Extending Decidable Clause Classes via Constraints

Reinhard Pichler *
Technische Universität Wien

Abstract

There are several well known possibilities which constrained clauses (= c-clauses, for short) provide in addition to standard clauses. In particular, many (even infinitely many) standard clauses can be represented by a single c-clause. Hence, many parallel inference steps on standard clauses can be encoded in a single inference step on c-clauses.

The aim of this work is to investigate another possibility offered by constrained clauses: We shall try to combine resolution based decision procedures with constrained clause logic in order to increase the expressive power of the resulting decision classes. Therefore, there are two questions on which this paper focusses:

1. In what sense do constrained clauses actually provide additional expressive power in comparison with standard clauses? The answer given here is that only constraints made up from conjunctions of disequations constitute a genuine extension w.r.t. standard clauses.
2. Is it possible to extend decision classes of standard clauses by the use of constrained clauses? The main result of this work is a positive answer to this question, namely a theorem which shows that standard clause classes decidable by certain resolution refinements remain decidable even if they are extended by constraints consisting of conjunctions of disequations.

In order to prove the termination of our decision procedures on constrained clauses, some kind of compactness theorem for unification problems will be derived, thus extending a related result from [LMM 86].

1 Introduction

The usefulness of constrained clauses (*c-clauses*, for short) in automated deduction is generally acknowledged: Constraints allow the representation of (possibly infinitely) many standard clauses in a single c-clause. Hence, a single inference step on c-clauses may in fact represent many analogous inference steps on standard clauses carried out in parallel. Furthermore, c-clauses play an important role in automated model building, since they allow a finite representation of (Herbrand) models which would require an infinite representation in standard clause logic (cf. [CZ 91]).

In [CZ 91], a refutational calculus on c-clauses was introduced, combining refutational rules with rules for model construction. This calculus was later modified and applied to decidable classes of c-clauses (cf. [CP 96]). However, the focus of the latter work was still on model building, i.e.: the authors present an algorithm for automated model building,

*Technische Universität Wien, Institut für Computersprachen, Abteilung für Anwendungen der Formalen Logik, Resselgasse 3/1/3, A-1040 Wien, AUSTRIA, reini@logic.at

which is applicable to known decision classes. The decision classes themselves are literal translations from standard clause logic.

The aim of this work is to combine decision procedures based on resolution refinements with c-clause logic in order to extend the decision classes themselves. Therefore, the first question to be answered here is in what sense do constrained clauses actually provide additional expressive power in comparison with standard clauses. The answer given in chapter 3 is that only constraints made up from conjunctions of disequations constitute a genuine extension w.r.t. standard clauses. A *disequation normal form* of constrained clauses will be defined in order to formalize this result. We then investigate the extension of decidable classes of standard clauses to c-clauses in disequation normal form. To this end, in chapter 4, we shall modify the c-clause calculus of Caferra et al. so as to make it suitable as a decision procedure. The main result of this work will be proven in chapter 5 where we show that a standard clause class decidable by certain resolution refinements remains decidable even if the clauses are extended by constraints consisting of arbitrary conjunctions of disequations. For the termination proof of this decision procedure, the strong compactness of sets of equations proven in [LMM 86] will be slightly extended. Finally, in chapter 6, some directions for future research will be mentioned.

2 Preliminaries

As far as constrained clause logic is concerned, we shall follow the approach of Caferra et al. (cf. [CZ 91] and [CP 96]), who in turn make use of equational problems in the sense of [CL 89]). Hence, in this chapter, we shall revise very briefly some basic definitions and results on equational problems and constrained clauses, which are needed in the subsequent chapters. For any details, the original papers (in particular, [CL 89] and [CZ 91]) have to be referred to.

Recall from [CL 89], that an *equational problem* is a formula of the form $\exists\vec{w}\forall\vec{y}\mathcal{P}(\vec{w},\vec{x},\vec{y})$, where $\mathcal{P}(\vec{w},\vec{x},\vec{y})$ is a quantifier-free formula with equality "=" as the only predicate symbol. Without loss of generality, we can assume that $\mathcal{P}(\vec{w},\vec{x},\vec{y})$ is in negation normal form, i.e.: every occurrence of the negation symbol $\neg$ has been shifted directly in front of an equation. By writing a negated equation $\neg(s = t)$ as a disequation $s \neq t$, all occurrences of the negation symbol can be eliminated. The trivially true problem is denoted by $\top$ and the trivially false one by $\bot$. An important special case of equational problems are *unification problems*, which contain neither universal quantifiers nor disequalities.

In the context of constrained clauses in the sense of [CZ 91], equational problems are only interpreted over the free term algebra. Furthermore, variables of a single sort are considered, i.e.: they may only take values from the same Herbrand universe H. A Herbrand interpretation over H is given through an H-ground substitution σ, whose domain coincides with the free variables of the equational problem. The trivial problem $\top$ evaluates to $\mathbf{T}$ in every interpretation. Likewise, $\bot$ always evaluates to $\mathbf{F}$. A single equation $s = t$ is validated by a ground substitution σ, if $s\sigma$ and $t\sigma$ are syntactically identical. Analogously, a single disequation $s \neq t$ is validated by σ, if $s\sigma$ and $t\sigma$ are syntactically different. The interpretation of the connectives $\wedge$, $\vee$, $\exists$ and $\forall$ is as usual. A ground substitution σ which validates an equational problem $\mathcal{P}$ is called a *solution*.

