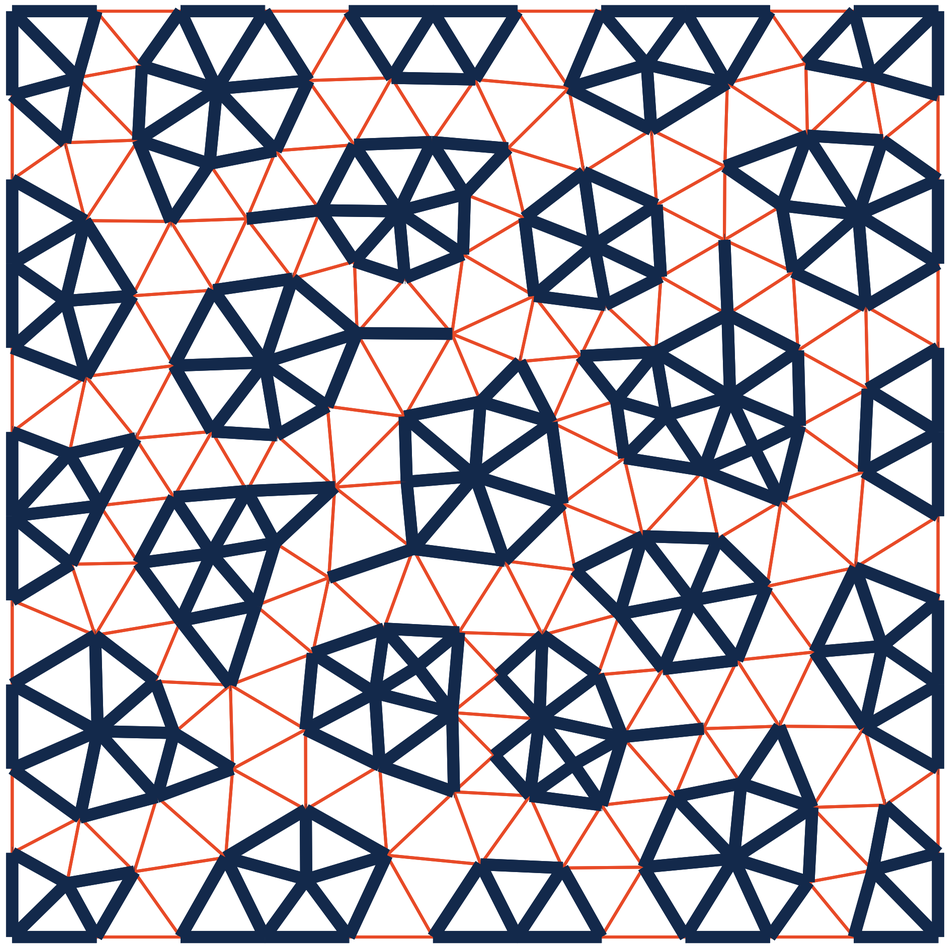
Algebraic Mutlgrid

Luke Olson
Department of Computer Science
University of Illinois at Urbana-Champaign

