Polynomial Evaluation

G52DOA - Derivation of Algorithms

Venanzio Capretta

Example of Program Derivation

Evaluation of Polynomials

We are going to derive an algorithm
to evaluate polynomials efficiently.
A polynomial is an expression like:

$$a_0 + a_1 \cdot x + a_2 \cdot x^2 + \cdots + a_{n-2} \cdot x^{n-2} + a_{n-1} \cdot x^{n-1}.$$

We assume that it is given as an array of coefficients.
Input: Array of numbers a of length n, number x.
Output: Value of: $\sum_{i=0}^{n-1} \mathrm{a}[i] \cdot \mathrm{x}^i$.

Polynomial Evaluation

We are looking for a program p
satisfying the following specification:

$$\{\top\}\ p\ \{\mathsf{y} = \sum_{i=0}^{n-1} \mathsf{a}[i] \cdot \mathsf{x}^i\}$$

with the requirement that the variables
a,n, and x are not modified in it.
The naive algorithm uses a loop, at each step
the power x^i is computed, multiplied by $\mathsf{a}[i]$
and added to the total.

Horner's Rule

A more efficient algorithm uses Horner's Rule: the polynomial can be rewritten as:

$$a_0 + x \cdot (a_1 + x \cdot (a_2 + x \cdot (\cdots x \cdot (a_{n-2} + x \cdot (a_{n-1})) \cdots))).$$

Idea:

- Use a counter k that runs backwards from n − 1 to 1;
- Start with y = a[n − 1];
- At each step multiply y by x and add a new element of a.

The Invariant of the Algorithm

From the informal description of the algorithm, we can deduce what the invariant should be:

$$I \equiv \mathsf{y} = \sum_{i=\mathsf{k}}^{\mathsf{n}-1} \mathsf{a}[i] \cdot \mathsf{x}^{i-\mathsf{k}} \wedge 0 \leq \mathsf{k} < \mathsf{n}$$

Now we just need to find some initialization commands that make I true at the beginning and a terminating loop that has I as invariant!

Polynomial Evaluation Algorithm

$\{\top\}$

$\{\text{invariant} : I\}$

$\{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$

Polynomial Evaluation Algorithm

$$\begin{array}{l|l|l} & & \{\top\} \\ \{\text{invariant} : I\} & \text{while } ? \text{ do } (& \\ &) & \{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\} \end{array}$$

Polynomial Evaluation Algorithm

$$\{\text{invariant} : I\} \quad \text{while } k > 0 \text{ do } (\quad) \quad \Big| \quad \{\top\} \quad \{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$$

Polynomial Evaluation Algorithm

		$\{\top\}$
$\{\text{invariant} : I\}$	while $k > 0$ do (	$\{I \wedge k > 0\}$
	)	$\{I\}$ $\{I \wedge \neg k > 0\}$ $\{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$

Polynomial Evaluation Algorithm

		$\{\top\}$
$\{\text{invariant} : I\}$	while $k > 0$ do (	$\{I \wedge k > 0\}$
	$k := k - 1;$	
	$y := S$	$\{I\}$
	)	$\{I \wedge \neg k > 0\}$
		$\{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$

Polynomial Evaluation Algorithm

		$\{\top\}$
$\{\text{invariant} : I\}$	while $k > 0$ do (	$\{I \wedge k > 0\}$
$\{R_1\}$	$k := k - 1;$	
$\{R_2\}$	$y := S$	$\{I\}$
	)	$\{I \wedge \neg k > 0\}$
		$\{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$

Polynomial Evaluation Algorithm

		$\{\top\}$
$\{\text{invariant} : I\}$	while $k > 0$ do (	$\{I \wedge k > 0\}$
$\{R_1\}$	$k := k - 1;$	
$\{R_2\}$	$y := a[k] + y \cdot x$	$\{I\}$
	)	$\{I \wedge \neg k > 0\}$
		$\{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$

Polynomial Evaluation Algorithm

		$\{\top\}$
	$k := n - 1;$	
	$y := a[k];$	$\{I\}$
$\{\text{invariant} : I\}$	while $k > 0$ do (	$\{I \wedge k > 0\}$
$\{R_1\}$	$k := k - 1;$	
$\{R_2\}$	$y := a[k] + y \cdot x$	$\{I\}$
	)	$\{I \wedge \neg k > 0\}$
		$\{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$

Polynomial Evaluation Algorithm

		$\{\top\}$
	if $n = 0$ then $y := 0$	
	else (	$\{n > 0\}$
	$k := n - 1;$	
	$y := a[k];$	$\{I\}$
$\{\text{invariant} : I\}$	while $k > 0$ do (	$\{I \wedge k > 0\}$
$\{R_1\}$	$k := k - 1;$	
$\{R_2\}$	$y := a[k] + y \cdot x$	$\{I\}$
	)	$\{I \wedge \neg k > 0\}$
	)	$\{y = \sum_{i=0}^{n-1} a[i] \cdot x^i\}$

Additional Derivations

$$R_2 \equiv I[S/\mathsf{y}]$$
$$R_1 \equiv I[S/\mathsf{y}][\mathsf{k}-1/\mathsf{k}]$$
$$\equiv S[\mathsf{k}-1/\mathsf{k}] = \textstyle\sum_{i=\mathsf{k}-1}^{\mathsf{n}-1} \mathsf{a}[i] \cdot \mathsf{x}^{i-\mathsf{k}+1} \wedge 0 \leq \mathsf{k}-1 < \mathsf{n}$$

Derivation of the value of S:

Assuming: $\mathsf{y} = \sum_{i=\mathsf{k}}^{\mathsf{n}-1} \mathsf{a}[i] \cdot \mathsf{x}^{i-\mathsf{k}}$

$$\begin{aligned} S[\mathsf{k}-1/\mathsf{k}] &= \textstyle\sum_{i=\mathsf{k}-1}^{\mathsf{n}-1} \mathsf{a}[i] \cdot \mathsf{x}^{i-\mathsf{k}+1} \\ &= \mathsf{a}[\mathsf{k}-1] \cdot \mathsf{x}^{\mathsf{k}-1-\mathsf{k}+1} + \textstyle\sum_{i=\mathsf{k}}^{\mathsf{n}-1} \mathsf{a}[i] \cdot \mathsf{x}^{i-\mathsf{k}+1} \\ &= \mathsf{a}[\mathsf{k}-1] + (\textstyle\sum_{i=\mathsf{k}}^{\mathsf{n}-1} \mathsf{a}[i] \cdot \mathsf{x}^{i-\mathsf{k}}) \cdot \mathsf{x} \\ &= \mathsf{a}[\mathsf{k}-1] + \mathsf{y} \cdot \mathsf{x} \\ &= (\mathsf{a}[\mathsf{k}] + \mathsf{y} \cdot \mathsf{x})[\mathsf{k}-1/\mathsf{k}] \end{aligned}$$

So: $S = \mathsf{a}[\mathsf{k}] + \mathsf{y} \cdot \mathsf{x}$.

Polynomial Evaluation Algorithm

Without a special case for empty array:

```
k := n;
y := 0;
while k > 0 do (
    k := k − 1;
    y := a[k] + y · x
    )
```

with the invariant:

$$I' \equiv y = \sum_{i=k}^{n-1} a[i] \cdot x^{i-k} \wedge 0 \leq k \leq n$$