Computational Logic for Computer Science

Mauricio Ayala-Rincón & Flávio L.C. de Moura

Grupo de Teoria da Computação, Universidade de Brasília (UnB)

Brasília D.F., Brazil

Research funded by

Brazilian Research Agencies: CNPq, CAPES and FAPDF

11^o Seminário Informal (+ Formal!) GTC/UnB
Brasília, Nov 28th, 29th 2013

Universidade de Brasília

Talk's Plan

Motivation: formalization - proofs & deduction

Computational proofs - logic & deduction
- Deduction Natural
- Deduction à la Gentzen

Formal proofs — Proofs in the Prototype Verification System - PVS

Formalizations versus programs
- A very very simple case study: insertion sort

Conclusions and Future Work

Motivation: classic teaching approach

Propositional logic
Semantic entailment vs deduction - completeness
Predicate logic
Semantic entailment vs deduction - completeness
Undecidability
Compactness and Löwenheim-Skolem theorems
Resolution

The focus on understanding formal logic notions gives no time for a careful analysis of deduction technologies and their usefulness in CS.

Motivation: computational teaching approach

Propositional and Predicate logic
Semantic entailment vs deduction - completeness
Tableaux
Sat solvers
Resolution
Model cheking
Formal Verification

The focus on teaching a variety of deduction approaches gives no time for assimilating the related technologies - the more ..., the less ...

Motivation: computational logic for CS

Induction and recursion

Classical and Intiutionistic Propositional and Predicate logic

Semantic entailment vs deduction - completeness

Natural deduction vs Gentzen Calculus

Program verification with induction and first-order deduction

Restricting computational logic to understand deduction and how this is applied in programming languages and program verification increases interest of CS and engineer students.

Computational proofs - logic & deduction

Table : Natural deduction for intuitionistic propositonal logic

introduction rules	elimination rules
$\dfrac{\varphi \quad \psi}{\varphi \wedge \psi}\ (\wedge_i)$	$\dfrac{\varphi \wedge \psi}{\varphi}\ (\wedge_e)$
$\dfrac{\varphi}{\varphi \vee \psi}\ (\vee_i)$	$\dfrac{\varphi \vee \psi \qquad \begin{matrix}[\varphi]^u \\ \vdots \\ \chi\end{matrix} \qquad \begin{matrix}[\psi]^v \\ \vdots \\ \chi\end{matrix}}{\chi}\ (\vee_e)\ u, v$
$\dfrac{\begin{matrix}[\varphi]^u \\ \vdots \\ \psi\end{matrix}}{\varphi \to \psi}\ (\to_i)\ u$	$\dfrac{\varphi \quad \varphi \to \psi}{\psi}\ (\to_e)$
$\dfrac{\begin{matrix}[\varphi]^u \\ \vdots \\ \bot\end{matrix}}{\neg\varphi}\ (\neg_i)\ u$	$\dfrac{\varphi \quad \neg\varphi}{\bot}\ (\neg_e)$

Computational proofs - logic & deduction

Table : Natural deduction for classical predicate logic

introduction rules	elimination rules
	$\dfrac{\begin{array}{c}[\neg\varphi]^u \\ \vdots \\ \bot\end{array}}{\varphi}$ (PBC) u
$\dfrac{\varphi\{x/x_0\}}{\forall_x \varphi}$ $(\forall_i)$ where x_0 cannot occur free in any open assumption.	$\dfrac{\forall_x \varphi}{\varphi\{x/t\}}$ $(\forall_e)$
$\dfrac{\varphi\{x/t\}}{\exists_x \varphi}$ $(\exists_i)$	$\dfrac{\exists_x \varphi \qquad \begin{array}{c}[\varphi\{x/x_0\}]^u \\ \vdots \\ \chi\end{array}}{\chi}$ $(\exists_e)$ u where x_0 cannot occur free in any open assumption on the right and in χ.

