $\phi^*(x) \equiv \phi^*(y) \equiv w$. The irredundant basis $\{\rho_2, \rho_3\}$ for $x = y$ is obtained as $\pi^*(x) \uplus \pi^*(y) = \{\rho_1\} \uplus \{\rho_1, \rho_2, \rho_3\}$.

The bases thus obtained are irredundant by Theorem 3.4 but they are not necessarily minimal.

Example 3.6 Let $\Gamma := \{x \overset{\rho_1}{=} z,\ y \overset{\rho_2}{=} z,\ x \overset{\rho_3}{=} y\}$, then the minimal basis for $y = x$ is $\{\rho_3\}$, but the method described above returns the basis $\{\rho_1, \rho_2\}$, which is irredundant but not minimal, since *union* discards the implied equality $x = y$.

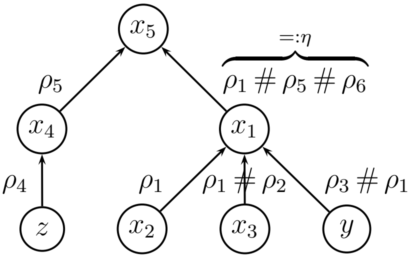
Figure 2 includes an efficient implementation of union-find-explain as an extension of the usual union-find algorithm (see, for example, [3]). Sets of assumptions are represented using the constructors # (join), 1 (reflexivity), and $e[s = t]$ (assumption). The corresponding set of assumptions $Ax(\rho)$ for such a representation ρ is obtained as follows.

$$\begin{aligned} Ax(1) &= \emptyset \\ Ax(e[s = t]) &= \{e[s = t]\} \\ Ax(\rho_1 \# \rho_2) &= Ax(\rho_1) \uplus Ax(\rho_2)\ . \end{aligned}$$

The find structure is implemented using the function p (parent) which is the identity on variables ($\lambda x.x$) initially, whereas the proof structure prf is initially 1 (reflexivity) on all inputs. $justify(x)$ is a straightforward implementation of π^*. Since $find(x)$ uses dynamic *path compression*, computations of π^* are memoized and $\#$-nodes may be structure-shared. $union(x, y)$ uses the *rank* structure for selecting the new canonical variable. Because of possible path compression in the initial finds, $prf(z)$ and $justify(z)$ coincide for input variables of *union*. We implicitly use the identities $\rho \# 1 = 1 \# \rho = \rho$.

$explain(x, y)$ computes the set $Ax(justify(x) \# justify(y))$ as required by Theorem 3.4. Notice that an equality is in the assumptions $Ax(p)$ of such a $\#$-dag p if, and only if, it occurs an odd number of times. We use reference counters $ref(\rho)$ for counting the number of occurrences of visited subdags. Subdags are visited in a breadth-first manner ignoring subdags with even reference counts. As a consequence, every 'node' in the $\#$-dag ρ is visited at most once in *collect*. Finally, the breadth-first traversal in *collect* is obtained by a FIFO queue with operators *enqueue*, *dequeue*, and *queue_is_empty*, which ensures that a node is dequeued only after its immediate parent nodes have been dequeued.

Example 3.7 Processing $\{x_1 \overset{\rho_1}{=} x_2,\ x_3 \overset{\rho_2}{=} x_2,\ y \overset{\rho_3}{=} x_2,\ z \overset{\rho_4}{=} x_4,\ x_4 \overset{\rho_5}{=} x_5,\ x_2 \overset{\rho_6}{=} x_4\}$ yields a union-find-explain structure



Thus, $explain(y, x_3) = collect((\rho_3 \# \rho_1) \# \eta \# (\rho_2 \# \rho_1) \# \eta) = \{\rho_2, \rho_3\}$ without visiting η, since this subdag occurs twice.

