stichting
mathematisch
centrum

MC

AFDELING INFORMATICA (DEPARTMENT OF COMPUTER SCIENCE)   IW 149/80   OKTOBER

J.A. BERGSTRA & J.V. TUCKER

EXPRESSIVENESS AND THE COMPLETENESS OF HOARE'S LOGIC

Preprint

kruislaan 413 1098 SJ amsterdam

Printed at the Mathematical Centre, 413 Kruislaan, Amsterdam.

The Mathematical Centre, founded the 11-th of February 1946, is a non-profit institution aiming at the promotion of pure mathematics and its applications. It is sponsored by the Netherlands Government through the Netherlands Organization for the Advancement of Pure Research (Z.W.O.).

1980 Mathematics subject classification: 035D35, 03D75, 68B10

ACM-Computing Review-category: 5.24

Expressiveness and the completeness of Hoare's logic[*)]

by

J.A. Bergstra[**)] & J.V. Tucker

ABSTRACT

We prove some theorems which reconsider the completeness of Hoare's logic for the partial correctness of while-programs when equipped with a first-order assertion language. The results are about the expressiveness of the assertion language and the rôle of specifications in completeness concerns for the logic: (1) expressiveness is not a necessary condition on a structure for its Hoare logic to be complete; (2) complete number theory is the only extension of Peano Arithmetic which yields a logically complete Hoare logic; (3) a computable structure with enumeration is expressive if, and only if, its Hoare logic is complete.

KEY WORDS & PHRASES: *Hoare's logic for while-programs, soundness, completeness, expressiveness, specifications, arithmetical computation*

[*)] This report is not for review as it will be submitted for publication elsewhere.

[**)] Department of Computer Science, University of Leiden, Wassenaarseweg 80,
Postbus 9512, 2300 RA LEIDEN, The Netherlands

INTRODUCTION

With the term *Hoare's logic* we mean the formal system for the manipulation of statements about the partial correctness of while-programs which was first described in HOARE [10]. In this paper we shall be concerned with the mathematical structure of this logic when it is equipped with a first-order assertion language L, and is set to analyse computation on essentially arithmetical structures by the members of the set of while-programs $\mathcal{WP}$. We will prove some theorems about the relationship between the expressiveness of the assertion language, the specifications of the structures, and the completeness of the logic itself: theorems which are technical comments on the nature of the completeness of the formal system, but which also reflect on two informal attitudes to data abstraction available when working with Hoare's logic.

The starting point for any mathematical study of Hoare's ideas is the seminal paper COOK [7] where the various syntactic and semantic components associated with the system were carefully examined, and the soundness of logic properly proved. Of particular interest to us is the rôle of the *oracle* or *structural specification* in Hoare's logic. This is a set $\mathcal{O}$ of assertions used in connection with the Rule of Consequence, and it is intended to formalise what information about data types is available to correctness proofs for the programs the data types support (cf. HOARE [10], Section 2). From [7], we know that *if* $\mathcal{O}$ *sound for a structure* A *then Hoare's logic* HL($\mathcal{O}$) *is sound for* $\mathcal{WP}$ *over* A, too. Up to the choice of program semantics for while-programs, Cook's analysis of Hoare's ideas is general, faithful and definitive. (For information on the issues involved in the choice of semantics consult GREIF & MEYER [9].)

Of course, in [7], Cook also considered the completeness of Hoare's logic, but with apparently less satisfying theoretical results: *under the hypotheses that* $\mathcal{O}$ *is a complete specification for structure* A *and that* L *is expressive for* $\mathcal{WP}$ *over* A *then* HL($\mathcal{O}$) *is complete for* $\mathcal{WP}$ *over* A. For example, if the standard arithmetic N of the natural numbers is specified by its first-order theory Th(N), called *complete number theory*, then its Hoare logic is complete for $\mathcal{WP}$ over N since L is expressive for $\mathcal{WP}$ over N.

Much theoretical effort has been expended in coming to terms with this assumption of expressiveness and with the paucity of expressible structures; and, by extension, in evaluating the kind of completeness Cook was able to provide. The writing on this theme is quite extensive, but one can usefully consult the invaluable survey article APT [1] to obtain a clear picture of current opinion. In summary, the basic material about while-programs is contained in WAND [18] and our own [4] (on incompleteness); and in LIPTON [12] (on expressiveness). In the case of a richer programming formalism the situation is far more perplexing since expressiveness can fail to be a sufficient condition for the existence of any kind of complete Hoare logic: see CLARKE [6]; but there remains a common ground of complex languages with complete Hoare logics over expressive structures and this has been charted in LIPTON [12], LANGMAACK & OLDEROG [11], in the monograph DE BAKKER [2] and, again, in APT [1]. Let us concentrate on the simple facts of life for while-programs.

Although expressiveness is not an unnatural condition from the point of view of computing on a structure, is it actually necessary for the completeness of the structure's Hoare logic?

THEOREM 1. *Expressiveness is not a necessary condition on a structure for the completeness of its Hoare logic. For any model* A *of complete number theory Hoare's logic is complete, but if* A *is not the standard model of arithmetic then* L *is not expressive for* $\mathcal{WP}$ *over* A.