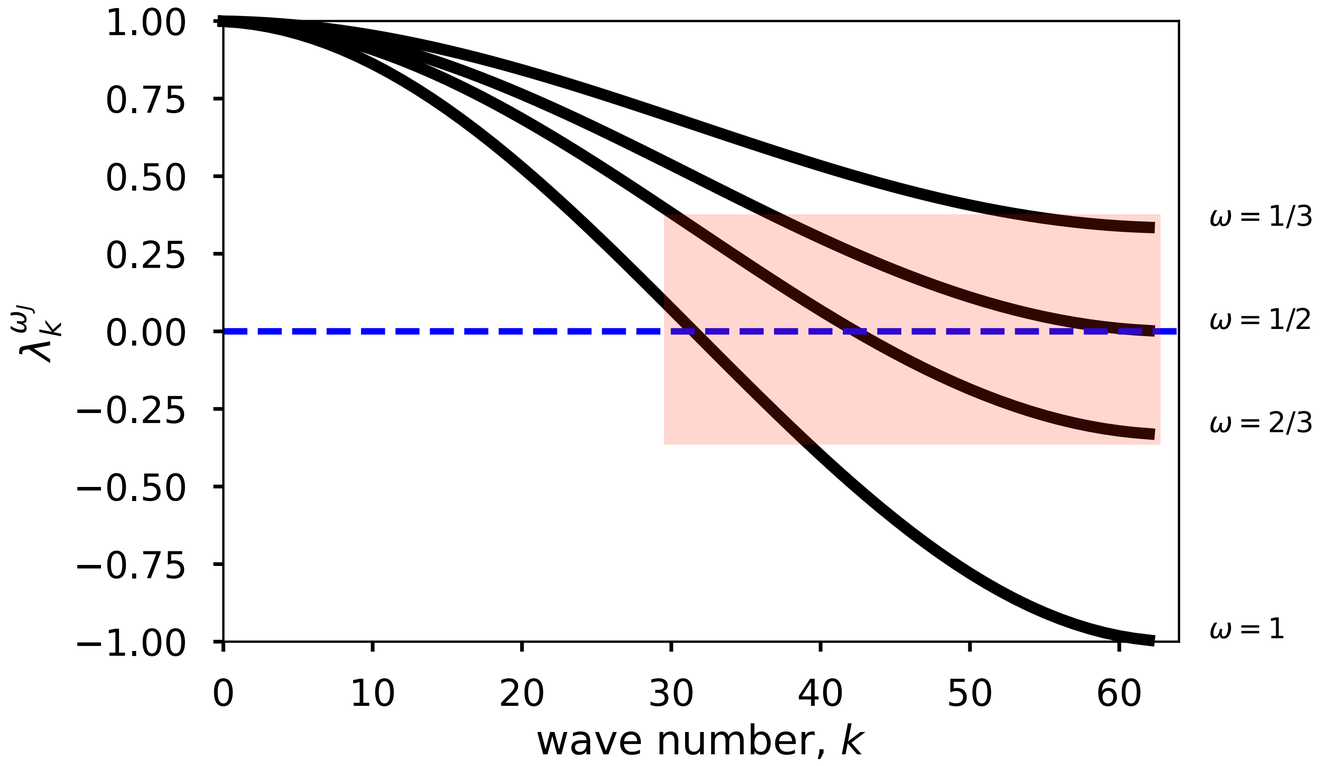
CS556 :: November 10, 2020

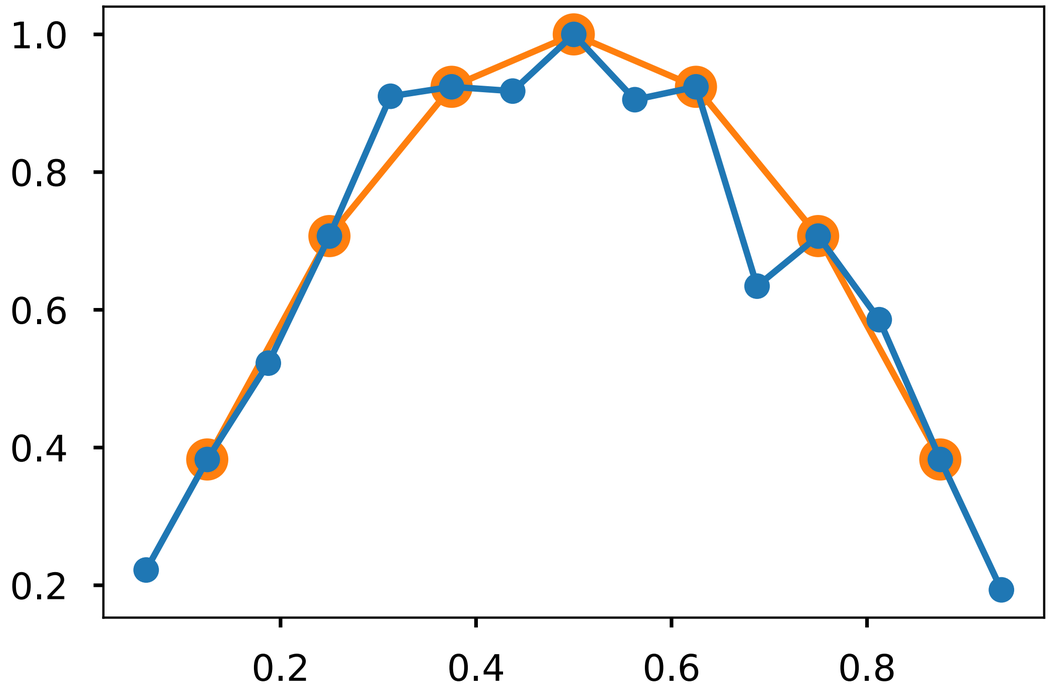


Objectives — today

1. **Algebraic multigrid**
   - Outline the basic components of an **algebraic** method
   - Compare and contrast different “styles” of AMG
   - Highlight advanced features such as interpolation

From last time …





- Smoothing:
  Reduce high frequency error

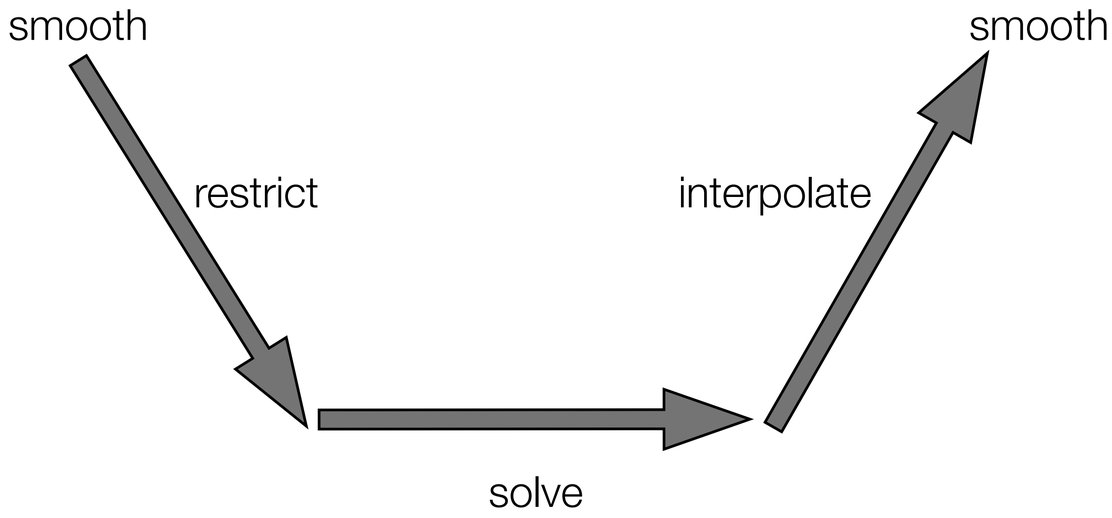
- Coarse-grid correction:
  Reduce smooth-ish things in the range of interpolation

$$e_1 = e_0 - P(P^T AP)^{-1} P^T AGe$$

Algorithm: two-level multigrid

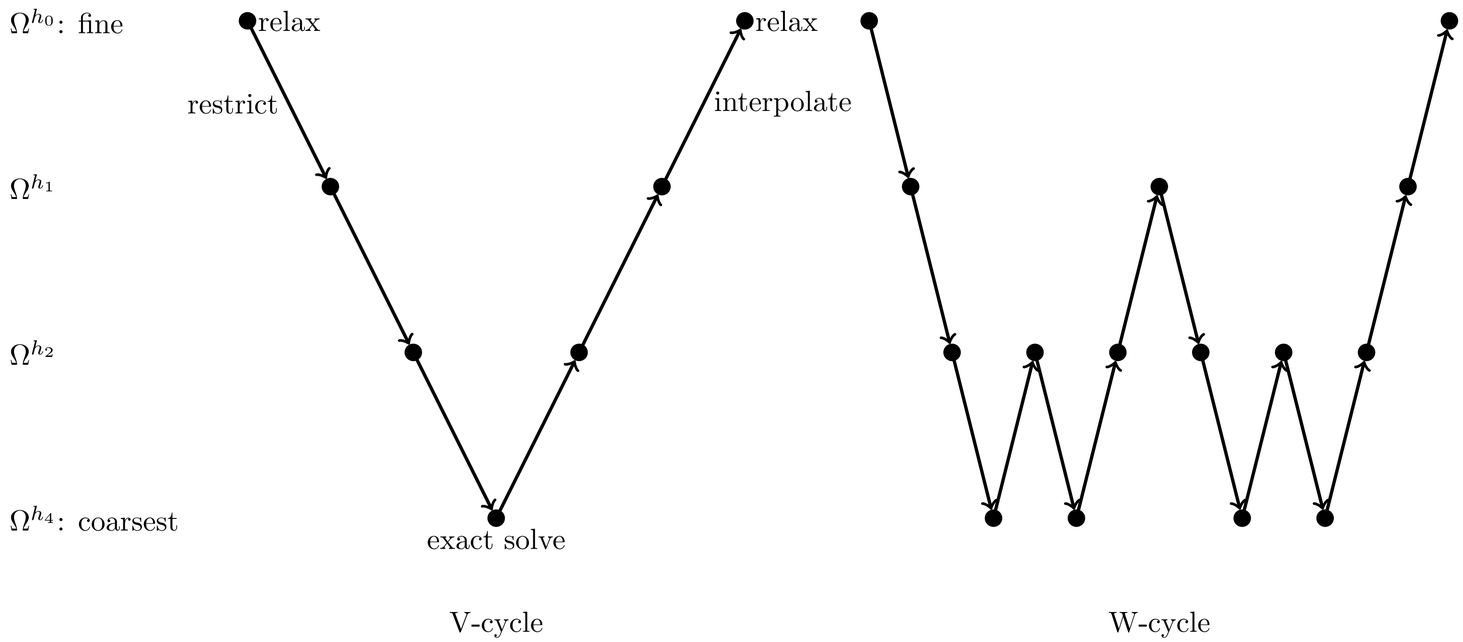
Input: initial guess

1. Smooth ν_{pre} times on $Au = f$
2. Compute $r = f - Au$
3. Compute $r_c = Rr$
4. Solve $A_c e_c = r_c$
5. Interpolate $\hat{e} = Pe_c$
6. Correct $u \leftarrow u + \hat{e}$
7. Smooth ν_{post} times on $Au = f$