If $h(n)$ denotes the maximum height of the find structure after n *union* and $find$ operations, then clearly $find$ takes at most $O(h(n))$ time and creates at most $O(h(n))$ new $\#$-nodes, *union* takes at most $O(h(n))$ time and space. So, a sequence of n *union* operations takes at most $O(n\, h(n))$ time and space.

```
$p \leftarrow (\lambda x.\ x); prf \leftarrow (\lambda \rho.\ 1); rank \leftarrow (\lambda x.\ 0); result \leftarrow \bot; ref \leftarrow (\lambda \rho.\ \bot)$

$find(x)\ =$
 if not$(x \equiv p(x))$ then
    $p(x) \leftarrow find(p(x));\ prf(x) \leftarrow (prf(p(x)) \# prf(x));$
 return $p(x)$

$union(x, y)\ =$
 $x' \leftarrow find(x);\ y' \leftarrow find(y);$
 if $rank(x') > rank(y')$ then
    $p(y') \leftarrow x';\ prf(y') \leftarrow (prf(x) \# prf(y) \# e[x = y])$
 else
    $p(x') \leftarrow y';\ prf(x') \leftarrow (prf(x) \# prf(y) \# e[x = y]);$
    if $rank(x') \equiv rank(y')$ then $rank(y') \leftarrow rank(y') + 1$

$explain(x, y)\ =$
 $result \leftarrow \emptyset;\ ref \leftarrow (\lambda \rho.\ 0);$
 $collect(justify(x) \# justify(y));$
 return $result$

$justify(x)\ =$
 if $x \equiv p(x)$ then 1 else $(prf(x) \# justify(p(x)))$

$collect(\rho)\ =$
 $register(\rho);$
 while not$(queue\_is\_empty())$ do
    $\tau \leftarrow dequeue();$
    if $is\_odd(ref(\tau))$ then
       if $\tau \equiv \rho_1 \# \rho_2$ then $register(\rho_1);\ register(\rho_2)$
       else if $\tau \equiv e[x = y]$ then $result \leftarrow result \cup \{e[x = y]\}$


$register(\rho)\ =$
 if $ref(\rho) \equiv 0$ then $enqueue(\rho);\ ref(\rho) \leftarrow 1$ else $ref(\rho) \leftarrow ref(\rho) + 1$
```

Fig. 2. Implementation of union-find-explain.

The *explain* is linear in the number of $\#$-nodes in $justify(x) \# justify(y)$, so it takes at most $O(n\, h(n))$ time. Since the algorithm is using the weighted-union heuristic based on the *rank* structure, $h(n)$ is bounded by $log(n)$. In fact, the run time of *explain* is bounded by $O(n\, \alpha(n))$, since the core union-find algorithm has this complexity and any (recursive) invocation of $find$ and $union$ produces only a constant number of $\#$ nodes.

4 Equality over Uninterpreted Functions

We consider the problem of inferring small justifications for problems in the theory $\mathcal{U}$ of equalities over uninterpreted functions. The starting point is abstract congruence closure (ACC) as defined by Kapur [8] and Bachmair and Tiwari [1]. ACC incrementally processes a finite set E of $\mathcal{U}$-equalities into an equivalent configuration V, U with V a union-find-explain structure and U a set of directed, flat equalities of the form $x = f(x_1, \ldots, x_n)$ with x, x_i variables. Irreducible configurations are congruence-closed in the sense that V implies $x = y$ if U contains $x = f(x_1, \ldots, x_n)$ and $y = f(x_1, \ldots, x_n)$. The length of any maximal derivation using the inference rules for constructing an ACC is at most quadratic in the input size. In [9] we define a canonizer $can_{(V,U)}(t)$ on irreducible configurations (V, U) and terms t for solving uniform word problems in $\mathcal{U}$; that is, $can_{(V,U)}(t_1) \equiv can_{(V,U)}(t_2)$ if, and only if, $E \to t_1 = t_2$ is $\mathcal{U}$-valid. In other words, a disequality $t_1 \neq t_2$ is $\mathcal{U}$-inconsistent with E if, and only if, $can_{(V,U)}(t_1) \equiv can_{(V,U)}(t_2)$.

Justifying congruence closure is based on the ACC procedure in [9], and the results of Section 3 for generating irredundant bases using union-find-explain are reused below. In contrast to the developments in Section 3, however, justifications are given in terms of proof terms (see Figure 1) instead of sets of assumptions. This use of proof terms suggests various optimizations.

Example 4.1 Consider the following equality chains.

$$\begin{array}{l} f_1(x_1) \overset{\tau_1}{=} x_1 \overset{\rho_1}{=} x_2 \quad \overset{\sigma_1}{=} f_1(x_{n+1}) \ , \\ \ldots \ , \\ f_n(x_1) \overset{\tau_n}{=} x_n \overset{\rho_n}{=} x_{n+1} \overset{\sigma_n}{=} f_n(x_{n+1}) \end{array}$$

With these equalities one obtains proofs $(i = 1, \ldots, n)$

$$\begin{array}{l} \pi_1^i := \tau_i;\ \rho_i;\ \sigma_i \\ \pi_2^i := cg_{f_i}(\rho_1;\ \ldots;\ \rho_n) \end{array}$$

for $f_i(x_1) = f_i(x_{n+1})$. These two proofs are essentially different in that the set of assumptions $Ax(\pi_1^i)$ and $Ax(\pi_2^i)$ are incomparable with respect to set inclusion. There are 2^n different proofs for the equality

$$g(f_1(x_1), ..., f_n(x_1)) = g(f_1(x_{n+1}), ..., f_n(x_{n+1}))$$

depending on whether π_1^i or π_2^i is chosen for establishing equality between the

ith argument terms, but the only irredundant proof is $cg_g(\pi_2^1, \ldots, \pi_2^n)$.

