High Performance Computing Algorithm Design and Novel Computing Environments

Eero Vainikko

Institute of Computer Science, Distributed Systems Research Group

CDC (*Centre for Dependable Computing*)
Follow-up Workshop
Tallinn, 21-22 January 2008

Outline

1. Part I Introduction: target problems
   - Systems of linear equations
   - Examples
2. Solving Large Systems of Lin. Eq.
   - Direct or Iterative Methods?
   - Domain Decomposition
3. DOUG
   - DOUG – fast parallel black box solver
   - DOUG & aggregation
   - DOUG on extended architectures
4. Part II Novel Computing Environments
   - Grid
   - Desktop Grid Solutions
   - Minimum Intrusion Grid
   - Friend-to-friend Computing

Target problems

Solve the system of linear equations

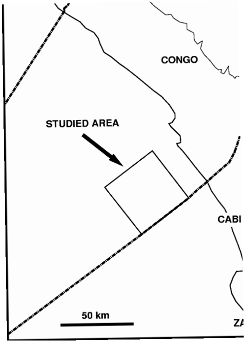
$$A\mathbf{x} = \mathbf{b}$$

where the matrix A is:

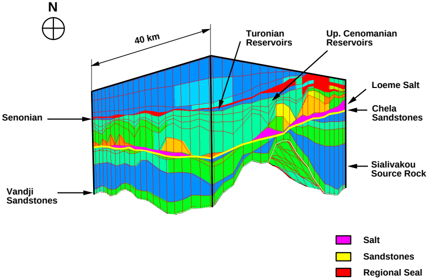
- sparse,
- large,
- with highly varying coefficients (for example, $|a_{ij}| \in [10^{-6}, 10^{6}]$)

Lithology example

Sedimentary basin simulation



Position of the studied area

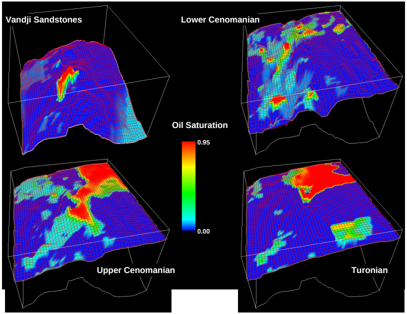


Lithology of the studied 3D-block

(Taken From F.Schneider Et Al. Oil & Gas Science And Technology 55 (1), 2000)

Congo Offshore oil reservoir simulation

Numerical Results (Congo Offshore)

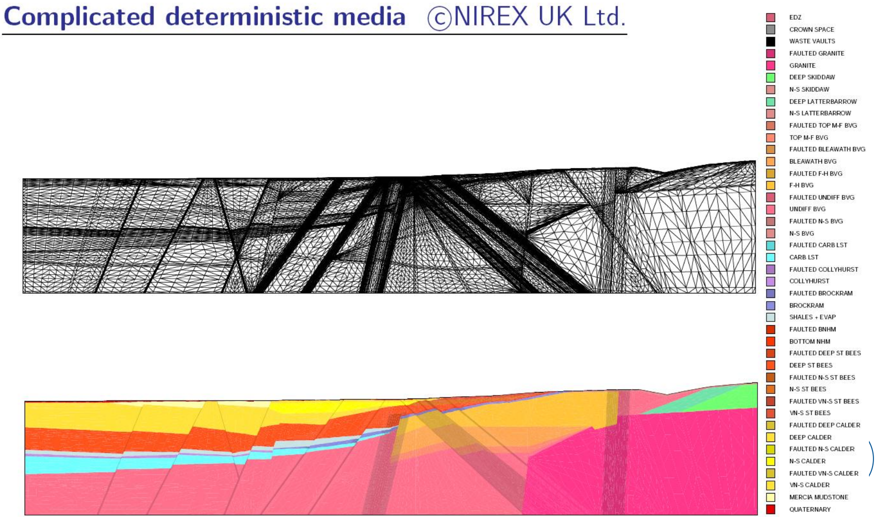


Computed oil saturation for the potential reservoir levels at present day

(Taken from {F.Schneider et al. Oil & Gas Science and Technology 55 (1), 2000))

