JOURNAL OF COMPUTER AND SYSTEM SCIENCES 25, 267–284 (1982)

Expressiveness and the Completeness of Hoare's Logic*

J. A. BERGSTRA

Department of Computer Science, University of Leiden, Wassenaarseweg 80, 2300 RA Leiden, The Netherlands

AND

J. V. TUCKER

Department of Computer Studies, University of Leeds, Leeds LS2 9JT Britain

Received March 16, 1981; revised March 24, 1982

Three theorems are proven which reconsider the completeness of Hoare's logic for the partial correctness of **while**-programs equipped with a first-order assertion language. The results are about the expressiveness of the assertion language and the role 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 and (3) a computable structure with enumeration is expressive if and only if its Hoare logic is complete.

INTRODUCTION

By 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 [14]. In the present paper we shall consider this logic when it is equipped with a first-order assertion language L, a first-order data type specification T, and is set to analyse computation on natural number arithmetic(s) by members of the set WP of **while**-programs. Principally, we shall be interested in the completeness of the logic: the extent to which the true partial correctness formulae can be formally demonstrated in the system. We shall prove three theorems that comment on the relationship between the expressiveness of the assertion language L, the semantics of the axiomatic specification T, and the completeness of the logic.

Background

The starting point for any mathematical study of Hoare's ideas is the seminal paper Cook [11], where the various syntactic and semantic components associated

* The results in this paper were obtained while the second author was at the Mathematical Centre, Amsterdam, and an earlier edition of this paper is registered there as MC Report IW 149/80.

0022-0000/82/060267-18$02.00/0

with the system were carefully examined, and the soundness of the logic properly proved. Of particular interest to us is the role of the *data type specification* in Hoare's logic. This is a set T of assertions used in connection with the Rule of Consequence, and it is intended to formalize what information about data types is available to correctness proofs for the programs the datatypes support (cf. Hoare [1.4, Sect. 2]). For example, proofs for arithmetical programs are based, in practice, on some simple algebraic properties of addition and multiplication (and so on) and on the induction principle. Thus, *in pratice*, it is sufficient to take T to be Peano Arithmetic PA. From [11, Sect. 5], we know that *if T is valid for a structure A, then Hoare's logic* HL(T) *is sound for* WP *over A*. Up to the choice of program semantics for **while**-programs, Cook's analysis of Hoare's ideas is general and definitive. (For information on the issues involved in the choice of semantics consult Greif and Meyer [13].)

In [11], Cook also considered the completeness of Hoare's logic, but with apparently less satisfying theoretical results: *under the hypothesis that T is a complete specification for structure A and L is expressive for* WP *over A, then* HL(T) *is complete for* WP *over A*. In particular, if the standard arithmetic N of the natural numbers is fully specified by its first-order theory Th(N), so called *complete number theory*, then its Hoare logic HL(N) = HL(Th(N)) is complete for WP over N since L is expressive for 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 valuating 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 relevant to **while**-programs is contained in Wand [24] and our own [4] (on incompleteness); and in Lipton [16] and Olderog [19] (on expressiveness). We address the subject of completeness at the conclusion of our technical work (Section 6).

New Results

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's 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* WP *over A.*

Theorem 1 illuminates a certain change of status for Hoare's logic in the passage from the Soundness Theorem to the Completeness Theorem in [11]; it changes from a system of reasoning based purely upon a data type specification T into a system of reasoning about a fixed data type A which is appropriately specified by T. 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 MOD(T)

of all models of a specification T which is what one expects to see in a *true* converse to the Soundness Theorem. Mathematically, a Hoare logic HL(T) reasons about program behaviour over MOD(T).

Let us say that a Hoare logic HL(T) is *logically complete* if any asserted program which is valid on *all* models of the specification T is provable in HL(T). From Theorem 1 it follows that this kind of completeness is possible for complete number theory; next we prove a disappointing fact:

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

There are two objectives for data abstractions which can influence an application of Hoare's logic:

BASIC DICHOTOMY. *Given that some specification T is a necessary constituent of the proof system, one may think of the specification as*

(I) *an instrument to analyse computation on a particular structure A having the properties of T; or*

(II) *as an abstract characterization of a (possibly ill-defined) class of legitimate implementations having the properties of T.*

Theorem 1 sugests that Cook's analysis of completeness for alternative (I) is not exhaustive. The structures in the theorem, however, hardly qualify as interesting data-type semantics; they are not computable, for instance: see Tennenbaum [22].

