Chapter 4: General Programming Techniques

Program derivation is not mechanical; in general, it is a challenging activity and it requires creativity. The derivations show where the creativity comes in.

Efficiency and $\mathcal{O}$-Notation

- Big-O:

  The time complexity is a function of the constants in the specification.

  $\mathcal{O}(f)$: upper bound of the number of steps (modulo a constant factor).

- Typical time complexities

$\mathcal{O}(2^N)$	exponential
$\mathcal{O}(N^2)$	quadratic
$\mathcal{O}(N)$	linear
$\mathcal{O}(\log N)$	logarithmic

Invariance Theorem

$$\{P\}\mathbf{do}\ B_0 \to S_0\ []\ B_1 \to S_1\ \mathbf{od}\{Q\}$$

provided that

(i) $[P \wedge \neg B_0 \wedge \neg B_1 \Rightarrow Q]$

(ii) P is invariant under S_0 and S_1.

- $\{P \wedge B_0\}S_0\{P\}$
- $\{P \wedge B_1\}S_1\{P\}$

(iii) bound function t such that

- $[P \wedge (B_0 \vee B_1) \Rightarrow t \geq 0]$
- $\{P \wedge B_0 \wedge t = C\}S_0\{t < C\}$
- $\{P \wedge B_1 \wedge t = C\}S_1\{t < C\}$

How to find a suitable invariant?

Taking Conjuncts as Invariant

$$\{\underline{P}\}\textbf{do } \underline{\neg Q} \rightarrow S_1 \textbf{ od } \{\underline{P \wedge Q}\}$$

- *Example 1*: Design S meeting

$$\{\mathit{true}\}S\{x \leq y\}.$$

Taking *true* as invariant P, $x \leq y$ as Q, we have

$$\{\mathit{true}\}\textbf{do } x > y \rightarrow x, y := y, x \textbf{ od } \{x \leq y\}$$

(Note: (1) $x - y$ is a bound function. (2) $x := y - 1$ is also fine.)

- *Example 2*: Design S meeting

$$\{true\}S\{a \leq b \wedge b \leq c \wedge c \leq d\}.$$

Taking *true* as invariant P, we have

$$\begin{array}{l}
\{true\} \\
\mathbf{do}\ a > b \rightarrow a, b := b, a \\
\ [] \quad b > c \rightarrow b, c := c, b \\
\ [] \quad c > d \rightarrow c, d := c, b \\
\mathbf{od} \\
\{a \leq b \wedge b \leq c \wedge c \leq d\}
\end{array}$$

(Why does it terminate?)

- *Example 3.* Design *divmod* meeting the specification:

$$
\begin{array}{l}
|[\\
\mathbf{con}\ A, B : \mathit{int}\ \{A \geq 0 \wedge B > 0\} \\
\mathbf{var}\ q, r : \mathit{int} \\
\mathit{divmod} \\
\{q = A\ \mathbf{div}\ B \wedge r = A\ \mathbf{mod}\ B\} \\
]|
\end{array}
$$

Note that according to the definitions of **div** and **mod**, the post-condition R is

$$R:\ \ A = q * B + r \wedge 0 \leq r \wedge r < B.$$

What is a suitable invariant?

From the post-condition

$$R:\; A = q * B + r \;\wedge\; 0 \leq r \;\wedge\; r < B$$

we may choose as invariant

$$P:\; A = q * B + r \;\wedge\; 0 \leq r$$

and as guard $\neg(r < B)$, leading to a program of the form:

$$\{P\}\mathbf{do}\; r \geq B \rightarrow S\; \mathbf{od}\{R\}.$$

- Initialization: $q, r := 0, A$, satisfying P
- Bound function: r
- Choose S as $r := r - B$.

$$
\begin{array}{ll}
 & P(r := r - B) \\
\equiv & \quad \{ \text{ substitution } \} \\
 & A = q * B + r - B \ \wedge \ 0 \leq r - B \\
\equiv & \quad \{ \text{ calculus } \} \\
 & A = \underline{(q-1)} * B + r \ \wedge \ B \leq r
\end{array}
$$

Having $q := q + 1$ can keep the invariant, i.e.,

$$P(q, r := q + 1, r - B) \ \Leftarrow \ P \ \wedge \ r \geq B.$$

This yields the following solution to *divmod*:

$$q, r, := 0, A;$$
$$\{\text{invariant: } A = p * B + r \ \wedge \ 0 \leq r, \text{ bound: } r\}$$
$$\mathbf{do}\ r \geq B \rightarrow q, r := q + 1, r - B\ \mathbf{od}$$
$$\{R\}$$

Replacing Constants by Variables

- *Example 1*

  Consider the problem of computation of A to the power B for given naturals A and B:

$$
\begin{array}{l}
|[\\
\textbf{con } A, B : int; \\
\textbf{var } r : int; \\
exponentiation \\
\{r = A^B\} \\
]|
\end{array}
$$

  What is a suitable invariant P?