Sellafield

Groundwater flow, Test case "Sellafield"



Direct methods

UMFPACK, SuperLU, MUMPS, Pardiso

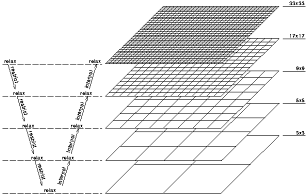
1. Analysing the sparsity structure to introduce less fill-in via re-ordering
2. Factorisation step
3. Solving step

- Roughly 100(1)-10(2)-1(3) time factor
  2D - often still OK,
  3D - the first step becomes too expensive, and still, the fill-in starts dominating.

Iterative methods

Richardson's, Krylov subspace methods, MultiGrid

- Richardson's type iterations
- Krylov subspace methods
- Geometric multigrid
- Algebraic multigrid
  - f-c colouring
  - aggregation-based



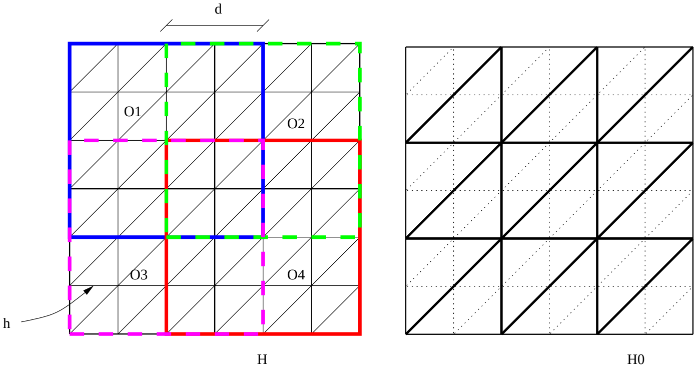
Domain Decomposition

Additive Schwarz methods

- Non-overlapping methods

substructuring methods, additive average methods and others.

- Overlapping methods



Domain Decomposition on Unstructured Grids

Domain Decomposition on Unstructured Grids

DOUG (*University of Bath, University of Tartu*) 1997 - 2008

I.G.Graham, M.Haggers, R. Scheichl, L.Stals, E.Vainikko, M.Tehver, K. Skaburskas, O. Batrašev, C. Pöcher,

Parallel implementation based on:

- **MPI**
- **UMFPACK**
- **METIS**
- **BLAS**
- Automatic parallelisation and load-balancing
- Systems of PDEs
- 2D & 3D problems

- 2-level Additiivne Schwarz method
- 2-level partitioning
- Automatic Coarse Grid generation
- Adaptive refinement of the coarse grid

DOUG Strategies

- Iterative solver based on Krylov subspace methods
  PCG, MINRES, BICGSTAB, 2-layered **FPGMRES** with left or right preconditioning.
- Non-blocking communication where at all possible
  Ax-operation: $\mathbf{y} := A\mathbf{x}$ – :-)
  Dot-product: $(\mathbf{x}, \mathbf{y})$ – :-(
- Preconditioner based on Domain Decomposition with 2-level solvers
  Applying the preconditioner P: solve for $\mathbf{z} : P\mathbf{z} = \mathbf{r}$. :?
- Subproblems are solved with a direct, sparse multifrontal solver (UMFPACK)

Aggregation-based Domain Decomposition methods

- Have been analysed only upto some extent
- **We are:** making use of strong connections
- Second aggregation for creating subdomains, or
- using rough aggregation before graph partitioner
- Major progress - development of the theory: **sharper bounds**

1. R. Scheichl and E. Vainikko, Robust Aggregation-Based Coarsening for Additive Schwarz in the Case of Highly Variable Coefficients, Proceddings of the European Conference on Computational Fluid Dynamics, ECCOMAS CFD 2006 (P. Wesseling, E. ONate, J. Periaux, Eds.), TU Delft, 2006.
2. R. Scheichl and E. Vainikko, Additive Schwarz and Aggregation-Based Coarsening for Elliptic Problems with Highly Variable Coefficients, Computing, 2007, 80(4), pp 319-343.

