Scalable sparse matrix solvers on supercomputers

Hans De Sterck

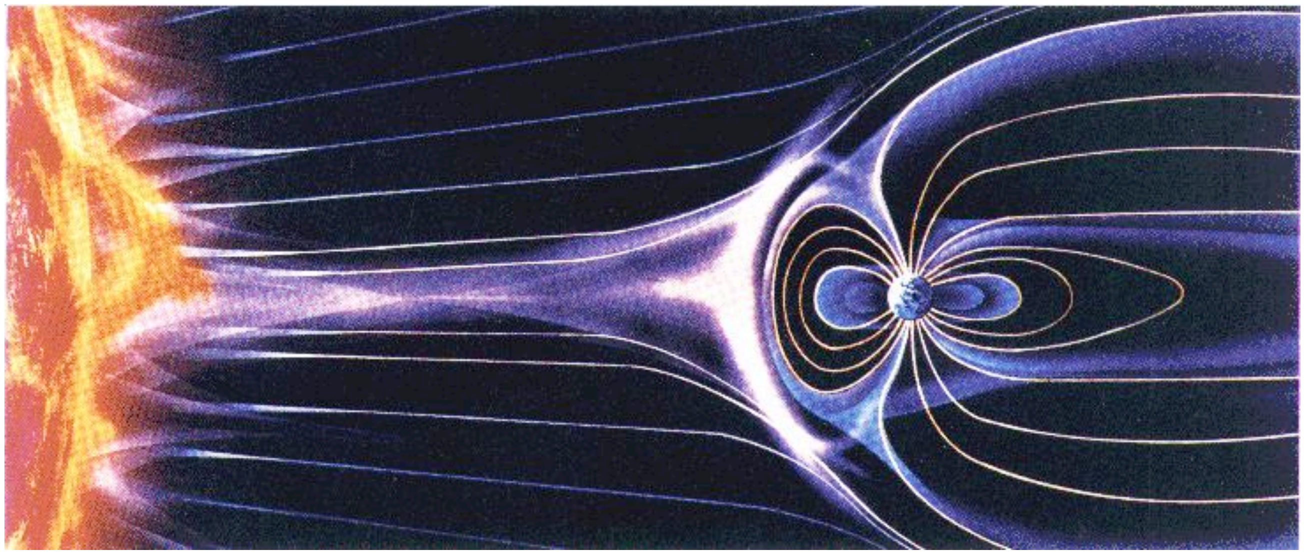
Department of Applied Mathematics
University of Waterloo
hdesterck@uwaterloo.ca

SONAD, Waterloo, 29 April 2005

Sparse matrix solvers

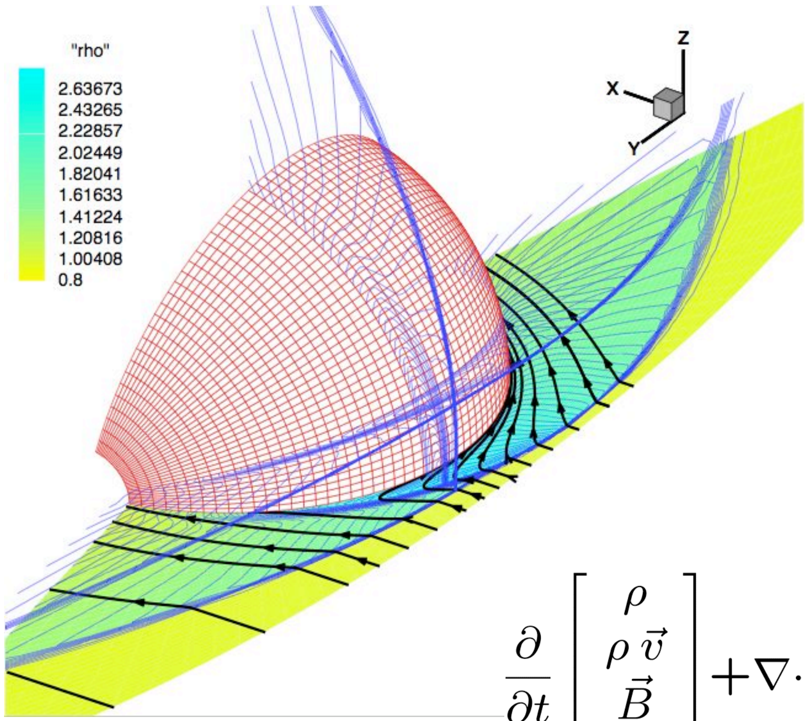
- solve $A x = b$
- $A \in R^{n \times n}$, $b \in R^n$, $x \in R^n$
- A large: n = millions, billions, ...
- A sparse: PDE discretization on a grid
- we want efficient, accurate and robust solvers

example 1: space weather prediction



- solar wind plasma (ionized gas) flows from sun to earth
- simulation goal: predict when solar eruptions reach earth (~ 4 days)

example 1: space weather prediction



- discretize Magnetohydro-dynamics (MHD) equations
- implicit time discretization

- A x = b
- 8 million unknowns

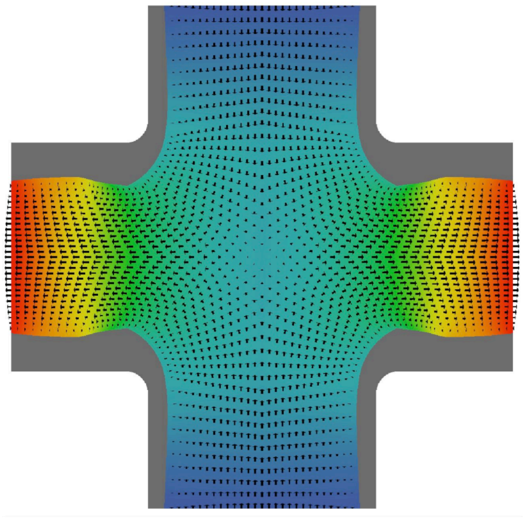
$$\frac{\partial}{\partial t}\begin{bmatrix}\rho\\ \rho\,\vec{v}\\ \vec{B}\\ e\end{bmatrix}+\nabla\cdot\begin{bmatrix}\rho\,\vec{v}\\ \rho\,\vec{v}\vec{v}+I\left(p+\vec{B}\cdot\vec{B}\,/2\right)-\vec{B}\vec{B}\\ \vec{v}\vec{B}-\vec{B}\vec{v}\\ \left(e+p+\vec{B}\cdot\vec{B}\,/2\right)\vec{v}-\left(\vec{v}\cdot\vec{B}\right)\vec{B}\end{bmatrix}=0$$

University of Waterloo

example 1: space weather prediction

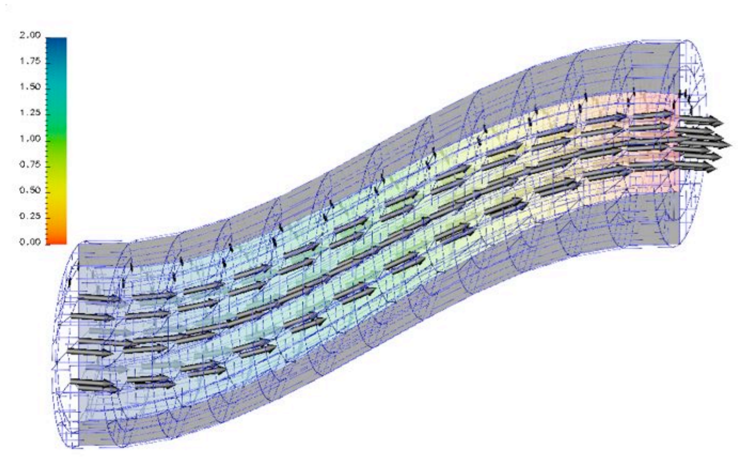
- problem with existing methods: scalability
  - 3D simulation
  - suppose: increase resolution by 2
  - $n \Rightarrow 8\ n$
  - if the methods scale well, $t_{exec} \Rightarrow 8\ t_{exec}$ (linear or $O(n)$ scaling)
  - existing methods often scale as $O(n^2)$ or worse: $t_{exec} \Rightarrow 64\ t_{exec}$
  - high problem resolutions quickly get out of reach
- need scalable methods!

