A Resolution Decision Procedure for the Guarded Fragment with Transitive Guards

Yevgeny Kazakov

MPI für Informatik, Saarbrücken, Germany
ykazakov@mpi-sb.mpg.de

Abstract. We show how well-known refinements of ordered resolution, in particular redundancy elimination and ordering constraints in combination with a selection function, can be used to obtain a decision procedure for the guarded fragment with transitive guards. Another contribution of the paper is a special scheme notation, that allows to describe saturation strategies and show their correctness in a concise form.

1 Introduction

The *guarded fragment* $\mathcal{GF}$ of first order logic has been introduced by Andréka, van Benthem & Németi (1998) to explain and generalize the good computational properties of modal and temporal logics. This is achieved essentially by restricting quantifications in first order formulae to the following "bounded" forms: $\forall\overline{x}.[G \to F]$ and $\exists\overline{y}.[G \wedge F]$, where G should be an atomic formula (so-called *guard*) containing all free variables of F. The guarded fragment is decidable in 2EXPTIME (Grädel 1999) and inherits many other nice computational properties from the modal logics like the finite model property, the interpolation property and invariance under an appropriate notion of bisimulation.

The *transitivity axiom:* $\forall xyz.(xTy \wedge yTz \to xTz)$ is an example of formula that cannot be expressed in $\mathcal{GF}$. Transitivity is important, since it is used to model discrete time (in temporal verification) and ordered structures (in program shape analysis). However, as it is shown by Grädel (1999), the guarded fragment becomes undecidable as long as transitivity is allowed. (Szwast & Tendera 2001) have considered an extension of $\mathcal{GF}$, that allows for a restricted use of transitive relations, namely in guard positions only. This, so called *guarded fragment with transitive guards* $\mathcal{GF}[\mathcal{TG}]$ was shown to be decidable in 2EXPTIME, that is of the same complexity as $\mathcal{GF}$.

The result obtained by (Szwast & Tendera 2001) is far not trivial and is done by establishing the *regular model property* for $\mathcal{GF}[\mathcal{TG}]$. This yields a decision procedure for $\mathcal{GF}[\mathcal{TG}]$ based on enumeration of certain regular structures. A practical disadvantage such procedures, is that without further optimizations, those methods exhibit the full worst-case complexity. Resolution-based approach, is a reasonable alternative to model-theoretic procedures, as its goal-oriented nature and numerous refinements allow to scale well between "easy" and "hard" instances of problems. In this paper we demonstrate the practical power of resolution refinements, such as redundancy elimination and usage of ordering constraints in combination with selection function. We present a first resolution-based decision procedure for $\mathcal{GF}[\mathcal{TG}]$. Another aspect that is demonstrated in our paper is the usage of resolution as a specification language for decision procedures. We introduce a special scheme notation that allows to describe resolution strategies in a concise form. This may provide a formal foundation for using resolution for specifying decision procedures and proving their correctness.

The framework of resolution theorem proving. The *ordered resolution calculus* $\mathcal{OR}^{\succ}_{Sel}$ is parametrized with an *admissible ordering* $\succ$ and a *selection function* Sel.[1] It can be formulated with the following inference rules (we mark eligible literals with asterisk and underline expressions to be unified):

Ordered (Hyper-)Resolution

$$HR\colon \frac{C_1 \vee \underline{A_1}^* \ldots C_n \vee \underline{A_n}^* \;\; D \vee \neg\underline{B_1}^* \vee ... \vee \neg\underline{B_n}^*}{C_1\sigma \vee ... \vee C_n\sigma \vee D\sigma} \;\left|\; \begin{array}{l} \textit{where (i) } \sigma = mgu(A_i, B_i)\textit{, (ii) } A_i \\ \textit{and } \neg B_i \textit{ are eligible } (1 \leq i \leq n). \end{array}\right.$$

Ordered Factoring

$$OF\colon \frac{C \vee \underline{A}^* \vee \underline{A'}}{C\sigma \vee A\sigma} \;\left|\; \begin{array}{l} \textit{where (i) } \sigma = mgu(A, A')\textit{, (ii) } A \\ \textit{is eligible.} \end{array}\right.$$

The conventional **Ordered Resolution** rule OR, is a partial case of the ordered (hyper-)resolution rule when $n = 1$. The calculus $\mathcal{OR}^{\succ}_{Sel}$ is refutationally complete for any choice of an admissible ordering $\succ$ and a selection function Sel. Moreover, the calculus is compatible with a general notion of redundancy which allows to make use of additional simplification rules.

