Vol. 22 No. 3 情報処理学会論文誌 May 1981

内包論理に基づく逐次型プログラムの論理†

沢 村 一††

R. Montague は自然言語（英語）に現れるさまざまな種類の内包的な語法を一つの形式的体系の中に取り入れるために一種の高階の様相論理である内包論理を展開した．この内包論理はまた言語としてのプログラミング言語に現れる内包性の問題をも解決することができる．これまで様相論理に含まれる概念とプログラムについて議論するさいに起る概念との間には意味論的および統語論的な両方の観点から密接な対応が存在するという理由で様相論理がプログラムの論証に対して適用されてきたが，この能力のゆえに内包論理の一般的枠組はプログラムの論理を考えるさいに様相論理よりもより十分な表現能力を提供している．

本文ではプログラミング言語の内包性にのみ注目して最初に，内包性の問題を解くために必要な言語要素を付加された内包論理について述べ，次にこの論理を基礎として，Hoare 論理の関数的変形となっている内包的 Hoare 論理 (IHL) を形式化する．

内包的 Hoare 論理は単に Hoare 論理の表現力に富んだ別形としてだけではなく，これまで提案されてきたいろいろな算法論理とさらにはプログラミング言語と自然言語の双方に対する統一的な論理体系の可能な一つの形体であると考えられる．

1. まえがき

本稿の目的は R. Montague の内包論理 (intensional logic)[1] に基づいて，逐次型プログラムに対するプログラムの論理として内包的 Hoare 論理を形式化し，内包論理が自然言語の論理分析に対してだけではなくプログラムの論理の基礎としても表現能力に富んだ論理的枠組を与えることを示すことである．

これまで種々のそれぞれが有利な特徴をもつプログラムの論理が提案されてきた[2),3)]．その中で特に，Burstall[4], Schwarz[5], Ashcroft[6], Pratt[7], Kröger[8], Manna と Waldinger[9], Manna と Pnueli[10),11)] らはプログラムの論理の構築に相様論理，時制論理などの非古典論理の有用性を示している．

一方，Montague[1] は自然言語の論理分析を行う上で，述語論理や様相論理では対応する表現をもたなかった自然言語（英語）に現れる内包的表現を一つの論理体系の中に組み入れるために高階の様相論理である内包論理を形式化した．また，Janssen と Boas[12),13)] は言語としてのプログラミング言語にも自然言語の場合と類似した内包性の問題が代入文の意味に関して生じ得ることを示し，内包論理の論理的概念を用いることによって解決した．

これらの結果と Montague の内包論理がその内に様相，時制概念などを含んでいることから，Montague の内包論理は逐次型および並列型の双方のプログラムの論理の基礎となる論理として有効であり，さらにプログラミング言語と自然言語の共通のベースとなる意味論体系を与える可能性を暗示している[14]．本稿では，Janssen と Boas による代入文に関する内包的意味論を簡単なプログラミング言語の制御構造にまで拡張し，プログラムの検証システムとして知られている逐次型プログラムに対する Hoare の論理[15] を内包論理の上に形式化する．第2章では，言語における内包性の問題を議論し，第3章では，内包言語とその解釈，第4章では，内包的 Hoare 論理 (IHL) とプログラムの正当性の証明例について述べる．

2. 言語の内包性

自然言語における言語表現の指示の問題に関しては言語の意味論の分野で多くの議論がなされてきた．言葉の意味を考えるとき，それが何を指示しているかを明確に規定することは重要である．次のような例を取り上げてみよう．

温度は 30° である (1)

温度は上昇する (2)

30° は上昇する (3)

$x=x+1\{p:=x;\ x:=x+1\}p=x$ (4)

$1=a(j)\{a(i):=1\}a(i)=a(j)$ (5)

† A Logic of Sequential Programs Based on Montague's Intensional Logic by HAJIME SAWAMURA (Division of Information Engineering, Faculty of Engineering, Hokkaido University).

†† 北海道大学工学部情報工学専攻 現：富士通国際情報社会科学研究所

文(1)と(2)から文(3)を導く推論は等号をもつ述語論理では妥当なもののように思われるが，明らかに我我の直観には合わないものである．また(4)，(5)は代入文に関する Hoare の論理の意味規則，$P(t)\{x:=t\}P(x)$ の配列変数 a，ポインタ変数 p を含む代入文に適用した例である．(4)は矛盾した命題 $x=x+1$，(5)は不十分な命題 $1=a(j)(i=j\vee 1=a(j)$ となるべきである) を生成している．

これらの自然言語文，代入文の例から容易に見られるように，これらの奇妙な推論，不正確な述語の生成を引き起こした原因は一つの表現を構成している構成要素を外延性を仮定し単純にほかの表現で置き換えたことにある．すなわち，文(1)は表現の指示対象 (外延) の同一性のみを主張しているが，文(2)は単にその表現の意味としての指示対象について何かを主張しているのではなく，その表現の別の指示物 (内包) について何かを主張しているのである．したがって(3)のような結論を導くことは許されない．同様の分析が代入文の例(4)，(5)にも当てはまる．すなわち，例(4)の $p=x$ はポインタ変数 p の指示対象 (値) が x そのものであることを述べている．一方，代入文 $x:=x+1$ は単純変数 x の指示対象 (値) を x の値に1を加えたものに等しくする効果をもっている．したがって前者の x (内包) に後者の x (外延) を代入することは許されない．配列に関する例(5)については配列を関数と捉えることによって一階論理をベースにする Hoare 論理においてすでに解決されている[16),17)]*．しかしながら，本稿ではこれを一つの配列変数に関する指示の問題として認識しその指示物である関数を次章の内包論理の中に実現することにする．

これまでの議論から，言語表現の意味を説明する場合それを外延的論理学のようにそれが指示する世界内の諸事物，すなわち外延と呼ばれる対象あるいは対象の集合などに関係づけるだけでは十分でなく，その表現に対して内包と呼ばれる意味 (概念) をも考慮する必要があることがわかる．この目的のために次章では内包的表現が可能な内包言語とそれに対する可能的多世界意味論を導入する．このとき，第4章の内包的 Hoare 論理では，可能的多世界は計算機の内部状態，変数の外延とは可能世界を指定したときの変数の取る値，変数の内包とは可能な世界をその世界における変数の値 (外延) に写像する関数として捉えられる．

* また，配列は抽象データタイプの理論と関連して別の見方も可能である[18)]．

3. 内包言語とその意味論

3.1 内包言語

型 (type) をもつ言語を導入する．

[型] e (実体)，t (真理値)，s (状態あるいは可能な世界) を三つの異なった対象とする．型の集合とは次の三つを満たす TYPE の最小集合である，

(1) $e, t\in$TYPE,

(2) $\alpha, \beta\in$TYPE ならば $<\alpha, \beta>\in$TYPE,

(3) $\alpha\in$TYPE ならば $<s, \alpha>\in$TYPE.

