Introducing GPUs to a Commercial Reservoir Simulator

Dominic Walsh, Paul Woodhams, Schlumberger

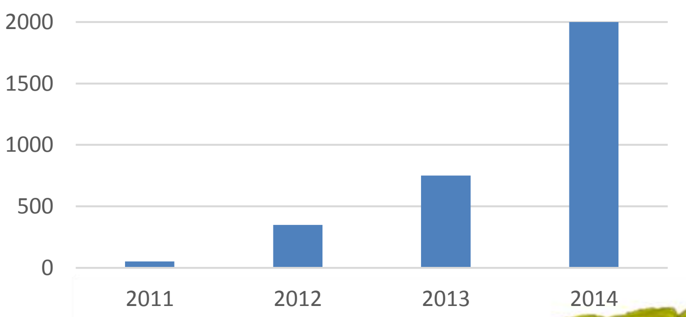
Yongpeng Zhang, Ken Esler, Stone Ridge Technology Inc.

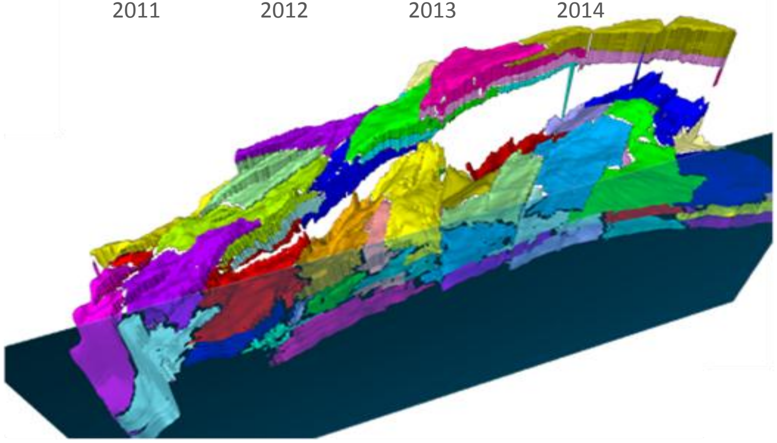
Reservoir Simulation

- Purpose: Estimate reserves, prediction of optimal recovery and production strategy
- Input: rock and fluid well data, production history
- Model size: 10^4 cells (laptops) - 10^9 cells (Linux clusters)
- Uncertainty: multiple realizations
- Embedded: Network→Plant→Economics

Very computationally demanding

Client Model Size (Millions Cells)





History and Problem Size

- GPU have been very successful in the Seismic domain
  - Seismic clusters are acquiring GPU & Infiniband → simulation ready

- Clients are being constrained by power envelopes

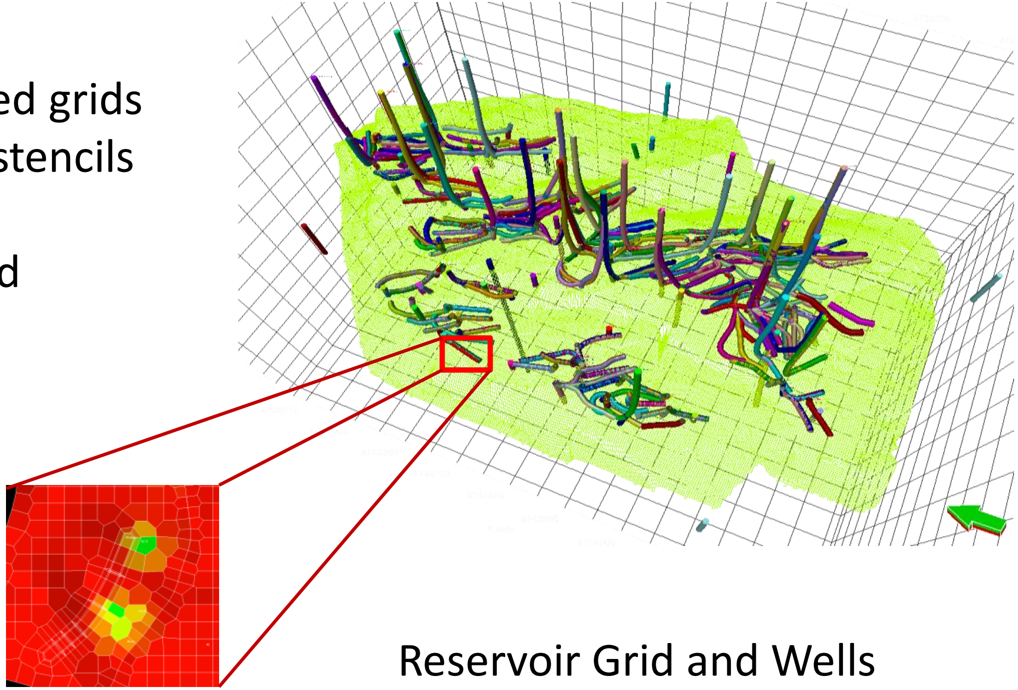
- New GPU Simulator?
  - ECLIPSE (circa 1984) is the industry standard
  - INTERSECT (circa 2010) is the “high fidelity” simulator
  - Testing and validation is measured in Man Decades
  - User base migration is expected to take 5-10 yr. timeframe

Can we take advantage of new GPU hardware while preserving this investment?