This example shows that the generation of irredundant proofs for equalities in $\mathcal{U}$ is expensive in general. Therefore, we introduce a weaker criterion that only requires irredundancy for *equality chaining subproofs*, that is, subproofs built up entirely from reflexivity, symmetry, and transitivity. The operator *unchain* transforms proofs to a set of *non-equality-chaining* subproofs. This operator is then used for defining a localized irredundancy criterion for proofs.

Definition 4.2 [Chaining]

$$\begin{aligned} unchain(\rho_1;\, \rho_2) &= unchain(\rho_1) \cup unchain(\rho_2) \\ unchain(\rho^{-1}) &= unchain(\rho) \\ unchain(\mathbf{1}_t) &= \emptyset \\ unchain(e_{s,t}) &= \{e_{s,t}\} \\ unchain(cg_f(\rho_1, \ldots, \rho_n)) &= \{cg_f(\rho_1, \ldots, \rho_n)\} \end{aligned}$$

If $\rho_1 \vdash t_1 = t_2$ and $\rho_2 \vdash t_2 = t_3$, then $unchain(\rho_1) \uplus unchain(\rho_2)$ is an irredundant basis (Definition 2.1) for $t_1 = t_3$. The unchaining operator is used for defining the notion of local irredundancy for proofs. Intuitively, a proof is locally irredundant if all pure equality chaining subproofs are irredundant.

Definition 4.3 [Locally Irredundant Proofs]
A proof ρ for $s = t$ is *locally irredundant* if

(i) $\rho \equiv \mathbf{1}_s$ or $\rho \equiv e_{s,t}$ or

(ii) $\rho \equiv cg_f(\rho_1, \ldots, \rho_n)$, and $\rho_1, \ldots, \rho_n$ are locally irredundant, or

(iii) $\rho \equiv \rho_1;\, \rho_2$ or $\rho \equiv {\rho_1}^{-1}$, and there is no proof ρ' for $s = t$, such that $unchain(\rho') \subset unchain(\rho)$, and for all $\tau \in unchain(\rho)$, τ is locally irredundant.

A locally irredundant proof is not necessarily irredundant.

Example 4.4 Reconsider Example 4.1. The proof $cg_g(\pi_1^1, \pi_2^2, \ldots, \pi_2^n)$, for example, is locally irredundant but it is not irredundant, since the proof $cg_g(\pi_2^1, \ldots, \pi_2^n)$ has a strictly smaller set of assumptions.

Example 4.5 Consider the judgement $cg_f(e_{x,y}, e_{y,z});\ cg_f({e_{x,y}}^{-1}, \mathbf{1}_z) \vdash f(x, y) = f(x, z)$. This proof is locally irredundant but it is not irredundant, since $e_{y,z}$ already justifies $f(x, y) = f(x, z)$ using the proof $cg_f(\mathbf{1}_x, e_{y,z})$.

The operator $chain_{s,t}$ simply converts a set of non-equality-chaining proofs for $s = t$, as obtained from the unchaining operator, into an equality chaining proof for $s = t$.

$$\pi^*(x) = \begin{cases} \mathbf{1}_x & , x \equiv \phi(x) \\ \pi(x) \downarrow \pi^*(\phi(x)) & , \mathit{otherwise} \end{cases}$$

$$\mathit{union}_{(\phi,\pi)}(\rho, x, y) = \begin{cases} (\phi, \pi) & , x' \equiv y' \\ (\phi[x' := y'], \pi[x' := (\pi^*(x)^{-1};\ \rho;\ \pi^*(y))]) & , y' \prec x' \\ (\phi[y' := x'], \pi[y' := (\pi^*(y)^{-1};\ \rho;\ \pi^*(x))]) & , x' \prec y' \end{cases}$$
$$\text{where } x' \equiv \phi^*(x),\ \ y' \equiv \phi^*(y)\ .$$

Fig. 3. Union-find-explain with proofs.