Now Theorem 1 illuminates a certain change of status for Hoare's logic in the passage from the Soundness Theorem to the Completeness Theorem; from a system of reasoning based purely upon a data type specification to a system of reasoning based upon a fixed data type which is appropriately specified. The alteration is effected by the kind of completeness sought for Hoare's logic: the set of valid asserted programs is defined by a single structure and not by the class of all models of a specification, as one expects to see in a "true" converse to the Soundness Theorem. The property of expressiveness certainly underlines this semantic emphasis, but expressiveness does not determine it and from Theorem 1 one can deduce this other kind of completeness is possible for arithmetic:

Let us say that a Hoare logic HL($\mathcal{O}$) is *logically complete* if any asserted program which is valid on *all* models of the specification $\mathcal{O}$ is provable in HL($\mathcal{O}$).

THEOREM 2. *Complete number theory is the* only *extension* T *of Peano Arithmetic for which* HL(T) *is logically complete.*

There are two attitudes toward data abstraction to govern one's work with Hoare's logic: given that some specification is a necessary constituent of the proof system, one may think of the specification, and hence the logic, as an instrument to analyse computation on a particular structure; or one may think of the specification as an abstract characterisation of an ill-defined class of legitimate implementations. Both ideas are commonplace in the literature on the semantics of data types, of course.

From the point of view of computing on a given structure, the examples in Theorem 1 can hardly qualify as interesting data types in their own right; they are not computable for example: see [16]. (Nevertheless, non-standard models of arithmetic are part and parcel of the concern for correctness simply because, *mathematically,* a Hoare logic HL($\mathcal{O}$) is not a system of reasoning about one particular structure (say, N) but about the class of all models of the specification $\mathcal{O}$ (say, Th(N))).

A close reading of Hoare's work in this area suggests that he intended his calculus to be a system of reasoning about programs running on *any legal implementation of the specification.* If we interpret a legal implementation of a specification $\mathcal{O}$ as simply a computable model of $\mathcal{O}$ then we have a mathematically intermediate notion of completeness in which validity is based upon all computable models of $\mathcal{O}$. In the case of arithmetic this collapses to a particular structure where expressiveness and completeness occur together. As it turns out this is a general phenomenon and, in particular, we have further reassurance of the usefulness of Cook's study of completeness:

THEOREM 3. *Let* A *be an infinite computable structure. Then* A *can be augmented by a computable enumeration, consisting of a distinguished constant* first *and a unary injective operator* next: A → A *such that* $A = \{\underline{\text{next}}^n(\underline{\text{first}}) : n \in \omega\}$, *to make a new structure* A_e *with the result that* L *is expressive for* $\mathcal{WP}$ *over* A_e *if, and only if,* HL(A_e) *is complete for* A.

We have greatly prolonged this introduction to accommodate our observations on the semantic and syntactic rôles of specifications; henceforth we deal with mathematical issues only. The first two sections concern the construction and basic properties of Hoare's logic while the next three sections technically discuss completeness and prove the theorems announced. Obviously, we are assuming the reader to be familiar with HOARE [10] and COOK [7], but little other knowledge is actually necessary. This paper is a close companion of our [4] about natural structures which possess no complete Hoare-like logics for their <u>while</u>-programs; and both papers are sequels to our [3] written with J. Tiuryn.

1. PRELIMINARIES ON ASSERTIONS AND PROGRAMS

In this and the next section we map out the technical prerequisites for the paper. In addition to the important sources HOARE [10], COOK [7], the reader would do well to consult the survey article APT [1].

The first-order language $L = L(\Sigma)$ of some signature Σ is based upon a set of variables $x_1, x_2, \ldots$ and its constant, function and relational symbols are those of Σ together with the boolean constants <u>true</u>, <u>false</u> and the equality relation. We assume L possesses the usual logical connectives and quantifiers; and the set of all algebraic terms of L we denote $T(\Sigma)$.

Using the syntax of L, the set $\mathcal{WP} = \mathcal{WP}(\Sigma)$ of all <u>while</u>-programs over Σ is defined in the customary way.

For any structure A of signature Σ, the semantics of the first-order language L over Σ as determined by A has its standard definition in model theory and this we assume to be understood. The validity of $\phi \in L$ over structure A we write $A \models \phi$.

If $\mathcal{O}$ is a set of assertions of L then the set of all formal theorems of $\mathcal{O}$ is denoted $\mathrm{Thm}(\mathcal{O})$; we write $\mathcal{O} \vdash \phi$ for $\phi \in \mathrm{Thm}(\mathcal{O})$. Such a set $\mathcal{O}$ of formulae is usually called a theory, but in the present context we prefer the more suggestive term *specification*. Two specifications $\mathcal{O}, \mathcal{O}'$ are *logically equivalent* if $\mathrm{Thm}(\mathcal{O}) = \mathrm{Thm}(\mathcal{O}')$. A specification $\mathcal{O}$ is *complete* if given any assertion $\phi \in L$, either $\mathcal{O} \vdash \phi$ or $\mathcal{O} \vdash - \phi$. The set $\mathrm{Th}(A)$ of all assertions true of a structure A is called the first-order theory of A; evidently $\mathrm{Th}(A)$ is a complete specification. The class of all models

$\mathcal{O}$ is denoted Mod($\mathcal{O}$); we write Mod($\mathcal{O}$) $\models \phi$ to mean that for every A $\in$ Mod(T), A $\models \phi$. Gödel's Completeness Theorem says this about specifications:

$$\mathcal{O} \vdash \phi \text{ if, any only if, } \mathrm{Mod}(\mathcal{O}) \models \phi.$$

For a proper discussion of these concepts the reader should consult CHANG & KEISLER [5].