In analogy with [CL 89] and [LMM 86], we use the following notation for comparing the syntactical form and the solution sets of equational problems, respectively: By $\mathcal{P} \equiv \mathcal{Q}$, we denote syntactical identity. $\mathcal{P} \approx \mathcal{Q}$ means that both problems have the same solution set and $\mathcal{P} \leq \mathcal{Q}$ means that all solutions of $\mathcal{P}$ are also solutions of $\mathcal{Q}$.

In [CZ 91], *constrained clauses* (c-clauses, for short) are defined as pairs $[c : \mathcal{P}]$, where c is a clause and $\mathcal{P}$ is an equational problem. Intuitively, $[c : \mathcal{P}]$ denotes the set $c\sigma$ of ground clauses, where σ is a ground solution of $\mathcal{P}$. Two c-clauses $[c : \mathcal{P}]$ and $[d : \mathcal{Q}]$ are equivalent, iff they represent the same set of ground instances. In [CZ 91], the equivalence of two c-clauses is denoted by $[c : \mathcal{P}] \equiv [d : \mathcal{Q}]$.

A *Herbrand interpretation* $\mathcal{I}$ w.r.t. some Herbrand universe H is given through a subset $\mathcal{I}(P) \subseteq H^n$ for every n-ary predicate symbol P. A ground atom $P(\vec{s})$ evaluates to $\mathbf{T}$ in $\mathcal{I}$, iff $\vec{s} \in \mathcal{I}(P)$. The generalization to negative ground literals $\neg P(\vec{s})$ and to ground clauses $L_1(\vec{s}_1) \vee \ldots \vee L_n(\vec{s}_n)$ follows from the usual interpretation of $\neg$ and $\vee$. A c-clause $[c : \mathcal{P}]$ evaluates to $\mathbf{T}$ in $\mathcal{I}$, iff $c\sigma$ evaluates to $\mathbf{T}$ for every ground solution σ of $\mathcal{P}$. The evaluation of a c-clause $[c : \mathcal{P}]$ to $\mathbf{T}$ in an interpretation $\mathcal{I}$ is denoted by $\mathcal{I} \models [c : \mathcal{P}]$.

3 Disequation Normal Form

The constraint part of a c-clause may play two essentially different roles: On the one hand, it may be simply a short-hand notation for a more complex standard clause or for a finite set of standard clauses, respectively. On the other hand, constraints may give us a means for a finite representation of clause sets, which only have an infinite representation in standard clause logic, e.g.: Over the Herbrand universe with signature $\Sigma = \{f, a\}$, the c-clause $C_1 = [P(x, y) : x = f(y) \vee (\exists z) y = f(z)]$ is equivalent to the set $\mathcal{C}_1 = \{P(f(y), y), P(x, f(z))\}$ of standard clauses. On the other hand, the c-clause $C_2 = [P(x, y) : x \neq y]$ has no finite representation in standard clause logic.

In [LMM 86], it is shown that (apart from some trivial cases) disequations cannot be represented by a finite disjunction of equations (cf. [LMM 86], theorem 6.3). Moreover, in [LM 87], an algorithm is presented for the following problem: Given terms $t, t_1, \ldots, t_n$ over an infinite Herbrand universe H. Is there a finite set of terms $\{s_1, \ldots, s_m\}$, s.t. the set of H-ground instances of t which are not instances of any term t_i, is identical to the union of the sets of H-ground instances of the terms s_j. In our terminology of c-clauses, this problem can be reformulated in the following way: Let $t, t_1, \ldots, t_n$ be terms over an infinite Herbrand universe s.t. t has no variables in common with the t_i's. Moreover, let $\vec{y}$ denote the set of all variables occurring in the t_i's and let P be a unary predicate symbol. Then the algorithm from [LM 87] also decides the the following question: Can the c-clause $[P(t) : \forall \vec{y}\, (t \neq t_1 \wedge \ldots \wedge t \neq t_n)]$ be represented by a finite set of standard clauses?

In this chapter, we shall prove a related result, namely: only disequations or conjunctions of disequations are more than just short-hand notations of clause sets that could be represented in standard clause logic as well. These considerations will be formalized by defining the so-called *disequation normal form* (= DeqNF) of c-clauses and by showing that every finite set of c-clauses with arbitrary constraints can be effectively transformed into an equivalent finite set of c-clauses in DeqNF.

Definition 3.1 (disequation normal form) *Let $C = [c : \mathcal{P}]$ be a c-clause over an infinite Herbrand universe H. We say that C is in* disequation normal form *(= DeqNF), iff the following conditions hold:*

1. *$\mathcal{P}$ is a quantifier-free conjunction of disequations or the empty conjunction $\top$.*
2. *All variables of the constraint part $\mathcal{P}$ also occur in the clause part c.*

Theorem 3.2 (transformation into DeqNF) *Let H be an infinite Herbrand universe and let $\mathcal{C}$ be a finite set of arbitrary c-clauses over H. Then $\mathcal{C}$ can be effectively transformed into an equivalent finite set of c-clauses $\mathcal{C}'$ in disequation normal form.*

Proof (sketch): We have to apply several transformation steps to every clause $[c : \mathcal{P}] \in \mathcal{C}$, until all resulting c-clauses have the desired form. These steps are sketched below. The details are worked out in a full version of this paper (cf. [Pic 98a]).

- step 1 (definition with constraints):
  By [CL 89], the constraint part $\mathcal{P}$ can be transformed into an equivalent equational problem in "definition with constraints solved form", i.e.: $\mathcal{P} \equiv \bot$ or $\mathcal{P} \equiv \top$ or $\mathcal{P}$ is a disjunction of problems $\mathcal{P}_i$ of the form

  $$(\exists w_1) \ldots (\exists w_m) x_1 = t_1 \wedge \ldots \wedge x_k = t_k \wedge z_1 \neq u_1 \wedge \ldots \wedge z_l \neq u_l,$$

  where all variables $x_1, \ldots, x_k$ occur only once in $\mathcal{P}_i$.