Structure

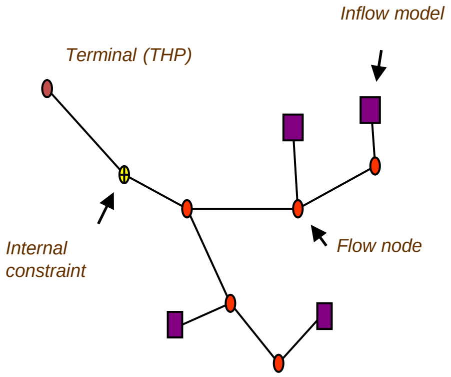
- Reservoir:
  - Deposition: Semi-structured grids
  - Finite Volume: Low order stencils
  - Static structure
  - Time Stepping: Implicit and adaptive
  - Up to Billions of cells

- Wells
  - Pipe flow
  - Introduce local structure
  - Up to 10^5 wells



Reservoir Grid and Wells

Irregularity and Nonlinearity



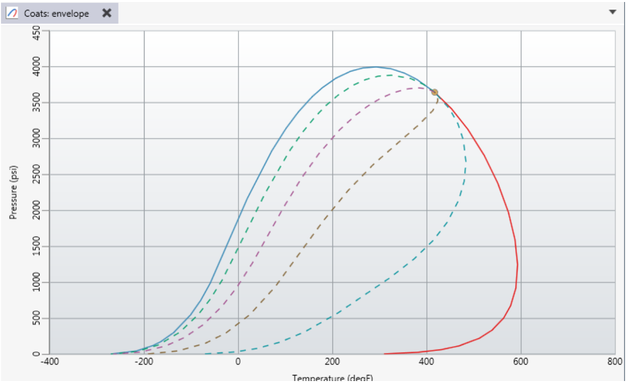
Inflow model

Terminal (THP)

Internal constraint

Flow node

Well Structure



Phase Envelope

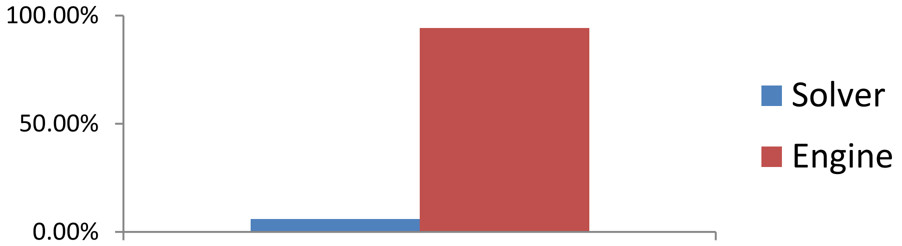
- Many small tightly-coupled sub-problems
- Time varying structure

- Complicated Fluid and Phase Modelling
  - Per-cell nonlinear systems
- Possibly non-reversible rock models

Phase I: Thermal Linear Solver

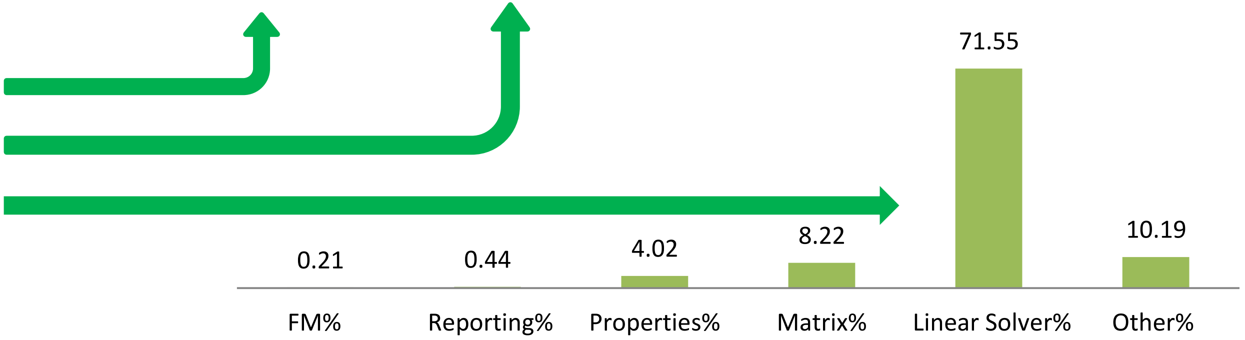
- ✓ Code volume
- ✓ Small problem size
- ✓ Fully Implicit
- ✓ Windows workstation

Lines of Code



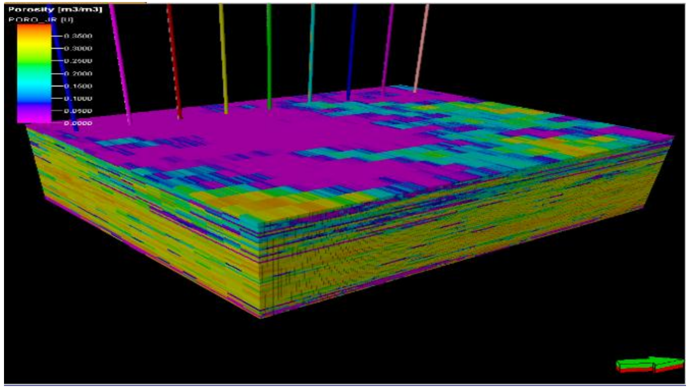
- ✓ Amdahl
  - Thermal
  - Single Box
  - Linear Solver

