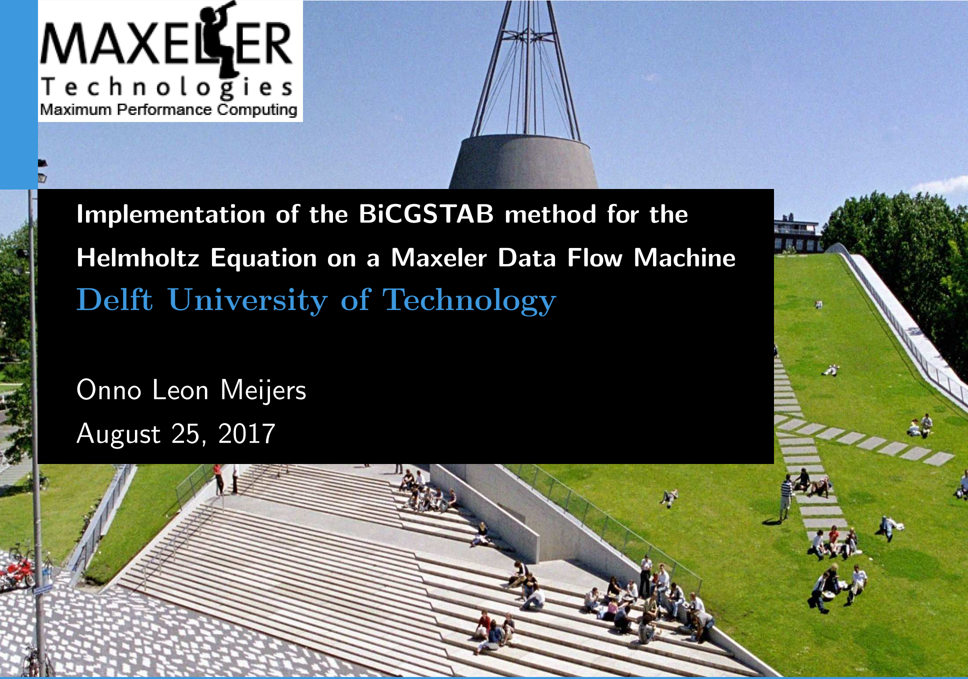


MAXELER Technologies
Maximum Performance Computing

Implementation of the BiCGSTAB method for the Helmholtz Equation on a Maxeler Data Flow Machine

Delft University of Technology

Onno Leon Meijers

August 25, 2017

Outline

1. Concepts
2. Data Flow
3. Results
4. Conclusions

1 Concepts

2 Data Flow

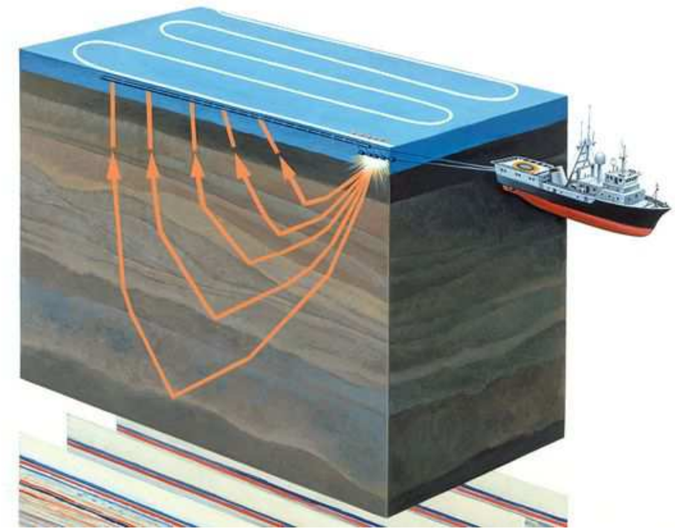
3 Results

4 Conclusions

TU Delft

What is under the surface?