- step 2 (elimination of disjunctions):
  Let $\mathcal{P} \equiv \mathcal{P}_1 \vee \ldots \vee \mathcal{P}_n$. Then $[c : \mathcal{P}]$ is equivalent to the c-clause set $\{[c : \mathcal{P}_1], \ldots, [c : \mathcal{P}_n]\}$.

- step 3 (elimination of existential quantifiers):
  Let $\mathcal{P}_i$ be an equational problem of the form $(\exists w_1) \ldots (\exists w_m) Q$ and suppose w.l.o.g. that the existentially bound variables w_i do not occur in the clause part (otherwise they can be renamed without changing the meaning of the equational problem). Then the existential quantifiers may be eliminated by m applications of the following "structural rule" from [CZ 91]:

  $$[c : \mathcal{Q}] \equiv [c : (\exists w)\mathcal{Q}], \text{ if } w \text{ does not occur in } c.$$

- step 4 (elimination of equations):
  Let $[c : \mathcal{Q}]$ be a c-clause with $\mathcal{Q} \equiv x_1 = t_1 \wedge \ldots \wedge x_k = t_k \wedge z_1 \neq u_1 \wedge \ldots \wedge z_l \neq u_l$, s.t. the variables $x_1, \ldots, x_k$ occur only once in $\mathcal{Q}$. Then the equivalence

  $$[c : \mathcal{Q}] \equiv [c\sigma : (z_1 \neq u_1 \wedge \ldots \wedge z_l \neq u_l)\sigma], \text{ where } \sigma = \{x_1 \leftarrow t_1, \ldots, x_k \leftarrow t_k\}.$$

  slightly extends another "structural rule" from [CZ 91].

- step 5 (elimination of "additional" free variables):
  Let $[d : \mathcal{R}]$ be a c-clause resulting from step 4 above. Then the transformation from step 3 can also be applied in the opposite direction, i.e.: all variables in $var(\mathcal{R}) - var(d)$ may be existentially quantified.

- step 6 (alternative way of eliminating the existential quantifiers):
  Let $[d : (\exists v_1) \ldots (\exists v_\alpha)\mathcal{R}]$ be a c-clause with $\mathcal{R} \equiv z_1 \neq u_1 \wedge \ldots \wedge z_l \neq u_l$. Then, by the cleaning rule CR_3 from [CL 89], all disequations containing a bound variable may be deleted. But then the prefix of existential quantifiers may also be deleted, since the resulting conjunction of disequations contains no bound variables anymore. Therefore, the following c-clauses are equivalent:

  $$[d : (\exists v_1) \ldots (\exists v_\alpha)\mathcal{R}] \equiv [d : z_{j_1} \neq u_{j_1} \wedge \ldots \wedge z_{j_{l'}} \neq u_{j_{l'}}],$$

  where $\{j_1, \ldots, j_{l'}\} = \{j | z_j \neq u_j \text{ contains no bound variable } v_i\}$. ◇

The steps 5 and 6 in the above proof were carried out separately for the sake of better readability. Note, however, that these two steps together result in the deletion of all disequations that contain an "additional" free variable v_i after step 4. Hence, in practice, they will naturally be contracted to a single transformation step. In fact, this is exactly what the normalization rule of our c-clause calculus in definition 4.1 will do.

4 A resolution-based calculus in c-clause logic

The original calculus of Caferra et al. (and, in particular, the cRes rule and cDsub rule) provides the basis for the construction of a decision procedure for certain classes of c-clauses (a short overview of this calculus can be found in [Pic 98b]). Starting from an arbitrary resolution refinement R_x on standard clauses, we define the following calculus $\mathcal{I}_x$ on c-clauses:

Definition 4.1 (calculus $\mathcal{I}_x$ on c-clauses) *Let R_x be some resolution refinement on standard clauses. Then the calculus $\mathcal{I}_x$ on c-clauses in DeqNF is defined through the following rules and rule application strategy:*

rules of $\mathcal{I}_x$:

1. *refined c-resolution: Let L denote some literal symbol (i.e.: a predicate symbol or a negated predicate symbol) and let L^d denote its dual. Furthermore, let $[L(\vec{s}_1) \vee \ldots \vee L(\vec{s}_n) \vee c : X]$ and $[L^d(\vec{t}_1) \vee \ldots \vee L^d(\vec{t}_m) \vee d : Y]$ be variable disjoint c-clauses in DeqNF. Then we define the $cRes_x$ rule in the following way:*

$$\frac{[L(\vec{s}_1) \vee \ldots \vee L(\vec{s}_n) \vee c : X] \qquad [L^d(\vec{t}_1) \vee \ldots \vee L^d(\vec{t}_m) \vee d : Y]}{[(c \vee d)\sigma : (X \wedge Y)\sigma]} \quad cRes_x,$$

   where $\sigma = mgu(\{\vec{s}_1, \ldots, \vec{s}_n, \vec{t}_1, \ldots, \vec{t}_m\})$ and $(c \vee d)\sigma$ is an R_x-resolvent of $L(\vec{s}_1) \vee \ldots \vee L(\vec{s}_n) \vee c$ and $L^d(\vec{t}_1) \vee \ldots \vee L^d(\vec{t}_m) \vee d$.