Definition 4.6 Let Π be a set of non-equality chaining proofs as obtained from $\mathit{unchain}(\rho)$ for $\rho \vdash s = t$; then:

$$\mathit{chain}_{s,t}(\Pi) = \begin{cases} \mathbf{1}_t & , \Pi = \emptyset \\ \rho;\ \mathit{chain}_{r,t}(\Pi - \{\rho\}) & , \rho \in \Pi \wedge \rho \vdash s = r \\ \rho^{-1};\ \mathit{chain}_{r,t}(\Pi - \{\rho\}) & , \rho \in \Pi \wedge \rho \vdash r = s \end{cases}$$

Obviously, $\mathit{chain}_{s,t}(\Pi) \vdash s = t$. Equality-chaining proofs are transformed by the proof transformer $\rho_1 \downarrow \rho_2$ in order to eliminate redundancies.

Definition 4.7 For $\rho_1 \vdash s_1 = t$ and $\rho_2 \vdash s_2 = t$, define the *join* of ρ_1 and ρ_2 as

$$\rho_1 \downarrow \rho_2 := \mathit{chain}_{s_1,s_2}(\mathit{unchain}(\rho_1) \uplus \mathit{unchain}(\rho_2)).$$

Notice that $(\rho_1 \downarrow \rho_2) \vdash s_1 = s_2$ and $Ax(\rho_1 \downarrow \rho_2) \subseteq (Ax(\rho_1) \cup Ax(\rho_2))$. By a slight abuse of notation, we will just write $\rho_1 \downarrow \rho_2$ for $\rho_1 \downarrow {\rho_2}^{-1}$, ${\rho_1}^{-1} \downarrow \rho_2$, or ${\rho_1}^{-1} \downarrow {\rho_2}^{-1}$ in the following.

The justifying ACC includes a union-find-explain structure (ϕ, π). Whereas, ϕ is identical to the one in Definition 3.1, the π component is now a function from variables to equality proof terms. The π^* and *union* operations on variables in Section 3 are adjusted to include proofs instead of sets of assumptions (Figure 3).

Definition 4.8 A union-find-explain structure (ϕ, π) is *locally irredundant* if, and only if, for all x and y with $\phi^*(x) \equiv \phi^*(y)$, the proof $\pi^*(x) \downarrow \pi^*(y)$ is locally irredundant.

By replacing symmetric difference on sets of assumptions by the join operator, irredundant proofs for variable equalities are obtained as in Theorem 3.4.

Lemma 4.9 Let (ϕ, π) be a union-find-explain structure with proofs for a set of equalities (see Figure 3), then:

(i) $\pi^*(x)$ is a locally irredundant proof for $x = \phi^*(x)$, and

(ii) (ϕ, π) is locally irredundant.

Proof. Similar to Theorem 3.4. □

Example 4.10 For the literals in Example 3.5 one obtains the union-find-explain structure (ϕ, π) with

$$\begin{aligned}&\phi := \{x \mapsto w, z \mapsto w, y \mapsto w\}\\&\pi := \{x \mapsto e_{x,w},\ z \mapsto ({e_{x,z}}^{-1};\, e_{x,w}),\ y \mapsto (e_{y,z};\, {e_{x,z}}^{-1};\, e_{x,w})\}\end{aligned}$$

and a proof of $x = y$ is $e_{x,z};\, {e_{y,z}}^{-1}$ because

$$\begin{aligned}&(e_{x,w}) \downarrow (e_{y,z};\, {e_{x,z}}^{-1};\, e_{x,w})\\&= \mathit{chain}_{x,y}(\mathit{unchain}(e_{x,w}) \uplus \mathit{unchain}(e_{y,z};\, {e_{x,z}}^{-1};\, e_{x,w}))\\&= \mathit{chain}_{x,y}(\{e_{x,w}\} \uplus \{e_{y,z}, e_{x,z}, e_{x,w}\})\\&= \mathit{chain}_{x,y}(\{e_{y,z}, e_{x,z}\})\\&= e_{x,z};\, {e_{y,z}}^{-1}.\end{aligned}$$

Configurations of justifying ACC include (V, U) with V a union-find-explain structure and U a renaming context for representing a finite set of equalities $u = f(x_1, \ldots, x_n)$.

Definition 4.11 A *renaming context* is a finite map of bindings of the form $u \mapsto f(x_1 : \rho_1, \ldots, x_n : \rho_n)$ with $u, x_1, \ldots, x_n \in \mathcal{V}$ and $\rho_i \vdash t_i = x_i$ for terms t_i $(i = 1, \ldots, n)$.