[原始記号] 原始記号は次のものから成る，

(1) 変数記号：各型に対して可算個の変数記号
$$x_\alpha{}^1, x_\alpha{}^2, \cdots\cdots,$$

(2) 定数記号：各型に対して可算個の定数記号
$$x_\alpha{}^1, x_\alpha{}^2, \cdots\cdots,$$

(3) 特別の記号：$\neg, \wedge, \vee, \supset$ (論理記号)，$\forall$，$\exists$ (量記号)，$\wedge$ (内包記号)，$\vee$ (外延記号)，$+, -, \times, \div$ (関数記号)，$>, <$ (関係記号)，$=$ (等号記号)，λ (関数抽象化記号)，$(,), \{,\}, [,]$ (補助記号)，true, false (論理定数記号)/，$\rightarrow$.

混乱が生じない限り添字は省略する．また型 α の定数記号(変数記号)の集合を表わすのに CON_α (VAR_α) と書き，$\mathrm{CON}=\bigcup_{\alpha\in\mathrm{TYPE}}\mathrm{CON}_\alpha$, $\mathrm{VAR}=\bigcup_{\alpha\in\mathrm{TYPE}}\mathrm{VAR}_\alpha$ とする．

[有意味な式] 型 α の有意味な式 (meaningful expression) の集合 ME_α を次のように帰納的に定める，

(1) $\mathrm{CON}_\alpha\subset\mathrm{ME}_\alpha$

(2) $\mathrm{VAR}_\alpha\subset\mathrm{ME}_\alpha$

(3) $A, B\in\mathrm{ME}_\alpha$ ならば $A=B\in\mathrm{ME}_t$

(4) $A, B\in\mathrm{ME}_t$ ならば $A>B, A<B\in\mathrm{ME}_t$

(5) $A, B\in\mathrm{ME}_t$ ならば $+(A), -(A), (A)\cdot(B)\in\mathrm{ME}_t$，ここで，$\cdot\in\{+, -, \times, \div\}$,

(6) $A, B\in\mathrm{ME}_t$ ならば $\neg(A), (A)\circ(B)\in\mathrm{ME}_t$，ここで，$\circ\in\{\wedge, \vee, \supset\}$,

(7) $A\in\mathrm{ME}_t$, $z\in\mathrm{VAR}$ ならば $\exists z(A), \forall z(A)\in\mathrm{ME}_t$,

(8) $A\in\mathrm{ME}_\beta, z\in\mathrm{VAR}_\alpha$ ならば $\lambda z(A)\in\mathrm{ME}_{<\alpha,\beta>}$,

(9) $A\in\mathrm{ME}_{<\alpha,\beta>}, B\in\mathrm{ME}_\alpha$ ならば $A(B)\in\mathrm{ME}_\beta$,

(10) $A\in\mathrm{ME}_{<s,\alpha>}$ ならば $\vee A\in\mathrm{ME}_\alpha$,

(11) $A\in\mathrm{ME}_\alpha$ ならば $\wedge A\in\mathrm{ME}_{<s,\alpha>}$,

(12) $A, B\in\mathrm{ME}_\alpha, p\in\mathrm{ME}_t$

ならば $(p\rightarrow A, B)\in \mathrm{ME}_\alpha$,

(13) $A\in\mathrm{ME}_\alpha, x\in\mathrm{CON}_{<s,\beta>}, B\in\mathrm{ME}_\beta$
ならば $\{B/\vee x\}A\in\mathrm{ME}_\alpha$.

3.2 解　釈

3.1 の有意味な式は内包モデルで解釈される.

[フレーム] D, Sを非空な集合とする. DとSに基づくフレームとは次のような集合の指標づけられた族 $(D_\alpha)_{\alpha\in\mathrm{TYPE}}$ をいう,

(1) $D_e=D$,

(2) $D_t=\{T, F\}$

(3) $D_{<s,\alpha>}=D_\alpha{}^{S}=\{f\mid f: S\rightarrow D_\alpha\}$,

(4) $D_{<\alpha,\beta>}=D_\beta{}^{D_\alpha}=\{f\mid f: D_\alpha\rightarrow D_\beta\}$.

[モデル] DとSに基づくモデルとはシステム$M=(D_\alpha, m)_{\alpha\in\mathrm{TYPE}}$ をいう, ここで,

(1) $(D_\alpha)_{\alpha\in\mathrm{TYPE}}$ はDとSに基づくフレーム,

(2) m (意味関数) は $c\in\mathrm{CON}_\alpha$ に対して $m(c)\in D_{<s,\alpha>}$ を与える関数である.

[割り当て関数] モデル M 上の割り当て関数 a とは型 α の変数 z_α に対して $a(z_\alpha)\in D_\alpha$ なる変数の集合上で定義された関数である.

aを一つの割り当て関数とするとき, $a(z/d)$は次のような割り当て関数 a' を表すものとする,

$$a'(y)=\begin{cases} d, & y=z \text{ のとき},\\ a(y), & y\neq z \text{のとき}.\end{cases}$$

状態 (可能な世界) を計算機の内部状態と同一視するために状態の集合を次のように限定する,

$$S=\prod_{\alpha\in\mathrm{TYPE}}\Big(\prod_{\mathrm{CON}_{<s,\alpha>}} D_\alpha\Big).$$

関数記号 $+, -, \times, \div$, 関係記号 $>, <$ は通常の意味をもつものとし, さらに論理定数 true, false, 論理記号 $\neg, \wedge, \vee, \supset$, 量記号 $\exists, \forall$ の意味も通常のように解釈されるものとする. 内包論理に固有のものについてのみその解釈の定義を次に与える.