2. *normalization: Let $[c : \mathcal{P}]$ be a c-clause derived by the $cRes_x$ rule above and let $u_{i_1} \neq v_{i_1}, \ldots, u_{i_k} \neq v_{i_k}$ denote the disequations in $\mathcal{P}$ containing a variable which does not occur in c. Then every disequation $u_{i_\alpha} \neq v_{i_\alpha}$ may be either replaced by $\bot$ (if $u_{i_\alpha} \equiv v_{i_\alpha}$ holds) or deleted (otherwise).*

3. *c-subsumption: Let $[c : X_1]$, ..., $[c : X_k]$ be c-clauses in the data base and let $[c : X_{k+1}]$ be a new $cRes_x$-resolvent. Then $[c : X_{k+1}]$ may be deleted, if $X_{k+1} \leq X_1 \vee \ldots \vee X_k$.*

rule application strategy $\mathcal{I}_x$:

We require that the c-subsumption rule be applied immediately to every newly derived c-clause. Furthermore, normalization has to be applied to every $cRes_x$ resolvent before it is added to the data base. As far as fairness is concerned, we choose the simple strategy of level saturation, i.e.: The whole generation of c-clauses derivable by n applications of the $cRes_x$ rule has to be derived before a resolvent is derived via $n+1$ $cRes_x$ rule applications.

The above calculus $\mathcal{I}_x$ differs from the original definition of Caferra et al. in various ways (cf. [CZ 91]). Analogously to Robinson's original definition of resolution, we consider factoring and resolution as a single inference step (cf. [Rob 65]). More importantly, we apply the unifying substitution σ explicitly both to the clause part and to the constraint part whereas, in [CZ 91], the unification is carried out implicitly by adding the corresponding equations $\vec{s}_i = \vec{t}_i$ to the constraints. Furthermore, the definition of c-subsumption given above can be considered as the original cDsub rule from [CZ 91] and [CP 96] together with a very restricted rule application strategy, namely: The cDsub rule may only be applied if all c-clauses involved have the same clause part and if, moreover, a finite number of applications of this rule ultimately leads to the actual deletion of $[c : X_{k+1}]$.

The correctness of the calculus $\mathcal{I}_x$ is easy to prove: The correctness of the resolution refinement cRes_x follows immediately from the corresponding correctness in standard clause logic. Furthermore the deletion of c-clauses via c-subsumption may only affect the completeness (but not the correctness) of a calculus. As far as the normalization rule is concerned, we have to distinguish two cases: A disequation $u_{i_\alpha} \not\approx v_{i_\alpha}$ where $u_{i_\alpha} \equiv v_{i_\alpha}$ holds, is trivially unsolvable. Hence, it is correct to replace it by $\bot$. If $u_{i_\alpha} \not\approx v_{i_\alpha}$ contains a variable not occurring in the clause part and $u_{i_\alpha} \not\equiv v_{i_\alpha}$ holds, then we may apply the transformation steps 5 and 6 from theorem 3.2 and delete this disequation.

On the other hand, the completeness of $\mathcal{I}_x$ is not so trivial. In particular, the compatibility of the c-subsumption rule with any resolution refinement need not necessarily hold. In [Pic 98b], the completeness of the calculus of Caferra et al. is shown by extending the well-known concept of semantic trees from standard clauses to c-clauses. In [KH 69], the completeness of unrefined resolution R_0, A-ordering resolution $\text{R}_{<_A}$ and semantic clash resolution $\text{R}_{\mathcal{M}}$ are proven via semantic trees in standard clause logic. Analogously, the refutational completeness of the calculi $\mathcal{I}_0$, $\mathcal{I}_{<_A}$ and $\mathcal{I}_{\mathcal{M}}$ based on these resolution refinements can be shown via semantic trees in c-clause logic.

5 Extension of decision classes from standard clause logic

The target of this chapter is to show how decision classes from standard clause logic can be extended by adding appropriate constraints.

To this end, we first have to show that the termination of a resolution refinement R_x on standard clauses can be easily carried over to the corresponding calculus $\mathcal{I}_x$ on c-clauses. For our termination proof in theorem 5.2, we need the following compactness theorem on unification problems which slightly extends a related result from [LMM 86] (cf. theorem A.5 in the appendix).

Theorem 5.1 (compactness of unification problems) *Let H be an infinite Herbrand universe and let V be a finite set of variables. Moreover, let $\mathcal{N} = \{Y_1, Y_2, Y_3 \ldots\}$ be an infinite set of unification problems over H with variables in V.*

Then there is a finite subset $\mathcal{N}' \subseteq \mathcal{N}$, s.t. $\mathcal{N}'$ is equivalent to the whole set $\mathcal{N}$. In particular, there exists a K s.t. for all $n > K$, the condition $Y_1 \wedge \ldots \wedge Y_K \leq Y_n$ holds.

Proof (sketch):
restriction to disjunctions of equations: W.l.o.g. we can assume that every unification problem Y_i is in CNF, i.e.: $Y_i \equiv (Z_{i1} \wedge \ldots \wedge Z_{in_i})$, where every Z_{ij} is a disjunction of equations. Furthermore, let $\mathcal{M} = \{Z_{11}, \ldots, Z_{1n_1}, Z_{21}, \ldots, Z_{2n_2}, \ldots\}$. It is then sufficient to show that there exists a finite subset $\mathcal{M}' \subseteq \mathcal{M}$, s.t. $\mathcal{M}'$ is equivalent to the whole set $\mathcal{M}$. Hence, w.l.o.g. we can restrict our considerations to the case where every unification problem $Y_i \in \mathcal{N}$ is a disjunction of equations, i.e.: $Y_i \equiv e_{i1} \vee \ldots \vee e_{ik_i}$