Definition 4.12 A pair (V, U) consisting of a union-find-explain structure and a renaming context is *locally irredundant* if

(i) V is a locally irredundant, and

(ii) for every $u \mapsto f(x_1 : \rho_1, \ldots, x_n : \rho_n)$ in U, $\rho_1, \ldots, \rho_n$ are locally irredundant proofs.

Abstract congruence closure (ACC) flattens input terms by introducing fresh renaming variables for nested flat subterms. The initial step in processing an equality or disequality in the ACC procedure compiles terms into variables by iteratively replacing flat subterms $f(x_1, \ldots, x_n)$ with a renaming variable from a possibly extended renaming context. The following canonizer includes flattening and always returns a variable which is equal to the argument term in a possibly extended renaming context.

Definition 4.13 For a union-find-explain structure $V = (\phi, \pi)$ and a justifying renaming context U, define:

$$can_{(V,U)}(x) = (\phi^*(x) : \pi^*(x),\ U)$$

$$can_{(V,U)}(f(t_1, \ldots, t_n)) = \begin{cases} (\phi^*(u) : cg_f(\rho_1 \downarrow \tau_1, \ldots, \rho_n \downarrow \tau_n);\ \pi^*(u), U_n) \\ \qquad \text{if } (u \mapsto f(x_1 : \tau_1, \ldots, x_n : \tau_n)) \in U_n \\ (v : \mathbf{1}_v,\ \{v \mapsto f(x_1 : \rho_1, \ldots, x_n : \rho_n)\} \cup U_n) \\ \qquad \text{otherwise (with } v \text{ fresh)} \end{cases}$$

$$\begin{aligned} \text{where} \quad (x_1 : \rho_1,\ U_1) &= can_{(V,U)}(t_1), \\ (x_2 : \rho_2,\ U_2) &= can_{(V,U_1)}(t_2), \ldots, \\ (x_n : \rho_n,\ U_n) &= can_{(V,U_{n-1})}(t_n)\ . \end{aligned}$$

$can_{(V,U)}(t)$ is a *canonizer* in the sense that the $x \equiv y$ for $(x, \_) = can_{(V,U)}(t_1)$ and $(y, \_) = can_{(V,U)}(t_2)$ if, and only if, $t_1 = t_2$ is validated by the equalities of (V, U) [9]. Moreover, this canonizer returns locally irredundant proofs for the equality of its source and target term.

Lemma 4.14 If (V, U) is locally irredundant, and $can_{(V,U)}(t) = (x : \rho, U')$, then

(i) (V, U') is also locally irredundant, and

(ii) ρ is a locally irredundant proof for $t = x$.

Proof. By induction on the structure of the term t, and Lemma 4.9. □

A justifying ACC inference system for processing equalities and disequalities over uninterpreted functions is described in Figure 4. *Configurations* (Γ, D, V, U) of this inference system consist of a finite set Γ of assumptions of the form $e_{s,t}$ and $d_{s,t}$, a finite set D of variable disequalities $d_{s,t} \mapsto (x : \rho_1) \neq (y : \rho_2)$ with $\rho_1 \vdash s = x$ and $\rho_2 \vdash t = y$, a union-find-explain structure $V = (\phi, \pi)$, and a renaming context U.

Definition 4.15 A configuration (Γ, D, V, U) is *locally irredundant* if V, U is locally irredundant according to Definition 4.12, and for every $d_{s,t} \mapsto (x : \rho_1) \neq (y : \rho_2)$ in D, ρ_1 and ρ_2 are locally irredundant.

The **eq** rule in Figure 4 processes input equalities $s = t$ by merging the corresponding decorated variables in V, and, similarly, **diseq** processes disequalities $s \neq t$. Disequalities $x \neq x$ are reduced to the unsatisfiable $\perp$ using the **bot** rule, whereas **cong** deduces variable equalities $u = v$ from $u = t$ and $v = t$ with t a flat term, and **uprop** and **dprop** propagate variable equalities into renaming contexts U and disequalities D, respectively. Here, **dprop** is applied symmetrically.