Ground research process



calculation process

Helmholtz equation:

$$\mathcal{A}_{k,\alpha}\mathbf{u}(\mathbf{x}) := -\boldsymbol{\Delta}\mathbf{u}(\mathbf{x}) - (\mathbf{1} - \alpha\mathbf{i})\mathbf{k}^2(\mathbf{x})\mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x})$$

u is the wave function and we have measurements.
g is the source and is known.
k is the wave number and is unknown.
α is the damping factor and is set.

calculation process

Helmholtz equation:

$$\mathcal{A}_{k,\alpha}\mathbf{u}(\mathbf{x}) := -\boldsymbol{\Delta}\mathbf{u}(\mathbf{x}) - (\mathbf{1} - \alpha\mathbf{i})\mathbf{k}^2(\mathbf{x})\mathbf{u}(\mathbf{x}) = \mathbf{g}(\mathbf{x})$$

u is the wave function and we have measurements.
g is the source and is known.
k is the wave number and is unknown.
α is the damping factor and is set.

$$\mathcal{A}_{k,\alpha}\mathbf{u} = \mathbf{g}$$

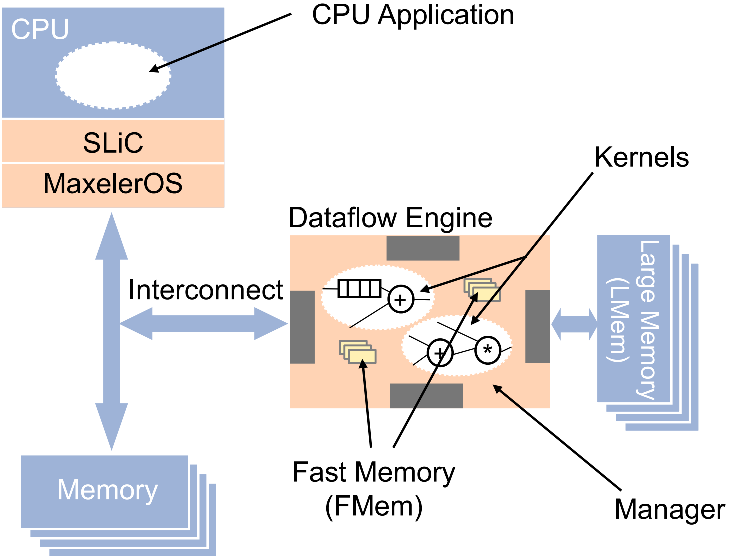
Solve for **u**.

Algorithm 1 Pseudocode for the BiCGSTAB method

$\mathbf{u} = \mathbf{v} = \mathbf{p} = \mathbf{0}$; $\mathbf{r_0} = \mathbf{r} = \mathbf{g} = \delta_{x_s,y_s,z_s}$; $\rho_{old} = \alpha = \omega = \rho_{new} = 1$;
for $i = 0, 1, 2, \ldots, maxit$ **do**
    $\beta = \frac{\rho_{new}}{\rho_{old}} \frac{\alpha}{\omega}$; $\rho_{old} = \rho_{new}$;
    $\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega \mathbf{v})$;
    $\mathbf{v} = A\mathbf{p}$;
    $\alpha = \frac{\rho_{old}}{(\mathbf{v}, \mathbf{r}_0)}$;
    $\mathbf{s} = \mathbf{r} - \alpha \mathbf{v}$;
    $\mathbf{t} = A\mathbf{s}$;
    $\omega = \frac{(\mathbf{t}, \mathbf{s})}{(\mathbf{t}, \mathbf{t})}$;
    $\mathbf{u} = \mathbf{u} + \alpha \mathbf{p} + \omega \mathbf{s}$;
    $\mathbf{r} = \mathbf{s} - \omega \mathbf{t}$;
    $\rho_{new} = (\mathbf{r}, \mathbf{r}_0)$;
    if $||\mathbf{r}||_2 < 10^{-6}$ **then**
        quit ;
    end if
end for

Maxeler Data Flow Machine

Maxeler Data Flow Machine



Title

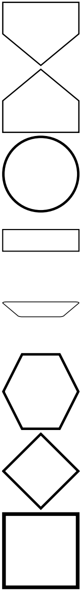
Implementation of the BiCGSTAB method for the Helmholtz Equation on a Maxeler Data Flow Machine.

