AUTOMATIC SYNTHESIS OF OPTIMAL INVARIANT ASSERTIONS: 
MATHEMATICAL FOUNDATIONS

Patrick Cousot* and Radhia Cousot**

Laboratoire d'Informatique, U.S.M.G., BP.63
38041 Grenoble cedex, France

1. INTRODUCTION

The problem of discovering invariant assertions of programs is explored in light of the fixpoint approach in the static analysis of programs, Cousot [1977a], Cousot[1977b]. Therefore symbolic execution approximations method. Therefore symbolic execution permits optimal invariant assertions to be discov- red provided that one can pass to the limit, that is consider infinite paths in the symbolic execution tree. Induction principles can be used for that pur- pose. In section 7 we show how difference equations
We denote by \( L^N \) the set of all vectors 
\[ X = (X_1, \ldots, X_n) \] 
the components of which belong to \( L \).

\( (L^N, \leq, \top, \bot, \cup, \cap) \) is a complete lattice with the usual "componentwise" definitions: 
\[ \{X \leq Y\} \iff \{\forall i \in \{1, \ldots, n\}, X_i \leq Y_i\}, \{X \leq \top\} = \{(X_1, \ldots, X_n)\}, \{X \leq \bot\} = \{(\bot_1, \ldots, \bot_n)\}, \{X \leq Y\} = \{(X_1 \cup Y_1, \ldots, X_n \cup Y_n)\}, \text{etc.} \]

A function of several variables \( f : L^N \to L \) is said to be isotone (continuous) in the variables jointly if and only if it is isotone (continuous) in the variables separately.

Hereafter, we will consider a system of continuous equations with \( n \) variables of the form:

\[
\begin{align*}
X_1 &= f_1(X_1, \ldots, X_n) \\
\vdots \\
X_n &= f_n(X_1, \ldots, X_n)
\end{align*}
\]

This system can be abbreviated by a *fixpoint equation* \( X = F(X) \) where \( X \) is the vector \((X_1, \ldots, X_n)\) and \( F \) a continuous function of type \( L^N \to L^N \).

An element \( X \) of \( L^N \) is a *fixpoint* of \( F : L^N \to L^N \) if \( X = F(X) \).

Let \( J \) be a non-empty subset of \( \{1, \ldots, n\} \). We denote by \( F_J \) the map \( L^N \to L^N \) defined by \( F_J(X_1, \ldots, X_n) = (Y_1, \ldots, Y_n) \) where \( \forall i \in \{1, n\} \) we have \( Y_i = \top \) if \( i \in J \) then \( F_i(X_1, \ldots, X_n) \) else \( X_i \).

**DEFINITION 2.1** A *chaotic iteration* corresponding to the operator \( F \) and starting with a given vector \( X^0 \) such that \( X^k \equiv F(X^k) \equiv \text{Ifp}(F) \) is a sequence \( X^k, k=0,1, \ldots \) of vectors of \( L^N \) defined recursively by 
\[ X^k = F_{J_k}(X^{k-1}) \] 
where \( J_k, k=0,1, \ldots \) is a sequence of subsets of \( \{1, \ldots, n\} \) such that \( \forall i \in \{1, n\}, \forall k \geq 0, \exists m \in \{1, \ldots, n\} : i \in \{1, \ldots, n\} \wedge i \in \{1, \ldots, n\} \).

**THEOREM 2.2** The limit \( X^\infty \) of a chaotic iteration \( X^0, X^1, X^{k+1}, \ldots \) is equal to \( \text{Ifp}(F) \).

**Lemma 2.2.1** \( \forall k \geq 0, X^k \equiv X^{k+1} \equiv F(X^k) \equiv \text{Ifp}(F) \).

**Proof:** Let us first remark that whenever \( X \equiv F(X) \bulls \}
chaotic iterations \[\exists m \geq 0 : (V_1, V_k, \exists l \in [0, m] : \ldots, k+1, \ldots, k+1, \ldots, k+1, \ldots, k+1, \ldots)\]

Proof of theorem 2.3: The proof that \( (\forall k : (k \in \mathbb{Z}^+) \land (k \neq 0)) \) and \( (\exists k : (k \in \mathbb{Z}^+) \land (k \neq 0)) \) so by contradiction absurd.
Program entry point $j$:

$$P_j(X, \bar{x}^j) = \{(x_i, x_i^j), i=1..m\}$$

The respective initial values of the variables $X=(X_1, ..., X_m)$ are the symbols $\bar{x}^j=(x_1^j, ..., x_m^j)$. We may eventually have $(X_i=\infty)$ when the variable $X_i$ is not

3.4. OPTIMAL INVARIANTS

According to Tarski's theorem the above system of equations of the form $P=F(P)$ has a least solution $P^{opt}$ (least for ordering $\leq$ that is $\Rightarrow$). We call $P^{opt}$ the set of optimal invariants since they imply
satisfying $P \iff F(P)$. This is not surprising since this condition is based on Tarski’s theorem which does not provide an algorithmic construction of the least fixpoint of $F$.