example 2: biomedical flows



$$-\nabla p + \Delta \vec{v} = 0 \qquad (1)$$

$$\nabla \cdot \vec{v} = 0 \qquad (2)$$



- blood flow in compliant vessels
- incompressible Navier-Stokes + elasticity equations
- 3D: large systems, need scalable solvers on parallel computers

(simulation results courtesy Jeff Heys)

University of Waterloo

Overview of presentation

(A) introduction - scalable solvers for PDEs

(B) multigrid iterative solvers

(C) scalable parallel algebraic multigrid solvers

(D) future work

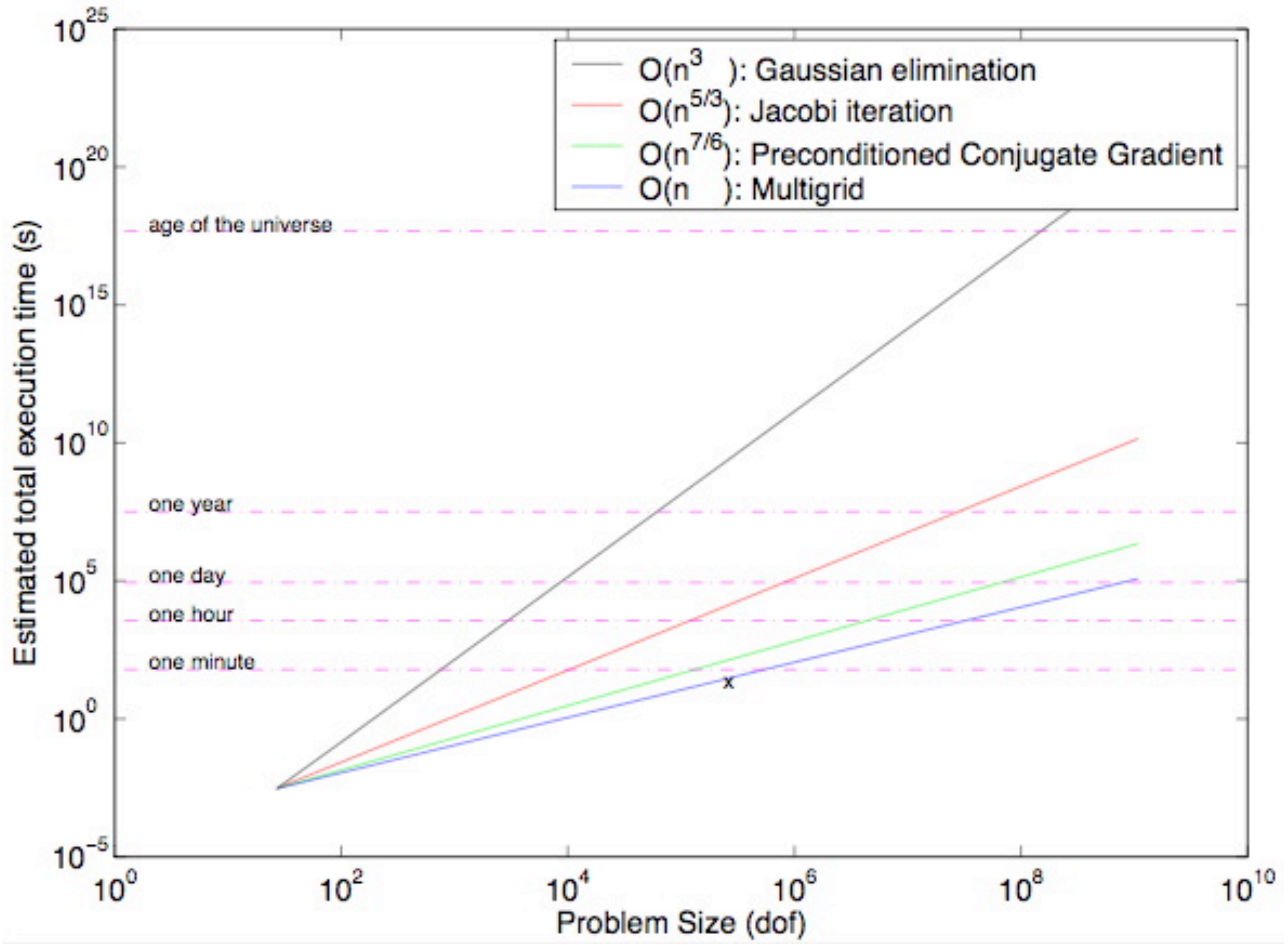
(A) introduction - scalable solvers for PDEs

two types of scalability

(1) algorithmic scalability

- on a single processor, using serial algorithm
- optimal algorithmic scalability:
  - number of operations $op_n = O(n)$
  - execution time $t_{exec} = O(n)$
    (both increase linearly with n)

(1) algorithmic scalability



SONAD, Waterloo, 29 April 2005
hdesterck@uwaterloo.ca

two types of scalability

(2) parallel scalability

- on a parallel computer, using parallelized algorithm
- communication overhead, may not scale well
- parallelization often attempts to minimize communication
- parallelized algorithm may be less scalable than serial algorithm (even with zero communication cost)

(2) parallel scalability

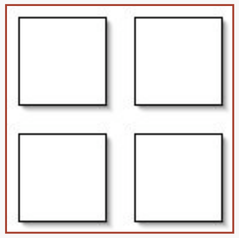
- consider constant problem size n_{proc} per processor

- p processors
- total problem size $n = p\ n_{proc}$

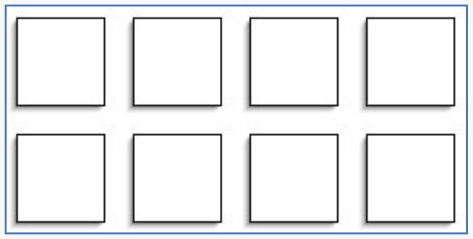
- double problem size, $2\ n = 2\ p\ n_{proc}$
- double number of processors, $2\ p$

- optimal scalability:

  t_{exec} = constant as n increases


p procs

$t_{exec} = t_0$


2 p procs

$t_{exec} = t_0$

University of Waterloo

SONAD, Waterloo, 29 April 2005
hdesterck@uwaterloo.ca

(B) multigrid iterative solvers

- solve A x = b
- goal: O(n) complexity

- 1D model problem:

$$-u''(x) = f(x)$$
$$u(0) = 0$$
$$u(1) = 0$$

- discretization:

$$\frac{-u_{i-1}+2u_i-u_{i+1}}{h^2} = f_i$$

1D model problem

$$-u_{i-1} + 2u_i - u_{i+1} = h^2\, f_i$$

- A x = b

$$A = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

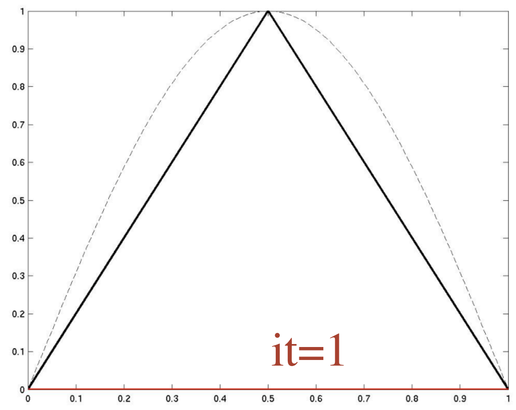
- first idea: use an iterative method (exploiting sparsity)
- for example: Gauss method (relaxation)