In [14], Hoare intended his calculus HL(T) to be a system of reasoning about programs operating on *any legal implementation of the specification T*. If we interpret a legal implementation of T as simply a computable model of T, then we have a mathematically intermediate notion of completeness based on the class CMOD(T) of all computable model of T.

A Hoare logic HL(T) is *computably complete* if any asserted program which is valid over CMOD(T) is provable in HL(T).

In the case of arithmetic computable completeness is Cook's completeness for CMOD(PA) $= \{N\}$ and here expressiveness and completeness occur together. Actually this is a general phenomenon and, we have further reassurance of the usefulness of Cook's study of completeness for alternative (I):

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

We have greatly prolonged this introduction to accommodate observations on the semantic and syntactic roles 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 discuss completeness and prove the theorems announced. Obviously, we are assuming the reader is familiar with Hoare [14] and Cook [11], but little other knowledge is actually necessary. This paper is a close companion of [4] about natural structures which possess no complete Hoare-like logics for their **while**–programs; and both papers are sequels to [3] written with Tiuryn. Subsequent work relevant to completeness [6–8] and this are discussed in a final section of concluding remarks.

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 [14], Cook [11], 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 **true**, **false** 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 $\mathrm{WP} = \mathrm{WP}(\Sigma)$ of all **while**-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 T is a set of assertions of L, then the set of all formal theorems of T is denoted $\mathrm{Thm}(\mathsf{T})$; we write $\mathsf{T} \vdash \phi$ for $\phi \in \mathrm{Thm}(\mathsf{T})$. Such a set T of formulae is usually called a theory, but in the present context we prefer the more suggestive term *specification*. Two specifications T, T' are *logically equivalent* if $\mathrm{Thm}(\mathsf{T}) = \mathrm{Thm}(\mathsf{T}')$. A specification T is *complete* if given any assertion $\phi \in L$, either $\mathsf{T} \vdash \phi$ or $\mathsf{T} \vdash \neg\phi$. 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 T is denoted $\mathrm{Mod}(\mathsf{T})$; we write $\mathrm{Mod}(\mathsf{T}) \models \phi$ to mean that for every $A \in \mathrm{Mod}(T)$, $A \models \phi$. Gödel's completeness theorem says this about specifications:

$$\mathsf{T} \vdash \phi \qquad \text{if and only if } \mathrm{Mod}(\mathsf{T}) \models \phi.$$

For a proper discussion of these concepts the reader should consult Chang and Keisler [9].

For the semantics of 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 [11]; the graph-theoretic semantics in Greibach [12]; 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 language as WP, can hardly be problematical. For definiteness, we have in mind a naive operational semantics based upon appropriate A-register machines which yield straightforward definitions of a *state* in a WP computation and of the *length* of a WP computation [23]. Thus, if $S \in \mathrm{WP}$ 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 $S \in \mathrm{WP}$ involve variables $x = (x_1, \ldots, x_n)$. Then for each $l \in \omega$ one can effectively find a formula $\mathrm{COMP}_{S,l}(x, y)$ of L, wherein $y = (y_1, \ldots, y_n)$ are new variables, such that for any A and any $a, b \in A^n$, $A \models \mathrm{COMP}_{S,l}(a, b)$ 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 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 $S \in \mathrm{WP}$ 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 $a \in A$, $A \models p(a)$ 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* T if it is valid on *every* model of T; in symbols, $\mathsf{T} \models \{p\}\, S\, \{q\}$ or $\mathrm{Mod}(\mathsf{T}) \models \{p\}\, S\, \{q\}$.

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* T is the set

$$PC(\mathsf{T}) = \{\{p\}\, S\, \{q\} : \mathrm{Mod}(\mathsf{T}) \models \{p\}\, S\, \{q\}\}.$$

Clearly,

$$\mathrm{PC}(\mathsf{T}) = \bigcap_{A \in \mathrm{Mod}(\mathsf{T})} \mathrm{PC}(A).$$

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

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

1.2 LEMMA. $A \models \{p\}\, S\, \{q\} \Leftrightarrow \mathrm{sp}_A(p, S) \subset \{b \in A^n : A \models q(b)\}$.

2. Hoare's Logic

Hoare's logic for **while**-programs over Σ, with assertion language L and specification or oracle $\mathsf{T} \subset L$, has the following axioms and proof rules for manipulating asserted programs: let $S, S_1, S_2 \in \mathrm{WP}$; $p, q, p_1, q_1, r \in L$; $b \in L$, a quantifier-free formula.