Mathematical proofs - logic & deduction

Table : Encoding $\neg$ - Rules of natural deduction for classical logic

introduction rules	elimination rules
$$\dfrac{\begin{matrix}[\varphi]^u \\ \vdots \\ \bot\end{matrix}}{\neg\varphi}\ (\neg_i), u$$	$$\dfrac{\varphi \quad \neg\varphi}{\bot}\ (\neg_e)$$

$$\dfrac{\begin{matrix}[\varphi]^u \\ \vdots \\ \bot\end{matrix}}{\varphi \to \bot}\ (\to_i), u \qquad \dfrac{\varphi \quad \varphi \to \bot}{\bot}\ (\to_e)$$

Mathematical proofs - logic & deduction

Interchangeable rules:

$$\frac{\neg\neg\phi}{\phi}\,(\neg\neg_e) \qquad \frac{}{\phi \vee \neg\phi}\,(\text{LEM}) \qquad \frac{\begin{array}{c}[\neg\phi]^u\\ \vdots\\ \bot\end{array}}{\phi}\,(\text{PBC})\ u$$

Mathematical proofs - logic & deduction

Examples of deductions. Assuming $(\neg\neg_e)$, (LEM) holds:

$$
\begin{prooftree}
\AxiomC{$[\neg(\phi \vee \neg\phi)]^x$}
\AxiomC{$[\neg(\phi \vee \neg\phi)]^x$}
\AxiomC{$[\phi]^u$}
\RightLabel{$(\vee_i)$}
\UnaryInfC{$\phi \vee \neg\phi$}
\RightLabel{$(\neg_e)$}
\BinaryInfC{$\bot$}
\RightLabel{$(\neg_i)\ u$}
\UnaryInfC{$\neg\phi$}
\RightLabel{$(\vee_i)$}
\UnaryInfC{$\phi \vee \neg\phi$}
\RightLabel{$(\neg_e)$}
\BinaryInfC{$\bot$}
\RightLabel{$(\neg_i)\ x$}
\UnaryInfC{$\neg\neg(\phi \vee \neg\phi)$}
\RightLabel{$(\neg\neg_e)$}
\UnaryInfC{$\phi \vee \neg\phi$}
\end{prooftree}
$$

Notation: $\neg\neg\phi \vdash \phi \vee \neg\phi$

Mathematical proofs - logic & deduction

A derivation of Peirce's law, $((\phi \to \psi) \to \phi) \to \phi$:

$$
\begin{prooftree}
\AxiomC{$[\neg\phi]^u$}
\AxiomC{$[((\phi \to \psi) \to \phi)]^x$}
\AxiomC{$[\neg\phi]^u$}
\RightLabel{$(\to_i)\ \emptyset$}
\UnaryInfC{$\neg\psi \to \neg\phi$}
\AxiomC{$[\neg\psi]^v$}
\RightLabel{$(\to_e)$}
\BinaryInfC{$\neg\phi$}
\AxiomC{$[\phi]^w$}
\RightLabel{$(\neg_e)$}
\BinaryInfC{$\bot$}
\RightLabel{(PBC) v}
\UnaryInfC{ψ}
\RightLabel{$(\to_i)\ w$}
\UnaryInfC{$\phi \to \psi$}
\RightLabel{$(\to_e)$}
\BinaryInfC{ϕ}
\RightLabel{$(\neg_e)$}
\BinaryInfC{$\bot$}
\RightLabel{(PBC) u}
\UnaryInfC{ϕ}
\RightLabel{$(\to_i)\ x$}
\UnaryInfC{$((\phi \to \psi) \to \phi) \to \phi$}
\end{prooftree}
$$

Notation: $\vdash ((\phi \to \psi) \to \phi) \to \phi$

Mathematical proofs - logic & deduction

More examples. A derivation for $\neg\forall x\,\phi \vdash \exists x\,\neg\phi$

$$
\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{[\neg\phi\{x/x_0\}]^u}{\exists x\,\neg\phi}\,(\exists_i) \qquad [\neg\exists x\,\neg\phi]^v}{\bot}\,(\neg_e)}{\phi\{x/x_0\}}\,(\mathrm{PBC})\ u}{\forall x\,\phi}\,(\forall_i) \qquad \neg\forall x\,\phi}{\bot}\,(\neg_e)}{\exists x\,\neg\phi}\,(\mathrm{PBC})\ v
$$

