N. N. Nepejvoda

A CONSTRUCTIVE LOGIC OF PROGRAM SCHEMATA
ON A DECIDABLE UNIVERSE

Algorithmic and dynamic logics are powerful tools for programming analysis. Their investigation is one of the most actual trends of the modern mathematical logic. But each such logic is in some sense like a centaurus: its language consists of two parts (algorithmic and logic ones) and these two parts are of a very different nature.

A logic of a completely new class is introduced and studied here: a constructive logic of program schemata $\Omega_1$. There is not in its language any explicite mention of programs, nevertheless its semantics is defined through realizability by program schemata. It is known (see e.g. [1]) that constructive logics also are powerful tools for the mentioned purposes. Furthermore, proof search strategies for constructive logics give us program synthesis algorithms and programming methodics.

The main peculiarities of our system by which many definitions are motivated are as follows; ruling out a possibility of infinite executions ab finitio and indeterminism.

**Definition 1.** Let a propositional vocabulary $P$ be a decidable set of propositional symbols. Let a functional vocabulary $F$ be a decidable set of functional symbols and evaluation of a set of states $S$ a function $z : S \rightarrow 2^P$. $p \in P$ is said to be true on $s \in S$ iff $p \in z(s)$. The relation “a formula $A$ of the classical propositional calculus ($CPC$) on $P$ is true in $s$” is defined in a usual way and denoted by $s \models A$.

The connectives of the $CPC$ are denoted by $\& \vee \Rightarrow \neg$.

**Definition 2.** Let a resource universe $W$ be a non-empty subset of $S \times On$ where $On$ in the set of countable ordinals. If $w = (s, \alpha) \in W$ then $w$ is
called to be a situation, let $s$ be the state of $w$, and $\alpha$ be the resource of $w$. A pair $(W, z)$ where $z$ is an evaluation of $S$ is called to be a resource interpretation of $P$. $w \models A$ iff $\text{state}(w) \models z A$.

$R$ is called to be a noninvertible relation on $W$ iff $R \subseteq W \times W$ and $\text{resource}(w') < \text{resource}(w)$ for each $(w, w') \in R$. $R$ is defined on $w$ iff there exists $w'$ such that $(w, w') \in R$. Let $\text{NIR}(W)$ be the set of all non-invertible relations on $W$. A pair $(\text{Res}, \text{ff})$ is called to be a noninvertable interpretation of $P, F$ iff $\text{Res}$ is a resource interpretation of $P$ and $\text{ff}$ is a function from $F$ to $\text{NIR}(W)$.

**Definition 3.** An orgraph $\Sigma$ is called to be a functional scheme on $P, F$ if the following holds:

(i) there are the initial and the final vertix of $\Sigma$. The remaining vertices are divided into three disjoint classes: functions, branchings and joining.

(ii) all functional vertices are accompanied by $f \in F$,

(iii) there is one arc from and no arcs in the initial vertix. There is one arc in and no arcs from the final vertix. There are the arc in and the arc from a functional vertix. There are some arcs from and one arc to a branching. There are one arc from and some arcs to a joining,

(iv) all arcs from branchings are accompanied by formulas of $\text{CPC}$.

An execution of $\Sigma$ starting with $w$ in the interpretation $I$ is a pair of sequences of the same length $(v_0, \ldots, v_n; w_0, \ldots, w_n)$ where:

(i) $v_0, \ldots, v_n$ is a way in $\Sigma$ starting with the initial vertix;

(ii) $w_0, \ldots, w_n$ is a sequence of situations and if $v_i$ is not a function, $w_{i+1} = w_i$; $w_0 = w$;

(iii) if $v_i$ is a function accompanied by $f_i$ then $(w_i, w_{i+1}) \in z(f_i)$;

(iv) if $v_i$ is a branching and the arc from $v_i$ to $v_{i+1}$ is accompanied by $A$ then $w_i \models A$.

An execution is called to be normal if $v_n$ is the final vertix, abnormal if $v_n$ is a function accompanied by $f_n$ and $z(f_n)$ is not defined on $w_n$ or $v_n$ is branching and there is no such $A$ accompanied to an arc from $v_n$ that $w_n \models A$. 
Σ is called to be nondegenerate if in each way from the initial vertex to the final one and in each cycle there occurs a function.