1. *Assignment axiom scheme.* 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.*

$$\{p\}\, S_1\{r\},\ \{r\}\, S_2\{q\} / \{p\}\, S_1; S_2\{q\}.$$

3. *Conditional rule.*

$$\{p \wedge b\}\, S_1\{q\},\ \{p \wedge \neg b\}\, S_2\{q\} / \{p\}\ \textbf{if}\ b\ \textbf{then}\ S_1\ \textbf{else}\ S_2\ \textbf{fi}\{q\}.$$

4. *Iteration rule.*

$$\{p \wedge b\}\, S\{p\} / \{p\}\ \textbf{while}\ b\ \textbf{do}\ S\ \textbf{od}\ \{p \wedge \neg b\}.$$

5. *Consequence rule.*

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

And, in connection with 5:

6. *Oracle axiom.* Each member of $\mathrm{Thm}(\mathsf{T})$ is an axiom.

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

2.1 Basic Uniqueness Lemma. *For any consistent specifications* T *and* T' *which are logically equivalent we have that* $\mathrm{HL}(\mathsf{T}) = \mathrm{HL}(\mathsf{T}')$ *and* $\mathrm{PC}(\mathsf{T}) = \mathrm{PC}(\mathsf{T}')$.

Proof. The equality of Hoare logics over logically equivalent specifications is obvious. If $\mathrm{Thm}(\mathsf{T}) = \mathrm{Thm}(\mathsf{T}')$, then $\mathrm{Mod}(\mathsf{T}) = \mathrm{Mod}(\mathsf{T}')$, by the soundness of first-order logic. Therefore $\mathrm{PC}(\mathsf{T}) = \mathrm{PC}(\mathsf{T}')$.

The Corollary to Theorem 1 in Cook [11] is as follows:

2.2 Soundedness Theorem. *For any specification* T, $\mathrm{HL}(\mathsf{T}) \subset \mathrm{PC}(\mathsf{T})$.

The Hoare logic HL(T) is said to be *logically complete* if HL(T) = PC(T). As noted in the Introduction, Cook choose to consider the completeness of Hoare's logic relative to a fixed structure rather than its logical completeness; we state [11, Theorem 3]:

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

2.3 COOK'S COMPLETENESS THEOREM. *For any structure A, if* T *is a complete specification for A in the sense that* $\mathrm{Thm}(\mathsf{T}) = \mathrm{Th}(A)$, *and if L is expressive for* WP *over A, then* $\mathrm{HL}(\mathsf{T}) = \mathrm{PC}(A)$.

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

$$\mathrm{HL}(A) = \mathrm{HL}(\mathrm{Th}(A)) \subset \mathrm{PC}(\mathrm{Th}(A)) \subset \mathrm{PC}(A)$$

and Completeness Theorem 2.3 says that if L is expressive for WP over A, then $\mathrm{HL}(A) = \mathrm{PC}(A)$.

Let $N = (\omega; 0, x+1, x \dot{-} 1, +, \times, \leqslant)$ to be standard model of arithmetic; the Corollary to Theorem 3 in [11] says:

2.4 COROLLARY. $\mathrm{HL}(N) = \mathrm{PC}(N)$.

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 $\mathrm{Th}(A) = \mathrm{Th}(B)$.

3.1 UNIQUENESS LEMMA FOR STRUCTURES. *If $A \equiv B$, then* $\mathrm{HL}(A) = \mathrm{HL}(B)$ *and* $\mathrm{PC}(A) = \mathrm{PC}(B)$.

