17 Axiomatic Semantics

Monday, April 21, 2025 12:28 PM

- Hoare Logic.

`69 C.A.R Hoare
"Tony"

What? A Formal system of logic Rules.
to reason rigorously about programs.

Formal = Mathematic.

Objectives :-
- Give Mathematical Meaning to Programs
- Prove that a program has certain properties
- Prove correctness of a program

Dijkstra, N. Wirth, knuth,, Lamport agree.
Program is a mathematical object.

REVIEW OF MATHEMARTICAL LOGIC:

Logic: The Mathematics of correct reasoning
- "true" or "false."
- Evaluate formulas to know whether they are true or false.

- Propositional Logic (boolean Logic)
  - Propositions

p q r s

Propositions are assigned true/false.

p = true
q = false

- Logical Connectives. $\wedge$ $\vee$ $\neg$ $\rightarrow$

what they mean is stipulated by Truth tables.

p	¬p
T	F
F	T

p	q	p ∧ q
T	T	T
T	F	F

p	q	p → q
T	T	T
T	F	F

r	¬r
T	F
F	T

p	q	p ∧ q
T	T	T
T	F	F
F	T	F
F	F	F

p	q	p → q
T	T	T
T	F	F
F	T	T
F	F	T

- Propositional Formulas

To be evaluated:

p	T
q	F
r	T

$((p \vee q) \rightarrow \neg r) \wedge (r \vee \neg q)$

T F T T

F T

F

F

- Inference Rules.

Rules of pre-evaluated formulas.

$\frac{S_1 \; S_2 \; S_3 \ldots S_k}{C}$ ← premises ← Conclusion.

"If all of S_i are True then C is true."

e.g. $\frac{p \wedge q}{p}$ $\frac{p \rightarrow q \quad p}{q}$ $\frac{}{p \vee \neg p.}$

Limitations:

e.g. "All men are mortal
Aristotle is a man
therefore Aristotle is mortal"

- Predicate Logic
  - Objects: a b aristotle; 1 7 garfield.
  - Variables over objects: X Y Z
  - Predicates.
    - represent properties or relationships between objects
    - have an "arity": a fixed number of arguments.

red(a) man(aristotle) friend(aristote, garfield)
odd(7)
lessthan(1, 7) has_lives(garfield, 7)
odd(X) friend(X, garfield)

lessthan(1, 7) has_lives(garfield, 7)

odd(X) friend(Y, garfield)

Ground predicates: , those without variables., can be assigned true / false.

lessthan(1,7) true man(aristotle) true

lessthan(7,1) false man(garfield) false.

- Quantifiers.

  $\forall X(p(x))$ true if p is true for all objects

  $\exists X(p(X))$ true if p is true for at least one object.

- Logical Connectives

  $\wedge \quad \vee \quad \neg \quad \rightarrow$

- Predicate Formulas

  Objects, variables, predicates, quantifiers, connectives

eg objects $\{a, b\}$ predicates $\{p, q, r\}$

$p(a) = T$
$p(b) = T$
$q(a) = F$
$q(b) = T$
$r(a) = F$
$r(b) = F$

$$(\forall X\, p(x) \vee \exists Y\, q(Y)) \rightarrow (\neg(\forall X\, r(x)))$$

T T F

T T

T

- Rules of inference

$$\frac{p(a) \quad p(b)}{p(a) \vee p(b)} \qquad \frac{\forall x\, p(x)}{p(c)} \qquad \frac{p(c)}{\exists X\, p(x)}$$

$$\frac{\forall X(p(x) \rightarrow q(x)), \quad p(c)}{q(c)}$$ "modus ponnens"

- The power of predicate logic

1) all men are mortal $\forall X(man(x) \rightarrow mortal(x))$
2) aristotle is a man $man(aristotle)$
   Therefore (modus ponnens.)
3) aristotle is mortal $mortal(aristotle)$

- Hoare Logic

  A logic for programs.

  - Objects.

    Program Variables and their assignments.

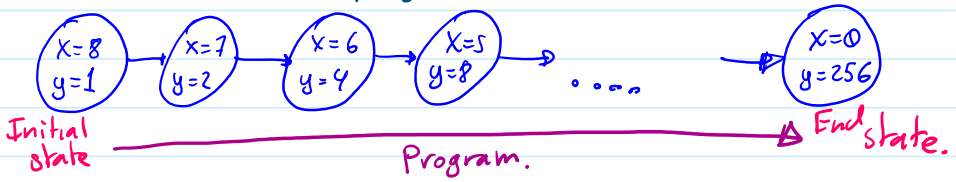
  - State

    An assignment of values to variables

  - Program Execution

    A sequence of states

    A sequence of transitions from an initial state to a final state.



