Multigrid methods

Algebraic Multigrid methods

Algebraic Multilevel Iteration methods

Residual correction

$A\mathbf{x} = \mathbf{b}, \mathbf{x}_{exact}, \mathbf{e}^{(k)} = \mathbf{x}_{exact} - \mathbf{x}^{(k)}$

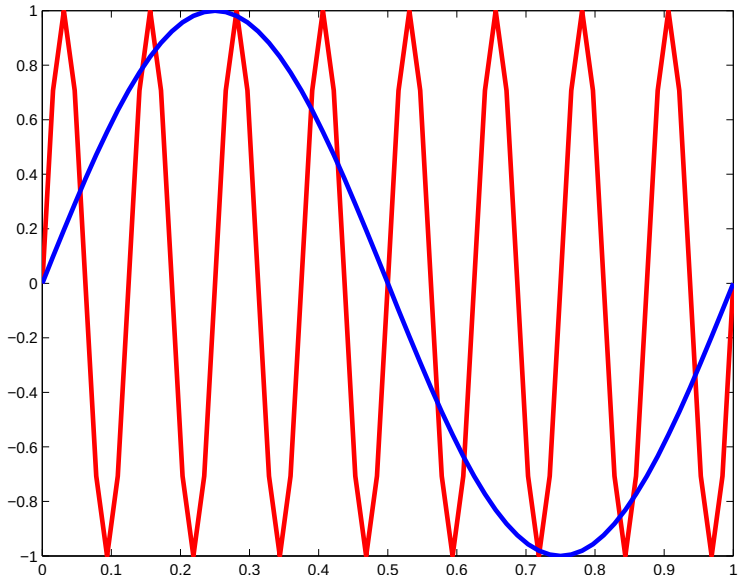
$\mathbf{r}^{(k)} = \mathbf{b} - A\mathbf{x}^{(k)}$

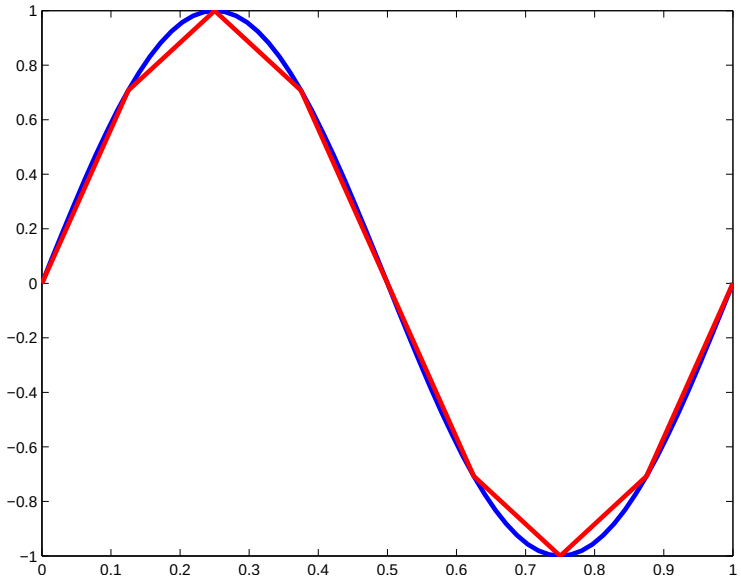
Residual equation:	$A\mathbf{e}^{(k)} = \mathbf{r}^{(k)}$
Residual correction:	$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{e}^{(k)}$
Recall:	$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + C^{-1}(\mathbf{b} - A\mathbf{x}^{(k)})$
Error propagation:	$\mathbf{e}^{(k+1)} = (I - C^{-1}A)\mathbf{e}^{(k)}$

Run Jacobi demo...

`student/NLA/Demos/Module3/L5`

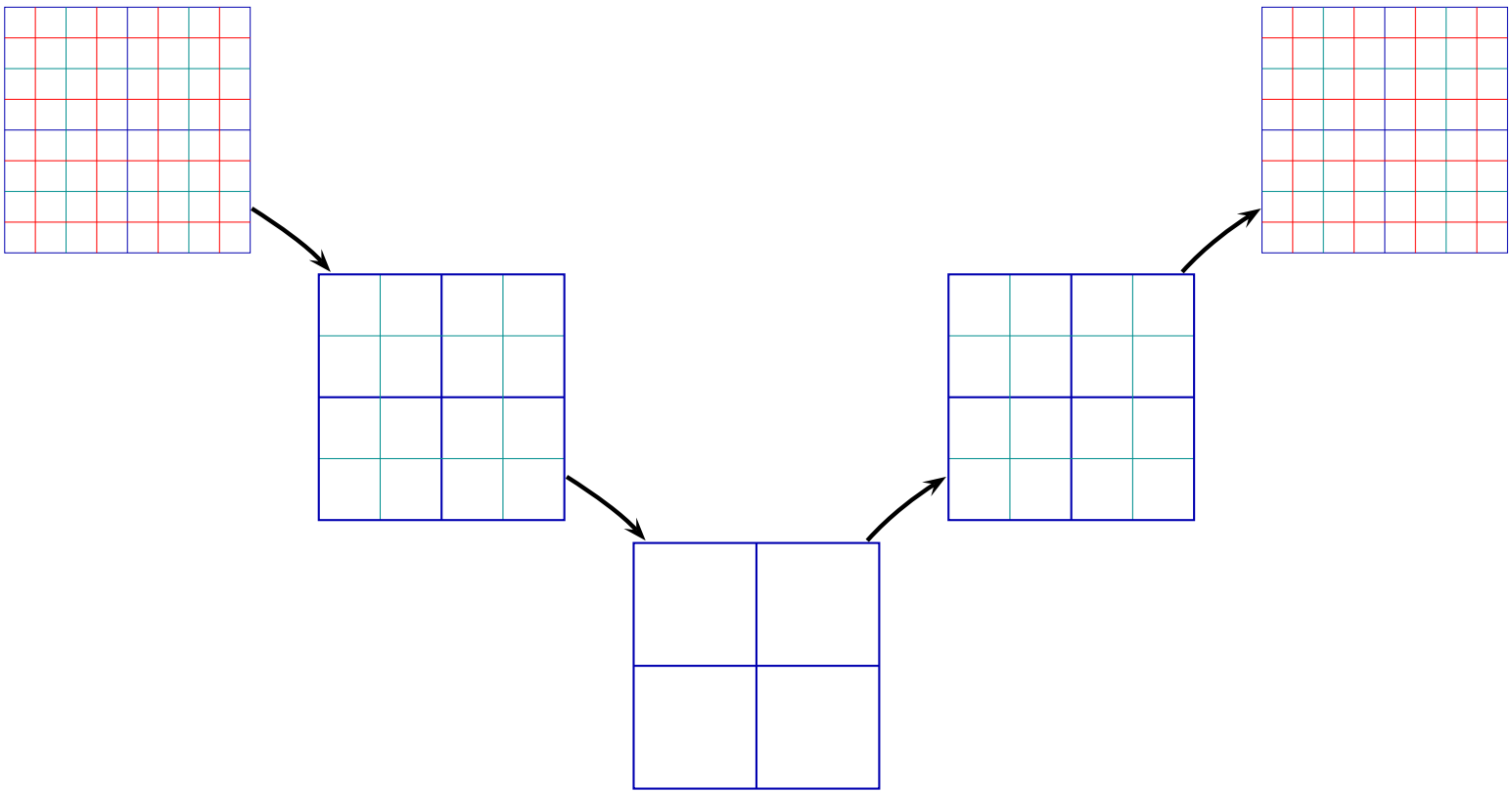
High and low frequencies - nonsmooth, smooth





Main idea: R. Fedorenko (1961), N.S. Bakhvalov (1966)

Reduce the error $\mathbf{e}^{(k)} = \mathbf{x}_{exact} - \mathbf{x}^{(k)}$ on the given (fine) grid by successive residual corrections on a hierarchy of (nested) coarser grids.



Some contributors:

Years	MG matches	AMG matches
1966-1986	384	9
1987-1996	1108	41
1997-2006	1251	156
2007-2013	823	178

Archi Brandt	Jan Mandel	Tom Manteiffel
Wolggang Hackbusch	Steve McCormick	Yvan Notay
Jurgen Ruge	Petr Vanec	Irad Yavneh
Klaus Stüben	Piet Hemker	Panayot Vassilevski

Ruge, J. W.; Stüben, K. Algebraic multigrid. Multigrid methods, 73-130, Frontiers Appl. Math., 3, SIAM, Philadelphia, PA, 1987.

Multilevel preconditioning methods: MG

Procedure MG: $\mathbf{u}^{(k)} \leftarrow MG\left(\mathbf{u}^{(k)}, \mathbf{f}^{(k)}, k, \{\nu_j^{(k)}\}_{j=1}^{k}\right)$;

if $k = 0$, **then** solve $A^{(0)}\mathbf{u}^{(0)} = \mathbf{f}^{(0)}$ exactly or by smoothing,

else

$\mathbf{u}^{(k)} \underset{s_1}{\leftarrow} \mathcal{S}_1^{(k)}\left(\mathbf{u}^{(k)}, \mathbf{f}^{(k)}\right)$, perform s_1 pre-smoothing steps,

Correct the residual:

$\mathbf{r}^{(k)} = A^{(k)}\mathbf{u}^{(k)} - \mathbf{f}^{(k)}$; form the current residual,

$\mathbf{r}^{(k-1)} \leftarrow \mathcal{R}\left(\mathbf{r}^{(k)}\right)$, restrict the residual on the next coarser grid,

$\mathbf{e}^{(k-1)} \leftarrow MG\left(\mathbf{0}, \mathbf{r}^{(k-1)}, k-1, \{\nu_j^{(k-1)}\}_{j=1}^{k-1}\right)$;

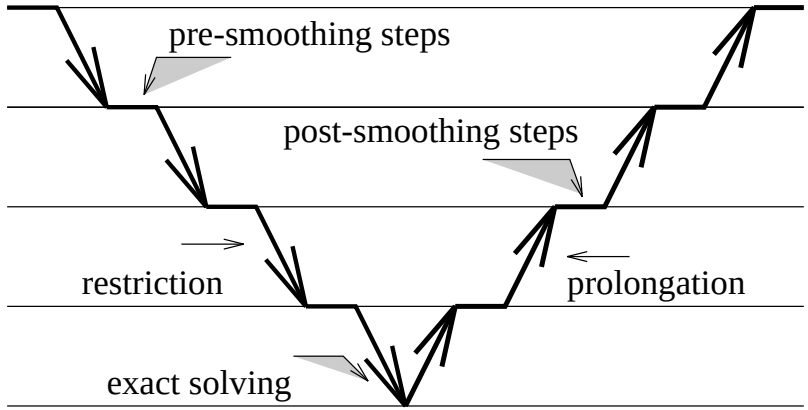
$\mathbf{e}^{(k)} \leftarrow \mathcal{P}\left(\mathbf{e}^{(k-1)}\right)$; prolong the error from the next coarser to the current grid,

$\mathbf{u}^{(k)} = \mathbf{u}^{(k)} - \mathbf{e}^{(k)}$; update the solution,

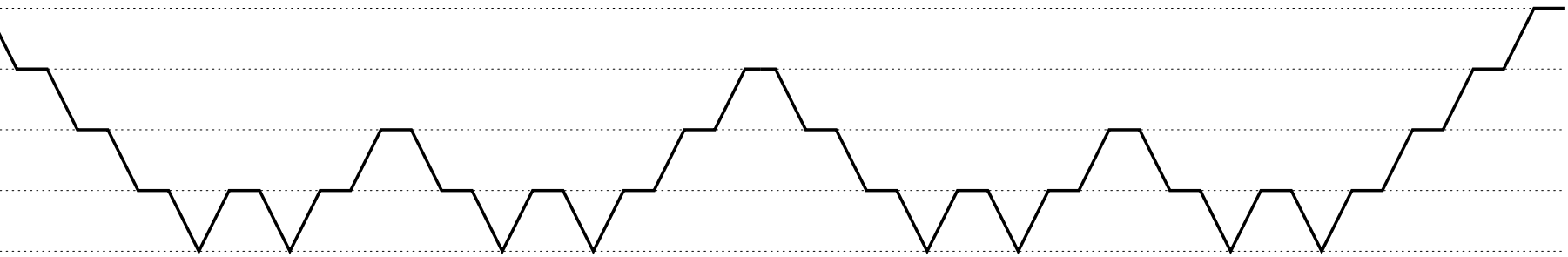
$\mathbf{u}^{(k)} \underset{s_2}{\leftarrow} \mathcal{S}_2^{(k)}\left(\mathbf{u}^{(k)}, \mathbf{f}^{(k)}\right)$, perform s_2 post-smoothing steps.

endif

end Procedure MG



One MG step (V-cycle)



The MG W-cycle

Nested iteration

Procedure NI: $\mathbf{u}^{(\ell)} \leftarrow NI\left(\mathbf{u}^{(0)}, \{\mathbf{f}^{(k)}\}_{k=1}^{(\ell)}, \ell, \{\nu^{(k)}\}_{k=1}^{\ell}\right)$;

$\mathbf{u}^{(0)} = {A^{(0)}}^{-1}\mathbf{f}^{(0)}$,

for k= 1 **to** ℓ **do**

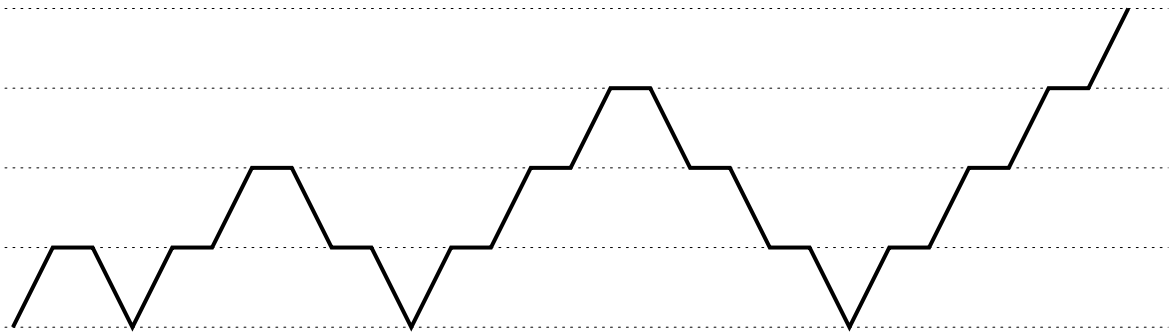
$\mathbf{u}^{(k)} = \mathcal{P}\left(\mathbf{u}^{(k-1)}\right)$;

$\mathbf{u}^{(k)} \leftarrow MG\left(\mathbf{u}^{(k)}, \mathbf{f}^{(k)}, k, \{\nu_j^{(k)}\}_{j=1}^{k}\right)$;

endfor

end Procedure NI

The so-called *full* MG corresponds to **Procedure** $NI(\cdot, \cdot, \ell, \{1, 1, \cdots, 1\})$



The full MG (V-cycle)

A compact formula presenting the MG procedure in terms of a recursively defined iteration matrix:
(i) Let $M^{(0)} = 0$,
(ii) For $k = 1$ to ℓ, define

$$M^{(k)} = \mathcal{S}^{(k)^{s_2}} \left(A^{(k)^{-1}} - \mathcal{P}_{k-1}^{k} \left(I - M^{(k-1)^{\nu}} \right) A^{(k-1)^{-1}} \mathcal{R}_{k}^{k-1} \right) A^{(k)} \mathcal{S}^{(k)^{s_1}},$$

where $\mathcal{S}^{(k)}$ is a smoothing iteration matrix (assuming $\mathcal{S}_1$ and $\mathcal{S}_2$ are the same), $\mathcal{R}_{k}^{k-1}$ and $\mathcal{P}_{k-1}^{k}$ are matrices which transfer data between two consecutive grids and correspond to the restriction and prolongation operators $\mathcal{R}$ and $\mathcal{P}$, respectively, and $\nu = 1$ and $\nu = 2$ correspond to the V- and W-cycles.

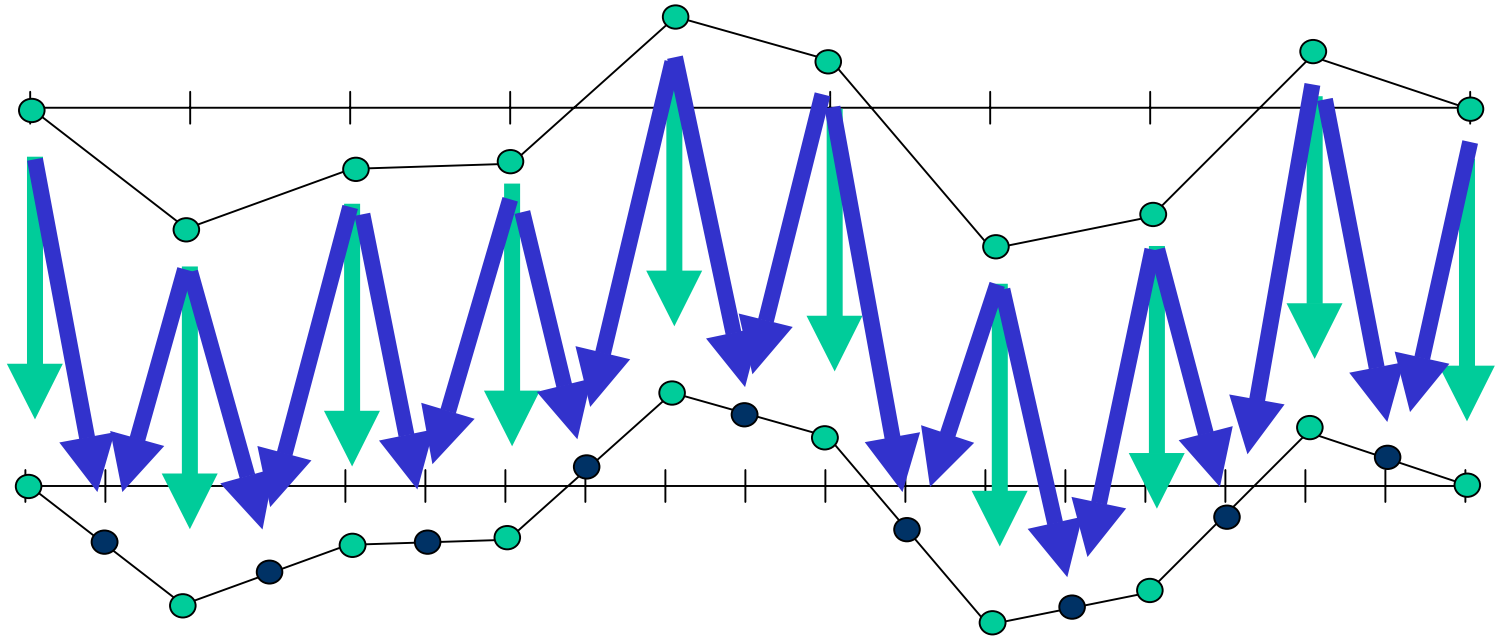
It turns out that in many cases the spectral radius of $M^{(\ell)}$, $\rho\left(M^{(\ell)}\right)$, is independent of ℓ, thus the rate of convergence of the NI method is optimal. Also, a mechanism to make the spectral radius of $M^{(\ell)}$ smaller is to choose s_1 and s_2 larger. The price for the latter is, clearly, a higher computational cost.

