Algorithm derivation through Flame

Victor Eijkhout, Susan Lindsey

COE 322 Fall 2021

Dijkstra quote, part 1

Today a usual technique is to make a program and then to test it. But: program testing can be a very effective way to show the presence of bugs, but is hopelessly inadequate for showing their absence. (cue laughter)

Dijkstra quote, part 2

The only effective way to raise the confidence level of a program significantly is to give a convincing proof of its correctness. But one should not first make the program and then prove its correctness, because then the requirement of providing the proof would only increase the poor programmer's burden. On the contrary: the programmer should let correctness proof and program grow hand in hand.

Matrix-vector product

$$y = Ax$$

Partitioned:

$$\begin{pmatrix} y_T \\ y_B \end{pmatrix} = \begin{pmatrix} A_T \\ A_B \end{pmatrix} (x)$$

Two equations:

$$\begin{cases} y_T = A_T x \\ y_B = A_B x \end{cases}$$

Inductive construction

$$\begin{pmatrix} y_T \\ y_B \end{pmatrix} = \begin{pmatrix} A_T \\ A_B \end{pmatrix} (x)$$

Assume only equation

$$y_T = A_T x$$

is satisfied, and grow the T block.

Algorithm outline

$$\begin{pmatrix} y_T \\ y_B \end{pmatrix} = \begin{pmatrix} A_T \\ A_B \end{pmatrix} (x)$$

While T is not the whole system

Predicate: $y_T = A_T x$ true

Update: grow T block by one

Predicate: $y_T = A_T x$ true for new/bigger T block

Note initial and final condition.

Inductive step

Here is the big trick
Before

$$\begin{pmatrix} y_T \\ y_B \end{pmatrix} = \begin{pmatrix} A_T \\ A_B \end{pmatrix} (x)$$

split:

$$\begin{pmatrix} y_1 \\ \cdots \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} A_1 \\ \cdots \\ A_2 \\ A_3 \end{pmatrix} (x)$$

Then the update step, and
After

$$\begin{pmatrix} y_1 \\ y_2 \\ \cdots \\ y_3 \end{pmatrix} = \begin{pmatrix} A_1 \\ A_2 \\ \cdots \\ A_3 \end{pmatrix} (x)$$

and unsplit

$$\begin{pmatrix} y_T \\ y_B \end{pmatrix} = \begin{pmatrix} A_T \\ A_B \end{pmatrix} (x)$$

Before the update:

$$\begin{pmatrix} y_1 \\ \dots \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} A_1 \\ \dots \\ A_2 \\ A_3 \end{pmatrix} (x)$$

so

$$y_1 = A_1 x$$

is true

Then the update step, and

After

$$\begin{pmatrix} y_1 \\ y_2 \\ \dots \\ y_3 \end{pmatrix} = \begin{pmatrix} A_1 \\ A_2 \\ \dots \\ A_3 \end{pmatrix} (x)$$

so

$$\begin{cases} y_1 = A_1 x & \text{we had this} \\ y_2 = A_2 x & \text{we need this} \end{cases}$$

Resulting algorithm

While T is not the whole system

Predicate: $y_T = A_T x$ true

Update: $y_2 = A_2 x$

Predicate: $y_T = A_T x$ true for new/bigger T block

Matrix-vector product, the other way around

$$y = Ax$$

Partitioned:

$$(y) = \begin{pmatrix} A_L & A_R \end{pmatrix} \begin{pmatrix} x_T \\ x_B \end{pmatrix}$$

Equation:

$$\left\{ y = A_L x_T + A_R x_B \right.$$

Inductive construction

$$(y) = \begin{pmatrix} A_L & A_R \end{pmatrix} \begin{pmatrix} x_T \\ x_B \end{pmatrix}$$

Assume

$$y = A_L x_T$$

is constructed, and grow the T block.

Inductive step

Before

$$(y) = \begin{pmatrix} A_L & A_R \end{pmatrix} \begin{pmatrix} x_T \\ x_B \end{pmatrix}$$

split:

$$(y) = \left(\begin{array}{c|cc} A_1 & A_2 & A_3 \end{array}\right) \begin{pmatrix} x_1 \\ \cdots \\ x_2 \\ x_3 \end{pmatrix}$$

Then the update step, and
After

$$(y) = \left(\begin{array}{cc|c} A_1 & A_2 & A_3 \end{array}\right) \begin{pmatrix} x_1 \\ x_2 \\ \cdots \\ x_3 \end{pmatrix}$$

and unsplit

$$(y) = \begin{pmatrix} A_L & A_R \end{pmatrix} \begin{pmatrix} x_T \\ x_B \end{pmatrix}$$

Derivation of the update

Before the update:

$$(y) = \begin{pmatrix} A_1 & \vdots & A_2 & A_3 \end{pmatrix} \begin{pmatrix} x_1 \\ \dots \\ x_2 \\ x_3 \end{pmatrix}$$

so

$$y = A_1 x_1$$

is true
Then the update step, and
After

$$(y) = \begin{pmatrix} A_1 & A_2 & \vdots & A_3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_3 \end{pmatrix}$$

so

$$y = A_1 x_1 + A_2 x_2$$

in other words, we need

$$y \leftarrow y + A_2 x_2$$

Resulting algorithm

While T is not the whole system

Predicate: $y = A_L x_T$ true

Update: $y \leftarrow y + A_2 x_2$

Predicate: $y = A_L x_T$ true for new/bigger T block

Two algorithms

for $r = 1, m$
    $y_r = A_{r,*}x_*$

$y \leftarrow 0$
for $c = 1, n$
    $y \leftarrow y + A_{*,c}x_c$