[有意味な式の解釈] 状態 s と割り当て関数 a に関して有意味な式 A_α のモデル M における解釈 $\bigvee_{s,a}{}^{M}(A_\alpha)$ は **3.1** の有意味な式の帰納的定義にそって次のように与えられる (M を省略して書く),

(1) $\bigvee_{s,a}(c_\alpha)=m(c_\alpha)(s), c_\alpha\in\mathrm{CON}_\alpha$,

(2) $\bigvee_{s,a}(z_\alpha)=a(z_\alpha), z_\alpha\in\mathrm{VAR}_\alpha$,

(3) $\bigvee_{s,a}(A=B)=\begin{cases} T, & \bigvee_{s,a}(A)=\bigvee_{s,a}(B) \text{ のとき}\\ F, & \text{それ以外のとき},\end{cases}$

(8) $\bigvee_{s,a}(\lambda z_\alpha(A_\beta))=D_\alpha$ 上の関数Fであって $d\in D_\alpha$ での値が $\bigvee_{s,a'}(A_\beta)$ に等しい関数, ここで, $a'=a(z_\alpha/d)$,

(9) $\bigvee_{s,a}(A_{<\alpha,\beta>}(B_\alpha))=\bigvee_{s,a}(A_{<\alpha,\beta>})(\bigvee_{s,a}(B_\alpha))$,

(10) $\bigvee_{s,a}(\vee A_{<s,\alpha>})=\bigvee_{s,a}(A_{<s,\alpha>})(s)$,

(11) $\bigvee_{s,a}(\wedge A_\alpha)=S$ 上の関数であって $t\in S$ での値が $\bigvee_{t,a}(A_\alpha)$ に等しい関数,

(12) $\bigvee_{s,a}((p_t\rightarrow A_\alpha, A_\beta))$
$$=\begin{cases}\bigvee_{s,a}(A_\alpha), \bigvee_{s,a}(p_t)=T \text{ のとき},\\ \bigvee_{s,a}(B_\alpha), \bigvee_{s,a}(p_t)=F \text{ のとき},\end{cases}$$

(13) $\bigvee_{s,a}(\{B_\beta/\vee x\}A_\alpha)=\bigvee_{t,a}(A_\alpha)$, ここで $t=<x\leftarrow\bigvee_{s,a}(B_\beta)>s$. これは型が $<s,\alpha>$ の任意の定数記号yに対して,
$$\bigvee_{t,a}(\vee y)=\begin{cases}\bigvee_{s,a}(\vee y), & y\neq x \text{ のとき},\\ \bigvee_{s,a}(B_\beta), & y=x \text{ のとき},\end{cases}$$
なる状態を表す.

[論理式] 型が t の有意味な式を論理式と呼ぶ. 論理式Aは $\bigvee_{s,a}{}^{M}(A)=T$ のときモデル M において状態 s と割り当て関数aによって満たされるといわれ, 記号で $\mathrm{M}, s, a\models A$ と書く. また, あらゆる s と aに対して, $\mathrm{M}, s, a\models A$ であれば論理式AはモデルMで真であるといわれ, 記号で $\mathrm{M}\models A$ と書く. さらに, あらゆるモデル M に対して, $\mathrm{M}\models A$ であれば論理式Aは妥当であるといわれ, 記号で $\models A$ と書く. 今後, 固定したモデルを考えるときは M を省略する.

次の諸定理は有意味な式の簡約のさいに用いられる.

定理 1 (Montague[1], Gallin[19]) 有意味な式 B のどんな自由変数も有意味な式Aの中の変数zに対してBを代入することによって束縛されることなく, すべての状態 s と t に対して $\bigvee_{s,a}(B)=\bigvee_{t,a}(B)$ ならば,

$$\models\lambda z(A)(B)=A^{z}_{B},$$

ここで A^{z}_{B} はAの中で自由に現れる zに Bを代入した結果を表す.

定理 2 (Montague[1], Gallin[19]) $\models\vee\wedge A=A$.

定理 3 (Janssen, Boas[12]) $x\in\mathrm{CON}_{<s,\alpha>}$, eを型がαの有意味な式とするとき, $\{e/\vee x\}$ に関して次の性質が成り立つ, ただし, $v\in\mathrm{VAR}$, $P, Q\in\mathrm{ME}_t$, $C\in\mathrm{CON}$, f_α, A, B を有意味な式とする,

(1) $\{e/\vee x\}\neg(P)=\neg\{e/\vee x\}P$,

(2) $\{e/\vee x\}(P\circ R)=\{e/\vee x\}P\circ\{e/\vee x\}R, \circ\in\{\wedge, \vee, \supset\}$,

(3) $\{e/\vee x\}Qv(P)=Qv(\{e/\vee x\}P)$, $Q\in\{\forall, \exists,$

λ}，ただし，vはeに自由に生起しないものとする（そうでなければ新しい変数でvを置き換える），

(4) $\{e/\vee x\}A(B)=\{e/\vee x\}A(\{e/\vee x\}B)$,

(5) $\{e/\vee x\}\wedge A=\wedge A$,

(6) $\{e/\vee x\}\{f/\vee x\}A=\{\{e/\vee x\}f/\vee x\}A$,

(7) $\{e/\vee x\}c=c$,

(8) $\{e/\vee x\}\vee x=e$,

(9) $\{e/\vee x\}\vee c=\vee c$，ただし，$c$と$x$は異なる．

定理1はその仮定から，式Bが変数記号であったり，$\wedge A$という形の式であるとき，および自由変数xが$\wedge$の作用域内には起こらないときλ変換が可能であることを主張している．定理2は式の内包の外延は式そのものであることを，定理3は$\{e/\vee x\}$が通常の代入操作とほとんど同じ性質をもっていることを示している．この演算子は次章のIHLの展開において，代入文の公理の中で用いられ，第2章で例示したような無差別な構文的代入操作を防ぐ効果をもつ．