MG ingredients

- smoothers (many different)
  - Jacobi, weighted Jacobi ($\omega diag(A)$, GS, SOR, SSOR, SPAI
- restriction and prolongation operators
- coarse level matrix (approximation properties)

1D Interpolation (Prolongation)

- Values at points on the coarse grid map unchanged to the fine grid
- Values at fine-grid points NOT on the coarse grid are the averages of their coarse-grid neighbors



1D Restriction by injection

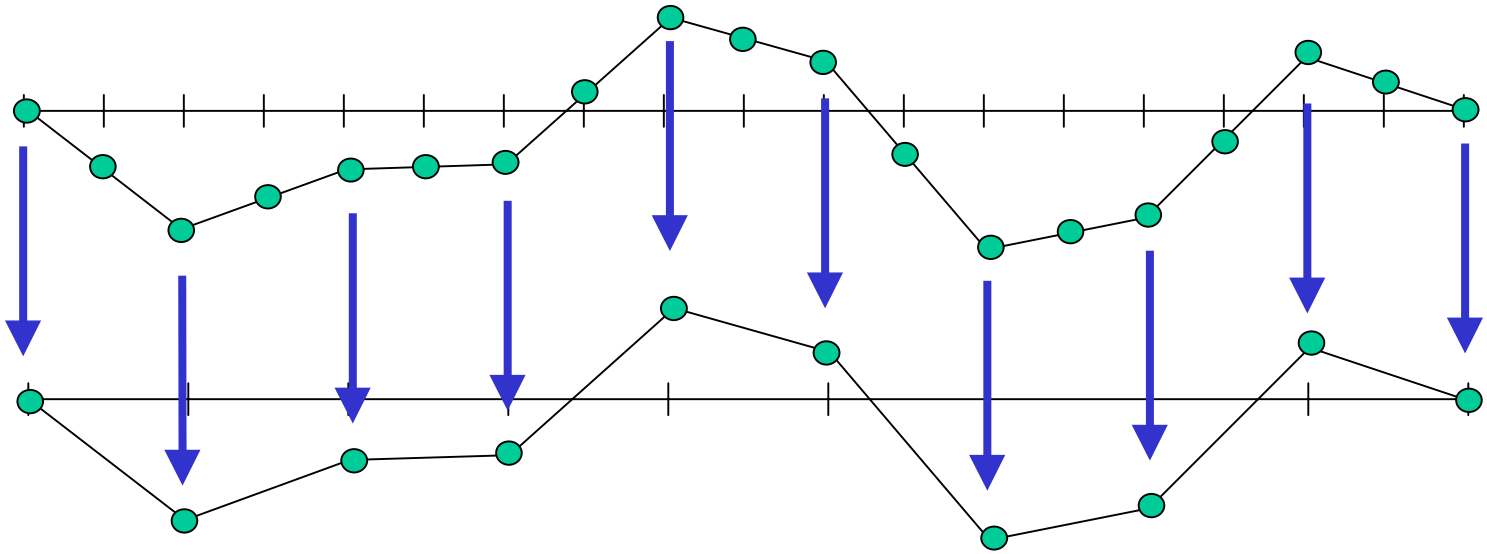
- Mapping from the fine grid to the coarse grid:

$$I_h^{2h} : \Omega^h \rightarrow \Omega^{2h}$$

- Let v^h, v^{2h} be defined on Ω^h, Ω^{2h}. Then

$$I_h^{2h} v^h = v^{2h}$$

where $v_i^{2h} = v_{2i}^h$.



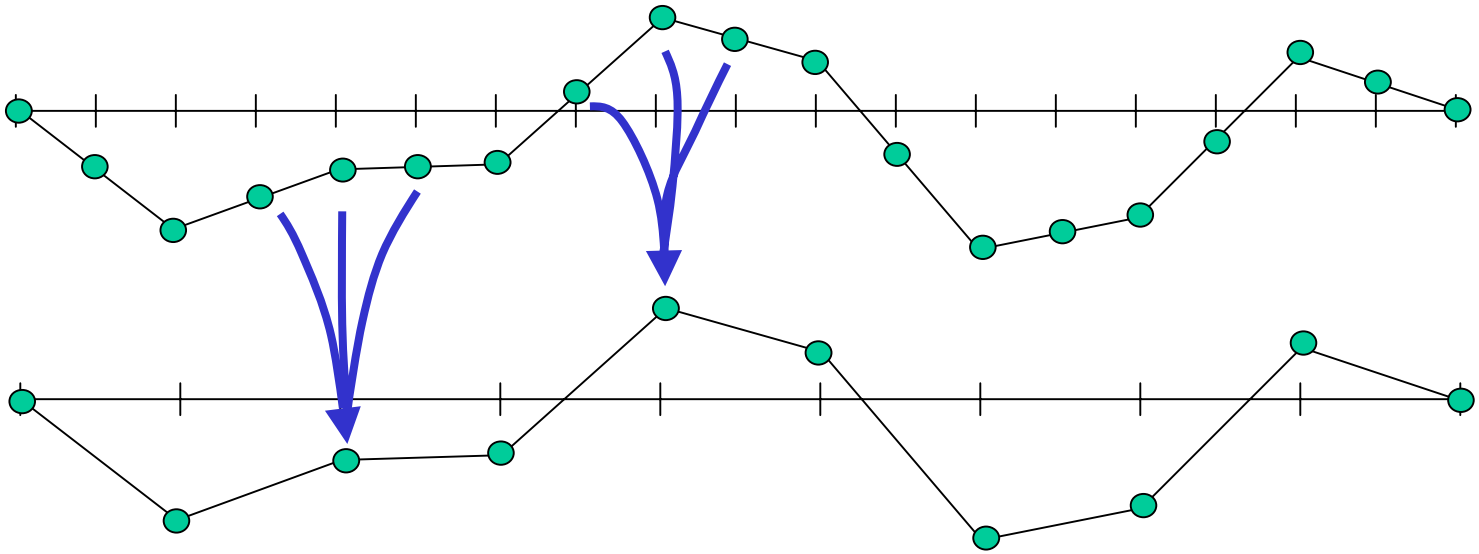
1D Restriction by full-weighting

- Let v^h, v^{2h} be defined on Ω^h, Ω^{2h}. Then

$$I_h^{2h} v^h = v^{2h}$$

where

$$v_i^{2h} = \frac{1}{4}(v_{2i-1}^h + 2v_{2i}^h + v_{2i+1}^h)$$



Prolongation and restriction are often nicely related

- For the 1D examples, linear interpolation and full-weighting are related by:

$$I_{2h}^{h} = \frac{1}{2}\begin{pmatrix} 1 & & \\ 2 & & \\ 1 & 1 & \\ & 2 & \\ & 1 & 1 \\ & & 2 \\ & & 1 \end{pmatrix} \qquad I_{h}^{2h} = \frac{1}{4}\begin{pmatrix} 1 & 2 & 1 & & & \\ & & 1 & 2 & 1 & \\ & & & & 1 & 2 & 1 \end{pmatrix}$$

- A commonly used, and highly useful, requirement is that

$$\boxed{I_{2h}^{h} = c\,(I_{h}^{2h})^{T}} \quad \text{for c in } \Re$$

2D Prolongation

$$v^h_{2i,2j} = v^{2h}_{ij}$$

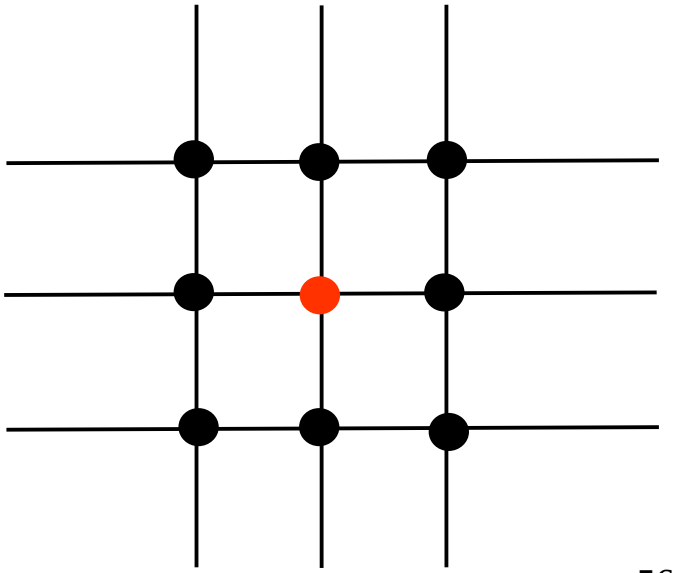
$$v^h_{2i+1,2j} = \frac{1}{2}\left(v^{2h}_{ij} + v^h_{i+1,j} \right)$$

$$v^h_{2i,2j+1} = \frac{1}{2}\left(v^{2h}_{ij} + v^h_{i,j+1} \right)$$

$$v^h_{2i+1,2j+1} = \frac{1}{4}\left(v^{2h}_{ij} + v^h_{i+1,j} + v^h_{i,j+1} + v^h_{i+1,j+1} \right)$$

$$\left[\begin{array}{ccc} \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \\ \frac{1}{2} & 1 & \frac{1}{2} \\ \frac{1}{4} & \frac{1}{2} & \frac{1}{4} \end{array}\right]$$

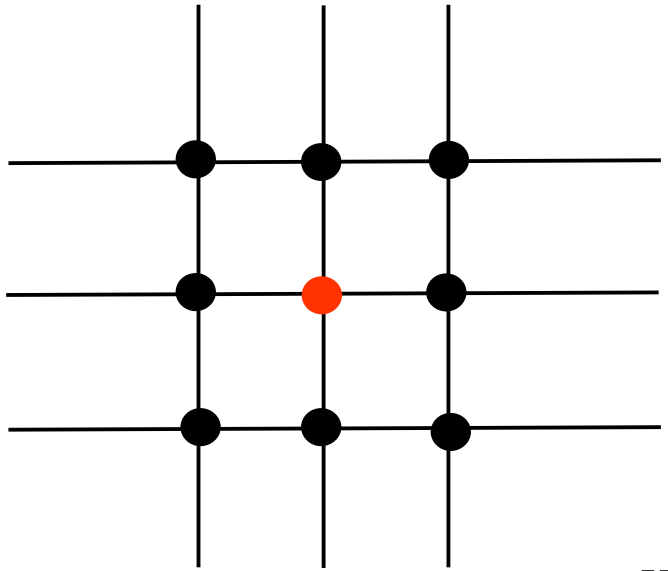
We denote the operator by using a "give to" stencil,] [. Centered over a c-point, ●, it shows what fraction of the c-point's value is contributed to neighboring f-points, ●.



2D Restriction (full-weighting)

$$\begin{bmatrix} \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \\ \frac{1}{8} & \frac{1}{4} & \frac{1}{8} \\ \frac{1}{16} & \frac{1}{8} & \frac{1}{16} \end{bmatrix}$$

We denote the operator by using a "give to" stencil, [ ]. Centered over a c-point, ●, it shows what fractions of the neighboring (●) f-points' value is contributed to the value at the c-point.



The variational properties

- The definition for A^{2h} that resulted from the foregoing line of reasoning is useful for both theoretical and practical reasons. Together with the commonly used relationship between restriction and prolongation we have the following "variational properties":

$$A^{2h} = I_h^{2h} A^h I_{2h}^h \qquad \text{(Galerkin Condition)}$$

$$I_{2h}^h = c\,(I_h^{2h})^T \qquad \text{for c in } \Re$$

MG: Rate of convergence and computational complexity

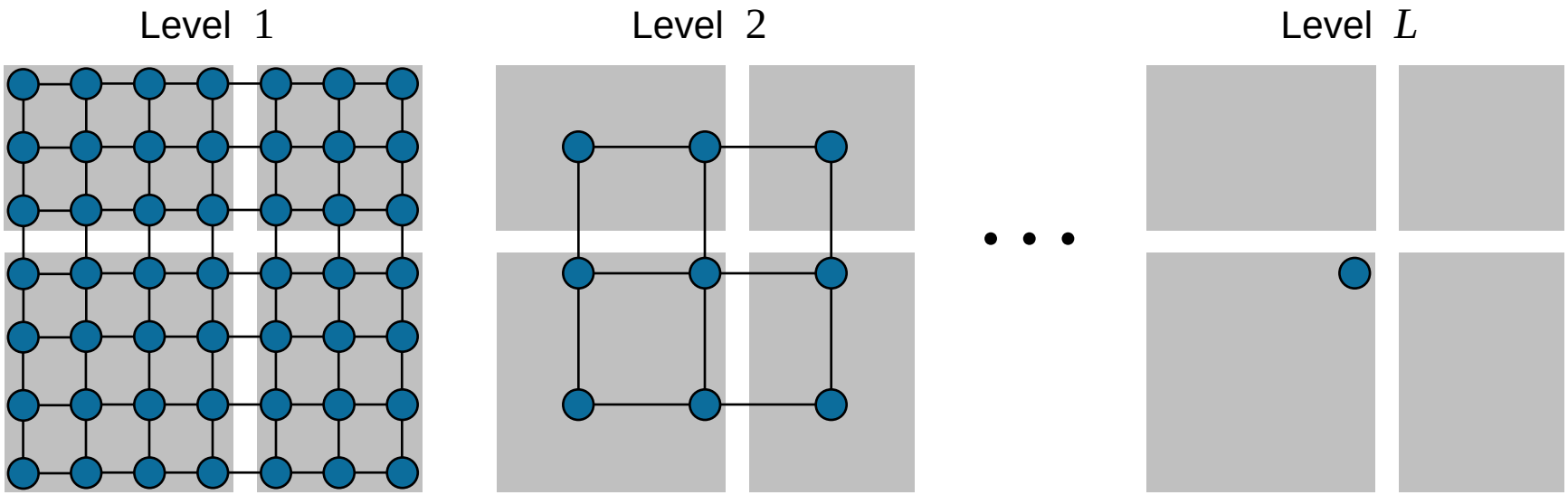
Algebraic Multigrid

AMG demo ...

AGMG http://homepages.ulb.ac.be/~ynotay/AGMG/

.../Projects/ComplexSymmetric

Approach for parallelizing multigrid is straightforward data decomposition



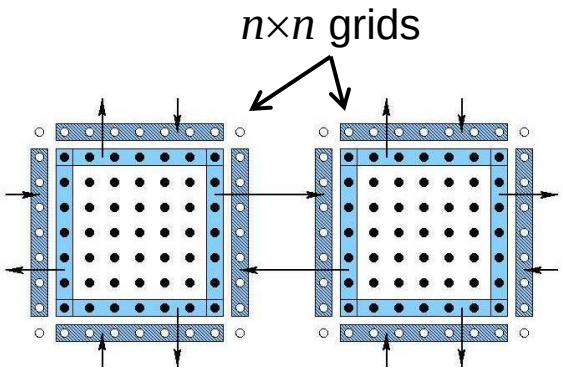
- Basic communication pattern is “nearest neighbor”
  - Relaxation, interpolation, & Galerkin not hard to implement
- Different neighbor processors on coarse grids
- Many idle processors on coarse grids (100K+ on BG/L)
  - Algorithms to take advantage have had limited success

Straightforward parallelization approach is optimal for V-cycles on structured grids (5-pt Laplacian example)

- Standard communication / computation models

$$T_{comm} = \alpha + m\beta \quad \textit{(communicate m doubles)}$$

$$T_{comp} = m\gamma \quad \textit{(compute m flops)}$$



- Time to do relaxation

$$T \approx 4\alpha + 4n\beta + 5n^2\gamma$$

- Time to do relaxation in a V(1,0) multigrid cycle

$$\begin{aligned} T_V &\approx (1+1+\cdots)4\alpha + (1+1/2+\ldots)4n\beta + (1+1/4+\ldots)5n^2\gamma \\ &\approx (\log N)4\alpha + (2)4n\beta + (4/3)5n^2\gamma \end{aligned}$$

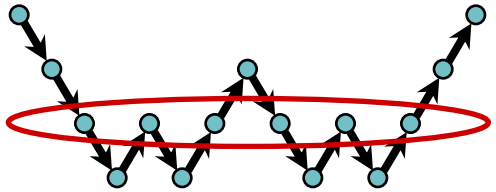
- For achieving optimality in general, the *log* term is unavoidable!