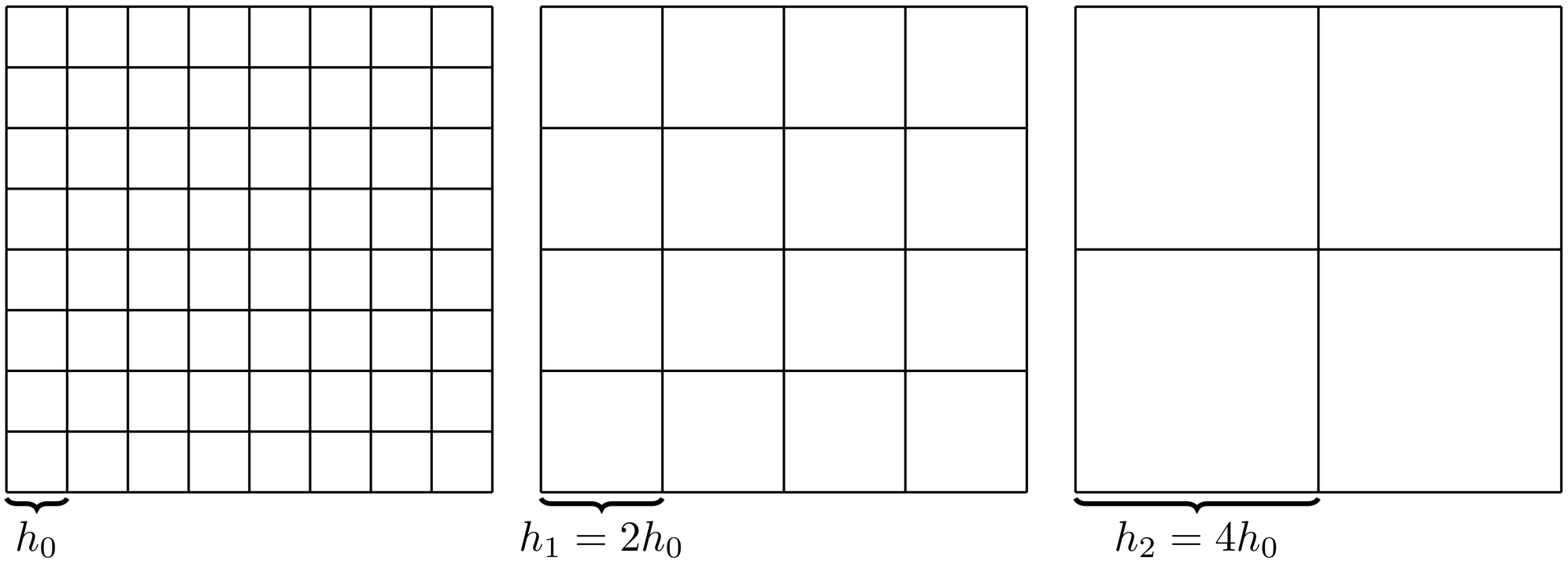
A two-level “V” cycle

The Multigrid V-Cycle and W-Cycle



- Two-grid cycle can expose issues with coarser interpolation
- W-Cycle can account for inadequate coarser level solves
- **Exact solve?** Usually a pseudo-inverse

Multigrid in 2D



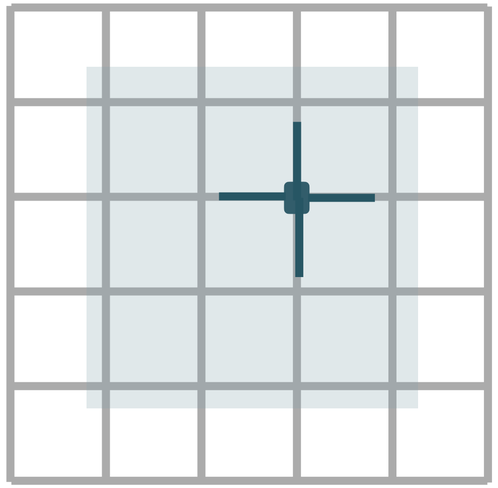
- Again, assume we have a series of uniform grids
- Relaxation remains the same (what is ω?)

Multigrid in 2D

- Model problem

$$-u_{xx} - u_{yy} = f$$
$$u = 0 \qquad \text{on boundary}$$

- Results in the stencil / matrix

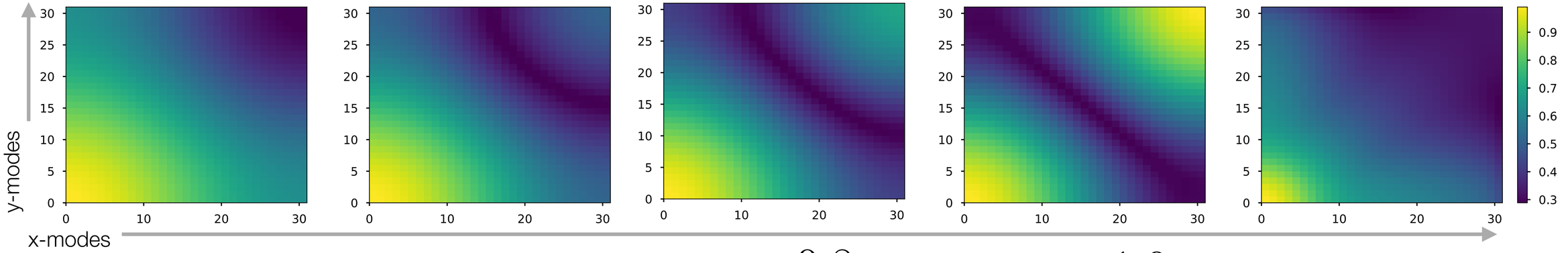


$$\begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix} \longrightarrow \frac{1}{h^2} \begin{bmatrix}
4 & -1 & & & -1 & & & & & & & & & & \\
-1 & 4 & -1 & & & -1 & & & & & & & & & \\
 & -1 & 4 & -1 & & & -1 & & & & & & & & \\
 & & -1 & 4 & & & & -1 & & & & & & & \\
-1 & & & & 4 & -1 & & & -1 & & & & & & \\
 & -1 & & & -1 & 4 & -1 & & & -1 & & & & & \\
 & & -1 & & & -1 & 4 & -1 & & & -1 & & & & \\
 & & & -1 & & & -1 & 4 & & & & -1 & & & \\
 & & & & -1 & & & & 4 & -1 & & & -1 & & \\
 & & & & & -1 & & & -1 & 4 & -1 & & & -1 & \\
 & & & & & & -1 & & & -1 & 4 & -1 & & & -1 \\
 & & & & & & & -1 & & & -1 & 4 & -1 & & \\
 & & & & & & & & -1 & & & -1 & 4 & -1 & \\
 & & & & & & & & & -1 & & & -1 & 4 &
\end{bmatrix}$$