We may introduce a *fresh variable* x and choose as invariant

$$P: \ r = A^x \ \wedge \ x \le B$$

and as bound $B - x$. This yields the following program scheme:

$$r, x := 1, 0 \ \{P\}; \ \mathbf{do} \ x \ne B \to S \ \mathbf{od}\{r = A^B\}$$

We investigate the efficient of increasing x by 1:

$$\begin{array}{ll} & P(x := x + 1) \\ \equiv & \quad \{ \text{ substitution } \} \\ & r = A^{x+1} \ \wedge \ (x+1) \le B \\ \equiv & \quad \{ \ A^{x+1} = A^x * A \ \} \\ & r = \mathbf{r} * A \ \wedge \ x \le B - 1 \\ \equiv & \quad \{ \text{ calculus } \} \\ & \underline{r = \mathbf{r} * A} \ \wedge \ x \le B \ \wedge \ x \ne B \end{array}$$

We obtain the following solution for *exponentiation*:

$$
\begin{array}{l}
|[\\
\textbf{var } x : \mathit{int}; \\
r, x := 1, 0; \\
\{\text{invariant: } P, \text{ bound: } B - x\} \\
\textbf{do } x \neq B \rightarrow r, x := r * A, x + 1 \textbf{ od} \\
\{r = A^B\} \\
]|
\end{array}
$$

- *Example 2*

  Derive a solution to *summation*, satisfying the following specification:

$$
\begin{array}{l}
|[\\
\textbf{con}\ N : int\{N \geq 0\};\ f : \textbf{array}\ [0..N)\ \textbf{of}\ int; \\
\textbf{var}\ x : int; \\
summation \\
\{x = (\Sigma i : 0 \leq i < N : f.i)\} \\
]|
\end{array}
$$

  We may choose as

  ▶ invariant: $P = P_0 \wedge P_1$

  * $P_0 = \{x = (\Sigma i : 0 \leq i < n : f.i)\}$, n is a fresh variable
  * $P_1 = 0 \leq n \leq N$: a bound for n

  ▶ bound function: $N - n$.

- Find initial values satisfying P:

$$x, n = 0, 0$$

- Investigate the effect of increasing n by 1 (according to the termination requirement), and we can get the following:

$$\{P \ \wedge \ n \neq N\} \ x, n = x + f.n, n + 1 \ \{P\}$$

- Final solution for *summation*:

$$
\begin{array}{l}
|[\\
\textbf{var } n : \mathit{int}; \{N \geq 0\} \\
n, x := 0, 0; \\
\{\text{invariant: } P_0 \wedge P_1, \text{ bound: } N - n\} \\
\textbf{do } n \neq N \rightarrow x, n := x + f.n, n + 1 \textbf{ od} \\
]|
\end{array}
$$

Strengthening Invariants

- *Example 1*

  Derive a program for the computation of Fibonacci function.

$$
\begin{array}{l}
|[\\
\textbf{con}\ N : \mathit{int}\ \{N \geq 0\}; \\
\textbf{var}\ x : \mathit{int}; \\
\mathit{fibonacci} \\
\{x = fib.N\} \\
]|
\end{array}
$$

  where fib is defined by

$$
\begin{array}{lcl}
fib.0 & = & 0 \\
fib.1 & = & 1 \\
fib.(n+2) & = & fib.n + fib.(n+1)
\end{array}
$$

We may have as invariant $P_0 \wedge P_1$, where

$$\begin{array}{ll} P_0 & x = fib.n \\ P_1 & 0 \leq n \leq N \end{array}$$

which is established by $n, x := 0, 0$. And we may take $N - n$ as bound function.

Notice that

$$\begin{array}{rcl} P_0(n := n+1; x := E) & = & E = fib.(n+1) \\ & = & E = \ldots\ x\ \ldots\ n\ \ldots ? \end{array}$$

By strengthening invariant to $P_0 \wedge P_1 \wedge Q$, where

$$Q: \; y = fib.(n+1)$$

and assuming $P_0 \wedge P_1 \wedge Q$, we have

$$\begin{array}{rcl}
P_0(n := n+1) & = & x = fib.(n+1) \\
 & = & x = y \\
P_1(n := n+1) & = & 0 \leq n+1 \leq N \\
 & \Leftarrow & P_1 \;\wedge\; n \neq N \\
Q_0(n := n+1) & = & y = fib.((n+1)+1) \\
 & = & y = fib.n + fib.(n+1) \\
 & = & y = x + y
\end{array}$$

So we can obtain the following solution.

$$
\begin{array}{l}
|[\\
\mathbf{var}\ n, y : \mathit{int}; \{N \geq 0\} \\
n, x, y := 0, 0, 1; \\
\{\text{invariant: } P_0 \wedge P_1 \wedge Q, \text{ bound: } N - n\} \\
\mathbf{do}\ n \neq N \rightarrow x, y, n := y, x + y, n + 1\ \mathbf{od} \\
\{x = fib.N \ \wedge \ y = fib.(N + 1)\} \\
]|
\end{array}
$$

This program has the complexity of $\mathcal{O}(N)$.