  - Predicate Formulas.

    - To talk about states

      Not to assign them true or False.

    - Represent the set of states in which the formula is true.

lessthan(y, 10)

Objects { x, y, $\mathbb{N}_0$ }

even(x) = { x=2 y=7 , x=4 y=0 , x=8 y=1 , ... }

lessthan(y,10) = { x=10 y=5 , x=9 y=3 , x=8 y=0 , ... }

y<10

  - Hoare Triples :- Formulas in Hoare Logic.

    $\{P\}\,C\,\{Q\}$

    where P, Q Predicate Formulas

    C "command" a piece of code.

    P "precondition"

    Q "Postcondition"

When is $\{P\}\,C\,\{Q\}$ = True?

If executing C in any state in {P} leads you to a state in {Q}

If executing C in any state in {P}
leads you to a state in {Q}

How is this useful

C :- your program
{P} :- a formula :- spec of the input.
{Q} :- a formula :- spec of the output.

If you prove {P}C{Q} is true,
you proved that the program **correct**

We CAN use Mathematical Logic to prove correctness of our programs.

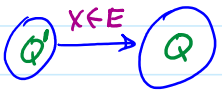
- Rules of Inference.

  cover { • assignment
  • conditionals
  • loops

- Axiom of Assignment

$$\{Q_{X \to E}\}\ X \leftarrow E\ \{Q\}$$

$Q_{X \to E}$: Replace every occurance of X in Q by E

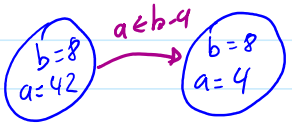


E.g

$$\{\ ?\ \}\ a \leftarrow b-4\ \{a > \overset{Q}{0}\}$$

$$\{b-4 > 0\}$$

$$\{\underbrace{b > 4}_{P}\}\ a \leftarrow b-4\ \{\underbrace{a > 0}_{Q}\} = \text{True}$$



Eg.

$$\{?\}\ a \leftarrow \overbrace{2+(b-2)}^{E}\ \{0 \leq \overset{Q}{a} \leq 10\}$$

$$\{0 \leq 2+(b-2) \leq 10\}$$

$$\{0 \leq 2b-4 \leq 10\}$$

$$\{4 \leq 2b \leq 14\}$$

$$\{2 \leq b \leq 7\}\ a \leftarrow 2+(b-2)\ \{0 \leq a \leq 10\} = \text{True}.$$

- Rule of Composition

$$\frac{\{P\}\ C_1\ \{R\}\ ,\ \{R\}\ C_2\ \{Q\}}{\{P\}\ C_1 ; C_2\ \{Q\}}$$

$$\frac{\{P\}\,C_1\,\{R\}\quad,\quad\{R\}\,C_2\,\{Q\}}{\{P\}\,C_1 ; C_2\,\{Q\}}$$

E.g.

$\{?\}\ a \leftarrow 3*b-1 ;\ \underset{C_2}{b \leftarrow 4*a-22}\ \{\underset{Q}{b > 10}\}$

$\{4\cdot a - 22 > 10\}\ b \leftarrow 4*a-22\ \{b > 10\}$

$\{4\cdot a > 32\}$

$\{\underset{R}{a > 8}\}\ b \leftarrow 4*a-22\ \{\underset{Q}{b > 10}\} = \text{true}$

$\{3\cdot b - 1 > 8\}\ a \leftarrow 3*b-1\ \{a > 8\}$

$\{3\cdot b > 9\}$

$\{\underset{P}{b > 3}\}\ a \leftarrow 3*b-1\ \{\underset{R}{a > 8}\} = \text{true}$

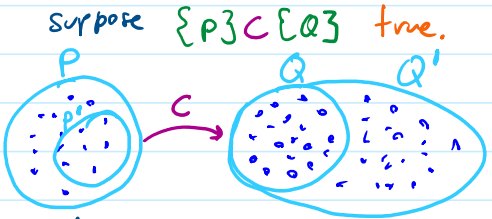
So:

$\{\underset{P}{b > 3}\}\ a \leftarrow 3*b-1 ;\ b \leftarrow 4*a-22\ \{\underset{Q}{b > 10}\} = \text{true}$

- Rule of Consequence.

$$\frac{P' \subseteq P,\ \{P\}C\{Q\},\ Q \subseteq Q'}{\{P'\}C\{Q'\}}$$

Picture:



Restricting the Precondition

Relaxing the postcondition.

E.g.

$\{b > 4\}\ a \leftarrow b-4\ \{a > 0\}$ True.