1 Concepts

2 Data Flow

3 Results

4 Conclusions



Input node.

Output node.

Computation nodes.

Constant set by CPU.

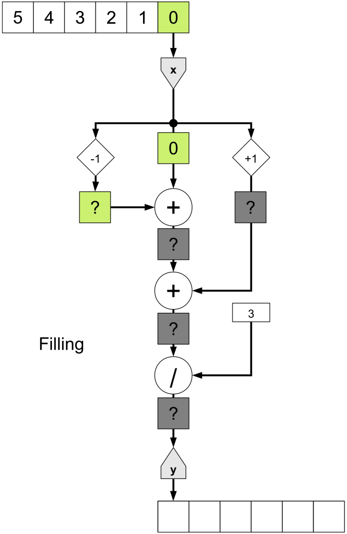
Multiplexer (mux).

Counter.

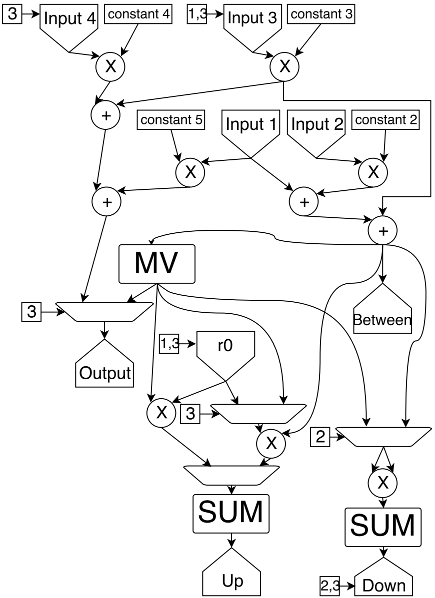
Stream offsets.

Part specifier.

Running average example



Filling



BiCGSTAB method

$\mathbf{u} = \mathbf{v} = \mathbf{p} = \mathbf{0}$;

$\mathbf{r_0} = \mathbf{r} = \mathbf{g} = \delta_{x_s,y_s,z_s}$;

$\rho_{old} = \alpha = \omega = \rho_{new} = 1$;

for($i = 0, 1, 2, \ldots, maxit$)

$\beta = \frac{\rho_{new}}{\rho_{old}} \frac{\alpha}{\omega}$; $\rho_{old} = \rho_{new}$;

$\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega\mathbf{v})$;

$\mathbf{v} = A\mathbf{p}$;

$\alpha = \frac{\rho_{old}}{(\mathbf{v},\mathbf{r}_0)}$;

$\mathbf{s} = \mathbf{r} - \alpha\mathbf{v}$;

$\mathbf{t} = A\mathbf{s}$;

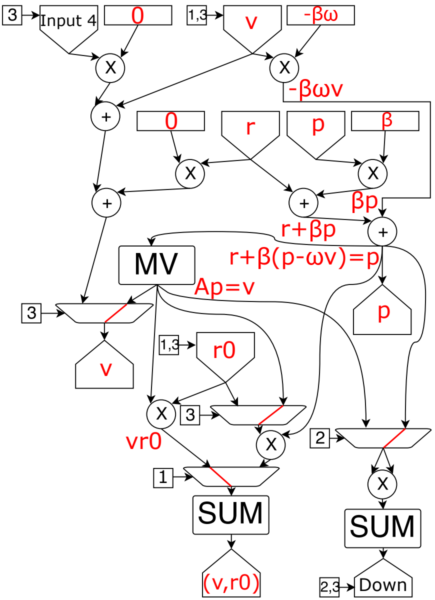
$\omega = \frac{(\mathbf{t},\mathbf{s})}{(\mathbf{t},\mathbf{t})}$;