- More precise: $T_{V,better} \approx T_V + (\log P)(4\beta + 5\gamma)$

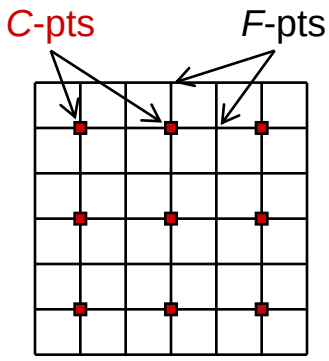
Additional comments on parallel multigrid

- W-cycles scale poorly:

$$T_W \approx (2^{\log N})4\alpha + (\log N)4n\beta + (2)5n^2\gamma$$

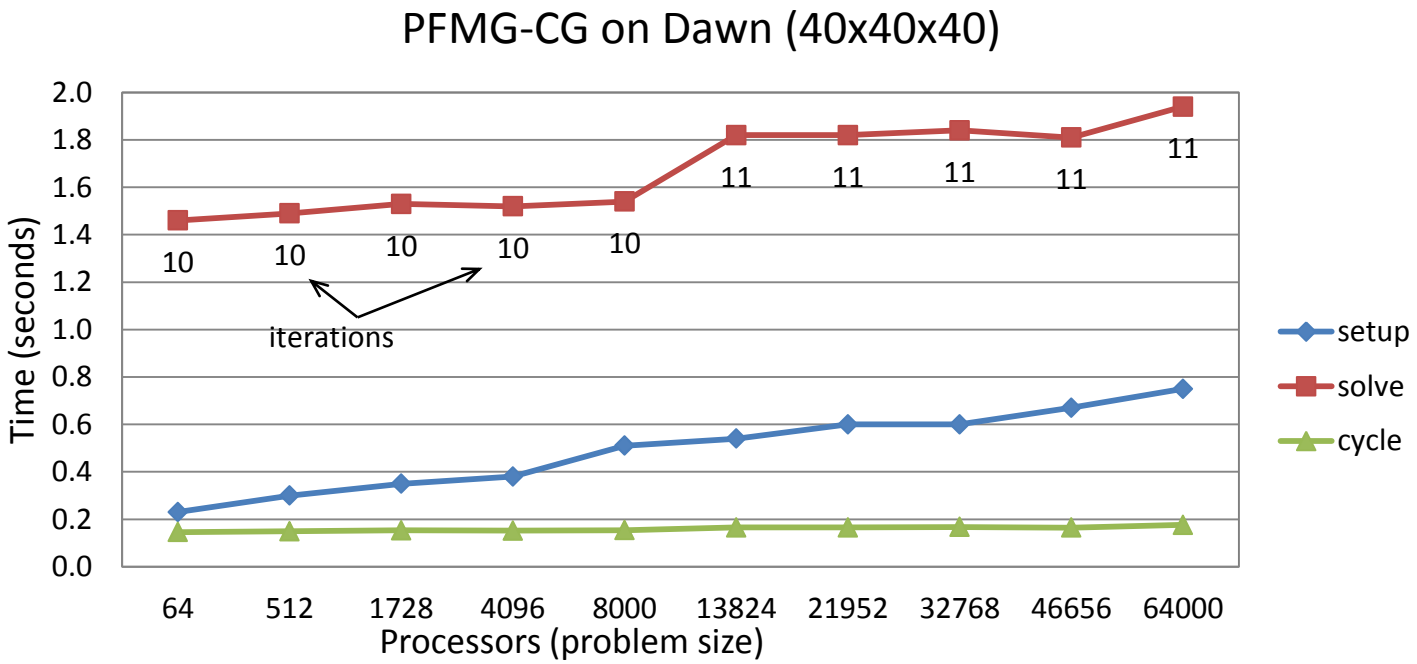


- Lexicographical Gauss-Seidel is too sequential
  - Use red/black or multi-color GS
  - Use weighted Jacobi, hybrid Jacobi/GS, L1
  - Use *C*-*F* relaxation (Jacobi on *C*-pts then *F*-pts)
  - Use Polynomial smoothers
- Parallel smoothers are often less effective



- Recent survey on parallel multigrid:
  - "A Survey of Parallelization Techniques for Multigrid Solvers," Chow, Falgout, Hu, Tuminaro, and Yang, *Parallel Processing For Scientific Computing*, Heroux, Raghavan, and Simon, editors, SIAM, series on Software, Environments, and Tools (2006)
- Recent paper on parallel smoothers:
  - "Multigrid Smoothers for Ultra-Parallel Computing," Baker, Falgout, Kolev, and Yang, *SIAM J. Sci. Comput.*, submitted. LLNL-JRNL-435315

Example weak scaling results on Dawn (an IBM BG/P system at LLNL) in 2010



- Laplacian on a cube; 40^3 = 64K grid points per processor; largest problem had 3 billion unknowns!
- PFMG is a semicoarsening multigrid solver in *hypre*
- Still room to improve setup implementation (these results already employ the assumed partition algorithm described later)

Basic multigrid research challenge

- Optimal $O(N)$ multigrid methods don't exist for some applications, even in serial
- Need to invent methods for these applications

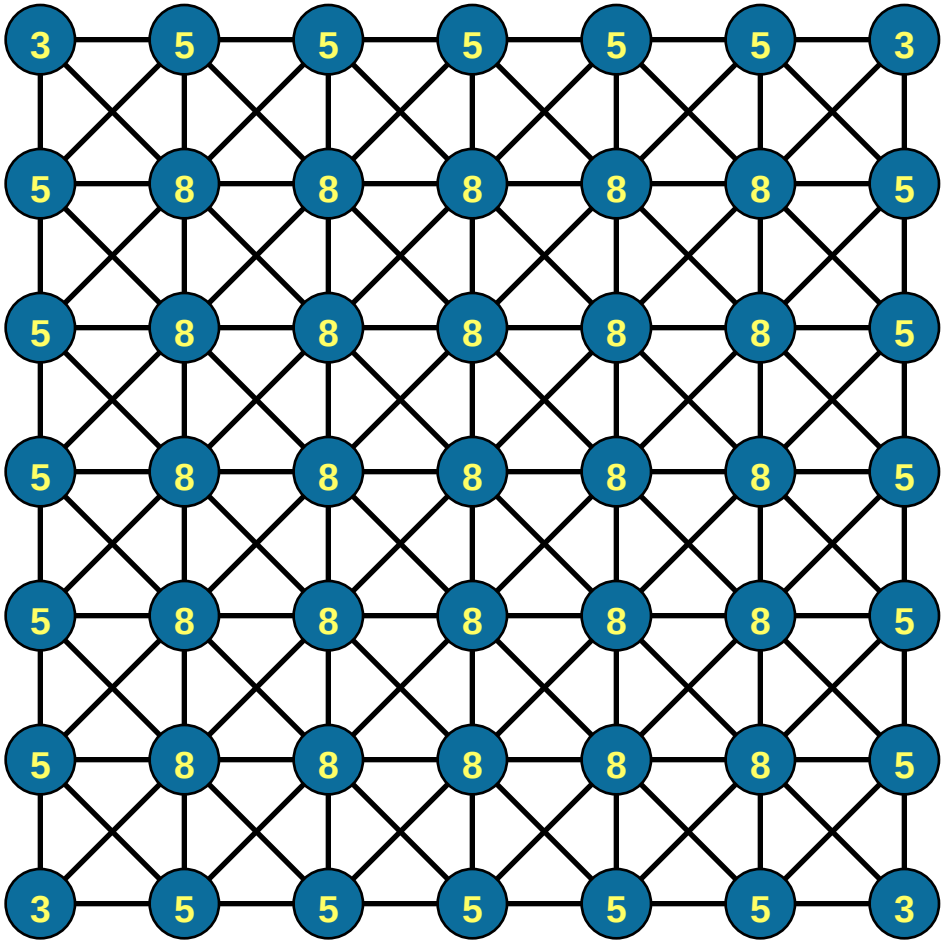
- However …

- Some of the classical and most proven techniques used in multigrid methods don't parallelize
  - Gauss-Seidel smoothers are inherently sequential
  - W-cycles have poor parallel scaling
- Parallel computing imposes additional restrictions on multigrid algorithmic development

Choosing the coarse grid

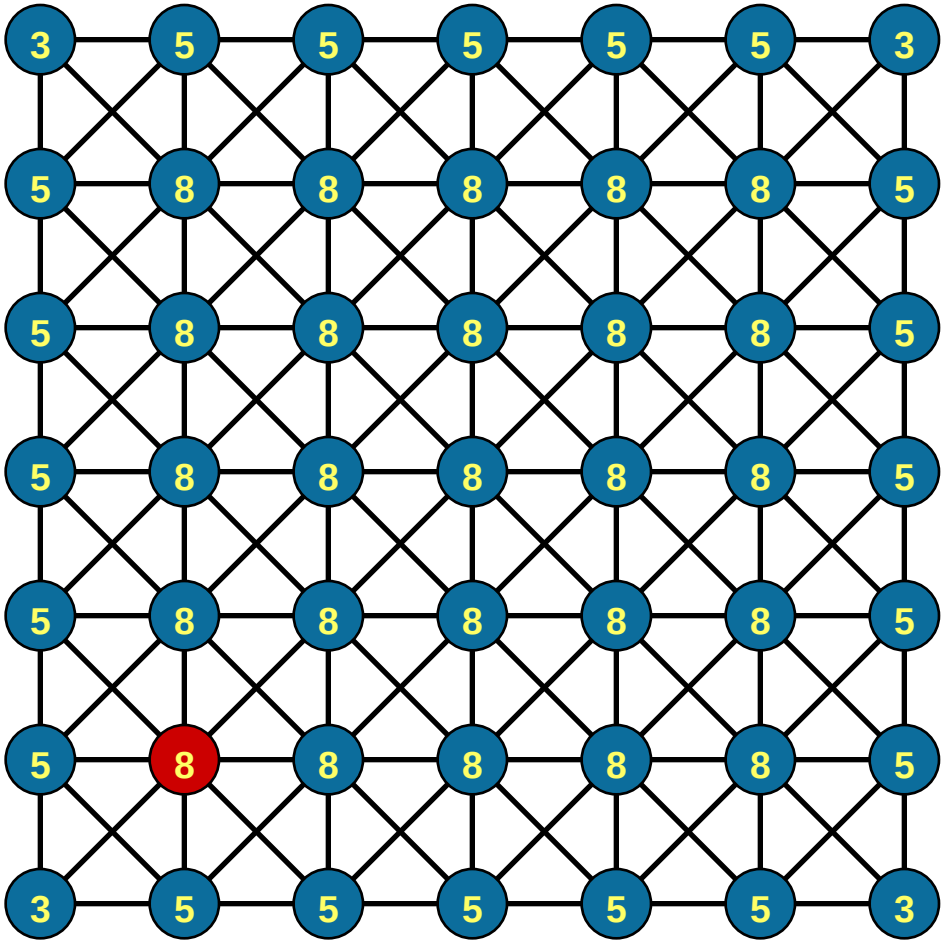
- In C-AMG, the coarse grid is a subset of the fine grid
- The basic coarsening procedure is as follows:
  - Define a strength matrix A_s by deleting weak connections in A
  - First pass: Choose an independent set of fine-grid points based on the graph of A_s
  - Second pass: Choose additional points if needed to satisfy interpolation requirements
- Coarsening partitions the grid into *C*- and *F*-points

C-AMG coarsening



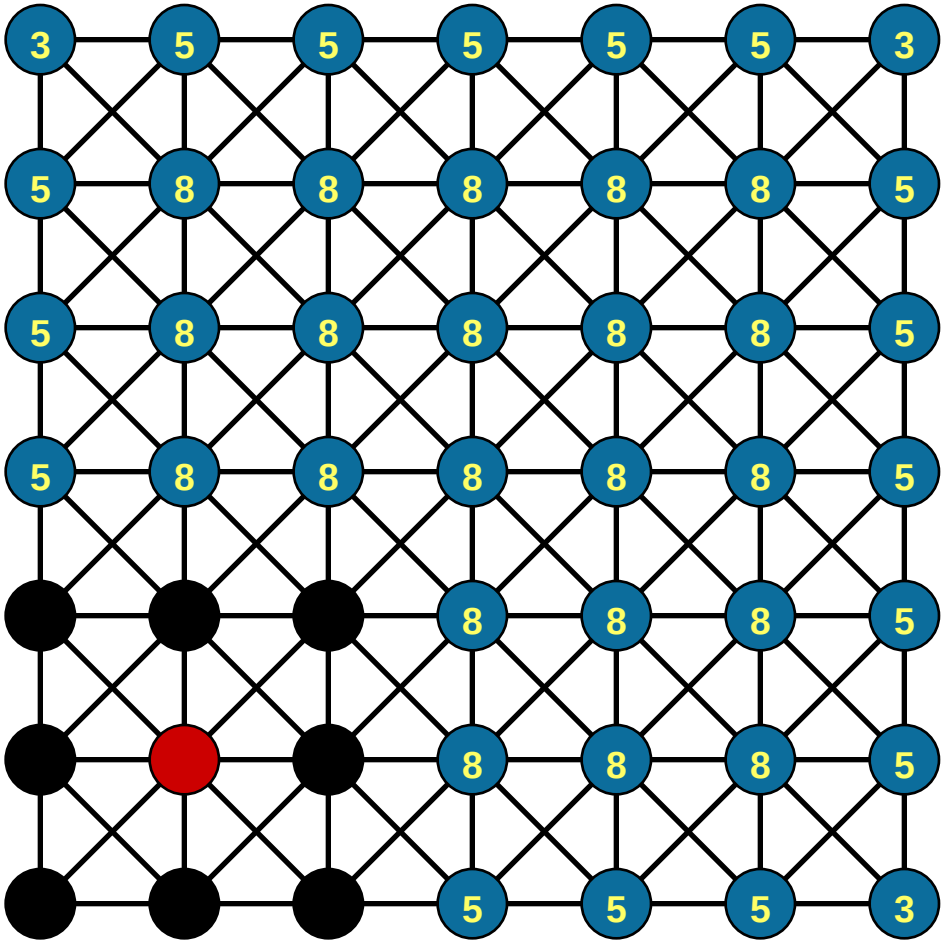
- select C-pt with maximal measure
- select neighbors as F-pts
- update measures of F-pt neighbors

C-AMG coarsening



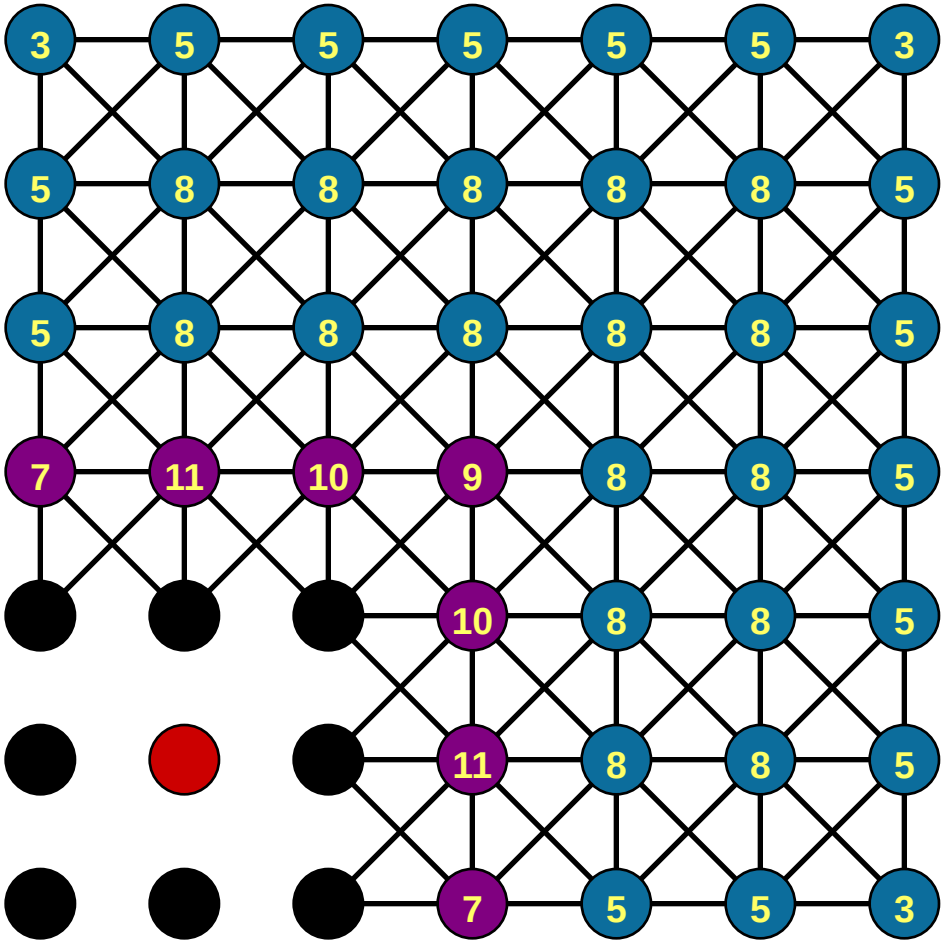
- ➔ **select C-pt with maximal measure**
- ➔ **select neighbors as F-pts**
- ➔ **update measures of F-pt neighbors**

