Taming the Many-core Beast
View from Industry on Impact of New Architectures on Algorithm Design

Tom Jönsthövel
Houston Technology Center
Simulators High Performance Computing Architect
What is Schlumberger?

- Leading provider of services to oil and gas industry
  - Seismic exploration
  - Drilling
  - Software (simulation/analytics/embedded)
- 90,000 employees, 140 nationalities, working in 85 countries
- Strong emphasis on developing new technology and innovation
Why Reservoir Simulation?

- Simulation to optimize production
  - Where to drill
  - How many wells
  - How long to produce
  - Where/when inject fluids or gas
- Simulation to reduce risks on million dollar investments
- Finite volume discretization
- Tens of millions of cells
- Structured but with
  - local grid refinement
  - faults
  - pinch-outs
- Unstructured
- High aspect ratios

- Simple vertical wells
- Deviated, horizontal, branched wells
- Pushing up to tens of thousands
- Conventional or multi-segmented
Formulation Porous Media Flow

\[
\frac{\partial}{\partial t} \left( \phi \sum_{\alpha} x_{\alpha i} b_{\alpha} S_{\alpha} \right) = \nabla \cdot \left( \sum_{\alpha} x_{\alpha i} b_{\alpha} \lambda_{\alpha} (\nabla p_{\alpha} - \rho_{\alpha} g \nabla d) \right) + q_{i}, \quad i = 1 \ldots n_{c}
\]

Darcy's law

\[
\sum_{\alpha} S_{\alpha} = 1
\]

Volume constraint

\[
\sum_{i} x_{\alpha i} = 1, \quad \alpha = 1 \ldots n_{\text{phases}}
\]

mass constraint

\[
f_{\alpha i} = f_{\beta i}, \quad \alpha \neq \beta, \quad i = 1 \ldots n_{c}
\]

mass equilibrium between phases
Simulation time

- Set of non linear of equations at given time

  - Newton Raphson
    - Start with initial solution non linear equations
    - Linearize non linear equations
    - Check convergence material balance error
    - **Update and solve linear equations**
      - 40 – 80% runtime
    - Scale and update solution non linear equations
    - Check for convergence solution updates
What does the user want?

• Emphasis on strong scalability

• Typical workflow
  • Build computer model (including mesh) from seismic information and geological information
  • Use history from produced fields to benchmark new model
  • Devise strategy for production

• Need to go faster for fixed problem size – how to achieve this?
What are the CPU hardware trends?

Introduction

AVX

Threading
Vectorization
Algorithmic Intensity

https://www.karlrupp.net/2013/06/cpu-gpu-and-mic-hardware-characteristics-over-time/
What are the constraints?

- Arithmetic Intensity
- Memory bound
- FLOPS

- Compute bound
- Vectorization
- Efficiency algorithm
What are the constraints and requirements?

Scalability
- Threading
- Communication avoiding

Performance
- Vectorization
- Algorithmic Intensity

Robustness
- Coarse grid correction
- Resolve local effects
- Interplay with Newton

Implementation
- Non intrusive to code base
- Maintainable

(more than ever) algorithms should follow hardware developments
Linear system overview

- **Dimension**
  - 1950s: $10^2$
  - 2014: $10^{10}$

- **Sparsity**

- **Block structure**
  - IMPES: Scalar
  - FIMP: Constant Block
  - AIM: Variable Block

- **Reservoir/wells**
  - $\begin{pmatrix} A_{rr} & A_{rw} \\ A_{wr} & A_{ww} \end{pmatrix}$

- **Conservation**

- **Variable line-up**

- **Heterogeneity**

- **Symmetry/directionality**

- **Diagonal dominance**
Linear solver

Linear system

Elliptic (pressure)

Hyperbolic (local effects)

x = ?

Right Hand Side
Scalability Pressure Solver
Robustness Resolution Local Effects

Inherently serial

ordering determines robustness
Scalability and Robustness Linear Solver

Local Effects

- Robustness suffers from increasing number of subdomains (Jacobi-effect)
- No overlap: amount of communication invariant under number subdomains

Pressure

- Increased amount of communication setup/solve with increasing number of subdomains
- Robustness invariant under number of subdomains

FGMRES

- Increased communication innerproducts (and norms) with increasing number of subdomains
Communication avoiding: reorder algorithms

- Iterative method to solve linear system
- Building blocks:
  - Sparse matrix-vector multiplication
  - **Innerproducts**
  - Sparse matrix-matrix multiplication
  - Dense matrix operation
- Fixed order of operations
- Scaling limited by reduction operation
Communication avoiding: reorder algorithms

Communication avoiding: NG-AMG

- AMG is most efficient solver for pressure ($O(N)$)
- Construct hierarchy of coarser problems
- Increase density at each level
- Fix sparsity pattern coarse matrices to reduce communication
- Post process matrices → remove non-zero off diagonal entries with smallest magnitude
- Lump reduced entries to main diagonal
Preserve robustness

Communication avoiding: NG-AMG

Wobbes (TU Delft), intern at AbTC in 2014
Comm avoiding / Threading / Algorithmic Intensity: Multiscale

Reduction in communication
Computation components amenable to threading
Higher arithmetic intensity

Localized calculations
Computation components amenable to threading

Outer loop

\[ x^0 = p^n \]
Update fine-scale \( A^\nu \) and \( b^\nu \) for pressure
Compute basis functions \( P^\nu \)
Solve \( A^\nu x^{\nu+1} = b^\nu \) with GMRES and MS & ILU0 preconditioner
Converge?
\[ p^{n+1} = x^{\nu+1} \]

\[ x^0 = s^n \]
Update fine-scale \( T^\nu \) and \( d^\nu \) for transport
Solve \( T^\nu x^{\nu+1} = d^\nu \) with Schwarz overlapping method
Converge?
\[ s^{n+1} = x^{\nu+1} \]
Comm avoiding / Threading / Algorithmic Intensity: Multiscale

Multi-scale vs AMG Pressure solve
Costs Building Blocks

- Smoothing high-frequency errors
- Construction matrices on coarser levels
- Construction restriction, prolongation

Computational cost
Threading: ILU

- Increase robustness by operating on original matrix
- Reduction in iterations increases performance and scalability

\[ x = x \]

Block Jacobi

Original matrix

MPI 1

MPI 2

MPI 3

Thread 0 - N
Threading: Multicoloring ILU

- Level scheduling: parallelism based on LU factorization → increasing problem size leads to decreasing level of parallelism
- In BILU(k), $|A|^{p+1} \sim |LU(p)|$
- High degree of parallelism exposed by BILU(k) factorization of $A_\pi = \pi A \pi^{-1}$ based on multicolored ($\pi$) of $|A|^{p+1}$

(39 levels, 50 iterations)  (9 colors, 100 iterations)
Threaded ILU: Graph problem

- Compute ILU elimination tree
- Identify independent sets
- Dynamic dispatching of tasks
- Number of iterations invariant under number of threads
Recap

• Massive parallelism has arrived and here to stay
• Industry standard solver is very robust but does not scale naturally on many-core architectures
• Alternatives available but robustness is key
• MPI-X/threading + vectorization are essential for CPU based code
• Not single component that needs overhaul (reducing communication, increasing parallelism, Newton-level, code modernization)