A derivation for $\exists x\,\neg\phi \vdash \neg\forall x\,\phi$

$$
\dfrac{\exists x\,\neg\phi \qquad \dfrac{\dfrac{[\neg\phi\{x/x_0\}]^u \qquad \dfrac{[\forall x\,\phi]^v}{\phi\{x/x_0\}}\,(\forall_e)}{\bot}\,\neg e}{\neg\forall x\,\phi}\,(\neg_i)\ v}{\neg\forall x\,\phi}\,(\exists_e)\ u
$$

Mathematical proofs - logic & deduction

More examples. A derivation for $\neg\exists x\,\phi \vdash \forall x\,\neg\phi$

$$\dfrac{\dfrac{\dfrac{\dfrac{[\phi\{x/x_0\}]^u}{\exists x\,\phi}\,(\exists_i) \qquad \neg\exists x\,\phi}{\bot}\,(\neg_e)}{\neg\phi\{x/x_0\}}\,(\neg_i)\ u}{\forall x\,\neg\phi}\,(\forall_i)$$

A derivation for $\forall x\,\neg\phi \vdash \neg\exists x\,\phi$

$$\dfrac{\dfrac{[\exists x\,\phi]^u \qquad \dfrac{\dfrac{\forall x\,\neg\phi}{\neg\phi\{x/x_0\}}\,(\forall_e) \qquad [\phi\{x/x_0\}]^v}{\bot}\,(\neg_e)}{\bot}\,(\exists_e)\ v}{\neg\exists x\,\phi}\,(\neg_i)\ u$$

Gentzen Calculus

sequents:

$$\Gamma \quad \Rightarrow \quad \Delta$$

$$\underset{\text{antecedent}}{\uparrow} \qquad\qquad \underset{\text{succedent}}{\uparrow}$$

Gentzen Calculus

Table : Rules of deduction *à la* Gentzen for predicate logic

left rules	right rules
Axioms:	
$\Gamma, \varphi \Rightarrow \varphi, \Delta \quad (Ax)$	$\bot, \Gamma \Rightarrow \Delta \quad (L_\bot)$
Structural rules:	
$\dfrac{\Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta} \; (LWeakening)$	$\dfrac{\Gamma \Rightarrow \Delta}{\Gamma \Rightarrow \Delta, \varphi} \; (RWeakening)$
$\dfrac{\varphi, \varphi, \Gamma \Rightarrow \Delta}{\varphi, \Gamma \Rightarrow \Delta} \; (LContraction)$	$\dfrac{\Gamma \Rightarrow \Delta, \varphi, \varphi}{\Gamma \Rightarrow \Delta, \varphi} \; (RContraction)$

Gentzen Calculus

Table : Rules of deduction *à la* Gentzen for predicate logic

left rules	right rules
Logical rules:	
$\dfrac{\varphi_{i\in\{1,2\}}, \Gamma \Rightarrow \Delta}{\varphi_1 \wedge \varphi_2, \Gamma \Rightarrow \Delta}\ (L_\wedge)$	$\dfrac{\Gamma \Rightarrow \Delta, \varphi \quad \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \wedge \psi}\ (R_\wedge)$
$\dfrac{\varphi, \Gamma \Rightarrow \Delta \quad \psi, \Gamma \Rightarrow \Delta}{\varphi \vee \psi, \Gamma \Rightarrow \Delta}\ (L_\vee)$	$\dfrac{\Gamma \Rightarrow \Delta, \varphi_{i\in\{1,2\}}}{\Gamma \Rightarrow \Delta, \varphi_1 \vee \varphi_2}\ (R_\vee)$
$\dfrac{\Gamma \Rightarrow \Delta, \varphi \quad \psi, \Gamma \Rightarrow \Delta}{\varphi \rightarrow \psi, \Gamma \Rightarrow \Delta}\ (L_\rightarrow)$	$\dfrac{\varphi, \Gamma \Rightarrow \Delta, \psi}{\Gamma \Rightarrow \Delta, \varphi \rightarrow \psi}\ (R_\rightarrow)$
$\dfrac{\varphi[x/t], \Gamma \Rightarrow \Delta}{\forall_x \varphi, \Gamma \Rightarrow \Delta}\ (L_\forall)$	$\dfrac{\Gamma \Rightarrow \Delta, \varphi[x/y]}{\Gamma \Rightarrow \Delta, \forall_x \varphi}\ (R_\forall), \quad y \notin \mathtt{fv}(\Gamma, \Delta)$
$\dfrac{\varphi[x/y], \Gamma \Rightarrow \Delta}{\exists_x \varphi, \Gamma \Rightarrow \Delta}\ (L_\exists), \quad y \notin \mathtt{fv}(\Gamma, \Delta)$	$\dfrac{\Gamma \Rightarrow \Delta, \varphi[x/t]}{\Gamma \Rightarrow \Delta, \exists_x \varphi}\ (R_\exists)$