Proof. The equality of the Hoare logics over elementary equivalent structures follows from 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 $S \in \mathrm{WP}$ involves n program variables. Given S and $l \in \omega$ 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 $a \in A^n$,

$$S_l(a) = S(a), \qquad \text{if} \quad |S(a)| \leqslant l,$$
$$= \text{undefined}, \qquad \text{otherwise.}$$

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

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

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

$$\begin{aligned}
A \models \{p\}\, S\, \{q\} &\Leftrightarrow A \models \mathrm{SP}(p, S) \to q, \qquad \text{by Lemma 1.2} \\
&\Leftrightarrow A \models \left[\bigvee_{l \in \omega} \mathrm{SP}(p, S_l)\right] \to q \\
&\Leftrightarrow A \models \bigwedge_{l \in \omega} [\mathrm{SP}(p, S_l) \to q] \\
&\Leftrightarrow \text{for every } l \in \omega, \quad A \models \mathrm{SP}(p, S_l) \to q \\
&\Leftrightarrow \text{for every } l \in \omega, \quad B \models \mathrm{SP}(p, S_l) \to q \quad \text{since} \quad A \equiv B \\
&\Leftrightarrow B \models \bigwedge_{l \in \omega} [\mathrm{SP}(p, S_l) \to q] \\
&\Leftrightarrow B \models \{p\}\, S\, \{q\}. \qquad \text{Q.E.D.}
\end{aligned}$$

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

Proof. Assume $\mathrm{HL}(A) = \mathrm{PC}(A)$. By Lemma 3.1, $\mathrm{HL}(A) = \mathrm{HL}(B)$ and $\mathrm{PC}(A) = \mathrm{PC}(B)$ so $\mathrm{HL}(B) = \mathrm{PC}(B)$. Q.E.D.

3.4 COROLLARY. $A \equiv B$ *if and only if* $\mathrm{PC}(A) = \mathrm{PC}(B)$.

Here is the first theorem from the Introduction:

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

Proof. Any model A of complete number theory is elementary equivalent to the standard model N; thus, $\mathrm{HL}(A)$ is complete by Corollaries 3.3 and 2.4. We show L is not expressive for A.

Let S be the following arithmetic program,

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

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 nonstandard, then S diverges from initial state (a, b) for any $a \in A$.

Consider the set $\mathrm{sp}_A(\mathbf{true}, S)$. Inspecting its definition we find that for $(a, b) \in A^2$,

$$(a, b) \in \mathrm{sp}_A(\mathbf{true}, S) \Leftrightarrow a = b \quad \text{and} \quad a \text{ is standard.}$$

Thus, $X = \{a \in A: (a, a) \in \mathrm{sp}_A(\mathbf{true}, S)\}$ is precisely the set of all standard numbers in A; Because A is nonstandard $\neg X$ is nonempty. If $\mathrm{sp}_A(\mathbf{true}, S)$ were first-order definable, then X and $\neg X$ would be first order from the axioms of Peano Arithmetic, we could prove the existence of a least element of $\neg X$. But A has no smallest nonstandard element because each nonzero 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* T *be a consistent specification which is complete. Then for each* $A \in \mathrm{Mod}(\mathsf{T})$ *it is the case that* $\mathrm{HL}(\mathsf{T}) = \mathrm{HL}(A)$ *and* $\mathrm{PC}(\mathsf{T}) = \mathrm{PC}(A)$. *In Particular, the following three statements are equivalent*:

(1) $\mathrm{HL}(\mathsf{T}) = \mathrm{PC}(\mathsf{T})$.

(2) *For each* $A \in \mathrm{Mod}(\mathsf{T})$, $\mathrm{HL}(A) = \mathrm{PC}(A)$.

(3) *For some* $A \in \mathrm{Mod}(\mathsf{T})$, $\mathrm{HL}(A) = \mathrm{PC}(A)$.

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

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

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

The equivalences in the theorem are easy corollaries of the first conclusions.

Q.E.D.

From Cook's Completeness Theorem 2.3 we can deduce

4.2 THEOREM. *Let* T *be a consistent specification which is complete. Then if* Mod(T) *contains an element A for which L is expressive for* WP *over A, then* HL(T) *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* **n** *denote the numeral in L corresponding to $n \in \omega$,*

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

(see Schoenfield [21]).

