A NOTE ON GLOBAL INDUCTION MECHANISMS IN A μ-CALCULUS WITH EXPLICIT APPROXIMATIONS

CHRISTOPH SPRENGER AND MADS DAM

1. Introduction

The first-order μ-calculus [7] provides a useful setting for semi-automatic program verification. It is expressive enough to encode, from the bottom up, a range of program logics (e.g. LTL, CTL, CTL*, Hoare Logic) as well as process calculi and programming languages including their data types and operational semantics. A framework based on this idea is described in [4] and has been applied to a substantial part of a real concurrent programming language Erlang in the Erlang Verification Tool [1].

A key aspect in the design of such a framework is proof search, in particular the handling of fixed point formulas. The standard approach, Park's fixed point induction rule (cf. [5]), is not suitable for proof search in practice. An alternative is to admit cyclic proof structures (cf. [6, 10, 3]) and look for sound discharge conditions which will ensure the well-foundedness of the inductive reasoning. In this paper, we study such discharge conditions in the context of a Gentzen-style proof system for the first-order μ-calculus, which is a variant of previous systems [3, 4, 8, 9]. In particular, it shares with [3, 4, 8] the technique, first proposed for the modal μ-calculus in [3], of introducing explicit approximation ordinal variables and ordering constraints between them into the proof system. Discharge conditions then rely on these ordering constraints. The use of approximation ordinals considerably simplifies earlier treatments (cf. [2]). A simple semantic condition was proposed in [3] expressing in a natural way the requirement of well-foundedness of all inductive reasoning. Due to its semantical nature it is not suitable for the purpose of practical proof. However, it serves as a useful reference to which other, more syntactical conditions may be compared.

Previous syntactic discharge conditions [3, 4, 8] turn out to be strictly stronger than the semantic condition. The main contribution of the present paper is the formulation of a syntactic condition that precisely matches the semantic one. We introduce *traces* which are non-increasing (w.r.t. the ordering constraints) sequences of dependent ordinal variables running along a path in the proof structure. They can be seen as a generalisation of the μ- and ν-traces in [6] to the setting of explicit approximants. The characterisation result is then obtained by introducing the notion of progress for traces and by establishing its close correspondence with the semantic notion of well-foundedness. For practical application we then give an automata-theoretic formulation of our discharge condition in terms of an inclusion of the languages recognised by two Büchi automata.

Finally, we remark that this work is part of a larger programme aiming at clarifying the relation between the global, lazy induction mechanism used here and an eager induction discipline as implemented in a local proof rule for well-founded induction on the ordinals.

2. Logic

The logic we consider augments first-order logic with two fixed point operators, a standard and an approximated one parametrised by an ordinal variable. Formulas ϕ and abstractions Φ_X of the μ-calculus over a first-order signature Σ are inductively defined as follows

$$\phi ::= t = t' \mid \neg\phi \mid \phi_1 \vee \phi_2 \mid \exists x.\phi \mid \Phi_X(\overline{t}) \qquad \text{and} \qquad \Phi_X ::= X \mid \mu X(\overline{x}).\phi \mid \mu^\kappa X(\overline{x}).\phi$$

where t ranges over Σ-terms, x over individual variables, κ over ordinal variables and X over predicate variables. Each abstraction Φ_X has the arity of X which is required to match the lengths of the vectors $\overline{x}$ and $\overline{t}$ of individual variables and terms, respectively. Fixed point abstraction formation is subject to the usual syntactic monotonicity condition.

A *model* $\mathcal{M}$ is a pair $(\mathcal{A},\rho)$ where $\mathcal{A}$ is a first-order Σ-structure with support set A and ρ is an *$\mathcal{A}$-environment* interpreting each variable according to its type. Formulas are interpreted as elements of the two-point lattice $\mathbf{2} = \{0,1\}$ and n-ary abstractions as elements of $\mathsf{Pred}(A^n) = \mathbf{2}^{A^n}$, the lattice n-ary predicates over A under the pointwise ordering. For first-order formulas the semantics is as expected. For abstractions and their application we have

$$\|X\|^{\mathcal{A}}_{\rho} = \rho(X) \quad \|\mu X(\overline{x}).\phi\|^{\mathcal{A}}_{\rho} = \mu\Psi \quad \|\mu^{\kappa} X(\overline{x}).\phi\|^{\mathcal{A}}_{\rho} = \mu^{\rho(\kappa)}\Psi \quad \|\Phi_X(\overline{t})\|^{\mathcal{A}}_{\rho} = \|\Phi_X\|^{\mathcal{A}}_{\rho}(\|\overline{t}\|^{\mathcal{A}}_{\rho})$$

where $\Psi = \lambda P.\lambda\overline{a}.\|\phi\|^{\mathcal{A}}_{\rho[P/X,\overline{a}/\overline{x}]} : \mathsf{Pred}(A^n) \to \mathsf{Pred}(A^n)$, $\mu\Psi$ is the least fixed point of the (monotone) function Ψ and $\mu^{\rho(\kappa)}\Psi$ is the approximation $\rho(\kappa)$ of $\mu\Psi$, both defined as usual.