THERMAL 16X Parallel Runtime %

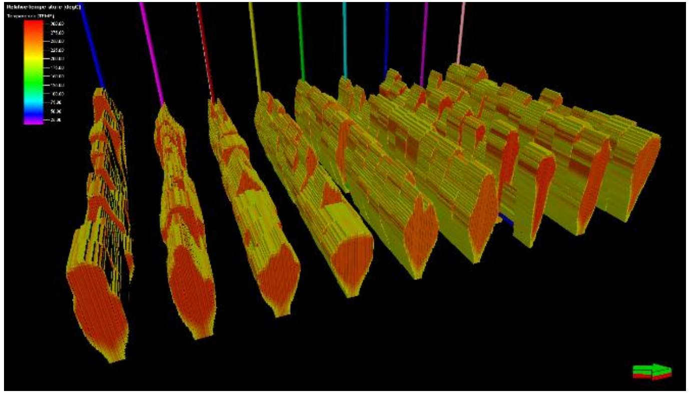


Test Model: THERM

Property Distribution



Solution Distribution



Small: 1 M Cells & 9 Well Pairs

Thermal: CH4 + Bitumen

2.5 yrs. steam injection

very strong transitions

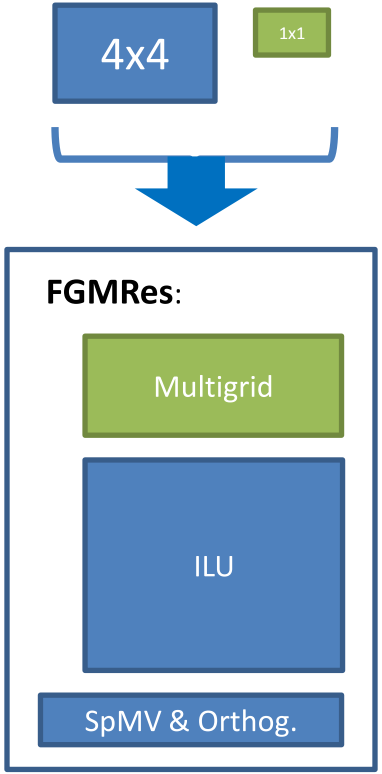
Numerically very demanding

Linear Solver Big Picture

- Iterative Solver with composite preconditioner:
  - Constrained Pressure Residual (CPR) method
  - Block 4x4 and Scalar Pressure systems

✓ Iterative solver: FGMRes
  ✓ Composite preconditioner
    ✓ Multigrid Pressure only
      ✓ GAMPACK
    ? ILU full system
      - Multi-color

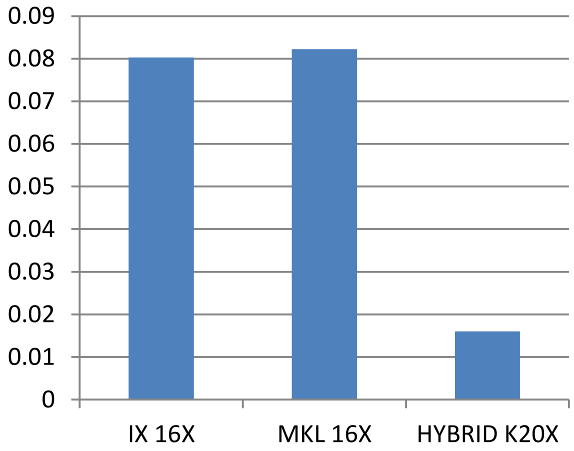
✓ SpMV & Orthogonalisation



[Parallel Forall Blog - HPCG Everett Phillips]