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

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

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

and consider the asserted program

$$\{\neg(x)\}\ S\{\textbf{false}\}$$

First, we claim that $\mathrm{Mod}(T) \models \{\neg\phi(x)\}\ S\{\textbf{false}\}$. For if $M \in \mathrm{Mod}(T)$, $m \in M$ and $M \models \neg\phi(m)$, then m is a nonstandard 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

$$\mathrm{HL}(T) \vdash \{\neg\phi(x)\}\ S\{\textbf{false}\}.$$

We now *unpick* a formal proof of the asserted program in HL(T) and from its intermediate assertions put together a proof for $T \vdash \forall x \cdot \varphi(x)$. This unpicking procedure is based on the following lemma which fixes a sensible form for the proofs to be unpicked:

4.4 LEMMA. *If* $\mathrm{HL}(T) \vdash \{p\}\ S\{q\}$, *then there is a proof of* $\{p\}\ S\{q\}$ *in which applications of the rule of consequence occur at two types of position, namely,* (1) *immediately after an assignment axiom or* (2) *immediately after an application of the iteration rule.*

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

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

Now, because y is not free in $\varphi(x)$, statement (a) implies that

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

(As a matter of fact this point requires a formal argument of a kind that will be proved in Step III.)

Step II. Consider Ib. By the **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) $\mathrm{HL}(T) \vdash \{\Theta \wedge x \neq y\}\, y := y + 1\{\Theta\}$,

(c) $T \vdash \Theta \wedge x = y \to$ **false**.

And this latter statement we rewrite

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

Step III. Consider IIb. 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) $\mathrm{HL}(T) \vdash \{\gamma[\, y/y + 1\,]\}\, y := y + 1\{\gamma\}$,

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

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

4.5 LEMMA. $T \vdash \neg\phi(x) \to \forall y \cdot \Theta(x, y) \wedge x \neq y$.

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

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

$T \vdash [\forall y \cdot \Theta(x, y) \vee x \neq y] \to \forall y \cdot x \neq y$

$T \vdash [\forall y \cdot x \neq y] \to$ **false**

By transitivity of implication,

$$T \vdash \neg\phi(x) \to \textbf{false}$$
$$T \vdash \phi(x).$$

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

Proof of Lemma 4.5. 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: $T \vdash \neg\phi(x) \to [\Theta(x, \mathbf{0}) \wedge x \neq \mathbf{0}]$

$T \vdash \neg\phi(x) \to \delta(x, \mathbf{0})$	from I(c);
$T \vdash \delta(x, \mathbf{0}) \to \Theta(x, \mathbf{0})$	from II(a);
$T \neg\phi(x) \to \Theta(x, \mathbf{0})$	by transitivity.
$T \vdash \neg\phi(x) \to x \neq \mathbf{0}$	since $T \vdash \phi(\mathbf{0})$.

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 for the semantics of a specification 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 $\mathrm{CPC}(T)$ be the set of all asserted programs valid on all computable models of T. Then Theorem 4.1 allows us to reduce completeness considerations of $\mathrm{HL}(T)$ with respect to $\mathrm{CPC}(T)$ to the case of an individual structure: if T is complete and possesses a computable model, then for any $A \in \mathrm{Mod}(T)$, $\mathrm{PC}(A) = \mathrm{CPC}(T) = \mathrm{PC}(T)$. 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 **first** of A and an injective operator **next**: $A \to A$ such that $A = \{\mathbf{next}^n(\mathbf{first})\colon n \in \omega\}$. By a *structure with enumeration* we mean a structure with such an enumeration named in its signature. In this last section we shall prove

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

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 [20] and Mal'cev [17].

A structure A is *computable* if there exists a recursive subset Ω of the set of natural numbers ω and a surjection $\alpha\colon \Omega \to A$ such that (1) the relation $\equiv_\alpha$ defined on Ω by $n \equiv_\alpha m \Leftrightarrow \alpha n = \alpha m$ 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:

$$\begin{array}{ccc} A^k & \xrightarrow{\sigma} & A \\ {\scriptstyle\alpha}\uparrow & & \uparrow{\scriptstyle\alpha} \\ \Omega^k & \xrightarrow{\hat{\sigma}} & \Omega \end{array} \qquad \begin{array}{ccc} A^k & \xrightarrow{R} & \{0,1\} \\ {\scriptstyle\alpha^k}\uparrow & \nearrow_{\hat{R}} & \\ \Omega^k & & \end{array}.$$

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

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

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 [17]

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 $S \in$ WP, the strongest postcondition $\mathrm{sp}_A(\phi, S)$ 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

$$b \in \mathrm{sp}_A(\phi, S) \Leftrightarrow \exists a \in A^n \qquad [(a, b) \in \mathrm{GRAPH}_A(S) \,\&\, A \models \phi(a)]$$

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 $S \in$ 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* WP *over A. If $X \subset A^n$ is semiconputable, 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, $\mathrm{succ}(x) = x + 1$. Moreover, the semicomputability of X can be identified with the recursive enmerability 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 $\mathrm{P} = (\omega; 0, \mathrm{succ})$ and for which HL(R) is complete for WP over R.