Gentzen Calculus

Derivation of the Peirce's law:

$$
\dfrac{\dfrac{(R_{\to})\ \dfrac{(RW)\ \dfrac{\varphi \Rightarrow \varphi\ \ (Ax)}{\varphi \Rightarrow \varphi, \psi}}{\Rightarrow \varphi, \varphi \to \psi} \qquad \varphi \Rightarrow \varphi\ \ (Ax)}{(\varphi \to \psi) \to \varphi \Rightarrow \varphi}\ (R_{\to})}{\Rightarrow ((\varphi \to \psi) \to \varphi) \to \varphi}\ (L_{\to})
$$

Gentzen Calculus

Cut rule:

$$\boxed{\dfrac{\Gamma \Rightarrow \Delta, \varphi \quad \varphi, \Gamma' \Rightarrow \Delta'}{\Gamma\Gamma' \Rightarrow \Delta\Delta'}\ (Cut)}$$

Gentzen Calculus

Example of application of (Cut): $\vdash \Rightarrow \neg\neg(\psi \vee \neg\psi)$.

$$
\dfrac{
\dfrac{
\dfrac{
\dfrac{
\dfrac{\psi \Rightarrow \psi, \bot \ (Ax)}{\Rightarrow \psi, \neg\psi}\,(R_{\rightarrow})
}{\Rightarrow \psi \vee \neg\psi, \neg\psi}\,(R_{\vee})
}{\Rightarrow \psi \vee \neg\psi, \psi \vee \neg\psi}\,(R_{\vee})
}{\Rightarrow \psi \vee \neg\psi}\,(RC)
\qquad
\dfrac{
\dfrac{(Ax)\ \psi \vee \neg\psi \Rightarrow \psi \vee \neg\psi \qquad \bot \Rightarrow \bot \ (L_{\bot})}{\psi \vee \neg\psi, \neg(\psi \vee \neg\psi) \Rightarrow \bot}\,(R_{\rightarrow})
}{\psi \vee \neg\psi \Rightarrow \neg\neg(\psi \vee \neg\psi)}\,(R_{\rightarrow})
}{\Rightarrow \neg\neg(\psi \vee \neg\psi)}\,(Cut)
$$

The Prototype Verification System - PVS

PVS is a verification system, developed by the SRI International Computer Science Laboratory, which consists of

1. a specification language:
    - based on higher-order logic;
    - a type system based on Church's simple theory of types augmented with subtypes and dependent types.
2. an interactive theorem prover:
    - based on Gentzen sequent calculus.

The Prototype Verification System - PVS — Libraries

- NASA LaRC PVS library includes
  - Structures, analysis, algebra, Graphs, Digraphs,
  - real arithmetic, floating point arithmetic, groups, interval arithmetic,
  - linear algebra, measure integration, metric spaces,
  - orders, probability, series, sets, topology,
  - term rewriting systems, unification, etc. etc.

Sequent calculus

- Sequents of the form: $\Gamma \vdash \Delta$.
  - Interpretation: from Γ one obtains Δ.
  - $A_1, A_2, ..., A_n \vdash B_1, B_2, ..., B_m$ interpreted as $A_1 \wedge A_2 \wedge ... \wedge A_n \vdash B_1 \vee B_2 \vee ... \vee B_m$.
- Inference rules
  - Premises and conclusions are simultaneously constructed.
  - Example: $\dfrac{\Gamma \vdash \Delta}{\Gamma' \vdash \Delta'}$
- Goal: $\vdash \Delta$.