Relaxation in 2D

$$\sin\left(\frac{k_x i\pi}{n+1}\right)\sin\left(\frac{k_y j\pi}{n+1}\right)$$

Convergence factor over 10 sweeps



$\omega = 0.5$ $\omega = 0.667$ $\omega = 0.8$ $\omega = 1.0$ GS

$I - \omega D^{-1} A$ $I - (D - E)^{-1} A$

- weighted Jacobi: Same issue — need to select a parameter
- Gauss-Seidel improved
- Red-Black Gauss-Seidel and other schemes even more effective

Interpolation in 2D

- Bilinear interpolation, tensor of 1D interpolation

$$\begin{bmatrix} 1 & 2 & 1 \end{bmatrix} \longrightarrow \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

- Example 3x3 -> 7x7

$$\begin{bmatrix} 0.5 & & \\ 1 & & \\ 0.5 & 0.5 & \\ & 1 & \\ & 0.5 & 0.5 \\ & & 1 \\ & & 0.5 \end{bmatrix} \otimes \begin{bmatrix} 0.5 & & \\ 1 & & \\ 0.5 & 0.5 & \\ & 1 & \\ & 0.5 & 0.5 \\ & & 1 \\ & & 0.5 \end{bmatrix} \longrightarrow \frac{1}{4} \begin{bmatrix} 1 & & & & & & \\ 2 & & & & & & \\ 1 & 1 & & & & & \\ & 2 & & & & & \\ & 1 & 1 & & & & \\ & & 2 & & & & \\ & & 1 & & & & \\ 2 & & & & & & \\ 4 & & & & & & \\ 2 & 2 & & & & & \\ & 4 & & & & & \\ & 2 & 2 & & & & \\ & & 4 & & & & \\ & & 2 & & & & \\ 1 & & & 1 & & & \\ 2 & & & 2 & & & \\ 1 & 1 & & 1 & 1 & & \\ & 2 & & & 2 & & \\ & 1 & 1 & & 1 & 1 & \\ & & 2 & & & 2 & \\ & & 1 & & & 1 & \\ & & & 2 & & & \\ & & & 4 & & & \\ & & & 2 & 2 & & \\ & & & & 4 & & \\ & & & & 2 & 2 & \\ & & & & & 4 & \\ & & & & & 2 & \\ & \vdots & & \vdots & & & \end{bmatrix}$$

Notebook

- `9-multigrid-2d.ipynb`

A few observations so far: **One**

- Let's consider a V(1,1) cycle — weighted Jacobi, etc.

- The **error** propagation for this looks like

$$e_1 = \underbrace{G(I - P(P^T A P)^{-1} P^T A) G e_0}_{M}$$

$$G = I - \omega D^{-1} A$$

$$M e_k \leftarrow 0?$$

- One thing we can do, is consider bounds on each operation.

A few observations so far: **One**

- Take the operator

$$M = G(I - P(P^T AP)^{-1} P^T A)G$$

- And makes some bounds

$$\|I - P(P^T AP)^{-1} P^T A\| \|G\|^2$$

approximation property | smoothing property

- General $\|G\| \leq 1$

  1D over [n/2,n]: $\|G\| \leq \frac{1}{3}$

  2D over [n/2,n]: $\|G\| \leq \frac{3}{5}$

A few observations so far: **One**

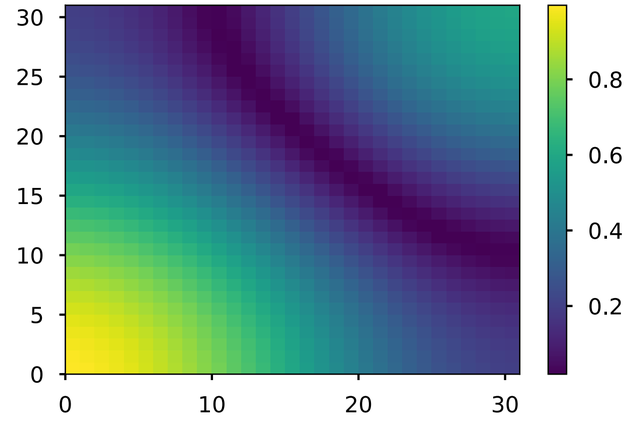
$$\|I - P(P^T AP)^{-1} P^T A\| \|G\|^2$$

approximation property | smoothing property

- General $\|G\| \leq 1$

1D over [n/2,n]: $\|G\| \leq \frac{1}{3}$

2D over [n/2,n]: $\|G\| \leq \frac{3}{5}$



- Also, if $w \quad \text{s.t.} \quad Aw \in \mathcal{N}(P^T)$

then $(I - P(P^T AP)^{-1} P^T A)w = w$

$$\|\cdot\| \geq 1$$

A few observations so far: **Two**

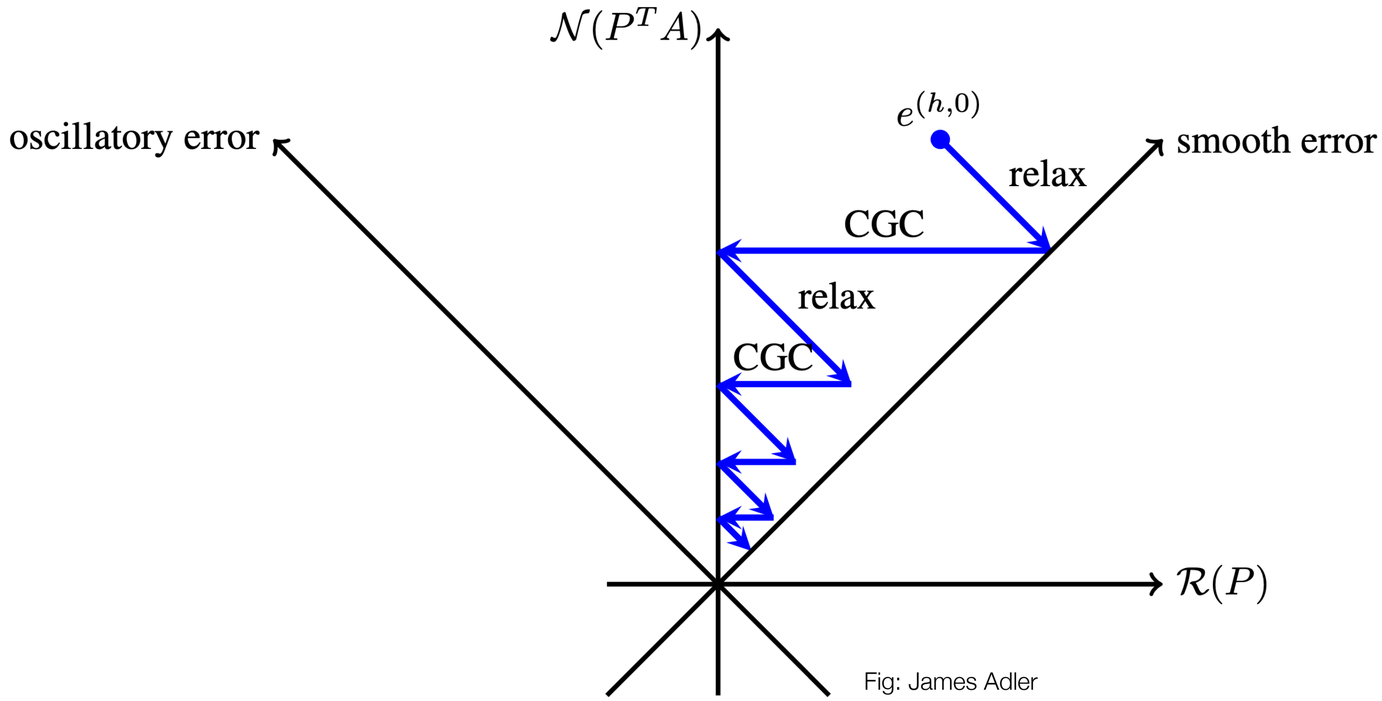


Fig: James Adler

- Complementary processes:
  - **relaxation:** targets for (Fourier) smoothing
  - **coarse grid correction:** targets things in the range of interpolation