内包論理は型の論理の上に構成されているので完全な公理系は存在しない．しかしながら，本稿の有意味な式の定義の(12)，(13)で与えられる構成要素を除く内包言語とそれに対する一般モデル (general model) に対しては完全であることがGallinによって示されている[19]．本稿の内包論理の一般モデルに対する完全性については未解決である．

4. 内包的 Hoare 論理 (IHL)

前章の内包言語はプログラミング言語に現れる種々の型をもつ変数とその意味を表現するのに十分な表現能力をもっている．この章では，さらに第3章の内包言語とその意味論を，次の節で与えるプログラミング言語の公理的意味論とプログラムの部分正当性を証明するための証明体系を同時に与えることができるHoare型の論理[15]を形式化するためにHoareの記法を導入し拡張する．このときプログラミング言語の意味論が第2章で議論したプログラミング言語に関する内包性の問題と共に，Montagueによって提唱された自然言語の論理分析の方法と全く同じプログラムに従って取り扱われていく．すなわちそれは三つの局面から成っている，

(1) プログラムの適格文が生成される (4.1節)，

(2) その適格文がHoareの記法と内包論理の式を用いて翻訳される (4.2, 4.3節)，

(3) その翻訳された式は可能的多世界を計算機の内部状態と同一視した内包論理学の可能的多世界意味論によって意味づけられる (4.3節)．

4.1 簡単なプログラミング言語

[記号]

(1) 定数記号：数値定数，論理定数 (true, false)，

(2) 変数記号：単純変数 $x, y, \cdots$，
配列変数 $a, b, \cdots$，
ポインタ変数 $p, q, \cdots$，

(3) 論理記号：$\neg, \wedge, \vee, \supset$，

(4) 算術・関係記号：$+, -, \times, \div, >, <$，

(5) 等号記号：$=$

(6) プログラム記号：:=, ;, if, then, else, fi, while, do, od, go, to, :, Null,

(7) ラベル記号：$l_1, l_2, \cdots$，

(8) 補助記号：(,), [,].

[算術式，ブール式] 算術式の集合Eおよびブール式の集合Fは第3章の有意味な式の帰納的な定義の(3)～(6)に対応して通常のように定義されるものとする．

[プログラム PROG]

(1) 空文：Null∈PROG，

(2) 代入文：$e, f\in E$, xが単純変数，aが配列変数，p, qがポインタ変数であるならば，$x:=f$, $a[e]:=f$, $p:=x$, $p:=q$, $p:=a[e]\in$PROG，

(3) 合成文：$S_1, S_2\in$PROG ならば $S_1; S_2\in$PROG，

(4) 条件文：$S, S_1, S_2\in$PROG, $p\in F$ ならば if p then S_1 else S_2 fi, if p then S fi$\in$PROG，

(5) くり返し文：$S\in$PROG, $p\in F$ ならば，while p do S od$\in$PROG，

(6) ジャンプ文：Lがラベルならば go to $L\in$PROG，

(7) ラベル付き文：$S\in$PROG, Lがラベルならば $L: S\in$PROG.

4.2 翻　訳

[記号の翻訳] 論理記号，算術記号，関係記号，等号記号，補助記号，論理定数 (true, false) は第3章の内包言語の対応する記号に翻訳する．数値定数は型がeの，単純変数は型が$\langle s, e\rangle$の内包言語の定数記号に翻訳し，配列変数，ポインタ変数はそれぞれ型が$\langle s, \langle e, e\rangle\rangle$, $\langle s, \langle s, e\rangle\rangle$の定数記号に翻訳する．翻訳された記号としては混乱が生じない限り同じ記号を用い

る.

[算術式，ブール式の翻訳] 算術式とブール式の翻訳はそれらの部分式から得られる．左辺が単純変数，配列変数である代入文の右辺の算術式とブール式に現れる単純変数 x，配列要素 $a[e]$，および左辺がポインタ変数である代入文の右辺の q はそれぞれ $\vee x, \vee a[e'], \vee q$ に，左辺がポインタ変数である代入文の右辺の x，$a[e]$ はそれぞれ x，$\wedge(\vee a[e'])$ に翻訳する．ここで e' は左辺が単純変数である代入文の右辺の算術式と同様の e の翻訳を表すものである．

以上の記号および算術式，ブール式に関する翻訳を行う関数を $'$ で表し翻訳関数と呼ぶ．

[プログラムの翻訳] プログラムの翻訳は次の節で Hoare の記法を用いて部分正当性の証明体系として与えられる*.

4.3 証明体系

[定義 1] $M_A=\{(s,t)|$ 状態 s でプログラム A が実行されるとき状態 t で停止する$\}$.

第3章の内包言語の有意味な式に次のものとその解釈を付加する，

(14) $P, Q \in ME_{\langle s,t\rangle}$，A がプログラムならば $P\{A\}Q \in ME_t$,

(14) $\bigvee\limits_{s,a}^{M}(P\{A\}Q)$

$$=\begin{cases} T, \bigvee\limits_{s,a}^{M}(\vee P)=T \text{ かつ } (s,t)\in M_A \text{ ならば} \\ \quad \bigvee\limits_{t,a}^{M}(\vee Q)=T \text{ が成り立つとき}, \\ F, \text{ それ以外のとき}. \end{cases}$$

[定義 2] 型が $\langle s,t\rangle$ の有意味な式を状態述語と呼ぶ.

[定義 3] $\vdash P\{A\}Q$ であるとき，プログラム A は入力状態述語 P と出力状態述語 Q（あるいは入出力仕様 $[P,Q]$）に関して（部分）正当であるという.

[証明体系] 以下の公理系の中で次のように記号を用いる，

P, Q, R ：状態述語,
S, S_1, S_2：プログラム,
p ：ブール式,
x ：単純変数あるいはポインタ変数,
a ：配列変数,
e, f ：算術式,
$'$ ：翻訳関数,
assertion (L)：ラベル L での状態述語（表明式）.