$$u_i^{new} = 1/2\,(u_{i-1}^{new} + u_{i+1}^{old} + h^2\, f_i)$$

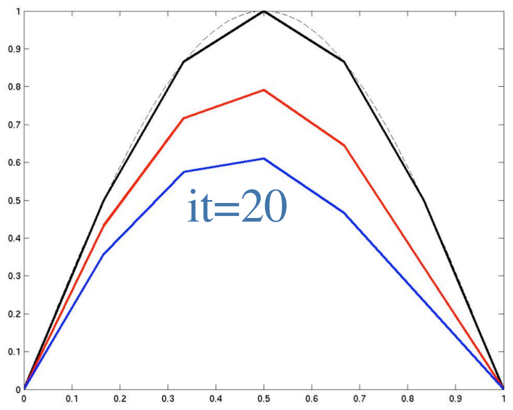
University of Waterloo

Gauss relaxation method: $-u''(x) = 0$

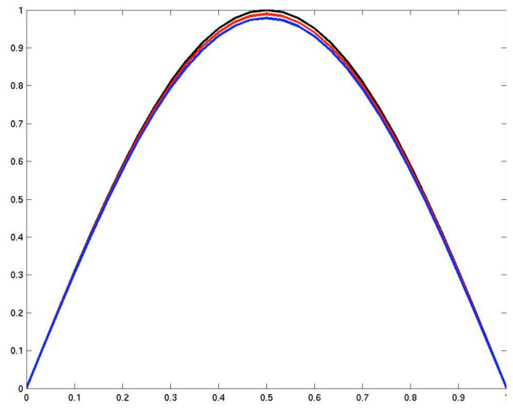
n=3



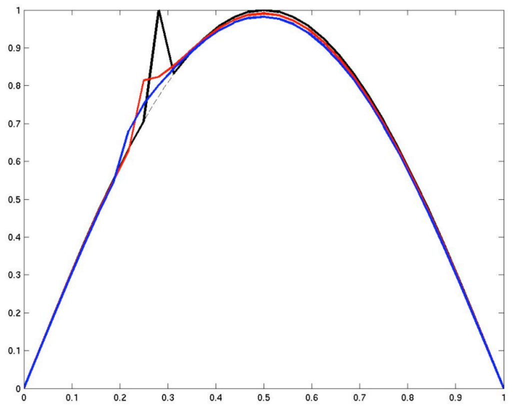
n=7



n=31



n=33

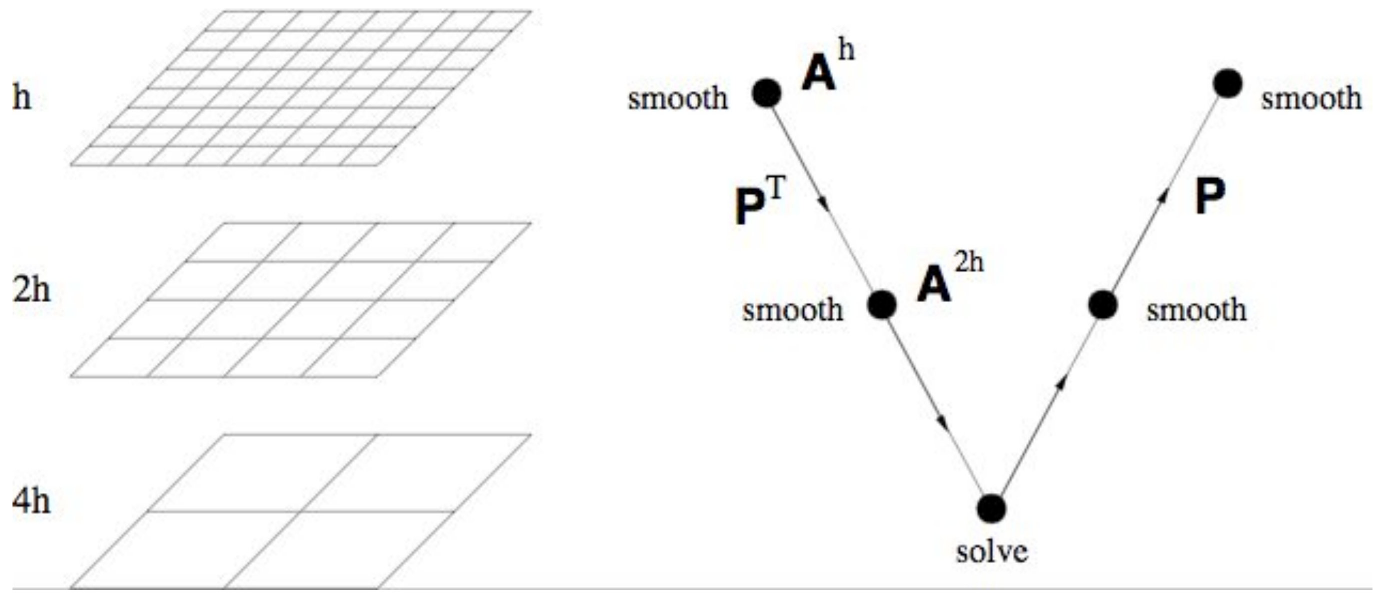


- need more iterations as n increases:

  not scalable
- but: for large n, high frequency errors disappear fast!
- note: low frequency for large n (fine grid), is high frequency for small n (coarse grid)
- second idea: use hierarchy of grid levels
  - correct high-frequency errors on fine grid
  - correct low-frequency errors on coarse grid

