

GPU Implementation of a Krylov solver Preconditioned by a Shifted Laplace Multigrid

Hans Knibbe*, Kees Vuik*, Kees Oosterlee#
Presented by Kees Lemmens*

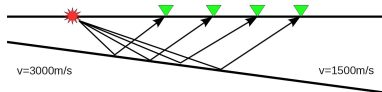
* Delft University of Technology

Dutch national research centre for mathematics and computer science (CWI)

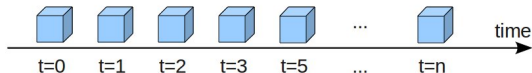
Outline

- 1 Motivation
- 2 Problem Description
- 3 Solver
- 4 GPU implementation
 - Single GPU
 - Multi-GPU
- 5 Results

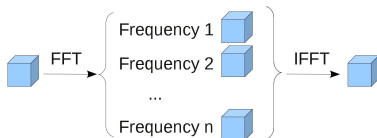
Geophysical Background



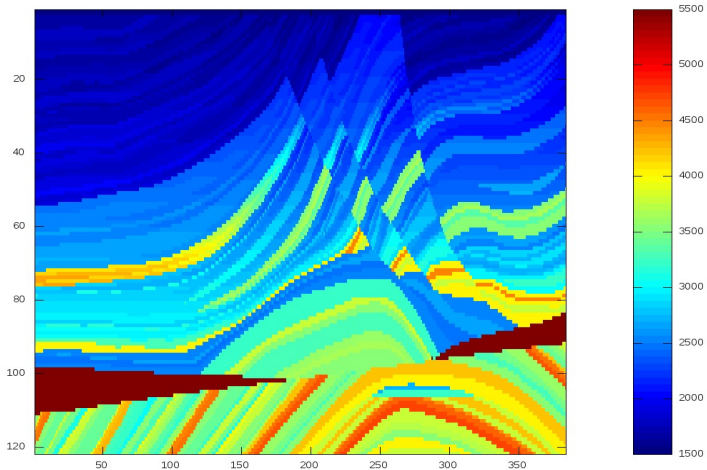
Time Domain



Frequency Domain



Marmousi 2D



Motivation

Problem Description

The Helmholtz equation in three dimensions is considered

$$-\frac{\partial^2 \phi}{\partial x^2} - \frac{\partial^2 \phi}{\partial y^2} - \frac{\partial^2 \phi}{\partial z^2} - (1 - \alpha i)k^2 \phi = g, \quad (1)$$

with non-reflecting first-order radiation boundary conditions

$$\left(-\frac{\partial}{\partial \eta} - ik \right) \phi = 0, \quad (2)$$

where $\phi = \phi(x, y, z)$ is the wave pressure field, $k = k(x, y, z)$ is the wavenumber, α is the damping coefficient, $g = g(x, y, z)$ is the source term.

Solver

Bi-CGSTAB preconditioned by shifted Laplace multigrid method. Equation solved in preconditioner is

$$-\frac{\partial^2 \phi}{\partial x^2} - \frac{\partial^2 \phi}{\partial y^2} - \frac{\partial^2 \phi}{\partial z^2} - (\beta_1 - \beta_2 i) k^2 \phi = g, \quad (3)$$

$\beta_1, \beta_2 \in \mathbb{R}$, with the same boundary conditions as the original problem.

Multigrid components:

- Matrix-dependent prolongation (2D: de Zeeuw, 1990, 3D: Zhebel, 2006)
- Standard restriction
- Multi-coloured Gauss-Seidel as a smoother

Preconditioner is computed in single precision.

Next subsection

- 1 Motivation
- 2 Problem Description
- 3 Solver
- 4 GPU implementation**
 - Single GPU
 - Multi-GPU
- 5 Results

Little-Green Machine



20 general computing nodes

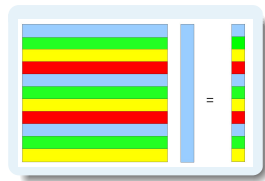
- 2 Intel quadcore E5620
- 24 GB RAM
- 2 TB disk
- 2 NVIDIA GTX480

