Automata Theory Approach to Predicate Intuitionistic Logic

Maciej Zielenkiewicz and Aleksy Schubert

Institute of Informatics, University of Warsaw, Warsaw, Poland

[maciekz,alx]@mimuw.edu.pl

1. Arcadian Automata

$\mathbb{A}$ is defined as a tuple $\langle \mathcal{A}, Q, q^0, \varphi^0, \mathcal{I}, i, \mathrm{fv} \rangle$, where

- $\mathcal{A} = \langle A, \leq \rangle$ is a finite tree;
- Q is a set of states;
- $q^0 \in Q$ is an *initial* state of the automaton;
- $\varphi^0 \in A$ is an *initial* tree node of the automaton;
- $\mathcal{I}$ is a set of all instructions;
- $i \colon Q \to \mathcal{P}(\mathcal{I})$ is a function which gives the set of instructions *available* in a given state;
- $\mathrm{fv} : A \to P(A)$ is a *binding* function.

Each state may be either existential or universal and belongs to an element $a \in A$, so $Q = Q^\exists \cup Q^\forall$, and $Q^\forall = \bigcup_{a \in A} Q^\forall_a$ and $Q^\exists = \bigcup_{a \in A} Q^\exists_a$.

2. Instantaneous description

An ID of $\mathbb{A}$ is a tuple $\langle q, \kappa, w, w', S, V \rangle$ where

- $q \in Q$ is the current state,
- κ is the current node in A,
- $w : A \rightharpoonup V$ is the interpretation of bindings associated with κ by $\mathrm{fv}(\kappa)$, in particular we require here that $\mathrm{fv}(\kappa) \subseteq \mathrm{dom}(w)$,
- $w' : A \rightharpoonup V$ is the auxiliary interpretation of bindings
- S is the *store* of the automaton, which is a set of pairs $\langle \rho, v \rangle$ where $\rho \in A$ and $v : A \rightharpoonup V$ and we require that $\mathrm{fv}(\rho) \subseteq \mathrm{dom}(v)$,
- V is the working domain of the automaton, i.e. a set of eigenvariables,

3. Instructions

1. q : store $\rho, \rho' q'$ adds a a new fact $\langle \rho, (w \ll w')|_{\mathrm{fv}(\rho)} \rangle$ to the store;
2. q : jmp ρ, q'
3. q : new ρ, q' extends the discourse domain V with a new element X;
4. q : check ρ, ρ', q' checks if approprate fact is stored in S
5. q : instL ρ, ρ', q' checks if the current node is an appropriate successor of ρ;
6. q : load ρ, q' loads an appropriately constructed interpretation to the additional register.

4. Structural decomposition instructions

(1) $\varphi_1 \to \varphi_2$ $\quad q^\forall_{\varphi_1 \to \varphi_2}$: store $\varphi_1, \varphi_2, q^\exists_{\varphi_2}$
$\Rightarrow \langle q^\forall_{\varphi_1 \to \varphi_2}, \varphi_1 \to \varphi_2, w, \emptyset, S, V \rangle \to \langle q^\exists_{\varphi_2}, \varphi_2, w, \emptyset, S \cup \{\langle \varphi_1, w|_{\mathrm{fv}(\varphi_1)} \rangle\}, V \rangle$

(2) $\varphi_1 \wedge \varphi_2$ $\quad q^\forall_{\varphi_1 \wedge \varphi_2}$: jmp $\varphi_1, q^\exists_{\varphi_1}$
$\Rightarrow \langle q^\forall_{\varphi_1 \wedge \varphi_2}, \varphi_1 \wedge \varphi_2, w, \emptyset, S, V \rangle \to \langle q^\exists_{\varphi_1}, \varphi_1, w, \emptyset, S, V \rangle$
$q^\forall_{\varphi_1 \wedge \varphi_2}$: jmp $\varphi_2, q^\exists_{\varphi_2}$
$\Rightarrow \langle q^\forall_{\varphi_1 \wedge \varphi_2}, \varphi_1 \wedge \varphi_2, w, \emptyset, S, V \rangle \to \langle q^\exists_{\varphi_2}, \varphi_2, w, \emptyset, S, V \rangle$