multiset of dimensions: Suppose that some unification problem S is in DNF, i.e.: $S \equiv Z_1 \vee \ldots \vee Z_N$, where every Z_i is a conjunction of equations. Furthermore, assume that all conjunctions Z_i are solvable. Then, by [LMM 86], proposition 3.3, the dimension $dim(Z_i)$ is well-defined (Recall that the dimension of a unification problem $\mathcal{P}$ denotes the number of unbound variables in a most general unifier of $\mathcal{P}$). Hence, for a unification problem S in DNF, we can define the multiset of dimensions as $DIM(S) := \{dim(Z_1), \ldots, dim(Z_N)\}$. If S is unsolvable, we set $DIM(S) := \emptyset$. The proof of theorem 5.1 will then be based on the following idea: Let $S_n \equiv Y_1 \wedge \ldots \wedge Y_n$ denote the conjunction of the first n unification

problems from $\mathcal{N}$ and let S'_n be an equivalent problem in DNF. We have to provide an appropriate algorithm for computing the DNF S'_n s.t. the following implication holds:

$$S_n \not\approx S_{n-1} \Rightarrow DIM(S'_n) \prec DIM(S'_{n-1}),$$

i.e.: If the solutions of S_n are a proper subset of the solutions of S_{n-1}, then the multiset $DIM(S'_n)$ is strictly smaller than $DIM(S'_{n-1})$. (For the details of such an algorithm see the appendix A.) Note that "$\leq$" is a well-founded ordering on the natural numbers. Hence, "$\preceq$" is well-founded on multisets over the natural numbers, i.e.: There exists no infinite, strictly decreasing sequence of multisets $DIM(S'_{n_1}), DIM(S'_{n_2}), \ldots$. But then $S_n \approx S_{n-1}$ holds for all but finitely many n's. Thus there exists a K s.t. for all $n > K$, the condition $Y_1 \wedge \ldots \wedge Y_{n-1} \approx Y_1 \wedge \ldots \wedge Y_{n-1} \wedge Y_n$ (or, equivalently, $Y_1 \wedge \ldots \wedge Y_{n-1} \leq Y_n$) holds. We can, therefore, conclude by a simple induction argument that $Y_1 \wedge \ldots \wedge Y_K \leq Y_n$ holds for all $n > K$. $\diamond$

The following theorem gives a sufficient criterion for the termination of the calculus $\mathcal{I}_x$, namely: $\mathcal{I}_x$ terminates on a set of c-clauses in DeqNF, if the original resolution refinement R_x terminates on the corresponding set of standard clauses.

Theorem 5.2 (Termination of $\mathcal{I}_x$) *Let R_x be a resolution refinement on standard clauses and let $\mathcal{I}_x$ be the corresponding calculus on c-clauses according to definition 4.1. Furthermore, let $\mathcal{C} = \{[c_1 : \mathcal{P}_1], \ldots, [c_n : \mathcal{P}_n]\}$ be a set of c-clauses over some infinite Herbrand universe H, s.t. $\mathcal{C}$ is in disequation normal form and $\mathcal{C}' = \{c_1, \ldots, c_n\}$ is the corresponding set of standard clauses consisting of the clause parts of $\mathcal{C}$. By $R_x(\mathcal{C}')$, we denote the clauses derivable from $\mathcal{C}'$ via finitely many applications of the resolution refinement R_x. Likewise, the set of c-clauses derivable from $\mathcal{C}$ via $\mathcal{I}_x$ will be denoted as $\mathcal{I}_x(\mathcal{C})$. Then the following implication holds:*

If $R_x(\mathcal{C}')$ is finite, then $\mathcal{I}_x(\mathcal{C})$ is also finite.

Proof: Suppose that $R_x(\mathcal{C}')$ is finite. We have to prove, that $\mathcal{I}_x(\mathcal{C})$ is also finite, i.e.: there are only finitely many distinct clause parts and constraint parts in $\mathcal{I}_x(\mathcal{C})$:

- (clause parts):
  By the definition of $\mathcal{I}_x$, the set $\mathcal{D} := \{c \mid [c : \mathcal{P}] \in \mathcal{I}_x(\mathcal{C})\}$ of clause parts produced by $\mathcal{I}_x$ is contained in the set $\mathrm{R}_x(\mathcal{C}')$ of standard clauses generated by R_x. Hence, the set $\mathcal{D}$ of clause parts in $\mathcal{I}_x(\mathcal{C})$ is finite.

- (constraint parts):
  We have to show that for every clause part c in $\mathcal{I}_x(\mathcal{C})$, there are only finitely many distinct constraint parts X_i s.t. $[c : X_i] \in \mathcal{I}_x(\mathcal{C})$. Suppose, on the contrary, that there is an infinite sequence $[c : X_1]$, $[c : X_2]$, $[c : X_3], \ldots$ of c-clauses in $\mathcal{I}_x(\mathcal{C})$ such that for all $n > 1$, $[c : X_n]$ is not c-subsumed by the previously derived c-clauses $\{[c : X_1], \ldots, [c : X_{n-1}]\}$. The X_i's are conjunctions of disequations over the infinite Herbrand universe H with variables in the finite set $V = var(c)$. Therefore, the equational problems $\neg X_1$, $\neg X_2$, $\neg X_3, \ldots$ are disjunctions of equations over the infinite Herbrand universe H with variables in the finite set V. By theorem 5.1, we know that there is a K s.t. for all $n > K$: $\neg X_1 \wedge \ldots \wedge \neg X_K \leq \neg X_n$ or, equivalently, $X_n \leq X_1 \vee \ldots \vee X_K$ holds. But then, by definition 4.1, $\{[c : X_1], \ldots, [c : X_K]\}$ c-subsumes $[c : X_n]$ for all $n > K$, which contradicts the assumption. $\diamond$