$\mathbf{u} = \mathbf{u} + \alpha\mathbf{p} + \omega\mathbf{s}$;

$\mathbf{r} = \mathbf{s} - \omega\mathbf{t}$;

$\rho_{new} = (\mathbf{r}, \mathbf{r}_0)$;

If($||\mathbf{r}||_2$ is small enough) then quit



BiCGSTAB method

Part 1

$\mathbf{u} = \mathbf{v} = \mathbf{p} = \mathbf{0}$;

$\mathbf{r_0} = \mathbf{r} = \mathbf{g} = \delta_{x_s,y_s,z_s}$;

$\rho_{old} = \alpha = \omega = \rho_{new} = 1$;

for$(i = 0, 1, 2, \ldots, maxit)$

$\beta = \frac{\rho_{new}}{\rho_{old}} \frac{\alpha}{\omega}$; $\rho_{old} = \rho_{new}$;

$\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega \mathbf{v})$;

$\mathbf{v} = A\mathbf{p}$;

$\alpha = \frac{\rho_{old}}{(\mathbf{v},\mathbf{r_0})}$;

$\mathbf{s} = \mathbf{r} - \alpha \mathbf{v}$;

$\mathbf{t} = A\mathbf{s}$;

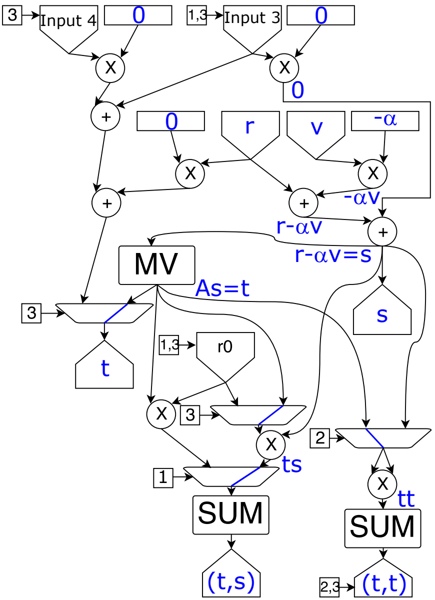
$\omega = \frac{(\mathbf{t},\mathbf{s})}{(\mathbf{t},\mathbf{t})}$;

$\mathbf{u} = \mathbf{u} + \alpha \mathbf{p} + \omega \mathbf{s}$;

$\mathbf{r} = \mathbf{s} - \omega \mathbf{t}$;

$\rho_{new} = (\mathbf{r}, \mathbf{r_0})$;

If($||\mathbf{r}||_2$ is small enough) then quit



BiCGSTAB method

Part 2

$\mathbf{u} = \mathbf{v} = \mathbf{p} = \mathbf{0}$;

$\mathbf{r_0} = \mathbf{r} = \mathbf{g} = \delta_{x_s,y_s,z_s}$;

$\rho_{old} = \alpha = \omega = \rho_{new} = 1$;

for($i = 0, 1, 2, \ldots, maxit$)

$\beta = \frac{\rho_{new}}{\rho_{old}} \frac{\alpha}{\omega}$; $\rho_{old} = \rho_{new}$;

$\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega \mathbf{v})$;

$\mathbf{v} = A\mathbf{p}$;

$\alpha = \frac{\rho_{old}}{(\mathbf{v}, \mathbf{r_0})}$;

$\mathbf{s} = \mathbf{r} - \alpha \mathbf{v}$;

$\mathbf{t} = A\mathbf{s}$;

$\omega = \frac{(\mathbf{t}, \mathbf{s})}{(\mathbf{t}, \mathbf{t})}$;