公理

割り当て公理：$\wedge(\{f'/\vee x\}\vee p)\{x:=f\}P$,
$\wedge(\{\lambda n(n=e'\to f',$
$\vee a[n])/\vee a\}\vee p)\{a[e]:=f\}P$

定理：$\vdash P$ であるような定理

推論規則

Consequence：

$$\frac{\vee P\supset\vee Q \quad Q\{S\}R}{P\{S\}R}, \quad \frac{P\{S\}Q \quad \vee Q\supset\vee R}{P\{S\}R}$$

Composition：$\dfrac{P\{S_1\}Q \quad Q\{S_2\}R}{P\{S_1;\ S_2\}R}$

Null：$\dfrac{\vee P\supset\vee R}{P\{\text{Null}\}R}$

Conditional：

$$\frac{\wedge(\vee P\wedge p')\{S_1\}Q \quad \wedge(\vee P\wedge\neg p')\{S_2\}Q}{P\{\text{if } p \text{ then } S_1 \text{ else } S_2 \text{ fi}\}Q},$$

$$\frac{\wedge(\vee P\wedge p')\{S\}Q \quad \vee P\wedge\neg p'\supset\vee Q}{P\{\text{if } p \text{ then } S \text{ fi}\}Q}$$

Label：$\dfrac{\text{assertion}\,(L)\{S\}P}{\text{assertion}\,(L)\{L:\ S\}P}$

Iteration：$\dfrac{\wedge(\vee P\wedge p')\{S\}P}{P\{\text{while } p \text{ do } S \text{ od}\}\wedge(\vee P\wedge\neg p')}$

Go to：$\dfrac{\vee P\supset\vee\text{assertion}\,(L)}{P\{\text{go to } L\}Q}$

IHL の基本的な論理式 $P\{S\}Q$ において P, Q は通常の Hoare 論理と異なり状態述語である．これはプログラムの仕様表現の真理値はプログラムの実行状態によって一般に変わるものであるので，そのことを IHL では陽に内包論理の一つの式で表すためである．A を型 t の論理式とするとき，$\wedge A$ は A の真理値が状態によって一般に異なることを示す関数である．したがって，状態述語 $\wedge A$ が与えられたとき状態 s での A の真理値（外延）を得るには関数 $\wedge A$ の状態 s を指定したときの値，すなわち式 $\vee\wedge A$ $(=A$，定理2参照$)$ から得られる.

Janssen と Boas の内包論理を用いる代入文の意味論は Dijkstra の述語変換子 (predicate transformer)[20] を内包論理の式で表現することによって与えられている[12),13)]（30ページの注参照）．たとえば，代入文 $x:=f$ の意味は $\lambda Q_{\langle s,t\rangle}\wedge[\{f'/\vee x\}\vee Q_{\langle s,t\rangle}]$ という型が $\langle\langle s,t\rangle,\langle s,t\rangle\rangle$ の述語変換子で与えられる (Dijkstra の wp の定義に相当する[20])．このとき，代入文 $x:=f$ の実行前の状態述語は実行後の状態述語を $\wedge P$ とすると，

$\lambda Q\wedge[\{f'/\vee x\}\vee Q](\wedge P)$
$=\wedge[\{f'/\vee x\}\vee\wedge P]$ (定理1による)

---

* プログラムの翻訳は Hoare の記法を導入することなく型が $\langle\langle s,t\rangle,\langle s,t\rangle\rangle$ で $\lambda P(A)$，ここで $P\in VAR_{\langle s,t\rangle}$, $A\in ME_{\langle s,t\rangle}$ という形式の内包言語の有意味な式を用いて直接的に与えることもできる[12)].

$=\wedge[\{f'/\vee x\}P]$ (定理2による)

となる．本稿では，Hoare 型の論理を構成するために，上記の形式の論理式を代入文の公理としている．

添字付き変数(配列変数)に対する代入文の公理の形式も配列変数を関数値変数ととらえることを除き全く同様である．このとらえ方はすでに Hoare, Wirth[16], Igarashi, London, Luckham[17] らに見られる．すなわち，代入文 $a[e]:=f$ は代入文 $a:=\lambda n(n=e\rightarrow f, a[n])$ の省略形であると見なされる．本稿では配列変数をこのような考え方に従って，関数値変数の自然な表現が可能である内包論理の枠組の中で取り扱っている．

IHL の推論規則は，プログラムの仕様表現が状態述語で与えられるので証明の段階で生ずる論理式が状態述語に外延演算子$\vee$を作用させてから作られなければならないという点を除き通常の Hoare 論理の形式に従っている．したがって，公理，推論規則の中に現れる状態述語記号を単に述語記号とし，外延，内包演算子，翻訳関数記号をすべて取り去るならば一階論理に基づく Hoare 論理に一致する (IHL は Hoare 論理の関数的変形である)．このことから，Hoare 論理[15]で証明可能であれば IHL でも証明可能であることがわかる．

4.4 IHL の妥当性

定理 4 (妥当性定理)　$\vdash P\{A\}Q$ ならば $\models P\{A\}Q$.

証明　はじめに，プログラムAの終了状態 $FS(A, s)$ をモデル M に相対的に定義する：

(1)　$FS(\text{Null}, s)=s$,

(2)　$x:=e$ が任意のタイプの代入文ならば，
$$FS(x:=e, s)=\langle x\leftarrow\underset{s,a}{\vee}(e')\rangle s,$$

(3)　$FS(S_1;\ S_2, s)=FS(S_2, FS(S_1, s))$,

(4)　FS (if p then S_1 else S_2 fi, s)
$$=\begin{cases}FS(S_1, s), & \underset{s,a}{\vee}(p')=T \text{ のとき},\\ FS(S_2, s), & \text{それ以外のとき},\end{cases}$$

(5)　FS (if p then S fi, s)
$$=\begin{cases}FS(S, s), & \underset{s,a}{\vee}(p')=T \text{ のとき},\\ s, & \text{それ以外のとき},\end{cases}$$

(6)　FS (while p do S od, s)
$$=\begin{cases}FS(\text{while } p \text{ do } S \text{ od}, FS(S, s)), & \\ \quad \underset{s,a}{\vee}(p')=T \text{ のとき}, & \\ s, & \text{それ以外のとき},\end{cases}$$

(7)　FS (go to $L, s)=s$,

(8)　$FS(L: S, s)=FS(S, s)$.