A few observations so far: **Three**

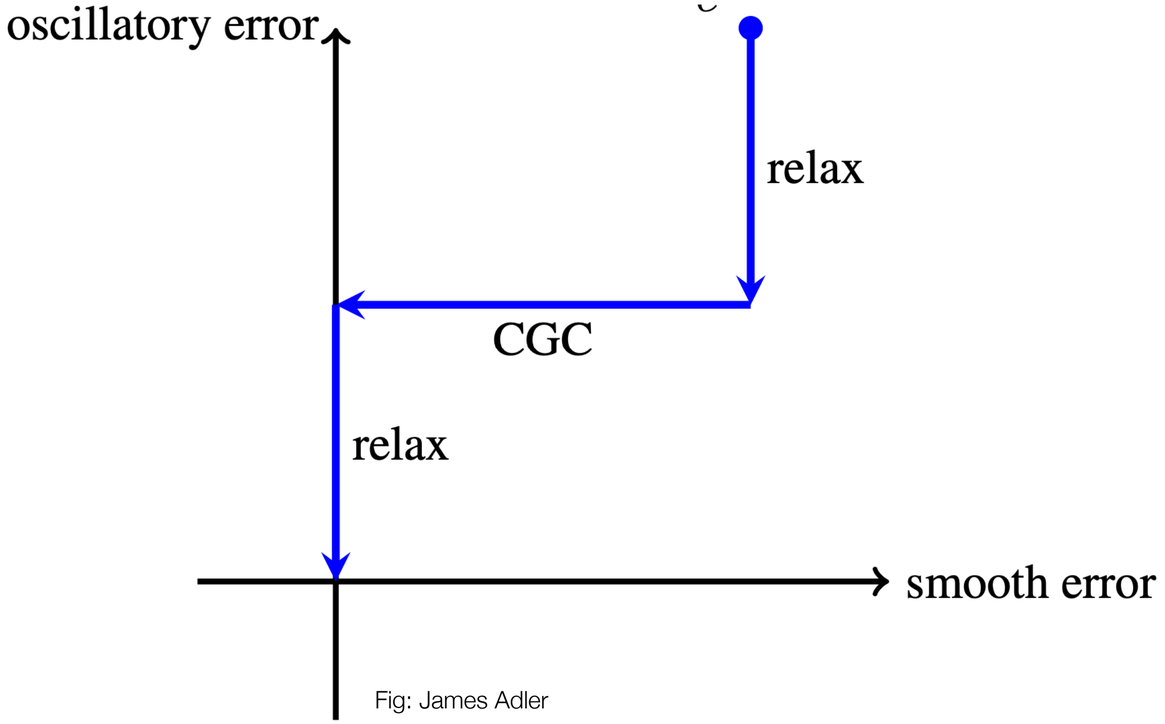


Fig: James Adler

$$e_1 \leftarrow (I - P(P^T AP)^{-1} P^T A) G e_0$$

$$Ge_0 \in \mathcal{R}(P) \quad \Rightarrow \quad e_1 = 0$$

interpolation should capture what relaxation misses

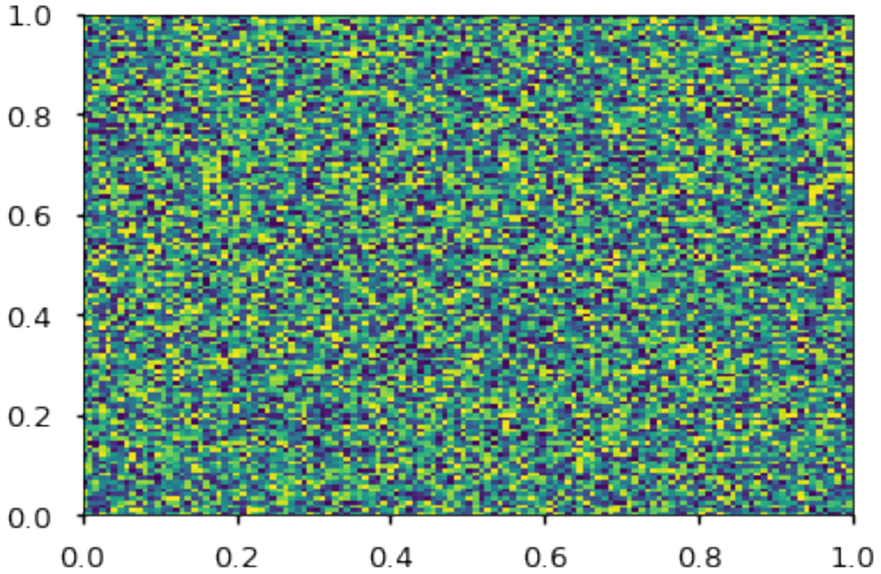
What can go wrong?!

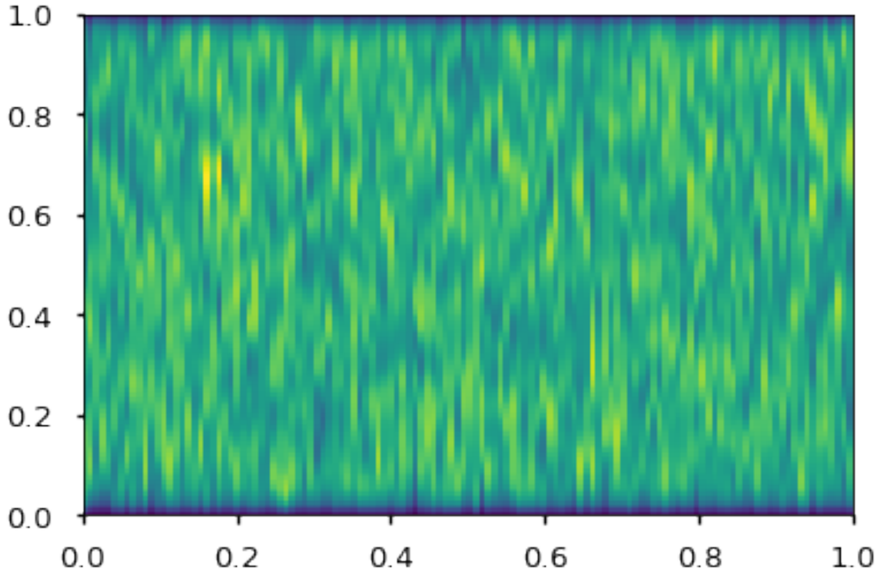
- Demo: `10-multigrid-anisotropy.ipynb`

Consider an *anisotropic problem*

$$-\varepsilon u_{xx} - u_{yy} = f$$
$$u = 0 \qquad \text{on boundary}$$

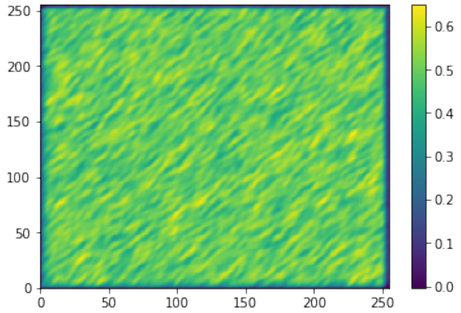
$$\begin{bmatrix} & -1 & \\ -\varepsilon & 2+2\varepsilon & -\varepsilon \\ & -1 & \end{bmatrix}$$

