# *Guarded Commands, Nondeterminancy, and Formal Derivation of Programs*

*E.W. Dijkstra, CACM 18,8 pp.453-457, (Aug. 1975).*
***A Discipline of Programming,*** *Prentice-Hall, 1976.*

*Two statements made from guarded commands*
*guarded_command* **::=** *guard* → *guarded_list*
*guarded_list* **::=** *boolean_expression*
*guarded_list* **::=** *statement* **{ ;** *statement* **}**
*guarded_command_set* **::=** *guarded_command* **{** | *guarded_command* **}**
*alternative_construct* **::=** ***if (*** *guarded_command_set* | ε ***)*** ***fi***
*repeatative_construct* **::=** ***do (*** *guarded_command_set* | ε ***)*** ***od***
*statement* **::=** *alternative_construct* | *repeatative_construct*
| *concurrent_assignment_statement* | ***skip*** | ***abort*** | ...

$\textbf{if}\ B_1 \rightarrow SL_1 \mid B_2 \rightarrow SL_2 \mid \ldots \mid B_n \rightarrow SL_n\ \textbf{fi}$
$\textbf{do}\ B_1 \rightarrow SL_1 \mid B_2 \rightarrow SL_2 \mid \ldots \mid B_n \rightarrow SL_n\ \textbf{od}$

## ***Concurrent assignment*** *vs* ***Sequential assignment***

$x, y := y, x$ $\qquad$ $x := y; y := x$

$y := x; x := y$

$t := x; x := y; y := t$

$x_1, x_2, \ldots, x_n := E_1, E_2, \ldots, E_n$ *Concurrent assignment statement*

$S_1; S_2; \ldots, S_n$ *Statements List(SL)*

*Two* ***separators*** *; and , has different meanings*

## *Alternative statement and Nondeterminancy*

$$\textbf{\textit{if}}\ \ x \geq y \rightarrow m := x$$
$$|\ \ x \leq y \rightarrow m := y$$
$$\textbf{\textit{fi}}.$$

## *Nondeterminancy of alternative statement*

$$\textbf{\textit{if}}\ x=y \rightarrow m := x \vee m := y$$

$$\textbf{\textit{if fi}} \equiv \textbf{\textit{abort}}$$

## *Repeatative statement*

$$q_1, q_2, q_3, q_4 := Q_1, Q_2, Q_3, Q_4;$$
$$\textbf{\textit{do}}\ q_1 > q_2 \rightarrow q_1, q_2 := q_2, q_1$$
$$|\ \ q_2 > q_3 \rightarrow q_2, q_3 := q_3, q_2$$
$$|\ \ q_3 > q_4 \rightarrow q_3, q_4 := q_4, q_3$$
$$\textbf{\textit{od}}.$$
$$\textbf{\textit{do od}} \equiv \textbf{\textit{skip}}$$

*Syntax of* ***alternatve*** *and* ***repeatative*** *statements*

$\textbf{\textit{if}}\ B_1 \rightarrow SL_1 \mid B_2 \rightarrow SL_2 \mid \ldots \mid B_n \rightarrow SL_n\ \textbf{\textit{fi}}.$

$\textbf{\textit{do}}\ B_1 \rightarrow SL_1 \mid B_2 \rightarrow SL_2 \mid \ldots \mid B_n \rightarrow SL_n\ \textbf{\textit{od}}.$

| | |
|---|---|
| *P* | *Loop invariance condition* |
| *BB* | *There exists at least one guard that is true* |

$P = 1 \leq {}^{\forall}i \leq 4$: $q_i$'s are permutation of $Q_i$

$\neg BB = \neg((q_1 > q_2) \vee (q_2 > q_3) \vee (q_3 > q_4))$

$= \neg(q_1 > q_2) \wedge \neg(q_2 > q_3) \wedge \neg(q_3 > q_4))$

$= (q_1 \leq q_2) \wedge (q_2 \leq q_3) \wedge (q_3 \leq q_4)$

$= (q_1 \leq q_2 \leq q_3 \leq q_4)$

*P is true before the loop(**initialization** of P),*
*P remains true in the loop(loop **invariance**), and*
*P is also true after the loop terminates(loop **terminating**),*
*then $P \wedge \neg BB$ is true after the loop.*
*for (init; BB; update) ... in C*

*Given $n > 0$ and $0 \leq \forall i < n$: f(i) is defined.*
*Determine k .϶. $0 \leq k < n \wedge (\forall i: 0 \leq i < n: f(k) \geq f(i))$.*

```
k, j := 0, 1;
do j ≠ n → if f(j) ≤ f(k) → j := j+1
            | f(j) ≥ f(k) → k := j; j :=j+1      vs k, j := j, j+1
            fi
od.
```

## Formal Derivation of Programs

$m = max(x, y)$

$R: (m = x \vee m = y) \wedge m \geq x \wedge m \geq y.$

"$m := x$" $R_m^{m=x} = (x = x \vee x = y) \wedge x \geq x \wedge x \geq y \equiv x \geq y.$

**if** $x \geq y \rightarrow m := x$ **fi**

"$m := y$" $R_m^{m=y} = (y = x \vee y = y) \wedge y \geq x \wedge y \geq y \equiv y \geq x.$

**if** $y \geq x \rightarrow m := y$ **fi**

**if** $x \geq y \rightarrow m := x$
| $y \geq x \rightarrow m := y$
**fi**.

*Given two positive numbers X and Y, find x .э. x = gcd(X, Y)*
*Loop invariance P: introduce two local variables x and y.*

$$gcd(X, Y) = gcd(x, y) \wedge x > 0 \wedge y > 0.$$

*Initialization of the loop invariance P*

$$x, y := X, Y$$

*Do something under the loop invariance of P*

$$gcd(x, y) = gcd(x-y, y) \text{ or } gcd(x, y-x)$$

"$x := x-y$" $\quad P_x^{x=x-y} = gcd(X, Y) = gcd(x-y, y) \wedge x-y > 0 \wedge y > 0$

$$= x > y.$$

**do** $x > y \rightarrow x := x - y$ **od**.

"$y := y-x$" $\quad P_y^{y=y-x} = gcd(X, Y) = gcd(x, y-x) \wedge x > 0 \wedge y-x > 0$

$$= y > x.$$

**do** $y > x \rightarrow y := y - x$ **od**.

$x, y := X, Y;$
**do** $x > y \rightarrow x := x - y$
  $| \; x < y \rightarrow y := y - x$
**od**.

*Does the loop* ***terminate****?*

$|x - y|$ *is a non-negative monotonically decreasing integer function.*
*Loop terminates when* $x=y$*, i.e.,* $|x - y|$ *becomes zero.*

**if** $X > 0$ **and** $Y > 0 \rightarrow$
  $x, y := X, Y;$
  **do** $x > y \rightarrow x := x - y$
    $| \; x < y \rightarrow y := y - x$
  **od**
**fi**

***if fi*** $\equiv$ ***abort*** ***do od*** $\equiv$ ***skip***