6. SYMBOLIC EXECUTION

The purpose of this section is to show that symbolic execution of a program consists in solving the semantic equations associated with this program by chaotic iterations.

6.1. SYMBOLIC CONTEXTS

Observe (3.4) that the invariant $P_i$ associated with a program point $i$ can be expressed in the normal form $P_i = \text{OR} p_j$ where each $p_j$ is of the form $(Q_j \land (X_1 = E_{ij_1} \land \ldots \land X_m = E_{ij_m}))$. Each $p_j$ describes

$$\{ \text{OR} (Q_j \land \ldots \land (X_k = E_{ij_1}, \ldots, E_{ij_m}) \land \ldots \land (X_m = E_{ij_m})) \}_{j \in \Delta} \rho = \{ <Q_j, E_{ij_1}, \ldots, E_{ij_m}, \ldots, E_{ij_m} > : j \in \Delta \}

$$

We will denote this rule by the shorthand notation:

$$\{P \} X_k := E(X_1, \ldots, X_m) \{P(X_k + E(P(X_1), \ldots, P(X_m)))\}

$$

However, when there is no ambiguity on which context must be used to evaluate the expression $E$ we will write more simply:

$$\{P \} X_k := E(X_1, \ldots, X_m) \{P(X_k + E(X_1, \ldots, X_m))\}

$$

The other rules can be deduced in the same way. In particular, the operation $\cup$ describes the union $P \cup Q$ of two symbolic contexts $P = \{p_1, \ldots, p_r\}$ and $Q = \{q_1, \ldots, q_s\}$ that is the set $\{p_1, \ldots, p_r, q_1, \ldots, q_s\}$ where superfluous equivalent program paths are eliminated whereas inaccessible paths (the path condition of which is false) are removed.

6.3. EXAMPLE
Step 2:

\[ P_0^2 = \{ \langle \text{true}, \overline{x}, \overline{y} \rangle \} \]
\[ P_1^2 = \{ \langle \text{true}, \overline{x}, \overline{y} \rangle, \langle \overline{x} \overline{y}, \overline{x} \overline{y} \rangle \} \]
\[ P_2^2 = \{ \langle \overline{x} \overline{y}, \overline{x}, \overline{y} \rangle, \langle \overline{x} \overline{y} \rangle \text{ or } \langle \overline{x} \overline{y} \rangle \text{ and } \langle \overline{x} \overline{y}, \overline{x} \overline{y} \rangle \} \]
\[ P_3^2 = \{ \langle \overline{x} \overline{y}, \overline{x}, \overline{y} \rangle, \langle \overline{x} \overline{y} \rangle \text{ and } \langle \overline{x} \overline{y}, \overline{x} \overline{y} \rangle \} \]

So that at iteration 2 we have built the following symbolic execution tree (Hantler & King[1976]):

\[
\begin{array}{c}
\text{true,} \overline{x}, \overline{y} \\
\text{true,} \overline{x}, \overline{y} \\
\langle \overline{x} \overline{y}, \overline{x}, \overline{y} \rangle \\
\langle \overline{x} \overline{y}, \overline{x}, \overline{y} \rangle \\
\langle \overline{x} \overline{y} \rangle \text{ or } \langle \overline{x} \overline{y} \rangle \text{ and } \langle \overline{x} \overline{y}, \overline{x} \overline{y} \rangle \\
\langle \overline{x} \overline{y} \rangle \text{ and } \langle \overline{x} \overline{y}, \overline{x} \overline{y} \rangle \\
\langle \overline{x} \overline{y} \rangle
\end{array}
\]

We have represented the symbolic context \( P_1 \) associated with program point \( i \) by the set of paths associated with each of the nodes labelled \( i \) in the above execution tree. Equivalently we could have represented the symbolic context associated with

\[
\{ (P(x^k) \text{ or } P(x^m)) \text{ and } \forall k, P(x^{km}) \Rightarrow P(x^{(k+1)m}) \} \Rightarrow (P(\lim x^m))
\]

(Yet \( P \) must be an admissible predicate chosen in order to remain true when passing to the limit. Rigorously we should apply the second principle of transfinite induction).