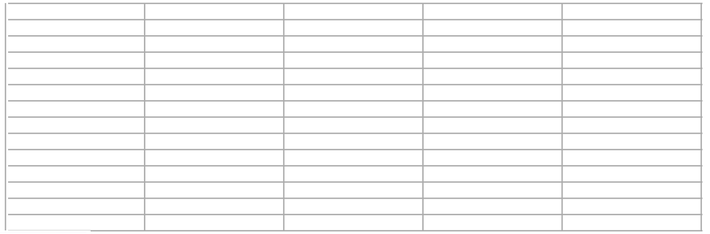


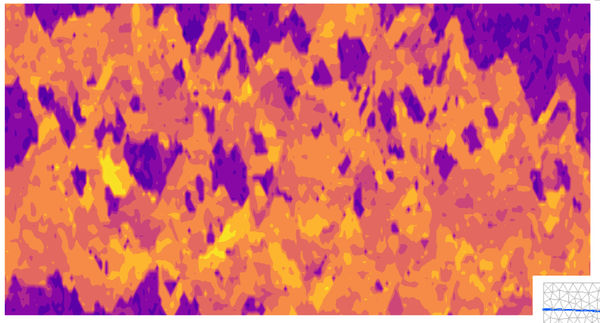


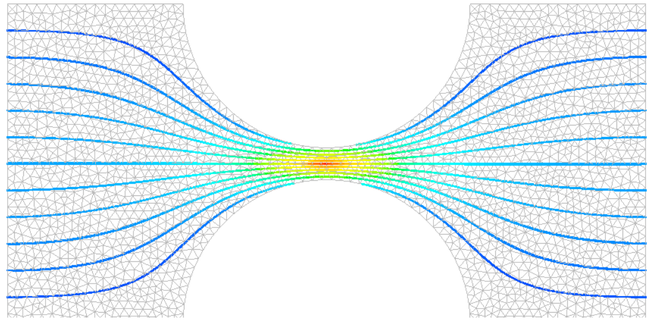
What can go wrong?!

- Anisotropy
- Mesh stretching
- Jumping coefficients
- Non-elliptic



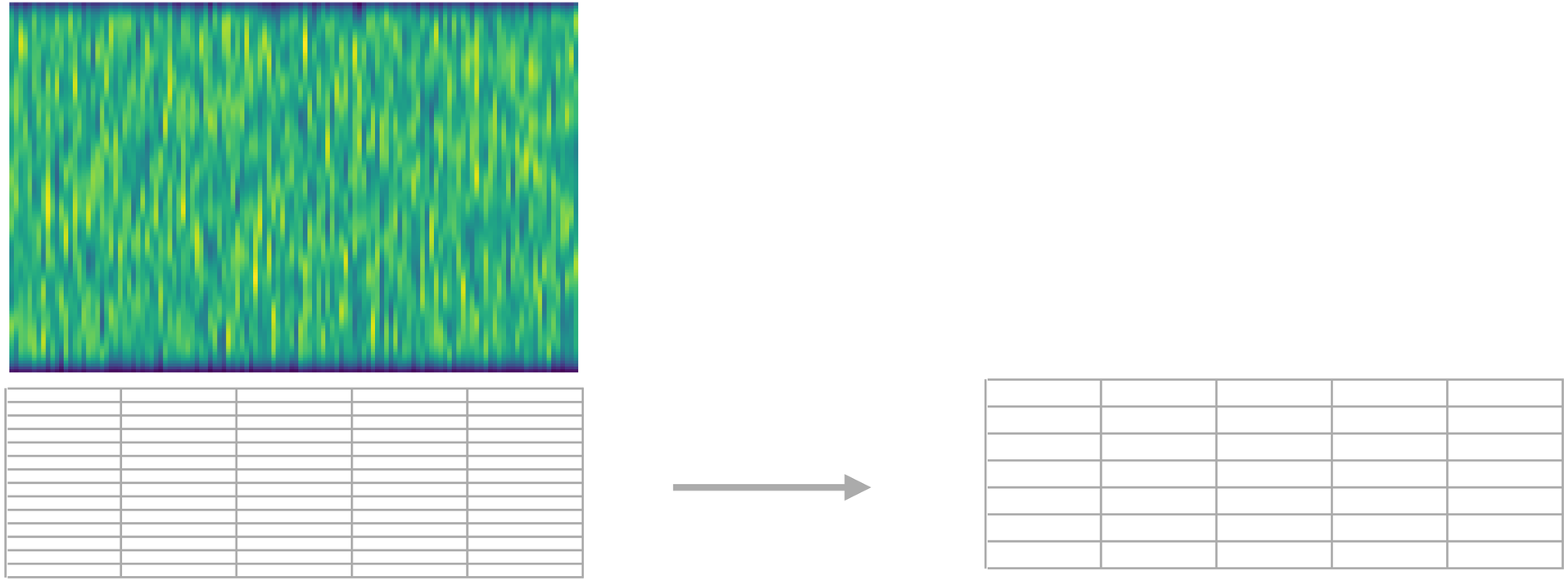






Options for more robust Multigrid

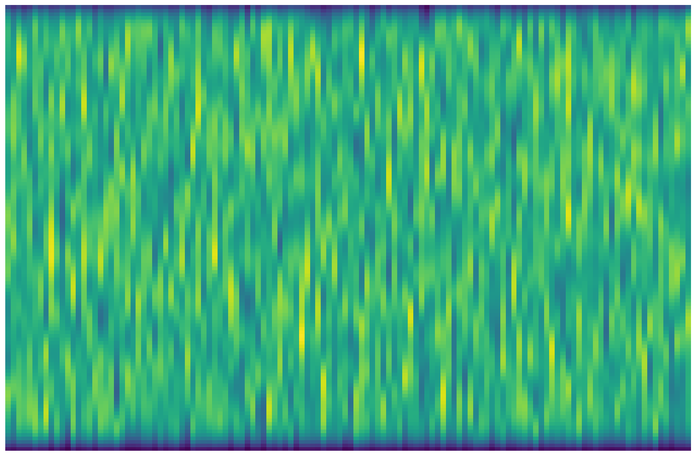
- **Semicoarsening:** Coarsen in the direction of smoothness