公理が妥当であること，くり返し文に関する推論規則が真であることを保存することのみを示す．ほかの推論規則も同様に示される．任意のモデルを固定し任意の状態 s と割り当て関数 a をとる．はじめに，任意のタイプの代入文の公理を考える．$s, a\models\{e'/\vee x\}\vee P$ と仮定すれば $\underset{s,a}{\vee}(\{e'/\vee x\}\vee P)=\underset{t,a}{\vee}(\vee P)$, ここで $t=\langle x\leftarrow\underset{s,a}{\vee}(e')\rangle s$. $(s, t)\in M_{x:=e}$ から，$t, a\models\vee P$, すなわち，$\underset{s,a}{\wedge}(\{e'/\vee x\}\vee P)\{x:=e\}P$ は妥当である．

次にくり返し文に関する推論規則を考える．$s, a\models p'$ でない (以後このことを $s, a\not\models p'$ と書き表わす) とき，この規則の上式は無内容的に成り立つ．このときその下式は終了状態の定義 (6) によって次の命題，

$s, a\models\vee P$ かつ $s, a\not\models p'$ ならば $s, a\models\vee P\wedge\neg p'$,

と同値になるので真である．$s, a\models p'$ であるとき，$s, a\models\wedge(\vee P\wedge p')\{S\}P$ と仮定する．すなわち，$(s, t)\in M_S$ であるような終了状態に対して，

$s, a\models\vee P\wedge p'$ ならば $t, a\models\vee P$　　(1)

を得る．推論規則の下式が成り立たないと仮定すると，$(s_0, s_n)\in M_{\text{while } p \text{ do } S \text{ od}}$ であるような初期状態 s_0 と終了状態 s_n，ある割り当て関数 b に対して，

$s_0, b\models\vee P$　　(2)

$s_n, b\not\models\vee P\wedge\neg p'$　　(3)

を得る．くり返し文が停止するならば引き続いた状態対の有限列 $(s_0, s_1), (s_1, s_2), \cdots, (s_{n-1}, s_n)$, ここで $(s_i, s_{i+1})\in M_S, 0\leq i\leq n-1$, が存在し，

$s_i, b\models p', 0\leq i\leq n-1$　　(4)

$s_n, b\not\models p'$　　(5)

を満たす．(2)と(4)を用いて(1)をくり返し適用するならば，

$s_n, b\models\vee P$　　(6)

を得る．しかしながら，(3)と(5)から $s_n, b\not\models\vee P$ となり，(6)に矛盾する．　　*Q. E. D.*

4.5 証 明 例

[例1]

プログラム：$i:=0$;
$j:=4$;
if random<0.5 then $p:=i$ else $p:=j$ fi

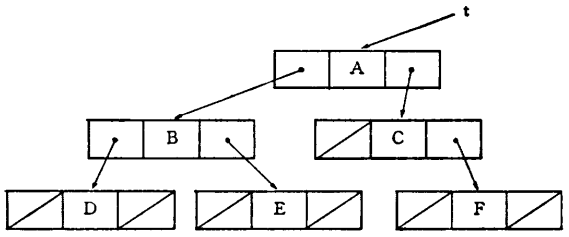
入出力仕様：[$\wedge$true, $\wedge(\vee P=i\wedge\vee i=0\vee\vee P=j\wedge\vee j=4)$].

証明：

1. $\wedge\text{true}\{i:=0\}\wedge(\vee i=0)$ (公理).
2. $\wedge(\vee i=0)\{j:=4\}\wedge(\vee i=0\wedge\vee j=4)$ (公理).
3. $\vee i=0\wedge\vee j=4\wedge\vee\text{random}\geq 0.5\supset\vee\text{random}\geq 0.5\wedge\vee j=4$ (定理).
4. $\vee i=0\wedge\vee j=4\wedge\vee\text{random}<0.5\supset\vee\text{random}<0.5\wedge\vee i=0$ (定理).

5. $\wedge(\vee i=0\wedge\vee j=4\wedge\vee$random $\geq 0.5)\{$NULL$\}\wedge(\vee$random$\geq 0.5\wedge\vee j=4)$ (3より).
6. $\wedge(\vee i=0\wedge\vee j=4\wedge\vee$random $<0,5)\{$NULL$\}\wedge(\vee$random$<0.5\wedge\vee i=0)$ (4より).
7. $\wedge(\vee i=0\wedge\vee j=4\wedge\vee$random $\geq 0.5\{p:=j\}\wedge(\vee$random$<0.5\wedge\vee p=i\wedge\vee i=0\vee\vee$random$\geq 0.5\wedge\vee p=j\wedge\vee j=4)$ (5より).
8. $\wedge(\vee i=0\wedge\vee j=4\wedge\vee$random $<0.5)\{p:=i\}\wedge(\vee$random$<0.5\wedge\vee p=i\wedge\vee i=0\vee\vee$random$\geq 0.5\wedge\vee p=j\wedge\vee j=4)$ (6より).
9. $\wedge$true$\{i:=0$; $j:=4$; if random <0.5 then $p:=i$ else $p:=j\}\wedge(\vee$random $<0.5\wedge\vee p=i\wedge\vee i=0\vee\vee$random $\geq 0.5\wedge\vee p=j\wedge\vee j=4)$ (7, 8より).
10. $\vee$random$<0.5\wedge\vee p=i\wedge\vee i=0\vee\vee$random $\geq 0.5\wedge\vee p=j\wedge\vee j=4\supset\vee p=i\wedge\vee i=0\vee\vee p=j\wedge\vee j=4$ (定理).
11. $\wedge$true$\{i:=0$; $j:=4$; if random <0.5 then $p:=i$ else $p:=j$ fi$\}\wedge(\vee p=i\wedge\vee i=0\vee\vee p=j\wedge\vee j=4)$ (9, 10より).