Example: Let us prove the trivial fact that \( \forall x(1, (x \rightarrow x)) \forall y, \overline{x} \overline{y} \) at point 3 of program 6.3.

Base: For the single path of \( P_1 \) we have \( \forall x(1, \overline{x} \overline{y}) \).

Induction step: We assume that at step \( k \) the symbolic context \( P_k \) is equal to \( \{ p_{i_1}, x_{i_1}, y_{i_1} \} : i \text{ in } D \) with the induction hypothesis \( \{ \text{for } x, \forall y, \forall x(1, (x \rightarrow x)) \forall y, \overline{x} \overline{y} \} \). The equations 6.3 allow the computation of \( P_{k+1} \):

\[
P_1^k = \{ p_1, x_{i_1}, y_{i_1} : i \text{ in } D \}
\]
\[
P_1^{k+1} = \{ \text{true,} \overline{x}, \overline{y}, p_1, x_{i_1}, y_{i_1} : i \text{ in } D \}
\]
\[
P_2^k = \{ \langle \overline{x} \overline{y}, \overline{x}, \overline{y} \rangle, p_1, x_{i_1}, y_{i_1} : i \text{ in } D \}
\]
\[
P_2^{k+1} = \{ \langle \overline{x} \overline{y}, \overline{x}, \overline{y} \rangle, (p_1 \text{ and } x_{i_1} \overline{y}_{i_1}), x_{i_1}, y_{i_1} : i \text{ in } D \}
\]

We must show that the hypothesis holds for all paths of \( P_3 \). This is trivial for the path \( \langle \overline{x} \overline{y}, \overline{x}, \overline{y} \rangle \).
7. SYNTHESIS OF OPTIMAL INARIANT ASSERTIONS: THE
USE OF DIFFEREECE EQUATIONS

7.1. DISCOVERY OF OPTIMAL SYMBOLIC CONTEXTS

\[ f_2^{i+1}(P_0) = f_2^i(<p_{ij}, x_{ij}, y_{ij}: j \in D_i>) \]
\[ = \{ <p_{ij}, \text{and } x_{ij} \geq y_{ij}, x_{ij} - y_{ij}, y_{ij}: j \in D_i> \} \]
We can express \( P_1^\infty \) by the following equivalent forms:

\[
P_1^\infty = \bigcup_{i \in \mathbb{N}} (f_1 f_3^i (f_6 f_5^i f_3)^i f_2)^j (P_0)
\]

(i and j correspond to individual counters for the loops)

\[
P_1^\infty = \bigcup_{k \in \mathbb{N}} f_1 (P_0) \cup f_4 f_5 f_3 (f_2 f_4 f_2 f_5 f_3)^k f_2 f_1 (P_0)
\]

(k is a common counter for the two loops).

Although these two forms of \( P_1 \) are equivalent one of them will be more suitable for generating the difference equations and the choice depends on the semantics of the considered program.

### 7.3. EXAMPLE

Let us consider the following program (taken from King\[1969\]): (\( \div \) is integer division with truncation)

\[
\begin{align*}
&z := 1; \\
&\{0\} \\
&\text{loop: } \{1\} \\
&\quad \text{if } y \not\equiv 0 \text{ then} \\
&\quad \quad z := z \div x; \\
&\quad \text{fi; } \\
&\quad (y,x) := (y \div 2, x^2); \\
&\quad \text{go to loop;} \\
&\quad \text{fi;}
\end{align*}
\]

The system of equations associated with this program is:

\[
\begin{align*}
P_0 & = \{\text{true}, x=a, y=b, z=1\} \\
P_1 & = P_0 \cup P_6 \\
P_2 & = P_1 \text{ and } (P_1(y) \not\equiv 0) \\
P_3 & = P_2 \text{ and } odd(P_2(y)) \\
P_4 & = P_3(x + P_3(x) + P_3(x)) \\
P_5 & = (P_2 \text{ and } even(P_2(y))) \cup P_6 \\
P_6 & = P_5(y + P_5(y) \div 2, x + (P_5(x))^2) \\
P_7 & = P_1 \text{ and } (P_1(y) = 0)
\end{align*}
\]