Sequent calculus in PVS

- Representation of $A_1, A_2, ..., A_n \vdash B_1, B_2, ..., B_m$:

```
 [-1]  A1
   :
 [-n]  An
|----------
 [1]  B1
   :
 [n]  Bn
```

- Proof tree: each node is labelled by a sequent.
- A PVS proof command (R) corresponds to the reverse application of an inference rule.
    - In general:

$$\frac{\Gamma \vdash \Delta}{\Gamma_1 \vdash \Delta_1 ... \Gamma_n \vdash \Delta_n}\ (R)$$

Some inference rules in PVS

- Structural:

$$\boxed{\frac{\Gamma, \Gamma' \vdash \Delta, \Delta'}{\Gamma \vdash \Delta}\ \textbf{(hide)}\ \ (W)} \quad \boxed{\frac{\Gamma, \Gamma' \vdash \Delta, \Delta'}{\Gamma, \Gamma', \Gamma' \vdash \Delta, \Delta', \Delta'}\ \textbf{(copy)}\ \ (C)}$$

- Axioms:

$$\boxed{\Gamma, A \vdash A, \Delta\ \ \textbf{(Ax)}\ \ (Ax)}$$

$$\boxed{\Gamma, FALSE \vdash \Delta\ \ \textbf{(False}\vdash\textbf{)}\ \ (L_\perp)} \quad \boxed{\Gamma \vdash TRUE, \Delta;\ \textbf{(}\vdash\textbf{True)}\ \ (L_\perp)}$$

Some inference rules in PVS

- Logical (propositional) rules:

$$\frac{\Gamma \vdash \Delta, \psi \vee \varphi}{\Gamma, \vdash \Delta, \psi, \varphi} \ \textbf{(flatten)} \quad (R_{\vee})$$

$$\frac{\psi \wedge \varphi, \Gamma, \vdash \Delta}{\psi, \varphi, \Gamma \vdash \Delta} \ \textbf{(flatten)} \quad (L_{\wedge})$$

$$\frac{\Gamma \vdash \Delta, \psi \rightarrow \varphi}{\psi, \Gamma \vdash \Delta, \varphi} \ \textbf{(flatten)} \quad (R_{\rightarrow})$$

Some inference rules in PVS

- Logical (propositional) rules:

$$\frac{\psi \vee \varphi, \Gamma \vdash \Delta}{\psi, \Gamma \vdash \Delta \quad \varphi, \Gamma \vdash \Delta} \; \textbf{(split)} \quad (L_{\vee})$$

$$\frac{\Gamma \vdash \Delta, \psi \wedge \varphi}{\Gamma \vdash \Delta, \psi \quad \Gamma \vdash \Delta, \varphi} \; \textbf{(split)} \quad (L_{\wedge})$$

$$\frac{\psi \rightarrow \varphi, \Gamma \vdash \Delta}{\varphi, \Gamma \vdash \Delta \quad \Gamma \vdash \Delta, \psi} \; \textbf{(split)} \quad (L_{\rightarrow})$$

Some inference rules in PVS

- Logical (classical) rules:

$$\frac{\forall_x \psi, \Gamma \vdash \Delta}{\psi[x/t], \Gamma \vdash \Delta} \textbf{ (inst) } (L_\forall) \qquad \frac{\Gamma \vdash \Delta, \forall_x \psi}{\Gamma \vdash \Delta, \psi[x/x_0]} \textbf{ (skolem) } (R_\forall)$$

$$\frac{\Gamma \vdash \Delta, \exists_x \psi}{\Gamma \vdash \Delta, \psi[x/t]} \textbf{ (inst) } (R_\exists) \qquad \frac{\exists_x \psi, \Gamma \vdash \Delta}{\psi[x/x_0], \Gamma \vdash \Delta} \textbf{ (skolem) } (L_\exists)$$