Schemes of expressions and clauses. To describe resolution-based decision procedures we have to reason about sets of clauses. We introduce a special notation that allows to represent sets of clauses in a compact form. We extend our vocabulary with additional symbols called *signature groups* (denoted by symbols with a "hat" $\hat{g}$) that represent sets of signature elements. We build *expressions schemes* and *clause schemes* by using them as usual elements of signature in expressions and clauses. Each expression (clause) scheme represents a set of expressions (clauses). For instance, if $\hat{f}_{all}$ denotes the set of all functional symbols, the scheme $\hat{f}_{all}(t)$ represents any term of the form $f(t)$ where $f \in \hat{f}_{all}$. We adopt the following notation for referring to arguments of expressions. By writing $\hat{e}\langle !t_1, ..., !t_n, s_1, ..., s_m\rangle$ we mean an expression starting with the expression symbol $e \in \hat{e}$, having all arguments $t_1, ..., t_n$ and optional arguments $s_1, ..., s_m$ (arranged in some way). The formal definitions for clause schemes is given in the extended version of the paper (de Nivelle & Kazakov 2004).

Example 1. Suppose $\hat{a}$ is a predicate group consisting of all predicate symbols and $\hat{\alpha} := \{\hat{a}, \neg\hat{a}\}$ is a *literal group* consisting of all (negated) predicate symbols. The clause scheme $\hat{C} = \neg !\hat{a}\langle !\overline{x}\rangle \vee \hat{\alpha}\langle !f(\overline{x}), \overline{x}\rangle$ represents any clause C such that ***(i)*** C has a nonempty subset of negative literals containing all variables x_1,..., x_n and no other arguments and ***(ii)*** other literals of C contain a subterm $f'(x_1,..., x_n)$ as an argument and possibly some variables from $x_1,..., x_n$. In particular, $\langle\hat{C}\rangle$ contains the clauses $\neg a(x, y, x) \vee b(y, f'(x, y))$, $\neg b(x, y) \vee \neg b(y, x)$ and $\neg p \vee \neg q(c, c)$, but not the clauses $\neg a(x, y, x) \vee b(f'(x, y), f'(y, x))$ or $\neg b(y, f'(x, y))$.

[1] For a description of the calculus and its variations see (Bachmair & Ganzinger 2001)

2 How to obtain a resolution decision procedure?

We demonstrate how a resolution decision procedure for the guarded fragment (de Nivelle & de Rijke 2003) can be specified using our notation. For describing the procedure it is convenient to use the *recursive definition* for the guarded fragment: $\mathcal{GF} ::= \mathtt{A} \mid \mathtt{F}_1 \lor \mathtt{F}_2 \mid \mathtt{F}_1 \land \mathtt{F}_2 \mid \neg\mathtt{F}_1 \mid \forall\overline{x}.(\mathtt{G}\to\mathtt{F}_1) \mid \exists\overline{x}.(\mathtt{G}\land\mathtt{F}_1)$. where $\mathtt{A}$ is an atom, $\mathtt{F}_i, i = 1, 2$ are guarded formulas, and $\mathtt{G}$ is an atom called *the guard* containing all free variables of $\mathtt{F}_1$. Then the guarded formulas in the negation normal form can be defined by:

$$[\mathcal{GF}]^{nnf} ::= (\neg)\mathtt{A} \mid \mathtt{F}_1 \lor \mathtt{F}_2 \mid \mathtt{F}_1 \land \mathtt{F}_2 \mid \forall\overline{y}.(\mathtt{G}\to\mathtt{F}_1) \mid \exists\overline{y}.(\mathtt{G}\land\mathtt{F}_1).$$

The important step of CNF transformation is a so-called *structural transformation*, that decomposes the formula by introducing *definitions* for all of its subformulae. We assume that to each subformula $\mathtt{F}'$ of $\mathtt{F}$, a unique predicate $P_{\mathtt{F}'} = p_{\mathtt{F}'}(\overline{x})$ is assigned. The structural transformation is defined as $\exists\overline{x}.P_{\mathtt{F}} \lor [\mathtt{F}]^{st}$, where $\mathtt{F}$ $[\mathtt{F}]^{st}$ is given below. In each row, $\overline{x}$ are the free variables of $\mathtt{F}$.