Restrict the precondition

$\{b > 10\}\ a \leftarrow b-4\ \{a > 0\}$ true. $\quad \{b > 10\} \subseteq \{b > 4\}$

Relax the postcondition

$\{b > 4\}\ a \leftarrow b-4\ \{a \geq 0\}$ true. $\quad \{a > 0\} \subseteq \{a \geq 0\}$

- Conditional Rule

$\{R \wedge P\}\,C_1\,\{Q\}\quad,\quad\{\neg B \wedge P\}\,C_2\,\{Q\}$

- Conditional Rule

$$\frac{\{B \wedge P\}\, C_1\, \{Q\},\ \{\neg B \wedge P\}\, C_2\, \{Q\}}{\{P\}\ \text{IF } B \text{ THEN } C_1 \text{ ELSE } C_2\ \{Q\}}$$

E.g. Prove:

$\{0 \le x \le 12\}$ IF $x < 12$ THEN $x \leftarrow x+1$ ELSE $x \leftarrow 0$ $\{0 \le x \le 12\}$ = True

We need to prove:

1) $\{x < 12 \wedge 0 \le x \le 12\}\ x \leftarrow x+1\ \{0 \le x \le 12\}$ = True.
   $\{0 \le x \le 11\}$

2) $\{\neg(x < 12) \wedge 0 \le x \le 12\}\ x \leftarrow 0\ \{0 \le x \le 12\}$ = True
   $\{x = 12\}$

Proof of 1)

$\{0 \le x+1 \le 12\}\ x \leftarrow x+1\ \{0 \le x \le 12\}$ = True.
$\{-1 \le x \le 11\}$

- $\{0 \le x \le 11\} \subseteq \{-1 \le x \le 11\}$ by rule of consequence.

$\{0 \le x \le 11\}\ x \leftarrow x+1\ \{0 \le x \le 12\}$ = True.

Proof of 2)

$\{0 \le 0 \le 12\}\ x \leftarrow 0\ \{0 \le x \le 12\}$
$\{\text{True}\}$ $\{x = 12\} \subseteq \{\text{True}\}$
by rule of consequence
$\{x = 12\}\ x \leftarrow 0\ \{0 \le x \le 12\}$ = True

- While Rule.

$$\frac{\{B \wedge P\}\, C\, \{P\}}{\{P\}\ \text{WHILE } B \text{ DO } C\ \{\neg B \wedge P\}}$$

($\neg B$: loop condition)

P: loop invariant.

P: is something that is true before and after the loop.

Properties of the invariant

- Initialization: Invariant is true before the loop
- Maintenance: Invariant is true at the end of each iteration
- Termination: Invariant is true after the loop is completed

Find the invariant P such that $\{P \wedge \neg B\}$ tells you something

- Termination Invariant is True after the loop is completed

Find the invariant P such that $\{P \wedge \neg B\}$ tells you something useful

E.g.

$\{x \le y\}$ WHILE $x < y$ DO $x \leftarrow x+1$ $\{x \le y \wedge \neg(x < y)\}$

$\{x = y\}$

Let's use loop invariants.

```
FUNCTION sum ( a[0.. n-1])
    sum ← 0; i ← 0
```

$\{P: sum = \sum_{0}^{i-1} a[i]\}$

```
    WHILE i < n DO
    |   sum ← sum + a[i]
    |   i ← i+1
```

$\{P \wedge \neg B: sum = \sum_{0}^{i-1} a[i] \wedge i = n\}$

```
    RETURN sum.
```

$\Rightarrow sum = \sum_{0}^{n-1} a[i]$

```
FUNCTION max ( a[0.. n-1])   // pre - array is not empty
    x ← a[0]
    i ← 1;
```

$\{P:$ x is the max of $a[0..i-1]\}$

```
    WHILE i < n DO
    |   IF a[i] > x THEN
    |   |   x ← a[i]
    |   i ← i+1
```

$\{P \wedge \neg B:$ x is the max of $a[0..i-1]$ AND $i = n$

$\Rightarrow$ x is the max of $a[0..n-1]$

```
    RETURN x;
```

```
FUNCTION Sort ( A[0.. n-1])
    i ← 0
    WHILE i < n-1 DO
    |   min ← i
    |   i ← i+1
```

$\{P: A[0..i-1]$ is Sorted. AND is a permutation of $A\}$

$[0..A[min]$ is the $=$ $A[i..i-1]]$

```
min ← i
j ← i+1
WHILE j ≤ n DO
    IF A[j] < A[min] THEN
        min ← j
    j ← j+1
Swap A[i] with A[min]
i ← i+1;
```

← {P'': A[min] is the smallest in A[i..j-1]}

← {P' ∧ ¬B' = A[min] is the smallest in A[i..n-1]}

← {P ∧ ¬B: A[0..n-1] is sorted AND is a permutation of A}
i = n