C-AMG coarsening



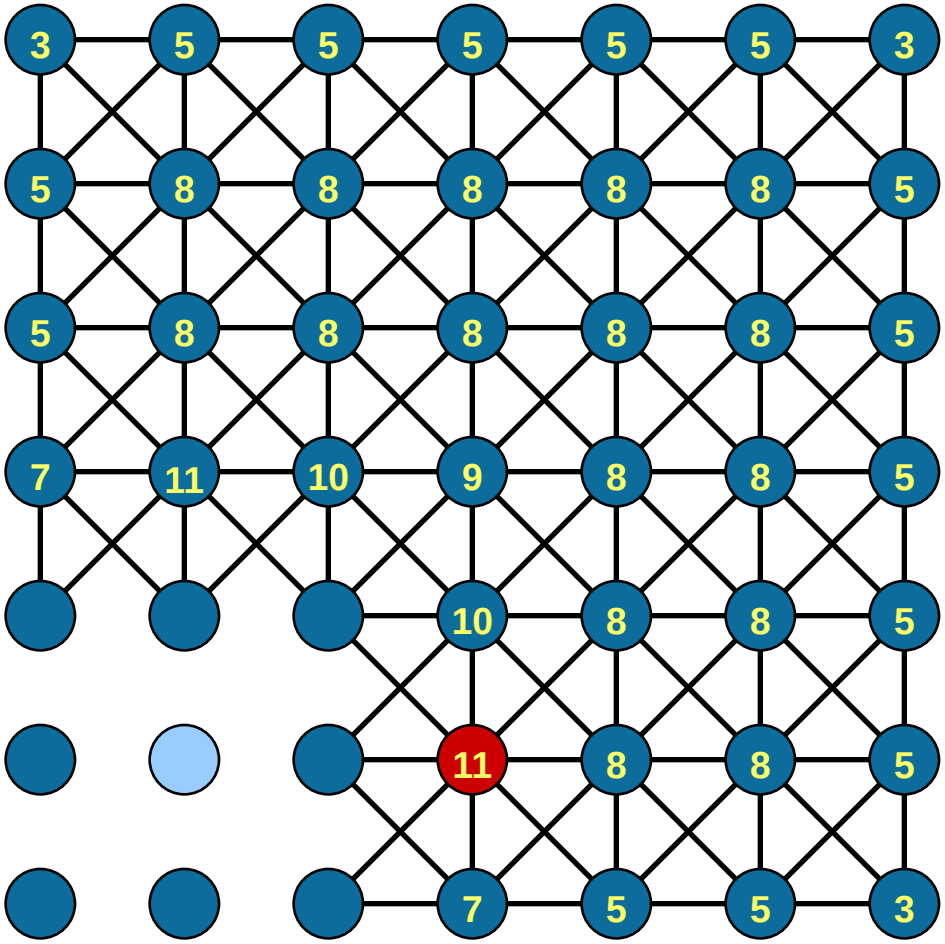
- select C-pt with maximal measure
- select neighbors as F-pts
- update measures of F-pt neighbors

C-AMG coarsening



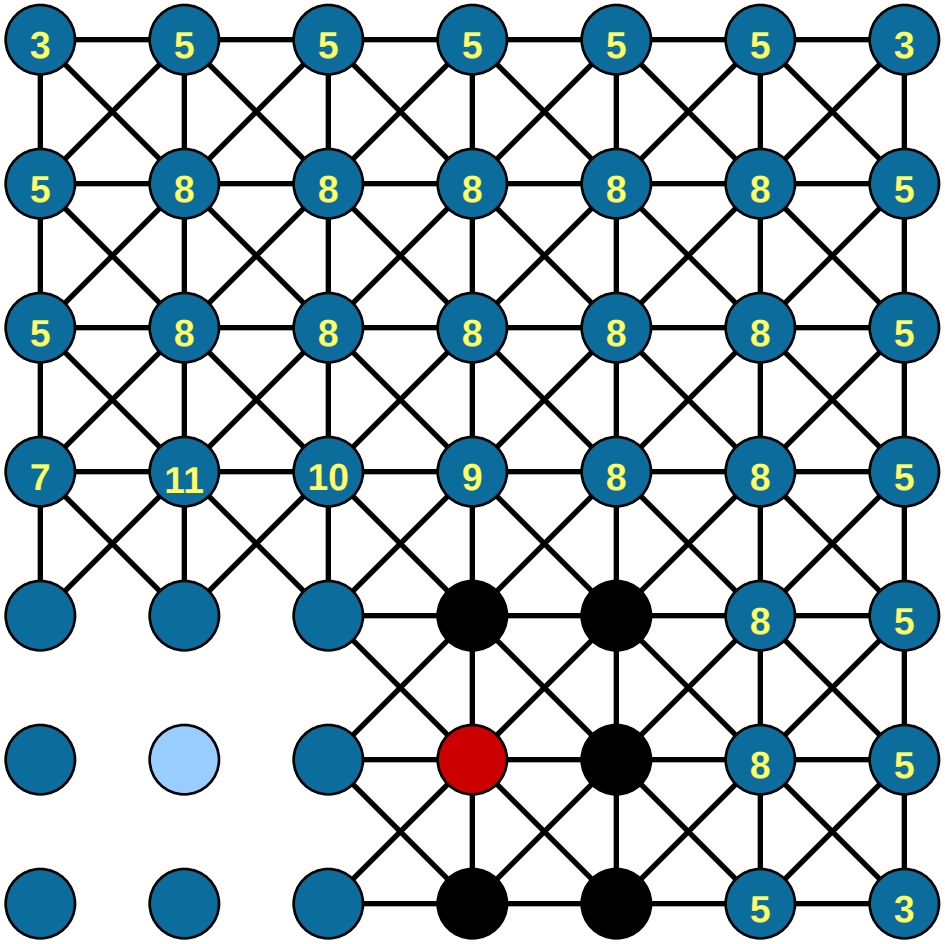
- select C-pt with maximal measure
- select neighbors as F-pts
- update measures of F-pt neighbors

C-AMG coarsening



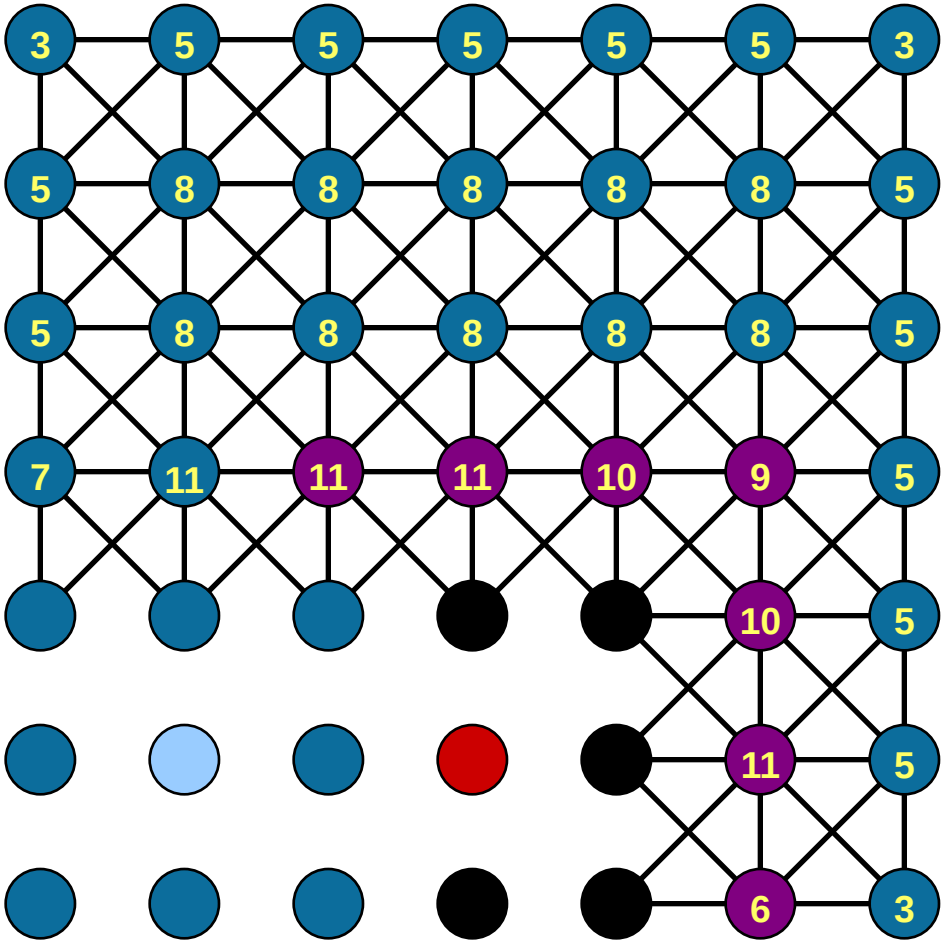
- select C-pt with maximal measure
- select neighbors as F-pts
- update measures of F-pt neighbors

C-AMG coarsening



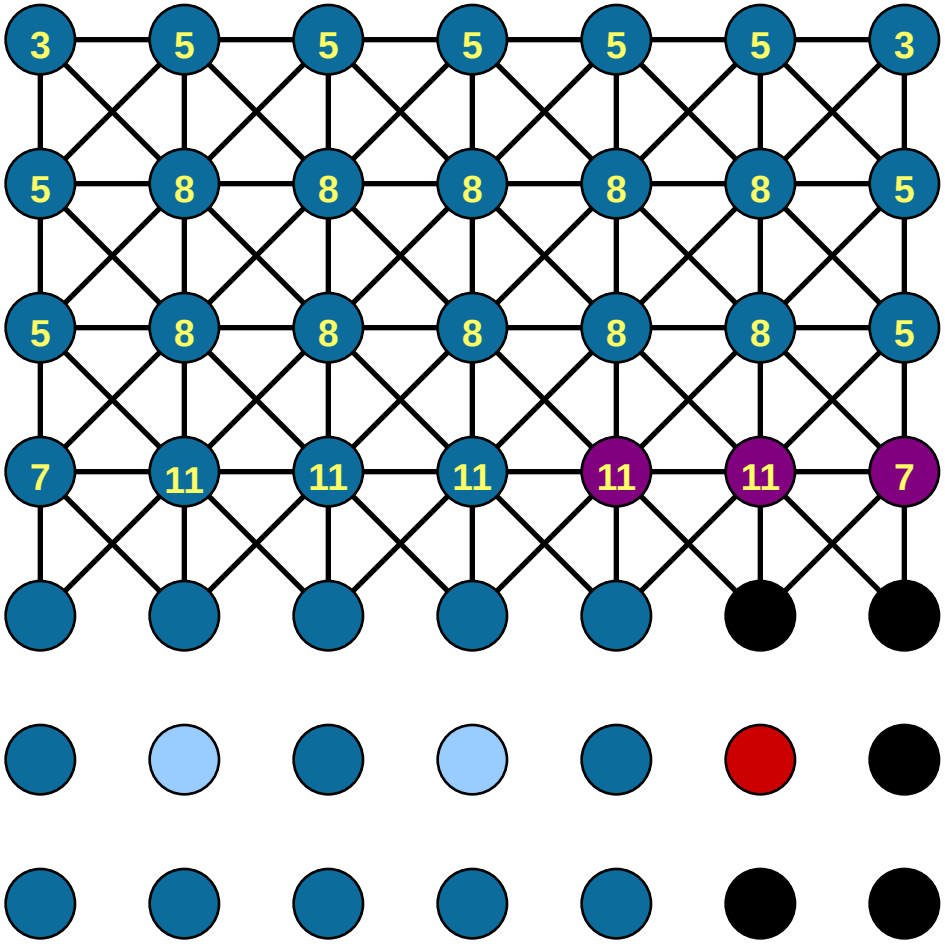
- select C-pt with maximal measure
- select neighbors as F-pts
- update measures of F-pt neighbors

C-AMG coarsening



- select C-pt with maximal measure
- select neighbors as F-pts
- update measures of F-pt neighbors

C-AMG coarsening

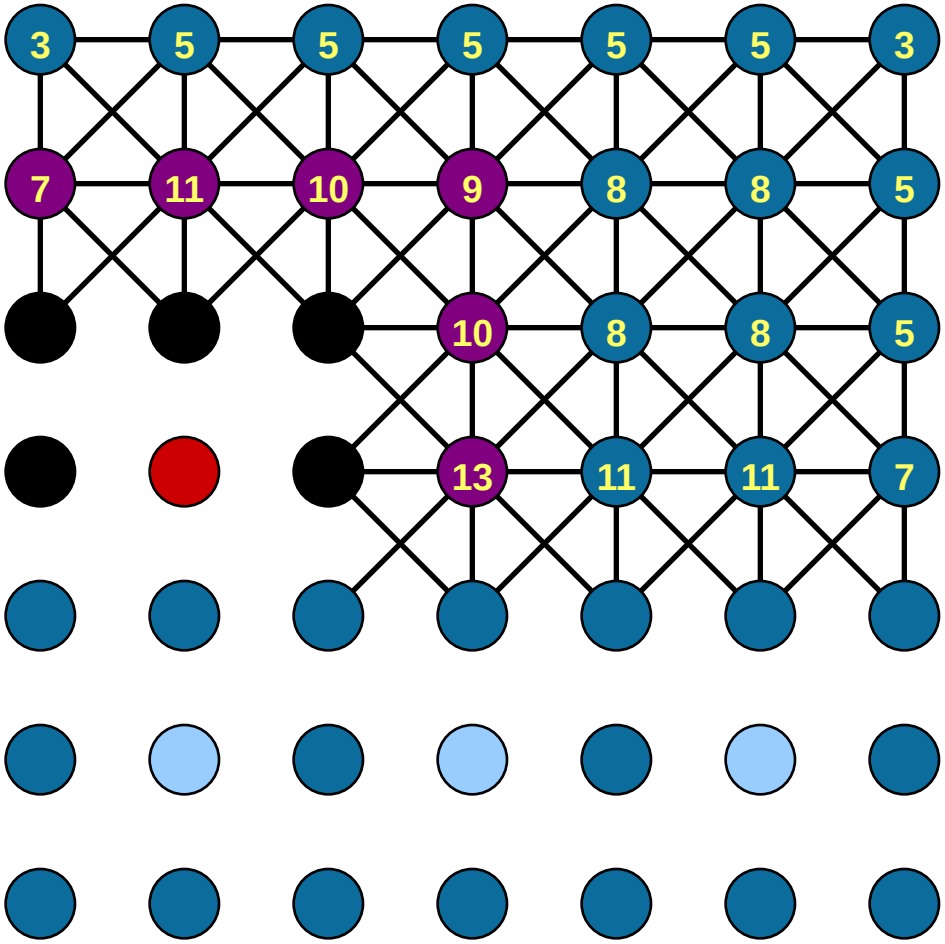


➔ select C-pt with maximal measure

➔ select neighbors as F-pts

➔ update measures of F-pt neighbors

C-AMG coarsening

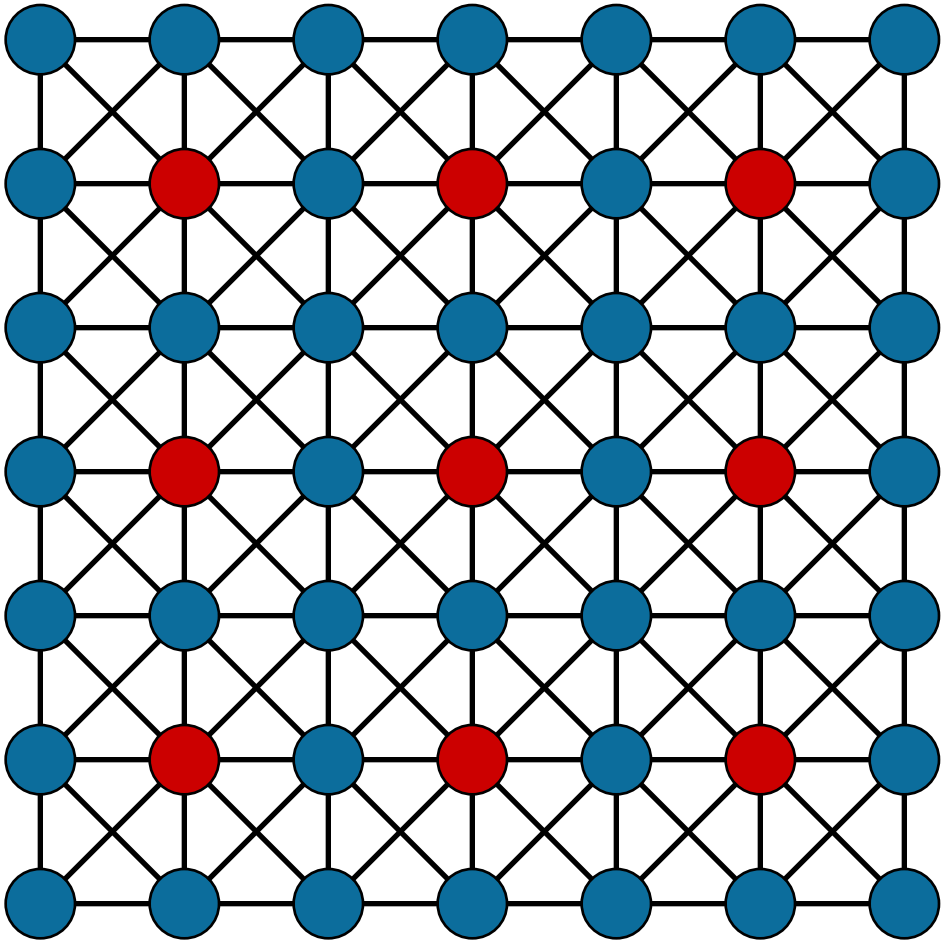


➔ select C-pt with maximal measure

➔ select neighbors as F-pts

➔ update measures of F-pt neighbors

C-AMG coarsening is inherently sequential



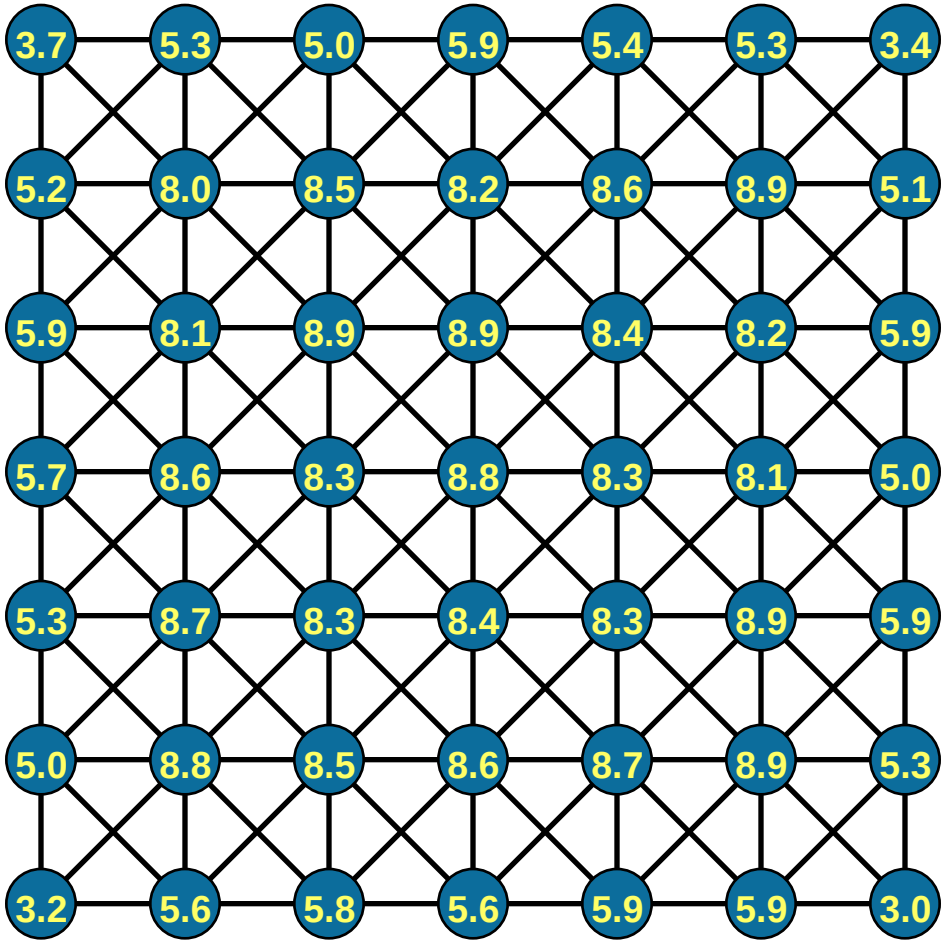
- select C-pt with maximal measure
- select neighbors as F-pts
- update measures of F-pt neighbors