Since \( P_2(y) = P_1(y) \), \( P_3(z) = P_1(z) \), \( P_4(x) = P_1(x) \), \( P_5(y) = P_1(y) \) and \( P_6(x) = P_1(x) \) we can simplify as follows:

\[
P_1 = P_6 \cup
\begin{align*}
&\{P_1 \text{ and } P_1(y) \not\equiv 0 \text{ and } even(P_1(y)) \}(y + P_1(y) \div 2, x + (P_5(x))^2, z + P_1(z)) \\
&\{P_1 \text{ and } P_1(y) \not\equiv 0 \text{ and } odd(P_1(y)) \}(y + P_1(y) \div 2, x + (P_5(x))^2, z + P_1(z) \star P_1(x))
\end{align*}
\]

Since the two alternatives differ only with respect to \( z \) we can factorize as follows:

\[
P_1 = P_0 \cup (P_1 \text{ and } y \not\equiv 0) (x \not\equiv 2, y \div 2, x \not\equiv 2, z \not\equiv (even(y) + 1) \not\equiv odd(y) \not\equiv x) \\
= P_0 \cup f(P_1)
\]

This formulation uses the conditional expressions of Sintzoff\[1975\]:

\[
\begin{align*}
&\text{true} \lor \text{false} = \text{true} \\
&\text{false} \lor \text{true} = \text{true} \\
&\text{false} \lor \text{false} = \text{false} \\
&\text{true} \lor \text{false} = \text{true} \\
&\text{false} \lor \text{true} = \text{true} \\
&\text{true} \lor \text{true} = \text{true} \\
&\text{true} \lor \text{false} = \text{true} \\
&\text{false} \lor \text{false} = \text{false} \\
&\text{true} \lor \text{true} = \text{true}
\end{align*}
\]

The general form of the solution for the equation defining \( P_1 \) is \( P_1 = \bigcup_{i=0}^\infty f_i(P_0) \). For determining \( f_i(P_0) \) we use the difference equations:

\[
f_i(P_0) = \begin{cases}
\text{true, } a, b, 1 \\
\text{false, } b, r, 2
\end{cases}
\]

\[
f_i+1(P_0) = \begin{cases}
f_i, f_i \\
q_i(x_1, y_1, z_i)
\end{cases}
\]

\[
f_i+1 = \begin{cases}
r_i, \text{odd}(y_1) \text{ to } 1 \\
r_i, \text{odd}(y_1) \text{ to } x_1
\end{cases}
\]

The formal resolution proceeds as follows:

\[
\begin{align*}
x_0 & = a \\
x_{i+1} & = x_i^2 \\
y_0 & = b \\
y_{i+1} & = y_i^2 \\
z_0 & = 1 \\
z_{i+1} & = z_i \star (even(b \div 2)^i + 1 \lor odd(b \div 2)^i + s_i^2) \\
z_i & = \begin{cases}
1, & \text{true} \\
0, & \text{false}
\end{cases}
\end{align*}
\]

\[
q_i = \begin{cases}
q_i, & r_i \not\equiv 0 \\
q_i, & \text{odd}(b \div 2)^i \not\equiv 0
\end{cases}
\]

\[
q_i = (b \div 2)^i \not\equiv 0 \text{ when } (i > 0)
\]
Finally the optimal output invariant of the program is:

\[ P_7 = P_1 \land (y=0) \]

\[ = \bigcup_{i=0}^{n} a_{i} \land (y_{i}=0), x_{i}, y_{i}, z_{i} \)

\[ = \bigcup_{b=0}^{1}, a_{b}, y_{b}, z_{b}=1 \]

\[ \cup ((3i+1) \land (b+2^{i} l=0), (b+2^{l} l=0)) \]

\[ x=a_{i}^{l}, \]

\[ y=(b+2^{l}) \]

\[ z= \bigcup_{j=0}^{l} (even(b+2^{j}) \rightarrow 1 \cup odd(b+2^{j}) \rightarrow 2^{j}) \]

Since the path condition at the halt point \{7\}: (b=0) or \{4i>0: (b+2^{l} i=0) and (b+2^{l} i=0)\} is true for any input value \( b \) of \( y \), the program always terminates.