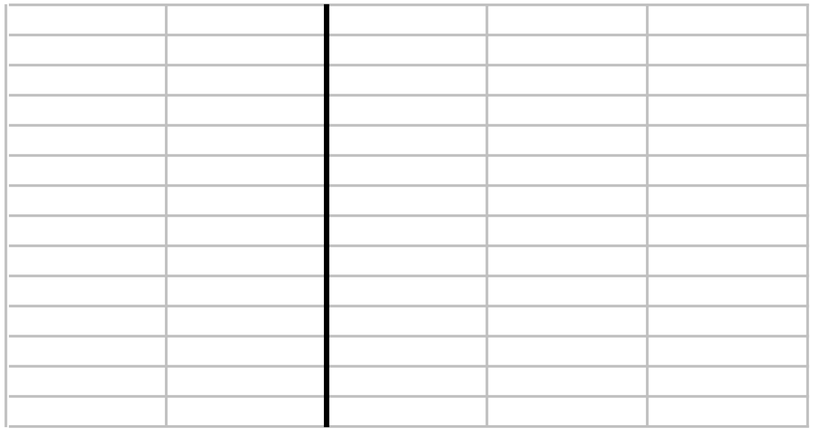


- Downside: if anisotropy varies in another direction, we need a different grid

Options for more robust Multigrid

- **Line/plane relaxation**
  Perform relaxation in groups (in a line)

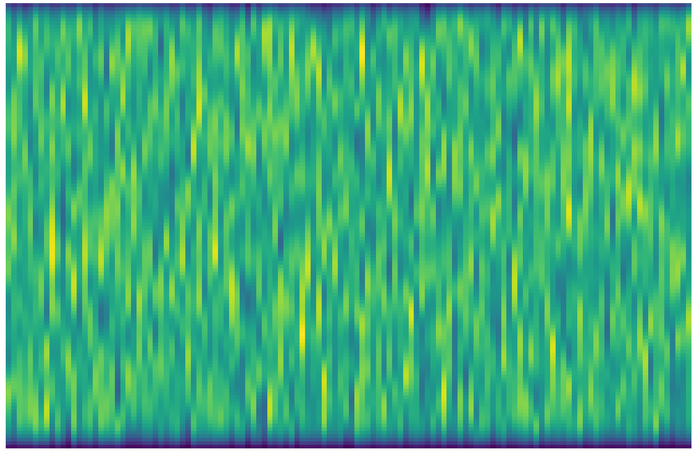


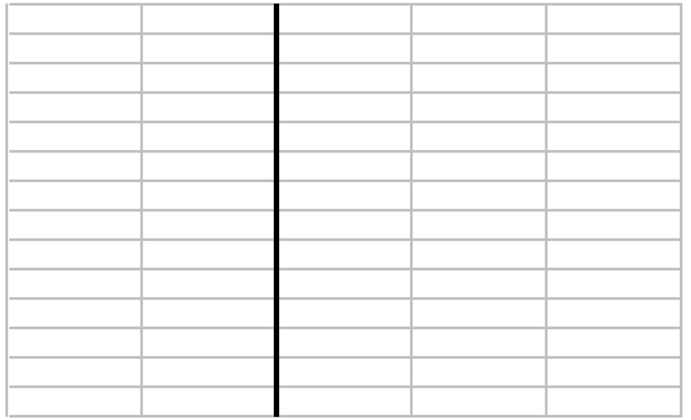


- Example:
  - smooth error in the y-direction (x-lines)
  - no smoothing in the x-direction (y-lines)

Options for more robust Multigrid

- **Line/plane relaxation**
- For each x-line (lines of strong anisotropy):
  - Eliminate the residual on the entire line
  - (Gauss-Siedel, by lines)



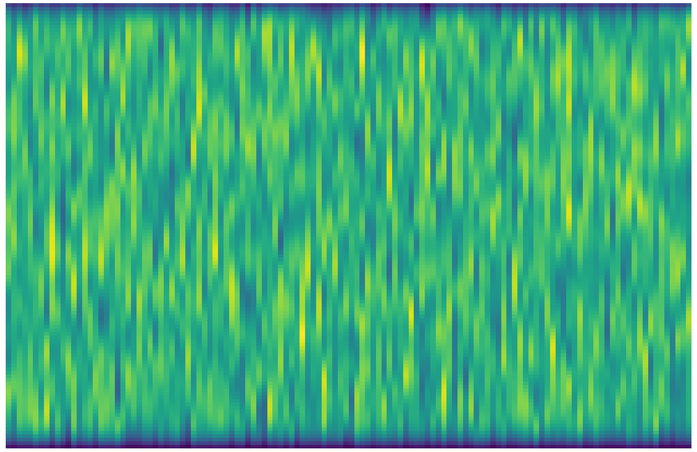


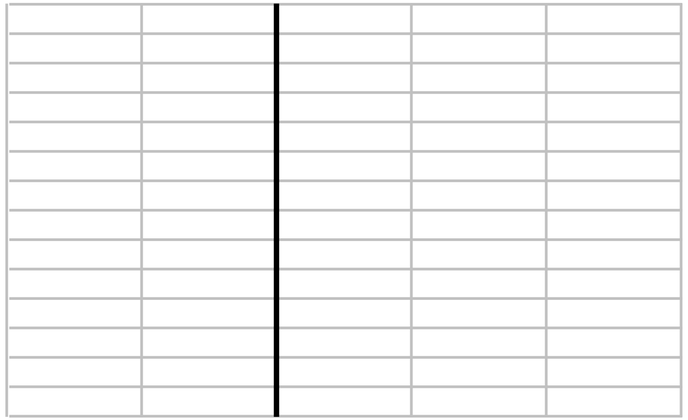
Options for more robust Multigrid

- **Line/plane relaxation**
-

$$A = (n+1)^2 \begin{bmatrix} A_{1D} & -\varepsilon I & & \\ -\varepsilon I & A_{1D} & -\varepsilon I & \\ & -\varepsilon I & A_{1D} & -\varepsilon I \\ & & \ddots & \\ & & -\varepsilon I & A_{1D} \end{bmatrix}$$

$$A_{1D} = (n+1)^2 \begin{bmatrix} 2+2\varepsilon & -1 & & \\ -1 & 2+2\varepsilon & -1 & \\ & & \ddots & \\ & & -1 & 2+2\varepsilon \end{bmatrix}$$