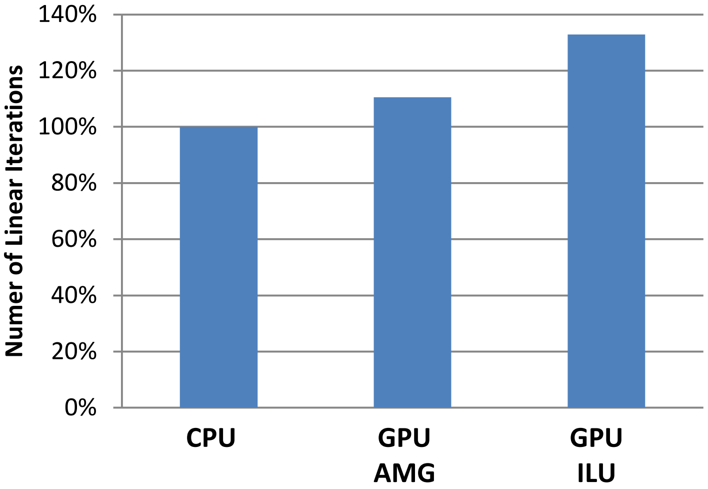
Preliminary indicators

Sparse Matrix Multiply



Good ~5X

Algorithmic Weakening

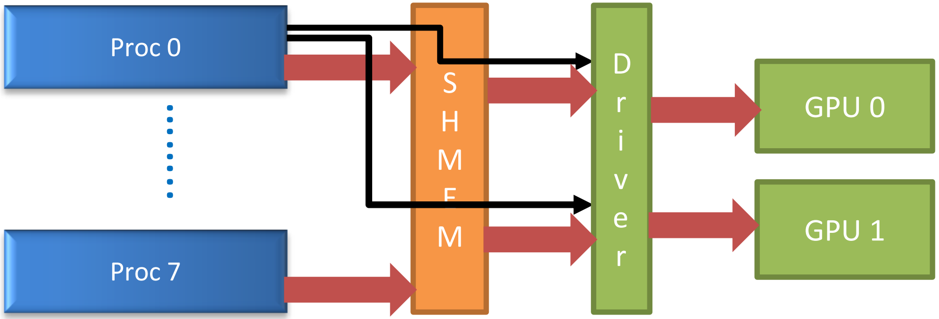


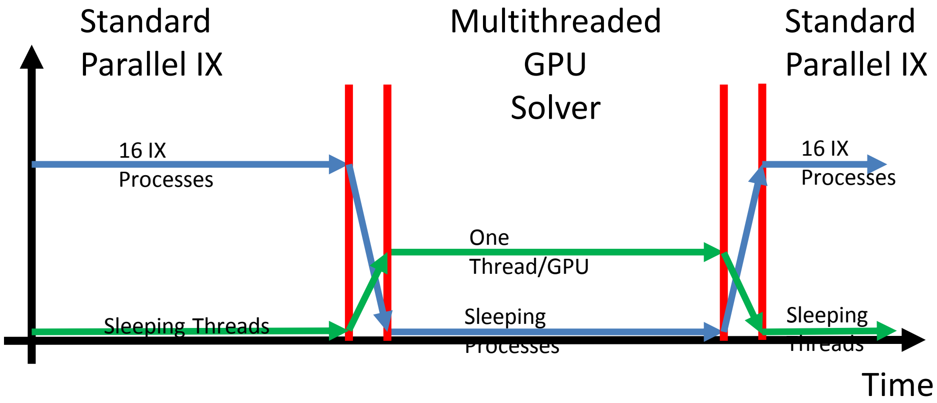
Challenge[†] ~0.9X-0.75X

[†] Fine-Grained Parallel Preconditioners for Fast GPU-based Solvers, Dimitar Lukarski GTC 2012
High Performance Algebraic Multigrid for Commercial Applications, Jonathan Cohen GTC 2013

Offload: MPI & Multiplexing

- INTERSECT:
  - MPI process per domain
- Device shared memory?
  - only Linux
  - not windows ☹
- Use threads to drive multiple cards
  - C++ NOT OpenMP
  - CUDA 7 ☺
- Transfer:
  - Stage on Host side
  - Pinned

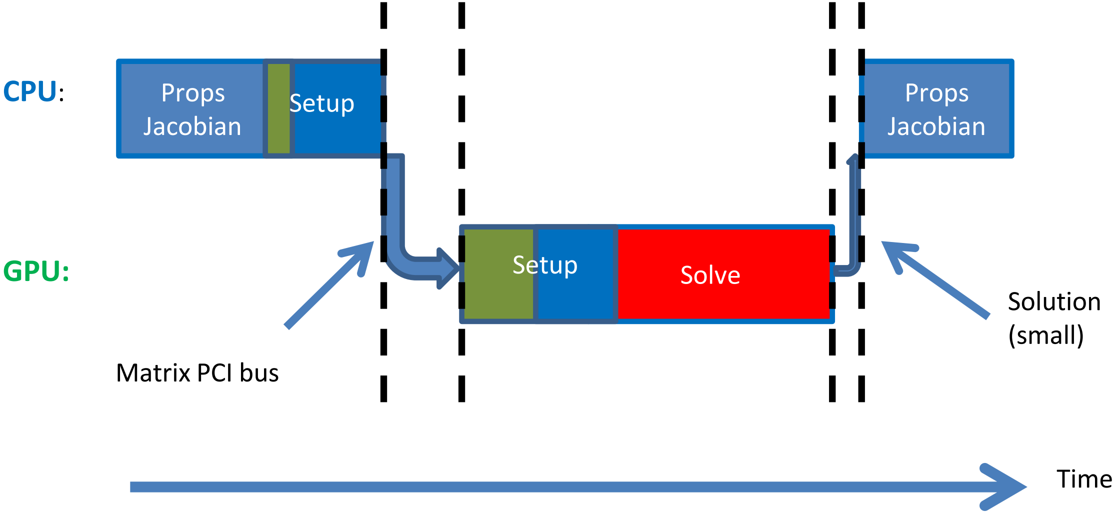




Transfer Cost

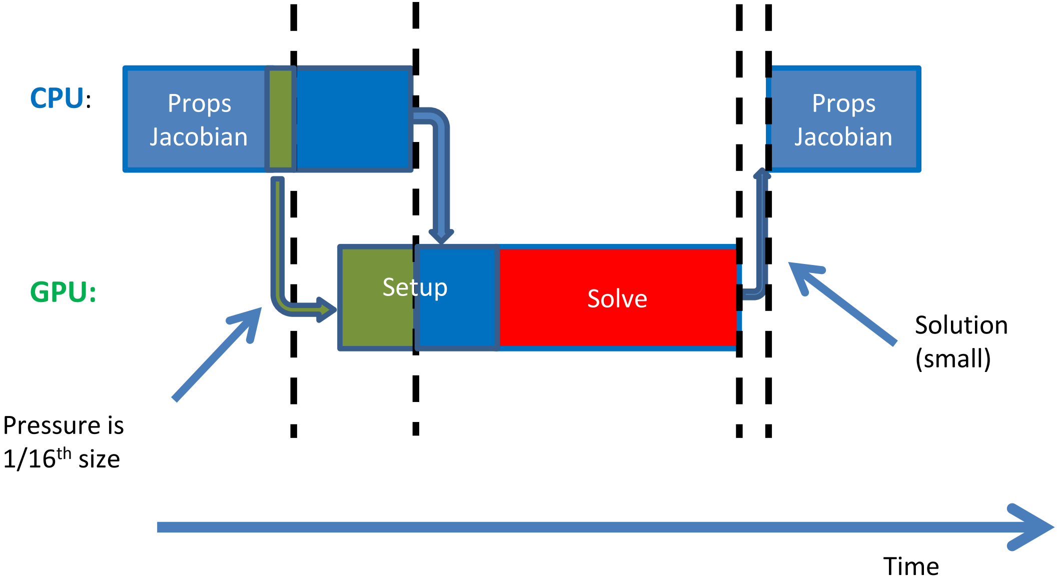
- Transfer cost is a significant fraction of complete CPU solve