［例2］ プログラムは次のような二進木をトラバースしその末端節を数えるプログラムである,



木は Burstall[4] に従って nil かあるいは木の左の枝を指す l link, 右の枝を指す r link をもっているものとし, 木の末端節を次のように定義する,

tips $(\vee t)=1$ if $\vee t=$nil
$=$tips $(\vee l$ link$(\vee t))$
$+$tips $(\vee r$link $(\vee t))$ otherwise.

プログラム:
$p:=t$;
$s:=$empty;
$c:=0$;
Loop: if $p\neq$ nil
then $s:=s\cup\{p\}$; $p:=l$ link(p); go to Loop
else $c:=c+1$;
if $s=$empty then go to Exit fi;
$p:=$top; $s:=s-\{$top$\}$;
$p:=r$ link (p); go to Loop
fi;
Exit:

このプログラムにおいて用いられている記号は次のものを表す,
s: スタック変数*,
$\cup$: スタックプッシュオペレーション,
$-$: スタックポップオペレーション,
top: スタックのトップ要素,
empty: 空スタック,
nil: ポインタ定数でその値は nil であるとする.
入力状態述語: $\wedge$true.
出力状態述語: $\wedge(\text{tips}(\vee t)=\vee c\wedge\vee p=\text{nil}\wedge\vee s=\text{empty})$.
ループ述語**: $\wedge(\text{tips}(\vee t)=\vee c+\text{tips}(\vee p)+\sum_{u\in\vee s}\text{tips}(\vee r\ \text{link}(u)))$.

証明図の概形は次のようになる,

$$\dfrac{\dfrac{\dfrac{}{3}}{1}\quad\dfrac{\dfrac{\dfrac{6}{4}\quad\dfrac{7\quad 8}{5}}{2}}{\wedge\text{true}\{\text{プログラム}\}\wedge(\text{tips}(\vee t)=\vee c\wedge\vee p=\text{nil}\wedge\vee s=\text{empty})}$$

ここで数字は次の論理式を表す,

1. $\wedge$true$\{p:=t$; $s:=$empty; $c=0\}\wedge(\text{tips}(\vee t)=\vee c+\text{tips}(\vee p)+\sum_{u\in\vee s}\text{tips}(\vee r\ \text{link}(u)))$ (Composition).
2. $\wedge(\text{tips}(\vee t)=\vee c+\text{tips}(\vee p)+\sum_{u\in\vee s}\text{tips}(\vee r\ \text{link}(u)))\{$if $t\neq$nil then $s:=s\cup\{t\}$; $t:=l$ link(t); go to Loop else $c:=c+1$; if $s=$empty then go to Exit fi; $t:=$top; $s:=s-\{$top$\}$; $t:=r$ link(t); go to Loop fi; Exit:$\}\wedge(\text{tips}(\vee t)=\vee c\wedge\vee p=\text{nil}\wedge\vee s=\text{empty})$ (Composition, Label).
3. true$\supset\text{tips}(\vee t)=0+\text{tips}(\vee t)+\sum_{u\in\text{empty}}\text{tips}(\vee r\ \text{link}))$ (Composition, Null).

* スタックは配列を用いても表すことができるが簡単さのためにスタック変数の値は順序集合であるとしている.
** Hoare 論理のループ不変式に相当する.

4. $\wedge(\mathrm{tips}(\vee t)=\vee c+\mathrm{tips}(\vee p)$
$+\sum_{u\in\vee s}\mathrm{tips}(\vee r\ \mathrm{link}(u))\wedge\vee p\neq\mathrm{nil})\{s:=s\cup\{p\};$
$p:=l\ \mathrm{link}(p);$ go to Loop$\}\wedge(\mathrm{tips}(\vee p)=\vee c\wedge\vee$
$p=\mathrm{nil}\wedge\vee s=\mathrm{empty})$　　(Conditional).

5. $\wedge(\mathrm{tips}(\vee t)=\vee c+\mathrm{tips}(\vee p)$
$+\sum_{u\in\vee s}\mathrm{tips}(\vee r\,\mathrm{link}(u))\wedge\vee p=\mathrm{nil})\{c:=c+1;$ if s
$=\mathrm{empty}$ then go to Exit fi; $p:=\mathrm{top};$ $s:=s$
$-\{\mathrm{top}\};$ $p:=r\ \mathrm{link}(p);$ go to Loop$\}\wedge(\mathrm{tips}(\vee t)$
$=\vee c\wedge\vee p=\mathrm{nil}\wedge\vee s=\mathrm{empty})$　(Conditional).

6. $\mathrm{tips}(\vee t)=\vee c+\mathrm{tips}(\vee p)$
$+\sum_{u\in\vee s}\mathrm{tips}(\vee r\ \mathrm{link}(u))\wedge\vee p\neq\mathrm{nil}\supset\mathrm{tips}(\vee t)$
$=\vee c+\mathrm{tips}(\vee l\,\mathrm{link}(\vee p))+\sum_{u\in\vee s\cup\{\vee p\}}\mathrm{tips}(\vee r\,\mathrm{link}(u))$
(Go to, Composition, Assignment).

7. $\mathrm{tips}(\vee t)=\vee c-1+\mathrm{tips}(\vee p)$
$+\sum_{u\in\vee s}\mathrm{tips}(\vee r\ \mathrm{link}(u))\wedge\vee p=\mathrm{nil}\supset(\vee s$
$=\mathrm{empty}\supset\mathrm{tips}(\vee t)=\vee c\wedge\vee p=\mathrm{nil}\wedge\vee s$
$=\mathrm{empty})$
(Conditional, Composition, Go to, Assignment).