(3) $\varphi_1 \vee \varphi_2$ $\quad q^\exists_{\varphi_1 \vee \varphi_2}$: jmp $\varphi_1, q^\exists_{\varphi_1}$
$\Rightarrow \langle q^\exists_{\varphi_1 \vee \varphi_2}, \varphi_1 \vee \varphi_2, w, \emptyset, S, V \rangle \to \langle q^\exists_{\varphi_1}, \varphi_1, w, \emptyset, S, V \rangle$
$q^\exists_{\varphi_1 \vee \varphi_2}$: jmp $\varphi_2, q^\exists_{\varphi_2}$
$\Rightarrow \langle q^\forall_{\varphi_1 \vee \varphi_2}, \varphi_1 \vee \varphi_2, w, \emptyset, S, V \rangle \to \langle q^\exists_{\varphi_2}, \varphi_2, w, \emptyset, S, V \rangle$

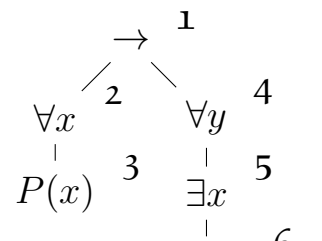
(4) $\forall X.\varphi$ $\quad q^\forall_{\forall X.\varphi}$: new $\varphi, q^\exists_\varphi$
$\Rightarrow \langle q^\forall_{\forall X.\varphi}, \forall X.\varphi, w, \emptyset, S, V \rangle \to \langle q^\exists_\varphi, \varphi, [\forall X.\varphi := Y] \ll w, \emptyset, S, V \cup \{Y\} \rangle$
where $Y \notin V$

(5) $\exists X.\varphi$ $\quad q^\forall_{\exists X.\varphi}$: instR $\varphi, q^\exists_\varphi$
$\Rightarrow \langle q^\forall_{\exists X.\varphi}, \exists X.\varphi, w, \emptyset, S, V \rangle \to \langle q^\exists_\varphi, \varphi, [\exists X.\varphi := Y] \ll w|_{\mathrm{fv}(\exists X.\varphi)}, \emptyset, S, V \rangle$
where $Y \in V$

5. Example

We built the Arcadian automaton for $\varphi_{\mathrm{pos}} = \forall x(P(x)) \to \forall y \exists x P(x)$ (tree shown in Fig. 1).

Figure 1: Syntax tree of ϕ_{pos}.



The set $\mathcal{I}$ of available instructions consists of elements:

(10) $q^\exists_6$: jmp $2, q^\exists_2$ $\quad$ (4) $q^\forall_4$: new $5, q^\exists_5$
(4) $q^\forall_2$: new $3, q^\exists_3$ $\quad$ (19) $q^\forall_{4,\exists}$: jmp $5, q^\exists_5$
(19) $q^\forall_{1,\exists}$: jmp $5, q^\exists_5$ $\quad$ (19) $q^\forall_{5,\exists}$: jmp $5, q^\exists_5$
(5) $q^\forall_5$: instR $6, q^\exists_6$ $\quad$ (20) $q^\forall_{1,\exists}$: instL $5, 1, q^\exists_1$
(10) $q^\exists_3$: jmp $2, q^\exists_2$ $\quad$ (20) $q^\forall_{4,\exists}$: instL $5, 4, q^\exists_4$
(1) $q^\forall_1$: store $2, 4, q^\exists_4$ $\quad$ (20) $q^\forall_{5,\exists}$: instL $5, 5, q^\exists_5$

We use some instructions available for any state. Here we spell out only the ones used in our successful run:

(6) $q^\exists_1$: jmp $1, q^\forall_1$ $\quad$ (6) $q^\exists_4$: jmp $4, q^\forall_4$
(6) $q^\exists_5$: jmp $5, q^\forall_5$ $\quad$ (13) $q^\exists_2$: check $2, 2, q^\forall_{\mathrm{axiom}}$

The red arrows show a successful run of the automaton.