For the semantics of $\mathcal{WP}$ as determined by a structure A, we leave the reader free to choose any sensible account of while-program computations which applies to an arbitrary structure: COOK [7]; the graph-theoretic semantics in GREIBACH [8]; the denotational semantics described in DE BAKKER [2]. What constraints must be placed on this choice are the necessities of formulating and proving certain lemmas, such as Lemma 1.1, and of verifying the soundness of Hoare's logic (Theorem 2.2). These conditions will be evident from the text and, for such a simple programming formula as $\mathcal{WP}$, can hardly be problematical. For definiteness, we have in mind a naïve operational semantics based upon appropriate A-register machines which yield straightforward definitions of a *state* in a $\mathcal{WP}$ computation and of the *length* of a $\mathcal{WP}$ computation [17]. Thus, if S $\in \mathcal{WP}$ involves n program variables and computes on structure A then we use elements of A^n to represent states in the computations of S. For a $\in A^n$, the length of the computation S(a) is denoted |S(a)|. The proof of the following fact is a routine matter:

1.1. LEMMA. *Let* S $\in \mathcal{WP}$ *involve variables* $x = (x_1,\ldots,x_n)$. *Then for each* $\ell \in \omega$ *one can effectively find a formula* $\mathrm{COMP}_{S,\ell}(x,y)$ of L, *wherein* $y = (y_1,\ldots,y_n)$ *are new variables, such that for any* A *and any* $a,b \in A^n$, $A \models \mathrm{COMP}_{S,\ell}(a,b)$ *if, and only if, the computation* S(a) *terminates in* ℓ *or less steps leaving the variables with values* $b = (b_1,\ldots,b_n)$.

From the syntax L and $\mathcal{WP}$, we make a new kind of syntactic object called the *asserted program*; this is a triple of the form {p}S{q} where p,q $\in$ L and S $\in \mathcal{WP}$ and the variables of p,q and S are the same. To the asserted programs we assign *partial correctness semantics*: the asserted program {p}S{q} is *valid on a structure* A (in symbols: A $\models$ {p}S{q}) if for each initial state a $\in A^n$, A $\models$ p(a) implies either S(a) terminates and

$A \models q(S(a))$ or $S(a)$ diverges. And the asserted program $\{p\}S\{q\}$ is *valid for a specification* $\mathcal{O}$ if it is valid on *every* model of $\mathcal{O}$; in symbols, $\mathcal{O} \models \{p\}S\{q\}$ or $\mathrm{Mod}(\mathcal{O}) \models \{p\}S\{q\}$.

The *partial correctness theory of a structure* A is the set

$$PC(A) = \{\{p\}S\{q\}: A \models \{p\}S\{q\}\};$$

and the *partial correctness theory of a specification* $\mathcal{O}$ is the set

$$PC(\mathcal{O}) = \{\{p\}S\{q\}: \mathrm{Mod}(\mathcal{O}) \models \{p\}S\{q\}\}.$$

Clearly,

$$PC(\mathcal{O}) = \bigcap_{A \in \mathrm{Mod}(\mathcal{O})} PC(A).$$

Finally, we define strongest postconditions. Let $\phi \in L$ and $S \in \mathcal{WP}$, both having n variables. The *strongest postcondition* of S and ϕ on a structure A is the set

$$sp_A(\phi,S) = \{b \in A^n: \exists a \in A^n.[S(a) \text{ terminates in final state } b \text{ and } A \models \phi(a)]\}$$

<u>1.2 LEMMA.</u> $A \models \{p\}S\{q\} \iff sp_A(p,S) \subset \{b \in A^n: A \models q(b)\}$.

2. HOARE'S LOGIC

Hoare's logic for <u>while</u>-programs over Σ, with assertion language L and specification or oracle $\mathcal{O} \subset L$, has the following axioms and proof rules for manipulating asserted programs: let $S, S_1, S_2 \in \mathcal{WP}$; $p,q,p_1,q_1,r \in L$; $b \in L$, a quantifier-free formula.

1. <u>Assignment axiom scheme</u>: for $t \in T(\Sigma)$ and x a variable of L, the asserted program

$$\{p[t/x]\}x := t\{p\}$$

is an axiom, where $p[t/x]$ stands for the result of substituting t for free occurrences of x in p.

2. Composition rule:

$$\frac{\{p\}S_1\{r\},\{r\}S_2\{q\}}{\{q\}S_1;S_2\{q\}}$$

3. Conditional rule:

$$\frac{\{p\wedge b\}S_1\{q\},\{p\wedge\neg b\}S_2\{q\}}{\{p\}\ \underline{\text{if}}\ b\ \underline{\text{then}}\ S_1\ \underline{\text{else}}\ S_2\ \underline{\text{fi}}\ \{q\}}$$

4. Iteration rule:

$$\frac{\{p\wedge b\}S\{p\}}{\{p\}\ \underline{\text{while}}\ b\ \underline{\text{do}}\ S\ \underline{\text{od}}\ \{p\wedge\neg b\}}$$

5. Consequence rule:

$$\frac{p \to p_1,\{p_1\}S\{q_1\},\ q_1 \to q}{\{p\}S\{q\}}$$

And, in connection with 5,

6. Oracle axiom: Each member of $\mathrm{Thm}(\mathcal{O})$ is an axiom.

The set of asserted programs derivable from these axioms by the proof rules we denote $\mathrm{HL}(\mathcal{O})$; we write $\mathrm{HL}(\mathcal{O}) \vdash \{p\}S\{q\}$ in place of $\{p\}S\{q\} \in \mathrm{HL}(\mathcal{O})$.