8. $\mathrm{tips}(\vee t)=\vee c-1+\mathrm{tips}(\vee p)$
$+\sum_{u\in\vee s}\mathrm{tips}(\vee r\ \mathrm{link}(u))\wedge\vee p=\mathrm{nil}\supset(\vee s\neq\mathrm{empty}$
$\supset\mathrm{tips}(\vee t)=\vee c+\mathrm{tips}(\vee r\ \mathrm{link}(\mathrm{top}))$
$+\sum_{u\in\vee s-\{\mathrm{top}\}}\mathrm{tips}(\vee r\ \mathrm{link}(u)))$
(Go to, Composition, Assignment, Conditional).

証明図の最も上に現れている論理式3, 6, 7, 8が真であることを見ることは容易である．したがって上のプログラムは与えられた入出力仕様に関して正当であることが証明された*.

5. むすび

IHL 構成の論理学的原理は Janssen と Boas[12),13)] による代入文の内包的意味論のそれであるが本論文の主要な寄与は次の2点にまとめられる．

1) Hoare の論理を，いろいろなタイプの変数の参照（指示）の問題に重点を置きながら，内包論理の枠組の中で形式化したこと，

2) Montague の内包論理が自然言語のほかにプログラムの論理の形式化においても有効な論理的基礎を与えることを明らかにしたこと．

プログラムの検証能力という観点から IHL を見るならば，もちろんプログラムの性質記述においては内包論理は優れているものの（単純，配列，ポインタ変数の指示の扱いとそれらを含むプログラムの仕様表現が内包論理の型の階層と内包，外延演算子を用いることによって直接的に与えられるという点で），証明能力（方法）という点では帰納的表明法に基づく従来の Hoare 論理と同等なものとなっているといえよう．

他方，本論文では内包論理の有効性を逐次型のプログラムに限って議論してきたのであるが，並列型のプログラムの証明には時制の概念と逐次型プログラムに対する方法とは異なった証明方法が要求されることが示されている[10)]．本来の Montague の内包論理には必要な時制演算子が含まれているがゆえに，並列型プログラムに対して内包論理は逐次型プログラムに対して有効であったプログラムの仕様法に加えて，内包論理に特徴的な証明法を提供することが期待できると考えられる．

謝辞　ご討論いただいた北大工学部前田隆助手，桃内佳雄助手，宮本衛市助教授に感謝する．また，日頃ご指導いただく富士通国際情報社会科学研究所北川敏男所長に感謝する．

* この証明は London[21)] による証明の関数的変形 (functional variant) になっている．

参考文献

1) Montague, R.: The proper treatment of quantification in ordinary English, in Formal philosophy-selected papers of Montague, R. Yale University Press, edited by Tomason, R. H. p. 369 (1977).
2) 沢村　一：算法論理，情報処理，Vol. 20, No. 8, pp. 725-734 (1979).
3) 沢村，前田，桃内：Algorithmic Logic の比較研究，電子通信学会，オートマトンと言語研究会，AL 78-27 (1978).
4) Burstall, R. M.: Program proving as hand simulation with a little induction, IFIP 74, pp. 308-312 (1974).
5) Schwarz, J.: Event based reasoning-A system for proving correct termination of programs, in Automata, Languages and Programming, edited by Michaelson, S. and Milner, R. Edinburgh Univ. Press, pp. 131-146 (1976).
6) Ashcroft, E. A. and Wadge, W. W.: Lucid-A formal system for writing and proving programs, SIAM J. on Computing, Vol. 5, No. 3, pp. 336-354 (1976).
7) Pratt, V. R.: Semantical considerations on Floyd-Hoare logic, 17th IEEE Symp. on Found. of Comp. Sci., pp. 109-121 (1976).
8) Kröger, F.: LAR: A logic of algorithmic reasoning, Acta Informatica, Vol. 8, Fasc. 3,

pp. 243-266 (1977).
9) Manna, Z. and Waldinger, R.: Is 'Sometime' sometimes better than 'Alwayes'?: Intermittent assertions in proving program correctness, CACM, Vol. 21, No. 2, pp. 159-172 (1978).
10) Pnueli, A.: The temporal semantics of concurrent programs, Lect. Notes, in Comp. Sci. 70, Springer-Verlag pp. 1-20 (1979).
11) Manna, Z. and Pnueli, A.: The modal logic of programs, Lect. Notes in Comp. Sci. 71, ALP, pp. 385-409, Springer-Verlag (1979).
12) Janssen, T. M. W. and Boas, P. van Emde: On the proper treatment of referencing, dereferencing and assignment, Lect. Notes in Comp. Sci. 52, pp. 282-300, Springer-Verlag (1977).
13) Janssen, T. M. W. and Boas, P. van Emde: The expressive power of intensional logic in the semantics of programming languages, Lect. Notes in Comp. Sci. 53, pp. 303-311 (1977).
14) 淵 一博：自然言語と計算機 —その内的な関連を探る，計算機による日本語談話行動の総合モデル化，科研費昭和53年度研究報告書 (1979).
15) Hoare, C. A. R.: An axiomatic basis for computer programming, CACM. Vol. 12, No. 10, pp. 576-580, 583 (1969).
16) Hoare, C. A. R. and Wirth, N.: An axiomatic definition of the programming language PASCAL, Acta Informatica, Vol. 2, pp. 335-355 (1973).
17) Igarashi, S., London, R. L. and Luckham, D. C.: Automatic program verification I: A logical basis and its implementation, Acta Informatica, Vol. 4, pp. 145-182 (1975).
18) Nakajima, R., Nakahara, H. and Honda, M.: Hierarchical program specification and verification—a Many—sorted logical approach—, Acta Informatica, Vol. 14, Fasc. 2, pp. 135-155 (1980).
19) Gallin, D.: Intensional and higher-order modal logic with applications to Montague semantics, North-Holland mathematics studies 19, p. 148 (1975).
20) Dijkstra, E. W.: A discipline of programming, Prentice-Hall, Inc., p. 217 (1976).
21) London, R. L.: Perspectives on program verification, in Current trends in programming methodology, Vol. 2, Program verification, edited by Yeh, R. T. pp. 151-172 (1977).

(昭和54年11月20日受付)
(昭和55年11月20日採録)