Aggregation

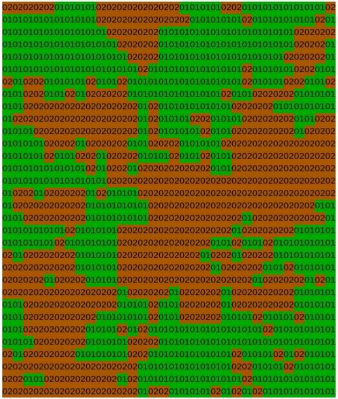
Key issues:

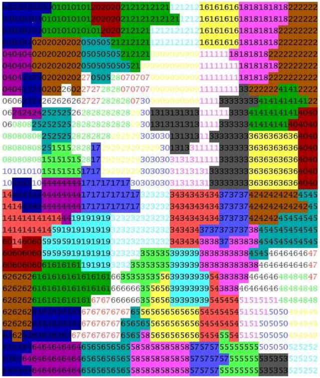
- how to find good aggregates?
- Smoothing step(s) for restriction and interpolation operators

Four (often conflicting) aims:

- follow adequatly underlying physial properties of the domain
- try to retain optimal aggregate size
- keep the shape of aggregates regular
- reduce communication => develop aggregates with smooth boundaries

Coefficient jump resolving property





Algorithm: Shape-preserving aggregation

Input: Matrix A, aggregation radius r, strong connection threashold α.
Output: Aggregate number for each node in the domain.

1. Scale A to unit diagonal matrix (all ones on diagonal)
2. Find the set S of matrix A strong connectons: $S = \cup_{i=1}^{n} S_i$, where $S_i \equiv \{j \neq i : |a_{ij}| \geq \alpha \max_{k \neq i} |a_{ik}|,$ unscale A; aggr_num:=0;
3. aggr_num:=aggr_num+1;

Choose a seednode x from G or if $G = \emptyset$, choose the first nonaggregated node x;
level:=0

4. If (level<r)
   - Add recursively all strongly connected non-aggregated neighbours to the aggregate aggr_num with level+1 and perform smoothing step on each level

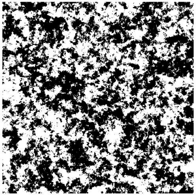
   elseif (level<$2r$)
   - Find layer(level+1)...layer($2r$).

   endif
5. On the longest layer(i), i=$r+1,...,2r$ add node(s) with shortest distance from x to the set G and goto step 3.

DOUG

Numerical results

Clipped random fields



$n = 256 \times 256$

Time in seconds (# iterations) for different methods:

jumpsize	Aggregation	AMG	UMFPACK
$1.5 * 10^1$	2.12 (24)	1.35 (14)	1.85
$2.2 * 10^2$	2.14 (27)	2.27 (27)	1.70
$3.3 * 10^3$	2.34 (29)	3.31 (40)	1.33
$4.9 * 10^4$	2.41 (26)	6.23 (77)	4.88

AMG - Aggregation-type Algebraic Multigrid [Bastian] (no smoothing - piecewise constant prolongation)

UMFPACK - Sparse direct solver

DOUG on Grid

- Is it possible to use GRID in it's full power for one (huge) linear system solution?