2.1 BASIC UNIQUENESS LEMMA. *For any consistent specifications $\mathcal{O}$ and $\mathcal{O}'$ which are logically equivalent we have that $\mathrm{HL}(\mathcal{O}) = \mathrm{HL}(\mathcal{O}')$ and $\mathrm{PC}(\mathcal{O}) = \mathrm{PC}(\mathcal{O}')$.*

PROOF. The equality of Hoare logics over logically equivalent specifications is obvious. If $\mathrm{Thm}(\mathcal{O}) = \mathrm{Thm}(\mathcal{O}')$ then $\mathrm{Mod}(\mathcal{O}) = \mathrm{Mod}(\mathcal{O}')$, by the soundness of first-order logic. Therefore $\mathrm{PC}(\mathcal{O}) = \mathrm{PC}(\mathcal{O}')$.

The Corollary to Theorem 1 in COOK [7] says this:

2.2 SOUNDNESS THEOREM. *For any specification $\mathcal{O}$, $\mathrm{HL}(\mathcal{O}) \subset \mathrm{PC}(\mathcal{O})$.*

The Hoare logic $HL(\mathcal{O})$ is said to be *logically complete* if $HL(\mathcal{O}) = PC(\mathcal{O})$. As noted in the Introduction, Cook chose to consider the completeness of Hoare's logic relative to a fixed structure rather than its logical completeness; we state Theorem 3 in [7]:

The assertion language L is *expressive* for $\mathcal{WP}$ over structure A if for every $\phi \in L$ and every $S \in \mathcal{WP}$, the strongest post-condition $sp_A(\phi,S)$ is first-order definable over A.

2.3 COOK'S COMPLETENESS THEOREM. *For any structure* A, *if* $\mathcal{O}$ *is a complete specification for* A *in the sense that* $Thm(\mathcal{O}) = Th(A)$, *and if* L *is expressive for* $\mathcal{WP}$ *over* A, *then* $HL(\mathcal{O}) = PC(A)$.

Hoare's logic for while-programs over a structure A is defined to be HL(Th(A)) and is denoted HL(A). From the Soundness Theorem 2.2, we know that

$$HL(A) = HL(Th(A)) \subset PC(Th(A)) \subset PC(A)$$

and the Completeness Theorem says that if L is expressive for $\mathcal{WP}$ over A then $HL(A) = PC(A)$.

Let $N = (\omega;0,x+1,x\dot{-}1\ +\ ,\times,\leq)$ to be standard model of arithmetic; the Corollary to Theorem 3 in [7] says:

2.4 COROLLARY. $HL(N) = PC(N)$.

3. COMPUTING ON A STRUCTURE

When using first-order logic to investigate properties of a given structure it must be kept in mind that the logical methods see the structure as an object unique up to elementary equivalence and not isomorphism. If A and B are structures of common signature then A is *elementary equivalent* to B (in symbols: $A \equiv B$) if $Th(A) = Th(B)$.

3.1 UNIQUENESS LEMMA FOR STRUCTURES.

If $A \equiv B$ *then* $HL(A) = HL(B)$ *and* $PC(A) = PC(B)$.

PROOF. The equality of the Hoare logics over elementary equivalent structures follows from the Uniqueness Lemma 2.1, and is obvious anyway. Consider the second statement about correctness theories.

Suppose $A \models \{p\}S\{q\}$ where $p,q \in L$, and $S \in \mathcal{WP}$ involves n program variables. Given S and $\ell \in \omega$ one can effectively find a while-program S_ℓ which applied to any input state of any Σ-structure A computes exactly as S computes for ℓ steps and then diverges if S has not terminated in that time. Thus, for $a \in A^n$,

$$S_\ell(a) = \begin{cases} S(a) & \text{if } |S(a)| \leq \ell \\ \text{undefined} & \text{otherwise.} \end{cases}$$

It is easy to prove from Lemma 1.1 the following fact:

3.2 LEMMA. *For any assertion* ϕ *of* L *one can effectively find a first-order formula* $SP(\phi,S_\ell)$ *which for every* Σ*-structure* A *defines the strongest postcondition* $sp_A(\phi,S_\ell)$.

Now define $SP(\phi,S) \equiv \bigvee_{\ell\in\omega} SP(\phi,S_\ell)$, an infinitary formula which uniformly defines the strongest postcondition of ϕ and S. We calculate as follows:

$$\begin{aligned} A \models \{p\}S\{q\} &\iff A \models SP(p,S) \rightarrow q && \text{by Lemma 1.2} \\ &\iff A \models [\textstyle\bigvee_{\ell\in\omega} SP(p,S_\ell)] \rightarrow q \\ &\iff A \models \textstyle\bigwedge_{\ell\in\omega}[SP(p,S_\ell) \rightarrow q] \\ &\iff \text{for every } \ell \in \omega,\ A \models SP(p,S_\ell) \rightarrow q \\ &\iff \text{for every } \ell \in \omega,\ B \models SP(p,S_\ell) \rightarrow q && \text{since } A \equiv B \\ &\iff B \models \textstyle\bigwedge_{\ell\in\omega}[SP(p,S_\ell) \rightarrow q] \\ &\iff B \models \{p\}S\{q\}. \end{aligned}$$

Q.E.D.