Parallel Coarsening Algorithms

- C-AMG coarsening algorithm is inherently sequential

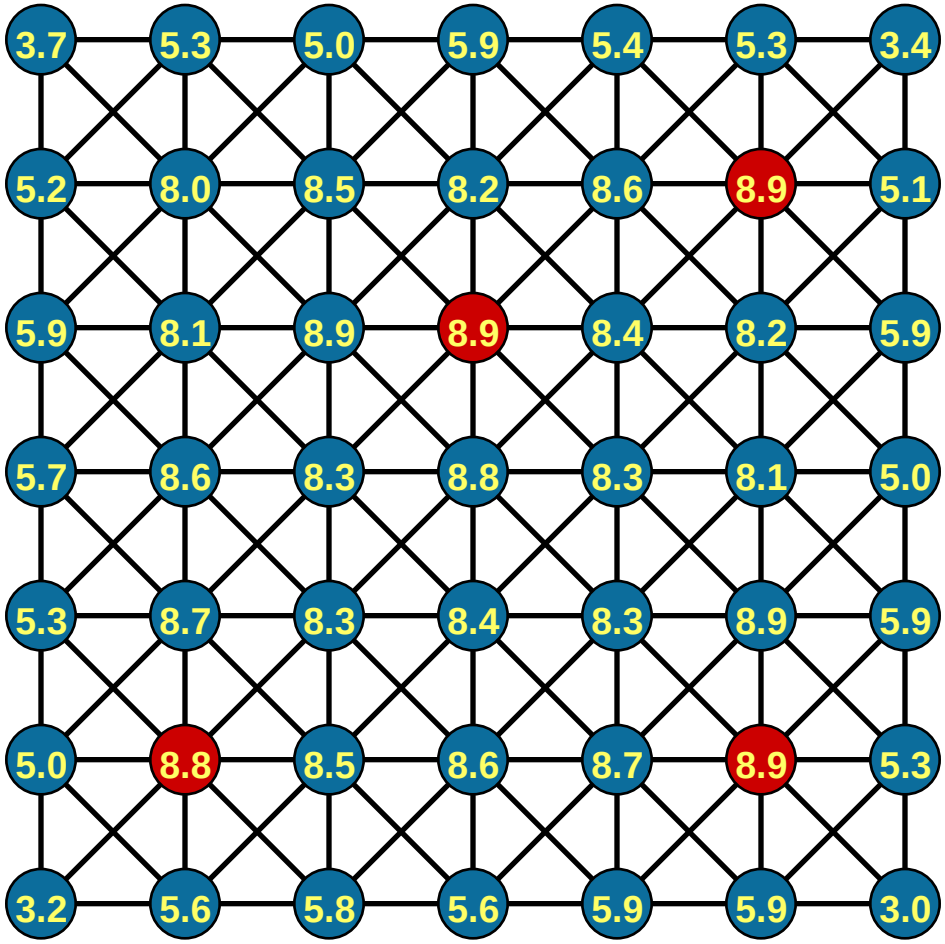
- Several parallel algorithms (in *hypre*):
  - CLJP (Cleary-Luby-Jones-Plassmann) – one-pass approach with random numbers to get concurrency (illustrated next)
  - Falgout – C-AMG on processor interior, then CLJP to finish
  - PMIS – CLJP without the 'C'; parallel version of C-AMG first pass
  - HMIS – C-AMG on processor interior, then PMIS to finish
  - CGC (Griebel, Metsch, Schweitzer) – compute several coarse grids on each processor, then solve a global graph problem to select the grids with the best "fit"
  - …
- Other parallel AMG codes use similar approaches

CLJP coarsening is fully parallel



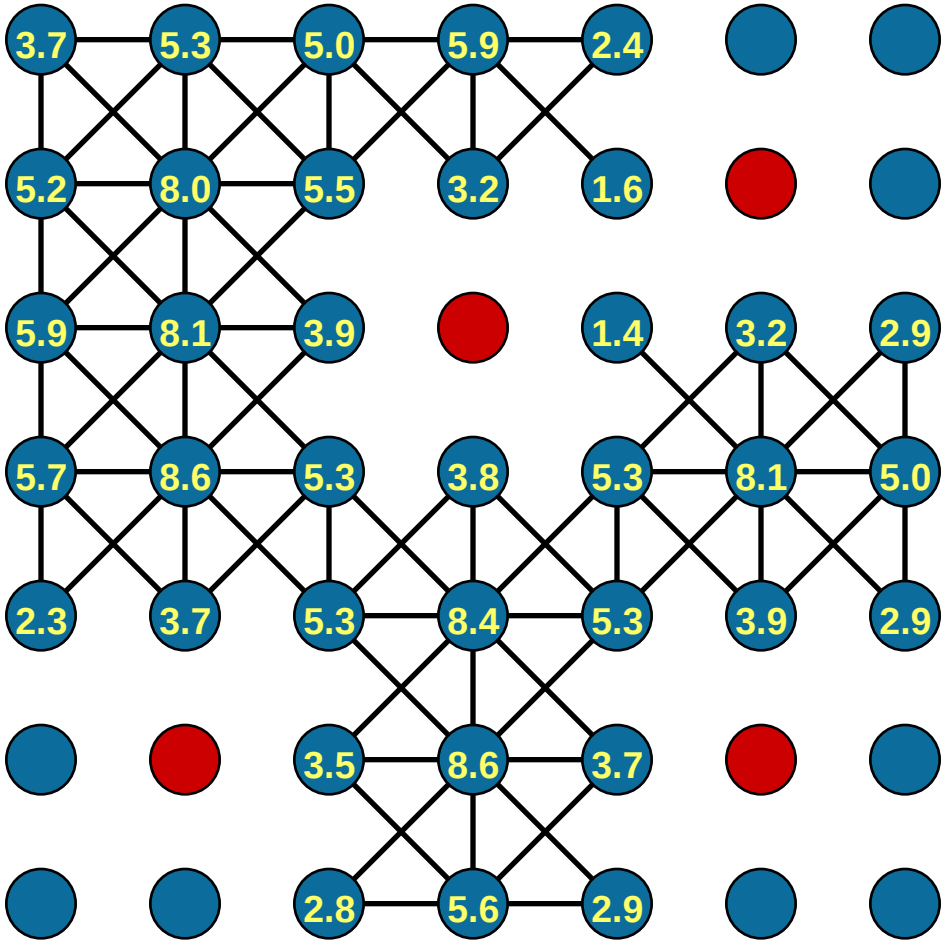
- select C-pts with maximal measure locally
- remove neighbor edges
- update neighbor measures

CLJP coarsening is fully parallel



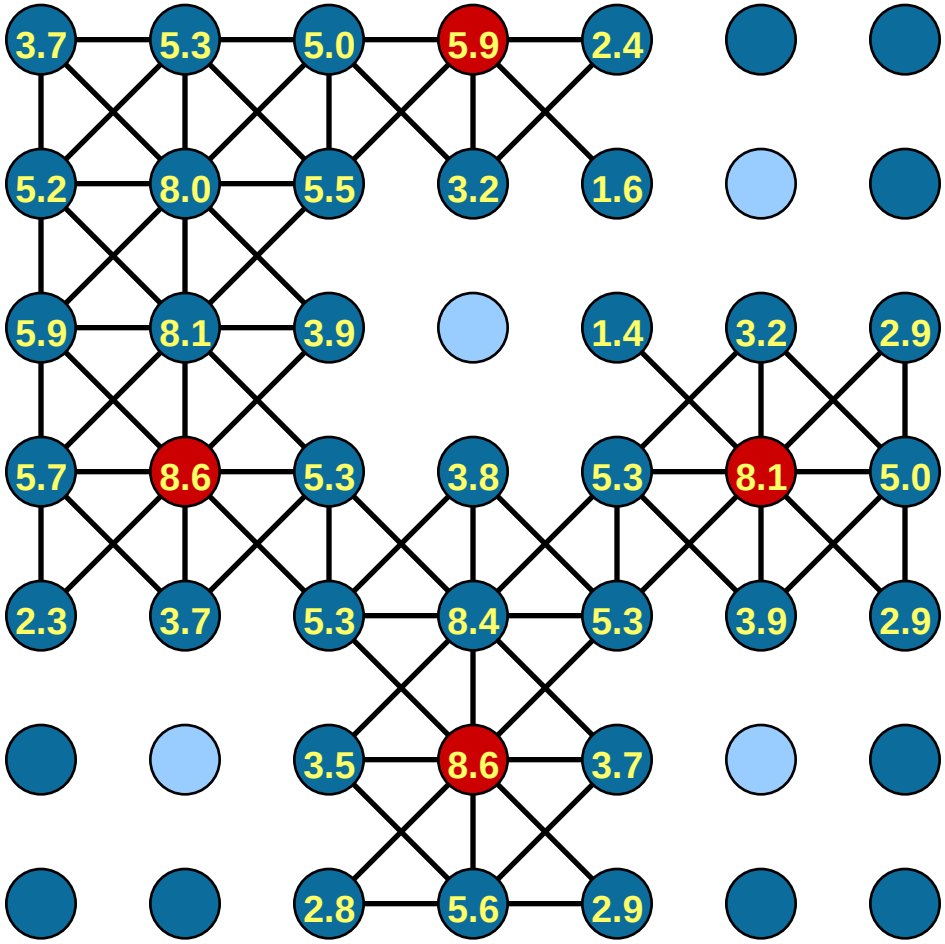
- select C-pts with maximal measure locally
- remove neighbor edges
- update neighbor measures

CLJP coarsening is fully parallel



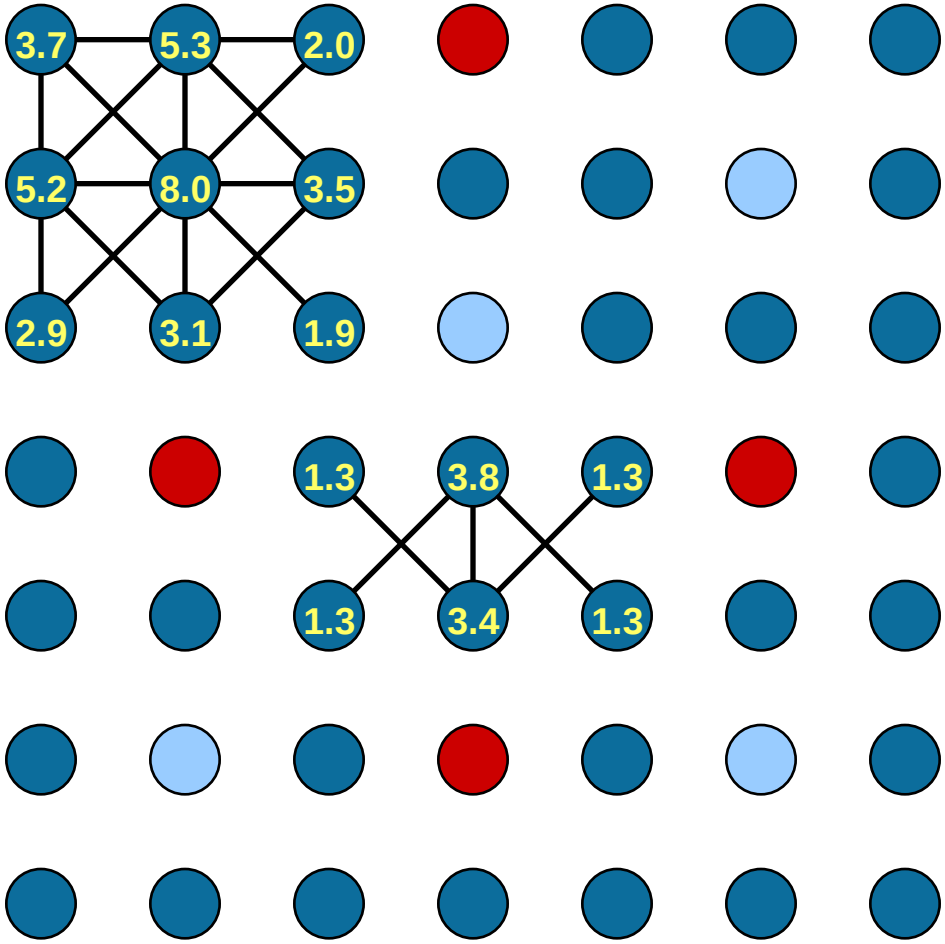
- ➔ **select C-pts with maximal measure locally**
- ➔ **remove neighbor edges**
- ➔ **update neighbor measures**

CLJP coarsening is fully parallel



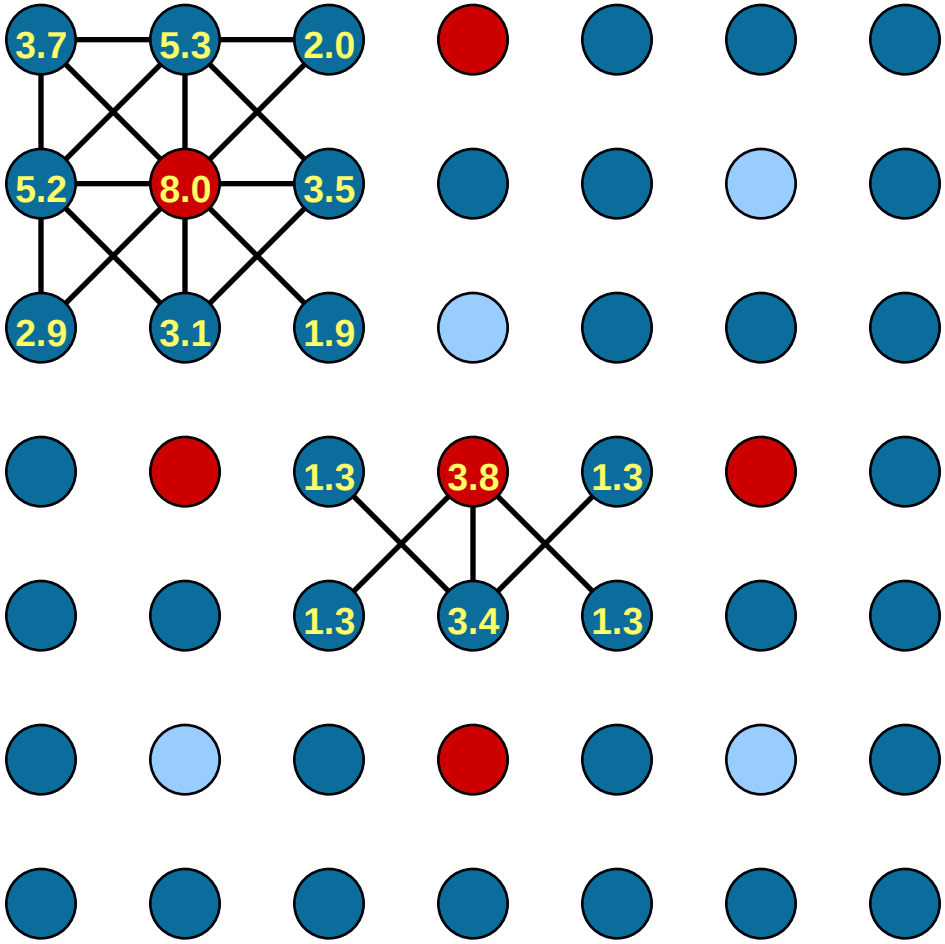
- ➔ **select C-pts with maximal measure locally**
- ➔ **remove neighbor edges**
- ➔ **update neighbor measures**

