프로그램의 상태

표시적 의미

이상적인 컴퓨터의 상태(State) 변화의 관점에서 정의될 수 있다.

연산의미론과 표기의미론의 차이

연산의미론

이상적인 컴퓨터의 상태변화의 관점에서 정의됨

기계의 상태를 사용

상태변화가 프로그래밍언어로 코딩된 알고리즘에 의해 정의됨

표기의미론

연산의미론과 거의 유사함
하지만 더 단순하며 언어프로그램의 모든 변수들의 값의 관점에서 정의됨.

프로그램의 상태를 사용

엄격한 수학함수에 의해 정의됨.

표기의미론에

상태 s가 순서쌍들의 집합으로 표현될때

$s = \{\langle i_1, V_1\rangle, \langle i_2, V_2\rangle, \ldots, \langle i_n, V_n\rangle\}$

변수의 이름 / 변수의 현재값, undef를 가질 수 있다.

undef : 현재 정의되어 있지 않음

→ Var Map

VARMAP이 변수 이름과 프로그램 상태를 갖는 함수라고 할때,

$VARMAP(i_j, s) \approx V_j$

프로그램과 프로그램 구조에 대한 대부분의 의미사상 함수는 상태로부터 상태로 사상한다.

⇒ 이러한 상태변화는 프로그램과 프로그램 구조의 의미를 정의하는데 사용된다.

따라서 언어 구조(식등)는 상태가 아닌 값으로 사상되기도 함.

식

- 대화용의 프로그래밍언어에서 기본적

가정	1. 어떠한 복작용도 갖지 않는다. 2. 매우 단순한 식만 다룬다. 3. 유일한 연산자는 + 와 * 4. 모든 이항 연산자. 5. 피연산자는 스칼라 정수변수 또는 정수 리터럴 6. 괄호 없음. n. 식의 값은 정수
BNF	<expr> → <dec_num> | <Var> | <binary_expr> <binary_expr> → <left_expr><operator><right_expr> <left_expr> → <dec_num> | <Var> <right_expr> → <dec_num> | <Var> <operator> → + | *
발생가능한오류	정의되지 않은 값을 갖는 변수 ⇒ 이런 오류는 여러개에 공통적 error를 오류값이라할때, 의미정의역 ⇒ ZU {error}
표기법	△= ; 수학 함수를 정의 ⇒ ; implication (연관) symbol 피연산자의 형식을 이라 연관된 case 나 switch라 연결

도움: 노드의 자식노드들을 참조

주어진 식 E와 상태 S에대한 요구된 사상함수

$M_e(\langle expr\rangle, s)\ \triangle =$
Case <expr> of
<dec_num> ⇒ Mdec(<dec_num>, s)
<var> ⇒ if VarMAP(<var>, s) == undef
then error
else VarMap(<var>, s)

<binary_expr> ⇒
if (M_e(<binary_expr>.<left_expr>, s) == Undef
OR M_e(<binary_expr>.<right_expr>, s) == Undef)
then error
else if (<binary_expr>.<operator> = '+')
then M_e(<binary_expr>.<left_expr>, s) +
M_e(<binary_expr>.<right_expr>, s)
else M_e(<binary_expr>.<left_expr>, s) *
M_e(<binary_expr>.<right_expr>, s)

배정문

배정문

1. 식으로부터 평가
2. 그 식의 값을 목적변수에 설정

이 경우 2 의 목적변수는 상태로부터 상태로 사상

배정문의

$Ma(X = E, s) \triangle =$
  if $Me(E, s) == error$
    then error
  else $s' = [\langle i_1, v_1' \rangle, \cdots \langle i_n, v_n' \rangle]$ 에서 '
    For $j = 1, 2, \cdots, n$
      if $i_j == X$
        then $v_j' = Me(E, s)$
        else $v_j' = VARMAP(i_j, s)$

값이 아닌 위치에 대한 변수

논리 사건-검사 루프

가정 | M_{sl} : 문장리스트라 상태 ⇒ 상태

M_b : 불리안 식 ⇒ 불리안 값(or Err)

예제 | M_l = (while B do L, s) Δ=

if M_b(B, s) == undef
then error
else if M_b(B, s) == false
then s
else if M_{sl}(L, s) == error
then error
else M_l(while B do L, M_{sl}(L, s))

실행중요족가 없을때
루프의 의미 == 지정된 횟수만큼 실행한 후의 프로그램변수들의 값

루프는 반복형태 ⇒ 재귀형태로변환

재귀제어
- 논리를 재귀(상태 사상함수에 의해 수학적으로 정의됨
- 반복된다는 수학적 엄격성으로 기술하기가 더 수월

중요한점
실제 프로그램의 루프처럼 비종료성(nontermination) 때문에
아무것도 계산하지 않을수 있다.

평가

객체와 함수들 ⇒ 프로그래밍 언어의 또 다른 구문요소에 대해 정의될 수 있음.

주어진 언어에 대해 만족할 시스템이 정의되어 있다면
⇒ 이 시스템은 그 언어로 작성된 전체 프로그램의 의미를 결정하는데 사용할 수 있다.

이는 매우 엄격한 방식으로 프로그래밍을 고려하는 프레임워크를 제공

형식 의미론은 언어 설계에 도구로서 사용될 수 있음

1. 다른 설계가 고려될 수 있다는 것을 알려줄 수 있다.
2. 형식적 기술은 언어를 간명하게 기술하는 데 매우 효율적 방법을 제공