Let us define

\[ R_\Sigma = \{(w, w') \mid \text{there are no abnormal executions of } \Sigma \text{ starting with } w \text{ and there is a normal execution starting with } w \text{ such that } w_n = w' \} \].

**Proposition 1.** If Σ is nondegenerate then \( R_\Sigma \) is non-invertible.

**Proposition 2.** If Σ is nondegenerate then each execution of Σ can be extended up to normal or abnormal and, hence, there are no infinite executions.

In the sequel of the paper only nondegenerate schemata are considered.

Let \( \text{error} \in F \) and \( z(\text{error}) = \emptyset \).

Now let us define our constructive language and realizability.

**Definition 4.** \( A \Rightarrow B \) where \( A \) and \( B \) are formulas of CPC on \( P \) is called to be a constructive formula on \( P \). \( R \) realizes \( A \Rightarrow B \) in \( I \) if for each \( w \in W \) such that \( w \models A \), \( R \) is defined on \( W \) and for each \( w' \) such that \( (w, w') \in R \), \( w' \models B \). Σ realizes \( A \Rightarrow B \) if \( R_\Sigma \) does. \( A \Rightarrow B \) is realizable in \( I \) if there exists a scheme Σ realizing \( A \Rightarrow B \).

A set of formulas of CPC on \( P \) and constructive formulas on \( P \) (classical and constructive axioms, resp.) is called to be a theory. A realization of a theory \( Th \) is a pair \((I, r)\) where \( I \) is a noninvertible interpretation of \( P, F \) and \( r(A \Rightarrow B) \in F \) for each constructive axiom of \( Th \) and \( z(r(A \Rightarrow B)) \) realizes \( A \Rightarrow B \) and for each classical axiom \( A \) of \( Th \) and for each \( w \in W \), \( w \models A \). \( Th \) is called to be consistent iff there exists a realization of \( Th \). \( A \) is a semantical consequence of \( Th \) iff \( A \) is realizable (true in each situation) in each realization of \( Th \).

Now let us construct a full normal system \( \Omega_1 \) for our semantic consequence relation.

**Axioms.** All tautologies of CPC and \( \text{false} \Rightarrow \text{false} \) where \( \text{false} \) is a contradiction of CPC.
Rules:

A1. Modus Ponens
\[ \frac{A \land B}{B} \]

A2. Relaxation rule
\[ \frac{A \Rightarrow B}{L \Rightarrow V} \]

A3. Conditional statement
\[ \frac{A_1 \Rightarrow B_1 \ldots A_n \Rightarrow B_n}{A_1 \lor \ldots \lor A_n \Rightarrow B_1 \lor \ldots \lor B_n} \]

A4. Loop
\[ \frac{A \lor L \Rightarrow B \lor L}{A \Rightarrow B} \]

A5. Infinite Loop
\[ \frac{A \Rightarrow A}{\neg \bot} \]

The derivability of \( X \) from axioms of \( Th \) and \( \Omega_1 \) by rules of \( \Omega_1 \) is denoted by \( Th \vdash_{\Omega_1} X \).

**Proposition 3.** If \( Th \vdash_{\Omega_1} A \Rightarrow B \) and \( Th \vdash_{\Omega_1} B \Rightarrow C \) then \( Th \vdash_{\Omega_1} A \Rightarrow C \).

**Proposition 4.** If \( Th \vdash_{\Omega_1} A \Rightarrow A \lor B \) then \( Th \vdash_{\Omega_1} A \Rightarrow B \).

**Proposition 5.** \( Th \) is inconsistent iff \( Th \vdash_{\Omega_1} T \Rightarrow \bot \) where \( T \) is a tautology of \( CPC \).

**Proposition 6.** If \( Th \vdash_{\Omega_1} A \Rightarrow B \) then \( Th \vdash_{\Omega_1} A \Rightarrow B \) in \( \Omega_1 \) without the infinite loop rule.

The analogue of Proposition 6 for classical formulas does not hold.

**Proposition 7.** If \( Th \vdash_{\Omega_1} A \Rightarrow B \) then there is a derivation of \( A \Rightarrow B \) of the following form:

\[ \frac{A \lor C \Rightarrow A_1 \lor \ldots \lor A_n \Rightarrow B_n}{A \lor C \Rightarrow B \lor C} \]

where \( A_1 \Rightarrow B_1, \ldots, A_n \Rightarrow B_n \) are axioms of \( Th \) or false \( \Rightarrow \) false.