PVS vs Gentzen Rules

	(flatten)	(split)	(inst)	(skolem)
$(L_\vee)$		$\times$		
$(R_\vee)$	$\times$			
$(L_\wedge)$	$\times$			
$(R_\wedge)$		$\times$		
$(L_\to)$		$\times$		
$(R_\to)$	$\times$			
$(L_\forall)$			$\times$	
$(R_\forall)$				$\times$
$(L_\exists)$				$\times$
$(R_\exists)$			$\times$	

PVS propositional derivation example

Derivation of the Peirce's law:

$$
\cfrac{\cfrac{\cfrac{\varphi \Rightarrow \varphi, \psi \;\; (Ax)}{\Rightarrow \varphi, \varphi \to \psi}\,(R_{\to}) \qquad \varphi \Rightarrow \varphi \;\; (Ax)}{(\varphi \to \psi) \to \varphi \Rightarrow \varphi}\,(R_{\to})}{\Rightarrow ((\varphi \to \psi) \to \varphi) \to \varphi}\,(L_{\to})
$$

$$
\begin{array}{c}
\vdash ((\varphi \to \psi) \to \varphi) \to \varphi \\
\hline
(\varphi \to \psi) \to \varphi \vdash \varphi \\
\hline
\cfrac{\vdash \varphi, \varphi \to \psi}{\varphi \vdash \varphi, \psi \;\; (Ax)}\,(flatten) \qquad \varphi \vdash \varphi \;\; (Ax)
\end{array}
\begin{array}{l}
\\ (flatten) \\ (split) \\ \\
\end{array}
$$

Additional rules in PVS

- Case:
  - Corresponds to the rule (Cut).

$$\boxed{\frac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta \qquad \Gamma \vdash A, \Delta}\ \textbf{(Case"}A\textbf{")}}$$

- Conditional: `IF-THEN-ELSE`.

$$\boxed{\frac{\Gamma, A \wedge B \vdash \Delta \qquad \Gamma, \neg A \wedge C \vdash \Delta}{\Gamma, \mathtt{IF}(A, B, C) \vdash \Delta}\ (\mathtt{split})}$$

$$\boxed{\frac{\Gamma, A \to B \vdash \Delta \qquad \Gamma, \neg A \to C \vdash \Delta}{\Gamma \vdash \mathtt{IF}(A, B, C)\Delta}\ (\mathtt{split})}$$

Case Study: insertion sort

$\texttt{insert}(x:\mathbb{N}, l:\mathit{list}[\mathbb{N}])$: RECURSIVE $\mathit{list}[\mathbb{N}]$ =
if $\mathit{null?}(l)$ **then**
  $\mathit{cons}(x, \mathit{null})$
else
  if $x \leq \mathit{car}(l)$ **then**
    $\mathit{cons}(x, l)$
  else
    $\mathit{cons}(\mathit{car}(l), \texttt{insert}(x, \mathit{cdr}(l)))$
  end
end
MEASURE $\mathit{length}(l)$

Case Study: insertion sort

in_sort(l : $list[\mathbb{N}]$) : RECURSIVE $list[\mathbb{N}]$ =
if $null?(l)$ **then**
| $null$
else
| insert($car(l)$, in_sort($cdr(l)$))
end
MEASURE $length(l)$

Insertion sort — correctness

$$\texttt{insert\_preserves\_order} : \mathsf{LEMMA}\ \forall (l : list[nat], x : nat) : \\ \texttt{is\_sorted?}(l) \to \texttt{is\_sorted?}(\texttt{insert}(x, l))$$

$$\texttt{insertion\_sort\_works} : \mathsf{LEMMA} \\ \forall (l : list[nat]) : \texttt{is\_sorted?}(\texttt{in\_sort}(l)) \quad \wedge \\ \texttt{permutations}(l, \texttt{in\_sort}(l))$$

Insertion sort — correctness formalization

The proof is by induction on $|l|$. Induction hypothesis (IH):
$\forall (l', x') : |l'| < |l| \rightarrow (\texttt{is\_sorted?}(l') \rightarrow \texttt{is\_sorted?}(\texttt{insert}(x', l'))$