CLJP coarsening is fully parallel



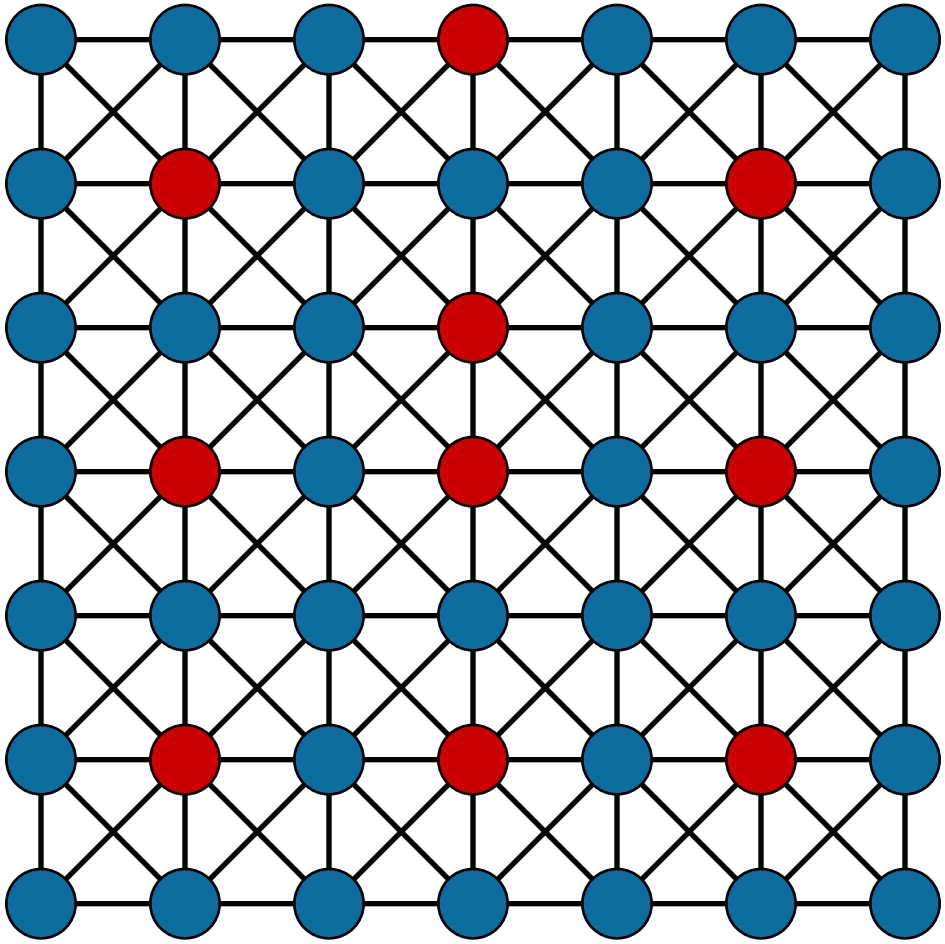
- select C-pts with maximal measure locally
- remove neighbor edges
- update neighbor measures

CLJP coarsening is fully parallel



- select C-pts with maximal measure locally
- remove neighbor edges
- update neighbor measures

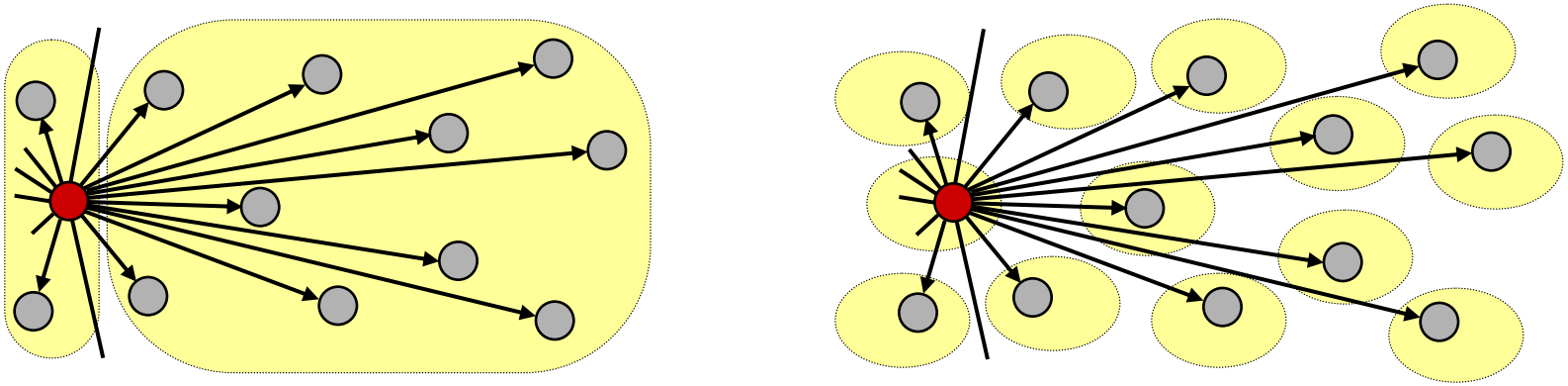
CLJP coarsening is fully parallel



- 10 C-points selected
- Standard AMG selects 9 C-points

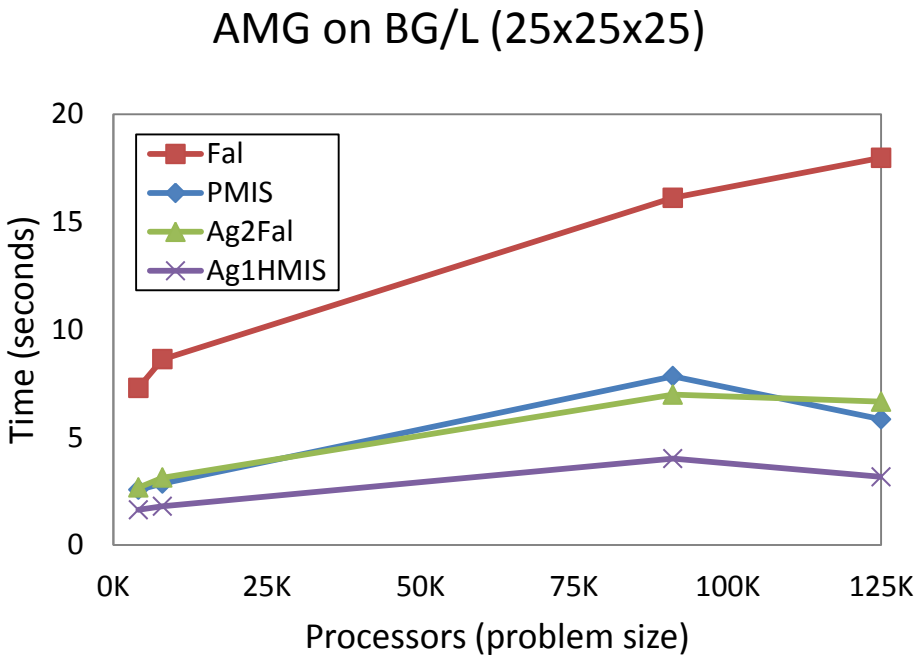
Parallel coarse-grid selection in AMG can produce unwanted side effects

- Non-uniform grids can lead to increased operator complexity and poor convergence
- Operator “stencil growth” reduces parallel efficiency



- Currently no guaranteed ways to control complexity
- Can ameliorate with more aggressive coarsening
- Requires long-range interpolation approaches

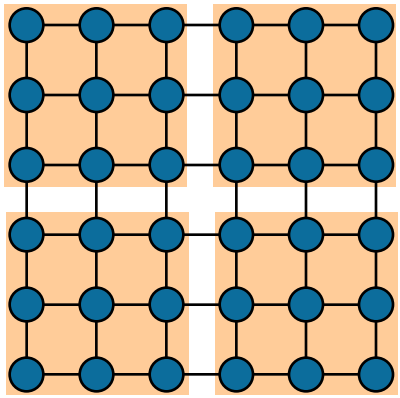
Parallel AMG in hypre now scales to 130K processors on BG/L … and beyond



- Largest problem above: 2B unknowns
- Largest problem to date: 26B unknowns on 98K processors of BG/L
- Most processors to date: 16B unknowns on 196K cores of Jaguar (Cray XT5 at ORNL)

SA builds interpolation by first chopping up a global basis, then smoothing it

- Tentative interpolation is constructed from “aggregates” (local QR factorization is used to orthonormalize)



$$\hat{P} \quad B_c \quad B$$

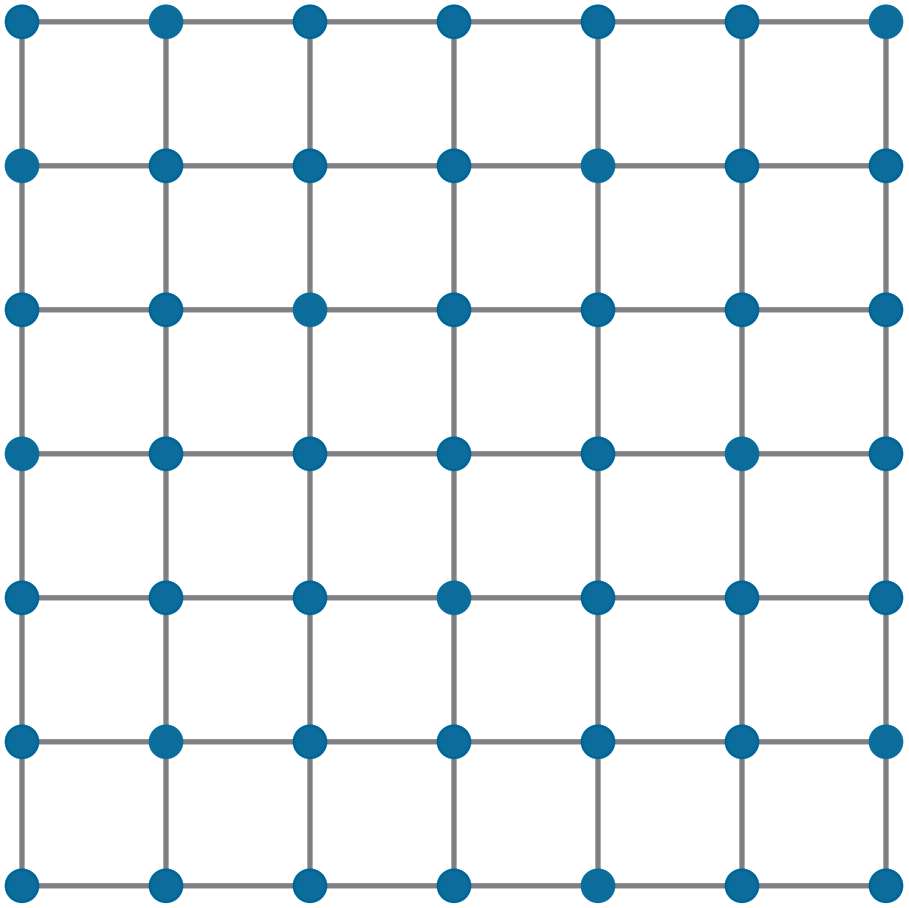
$$\begin{bmatrix} \\ \\ \\ \\ \\ \end{bmatrix} \begin{bmatrix} \\ \\ \\ \end{bmatrix} = \begin{bmatrix} \\ \\ \\ \\ \\ \end{bmatrix}$$

$$; \quad \hat{P}^T \hat{P} = I$$

- Smoothing adds basis overlap and improves approximation property

$$P = S\hat{P}$$

SA coarsening (5-pt Laplacian)



Phase 1:

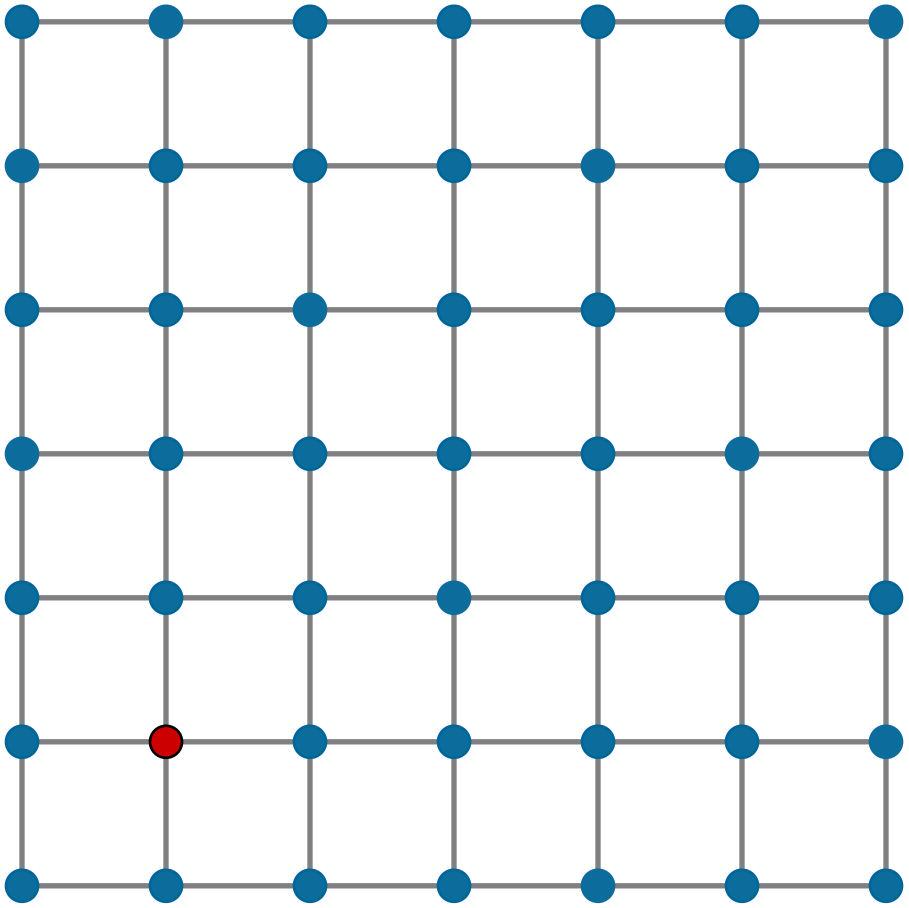
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

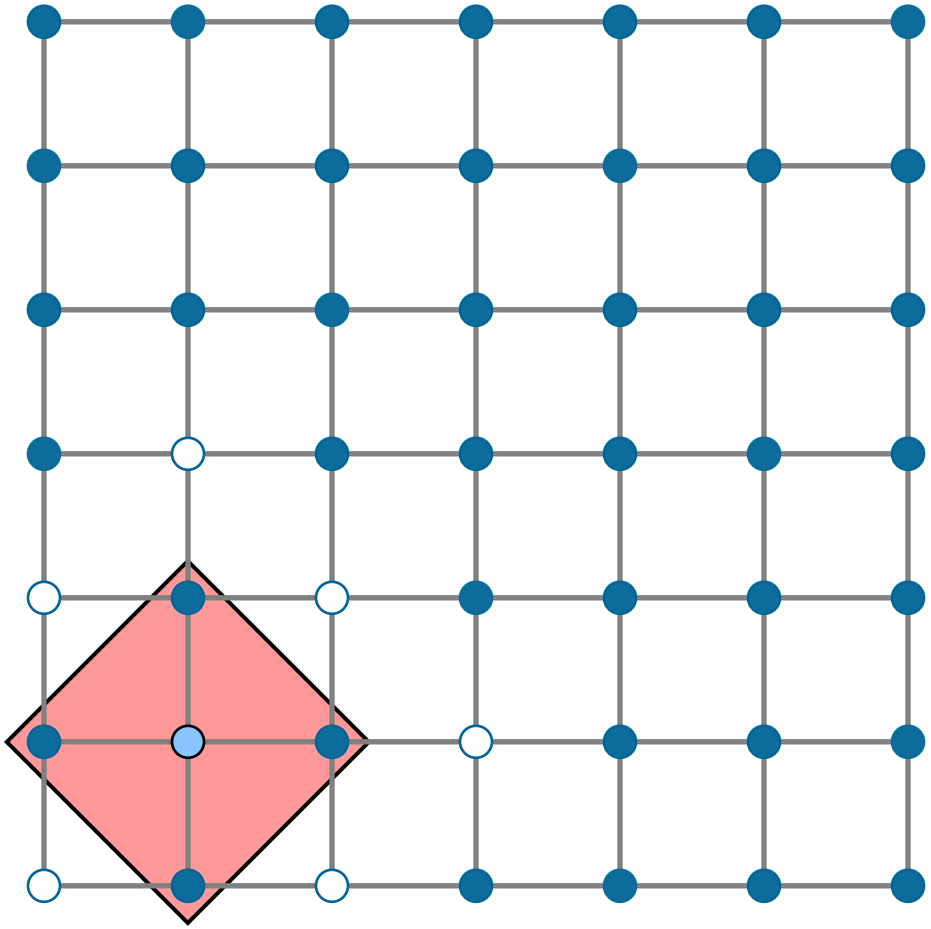
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