PROBLEM:

- Dynamic nature of GRID *versus*:
  - good parallel solvers need synchronisation steps :-(
  - no fault tolerance in mainstream MPI implementations :-(
- => A) need for methods that do not need regular synchronisation
- => B) Need for fault-tolerant communication libraries

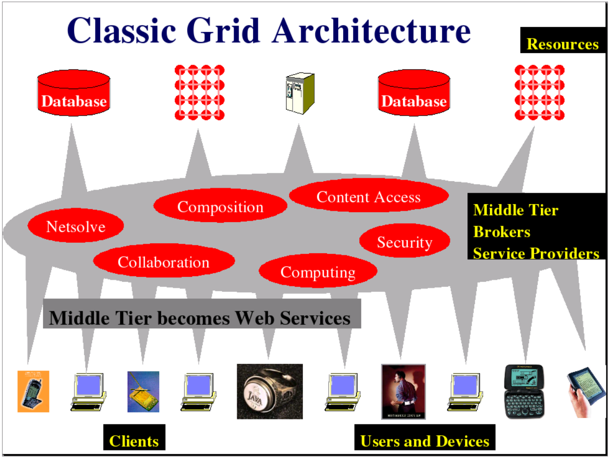
Possible solutions

- **A) Algorithms**
  - Asynchronous DD methods (based on Richardson's type iteration methods)
  - Possible asynchronous Krylov subspace methods
    – Using flexible GMRES
    – some synchronisation still needed

- **B) Fault tolerance**
  - Need for fault tolerant libraries
    – e.g. FT-MPI, OpenMPI
    – need for a standard
    MPI standard says currently: "*Fault tolerance is a property of an application, not the MPI Library*"
    – no FT MPI library available for GRID.

The Grid Vision

What was promised



- As easy as power-grid
- Everywhere
- Accessible to everybody
- Access to any desired resource
    - CPU cycles
    - Storage
    - Special devices

The Grid - what we've got so far

Batch processor

- Batch processing
  - With global scale, though
  - Parallel jobs still available, within a single cluster
  - You really need to know the resource you are running on
- Grid = merely web-services in another wrapping!
- Storage – really unconfortable for user

Problems with existing models

(Not every problem in every middleware)

- Single point of failure
- Lack of scheduling
- Poor scalability
- No means for privacy
- No way for cycle stealing (scavenging)
- Firewall problem
- Requires very large installation
- No economy

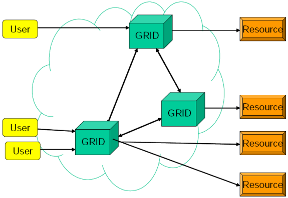
Desktop Grids

- Condor
- BOINC
  - Seti@home
  - Einstein@home
  - Proteins@home
  - Africa@home
  - SZTAKI@home
  - Folding@home
- ALCHEMI
- JNGI
- **Minimum Intrusion Grid**
- **Friend-to-friend Computing**

Minimum Intrusion Grid (MiG) Design

http://mig-1.imada.sdu.dk/MiG/index.html

- Non-intrusive
- Scalable
- Autonomous
- Anonymous
- Fault tolerance
- Firewall compliant
- Strong scheduling
  - Real-time access to CPU-s
- Allowing multiple jobs to share a CPU for low-intensity jobs
- Cooperative support



- BUT still, for DOUG we need (fast) communication between computing nodes!

Friend-to-friend Computing (F2F)

Called also: Spontaneous Desktop Grid

http://f2f.ulno.net

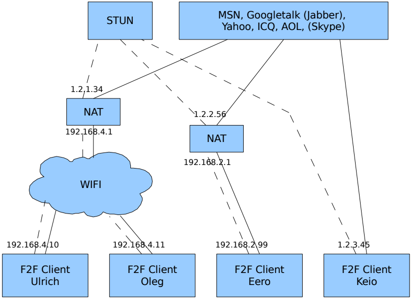
- Developed at the Institute of Computer Science, University of Tartu (U.Norbisrath and E.Vainikko)
- Computing environment based on spontaneous groups
- Instant Messaging friend groups used for triggering the Computing Grid Environment
- `Merging` ideas from
  - Peer-to-peer
  - High Performance Computing
  - Social networks in instant messaging
- Extremely easy concept for
  - authentication
  - authorization

  based on inherent trust between friends

F2F Architecture

Instant messaging protocols/solutions supported:

- MSN
- GoogleTalk (Jabber)
- Yahoo
- ICQ
- AOL
- (Skype)



Overview