3.3 COROLLARY. *If* HL(A) *is complete and* $A \equiv B$ *then* HL(B) *is complete.*

PROOF. Assume HL(A) = PC(A). By Lemma 3.1, HL(A) = HL(B) and PC(A) = PC(B) so HL(B) = PC(B). Q.E.D.

Here is the first theorem of the Introduction.

3.4 THEOREM. *For every model* A *of complete number theory* Th(N), HL(A) *is complete; but if* A *is non-standard then* L *is not expressive for* $\mathcal{WP}$ *over* A.

PROOF. Any model A of complete number theory is elementary equivalent to the standard model N; thus, HL(A) is complete by Corollary 3.3 and Corollary 2.4. We show L is not expressive for A.

Let S be the following arithmetic program,

$$x:=y;\ \underline{\text{while}}\ x \neq 0\ \underline{\text{do}}\ x:=x\dot{-}1\ \underline{\text{od}};\ x:=y.$$

On the structure A, S attempts to count down from the value of y to 0: given initial state $(a,b) \in A^2$ if S terminates then its final state is (b,b) and b is a standard number in A; if b is non-standard then S diverges from initial state (a,b) for any $a \in A$.

Consider the set $sp_A(\underline{\text{true}},S)$. Inspecting its definition we find that for $(a,b) \in A^2$,

$$(a,b) \in sp_A(\underline{\text{true}},S) \iff a = b \text{ and } a \text{ is standard.}$$

Thus, $X = \{a \in A: (a,a) \in sp_A(\underline{\text{true}},S)\}$ is precisely the set of all standard numbers in A. If $sp_A(\underline{\text{true}},S)$ were first-order definable then, from the axioms of Peano Arithmetic, we could prove the existence of a least element of ¬ X. But A has no smallest non-standard element because each non-zero element has a predecessor. Q.E.D.

4. COMPUTING WITH A SPECIFICATION

Let us begin by establishing a general connection between the logical completeness of Hoare's logic based upon a specification and the completeness of the logic as it is determined by a particular structure.

4.1 THEOREM. *Let* $\mathcal{O}$ *be a consistent specification which is complete. Then for each* $A \in \text{Mod}(\mathcal{O})$ *it is the case that* $HL(\mathcal{O}) = HL(A)$ *and* $PC(\mathcal{O}) = PC(A)$. *In particular, the following three statements are equivalent:*

1. $HL(\mathcal{O}) = PC(\mathcal{O})$.
2. *For each* $A \in Mod(\mathcal{O})$, $HL(A) = PC(A)$.
3. *For some* $A \in Mod(\mathcal{O})$, $HL(A) = PC(A)$.

PROOF. If $A \in Mod(\mathcal{O})$ then $Thm(\mathcal{O}) = Th(A)$ because $\mathcal{O}$ is complete. On inspecting the appropriate definitions one sees that $HL(\mathcal{O}) = HL(A)$. Consider the correctness theories. The completenss of $\mathcal{O}$ implies $PC(\mathcal{O}) = PC(Th(A))$ and we must show that $PC(Th(A)) = PC(A)$. Now,

$$PC(Th(A)) = \bigcap_{B \in Mod(Th(A))} PC(B).$$

Since all models of Th(A) are elementary equivalent to A, the Uniqueness Lemma 3.1 reduces the intersection to $PC(Th(A)) = PC(A)$.

The equivalence in the theorem are easy corollaries of the first conclusions. Q.E.D.

From Cook's Completeness Theorem 2.3 we can deduce this next theorem.

4.2 THEOREM. *Let* $\mathcal{O}$ *be a consistent specification which is complete. Then if* $Mod(\mathcal{O})$ *contains an element* A *for which* L *is expressive for* $\mathcal{WP}$ *over* A *then* $HL(\mathcal{O})$ *is logically complete.*

Here is the second theorem stated in the Introduction.

4.3 THEOREM. *Complete number theory* Th(N) *is the only extension* T *of Peano arithmetic for which* HL(T) *is logically complete.*

PROOF. Hoare's logic based on complete number theory is logically complete by Theorem 4.1 and Corollary 2.4. We prove that for any extension T of Peano Arithmetic, if HL(T) is complete for Mod(T) then T satisfies the following ω-Rule: *let* ϕ *be any formula of* L *and let* $\underline{n}$ *denote the numeral in* L *corresponding to* $n \in \omega$.

$$\frac{T \vdash \phi(\underline{n}) \text{ for each } n \in \omega}{T \vdash \forall x \phi(x)}$$

With this ω-Rule it is a routine matter to show that $Thm(T) = Th(N)$. First, one proves that $Th(N) \subset Thm(T)$ by induction on the complexity of formulae and using the ω-Rule. This done, the equality $Th(N) = Thm(T)$ follows immediately from the completeness of $Th(N)$.

Let us prove the ω-Rule. Let ϕ be a formula and suppose $T \vdash \phi(\underline{n})$ for all $n \in \omega$. Let S denote the following program

$$y := 0;\ \underline{\text{while}}\ x \neq y\ \underline{\text{do}}\ y := y+1\ \underline{\text{od}}$$

and consider the asserted program

$$\{\neg\ \phi(x)\}S\{\underline{\text{false}}\}$$

First, we claim that $Mod(T) \models \{\neg\ \phi(x)\}S\{\underline{\text{false}}\}$. For if $M \in Mod(T)$, $m \in M$ and $M \models \neg\ \phi(m)$ then m is a non-standard element of M because we are assuming ϕ provable on all the standard numbers. Thus, the precondition $\neg\ \phi(x)$ guarantees that the program diverges and so the asserted program is valid.