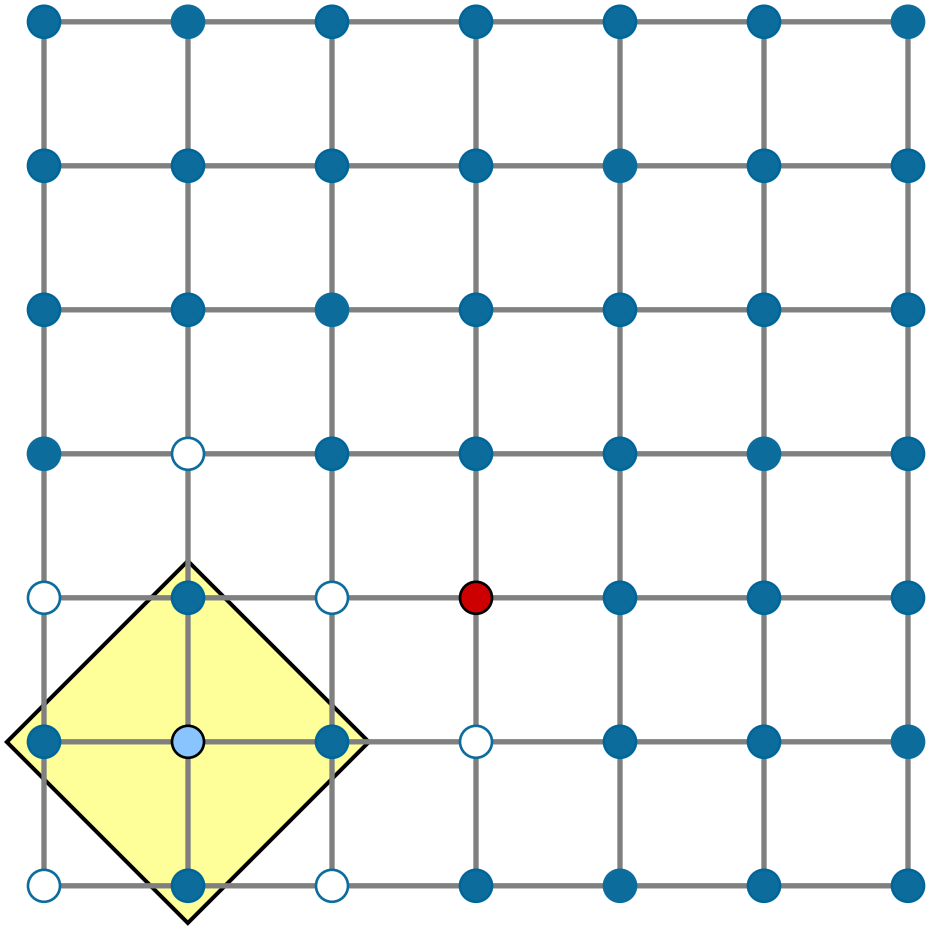
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

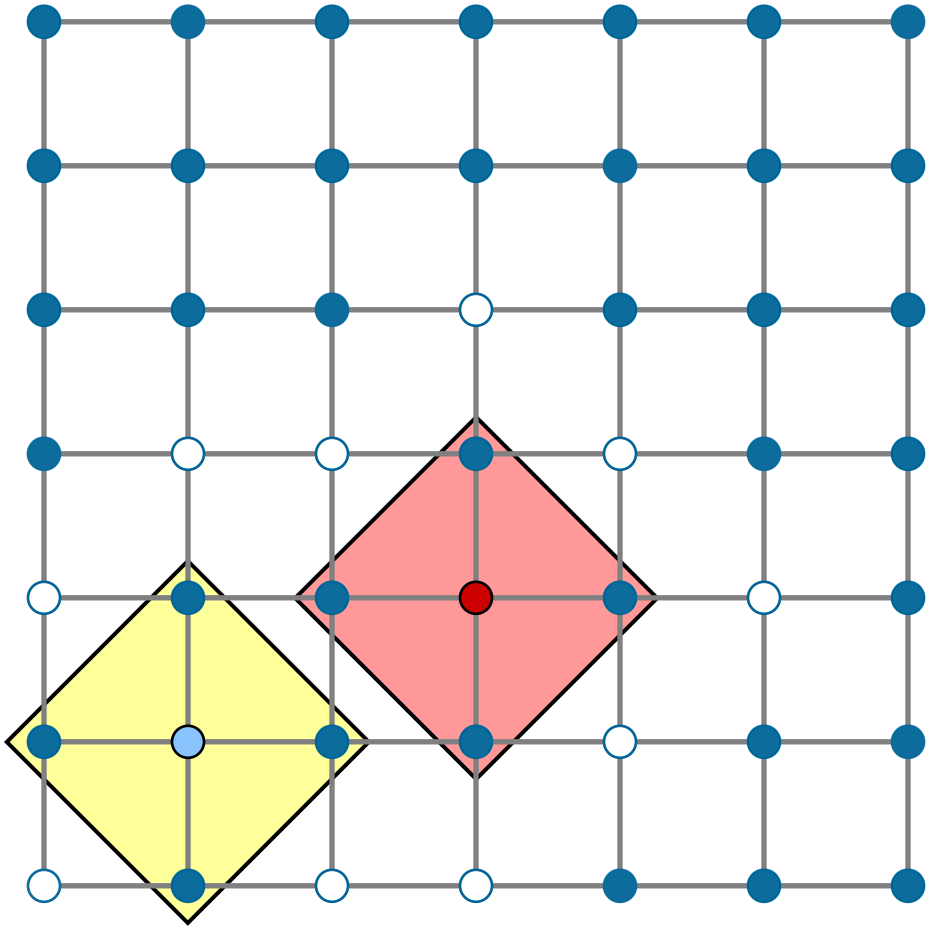
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

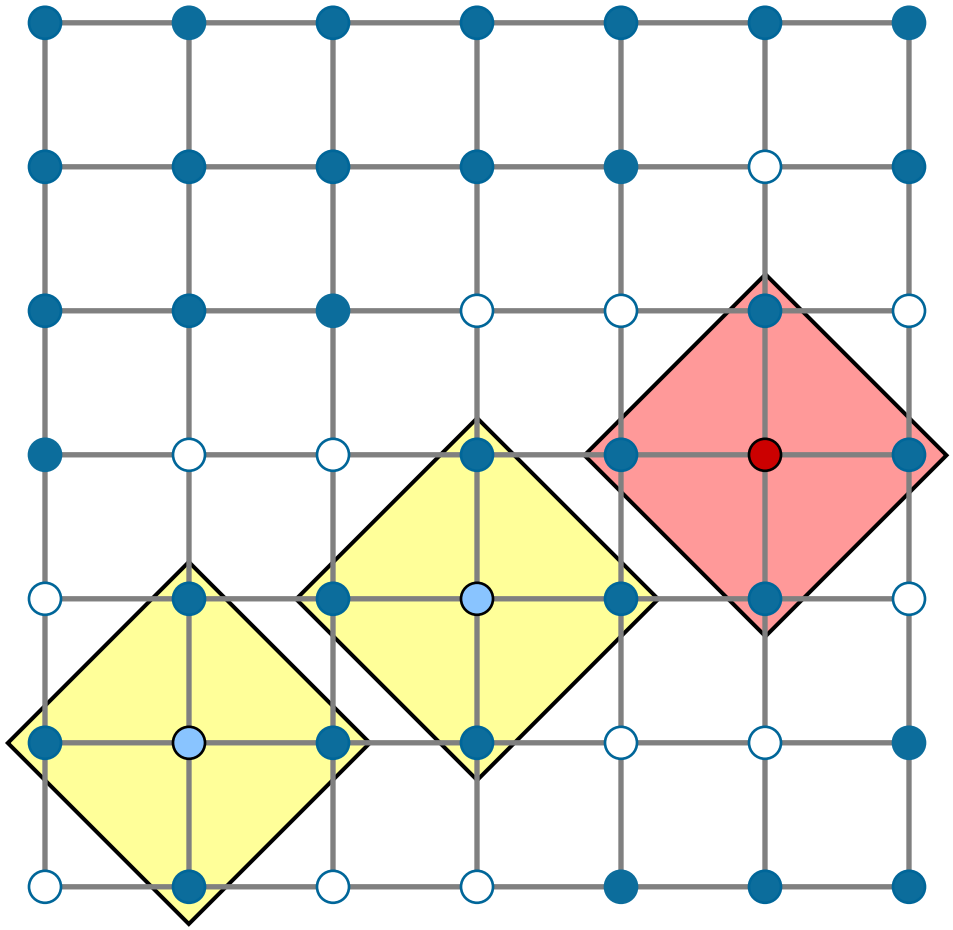
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

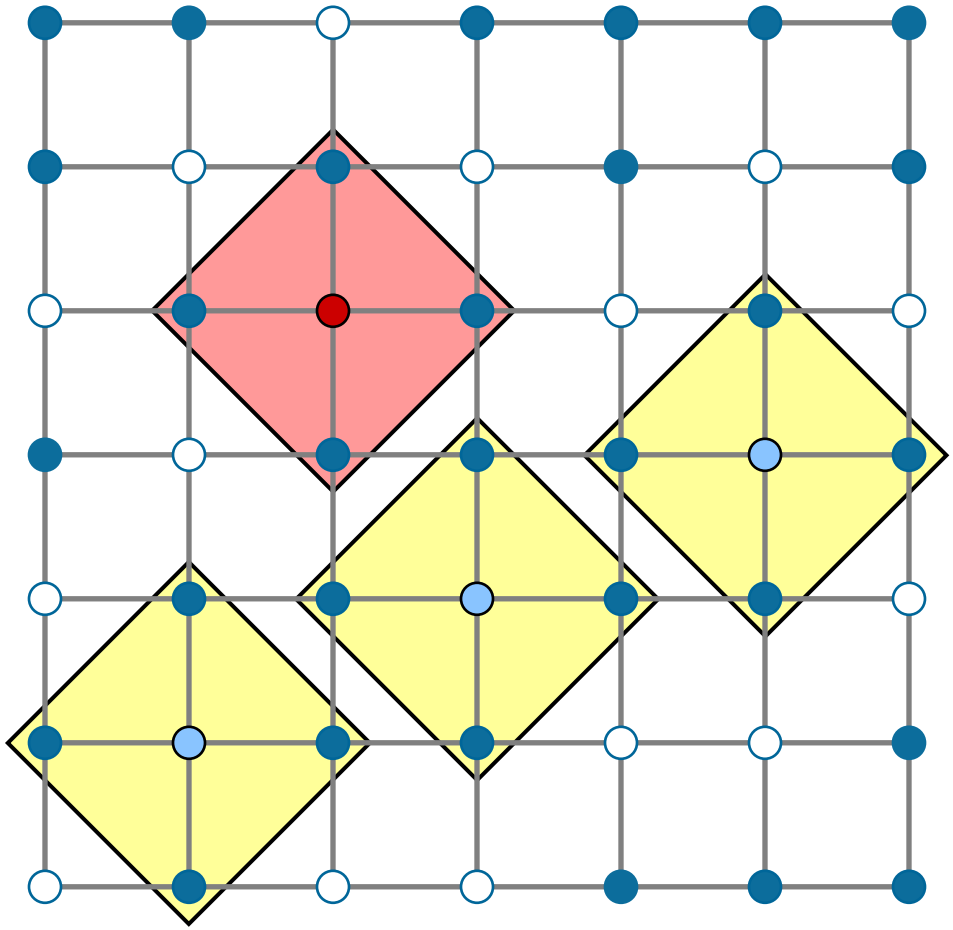
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

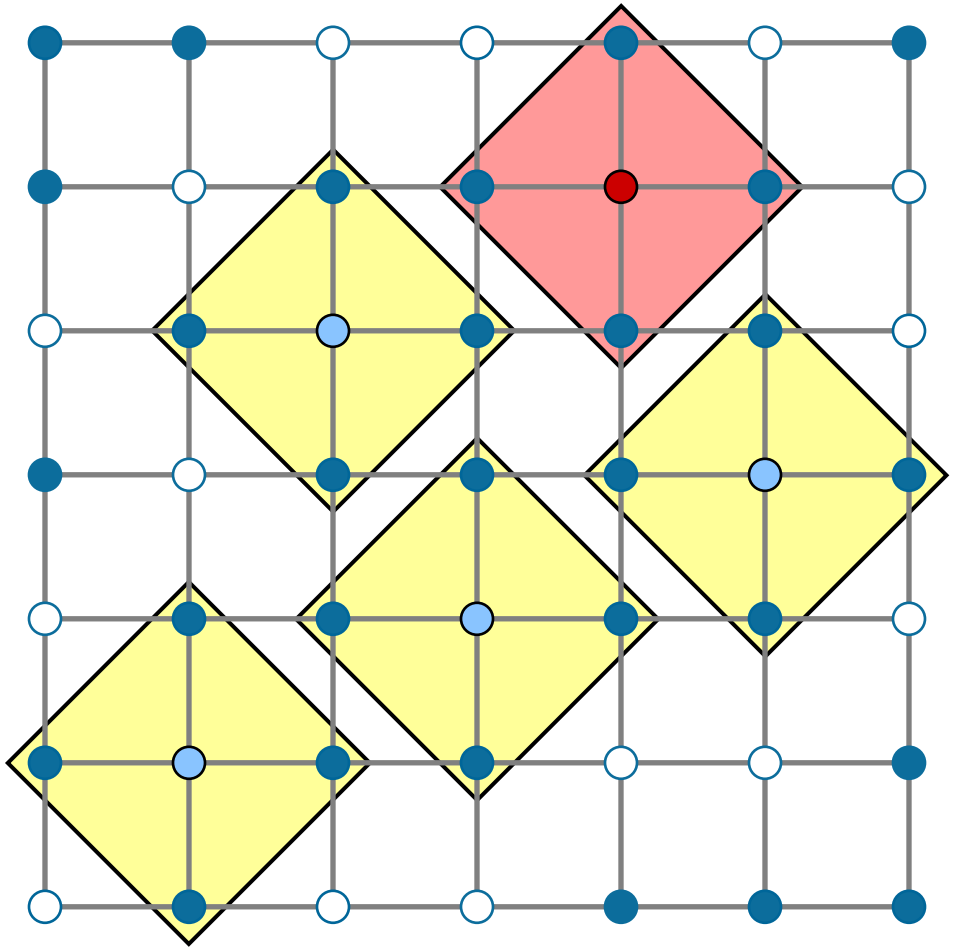
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

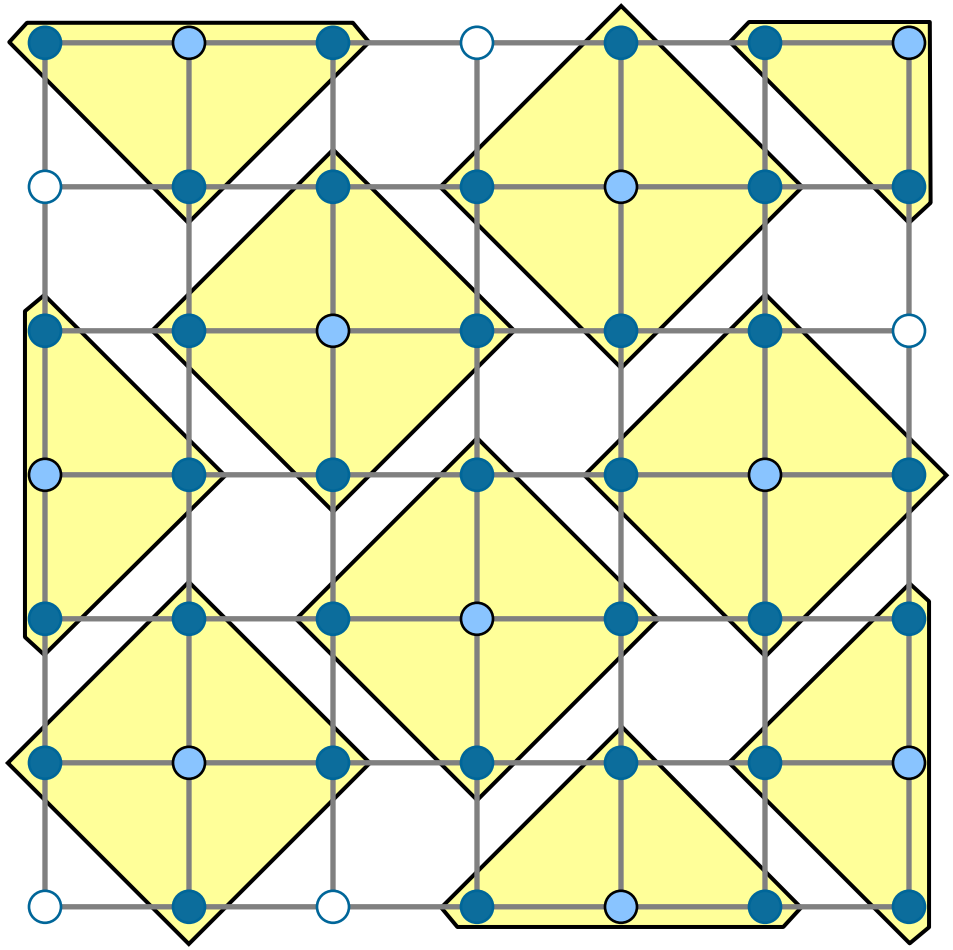
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