$$\begin{array}{rl}
[\mathtt{F}]_g^{st} := [(\neg)\mathtt{A}]_g^{st} : \forall\overline{x}.(P_{\mathtt{F}} \to (\neg)\mathtt{A}) & \mid \quad \neg p_{\mathtt{F}}(\overline{x}) \lor (\neg)a\langle\overline{x}\rangle \\
[\mathtt{F}_1 \between \mathtt{F}_2]_g^{st} : \forall\overline{x}.(P_{\mathtt{F}} \to [P_{\mathtt{F}_1} \between P_{\mathtt{F}_2}]) \land [\mathtt{F}_1]_g^{st} \land [\mathtt{F}_2]_g^{st} & \mid \quad \neg p_{\mathtt{F}}(\overline{x}) \lor p_{\mathtt{F}_i}\langle\overline{x}\rangle \ [\lor\ p_{\mathtt{F}_j}\langle\overline{x}\rangle] \\
[\forall\overline{y}.(\mathtt{G}\to\mathtt{F}_1)]_g^{st} : \forall\overline{x}.(P_{\mathtt{F}} \to \forall\overline{y}.[\mathtt{G}\to P_{\mathtt{F}_1}]) \land [\mathtt{F}_1]_g^{st} & \mid \quad \neg g\langle !\overline{x}, !\overline{y}\rangle \lor \neg p_{\mathtt{F}}(\overline{x}) \lor p_{\mathtt{F}_1}\langle\overline{x}, \overline{y}\rangle \\
[\exists y.\mathtt{F}_1]_g^{st} : \forall\overline{x}.(P_{\mathtt{F}} \to \exists y.P_{\mathtt{F}_1}) \land [\mathtt{F}_1]_g^{st}. & \quad\ \ \neg p_{\mathtt{F}}(\overline{x}) \lor p_{\mathtt{F}_1}\langle f(\overline{x}), !\overline{x}\rangle
\end{array}$$

The transformation produce an equisatisfiable formula that is a conjunction of simple *definitions* of the form $\forall x.(P_{\mathtt{F}} \to \dots)$. Applying the usual skolemization and writing the result in a clause form, we obtain the clauses shown to the right of the definition for $[\mathtt{F}]_g^{st}$. It is easy to see that the clauses for $P_{\mathtt{F}} \land [\mathtt{F}]_g^{st}$ fall into the set of clauses described by the following clause schemes:

$$\begin{array}{l} 1.\ \hat{\alpha}\langle\hat{c}\rangle; \\ 2.\ \neg!\hat{a}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle. \end{array} \tag{G}$$

where the predicate group $\hat{a}$ consists of all (initial and introduced) predicate symbols and the literal group $\hat{\alpha}$ consists of all literal symbols.

Saturation of the clause set given by the schemes from (G) can be compactly described using the scheme notation:

1 $\hat{\alpha}^*$	2 $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle$
1.1 $\hat{\alpha}\langle\hat{c}\rangle \lor \underline{\hat{a}\langle\hat{c}\rangle}^*$: $\mathtt{OR.1}$ 1.2 $\hat{\alpha}\langle\hat{c}\rangle \lor \neg\overline{\hat{a}\langle\hat{c}\rangle}^*$: $\mathtt{OR.2}$ 1.3 $\hat{\alpha}\langle\hat{c}\rangle \lor \underline{\hat{a}\overline{\langle\hat{c}\rangle}}^* \lor \underline{\hat{a}\langle\hat{c}\rangle}$: $\mathtt{OF}$ $\mathtt{OR}[1.1; 1.2]$: $\hat{\alpha}\langle\hat{c}\rangle$: 1 $\mathtt{OF}[1.3]$: $\hat{\alpha}\langle\hat{c}\rangle \lor \hat{a}\langle\hat{c}\rangle$: 1	2.1 $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle \lor \hat{\alpha}\langle !f(\overline{x}), \overline{x}\rangle^*$
	2.1.1 $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle \lor \underline{\hat{a}\langle !f(\overline{x}), \overline{x}\rangle}^*$: $\mathtt{OR.1}$ 2.1.2 $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle \lor \overline{\neg\hat{a}\langle !f(\overline{x}), \overline{x}\rangle}^*$: $\mathtt{OR.2}$ 2.1.3 $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle \lor \underline{\hat{a}\overline{\langle !f(\overline{x}), \overline{x}\rangle}}^* \lor \underline{\hat{a}\langle f(\overline{x}), \overline{x}\rangle}$: $\mathtt{OF}$ $\mathtt{OR}[2.1.1; 2.1.2]$: $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle$: 2 $\mathtt{OF}[2.1.3]$: $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle \lor \hat{a}\langle !f(\overline{x}), \overline{x}\rangle$: 2
	2.2 $\neg\underline{\hat{g}\langle !\overline{x}\rangle}^* \lor \neg\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle\overline{x}\rangle$: $\mathtt{OR.2}$ $\mathtt{OR}[1.1; 2.2]$: $\hat{\alpha}$: 1 $\mathtt{OR}[2.1.1; 2.2]$: $\neg!\hat{g}\langle !\overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle \lor \hat{\alpha}\langle f(\overline{x}), \overline{x}\rangle$: 2