Founded by

- University of Leiden
- NWO
- TU Delft
- KNMI

Sparse Matrix-Vector Multiplication

$$x_i = \sum_{j=1}^N A_{ij} b_j$$

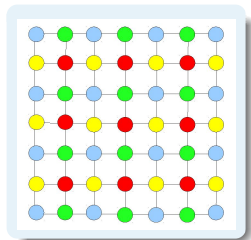


Size	Time 8-cores (ms)	Time GPU (ms)	CPU/GPU
10,000	1.5	0.1	15
100,000	8.4	0.1	84
1,000,000	64.9	1.1	59
5,000,000	328.1	5.4	61
20,000,000	1334.6	22	61

Gauss-Seidel Smoother

For each color compute

$$x_i = (1 - \omega)x_i + \omega\left(b_i - \frac{\sum_j A_{ij}x_j}{A_{ii}}\right),$$

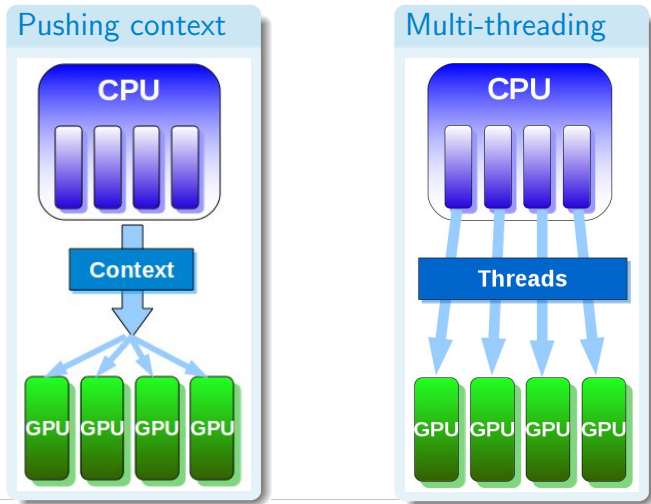


Size	Time 8-cores (ms)	Time GPU (ms)	CPU/GPU
10,000	4	0.6	7
100,000	23.4	0.8	29
1,000,000	164.5	2.5	66
5,000,000	625.9	10.3	61
20,000,000	3733.9	39.4	95

Next subsection

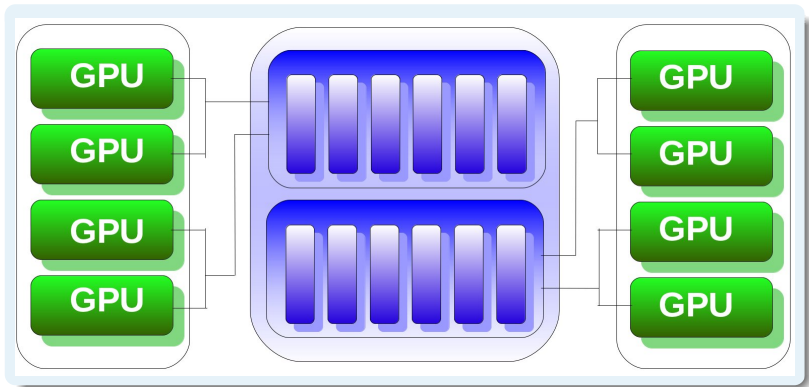
- 1 Motivation
- 2 Problem Description
- 3 Solver
- 4 GPU implementation**
 - Single GPU
 - **Multi-GPU**
- 5 Results

Multi-GPU



NVidia Computer

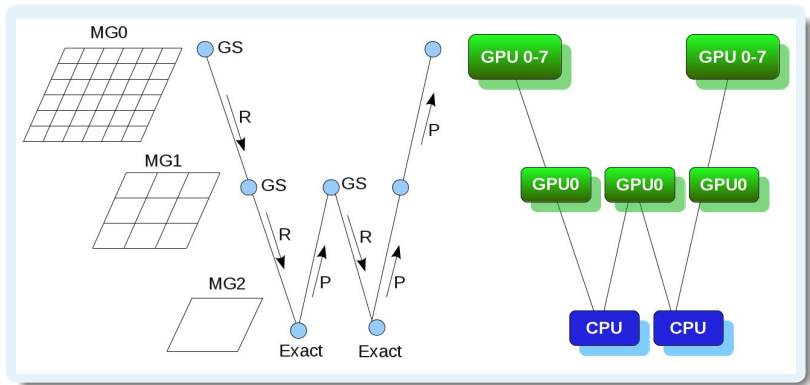
8 GPUs, each GPU has 448 cores, 3 GB RAM
12 cores (2 Westmeres), 48 GB RAM