It remains to show that \( P_7(z) = a \land b \) to prove total correctness. In the non-obvious case \( b \neq 0 \), we have to know that every number \( b \) can be expressed in binary form:

\[ b = ((3i : (b+2^{l} i=0) \cup (b+2^{l} i=0)) \rightarrow \text{sign}(b) \sum_{i=0}^{l} \bigcap_{j=0}^{l} (even(b+2^{j}) \rightarrow 1 \cup odd(b+2^{j}) \rightarrow 2^{j})) \]

Evaluating \( a \land b \) using the property that:

\[ a_{i}^{l} = 2^{l} \]

we get:

\[ a_{i}^{l} \land b = ((3i : (b+2^{l} i=0) \cup (b+2^{l} i=0)) \rightarrow \text{sign}(b) \sum_{i=0}^{l} \bigcap_{j=0}^{l} (even(b+2^{j}) \rightarrow 1 \cup odd(b+2^{j}) \rightarrow 2^{j})) \]

which is the desired result.

8. SYNTHESIS OF APPROXIMATE INVARIANTS

Let us consider a system of equations \( X=F(X) \) associated with a given program. We have seen that a set \( P \) of approximate invariants must satisfy \( \text{LP}(F) \subseteq P \), that is \( \text{LP}(F) \subseteq P \) with lattice notations. We can obtain such a \( P \) by "strengthening" the terms of a chaotic iteration sequence.

DEFINITION 8.1 A strengthened chaotic iteration corresponding to the continuous operator \( F \in \mathbb{R}^{n} \rightarrow \mathbb{R}^{n} \) and starting with a given vector \( X^0 \) such that \( X^{k} \in \text{Fix}(X) \) is a sequence \( X^{k} \), \( k=0,1, \ldots \) of vectors of \( \mathbb{R}^{n} \) defined recursively by \( X^{k+1} = F^{k}(X) \) where \( \forall k \in [0, \infty] \),

\[ \text{if } (i \in J_{k} \text{ and } F_{i}(X^{k}) \in X^{k}_{i}) \text{ or } (i \notin J_{k} \text{) then } \]

\[ F^{k}(X^{k}) \in X^{k} \]

\[ \text{else } X^{k}_{1} \cup F_{1}(X^{k}) \in F^{k}(X^{k}) \]

and \( J_{k}, k=0,1, \ldots \) is a sequence of subsets of \( \{1, \ldots, n\} \) such that \( \forall k \in [0, \infty] \), \( \forall i \in [1, n] \), \( i \notin J_{k+i} \).

DEFINITION 8.2 A strengthened chaotic iteration is said to stabilize after \( s \) steps if and only if \( \forall k \geq s : (X^{k} \in \mathbb{R}^{m} X^{k}) \) and \( \forall r \leq s : (X^{k} \in \mathbb{R}^{m} X^{k}) \).

Notice that stabilization can always be enforced. Proof: take \( X^{s} \in \mathbb{R}^{m} X^{s} \) (but this choice is of no practical interest).

THEOREM 8.3 The limit \( X^{S} \) of a chaotic iteration sequence \( X^{0}, \ldots, X^{k}, \ldots \) which stabilizes after \( s \) steps is such that \( \text{LP}(F) \subseteq X^{S} \).

Proof: Let us first prove that a chaotic iteration sequence is an increasing chain.

Basis: since \( X^{0} \in F(X^{0}) \), we have \( X^{0} \subseteq F_{0}(X^{0}) \subseteq F^{s}(X^{0}) \subseteq X^{1} \).

Induction step: suppose that \( X^{k} \subseteq X^{k+1} \). We have \( X^{k+1} \subseteq F_{1}(X^{k+1}) \). \( 

\forall i \in [1, n] \), \( i \notin J_{k+i} \). If \( i \in J_{k+i} \) then \( F_{i}(X^{k+i}) \subseteq F^{k+i}(X^{k+i}) \) else \( F_{i}(X^{k+i}) \subseteq F^{k+i}(X^{k+i}) \) proving that \( X^{k+i} \subseteq X^{k+i} \). Hence by recurrence on \( k \) : \( X^{S} \subseteq X^{S+i} \). According to the definition of a chaotic iteration sequence, this implies \( X^{S} \subseteq \mathbb{R}^{m} X^{S+i} \) which implies \( X^{S} \subseteq \mathbb{R}^{m} X^{S+i} \). Let us now prove that \( F(X^{S}) \subseteq X^{S} \). \( 