eq

$$\frac{(\{e_{s,t}\} \cup \Gamma,\ V,\ D,\ U)}{(\Gamma,\ \mathit{union}_V((\tau^{-1};\, e_{s,t};\, \sigma), x, y),\ D,\ U'')}$$

where $(x : \tau,\ U') = \mathit{can}_{(V,U)}(s)$ and $(y : \sigma,\ U'') = \mathit{can}_{(V,U')}(t)$

diseq

$$\frac{(\{d_{s,t}\} \cup \Gamma,\ V,\ D,\ U)}{(\Gamma,\ V,\ \{d_{s,t} \mapsto (x : \tau) \neq (y : \sigma)\} \cup D,\ U'')}$$

where $(x : \tau,\ U') = \mathit{can}_{(V,U)}(s)$, and $(y : \sigma,\ U'') = \mathit{can}_{(V,U')}(t)$

bot

$$\frac{(\Gamma,\ V,\ \{d_{s,t} \mapsto (x : \tau) \neq (x : \sigma)\} \cup D,\ U)}{\mathit{unsat}(d_{s,t}, \tau \downarrow \sigma)}$$

uprop

$$\frac{(\Gamma,\ (\phi,\pi),\ D,\ \{u \mapsto f(x_1 : \rho_1, \ldots, x_i : \rho_i, \ldots, x_n : \rho_n)\} \cup U)}{(\Gamma,\ (\phi,\pi),\ D,\ \{u \mapsto f(x_1 : \rho_1, \ldots, y : \rho_i \downarrow \tau, \ldots, x_n : \rho_n)\} \cup U)}$$

where $\phi^*(x_i) \equiv y$ and $\pi^*(x_i) = \tau$

dprop

$$\frac{(\Gamma,\ (\phi,\pi),\ \{d_{s,t} \mapsto (x : \sigma_1) \neq (z : \sigma_2)\} \cup D,\ U)}{(\Gamma,\ (\phi,\pi),\ \{d_{s,t} \mapsto (y : \tau \downarrow \sigma_1) \neq (z : \sigma_2)\} \cup D,\ U)}$$

where $\phi^*(x) \equiv y$ and $\pi^*(x) = \tau$

cong

$$\frac{(\Gamma,\ V,\ D,\ \left\{\begin{array}{l} u \mapsto f(x_1 : \rho_1, , \ldots, x_n : \rho_n), \\ v \mapsto f(x_1 : \tau_1, , \ldots, x_n : \tau_n) \end{array}\right\} \cup U)}{(\Gamma,\ V',\ D,\ \{u \mapsto f(x_1 : \rho_1, , \ldots, x_n : \rho_n)\} \cup U)}$$

where $V' = \mathit{union}_V(\mathit{cg}_f(\rho_1 \downarrow \tau_1, \ldots, \rho_n \downarrow \tau_n), u, v)$

Fig. 4. Abstract congruence closure with proofs.

For termination, soundness and completion of undecorated versions of this abstract congruence closure procedure see, for example, [1,9].

Theorem 4.16 Let Γ be an unsatisfiable, finite set of $\mathcal{U}$-equalities and $\mathcal{U}$-disequalities, and let $\mathit{unsat}(d_{s,t}, \rho)$ be an irreducible configuration with respect to the justifying ACC inference system in Figure 4 and starting configuration $(\Gamma, (\lambda x.x, \lambda x.\mathbf{1}_x), \emptyset, \emptyset)$, then ρ is a locally irredundant proof.

Proof. Clearly the initial configuration is locally irredundant. By Lemmas 4.9 and 4.14, all rules but **bot** preserve local irredundancy. The proof $\rho \equiv \tau \downarrow \sigma$ obtained by the application of the **bot** rule is locally irredundant, because τ and σ are locally irredundant proofs for $s = x$ and $t = x$ (Lemma 4.9). □

Example 4.17 Processing $\{x_1 \overset{\rho_1}{\doteq} f(x_2),\ x_3 \overset{\rho_2}{\doteq} f(x_4),\ x_5 \overset{\rho_3}{\doteq} x_6,\ x_2 \overset{\rho_4}{\doteq} x_5,\ x_4 \overset{\rho_5}{\doteq}$

$x_5\}$ from the left to right yields the final configuration

$$(\emptyset, \left\{ \begin{array}{lll} x_1 \xrightarrow{\rho_1} u_1, & x_3 \xrightarrow{\rho_2} u_2, & x_5 \xrightarrow{\rho_3} x_6 \\ x_2 \xrightarrow{\rho_4;\, \rho_3} x_6, & x_4 \xrightarrow{\rho_5;\, \rho_3} x_6, & u_1 \xrightarrow{cg_f(\rho_4;\, {\rho_5}^{-1})} u_2 \end{array} \right\}, \emptyset, \{u_2 \mapsto f(x_6 : \rho_5;\, \rho_3)\})$$