- Naïve implementation not sufficient

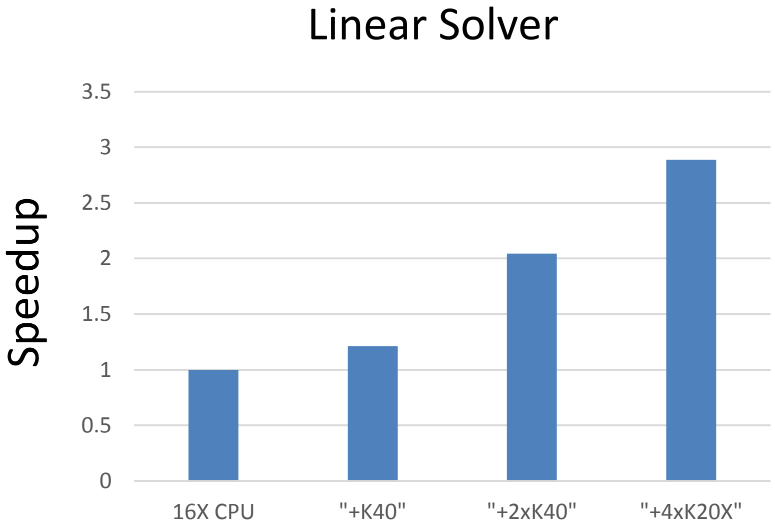


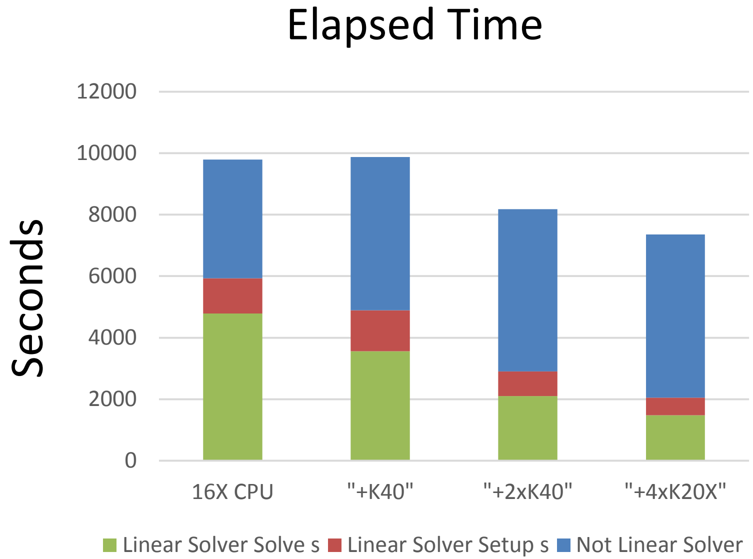
Overlapping & CPR

- CPR is a *composite* preconditioner!
  - Pressure is 1/16th
  - AMG: small but costly
  - Second stage is relatively cheap
- Use streams
  - per matrix
  - per thread/GPU
    - Lambda's in CUDA 7 ☺
- Use mixed precision



THERM Results

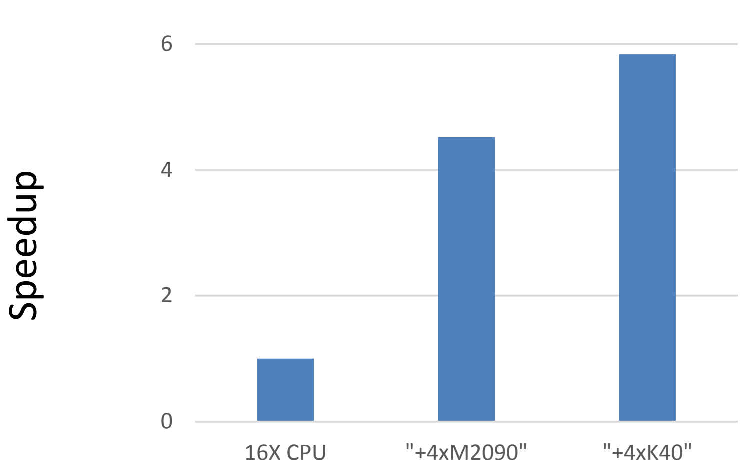




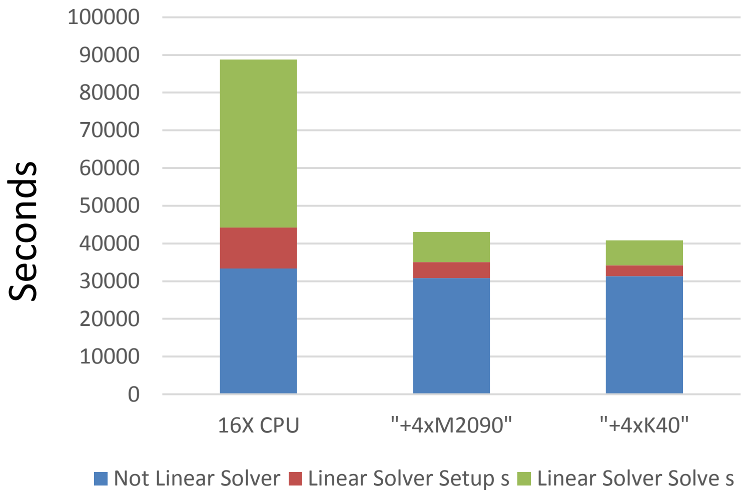
Good Solver speedup

Still carrying a lot of non-solver time
Marginal benefit

Larger Model: THERM_L (4M)