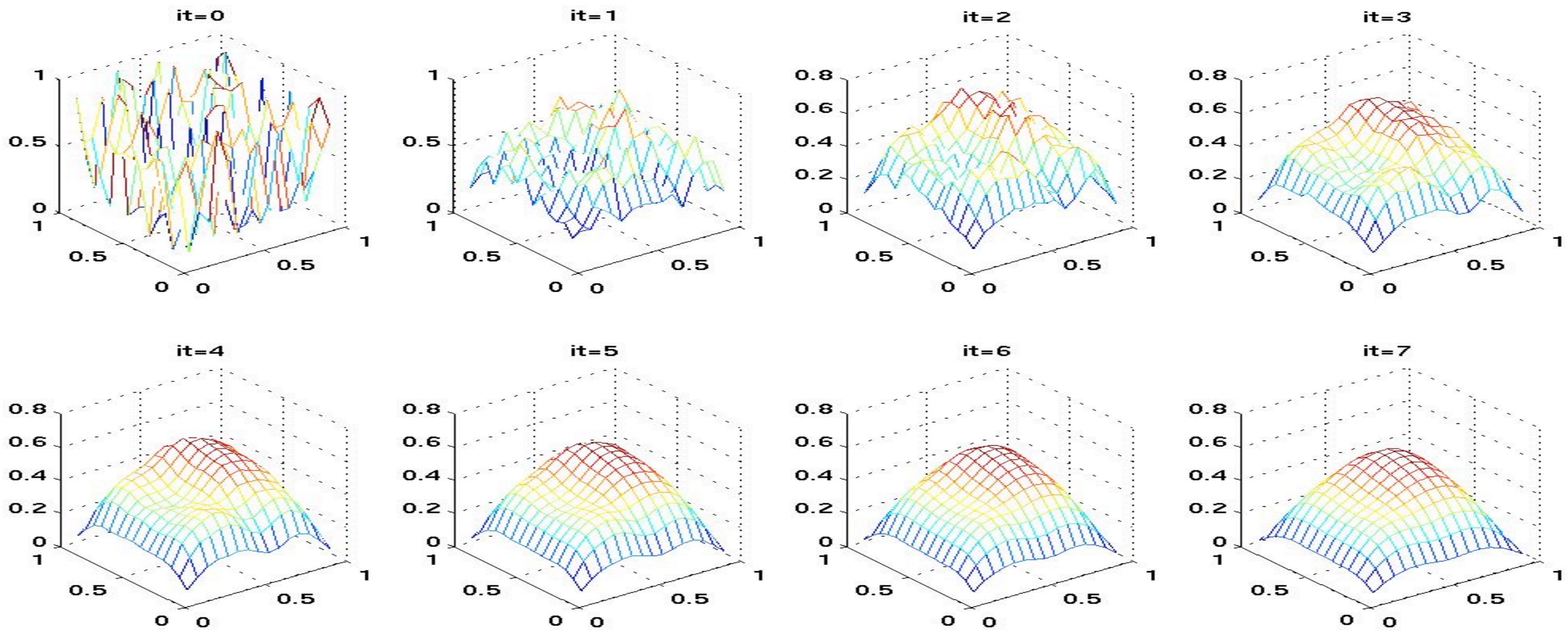
University of Waterloo

multigrid hierarchy: V-cycle



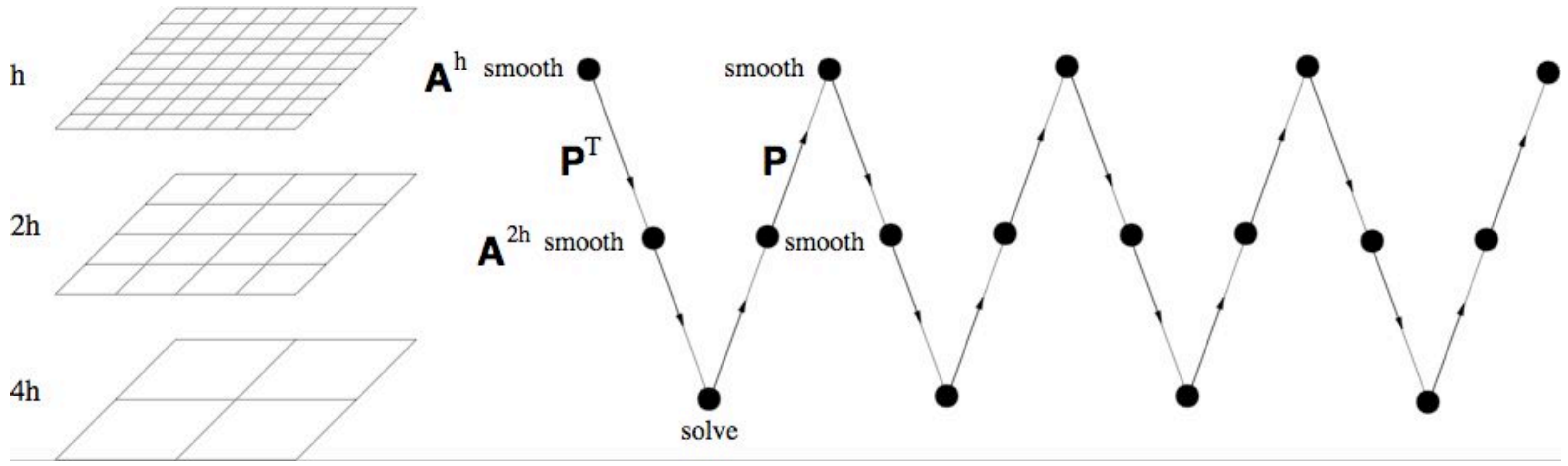
- multigrid V-cycle:
  - relax (=smooth) on various grids
  - transfer error using restriction (P^T) and interpolation (P)

2D model problem: $-u_{xx} - u_{yy} = f(x, y)$



- high-frequency error is removed by relaxation
- low-frequency-error needs to be removed by coarse-grid correction

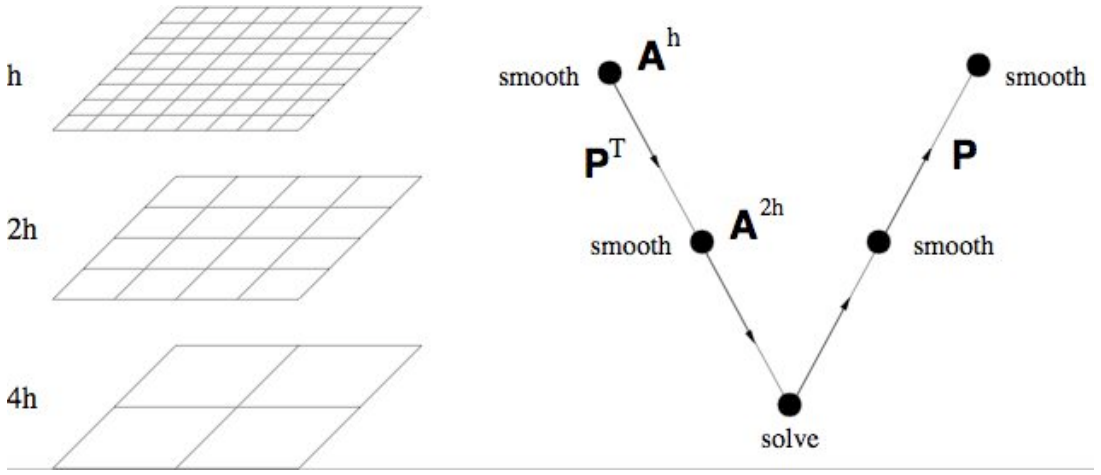
multigrid hierarchy: V-cycle



- in every V-cycle, error is reduced by convergence factor ρ
- repeat m V-cycles, error is reduced by ρ^m

- for large classes of elliptic PDEs, ρ is bounded away from 1 uniformly in n
- ‘deeper’ cycles have same ρ as ‘shallower’ cycles

amount of work per V-cycle



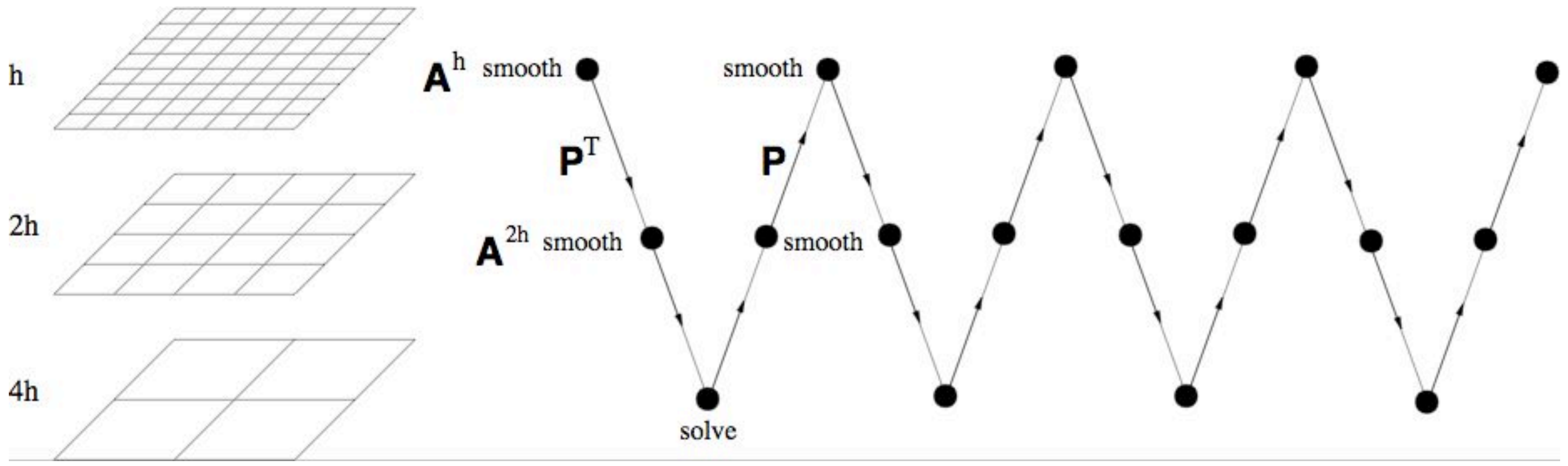
$$A^h = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & & & \\ -1 & 2 & -1 & & & \\ & -1 & 2 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 2 & -1 \\ & & & & -1 & 2 \end{bmatrix}$$

- work for 1 relaxation on fine grid is called a Work Unit (WU)
- 1 relaxation (or WU) is O(n) ! (number of rows in A^h)
- number of WUs in V-cycle:

  $2 \,.\, (1 + (1/2)^2 + (1/2)^4 + (1/2)^8 + \ldots + (1/2)^{2m})$ WUs
- geometric series: $\sum_{i=0}^{m} (1/4)^i < 1 / (1-(1/4)) = 4/3$
- total work per V-cycle < 8/3 WUs : O(n) !