Since HL(T) is complete for Mod(T) we know that

$$HL(T) \vdash \{\neg\ \phi(x)\}S\{\underline{\text{false}}\}.$$

We now unpick a formal proof of the asserted program in HL(T) and from its intermediate assertions put together a proof for $T \vdash \forall x.\phi(x)$. Starting from the conclusion of the Hoare logic proof we step backward 3 times always seeking theorems of T.

STEP I. By the Composition Rule, there must be a formula $\delta = \delta(x,y)$ containing free variables x,y, but also other unnamed variables, such that

(a) $HL(T) \vdash \{\neg\ \phi(x)\}\ y := 0\ \{\delta(x,y)\}$

(b) $HL(T) \vdash \{\delta(x,y)\}\ \underline{\text{while}}\ x \neq y\ \underline{\text{do}}\ y := y+1\ \underline{\text{od}}\ \{\underline{\text{false}}\}$

Clearly, (a) implies that

(c) $T \vdash \neg\ \phi(x) \wedge y = 0 \rightarrow \delta(x,0)$.

STEP II. Consider I(b). By the $\underline{\text{while}}$-Rule and the Rule of Consequence, an intermediate assertion $\Theta = \Theta(x,y)$ must exist to satisfy

(a) $T \vdash \delta \to \Theta$

(b) $HL(T) \vdash \{\Theta \wedge x \neq y\}\ y := y+1\{\Theta\}$

(c) $T \vdash \Theta \wedge x = y \to \underline{false}$

And this latter statement we rewrite

(d) $T \vdash \Theta \to x \neq y.$

STEP III. Consider II(b). This statement is derived via the Rule of Consequence from an appeal to an assignment axiom: there exists $\gamma = \gamma(x,y)$ such that

(a) $T \vdash \Theta \wedge x \neq y \to \gamma[y/y+1]$

(b) $HL(T) \vdash \{\gamma[y/y+1]\}\ y:= y+1\{\gamma\}$

(c) $T \vdash \gamma \to \Theta$

Now we can show that $T \vdash \forall x.\phi(x)$. This involves a little logical calculation with the 6 formal theorems of T which we organize around the following lemma

<u>4.4 LEMMA</u>. $T \vdash \neg\, \phi(x) \to \forall y.\Theta(x,y) \wedge x \neq y.$

Given this lemma, the remainder of the proof is simply a formal deduction:

$T \vdash \neg\, \phi(x) \to [\forall y.\Theta(x,y) \wedge x \neq y]$ this is Lemma 4.4;

$T \vdash [\forall y.\Theta(x,y) \wedge x \neq y] \to \forall y.x \neq y$

$T \vdash [\forall y.x \neq y] \to \underline{false}$

By transitivity of implication,

$T \vdash \neg\, \phi(x) \to \underline{false}$

$T \vdash \phi(x)$

Reinstating the universal quantifier we have $T \vdash \forall x.\phi(x)$.

<u>PROOF OF LEMMA 4.4</u>. We use the axiom scheme of induction belonging to Peano Arithmetic and which is available for T. It is enough to derive a

basis theorem and an induction step theorem

Basis: $T \vdash \neg\,\phi(x) \to [\theta(x,\underline{0}) \wedge x \neq \underline{0}]$

$T \vdash \neg\,\phi(x) \to \delta(x,\underline{0})$ from I(c);

$T \vdash \delta(x,\underline{0}) \to \theta(x,\underline{0})$ from II(a);

$T \vdash \neg\,\phi(x) \to \theta(x,\underline{0})$ by transitivity.

$T \vdash \neg\,\phi(x) \to x \neq \underline{0}$ since $T \vdash \phi(\underline{0})$.

Whence the basis theorem is obtained by conjoining these last two statements.

Induction step: $T \vdash [\theta(x,y) \wedge x \neq y] \to [\theta(x,y+1) \wedge x \neq y+1]$

$T \vdash [\theta(x,y) \wedge x \neq y] \to \gamma(x,y+1)$ this is III(a);

$T \vdash \gamma(x,y+1) \to \theta(x,y+1)$ from III(c);

$T \vdash [\theta(x,y) \wedge x \neq y] \to \theta(x,y+1)$ by transitivity.

$T \vdash \theta(x,y+1) \to x \neq y+1$ from II(d).

Whence the induction theorem is obtained from these last two statements.

Q.E.D.

5. EXPRESSIBILITY AND COMPLETENESS FOR COMPUTABLE STRUCTURES

If Hoare's intentions are not quite faithfully represented by the mathematics of Sections 3 and 4 then at least it adequately supports the suggestion, made in the Introduction, of defining a third kind of completeness from the class of computable models of a specification. Let $CPC(\mathcal{O})$ be the set of all asserted programs valid on all computable models of $\mathcal{O}$. Then Theorem 4.1 allows us to reduce completeness considerations of $HL(\mathcal{O})$ with respect to $CPC(\mathcal{O})$ to the case of an individual structure: if $\mathcal{O}$ is complete and possesses a computable model then for any $A \in Mod(\mathcal{O})$, $PC(A) = CPC(\mathcal{O}) = PC(\mathcal{O})$. So it is, we are led to take an interest in Hoare's logic over particular computable structures.

By an *enumeration* for a structure A we mean a distinguished element $\underline{\text{first}}$ of A and an injective operator $\underline{\text{next}}$: $A \to A$ such that $A = \{\underline{\text{next}}^n(\underline{\text{first}}) : n \in \omega\}$. By a *structure with enumeration* we mean a structure with such an enumeration named in its signature. In this last section we will prove this theorem.