3. Proof Rules

Sequents of the proof system are of the form $\Gamma \vdash_{\mathcal{O}} \Delta$, where Γ and Δ are finite multi-sets of μ-calculus formulas and $\mathcal{O} = (|\mathcal{O}|, <_{\mathcal{O}})$ is a strict partial order on a finite set $|\mathcal{O}|$ of ordinal variables. Following [8] the latter is used to record constraints between ordinal variables. An environment ρ *respects* $\mathcal{O}$ if $\rho(\iota) < \rho(\kappa)$ whenever $\iota <_{\mathcal{O}} \kappa$. A sequent $\Gamma \vdash_{\mathcal{O}} \Delta$ is *valid* if the formula $\bigwedge\Gamma \to \bigvee\Delta$ is satisfied in all models $\mathcal{M} = (\mathcal{A},\rho)$, where ρ respects $\mathcal{O}$.

The proof system extends the standard Gentzen-style rules for first-order logic with the following rules for the fixed point operators (we omit an ordinal constraint strengthening rule [8] for brevity):

$$(\mu\text{-L}) \qquad \frac{\Gamma, (\mu^{\kappa} X(\overline{x}).\phi)(\overline{t}) \vdash_{\mathcal{O}'} \Delta}{\Gamma, (\mu X(\overline{x}).\phi)(\overline{t}) \vdash_{\mathcal{O}} \Delta} \qquad \kappa \notin |\mathcal{O}|,\ \mathcal{O}' = (|\mathcal{O}| \cup \{\kappa\}, <_{\mathcal{O}})$$

$$(\mu\text{-R}) \qquad \frac{\Gamma \vdash_{\mathcal{O}} \phi[\mu X(\overline{x}).\phi/X,\ \overline{t}/\overline{x}], \Delta}{\Gamma \vdash_{\mathcal{O}} (\mu X(\overline{x}).\phi)(\overline{t}), \Delta}$$

$$(\mu^{\kappa}\text{-L}) \qquad \frac{\Gamma, \phi[\mu^{\kappa'} X(\overline{x}).\phi/X,\ \overline{t}/\overline{x}] \vdash_{\mathcal{O}'} \Delta}{\Gamma, (\mu^{\kappa} X(\overline{x}).\phi)(\overline{t}) \vdash_{\mathcal{O}} \Delta} \qquad \kappa' \notin |\mathcal{O}|,\ \mathcal{O}' = (|\mathcal{O}| \cup \{\kappa'\}, (<_{\mathcal{O}} \cup \{(\kappa',\kappa)\})^+)$$

$$(\mu^{\kappa}\text{-R}) \qquad \frac{\Gamma \vdash_{\mathcal{O}} \phi[\mu^{\kappa'} X(\overline{x}).\phi/X,\ \overline{t}/\overline{x}], \Delta}{\Gamma \vdash_{\mathcal{O}} (\mu^{\kappa} X(\overline{x}).\phi)(\overline{t}), \Delta} \qquad \kappa' <_{\mathcal{O}} \kappa$$