University of Waterloo

scalable solver: O(n)



- scalable method: O(n)
  - work per V-cycle is O(n)
  - number of V-cycles required is independent of n
    - because ρ independent of n

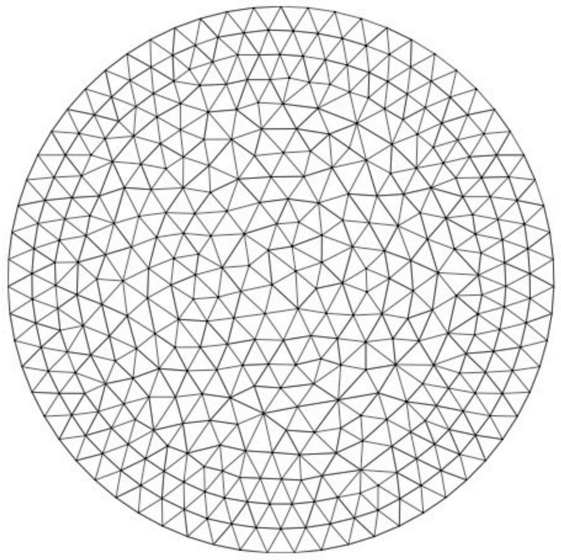
(C) scalable parallel algebraic multigrid solvers

- algebraic multigrid:
  - unstructured grid problems
  - matrix problems without grid

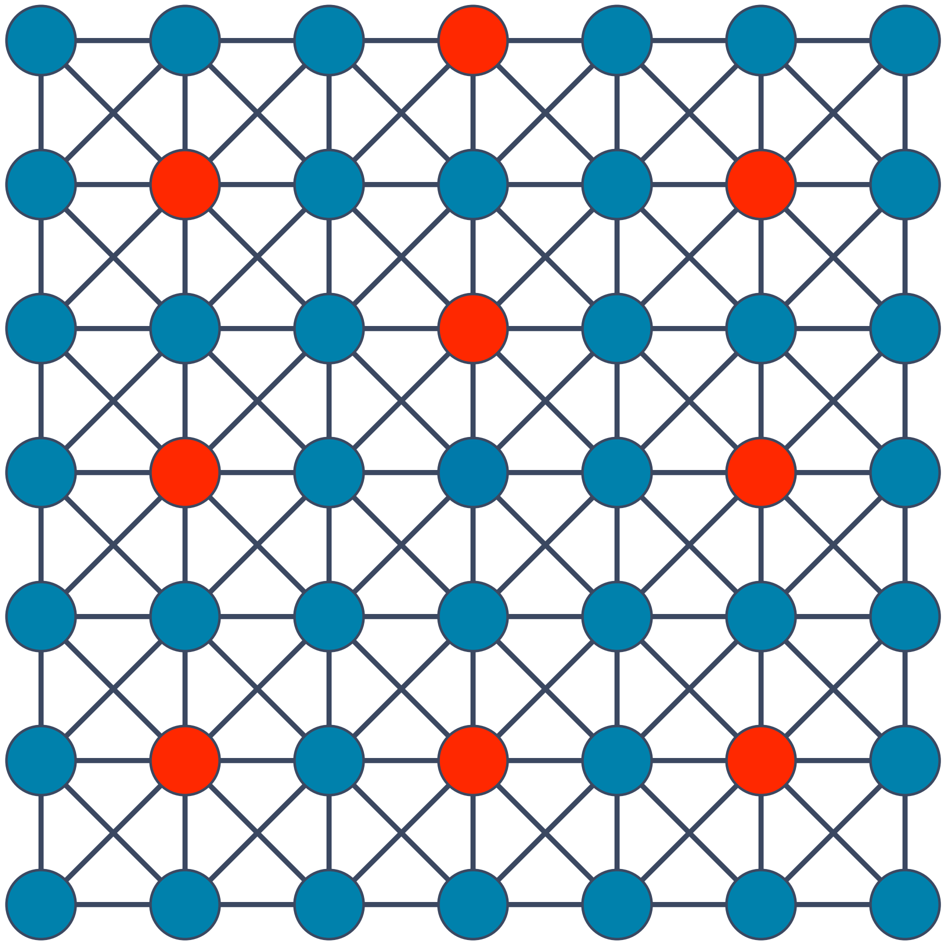
  automatically determine, only from information in matrix:

  - coarse ‘grids’
  - coarse grid operators A^h
  - interpolation operators P^h

[Brandt, McCormick, Ruge, Stueben]



AMG coarse and fine grids



- **select C-pts**
- **others points are F-pts**
- **F-pts interpolate from C-pts**

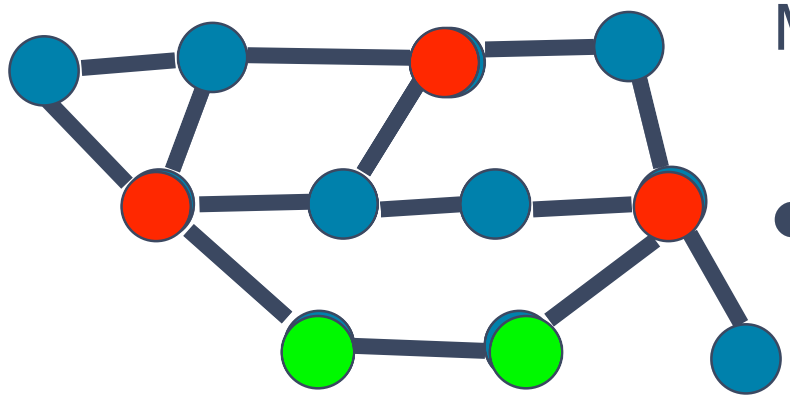
AMG coarsening and interpolation

- only large a_{ij}, ‘strong connections’ are important
- define strength matrix S:

$$A = \begin{bmatrix} x & x & x & & \\ & x & x & & x \\ & x & x & x & x \\ & x & x & x & \\ x & & & x & x \end{bmatrix} \qquad S = \begin{bmatrix} 1 & 1 & 0 & & \\ & 1 & 0 & & 1 \\ & 0 & 0 & 1 & 1 \\ & 1 & 0 & 1 & \\ 1 & & & 1 & 0 \end{bmatrix}$$

- consider the undirected graph of S
- apply parallel maximal independent set algorithms to graph(S) [Luby, 1986]

classical AMG coarsening and interpolation



(Ruge, Stueben, Cleary)

- (C1) Maximal Independent Set:

Independent: no two C-points are connected

Maximal: if one more C-point is added, the independence is lost

- (C2) All F-F connections require connections to a common C-point (for good interpolation)

- F-points have to be changed into C-points, to ensure (C2); (C1) is violated

more C-points, higher complexity

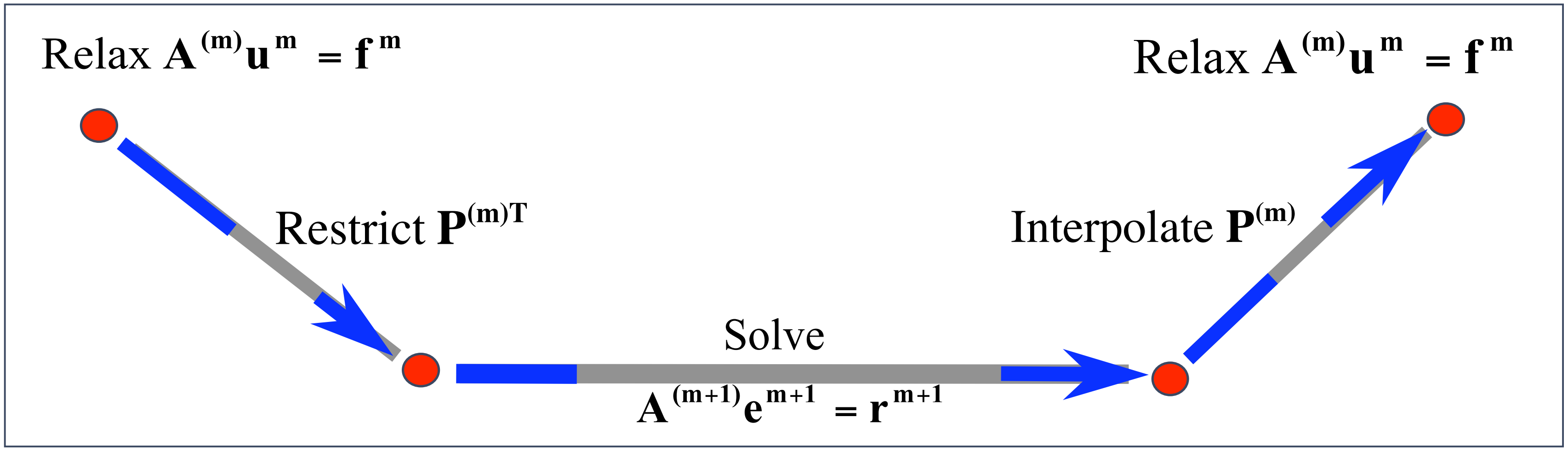
AMG building blocks

Setup Phase:

- Select coarse "grids"
- Define interpolation, $\mathbf{P}^{(m)}, \quad m = 1,2,\ldots$
- Define restriction and coarse-grid operators

$$\mathbf{R}^{(m)} = \mathbf{P}^{(m)T} \qquad \mathbf{A}^{(m+1)} = \mathbf{P}^{(m)T}\mathbf{A}^{(m)}\mathbf{P}^{(m)}$$

Solve Phase



Relax $\mathbf{A}^{(m)}\mathbf{u}^{m} = \mathbf{f}^{m}$
Relax $\mathbf{A}^{(m)}\mathbf{u}^{m} = \mathbf{f}^{m}$
Restrict $\mathbf{P}^{(m)T}$
Interpolate $\mathbf{P}^{(m)}$
Solve
$\mathbf{A}^{(m+1)}\mathbf{e}^{m+1} = \mathbf{r}^{m+1}$

AMG complexity - scalability

- Operator complexity $C_{op} = \dfrac{\sum_i \text{nonzeros}(A_i)}{\text{nonzeros}(A_0)}$
  - e.g., 3D, ideally: $C_{op} = 1 + 1/8 + 1/64 + \ldots < 8/7$
  - measure of memory use, and work in solve phase

- scalable algorithm:

O(n) operations per V-cycle (C_{op} bounded)

and

number of V-cycles independent of n

(ρ independent of n)

Classical coarsening: scalability results

- example: finite difference Laplacian, parallel CLJP coarsening algorithm
- 2D (5-point): near-optimal scalability (250^2 dof/proc)

Procs	C_{op}	t_{tot}	Iter
16	4.48	2.89	9
64	4.50	3.85	9
256	4.50	5.01	9

Classical coarsening: complexity growth in some cases

- 3D (7-point): complexity growth

dof	C_{op}	Iter
32^3	16.17	8
64^3	22.51	11

- increased memory use, long solution times, long setup times, loss of scalability

Classical coarsening: complexity growth in some cases

- 4D (9-point), 5D (11-point): complexity growth!!

	dof	C_{op}	Iter
4D	20^4	127.5	8
5D	9^5	256.9	5

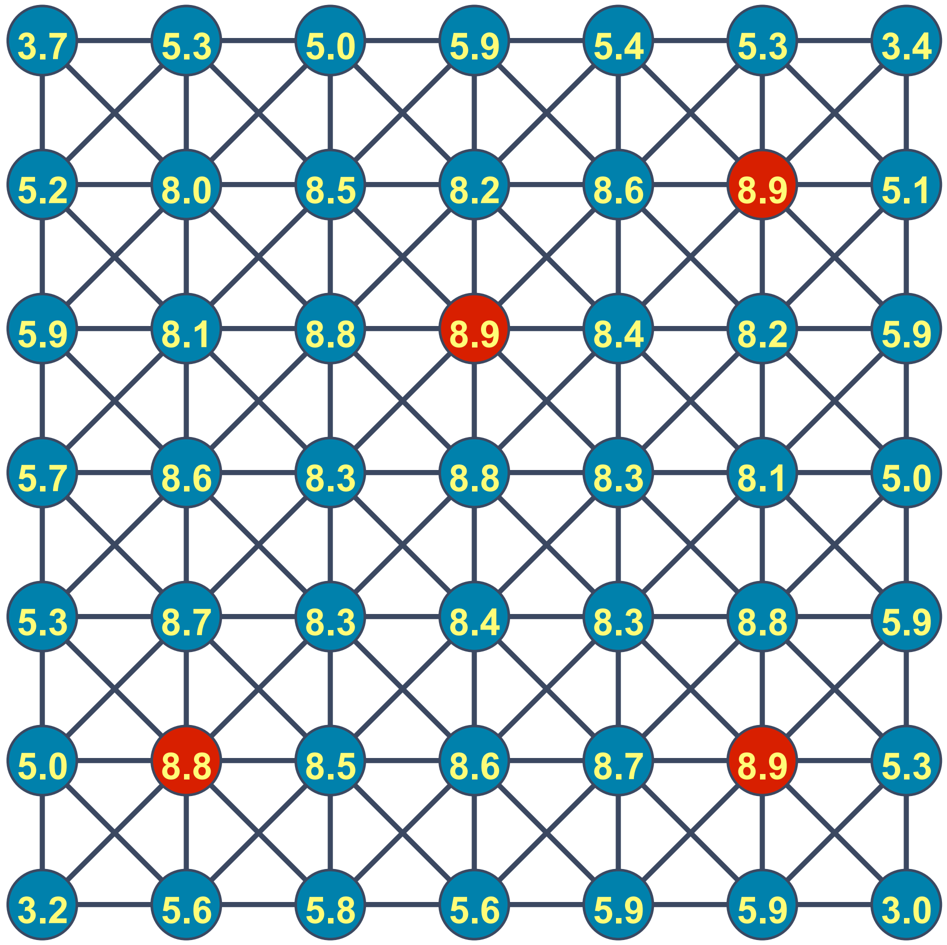
- excessive memory use

(results by Jeff Butler)

our approach to reduce complexity: PMIS coarsening (De Sterck, Yang, SIAM J. Matrix Analysis, 2004, submitted)

- Parallel Modified Independent Set (PMIS)
- do not enforce condition (C2)
- convergence acceleration using GMRES
- weighted independent set algorithm: points i that influence many equations (λ_i large), are good candidates for C-points
- add random number between 0 and 1 to λ_i to break ties
- parallel algorithm!

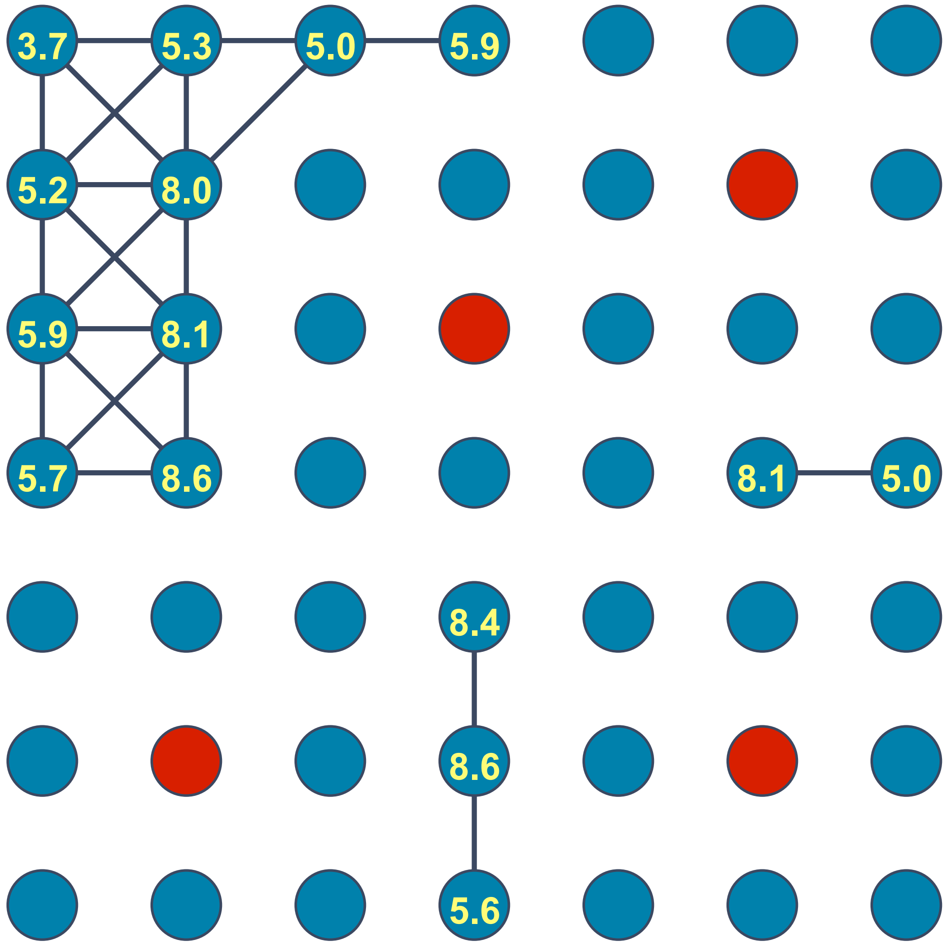
PMIS select 1



- **select C-pts with maximal measure locally**
- **make neighbor F-pts**
- **remove neighbor edges**