\forall i \in [1, n] \), \( 3\epsilon \in [0, m] \) such that \( i \notin J_{k+i} \). Therefore \( F_{i}(X^{S+i}) \subseteq X^{S+i} \). \( F_{i}(X^{S+i}) \subseteq F^{S+i}(X^{S+i}) \) since \( X^{S+i} \subseteq \mathbb{R}^{m} X^{S+i} \). Hence in both cases \( \forall i \in [1, n] \), \( F_{i}(X^{S+i}) \subseteq X^{S+i} \) proving that \( F(X^{S}) \subseteq X^{S+i} \). Also, according to Tarski's theorem \( \text{LP}(F) \subseteq X^{S} \). We have shown that \( \text{LP}(F) \subseteq X^{S} \).
approximation sequence at §3.4:

\[ p_2^\delta = \text{false} \]

\[ p_2^\delta = (x \geq y) \land (x = \overline{y}) \land (y = \overline{y}) \]

\[ p_2^\delta = [(x \geq y) \land (x = \overline{y}) \land (y = \overline{y})] \]

\[ \lor \]

\[ (\forall k \in [1, 2]: x = ky) \land (x = \overline{y}) \land (y = \overline{y}) \]

\[ p_2^\delta = [(x \geq y) \land (x = \overline{y}) \land (y = \overline{y})] \]

\[ \lor \]

\[ (\forall k \in [1, 2]: x = ky) \land (x = \overline{y}) \land (y = \overline{y}) \]

\[ \lor \]

\[ (\forall k \in [1, 3]: x = ky) \land (x = \overline{y}) \land (y = \overline{y}) \]

An easy guess is that \( p_2^\delta \Rightarrow [\exists k \in [0, 2]: (x = \overline{y} - k \overline{y}) \land (y = \overline{y})] \). Therefore, taking this assertion as a strengthened version of \( p_2^\delta \) we compute \( p_2^\alpha \):

\[ p_2^\alpha = [\exists v : [\exists k \in [0, 2]: (v = \overline{y} - k \overline{y}) \land (y = \overline{y}) \land (x = v - y)] \]

\[ = [\exists k \in [1, 3]: (x = \overline{y} - k \overline{y}) \land (y = \overline{y})] \]

\[ p_2^\alpha = [(x \geq y) \land (x = \overline{y}) \land (y = \overline{y})] \]

\[ \lor \]

\[ (\forall k \in [1, 3]: (x = \overline{y} - k \overline{y}) \land (x = \overline{y} - k \overline{y}) \land (y = \overline{y})] \]

Since \( p_2^\alpha \) does not imply \( p_2^\delta \) we strengthen \( p_2^\alpha \) to get:

\[ p_2^\alpha = [\exists k \geq 0 : (x = \overline{y} - k \overline{y}) \land (y = \overline{y})] \]

Iterating again we obtain:

\[ p_2^\alpha = [(x \geq y) \land (x = \overline{y}) \land (y = \overline{y})] \]

\[ \lor \]

\[ (\forall k \geq 0: (x = \overline{y} - k \overline{y}) \land (x = \overline{y} - k \overline{y}) \land (y = \overline{y})] \]

The idea of solving these equations by means of chaotic successive approximations is so natural that it was first understood as a program execution on symbolic values. We think that even a semi-automatic resolution of the semantic equations is a terrific task which result can hardly be controlled by the programmer. Therefore an important idea for automating the resolution of these equations is the one of approximation. Compile time checks, global dataflow analysis, program performance prediction or proofs of partial correctness are examples of approximate analyses of programs. It is our experience that the exact or total properties of a program can often be determined by addition of several approximate analyses of specific properties. The determination of a range of complementary logical properties of program should allow the analysis of program properties by successive approximations. Each of these specific properties should be simple enough to allow the automatic resolution of the corresponding equations. The spectrum of these properties should be wide enough to cover the total semantics of programs. Interaction with the programmer for guiding the


Kleene[1952]. Kleene, S.C. Introduction to meta-
Proceedings of the Symposium on
Artificial Intelligence and Programming Languages

Association for Computing Machinery
The papers in this volume were presented at the ACM Symposium on Artificial Intelligence and Programming Languages, sponsored jointly by SIGART and SIGPLAN, and held at the University of Rochester, August 15-17, 1977.

The Program Committee wishes to thank all those who submitted papers for consideration.