First, canonization of x_1 yields $(x_1 : \mathbf{1}_{x_1})$ and canonization of $f(x_2)$ yields $(u_1 : \mathbf{1}_{u_1})$, where u_1 is a fresh variable and $u_1 \mapsto f(x_2 : \mathbf{1}_{x_2})$ is the corresponding renaming. Using rule **eq**, variables x_1 and u_1 are merged with proof $({\mathbf{1}_{x_1}}^{-1};\, \rho_1;\, \mathbf{1}_{u_1}) = \rho_1$. Second, when processing $x_3 = f(x_4)$, x_3 canonizes to $(x_3 : \mathbf{1}_{x_3})$ and $f(x_4)$ canonizes to $(u_2 : \mathbf{1}_{u_2})$, where u_2 is fresh variable and $u_2 \mapsto f(x_4 : \mathbf{1}_{x_4})$ is the corresponding renaming. Thus, x_3 and u_2 are merged with proof ρ_2 (rule **eq**). Third, x_5 and x_6 are canonized to $(x_5 : \mathbf{1}_{x_5})$ and $(x_6 : \mathbf{1}_{x_6})$, respectively. Using rule **eq**, variables x_5 and x_6 are merged with proof ρ_3. Forth, x_2 and x_5 are canonized to $(x_2 : \mathbf{1}_{x_2})$ and $(x_6 : \rho_3)$, respectively. Using rule **eq**, variables x_2 and x_6 are merged with proof $\rho_4;\, \rho_3$. This variable equality is propagated (rule **uprop**) to obtain the instantiated renaming $u_1 \mapsto f(x_6 : \rho_4;\, \rho_3)$. Finally, x_4 and x_5 are canonized to $(x_4 : \mathbf{1}_{x_4})$ and $(x_6 : \rho_3)$, respectively. Using rule **eq**, variables x_4 and x_6 are merged with proof $\rho_5;\, \rho_3$. This variable equality is propagated (rule **uprop**) to obtain the instantiated renaming $u_2 \mapsto f(x_6 : \rho_5;\, \rho_3)$. Using rule **cong**, variables u_1 and u_2 are merged with proof $cg_f((\rho_4;\, \rho_3) \downarrow (\rho_5;\, \rho_3)) = cg_f(\rho_4;\, {\rho_5}^{-1})$, and the renaming $u_1 \mapsto f(x_6 : \rho_4;\, \rho_3)$ is removed from U.

Now, consider the implied equality $f(x_2) = f(x_4)$. Its left hand side $f(x_2)$ canonizes to $(u_2 : cg_f(\rho_4;\, {\rho_5}^{-1}))$, whereas $f(x_4)$ canonizes to $(u_2 : cg_f((\rho_5;\, \rho_3) \downarrow (\rho_5;\, \rho_3))) = (u_2 : \mathbf{1}_{x_4})$. Thus $(cg_f(\rho_4;\, {\rho_5}^{-1})) \downarrow \mathbf{1}_{x_4} = cg_f(\rho_4;\, {\rho_5}^{-1})$ is a locally irredundant proof for $f(x_2) = f(x_4)$. This proof also happens to be minimal. Now, consider the implied equality $f(x_2) = f(x_6)$. The right hand side $f(x_6)$ canonizes to $(u_2 : cg_f(\rho_5;\, \rho_3))$. Thus $cg_f(\rho_4;\, {\rho_5}^{-1});\, cg_f(\rho_5;\, \rho_3)$ is a locally irredundant proof for $f(x_2) = f(x_6)$, but it is not irredundant. However, this proof can be simplied using the following identity on proofs.

$$(cg_f(\tau_1, \ldots, \tau_n); cg_f(\sigma_1, \ldots, \sigma_n)) \;=\; cg_f(\tau_1 \downarrow \sigma_1, \ldots, \tau_n \downarrow \sigma_n)$$

Thus we obtain the irredundant proof $cg_f((\rho_4;\, {\rho_5}^{-1}) \downarrow (\rho_5;\, \rho_3)) = cg_f(\rho_4;\, \rho_3)$ for $f(x_2) = f(x_6)$.