4. Discharge Conditions

A derivation may be stopped at a leaf node N if N is a substitution instance of some node M in the derivation tree. It is worth noting that M need not lie on the path from the root node to N. We write $N(\Gamma \vdash_{\mathcal{O}} \Delta)$ to mean that N is labeled by the sequent $\Gamma \vdash_{\mathcal{O}} \Delta$. Given a node $M(\Gamma \vdash_{\mathcal{O}} \Delta)$ and a leaf $N(\Gamma' \vdash_{\mathcal{O}'} \Delta')$ of a derivation tree and a substitution σ, we say that $R = (M, N, \sigma)$ is a *repeat,* if $\Gamma\sigma \subseteq \Gamma'$, $\Delta\sigma \subseteq \Delta'$ and $\mathcal{O}\sigma \subseteq \mathcal{O}'$, where σ is order-preserving in the latter inclusion. We assume that σ maps predicate variables to predicate variables. N is called a *repeat node* and M its *companion.* A *pre-proof* $\mathcal{P} = (\mathcal{D},\mathcal{R})$ for $\Gamma \vdash_{\mathcal{O}} \Delta$ is a derivation tree $\mathcal{D}$ whose root node is labeled by $\Gamma \vdash_{\mathcal{O}} \Delta$ together with a set of repeats $\mathcal{R}$ such that each non-axiom leaf of $\mathcal{D}$ belongs to exactly one repeat in $\mathcal{R}$. A *path* $\pi = N_0 \cdots N_i \cdots$ of $\mathcal{P}$ is a (finite or infinite) sequence of nodes of $\mathcal{D}$ starting in the root N_0 of $\mathcal{D}$ such that for any two successive nodes N_i and N_{i+1} either (N_i, N_{i+1}) is an edge of $\mathcal{D}$ or there is a substitution σ such that $(N_{i+1}, N_i, \sigma) \in \mathcal{R}$ is a repeat.

Intuitively, each repeat should correspond to the application of an induction hypothesis, but we need to make sure that the inductive reasoning embodied in each individual repeat and their combinations is indeed well-founded. This requires a (necessarily global) *discharge condition* that identifies the legal proofs. We present three such conditions, the semantical one from [3], a new syntactical condition as well as its automata-theoretic formulation and establish their equivalence.

Let $\mathcal{P} = (\mathcal{D},\mathcal{R})$ be a pre-proof, $\mathcal{A}$ a Σ-structure and $\Pi = (N_0,\rho_0)\cdots(N_i,\rho_i)\cdots$ a (finite or infinite) sequence of pairs of nodes $N_i(\Gamma_i \vdash_{\mathcal{O}_i} \Delta_i)$ of $\mathcal{D}$ and $\mathcal{A}$-environments ρ_i. Then Π is called a *run* of $\mathcal{P}$ if N_0 is the root of $\mathcal{D}$, each ρ_i respects $\mathcal{O}_i$ and for each pair (N_i, N_{i+1}) either

(1) (N_i, N_{i+1}) is an edge of $\mathcal{D}$ and ρ_{i+1}, ρ_i agree on free variables common to N_i and N_{i+1}, or
(2) $(N_{i+1}, N_i, \sigma) \in \mathcal{R}$ is a repeat and $\rho_{i+1} = \rho_i \circ \sigma$.

The run Π is said to follow the path $\pi = N_0 \cdots N_i \cdots$. A pre-proof $\mathcal{P}$ for $\Gamma \vdash_{\mathcal{O}} \Delta$ satisfies *discharge condition (R-DC)* if all runs of $\mathcal{P}$ are finite, in which case $\mathcal{P}$ is called a *proof* for $\Gamma \vdash_{\mathcal{O}} \Delta$.

Theorem 1. **(Soundness)** *If there is a proof for* $\Gamma \vdash_{\mathcal{O}} \Delta$ *then* $\Gamma \vdash_{\mathcal{O}} \Delta$ *is valid.*

As condition (R-DC) captures the well-foundedness of the reasoning in a pre-proof in a very natural way, it serves as our reference discharge condition. Due to its semantical nature it is, however, hardly usable in practical proofs and we therefore introduce an alternative, purely syntactical, discharge condition.

Let $\tau = (N_0, w_0) \cdots (N_i, w_i) \cdots$ be a (finite or infinite) sequence of pairs of nodes $N_i(\Gamma_i \vdash_{\mathcal{O}_i} \Delta_i)$ of $\mathcal{D}$ and non-empty words w_i over the alphabet of ordinal variables. Then τ is called a *trace* of $\mathcal{P}$ if $w_i(j) \in |\mathcal{O}_i|$ and $w_i(j+1) <_{\mathcal{O}_i} w_i(j)$ for all i and j and for each pair (N_i, N_{i+1}) either