**The Completeness Theorem.** \( Th \vdash_{\Omega_1} X \) iff \( X \) is a semantic consequence from \( Th \).
However, $\Omega_1$ in this formulation is not adequate for all constructive purposes because there is no direct correspondence between proofs and extracted program schemata. Here, as usual, a natural deduction calculus is more adequate. But there are some fundamental obstacles in formulation of this calculus overcome by the notion of a generalized calculus from [2].

A generalized calculus is given by a finite collection of deciding algorithms: an algorithm of well-formedness testing for each class of syntactical objects, a local algorithm of application correctness testing for each rule and a global algorithm $Gl$ of the whole structure of proof correctness testing. Proofs are represented as graphs with typed vertices and arcs. Vertices are divided into three classes accordingly to their types: informational vertices (e.g. formulas) which can be accompanied by a well-formed construction of some syntactic class, structural vertices (e.g. subproofs) and rule applications. Usually algorithms of syntactic correctness testing are standard, ones of applications testing are described by images of corresponding neighbourhood of an application, $Gl$ cannot use the information which informational vertices are accompanied by and is described by a collection of demands on the structure of proof graph.

Now let us define a generalized calculus of natural deduction for our $\Omega_1$.

Informational vertices are divided into three types: classical formulas denoted by $+A$, situations denoted by $\bullet A$, plans denoted by $\bullet A \Rightarrow B$. There is a structural vertex called by “main proof”, all other structural vertices are subordinated proofs and are denoted by $\Box$. Arcs from structural vertices are of the type “subordinating arcs”. They subordinate all subordinated proofs and all classical formulas and plans to the main proof and each situation to a subordinated proof. Some situations can be marked as assumptions or results. An arc leading to an assumption is denoted by $\bowtie$, to a result done by $\blacktriangledown$. All other arcs from structural vertices are omitted on our figures. A rule application is denoted by $\emptyset \circ R$ where $R$ is the abbreviation for a rule. The other global conditions are the following.

Each subordinated proof contains one assumption and one result. Each situation, classical formula or plan is generated by a single rule application (excluding assumptions which are not generated by a rule) and used by a single rule application (excluding one formula in the main proof called by the proved theorem and results of subordinated proofs; these formulas are not used by rules). All situations which are premisses of the single rule application are subordinated to the same subordinated proof. All classical
formulas, plans and subordinated proofs can be arranged into a sequence where premisses are before consequences and formulas used inside a subordinated proof are before this proof. There is a $FA$ rule application on each way from the assumption to the result of a subordinated proof and inside of each cycle within a subordinated proof.

Proof applications may be of the following forms.
1. Tautology

$(A \text{ being a tautology or an axiom of } Th)$

\[ \text{Taut} \]
\[ +A \]

2. Generating the axiom

$(A \Rightarrow B \text{ is an axiom of } Th \text{ or } F \Rightarrow F)$

\[ \text{GAx} \]
\[ A \Rightarrow B \]

3. Modus Ponens

\[ \text{MP} \]
\[ +A \ \ \ \ +A \supset B \]
\[ +B \]

4. Function application

\[ \text{FA} \]
\[ A \Rightarrow B \]
\[ +B \]

5. Joining

\[ \text{J} \]
\[ A \ldots A_n \supset A_1 \lor \ldots \lor A_n \supset B \]
\[ +B \]

6. Branching

\[ \text{Br} \]
\[ A \supset B_1 \lor \ldots \lor B_n \]
\[ *B_1 \ldots *B_n \]
7. Function description

\[
\begin{align*}
\text{\textcircled{as}} & \Rightarrow \text{\textbullet} A \\
\downarrow & \Rightarrow \text{\textbullet} B \\
\circ FD & \\
\downarrow & \text{\textbullet} A \Rightarrow B \\
\text{\textbullet} A \Rightarrow A & \\
\downarrow & \circ IL \\
\downarrow & \oplus \neg A
\end{align*}
\]

8. Infinite loop

The completeness theorem holds for the natural deduction system as well.

References