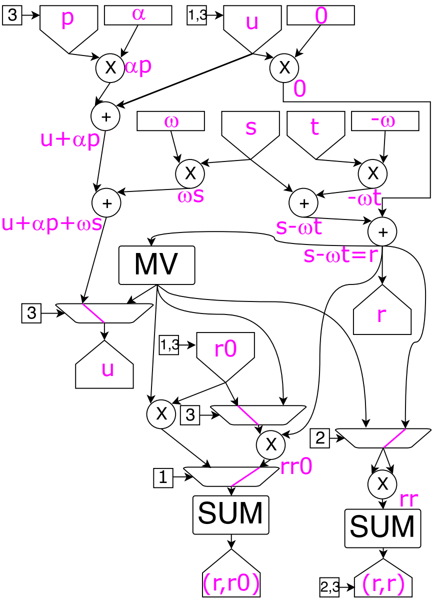
$\mathbf{u} = \mathbf{u} + \alpha \mathbf{p} + \omega \mathbf{s}$;

$\mathbf{r} = \mathbf{s} - \omega \mathbf{t}$;

$\rho_{new} = (\mathbf{r}, \mathbf{r_0})$;

If($||\mathbf{r}||_2$ is small enough) then quit

TU Delft



BiCGSTAB method

Part 3

$\mathbf{u} = \mathbf{v} = \mathbf{p} = \mathbf{0};$

$\mathbf{r_0} = \mathbf{r} = \mathbf{g} = \delta_{x_s,y_s,z_s};$

$\rho_{old} = \alpha = \omega = \rho_{new} = 1;$

$\text{for}(i = 0, 1, 2, \ldots, maxit)$

$\beta = \frac{\rho_{new}}{\rho_{old}} \frac{\alpha}{\omega};\ \rho_{old} = \rho_{new};$

$\mathbf{p} = \mathbf{r} + \beta(\mathbf{p} - \omega \mathbf{v});$

$\mathbf{v} = A\mathbf{p};$

$\alpha = \frac{\rho_{old}}{(\mathbf{v},\mathbf{r_0})};$

$\mathbf{s} = \mathbf{r} - \alpha \mathbf{v};$

$\mathbf{t} = A\mathbf{s};$

$\omega = \frac{(\mathbf{t},\mathbf{s})}{(\mathbf{t},\mathbf{t})};$

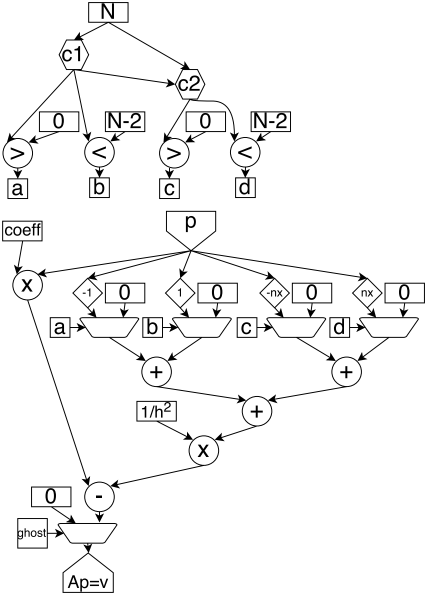
$\mathbf{u} = \mathbf{u} + \alpha \mathbf{p} + \omega \mathbf{s};$

$\mathbf{r} = \mathbf{s} - \omega \mathbf{t};$

$\rho_{new} = (\mathbf{r}, \mathbf{r}_0);$

If($||\mathbf{r}||_2$ is small enough) then quit

TU Delft



$$\mathcal{A}_{k,\alpha}\mathbf{u} := -\boldsymbol{\Delta}\mathbf{u} - (\mathbf{1} - \alpha\mathbf{i})\mathbf{k}^2\mathbf{u} \approx$$

$$\frac{1}{h^2}\begin{bmatrix} 0 & 1 & 0 \\ 1 & -4 + (1-\alpha\mathbf{i})\mathbf{k}^2\mathbf{h}^2 & 1 \\ 0 & 1 & 0 \end{bmatrix}\mathbf{u}$$

grid:

1	2	3	4	5	6
7	8	9	10	11	12
13	14	15	16	17	18
19	20	21	22	23	24
25	26	27	28	29	30
31	32	33	34	35	36

TU Delft

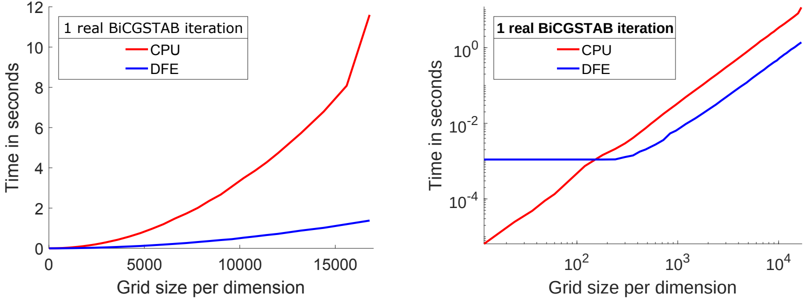
1 Concepts

2 Data Flow

3 Results

4 Conclusions

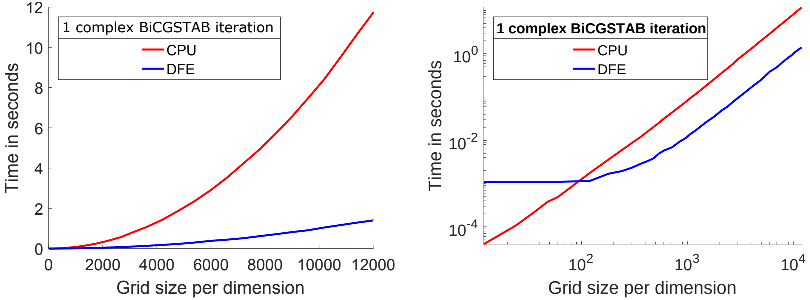
Real valued implementation



(a) normal scale. (b) logarithmic scale.

Figure: Calculation times of 1 real valued BiCGSTAB iteration.

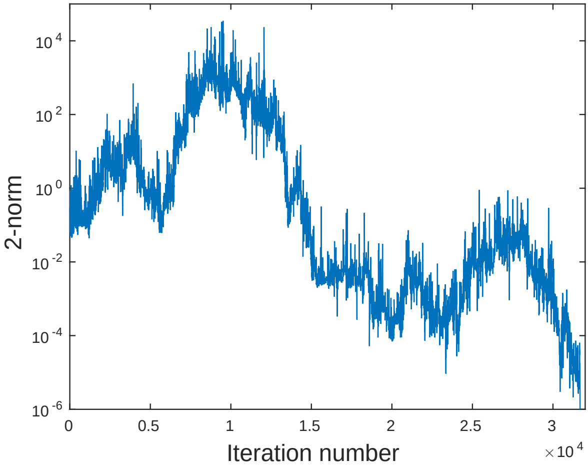
Complex valued problem.



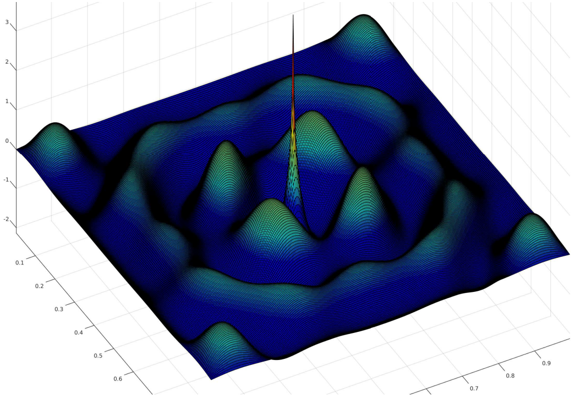
(a) normal scale. (b) logarithmic scale.