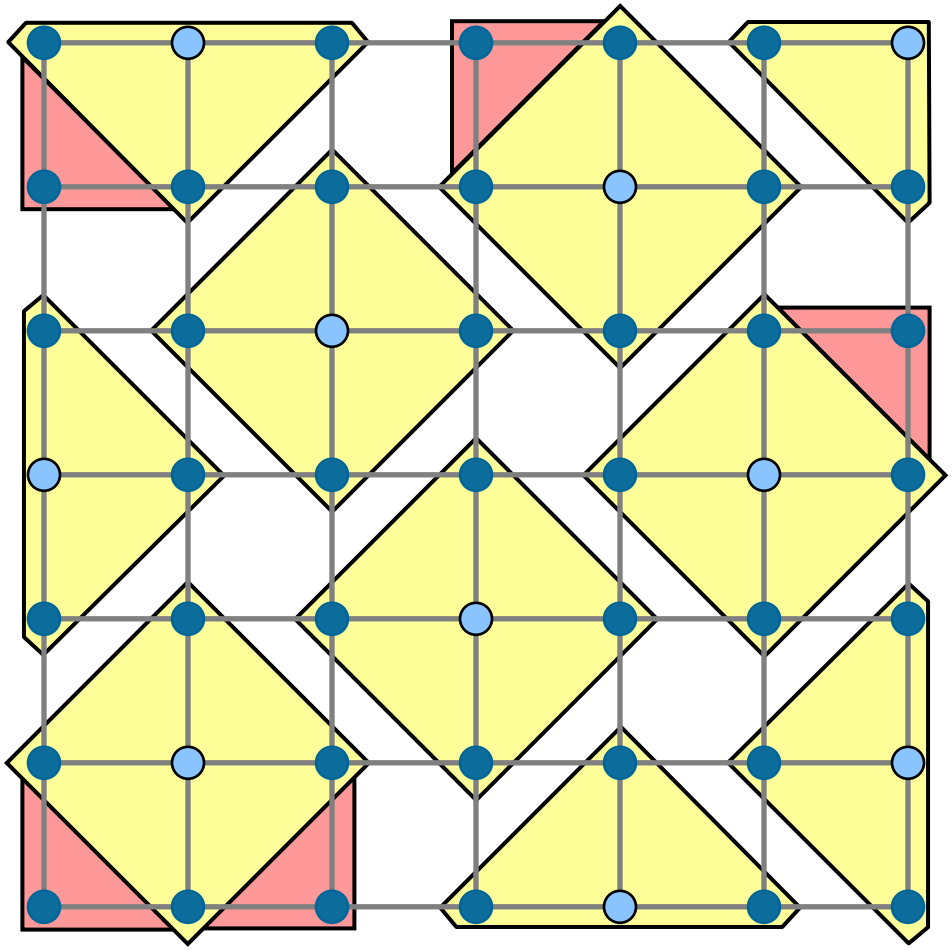
a) Pick root pt not adjacent to agg

b) Aggregate root and neighbors

Phase 2:

Move pts into nearby aggs or new aggs

SA coarsening (5-pt Laplacian)



Phase 1:

a) Pick root pt not adjacent to agg

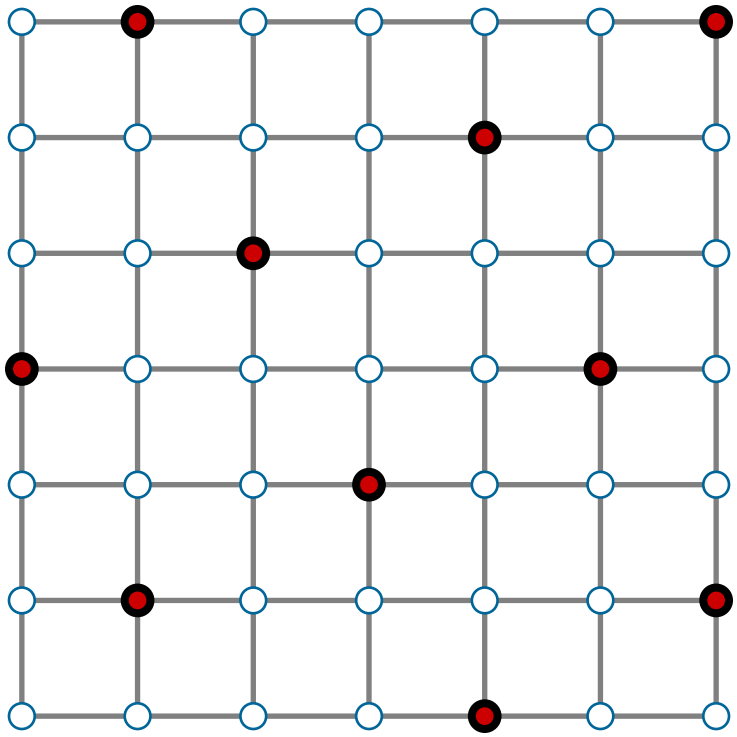
b) Aggregate root and neighbors

Phase 2:

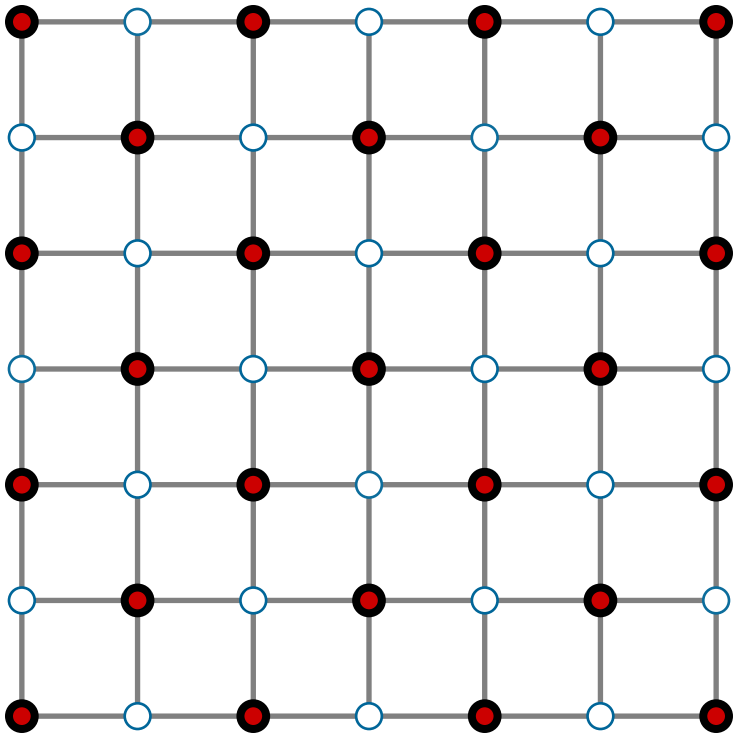
Move pts into nearby aggs or new aggs

SA coarsening is traditionally more aggressive than C-AMG coarsening (5-pt Laplacian example)

SA Seed Points (10)



C-AMG Grid (25)



Operator complexities are usually smaller, too

Additional comments on SA…

- Usual prolongator smoother is damped Jacobi
- Strength of connection is usually defined differently

$$|a_{ij}| > \theta \sqrt{|a_{ii} a_{jj}|}$$

- Special care must be taken for anisotropic problems to keep complexity low
  - Thresholded prolongator smoothing
  - Basis shifting approach
- Parallel SA coarsening has issues similar to C-AMG

AMG: The ideal prolongation and restriction

Reference: Wiesner, Tuminaro, Wall, Gee
Multigrid transfers for nonsymmetric systems based on Schur complements and Galerkin projections, *NLA, 2013*
AMG and the Schur complement

$$\begin{pmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{pmatrix} \begin{pmatrix} x_f \\ x_c \end{pmatrix} = \begin{pmatrix} b_f \\ b_c \end{pmatrix}.$$

Assuming A_{ff} to be invertible, A has the corresponding LDU decomposition

$$\begin{pmatrix} A_{ff} & A_{fc} \\ A_{cf} & A_{cc} \end{pmatrix} = \begin{pmatrix} I & 0 \\ A_{cf}A_{ff}^{-1} & I \end{pmatrix} \begin{pmatrix} A_{ff} & 0 \\ 0 & S \end{pmatrix} \begin{pmatrix} I & A_{ff}^{-1}A_{fc} \\ 0 & I \end{pmatrix}$$

where $S = A_{cc} - A_{cf}A_{ff}^{-1}A_{fc}$ and is referred to as the Schur complement.

Define

$$\mathcal{R}^{opt} = \begin{pmatrix} -A_{cf}A_{ff}^{-1} & I \end{pmatrix},\ \mathcal{P}^{opt} = \begin{pmatrix} -A_{ff}^{-1}A_{fc} \\ I \end{pmatrix} \quad \text{and} \quad \hat{I} = \begin{pmatrix} I \\ 0 \end{pmatrix}.$$

One can easily verify that $S = \mathcal{R}^{opt} A \mathcal{P}^{opt}$,

$$\begin{pmatrix} I & 0 \\ A_{cf}A_{ff}^{-1} & I \end{pmatrix}^{-1} = \begin{pmatrix} \hat{I}^T \\ \mathcal{R}^{opt} \end{pmatrix} \quad \text{and} \quad \begin{pmatrix} I & A_{ff}^{-1}A_{fc} \\ 0 & I \end{pmatrix}^{-1} = \begin{pmatrix} \hat{I} & \mathcal{P}^{opt} \end{pmatrix}.$$

Application of the inverses of the three operators in the exact factorization is equivalent to restriction at the c-points , followed by solution of two systems: A_{ff} which can be interpreted as relaxation and $\mathcal{R}^{opt} A \mathcal{P}^{opt}$ which is the coarse correction. Finally, the coarse correction is interpolated and added to the relaxation solution. As this procedure is exact, it converges in one iteration.

Further work:
how to approximate $\mathcal{R}^{opt}$, $\mathcal{P}^{opt}$ and S, or rather the coarse correction $\mathcal{R}^{opt} A \mathcal{P}^{opt}$, which is nothing but $A_{cf} A_{ff}^{-1} A_{fc}$.

We enter the full block factorized preconditioning framework, that can be seen as purely algebraic and not related to MG.

Algebraic Multilevel Iteration Methods (AMLI)

The so-called AMLI methods have been developed by Owe Axelsson and Panayot Vassilevski in a series of papers betwee 1989 and 1991.
These methods were originally developed for elliptix problems and spd matrices, and are the first regularity-free optimal order preconditioning methods.

Sequence of matrices $\left\{A^{(k)}\right\}_{k=k_0}^{\ell}$

$$N_{k_0} \subset N_{k_0+1} \subset \ldots \subset N_\ell$$

$$A^{(k)} = \begin{bmatrix} A_{11}^{(k)} & A_{12}^{(k)} \\ \\ A_{21}^{(k)} & A_{22}^{(k)} \end{bmatrix} \begin{matrix} \} \; N_k \backslash N_{k-1} \\ \\ \} \; N_{k-1} \end{matrix} .$$

$A^{(k)}$ has to approximate $S_{A^{(k+1)}}$ in some way. For instance,

$$A^{(k)} = A_{22}^{(k+1)} - A_{21}^{(k+1)} B_{11}^{(k+1)} A_{12}^{(k+1)}.$$

where $B_{11}^{(k+1)}$ is some sparse, positive definite, nonnegative and symmetric approximation of ${A_{11}^{(k+1)}}^{-1}$.
How to split N_{k+1} into two parts: the order n_k of the matrices $A^{(k)}$ should decrease geometrically:

$$\frac{n_{k+1}}{n_k} = \rho_k \geq \rho > 1.$$

$M^{(k_0)} = A^{(k_0)}$,

for $k = k_0, k_0 + 1, \ldots \ell - 1$

$$M^{(k+1)} = \begin{bmatrix} A_{11}^{(k+1)} & 0 \\ \\ A_{21}^{(k+1)} & \widetilde{S}^{(k)} \end{bmatrix} \begin{bmatrix} I_1^{(k+1)} & {A_{11}^{(k+1)}}^{-1} A_{12}^{(k+1)} \\ \\ 0 & I_2^{(k+1)} \end{bmatrix},$$

endfor

where $\widetilde{S}^{(k)}$ can be, for instance:

$$\widetilde{S}^{(k)} = A^{(k)} \left[I - P_\nu({M^{(k)}}^{-1} A^{(k)}) \right]^{-1},$$

$P_\nu(t)$ denotes a polynomial of degree ν.

We could use some other way of stabilization.

Forward sweep:

$$\text{Solve} \quad \begin{bmatrix} A_{11}^{(k+1)} & 0 \\ & \\ A_{21}^{(k+1)} & \widetilde{S}^{(k)} \end{bmatrix} \begin{bmatrix} \mathbf{w}_1 \\ \\ \mathbf{w}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{y}_1 \\ \\ \mathbf{y}_2 \end{bmatrix}, \qquad \text{i.e.}$$

$$\begin{aligned} (F1) \quad \mathbf{w}_1 &= A_{11}^{(k+1)^{-1}} \mathbf{y}_1, \\ (F2) \quad \mathbf{w}_2 &= \widetilde{S}^{(k)^{-1}} \left(\mathbf{y}_2 - A_{21}^{(k+1)} \mathbf{w}_1 \right). \end{aligned}$$

Backward sweep:

$$\text{Solve} \quad \begin{bmatrix} I_1^{(k+1)} & A_{11}^{(k+1)^{-1}} A_{12}^{(k+1)} \\ & \\ 0 & I_2^{(k+1)} \end{bmatrix} \begin{bmatrix} \mathbf{x}_1 \\ \\ \mathbf{x}_2 \end{bmatrix} = \begin{bmatrix} \mathbf{w}_1 \\ \\ \mathbf{w}_2 \end{bmatrix}, \qquad \text{i.e.}$$

$$\begin{aligned} (B1) \quad \mathbf{x}_2 &= \mathbf{w}_2, \\ (B2) \quad \mathbf{x}_1 &= \mathbf{w}_1 - A_{11}^{(k+1)^{-1}} A_{12}^{(k+1)} \mathbf{x}_2. \end{aligned}$$

Procedure $AMLI$: $\mathbf{u}^{(k)} \leftarrow AMLI\left(\mathbf{f}^{(k)}, k, \nu_k, \{a_j^{(k)}\}_{j=0}^{\nu_k}\right)$;

$[\mathbf{f}_1^{(k)}, \mathbf{f}_2^{(k)}] \leftarrow \mathbf{f}^{(k)}$,

$\mathbf{w}_1^{(k)} = B_{11}^{(k)}\, \mathbf{f}_1^{(k)}$,

$\mathbf{w}_2^{(k)} = \mathbf{f}_2^{(k)} - A_{21}^{(k)}\, \mathbf{w}_1^{(k)}$,

$k = k - 1$,

if $k = 0$ **then** $\mathbf{u}_2^{(0)} = A^{(0)}\, \mathbf{w}_2^{(1)}$, solve on the coarsest level exactly;

else

$\mathbf{u}_2^{(k)} \leftarrow AMLI\left(a_{\nu_k}^{(k)} \mathbf{w}_2^{(k)}, k, \nu_k, \{a_j^{(k)}\}_{j=0}^{\nu_k}\right)$;

for $j = 1$ **to** $\nu_k - 1$:

$\mathbf{u}_2^{(k)} \leftarrow AMLI\left(A^{(k)}\, \mathbf{u}_2^{(k)} + a_{\nu_k - j}^{(k)} \mathbf{w}_2^{(k)}, k, \nu_k, \{a_j^{(k)}\}_{j=0}^{\nu_k}\right)$;

endfor

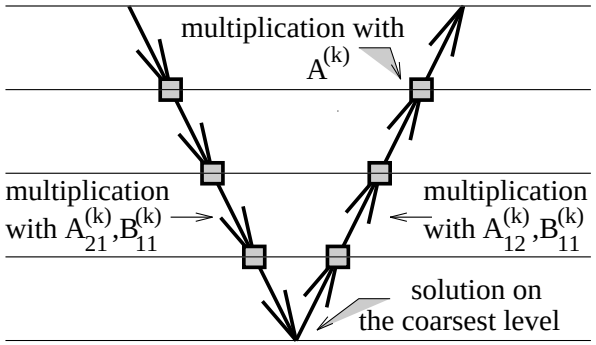
endif

$k = k + 1$,

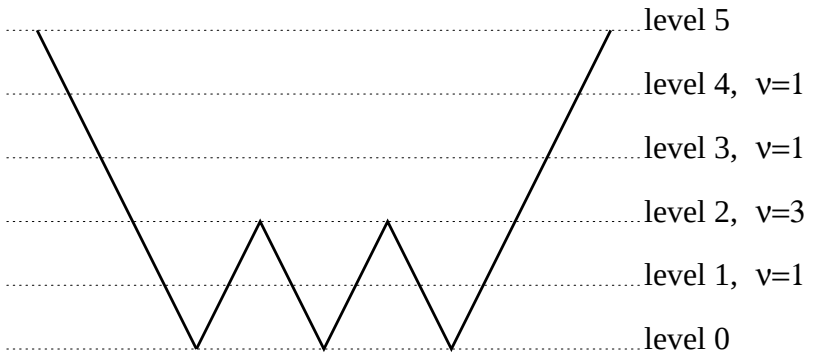
$\mathbf{u}_1^{(k)} = \mathbf{w}_1^{(k)} - B_{11}^{(k)} A_{12}^{(k)}\, \mathbf{u}_2^{(k)}$,

$\mathbf{u}^{(k)} \leftarrow [\mathbf{u}_1^{(k)}, \mathbf{u}_2^{(k)}]$

end Procedure $AMLI$



One AMLI step (V-cycle)



ν-fold W-cycle, $[1, 1, 3, 1]$

AMLI: Computational complexity

Level no.	*Polynomial degree/ inner iterations*
l	ν
l−1	*1*
...	...
l−m+1	*1*
l−m	ν
l−m−1	*1*
...	...
l−2m+1	*1*
...	
	ν
	1
...	

$$
\begin{aligned}
w_\ell = & C(n_\ell + \cdots + n_{\ell-\mu}) \\
& + C\nu(n_{\ell-\mu-1} + \cdots + n_{\ell-2\mu-1}) \\
& + C\nu^2(n_{\ell-2\mu-2} + \cdots + n_{\ell-3\mu-2}) \\
& + \cdots \\
\le & C n_\ell \left[1 + \tfrac{1}{\rho} + \cdots + \left(\tfrac{1}{\rho}\right)^\mu\right] \frac{1}{1 - \nu\rho^{-(\mu+1)}},
\end{aligned}
$$

where $1 < \rho \le \rho_k = \frac{n_{k+1}}{n_k}$, $k = 0, 1, \ldots \ell-1$. Hence

$$\nu < \rho^{\mu+1}$$

Time to try ready packages!