The table represents the case analysis of possible resolution inferences between the clauses from (G). The clause schemes are spread in the table on different levels of precision, according to the possibilities for eligible literal and application of inference rules. For example, $\texttt{OR.1}$ marked to the right of the clause scheme 1.1 means that a clause represented by this scheme may act as a first premise of the ordered resolution rule. Below the last level, inferences between preceding clauses are drawn and their conclusions are identified as instances of clause schemes. The table shows that the clause class (G) is closed under ordered resolution, therefore, decidability of $\mathcal{GF}$ follows from the fact that, for any guarded formula, the resolution procedure can generate only clauses from (G), which is a finite set for a fixed signature.

2.1 Deciding the guarded fragment with transitivity

Extending the above procedure to the guarded fragment with transitivity is not a trivial task, since resolution with transitivity axioms may produce infinitely many clauses in simple cases:

1. $\neg(\underline{xTy})^* \vee \neg(yTz) \vee xTz$;
2. $\alpha(x) \vee \underline{f(x)Tx}^*$;
$\texttt{OR}[2;1]$: 3. $\alpha(x) \vee \neg(xTz) \vee \underline{f(x)Tz}^*$;
$\texttt{OR}[3;1]$: 4. $\alpha(x) \vee \neg(xTz) \vee \neg(zTz_1) \vee \underline{f(x)Tz_1}^*$;
$\ldots\ldots\vdots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

1. $\neg(\underline{xTy})^* \vee \neg(\underline{yTz})^* \vee xTz$;
2. $\alpha(x) \vee \underline{f(x)Tx}^*$;
$\texttt{HR}[2,2;1]$: 3. $\alpha(x) \vee \underline{ff(x)Tx}^*$;
$\texttt{HR}[3,2;1]$: 4. $\alpha(x) \vee \underline{fff(x)Tx}^*$;
$\ldots\ldots\ldots\vdots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

To resolve these problems we consider the *constrained version* of the *transitivity clause* and use the constrains to show the redundancy of dangerous inferences:

$\texttt{T}$ $\quad\neg(xTy) \vee \neg(yTz) \vee xTz$;
$\texttt{T.1.}$ $\neg(\underline{xTy})^* \vee \neg(yTz) \vee xTz \quad | \; x \succ max(y,z)$;
$\texttt{T.2.}$ $\neg(xTy) \vee \neg(\underline{yTz})^* \vee xTz \quad | \; z \succ max(y,x)$;
$\texttt{T.3.}$ $\neg(\underline{xTy})^* \vee \neg(\underline{yTx})^* \vee xTx \; | \; x \succ y$;
$\texttt{T.4.}$ $\neg(\underline{xTy})^* \vee \neg(\underline{yTz})^* \vee xTz \; | \; y \succeq max(x,z)$;

For every constraint variant of the transitivity clause we set the selection function *Sel* to select the negative literals containing the maximal variable of the clause. This immediately prevents the situation shown on the right example above, since the constraints for the inference are not satisfiable. The following lemma resolves the problem shown on the left example above:

Lemma 1. *Let N be a clause set containing the clause:*

1. $C \vee \underline{t_1Tt_2}^*$; *together with the result of the inference:*
$\texttt{OR}[1;\texttt{T.1}]$*:* 2. $C \vee \neg(t_2Tz) \vee \underline{t_1Tz}^* \; | \; t_1 \succ max(t_2,z)$;

Then the following inference is redundant:

$\texttt{OR}[2;\texttt{T.1}]$*:* $C \vee \neg(t_2Tz) \vee \neg(zTz_1) \vee t_1Tz_1 \; | \; t_1 \succ max(t_2,z,z_1)$;

One could see that the inference $\texttt{OR}[3;1]$ above is redundant by Lemma 1. Similar lemmas can block other dangerous inferences with transitivity. Redundancy and ordering constraints, however, do not help avoiding situations as the one below:

1. $\alpha(x) \vee \underline{f(x)Tx}^*$;
2. $\neg(\underline{xTy})^* \vee a(x) \vee \beta(y)$;
3. $\neg(\underline{xTy})^* \vee \neg a(x) \vee \beta'(y)$;
$\texttt{OR}[1;\texttt{T.1}]$: 5. $\alpha(x) \vee \neg(xTz) \vee \underline{f(x)Tz}^* \; | \; f(x) \succeq max(x,z)$;
$\texttt{OR}[5;2]$: 6. $\alpha(x) \vee \neg(xTz) \vee \underline{a(f(x))}^* \vee \beta(z) \; | \; f(x) \succeq max(x,z)$;
$\texttt{OR}[5;3]$: 7. $\alpha(x) \vee \neg(xTz_1) \vee \underline{\neg a(f(x))}^* \vee \beta'(z_1) \; | \; f(x) \succeq max(x,z_1)$;
$\texttt{OR}[6;7]$: 8. $\alpha(x) \vee \neg(xTz) \vee \neg(xTz_1) \vee \beta(z) \vee \beta'(z_1) \; | \; f(x) \succeq max(x,z,z_1)$;
$\ldots\ldots\ldots\vdots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$

The problem here is that the maximal literal of a clauses 6 and 7 does not contain all variables of the clause – the main argument for showing decidability by resolution is violated. To avoid the inferences OR[5; 2] and OR[5; 3] that produced these clauses we have designed a special inference rule:

Transitive Recursion

$$TR\colon \frac{\neg(x\hat{T}y)^* \vee \alpha(x) \vee \beta(y)}{\begin{array}{c} \neg(x\hat{T}y) \vee \alpha(x) \vee u^{\hat{T}}_{\alpha(\cdot)}(y) \\ \neg(x\hat{T}y) \vee \neg u^{\hat{T}}_{\alpha(\cdot)}(x) \vee u^{\hat{T}}_{\alpha(\cdot)}(y) \\ \neg u^{\hat{T}}_{\alpha(\cdot)}(y) \vee \beta(y) \end{array}} \left| \begin{array}{l} \textit{where (i) } \hat{T} \textit{ is a not empty set of} \\ \textit{transitive predicates (ii) } u^{\hat{T}}_{\alpha} \textit{ is a} \\ \textit{special unary predicate indexed by} \\ \alpha \textit{ and } \hat{T}. \end{array} \right.$$

This inference rule does not make the clauses 2 and 3, to which it applies, redundant, but it makes redundant the inferences OR[5; 2] and OR[5; 3] with them. Collecting all these refinements together, we have designed a resolution-based decision procedure for $\mathcal{GF}[\mathcal{TG}]$ of the optimal complexity (for details see de Nivelle & Kazakov 2004). In particular, we have reproved the following theorem:

Theorem 1. (Szwast & Tendera 2001) *The guarded fragment with transitive guards is decidable in double exponential time.*

3 Conclusions and Future Work

The resolution decision procedure for $\mathcal{GF}[\mathcal{TG}]$ presented in the paper can shed light on the reasons why this fragment is so fragile with respect to decidability and which decidable extensions it may have. In fact we have shown decidability of a larger fragment: it is possible to admit non-empty *conjunctions* of transitive relations as guards. As a future work we try to extend our approach for other theories like theories of general compositional axioms $\forall xyz.(xSy \wedge yTz \to xHz)$ and theories of linear, branching and dense total orderings without endpoints.

References

Andréka, H., van Benthem, J. & Németi, I. (1998), 'Modal languages and bounded fragments of predicate logic', *Journal of Philosophical Logic* **27**, 217–274.

Bachmair, L. & Ganzinger, H. (2001), Resolution theorem proving, *in* A. Robinson & A. Voronkov, eds, 'Handbook of Automated Reasoning', Vol. I, Elsevier Science, chapter 2, pp. 19–99.

de Nivelle, H. & de Rijke, M. (2003), 'Deciding the guarded fragments by resolution', *Journal of Symbolic Computation* **35**, 21–58.

de Nivelle, H. & Kazakov, Y. (2004), Resolution decision procedures for the guarded fragment with transitive guards, Research Report MPI-I-2004-2-001, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany.

Grädel, E. (1999), 'On the restraining power of guards', *Journal of Symbolic Logic* **64(4)**, 1719–1742.

Szwast, W. & Tendera, L. (2001), On the decision problem for the guarded fragment with transitivity, *in* 'Proc. 16th IEEE Symposium on Logic in Computer Science', pp. 147–156.