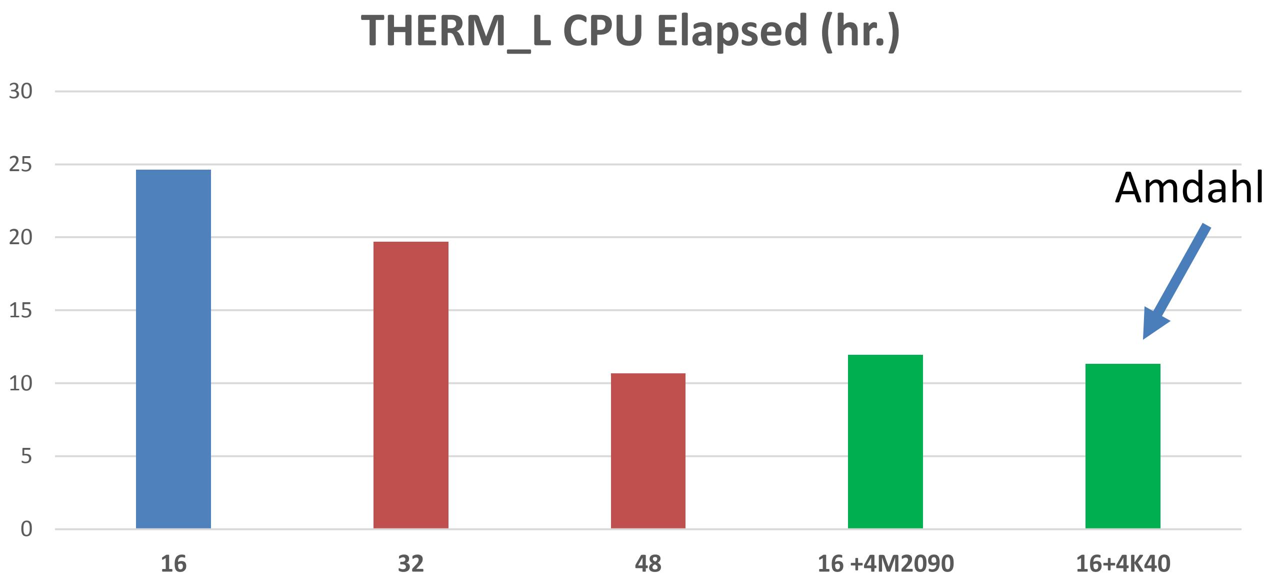

Linear Solver

Better solver speedup
More work on cards


Elapsed Time

Bigger impact

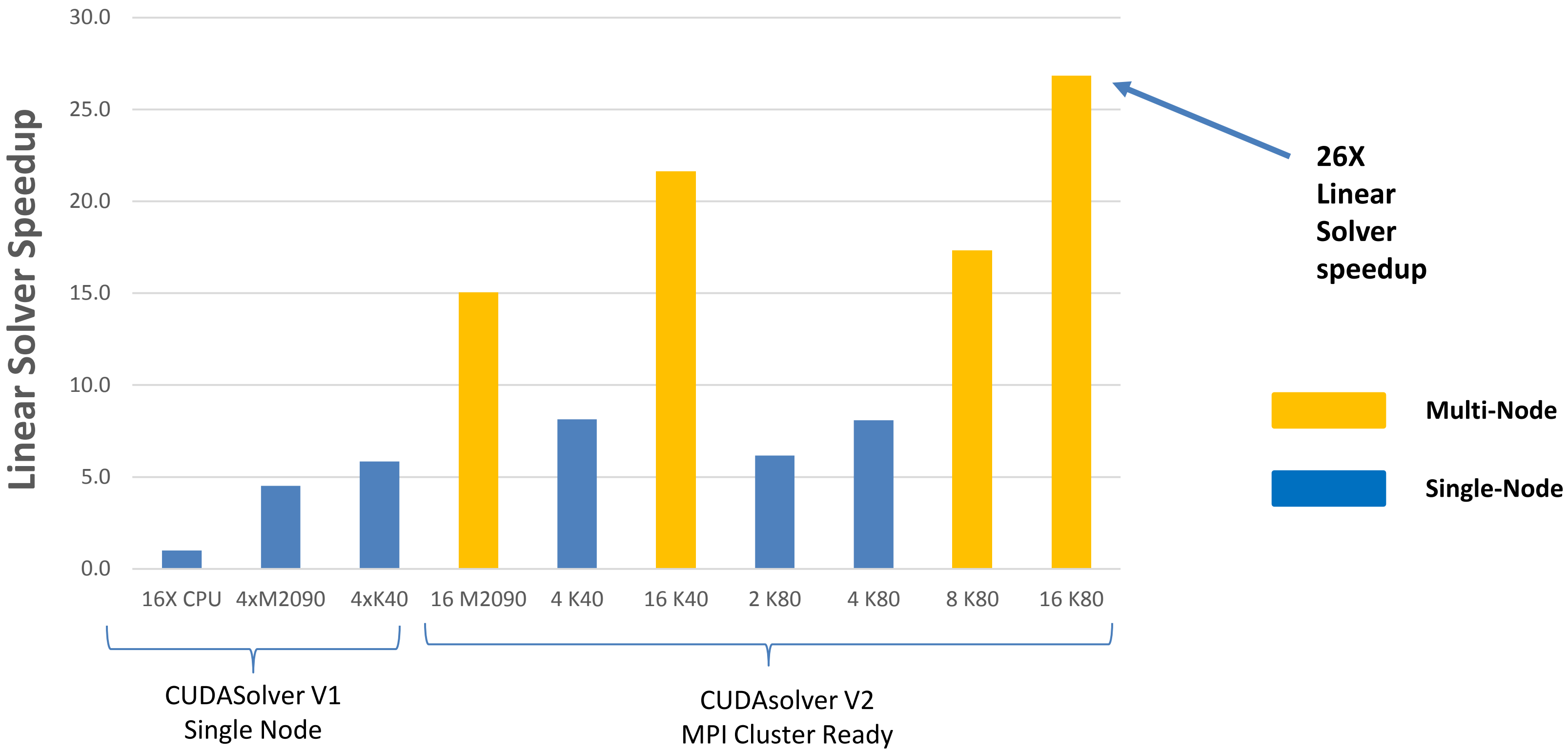
Implications: GPU vs CPU



Currently need 48 nodes to match GPU performance

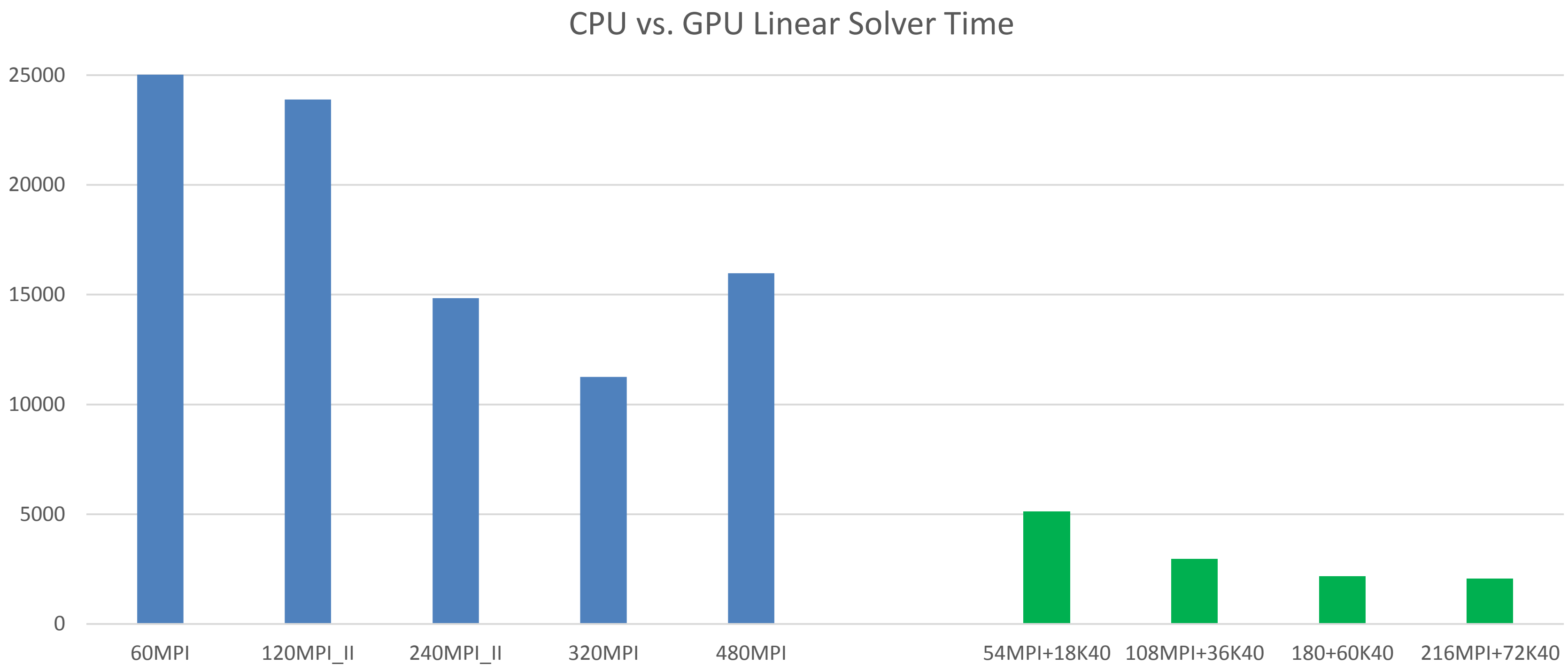
Increased CPU's does not speedup

THERM_L Strong Scaling