The following theorem constitutes the main result of this work:

Theorem 5.3 (decision procedure $\mathcal{I}_x$) *Let R_x denote either unrefined resolution R_0, A-ordering resolution $R_{<_A}$ or semantic clash resolution $R_{\mathcal{M}}$. Furthermore, let $\mathcal{K}$ denote a class of standard clauses over an infinite Herbrand universe H, s.t. $\mathcal{K}$ is decidable by R_x. Then $\mathcal{K}$ can be extended to the decidable class $\mathcal{K}'$ of c-clauses in the following way:*

$\mathcal{C}' = \{[c_1 : \mathcal{P}_1], \ldots, [c_n : \mathcal{P}_n]\}$ is in $\mathcal{K}'$, iff $\mathcal{C} := \{c_1, \ldots, c_n\}$ is a set of standard clauses in $\mathcal{K}$ and for all $i \in \{1, \ldots, n\}$, $\mathcal{P}_i$ is a quantifier-free conjunction of disequations with $var(\mathcal{P}_i) \subseteq var(c_i)$.

Then $\mathcal{I}_x$ is a decision procedure for the class $\mathcal{K}'$.

Proof: First of all, we have to make sure that the class $\mathcal{K}'$ is closed w.r.t. the calculus $\mathcal{I}_x$: But the clause parts of the derivable c-clauses result from applications of R_x to $\mathcal{K}$. Furthermore, all input c-clauses are in disequation normal form, which is preserved by the rules from $\mathcal{I}_x$. Hence, the resulting c-clauses are in $\mathcal{K}'$. (Note that the DeqNF may be destroyed by cRes_x since the constraint part of the resolvent may contain variables not occurring in the clause part, e.g.: In case of unrefined resolution R_0, the c-clause $[P(x) : y \neq a \wedge y \neq b]$ may be derived by the cRes_0 rule from $[P(x) \vee Q(y) : y \neq a]$ and $[\neg Q(y') : y' \neq b]$. However, by the rule application strategy, the normalization rule must be applied to the resolvent, which allows for the elimination of these variables, i.e.: in the above example, only the c-clause $[P(x) : \top]$ is actually stored.)

The refutational completeness of the calculus $\mathcal{I}_x$ based on the above mentioned resolution refinements R_x is proven in [Pic 98b] (cf. the remark in chapter 4). The termination of the calculus $\mathcal{I}_x$ on the class $\mathcal{K}'$ follows from the theorem 5.2. $\diamond$

The following examples illustrate how this theorem can lead to a genuine extension of standard clause classes decidable by A-ordering resolution $\mathrm{R}_{<_A}$ or semantic clash resolution $\mathrm{R}_{\mathcal{M}}$, respectively.

Example 5.4 (extension of the decision class MON*) *Let the following clauses and c-clauses be defined over the Herbrand universe H with signature $\Sigma = \{a^0, b^0, f^2\}$, where the exponent denotes the arity of each symbol. Recall from [Lei 97] that the class MON* is decidable by an A-ordering resolution refinement. MON* essentially denotes those sets of clauses, where all predicate symbols have arity 1 and variables may only occur in variable arguments or in functional arguments of the form $g(y_1, \ldots, y_\alpha)$, e.g.: $\mathcal{C}' := \{P(x), P(f(x,y)) \vee Q(x), \neg Q(a)\}$. Then the following sets $\mathcal{C}_1$ and $\mathcal{C}_2$ of c-clauses in disequation normal form are obtained from $\mathcal{C}'$ by adding appropriate constraints:*

By theorem 5.3, the c-clause set $\mathcal{C}_1 := \{[P(x) : \top], [P(f(x,y)) \vee Q(x) : x \neq y], [\neg Q(a) : \top]\}$ belongs to a decidable class of c-clauses even though it does not have a finite representation in standard clause logic.

The c-clause set $\mathcal{C}_2 := \{[P(x) : \top], [P(f(x,y)) \vee Q(x) : y \neq a \wedge y \neq b], [\neg Q(a) : \top]\}$ is based on the same set of standard clauses and, therefore, belongs to a decidable class of c-clauses by theorem 5.3. However, for this Herbrand universe, the equational problem $y = a \vee y = b \vee (\exists u)(\exists v) y = f(u,v)$ is valid. Hence, $\mathcal{C}_2$ is equivalent to the c-clause set $\{[P(x) : \top], [P(f(x,y)) \vee Q(x) : (\exists u)(\exists v) y = f(u,v)], [\neg Q(a) : \top]\}$ which, in turn, is equivalent to the standard clause set $\mathcal{C}_2' = \{P(x), P(f(x, f(u,v))), \neg Q(a)\}$. Note that the set $\mathcal{C}_2'$ is outside the class MON*, due to the nested functional term in $P(f(x, f(u,v)))$. Hence, in this case, the transformation into c-clauses allows for an extension of the decidable standard clause class itself.