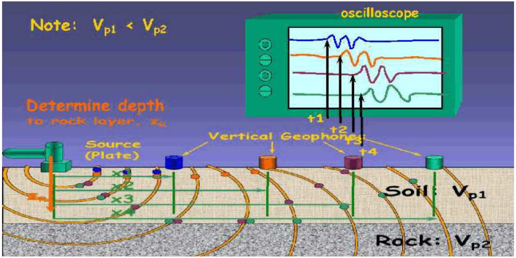
Figure: Calculation times of 1 complex valued BiCGSTAB iteration.

Convergence pattern of BiCGSTAB of a grid of 324×324



A plot of the solution of the 2D Helmholtz equation with Dirichlet boundary conditions.





[1]

[1]https://www.omicsonline.org/articles-images/Geology-Geophysics-Show-the-path-seismic-waves-refraction-from-source-5-259-g008.png

1. Concepts
2. Data Flow
3. Results
4. Conclusions

Conclusions

1. The PCIe communication speed between the CPU and the DFE is not high enough to only do 1 calculation on the DFE, like the matrix vector multiplication, and send the results back.

Conclusions

2. The BiCGSTAB method can be implemented on the DFE. This is done by splitting the algorithm in 3 parts.

Conclusions

3. An improvement can be seen over the GPU implementation of HP Knibbe[2]. This DFE implementation is 2.4 times faster for the BiCGSTAB method without preconditioner.

[2] HP Knibbe, Reduction of computing time for seismic applications based on the Helmholtz equation by Graphics Processing Units, PhD thesis, TU Delft, Delft University of Technology, 2015

Conclusions

4. Finally, an increase in communication speed between the large memory and the DFE will result in better performance of this BiCGSTAB implementation. When the communication speed is doubled the calculation time will be halved.

TU Delft

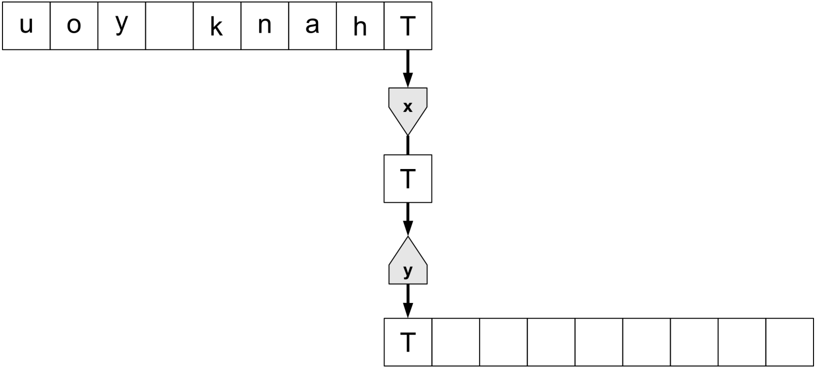
Conclusions

1. The PCIe communication speed between the CPU and the DFE is not high enough to only do 1 calculation on the DFE, like the matrix vector multiplication, and send the results back.
2. The BiCGSTAB method can be implemented on the DFE. This is done by splitting the algorithm in 3 parts.
3. An improvement can be seen over the GPU implementation of HP Knibbe[2]. This DFE implementation is 2.4 times faster for the BiCGSTAB method without preconditioner.
4. Finally, an increase in communication speed between the large memory and the DFE will result in better performance of this BiCGSTAB implementation. When the communication speed is doubled the calculation time will be halved.

[2] HP Knibbe, Reduction of computing time for seismic applications based on the Helmholtz equation by Graphics Processing Units, PhD thesis, TU Delft, Delft University of Technology, 2015

Future research

1. Sparse Matrix vector product on DFE.
2. Calculate partial sums on DFE.
3. Source point kernel instead of inner-product.
4. Implement the multi grid preconditioner on the DFE.
5. Make **k** a variable.
6. Implement the 3D problem.



TU Delft