Multi-GPU Approach

- 1 **Data-parallel approach** (e.g. vector operations on multi-GPU)
 - 1 Relatively easy to implement
 - 2 CPU→GPU→CPU data transfer
- 2 **Split of the algorithm** (e.g. solver on one GPU, preconditioner on the another one)
 - 1 No or little data transfers
 - 2 Find the best way to split the algorithm
- 3 **Domain-Decomposition approach** (e.g. each domain on a different GPU)
 - 1 Exchange of halos (still data transfer)
 - 2 Can affect convergence of the preconditioned method

Multigrid on Multi-GPU

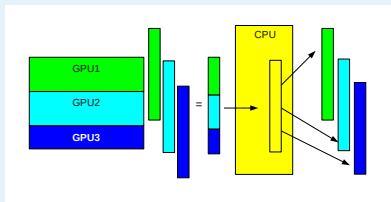


Multi-GPU Issues

- Limited GPU memory size so need multiple GPUs for large problems.
- Efficient memory reuse to avoid allocation/deallocation, e.g. pool of GPU-vectors.
- Limit communications CPU→GPU and GPU→CPU.
- Each GPU need separate texture reference.
- Cublas vectors limited to 512 MB.

Speedups for Sparse MVM

Row-wise splitting



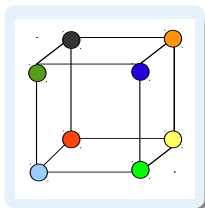
Speedups single GPU versus Multi-GPU

Size	12-cores/1 GPU	12-cores/8-GPUs
1,000,000	51.9	10.3
30,000,000	47.4	10.4
100,000,000	-	6.8

Gauss-Seidel Smoother

For each color

$$x_i = (1 - \omega)x_i + \omega\left(b_i - \frac{\sum_j A_{ij}x_j}{A_{ii}}\right),$$



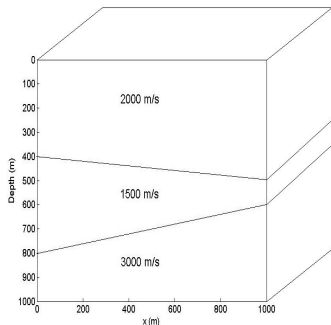
Size	12-cores/1 GPU	12-cores/8-GPUs
5,000,000	16.5	5.2
30,000,000	89.1	6.1
100,000,000	-	4.4

Bi-CGSTAB

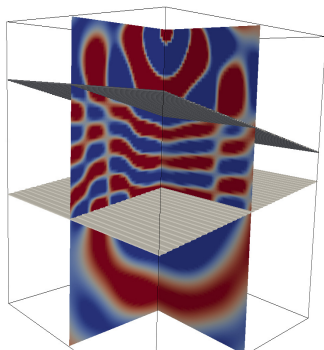
Timings for Bi-CGSTAB, single precision

n	12-cores	1 GPU	Speedup	8-GPU	Speedup
5,000,000	24 s	0.8 s	29.8	2.3 s	10.5
15,000,000	82 s	2 s	38.1	5.8 s	14.2
100,000,000	395 s	-	-	28.6 s	13.8

Wedge problem



The velocity profile of the wedge problem.



Real part of solution of the wedge problem, $f = 30$ Hz.

Bi-CGSTAB preconditioned by shifted Laplace multigrid

Problem size $350 \times 350 \times 350 \approx 43,000,000$ unknowns

	Bi-CGSTAB (DP)	Preconditioner (SP)	Total
12-cores	94 s	690 s	784 s
1 GPU	13 s	47 s	60 s
Speedup CPU/GPU	7.2	14.7	13.1
8 GPUs	83 s	86 s	169 s
Speedup CPU/GPUs	1.1	7.9	4.6
2 GPUs+split	12 s	38 s	50 s
Speedup CPU/GPU	7.8	18.2	15.5

Acknowledgments

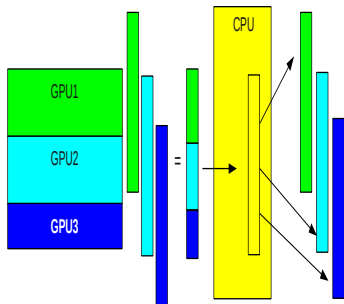
- NVIDIA for access to the latest many-core-multi-GPU architecture
- TU Delft for access to the cluster

Thank you for your attention!

For further questions please send e-mail to
hknibbe@gmail.com

Sparse MVM on Multi-GPU

Row-wise splitting



Column-wise splitting