5.1 THEOREM. *Each infinite computable structure possesses a computable enumeration. If* A *is a computable structure with enumeration then* L *is expressive for* $\mathcal{WP}$ *over* A *if, and only if,* HL(A) = PC(A).

Finite structures are computable, of course, and as L is always expressive for them their Hoare logics are always complete. Presburger Arithmetic is the simplest computable structure with enumeration; L is not expressive for it and its Hoare logic is incomplete. For the standard model of arithmetic N, the ring of integers, and the field of rational numbers, L is expressive and Hoare's logic is complete. But for the fields of real algebraic numbers and algebraic numbers, L is again not expressive and Hoare's logic is incomplete, [4].

Of course, before proving Theorem 5.1 we are obliged to say something about computable structures. Our definition is the standard *formal* definition of the concept of a computable structure and it derives from RABIN [15] and MAL'CEV [13].

A structure A is *computable* if there exists a recursive subset Ω of the set of natural numbers ω and a surjection $\alpha:\Omega\to A$ such that (1) the relation $\equiv_\alpha$ defined on Ω by $n\equiv_\alpha m \Longleftrightarrow \alpha n = \alpha m$ in A is recursive; and (2) for each k-ary operation σ and each k-ary relation R of A there exist recursive functions $\hat{\sigma}$ and $\hat{R}$ which commute the following diagrams

$$\begin{tikzcd}
A^k \arrow[r, "\sigma"] & A \\
\Omega^k \arrow[u, "\alpha^k"] \arrow[r, "\hat{\sigma}"] & \Omega \arrow[u, "\alpha"']
\end{tikzcd}
\qquad\qquad
\begin{tikzcd}
A^k \arrow[r, "R"] & \{0,1\} \\
\Omega^k \arrow[u, "\alpha^k"] \arrow[ur, "\hat{R}"'] &
\end{tikzcd}$$

wherein $\alpha^k(x_1,\ldots,x_k) = (\alpha x_1,\ldots,\alpha x_k)$ and R is identified with its characteristic function.

Let A be a computable structure with coding α. A set $S \subset A^n$ is said to be (α-)computable or (α-)semicomputable accordingly as

$$\alpha^{-1}S = \{(x_1,\ldots,x_n) \in \Omega^n : (\alpha x_1,\ldots,\alpha x_n) \in S\}$$

is recursive or r.e.

5.2 LEMMA. *Every infinite computable structure A is isomorphic to a recursive number algebra R whose domain is the set of natural numbers ω and in which the r.e. subsets of ω correspond with the semicomputable subsets of A. The zero and successor on ω induce a computable enumeration of A; moreover, if A is a computable structure with enumeration then the isomorphism and algebra R can be chosen so as to allow zero and successor on ω to correspond to the given enumeration of A.*

The lemma is not difficult to formally prove; the reader may care to consult MAL'CEV [13].

PROOF OF THEOREM 5.1. One implication is Cook's Completeness Theorem 2.3. Let A be a computable structure with enumeration and assume HL(A) is complete; we show that for $\phi \in L$ and $S \in \mathcal{WP}$, the strongest postcondition $sp_A(\phi,S)$ is first-order definable over A. Let ϕ and S involve n variables and define

$$\mathrm{GRAPH}_A(S) = \{(a,b) \in A^n \times A^n : S(a) \text{ terminates in final state } b\}.$$

Then

$$b \in sp_A(\phi,S) \Longleftrightarrow \exists a \in A^n. \ [(a,b) \in \mathrm{GRAPH}_A(S) \ \& \ A \models \phi(a)]$$

and so it is sufficient to prove that $\mathrm{GRAPH}_A(S)$ is first-order definable. The following lemma we leave as an exercise:

5.3 LEMMA. *For any computable structure A and any $S \in \mathcal{WP}$, the set $\mathrm{GRAPH}_A(S)$ is semicomputable.*

Whence the theorem follows from this proposition.

5.4 PROPOSITION. *Let A be an algebraic with enumeration having signature Σ. Assume A is computable and that HL(A) is complete for $\mathcal{WP}$ over A. If $X \subset A^n$ is semicomputable then X is first-order definable over Σ.*

PROOF. By the normalising Lemma 5.2 we can assume A to be isomorphic to a recursive number algebra R with domain ω and whose enumeration is given by first element 0 and next operator, succ(x) = x+1. Moreover, the semi-computability of X can be identified with the recursive enumerability of $\alpha^{-1}X$ where $\alpha : R \to A$ is the isomorphism. Thus, technically, the matter reduces to proving that any recursively enumerable set $Y \subset \omega^n$ is first-order over the signature Σ in any numerical structure R which is a recursive expansion of Presburger Arithmetic P = $(\omega; 0, \mathrm{succ})$ and for which HL(R) is complete for $\mathcal{WP}$ over R.

By Matijacevič's Diophantine Theorem [14], it is clear that it is sufficient to prove that ordinary addition and multiplication on ω is first-order over Σ. Using the completeness of HL(R) for $\mathcal{WP}$ over R we shall show that

$$\underline{\text{plus}} = \{(x,y,z) \in \omega^3 : x+y = z\}$$

is first-order over Σ; we carry out the argument in detail and leave the case of

$$\underline{\text{mult}} = \{(x,y,z) \in \omega^3 : x \times y = z\}$$

as an exercise.