(1) (N_i, N_{i+1}) is an edge of $\mathcal{D}$ and $w_i(*) = w_{i+1}(0)$, or

(2) (N_{i+1}, N_i, σ) is a repeat in $\mathcal{R}$ and $w_i(*) = \sigma(w_{i+1}(0))$,

where $w_i(*)$ denotes the last letter of w_i. Intuitively, a trace τ records a sequence of vertical dependencies (w_i is a descending chain in $\mathcal{O}_i$) and horizontal dependencies (linking $\mathcal{O}_i$ and $\mathcal{O}_{i+1}$ as in (1), (2)) between ordinal variables. So τ roughly associates a non-increasing chain of ordinal variables with (a suffix of) a path. We say that a trace τ *progresses* at position i if $|w_i| > 1$ and that τ is *progressive* if it progresses at infinitely many positions. A path π of $\mathcal{P}$ is said to be *progressive* if there is a progressive trace $\tau = (N_0, w_0)(N_1, w_1) \cdots$ such that $N_0 N_1 \cdots$ is a suffix of π. Our syntactic *discharge condition (T-DC)* requires that all infinite paths of $\mathcal{P}$ are progressive. If an infinite path π is progressive, as witnessed by some τ, then it cannot be followed by an infinite run, as respecting the dependencies in τ would lead to an infinite decreasing chain of ordinals. Conversely, π can be extended to an infinite run if there is no progressive trace following π. Thus,

Theorem 2. *A pre-proof satisfies (R-DC) if and only if it satisfies (T-DC).*

We now turn to an automata-theoretic formulation of these conditions, which is more suitable for the practical application in a proof tool. Let $\mathcal{P} = (\mathcal{D}, \mathcal{R})$ be a pre-proof. Since $\mathcal{O}_i \subseteq \mathcal{O}_{i+1}$ whenever (N_i, N_{i+1}) is an edge of $\mathcal{D}$, we may without loss of generality for condition (T-DC) restrict our attention to the *normal* traces of $\mathcal{P}$ making progress at most at repeat nodes. Based on this observation we construct two Büchi automata over the alphabet $\mathcal{R}$. Automaton B_1 recognises sequences of repeats that are traversed on paths of $\mathcal{P}$. Automaton B_2 recognises sequences of repeats that are from some point on connected through the ordinal variables they preserve. More precisely, let $s = R_0 R_1 \cdots$ be a sequence of repeats with $R_i = (M_i, N_i, \sigma_i)$ and $N_i(\Gamma_i \vdash_{\mathcal{O}_i} \Delta_i)$. Then $r = \bullet^j (\kappa_j, R_j, \kappa_{j+1})(\kappa_{j+1}, R_{j+1}, \kappa_{j+2})(\kappa_{j+2}, R_{j+2}, \kappa_{j+3}) \cdots$ is a run of B_2 on s if $j \geq 0$ and $\sigma_i(\kappa_{i+1}) \leq_{\mathcal{O}_i} \kappa_i$ for all $i \geq j$. It is accepting if $\sigma_i(\kappa_{i+1}) <_{\mathcal{O}_i} \kappa_i$ for infinitely many i. Note that if $u \in L(B_1)$ then r induces a normal trace, which is progressive precisely if r is accepting. Hence,

Theorem 3. *A pre-proof satisfies (T-DC) if and only if* $L(B_1) \subseteq L(B_2)$.

References

[1] T. Arts, M. Dam, L. Fredlund, and D. Gurov. System description: Verification of distributed Erlang programs. In *Proc. CADE'98*, volume 1421 of *Lecture Notes in Artificial Intelligence*, pages 38–41, 1998.

[2] M. Dam. Proving properties of dynamic process networks. *Information and Computation*, 140:95–114, 1998.

[3] M. Dam and D. Gurov. μ-calculus with explicit points and approximations. In *Fixed Points in Computer Science, FICS 2000*, 2000.

[4] L. Fredlund. *A Framework for Reasoning about Erlang Code.* PhD thesis, Royal Institute of Technology, Stockholm, Sweden, 2001.

[5] D. Kozen. Results on the propositional μ-calculus. *Theoretical Computer Science*, 27:333–354, 1983.

[6] D. Niwiński and I. Walukiewicz. Games for the μ-calculus. *Theoretical Computer Science*, 163:99–116, 1997.

[7] D. Park. Finiteness is mu-ineffable. *Theoretical Computer Science*, 3(2):173–181, 1976.

[8] U. Schöpp. Formal verification of processes. Master's thesis, University of Edinburgh, 2001.

[9] U. Schöpp and A. Simpson. Verifying temporal properties using explicit approximants: Completeness for context-free processes. In *FOSSACS '02*, Lecture Notes in Computer Science. Springer-Verlag, 2002. to appear.

[10] C. Stirling and D. Walker. Local model checking in the modal μ-calculus. *Theoretical Computer Science*, 89:161–177, 1991.

Christoph Sprenger, Swedish Institute of Computer Science (SICS), Kista, Sweden
E-mail address: `sprenger@sics.se`

Mads Dam, Royal Institute of Technology (KTH), Kista, Sweden
E-mail address: `mfd@kth.se`