By Matijacevič's diophantine theorem [18], 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 WP over R we shall show that

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

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

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

as an exercise.

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

$$S_1 \equiv z_1 := x; \quad u := 0;$$
$$\textbf{while } u \neq y \textbf{ do } u := \mathrm{succ}(u); \quad z_1 := \mathrm{succ}(z_1) \textbf{ od};$$
$$u := 0; \quad z_2 := 0$$
$$S_2 \equiv z_2 := x; \quad u := 0;$$
$$\textbf{while } u \neq y \textbf{ do } u := \mathrm{succ}(u); \quad z_2 := \mathrm{succ}(z_2) \textbf{ od};$$

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

$$\mathrm{HL}(R) \vdash \{\textbf{true}\}\ S\{z_1 = z_2\}.$$

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

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

and so, by the Soundness Theorem,

$$R \models \{\textbf{true}\}\ S_1\{\delta\} \qquad \text{and} \qquad R \models \{\delta\}\ S_2\{z_1 = z_2\}.$$

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

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

and, therefore, that

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

Contrapositively assume $(a, b, c) \notin \textbf{plus}$. 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

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

and that **plus** is first order. Q.E.D.

6. Concluding Remarks: What Is a Completeness Theorem?

The Basic Dichotomy stated in the Introduction emphasises two attitudes toward completeness theorems for Hoare's system. The first is associated with alternative (I).

which focuses on a particular structure A, and has prevailed since Cook [11]: attitude (I) sees a completeness theorem as a statement that there are enough axioms and rules in a logical system to accomplish a certain set of proofs. In particular, the role of expressiveness in Cook's completeness theorem is justified by the following informal, yet plausible, interpretation of theorem: *provided that there are no inadequacies in the assertion language to express the effects of* WP *computations on A, then the axioms and rules of inference of Hoare's logic are sufficient to prove all partial correctness assertions true of A.*

With precisely this kind of interpretation in mind, many proof systems have been developed for more complex programming languages for which completeness theorems have been proved after the fashion of Cook *viz.* using a completeness defined by the true partial correctness assertions on an expressive structure (see the survey Apt [1]). It is known, however, that for very rich programming languages expressiveness is not sufficient for completeness and there exist expressive structures for which no completeness theorem can be proved (Clarke [10]).

Thus Theorem 1 is to be interpreted as *the assertion language may be inadequate to deal with* WP*'s computational properties on A yet Hoare's logic may still prove all there is to prove for A.*

Coupled with the fact that there are many natural structures that are not expressive [4], especially among the two or more sorted structures [7], the results established do not complement attitude (I) and Cook-style completeness theorems. There remains the following, however:

Open Problem

Does there exist a computable structure A which is not expressive but for which $\mathrm{HL}(A) = \mathrm{PC}(A)$?

Alternative (II) of the Basic Dichotomy is part of the idea of using Hoare logic to axiomatically specify a program language: an idea discussed in Hoare [14] and later re-examined in Hoare and Wirth [25]. In analogy to the Gödel completeness theorem one might have expected that every specification T gave the rise to a logically complete Hoare logic $\mathrm{HL}(T)$. Theorem 2 rules this out even for the important special specification, Peano Arithmetic. In connection with Theorem 2 it can be mentioned that, as a proof system, HL(PA) behaves well and can be used to study the limits of formal verification in practice [5] and to obtain results such as the fact that incomplete specifications can have logically complete Hoare logics [6]. For completeness, however, alternative (II) receives little consolation from the idea of computable completeness and Theorem 3.

Alternative (II) finds natural a second attittude to completeness: attitude (II) sees a completeness theorem as a statement that the logical system characterizes the semantics in question. In particular, *if a logical system is incomplete with respect to a semantics, then the system is not talking about that semantics.* With this interpretation in mind, there is but one conclusion for the absence of general completeness

theorems for Hoare's logic: according to Hoare's logic the semantics of WP is *not* the conventional combinatorial semantics we adopted in Section 1. This idea is studied in depth in our paper on axiomatic semantics [8].