- *Example 2*

  Derive, given array $f[0..N)$, a program for S, satisfying the following specification.

$$
\begin{array}{l}
|[\\
\textbf{con } N : \mathit{int}\ \{N \geq 0\};\ f :\ \textbf{array}[0..N)]\ \textbf{of}\ \mathit{int}; \\
\textbf{var } r : \mathit{int}; \\
S \\
\{r = (\#i, j\ :\ 0 \leq i < j < N\ :\ f.i \leq 0\ \wedge\ f.j \geq 0)\} \\
]|
\end{array}
$$

Derivation

1. Deriving Invariants.

   By replacing constants by variables, we come up with the following invariants:

   $$\begin{array}{ll} P_0: & r = (\#i, j \ : \ 0 \le i < j < n \ : \ f.i \le 0 \ \wedge \ f.j \ge 0) \\ P_1: & 0 \le n \le N \end{array}$$

   which are initialized by $n, r := 0, 0$.

2. We change $n := n + 1$, and derive programs keeping invariants. Assuming $P_0 \wedge P_1 \wedge n \neq N$, we have

$$
\begin{array}{ll}
 & (\#i, j \ : \ 0 \leq i < j < n+1 \ : \ f.i \leq 0 \ \wedge \ f.j \geq 0) \\
= & \quad \{ \text{ split off } j = n \text{ } \} \\
 & (\#i, j \ : \ 0 \leq i < j < n \ : \ f.i \leq 0 \ \wedge \ f.j \geq 0) + \\
 & (\#i \ : \ 0 \leq i < n \ : \ f.i \leq 0 \ \wedge \ f.n \geq 0) \\
= & \quad \{ \ P_0 \ \} \\
 & r + (\#i \ : \ 0 \leq i < n \ : \ f.i \leq 0 \ \wedge \ f.n \geq 0) \\
= & \quad \{ \text{ case analysis } \} \\
 & \begin{array}{ll} r & \text{if } f.n < 0 \\ r + (\#i \ : \ 0 \leq i < n \ : \ f.i \leq 0) & \text{if } f.n \geq 0 \end{array} \\
= & \quad \{ \text{ introduction of invariant } Q : \ s = (\#i \ : \ 0 \leq i < n \ : \ f.i \leq 0) \text{ } \} \\
 & \begin{array}{ll} r & \text{if } f.n < 0 \\ r + s & \text{if } f.n \geq 0 \end{array}
\end{array}
$$

For the invariance of Q, we derive, assuming $P_0 \wedge P_1 \wedge Q \wedge n \neq N$,

$$
\begin{array}{ll}
 & (\#i \ : \ 0 \leq i < n+1 \ : \ f.i \leq 0) \\
= & \quad \{ \text{ split off } i = n \ \} \\
 & (\#i \ : \ 0 \leq i < n \ : \ f.i \leq 0) + \#.(f.n \leq 0) \\
= & \quad \{ \text{ Q } \} \\
 & s + \#.(f.n \leq 0) \\
= & \quad \{ \text{ definition of } \# \ \} \\
 & \begin{array}{ll} s & \text{if } f.n > 0 \\ s+1 & \text{if } f.n \leq 0 \end{array}
\end{array}
$$

3. We summarize the above and obtain the following solution.

$|[$

var $n, s : \mathit{int}; \{N \geq 0\}$

$n, r, s := 0, 0, 0;$

$\{$invariant: $P_0 \wedge P_1 \wedge Q$, bound $N - n\}$

do $n \neq N \rightarrow$

  if $f.n < 0 \rightarrow$ **skip**

  $[\,]$ $f.n \geq 0 \rightarrow r := r + s$

  fi;

  if $f.n > 0 \rightarrow$ **skip**

  $[\,]$ $f.n \leq 0 \rightarrow s := s + 1$

  fi;

  $n := n + 1$

od

$]|$

$\{r = (\#i, j \;:\; 0 \leq i < j < N \;:\; f.i \leq 0 \;\wedge\; f.j \geq 0)\}$

Exercises

Derive a solution for the following programming problems.

[Problem 4-1]

$$
\begin{array}{l}
|[\\
\textbf{con}\ N, X : \mathit{int}\ \{N \geq 0\};\ f : \textbf{array}\ [0..N)\ \textbf{of}\ \mathit{int}; \\
\textbf{var}\ r : \mathit{int} \\
S \\
\{r = (\Sigma i : 0 \leq i < N : f.i * X^i)\} \\
]|.
\end{array}
$$

[Problem 4-2]

$|[$

$\mathbf{con}\ N : int\ \{N \geq 1\};\ A : \mathbf{array}\ [0..N)\ \mathbf{of}\ int;$

$\mathbf{var}\ r : int$

S

$\{r = (\mathbf{max}\ p\ q : 0 \leq p < q < N : A.p - A.q)\}$

$]|.$