Sequent:

$$\begin{array}{l} \forall (l', x') : |l'| < |l| \rightarrow (\texttt{is\_sorted?}(l') \rightarrow \texttt{is\_sorted?}(\texttt{insert}(x', l')) \\ \Rightarrow \\ \texttt{is\_sorted?}(l) \rightarrow \texttt{is\_sorted?}(\texttt{insert}(x, l)) \end{array}$$

... two interesting sequents should be proved:

$$\begin{array}{l} \mathit{null?}(l), \texttt{is\_sorted?}(l), \mathit{IH} \\ \Rightarrow \\ \texttt{is\_sorted?}(\texttt{insert}(x, l)) \end{array} \quad \text{and} \quad \begin{array}{l} \texttt{is\_sorted?}(l), \mathit{IH} \\ \Rightarrow \\ \mathit{null?}(l), \texttt{is\_sorted?}(\texttt{insert}(x, l)) \end{array}$$

Insertion sort — correctness formalization

The former sequent is easily proved. For the latter sequent, `insert` is expanded obtaining:

$$\begin{array}{l}\texttt{is\_sorted?}(l), IH \\ \Rightarrow \\ null?(l), \\ \texttt{is\_sorted?}(\ \textbf{if}\ x \leq car(l)\ \textbf{then}\ cons(x,l)\ \textbf{else} \\ cons(car(l), insert(x, cdr(l))))\end{array}$$

Applying logical commands such as `(lift-if)` and `(prop)`, that guided an application of the (Cut) rule by the guard $x \leq car(l)$ of the **if-then-else** command this gives two sequents:

$$\begin{array}{ll} & x \leq car(l), \texttt{is\_sorted?}(l), IH \Rightarrow null?(l), \\ & \texttt{is\_sorted?}(cons(x,l)) \\ \text{and} & \texttt{is\_sorted?}(l), IH \Rightarrow null?(l), x \leq car(l), \\ & \texttt{is\_sorted?}(cons(car(l), insert(x, cdr(l))))\end{array}$$

Insertion sort — correctness formalization

The former sequent is easily proved. For the latter one, the *IH* is used by applying the PVS instantiation command (`inst`) which corresponds to $(L_\exists)$, obtaining the sequent:

$$
\begin{array}{l}
\texttt{is\_sorted?}(l),\\
|cdr(l)| < |l| \to (\texttt{is\_sorted?}(cdr(l)) \to\\
\texttt{is\_sorted?}(\texttt{insert}(x, cdr(l)))\\
\Rightarrow\\
null?(l),\ x \leq car(l)\\
\texttt{is\_sorted?}(cons(car(l), \texttt{insert}(x, cdr(l))))
\end{array}
$$

Insertion sort — correctness formalization

By applications of the command (`prop`), guided applications of $(L_\rightarrow)$ by the premises of the implications in the antecedent, that is, $|cdr(l)| < |l|$ and $\mathtt{is\_sorted?}(cdr(l))$, are done, obtaining the interesting sequent below that follows easily.

$$
\begin{array}{l}
\mathtt{is\_sorted?}(l),\\
|cdr(l)| < |l|,\\
\mathtt{is\_sorted?}(cdr(l)),\\
\mathtt{is\_sorted?}(\mathtt{insert}(x, cdr(l)))\\
\Rightarrow\\
null?(l),\ x \leq car(l)\\
\mathtt{is\_sorted?}(cons(car(l), \mathtt{insert}(x, cdr(l))))
\end{array}
$$

Conclusions

- Nowadays, computational logic is intensively applied in formal methods.
- In computer sciences, a useful training on "computational" logic should focus on derivation/proof techniques.
- Understanding proof theory is essential to mastering proof assistants:
  - to provide mathematical proofs of robustness of computational systems and
  - well-accepted quality certificates.

Work in Progress

- Textbook on truly *computational logic* with concrete applications.
  - M.Ayala-Rincón & F.L.C.de Moura *Applied Logic for Computer Scientists: computational deduction and formal proofs*, 2014, UTiCS series, Springer.

GTC at the Universidade de Brasília.