Example 5.5 (extension of the decision class PVD) *Let the following clauses and c-clauses be defined over the Herbrand universe H with signature $\Sigma = \{a^0, f^2, g^1\}$. In [Lei 97], the class PVD is shown to be decidable by hyperresolution. Recall that PVD (= positive variable dominated) consists of the sets of clauses, where all variables from unnegated literals also occur in negated ones and where furthermore the maximum depth of occurrence of each variable in the unnegated literals is not greater than the maximum depth of occurrence of this variable in the negated literals, e.g.: $\mathcal{C}' := \{\neg P(x), P(g(x)) \vee Q(f(y,x)), \neg Q(x)\}$. Then the set $\mathcal{C} = \{[\neg P(x) : \top], [P(g(x)) \vee Q(f(y,x)) : x \neq y], [\neg Q(x) : x \neq a]\}$ does not have a finite representation in standard clause logic. Nevertheless, by theorem 5.3, it belongs to a decidable class of c-clauses.*

6 Concluding Remarks and Future Work

After briefly revising some basic concepts concerning equational problems and constrained clauses, we have provided an answer to the question in what sense constraints actually do increase the expressive power of clauses. The result was formalized by defining the so-called disequation normal form (= DeqNF) of c-clauses, i.e.: the constraint part consists of conjunctions of disequations and contains only variables which also occur in the clause part. Furthermore we proved that every finite set of c-clauses can be effectively transformed into an equivalent finite set of c-clauses in DeqNF. The main result of this paper is the extension of standard clause classes decidable by certain resolution refinements to c-clauses in DeqNF.

So far, we have only considered clause classes which are decidable by some resolution refinement. However, many more ingredients for resolution-based decision procedures can be found in the literature (cf. [FLTZ 93]), e.g.: subsumption, the splitting rule, condensing, partial saturation, etc. Furthermore, there are several paramodulation-based decision procedures for decision classes containing equality. It still remains to investigate, how a standard clause calculus based on these methods can be extended to c-clause logic without destroying the refutational completeness and the termination property.

In [FLTZ 93], resolution based decision methods for many classes of standard clauses are presented. The expressive power gained through the extension of these classes to c-clause logic and potential applications of the resulting languages deserve further study. In particular, what kind of problems can be expressed by the resulting classes of c-clauses in DeqNF?

References

[CL 89] H. Comon, P. Lescanne: Equational Problems and Disunification, Journal of Symbolic Computation, vol. 7, pp. 371-425, (1989).

[CP 96] R.Caferra, N.Peltier: Decision Procedures using Model Building Techniques, Proceedings of CSL'95, LNCS 1092, pp.130-144, Springer (1996).

[CZ 91] R.Caferra, N.Zabel: Extending Resolution for Model Construction, Proceedings of Logics in AI - JELIA '90, LNAI 478, pp. 153-169, Springer (1991).

[FLTZ 93] C.Fermüller, A.Leitsch, T.Tammet, N. Zamov: Resolution Methods for the Decision Problem. LNAI 679, Springer (1993).

[KH 69] R.Kowalski, P.J.Hayes: Semantic Trees in Automated Theorem Proving, in Machine Intelligence 4, pp. 87-101 (1969).

[Lei 97] A.Leitsch: The Resolution Calculus, Texts in Theoretical Computer Science, Springer (1997).

[LM 87] J.-L.Lassez, K.Marriott: Explicit Representation of Terms defined by Counter Examples, Journal of Automated Reasoning, Vol 3, pp. 301-317 (1987).

[LMM 86] J.-L.Lassez, M.J.Maher, K.Marriott: Unification Revisited, Foundations of Logic and Functional Programming, LNCS 306, pp. 67-113, Springer (1986).

[Pic 98a] R.Pichler: Extending Decidable Clause Classes via Constraints (full version), technical report TR-CS-RP-98-2 of the Technical University of Vienna, available as `ftp://ftp.logic.at/pub/reini/constr.ps` (1998).

[Pic 98b] R.Pichler: Completeness and Redundancy in Constrained Clause Logic, in Proceedings of FTP'98 (International Workshop, First-Order Theorem Proving), Vienna (1998).

[Rob 65] J.A.Robinson: A machine oriented logic based on the resolution principle, Journal of the ACM, Vol 12, No 1, pp. 23- 41 (1965).

Appendix

A Proof of Theorem 5.1

For the proof of theorem 5.1, we make use of the following basic definitions and results on unification (from [LMM 86]):

We only consider equation sets here with a finite set of variables V over a non-trivial Herbrand universe H. The following definitions form the basis of the considerations in [LMM 86]:

Definition A.1 (solved form of an equation set) *An equation set $\mathcal{E}$ is called* solved, *iff it has the form $\{v_1 = t_1, \ldots, v_n = t_n\}$ s.t. the v_i's are pairwise distinct variables which do not occur on the right hand side of any equation. The variables $v_1, \ldots, v_n$ are said to be* eliminable *and the remaining variables from V are called* parameters.

Definition A.2 (dimension of an equation set) *Let $\mathcal{E}$ be a solvable set of equations and let $\mathcal{E}'$ be some solved form equivalent to $\mathcal{E}$. Then $dim(\mathcal{E})$ (=* dimension *of $\mathcal{E}$) denotes the number of parameters in $\mathcal{E}'$.*

The following results on the dimension $dim(\mathcal{E})$ of an equation set $\mathcal{E}$ are proven in [LMM 86], propositions 3.3 and 3.5, respectively:

Proposition A.3 (well-defined dimension) *The dimension $dim(\mathcal{E})$ of a (solvable) set of equations is well-defined, i.e.: all solved forms $\mathcal{E}'$ equivalent to $\mathcal{E}$ have the same number of parameters.*

Proposition A.4 (dimension and solution sets) *Let $\mathcal{E}_1$ and $\mathcal{E}_2$ be two solvable sets of equations. Then the following implication holds:*
If the solutions of $\mathcal{E}_1$ are a proper subset of the solutions of $\mathcal{E}_2$, then $dim(\mathcal{E}_1) < dim(\mathcal{E}_2)$.