REFERENCES

1. K. R. APT, Ten years of Hoare's Logic: a survey—Part 1, *ACM Trans. Programming Languages and Systems* **3** (1981), 431–483.
2. J. W. DE BAKKER, Mathematical theory of program correctness, Prentice–Hall International, London, 1980.
3. J. A. BERGSTRA, J. TIURYN, AND J. V. TUCKER, Floyd's principle, correctness theories and program equivalence, *Theoret. Comput. Sci.* **17** (1982), 113–149.
4. J. A. BERGSTRA AND J. V. TUCKER, Some natural structures which fail to possess a sound and decidable Hoare-like logic for their while-programs, *Theoret. Comput. Sci.* **17** (1982), 303–315.
5. J. A. BERGSTRA AND J. V. TUCKER, "Hoare's Logic and Peano's Arithmetic," Mathematical Centre, *Theoret. Comp. Sci.*, to appear.
6. J. A. BERGSTRA AND J. V. TUCKER, "Two Theorems About the Completeness of Hoare's Logic," Inf. Proc. Letters, to appear.
7. J. A. BERGSTRA AND J. V. TUCKER, Hoare's Logic for Programming Languages with Two Data Types, in preparation.
8. J. A. BERGSTRA AND J. V. TUCKER, "The Axiomatic Semantics of While-Programs Based on Hoare's Logic," Circulated Notes, Aber Ogwr, April 1981; paper in preparation.
9. C. C. CHANG AND H. J. KEISLER, "Model Theory," North-Holland, Amsterdam, 1973.
10. E. M. CLARKE, Programming language constructs for which it is impossible to obtain good Hoare-like axioms, *J. Assoc. Comput. Mach.* **26** (1979), 129–147.
11. S. A. COOK, Soundness and completeness of an axiom system for program verification, *SIAM J. Comput.* **7** (1978) 70–90; Corrigendum, **10**(1981), p. 612.
12. S. A. GREIBACH, "Theory of Program Structures: Schemes, Semantics, Verification," Springer-Verlag, Berlin, 1975.
13. I. GREIF AND A. R. MEYER, Specifying the semantics of while-programs: a trivial and critique of a paper by Hoare and Lauer, *ACM Trans. Programming Languages and Systems* **3** (1981), 484–507.
14. C. A. R. HOARE, An axiomatic basis for computer programming, *Commun. ACM* **12** (1969), 576–580.
15. H. LANGMAACK AND E.-R. OLDEROG, Present-day Hoare-like systems for programming languages with procedures: power, limits and most likely extensions, *in*, "Automata, anguages and programming, Seventh Colloquium, Noordwijkerhout, July 1980" (J. W. de Bakker and J. van Leeuwen, Eds.) pp. 363–373, Springer-Verlag, Berlin, 1980.
16. R. J. LIPTON, A necessary and sufficient condition for the existence of Hoare logics, *in* "Proceedings, 18th IEEE Symposium on Foundations of Computer Science," 1–6, Providence, Providence, R. I., 1977.
17. A. I. MAL'CEV, Constructive algebras, I., *Russian Math. Surveys* **16** (1961), 77–129.
18. Y. MANIN, "A Course in Mathematical Logic," Springer-Verlag, New York, 1977.
19. E.-R. OLDEROG, "General Equivalence of Expressivity Definitions Using Stonger Postconditions, Respectively Weaker Preconditions," Christian-Albrechts Universität, Kiel, Institut für Informatik, Bericht 8007, 1980.
20. M. O. RABIN, Computable algebra, general theory and the theory of computable fields, *Trans. Amer. Math. Soc.* **95** (1960), 341–360.
21. J. R. SCHOENFIELD, "Mathematical Logic," Addison–Wesley, Reading, Mass., 1967.
22. S. TENNEBAUM, Non-Archimedean models for arithmetic, *Notices Amer. Math. Soc.* **6** (1959), 270.

23. J. V. TUCKER, Computing in algebraic systems, *in* "Recursion Theory, Its Generalisations and Applications," (F. R. Drake and S. S. Wainer, Eds.) Cambridge Univ. Press, 1980.
24. M. WAND, A new incompleteness result for Hoare's system, *J. Assoc. Comput. Mach.* **25** (1978), 168–175.
25. C. A. R. HOARE AND N. WIRTH, An axiomatic definition of the programming language PASCAL. *Acta Inform.* **2** (1973) 335–355.