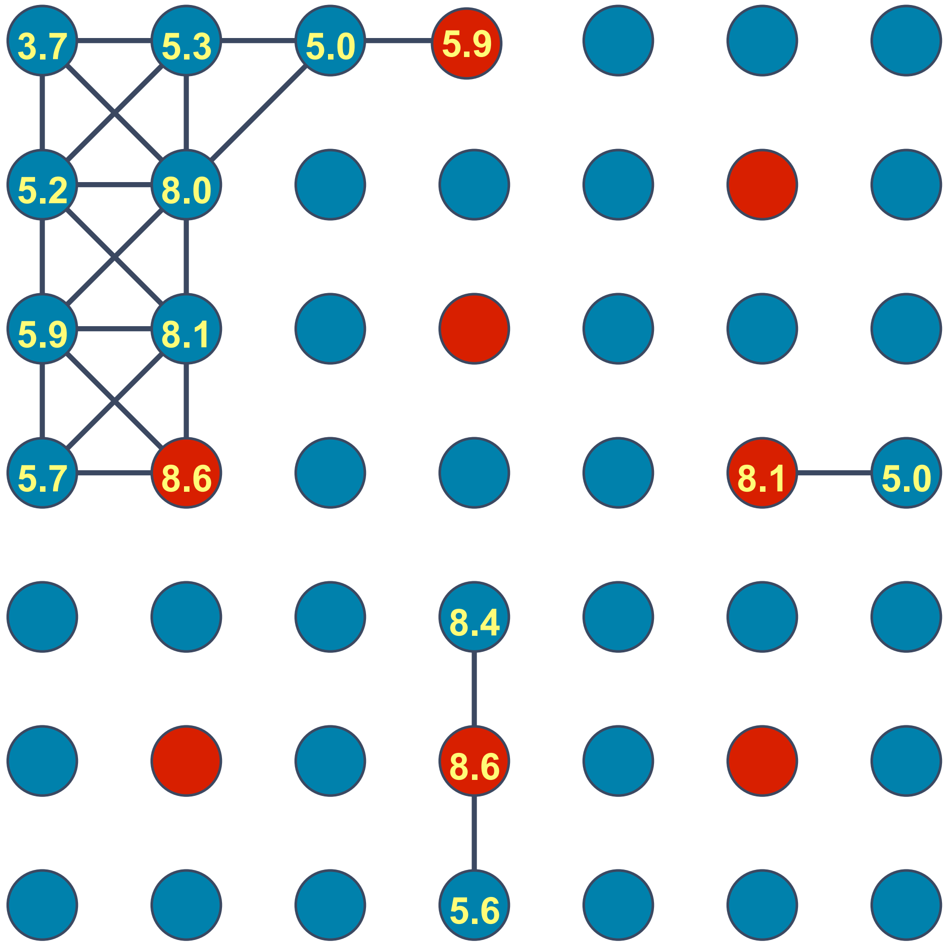
University of Waterloo

PMIS: remove and update 1



- select C-pts with maximal measure locally
- make neighbors F-pts
- remove neighbor edges

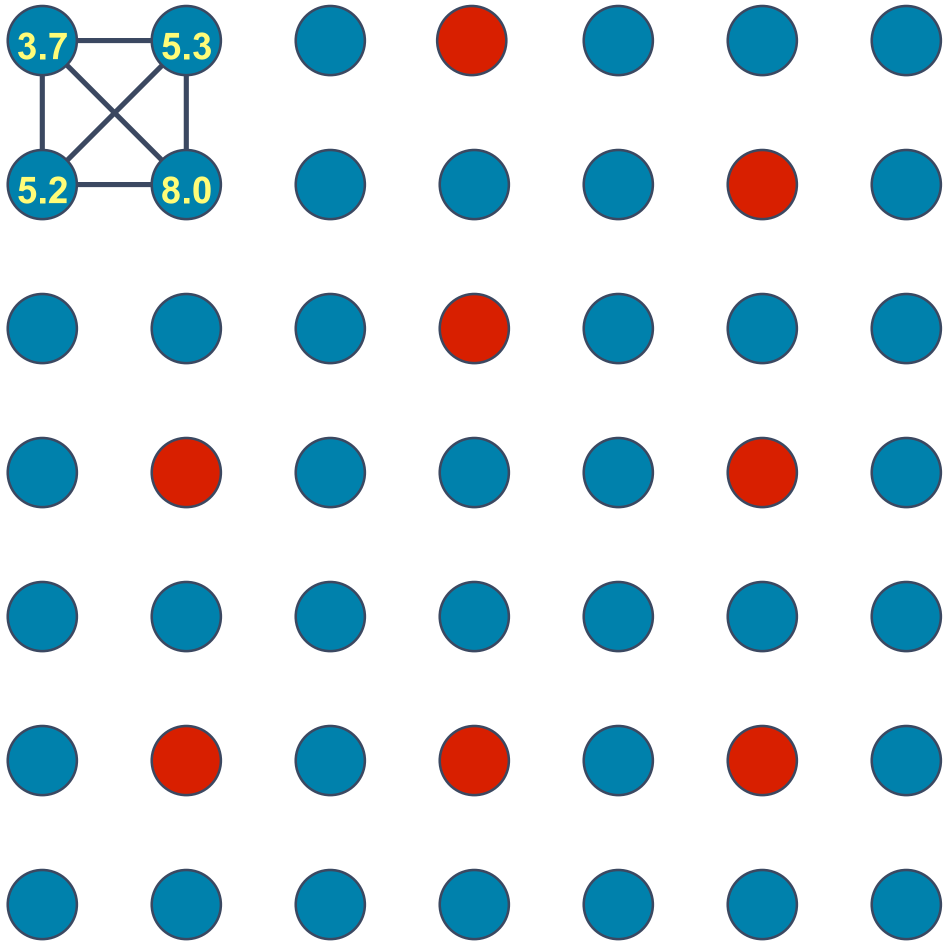
PMIS: select 2



- select C-pts with maximal measure locally
- make neighbors F-pts
- remove neighbor edges

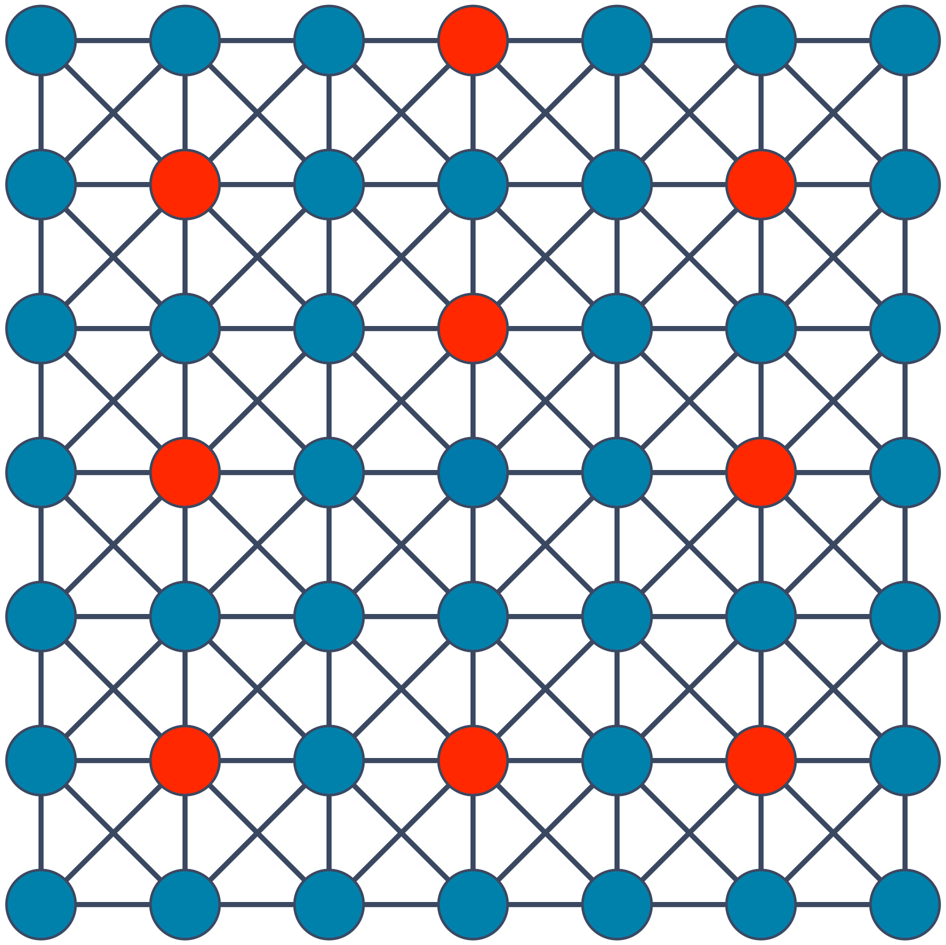
University of Waterloo

PMIS:
remove and update 2



- select C-pts with maximal measure locally
- **make neighbors F-pts**
- **remove neighbor edges**