The following result on the strong compactness of sets of equations from [LMM 86], theorem 3.11, is somehow similar to our theorem 5.1:

Theorem A.5 (strong compactness) *Let $\mathcal{E}$ and $\mathcal{E}_1, \ldots, \mathcal{E}_n$ be equation sets over an infinite Herbrand universe. Then the following implication holds:*
If $\mathcal{E} \approx \mathcal{E}_1 \vee \ldots \vee \mathcal{E}_n$, then there exists some j s.t. $\mathcal{E} \approx \mathcal{E}_j$

Multisets are a common technique in termination proofs with a broad field of applications due to the property of "well-foundedness", which the multiset ordering "$\preceq$" inherits from the original ordering "$\leq$". For details (and also for an interesting application of multiset

orderings, namely the transformation of arbitrary equational problems into "definition with constraints solved form" mentioned in the proof of theorem 3.2), [CL 89] should be referred to.

We are now ready to prove the following theorem from chapter 5:

Theorem 5.1 (compactness of unification problems) *Let H be an infinite Herbrand universe and let V be a finite set of variables. Moreover, let $\mathcal{N} = \{Y_1, Y_2, Y_3 \ldots\}$ be an infinite set of unification problems over H with variables in V.*

Then there is a finite subset $\mathcal{N}' \subseteq \mathcal{N}$, s.t. $\mathcal{N}'$ is equivalent to the whole set $\mathcal{N}$. In particular, there exists a K s.t. for all $n > K$, the condition $Y_1 \wedge \ldots \wedge Y_K \leq Y_n$ holds.

Proof: The only part missing in the proof sketch from chapter 5 is an algorithm for computing a sequence of DNFs S'_n equivalent to $S_n \equiv Y_1 \wedge \ldots \wedge Y_n$, s.t. the implication

$$S_n \not\approx S_{n-1} \Rightarrow DIM(S'_n) \prec DIM(S'_{n-1})$$

holds. We construct the DNFs S'_n inductively as follows:
By assumption, $S_1 \equiv Y_1 \equiv e_{11} \vee \ldots \vee e_{1k_1}$ already is in DNF, i.e.: $S'_1 := S_1$.
Now suppose that $S'_{n-1} \equiv Z_1 \vee \ldots \vee Z_N$ is the DNF corresponding to S_{n-1}. Then $S_n \equiv Y_1 \wedge \ldots \wedge Y_{n-1} \wedge Y_n \equiv S_{n-1} \wedge Y_n$ is equivalent to the following problem:

$$S_n \approx (Z_1 \vee \ldots \vee Z_N) \wedge (e_{n1} \vee \ldots \vee e_{nk_n}) \approx (Z_1 \wedge (e_{n1} \vee \ldots \vee e_{nk_n})) \vee \ldots \vee (Z_N \wedge (e_{n1} \vee \ldots \vee e_{nk_n}))$$

In order to transform this problem into a disjunctive normal form S'_n, we transform every disjunct $Z_i \wedge (e_{n1} \vee \ldots \vee e_{nk_n})$ into an equivalent problem D_i in DNF in the following way:

$$D_i := \begin{cases} Z_i \quad \text{if there exists a } j \text{ s.t. } (Z_i \wedge e_{nj}) \approx Z_i \\ \bot \quad \text{if for all } j\text{:} \quad (Z_i \wedge e_{nj}) \approx \bot \\ (Z_i \wedge e_{nj_1}) \vee \ldots \vee (Z_i \wedge e_{nj_l}) \quad \text{otherwise,} \\ \quad \text{where } \{j_1, \ldots, j_l\} = \{j \mid Z_i \wedge e_{nj} \text{ is solvable}\} \end{cases}$$

Finally, all unsolvable problems D_i have to be deleted from S'_n. Then every D_i and, therefore, also the disjunction S'_n clearly is in DNF. Moreover, the equivalence $D_i \approx Z_i \wedge (e_{n1} \vee \ldots \vee e_{nk_n})$ and, hence, also the equivalence $S'_n \approx S_n$ are easy to prove. Furthermore, the multiset $DIM(S'_n)$ can be computed from $DIM(S'_{n-1})$ in the following way:

1. If $D_i \equiv Z_i$, then $dim(Z_i) \in DIM(S'_{n-1})$ is left unchanged in $DIM(S'_n)$.
2. If $D_i \equiv \bot$, then $dim(Z_i) \in DIM(S'_{n-1})$ is deleted from $DIM(S'_n)$.
3. If $D_i \equiv (Z_i \wedge e_{nj_1}) \vee \ldots \vee (Z_i \wedge e_{nj_l})$ (according to case 3 of the above definition of D_i), then $dim(Z_i)$ is replaced by $dim(Z_i \wedge e_{nj_1}), \ldots, dim(Z_i \wedge e_{nj_l})$.

Remember that case 3 of the definition of D_i only applies if $Z_i \wedge e_{nj_\alpha} < Z_i$ for all α, i.e.: the solutions of every problem $(Z_i \wedge e_{nj_\alpha})$ are a proper subset of the solutions of Z_i. But then, by proposition 3.5 from [LMM 86], $dim(Z_i \wedge e_{nj_\alpha}) < dim(Z_i)$ also holds. Hence, by the definition of multiset orderings, $DIM(S'_n) = DIM(S'_{n-1})$, iff case 1 applies to all disjuncts D_i of S'_n, and $DIM(S'_n) \prec DIM(S'_{n-1})$ otherwise. In other words, $DIM(S'_n) = DIM(S'_{n-1})$, iff $S_n \approx S'_{n-1}$, and $DIM(S'_n) \prec DIM(S'_{n-1})$ otherwise. $\diamond$