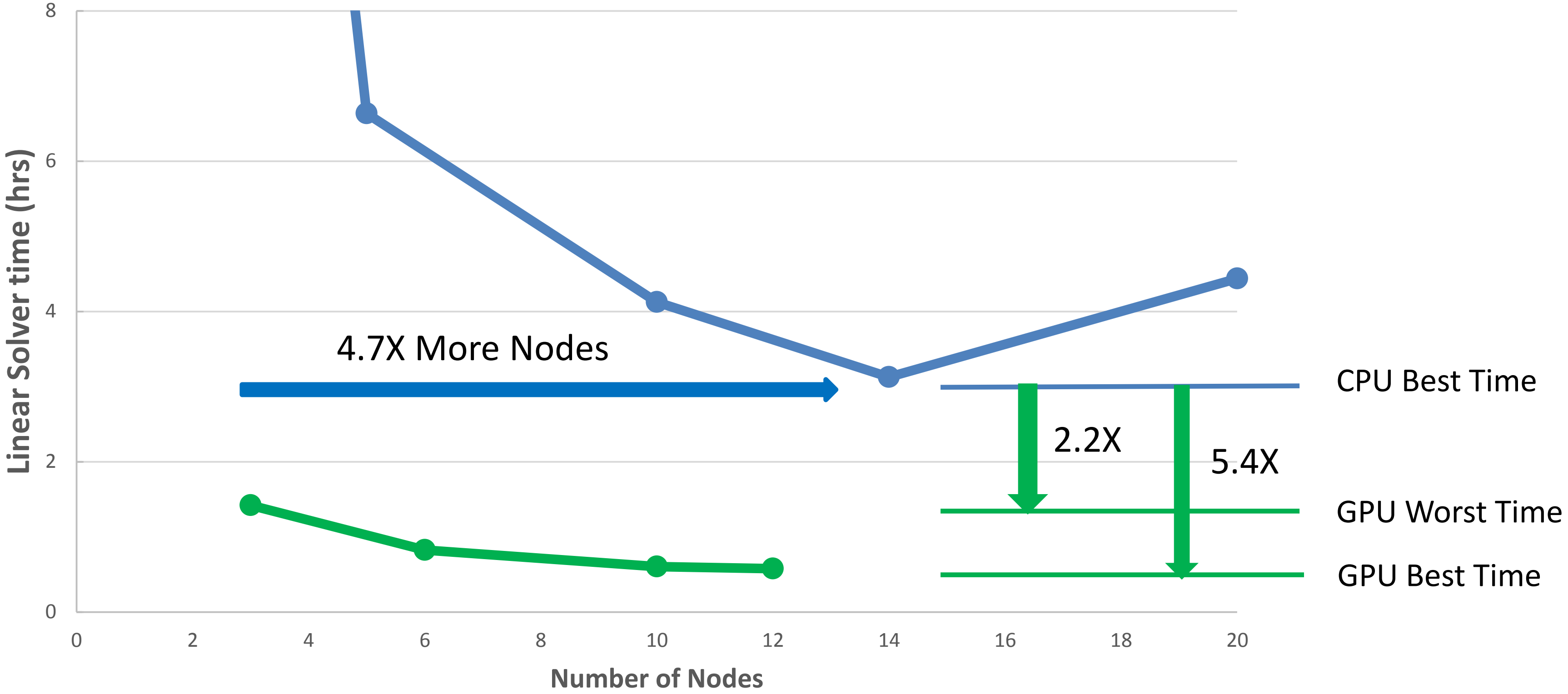


26X Linear Solver speedup

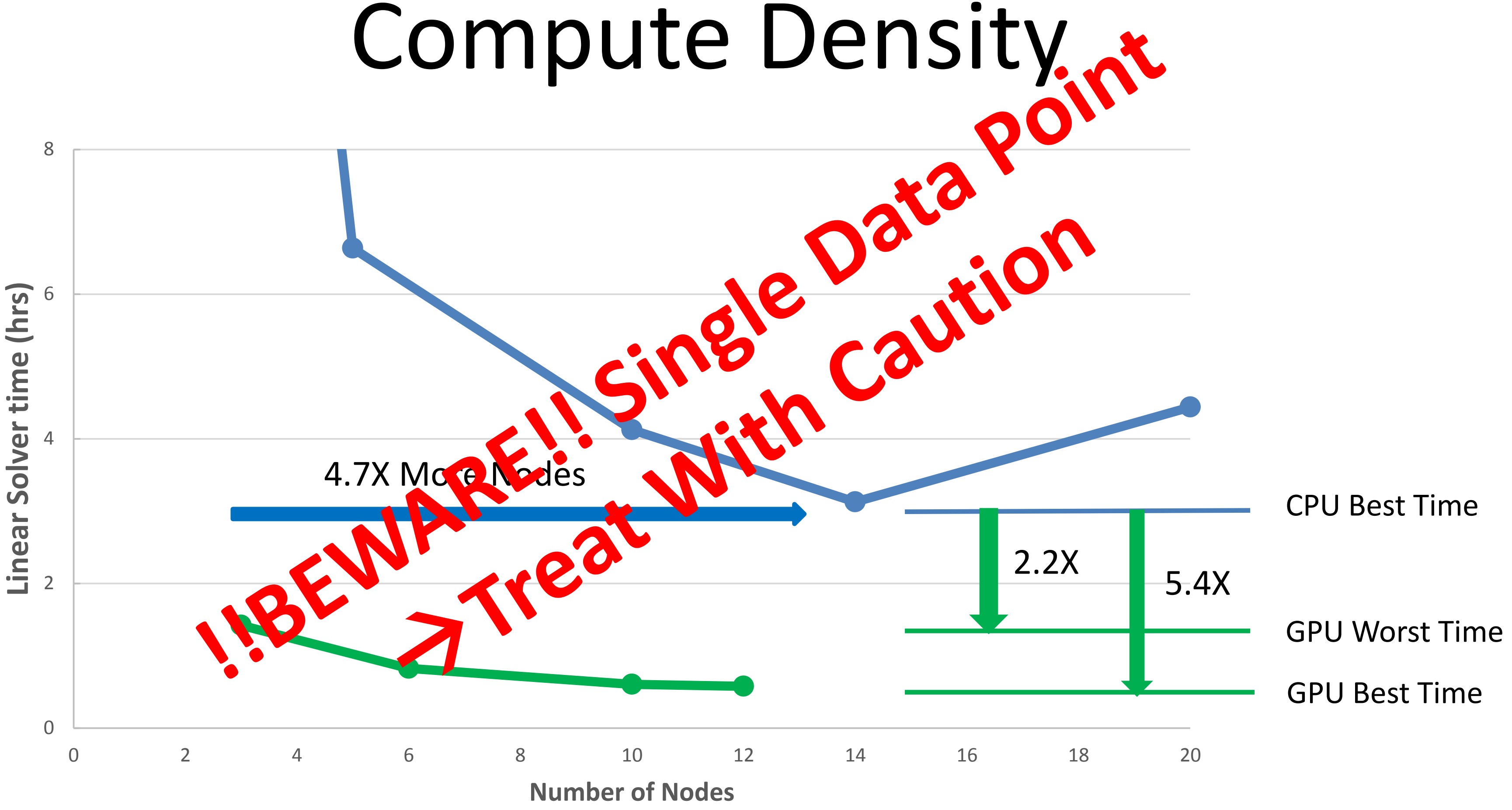
THERM_XL(16M): Strong Scaling



Compute Density



Compute Density



!!BEWARE!! Single Data Point

→ Treat With Caution

Next Steps

- Commercialize current solution
- Lessons learnt →CPU Solver
- Cluster hardware implications?

- Linear Solver is not enough →extend GPU
  - Wells too small & too complicated, remain on CPU
  - Reservoir
    - Jacobian construction
    - Property calculation

- Requirements:
  - Single code base: OpenACC?, Custom?
  - Overlapping rework

Thanks and Acknowledgements

Many thanks to Jonathan Cohen, Julian Demouth, Patrice Castonguay,
Justin Luitjens, Ken Hester & Doug Holt

Questions?