Consider the following composite program $S \equiv S_1;S_2 \in \mathcal{WP}$ having variables x,y,z_1,z_2,u and defined by these programs

```
S₁ ≡ z₁ := x; u := 0;
     while u ≠ y do u := succ(u); z₁ := succ(z₁) od;
     u := 0; z₂ := 0
S₂ ≡ z₂ := x; u := 0;
     while u ≠ y do u := succ(u); z₂ := succ(z₂) od;
```

Both programs add the values of x and y; but program S_1 tidies up the values of its auxiliary variables so that from state (a,b,c,d,e) it computes and terminates in state (a,b,a+b,0,0).

Clearly, $R \models \{\underline{\text{true}}\}S\{z_1=z_2\}$ and by the completeness of HL(R) we know that

$$\text{HL}(R) \vdash \{\underline{\text{true}}\}S\{z_1=z_2\}$$

By the Composition Rule, there must exist a first-order intermediate assertion δ such that

$$HL(R) \vdash \{\underline{true}\}S_1\{\delta\} \quad \text{and } HL(R) \vdash \{\delta\}S_2\{z_1=z_2\}$$

and so, by the Soundness Theorem,

$$R \models \{\underline{true}\}S_1\{\delta\} \text{ and } R \models \{\delta\}S_2\{z_1=z_2\}$$

Given the form of the final states of S_1 we know that

$$\text{for all } a,b \in \omega\ ,\ R \models \delta(a,b,a+b,0,0)$$

and, therefore, that

$$(a,b,c) \in \underline{plus} \Rightarrow R \models \delta(a,b,c,0,0)$$

Contrapositively assume $(a,b,c) \notin \underline{plus}$. Then $a + b \neq c$ implies that for any initial state (a,b,c,d,e) the program S_2 will terminate but $R \not\models z_1 = z_2$. The validity of the asserted program $\{\delta\}S_2\{z_1=z_2\}$ immediately implies

$$(a,b,c) \notin \underline{plus} \Rightarrow R \models \neg\ \delta(a,b,c,0,0)$$

and that $\underline{plus}$ is first order. Q.E.D.

REFERENCES

[1] APT, K.R., *Ten years of Hoare's logic, a survey* in F.V. JENSEN, B.H. MAYOH & K.K. MØLLER (eds.) *Proceedings from 5th Scandinavian Logic Symposium,* Aalborg University Press, Aalborg, 1979, 1-44.

[2] DE BAKKER, J.W., *Mathematical theory of program correctness,* Prentice-Hall International, London, 1980.

[3] BERGSTRA, J.A., J. TIURYN & J.V. TUCKER, *Floyd's principle, correctness theories and program equivalence,* Mathematical Centre, Department of Computer Science Research Report IW 145, Amsterdam, 1980.

[4] BERGSTRA, J.A. & J.V. TUCKER, *The field of algebraic numbers fails to possess even a nice sound, if relatively incomplete, Hoare-like logic for its while-programs,* Mathematical Centre, Department of Computer Science Research Report IW 136, Amsterdam, 1980.

[5] CHANG, C.C. & H.J. KEISLER, *Model theory,* North-Holland, Amsterdam, 1973.

[6] CLARKE, E.M., *Programming language constructs for which it is impossible to obtain good Hoare-like axioms,* J. Association Computing Machinery 26 (1979) 129-147.

[7] COOK, S.A., *Soundness and completeness of an axiom system for program verification,* SIAM J. Computing 7 (1978) 70-90.

[8] GREIBACH, S.A., *Theory of program structures: schemes, semantics, verification,* Springer-Verlag, Berlin, 1975.

[9] GREIF, I, & A.R. MEYER, *Specifying program language semantics: a tutorial and critique of a paper by Hoare and Lauer,* Proceedings Sixth ACM Symposium on Principles of Programming Languages, ACM, New York, 1979, 180-189.

[10] HOARE, C.A.R., *An axiomatic basis for computer programming,* Communications Association Computing Machinery 12 (1969) 576-580.

[11] LANGMAACK, H. & E.-R. OLDEROG, *Present-day Hoare-like systems for programming languages with procedures: power, limits and most likely extensions,* in: J.W. de Bakker & J. van Leeuwen (eds.) *Automata, languages and programming, Seventh Colloquium, Noordwijkerhout, July 1980,* Springer-Verlag, Berlin, 1980, 363-373.

[12] LIPTON, R.J., *A necessary and sufficient condition for the existence of Hoare logics,* 18th IEEE Symposium on Foundations of Computer Science, Providence, R.I., 1977, 1-6.

[13] MAL'CEV, A.I., *Constructive algebras, I.,* Russian Mathematical Surveys, 16 (1961) 77-129.

[14] MANIN, Y., *A course in mathematical logic,* Springer-Verlag, New York, 1977.

[15] RABIN, M.O., *Computable algebra, general theory and the theory of computable fields,* Transactions American Mathematical Society, 95 (1960) 341-360.

[16] TENNENBAUM, S., *Non-archimedean models for arithmetic,* American Math. Soc. Notices 6 (1959) 270.

[17] TUCKER, J.V., *Computing in algebraic systems,* in F.R. DRAKE & S.S, WAINER, (eds.) *Recursion theory, its generalisations and applications,* Cambridge University Press, Cambridge 1980.

[18] WAND, M., *A new incompleteness result for Hoare's system,* J. Association Computing Machinery, 25 (1978) 168-175.