PMIS:
final grid



- select C-pts with maximal measure locally
- make neighbor F-pts
- remove neighbor edges

University of Waterloo

PMIS coarsening: reduce complexity

- finite difference Laplacian (CLJP - PMIS+GMRES)

	dof	C_{op}	Iter	t_{tot}
2D	120^2	4.16	12	0.22
	120^2	1.90	24	0.24
3D	100^3	25.94	12	129.42
	100^3	2.36	20	27.68
4D	20^4	127.5	8	88.39
	20^4	2.95	11	4.31
5D	9^5	256.9	5	73.92
	8^5	3.14	8	0.91
	20^5	4.02	12	181.93

parallel scaling tests: MCR linux cluster

- study algorithmic and parallel scalability

- 2304 processors
  (2.4-GHz Xeon, 2 GB/proc)
- 11.2 TeraFlops
  (11.2×10^{12} floating point operations per second)
- Quadrics fast interconnection network

- at Lawrence Livermore National Laboratory, California, USA

University of Waterloo

PMIS results: 7-point finite difference Laplacian in 3D, 40^3 dof per proc

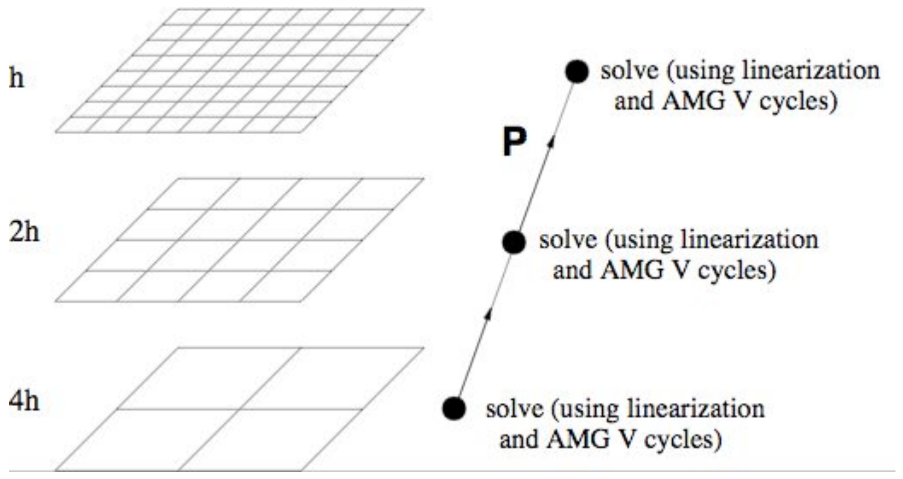
CLJP and PMIS-GMRES(10)

proc	C_{op}	Levels	t_{setup}	t_{solve}	Iter	t_{total}
1	14.39	15	1.88	1.47	6	3.35
512	17.02	22	22.33	13.50	10	35.83
1331	17.19	23	29.57	16.68	10	46.25
1	*2.32*	*7*	*0.41*	*0.87*	*13*	*1.28*
512	*2.37*	*10*	*5.04*	*7.73*	*25*	*12.77*
1331	*2.37*	*10*	*8.28*	*9.71*	*28*	*17.99*

University of Waterloo

(D) future work

- scalable multigrid solvers for PDEs of hyperbolic type
  - O(n) scaling for hyperbolic PDE systems is difficult
  - AMG is a challenge
- scalable solvers for nonlinear PDE systems: use nested iteration


h
2h
4h
P
solve (using linearization and AMG V cycles)
solve (using linearization and AMG V cycles)
solve (using linearization and AMG V cycles)

- improve scalability for very large machines...

Top 500 Supercomputer list (November 2004)

Rank	Site	Computer	Country	TeraFlops	Processors
1	Lawrence Livermore National Laboratory	IBM BlueGene/L	*US*	135	65,536
2	NASA/Ames Research Center/NAS	SGI Altix	*US*	51	10,160
3	The Earth Simulator Center	NEC Earth-Simulator	Japan	35	5,120
4	Barcelona Supercomputer Center	IBM eServer	Spain	20	3,564
5	Lawrence Livermore National Laboratory	Intel Itanium2	*US*	19	4,096
6	Los Alamos National Laboratory	ASCI Q - HP AlphaServer	*US*	13	8,192
7	Virginia Tech	1100 Dual 2.3 GHz Apple XServe	*US*	12	2,200
8	IBM - Rochester	IBM BlueGene/L	*US*	11	8,192
9	Naval Oceanographic Office	IBM eServer	*US*	10	2,944
10	NCSA	Dell P4 Xeon	*US*	10	2,500
11	ECMWF	IBM eServer	UK	10	2,176
12	ECMWF	IBM eServer	UK	10	2,176
	...				
17	Shanghai Supercomputer Center	Dawning 4000A, Opteron	China	8	2,560
18	Los Alamos National Laboratory	LNX Opteron	*US*	8	2,816
19	Lawrence Livermore National Laboratory	MCR Linux Cluster Xeon	*US*	8	2,304
20	Lawrence Livermore National Laboratory	ASCI White, IBM SP Power3	*US*	7	8,192

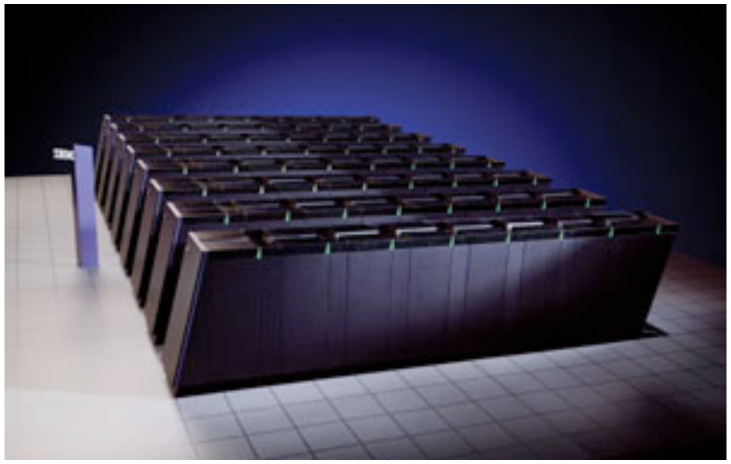
- scalable results were presented for MCR (#19), 2,000 procs
- next target: Blue Gene/L (#1), 65,000 procs

LLNL Blue Gene/L

- dual-processor nodes optimized for data access
- each node: one processor for simulation, one for communication
- only 256MB ram per processor
- lightweight, single-process linux kernel
- Blue Gene/L will be fully operational in July 2005, with 130,000 procs

LLNL Blue Gene/L

- our code currently runs on LLNL Blue Gene/L
- one preliminary result, on 8,000 processors: PMIS works, CLJP runs out of memory...
- more tests to follow...
- scalability up to 130,000 processors?



collaborators

- Ulrike Yang, Rob Falgout

  Center for Applied Scientific Computing, Lawrence Livermore National Laboratory, California
- Tom Manteuffel, Steve McCormick, John Ruge

  University of Colorado at Boulder
- Jeff Heys

  Arizona State University
- Jeff Butler

  University of Waterloo