Lemma 4.18

$$Ax(cg_f(\tau_1 \downarrow \sigma_1, \ldots, \tau_n \downarrow \sigma_n)) \subseteq Ax(cg_f(\tau_1, \ldots, \tau_n); cg_f(\sigma_1, \ldots, \sigma_n))$$

Proof. Using $Ax(\rho_1 \downarrow \rho_2) \subseteq (Ax(\rho_1) \cup Ax(\rho_2))$. □

In general justifying ACC does not yield irredundant proofs or even minimal proofs (Example 4.1), but irredundant proofs may be obtained by replacing equality-chaining subproofs σ with congruence subproofs τ whenever $Ax(\tau) \subseteq Ax(\sigma)$. For example, the proof $cg_f(\sigma)$; $cg_f(\tau)$ might be replaced by $cg_f(\sigma \downarrow \tau)$. In cases $Ax(\sigma)$ and $Ax(\tau)$ are incomparable with respect to set inclusion (as is the case in Example 4.1), however, it is unclear which proof should be used, since the proof transformation may now depend on the structure of the complete proof. Therefore, it seems too expensive, in practice, to produce (globally) irredundant proofs, and, arguably, maintaining locally irredundant proofs is a good compromise between the conflicting goals of conciseness of justifications and associated computational costs.

5 Conclusions

We have presented systems for proving irredundant proofs for variable equality and extended this proof-producing system to obtain "small", that is, locally irredundant proofs, for abstract congruence closure. The main characteristics of our proof-producing extensions is that they do not change the underlying algorithms and therefore the algorithmic advantage of canonization is maintained. Although it is possible to maintain fully irredundant proofs for congruence closure and other theories, this may be prohibitively expensive in practice as search is involved.

Our approach can be extended to also work for other inference systems such as Gaussian elimination for the linear arithmetic equality theory. Polynomials are decorated with proofs to obtain $(q_0 + q_1 x_1 : \sigma_1 + \ldots + q_n x_n : \sigma_n) : \rho$. Such a decoration represents a proof $(\rho;\ cg(\sigma_1, \ldots, \sigma_n))$ for $p' = q_0 + q_1 x_1 + \ldots + q_n x_n$, for some source polynomial p'. Our justifying equality framework can also be applied to obtain small proofs for a Shostak combination [9].

References

[1] L. Bachmair and A. Tiwari. Abstract congruence closure and specializations. In D. McAllester, editor, *Automated Deduction—CADE-17*, volume 1831 of *LNAI*, pages 64–78. Springer, June 2000.

[2] C.W. Barrett, D.L. Dill, and A. Stump. Checking satisfiability of first-order formulas by incremental translation to SAT. In *Computer-Aided Verification, CAV '2002*, volume 2404 of *LNCS*, pages 236–249. Springer, July 2002.

[3] T.H. Cormen, C.E. Leiserson, and R.L. Rivest. *Introduction to Algorithms*. The MIT Press, Cambridge, MA, 1990.

[4] S. Das and D.L. Dill. Counter-example based predicate discovery in predicate abstraction. In *Formal Methods in Computer-Aided Design*, LNCS, pages 19–32. Springer, November 2002.

[5] L. de Moura and H. Rueß. Lemmas on demand for satisfiability solvers. In *Fifth International Symposium on the Theory and Applications of Satisfiability Testing (SAT'02)*, May 2002. Extended version at http://www.csl.sri.com/users/ruess/papers/SAT2002/index.html.

[6] L. de Moura, H. Rueß, and M. Sorea. Lazy theorem proving for bounded model checking over infinite domains. In A. Voronkov, editor, *International Conference on Automated Deduction (CADE'02)*, volume 2392 of *LNCS*, pages 438–455. Springer, July 2002.

[7] Z. Galil and O. Margalit. Witnesses for boolean matrix multiplication and for transitive closure. *Journal of Complexity*, 9:201–221, 1993.

[8] D. Kapur. Shostak's congruence closure as completion. In H. Comon, editor, *Rewriting Techniques and Applications, RTA 1997*, volume 1103 of *LNCS*, pages 23–37. Springer, July 1997.

[9] N. Shankar and H. Rueß. Combining Shostak theories. In S. Tison, editor, *RTA'02*, volume 2378 of *LNCS*, pages 1–18. Springer, 2002.

[10] J. R. Shoenfield. *Mathematical Logic*. Addison-Wesley, Reading, MA, 1967.

[11] R.E. Tarjan. Efficiency of a good but not linear set union algorithm. *Journal of the ACM*, pages 215–225, 1975.

[12] A. Tiwari. Personal communication. June 2004.

[13] S.A. Wolfman and D.S. Weld. The LPSAT engine: Its application to resource planning. In *IJCAI*, pages 310–317, 1999.