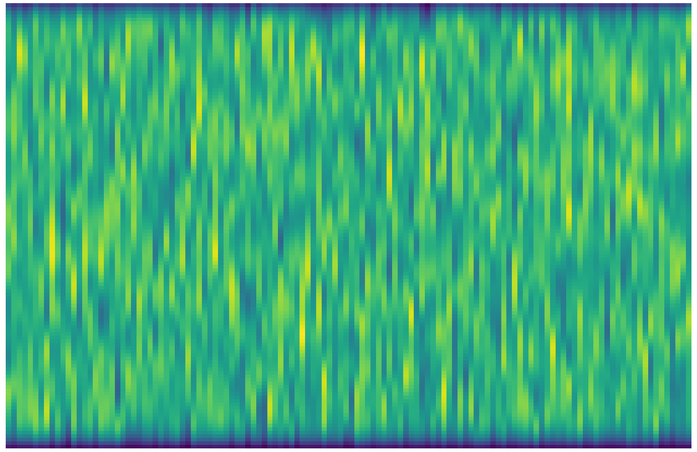
Options for more robust Multigrid

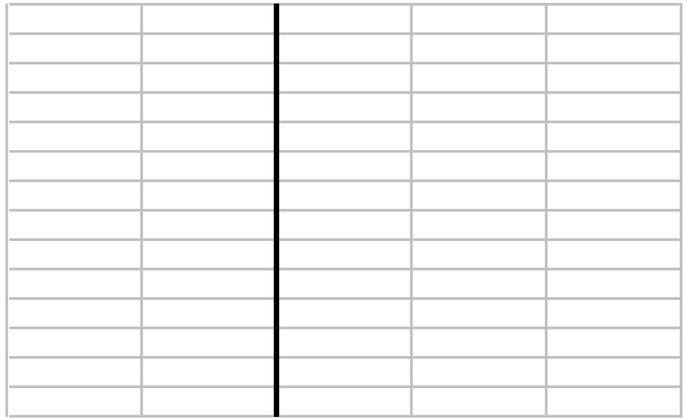
- **Line/plane relaxation**
- For each line, solve

$$A_{1D} v_k = g_k$$

$$v_k = \begin{bmatrix} \vdots \\ v_{k,j-1} \\ v_{k,j} \\ v_{k,j+1} \\ \vdots \end{bmatrix} \quad g_k = \begin{bmatrix} \vdots \\ f_{k,j-1} + \varepsilon \left(v_{k-1,j-1} + v_{k+1,j-1} \right) \\ f_{k,j} + \varepsilon \left(v_{k-1,j} + v_{k+1,j} \right) \\ f_{k,j+1} + \varepsilon \left(v_{k-1,j+1} + v_{k+1,j+1} \right) \\ \vdots \end{bmatrix}$$

- In 3D, lines become planes…



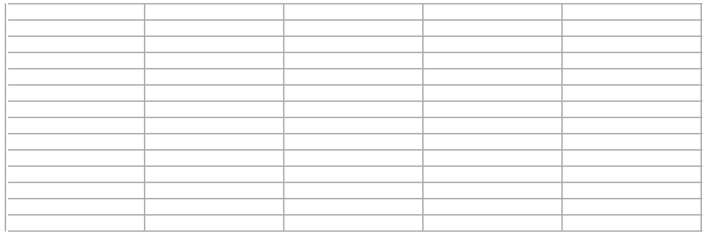


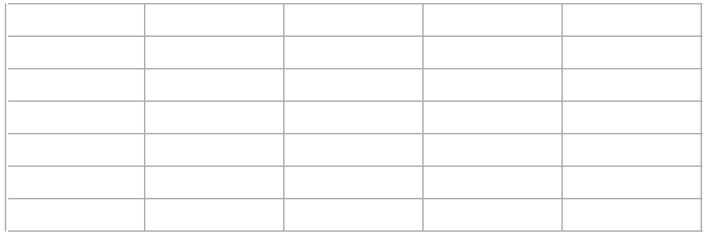
Notebook

- `11-multigrid-2d-line-relaxation.ipynb`

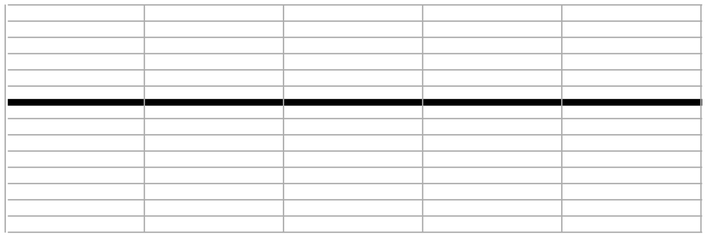
Options for more robust Multigrid

- **Semicoarsening**
  Coarsen in the direction of smoothness





- **Line/plane relaxation**
  Perform relaxation in groups (in a line)



- **Operator based Interpolation** (e.g. BoxMG)

$$Ae = 0$$

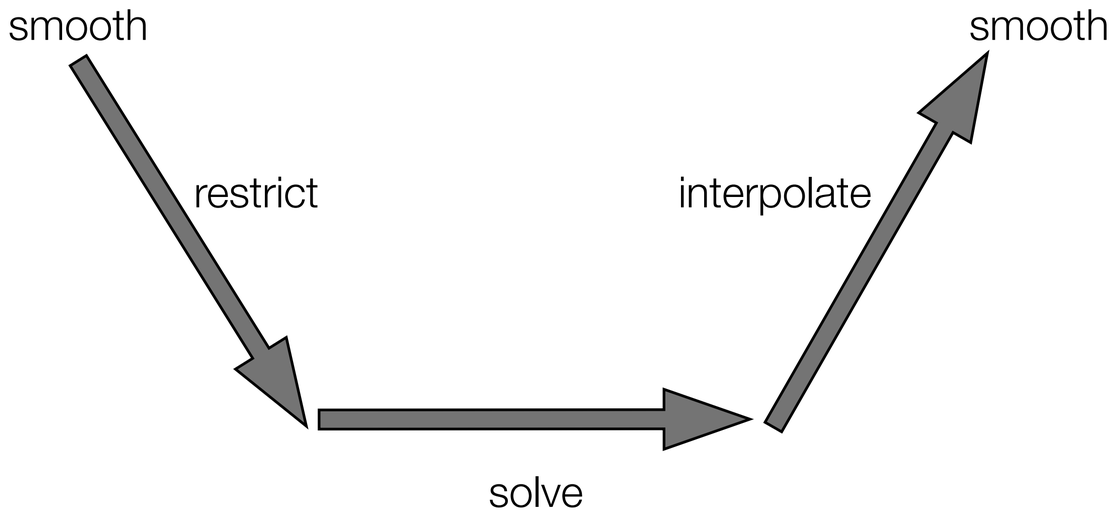
J. E. Dendy, Black box multigrid, J. Comput. Phys., 1982

J. E. Dendy and J. D. Moulton, Black box multigrid with coarsening by a factor of three, J. Numer. Lin. Alg. App., 2010

Algorithm: two-level multigrid

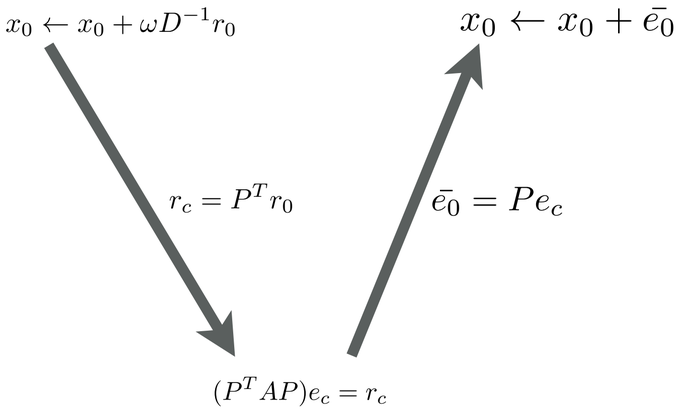
Input: initial guess

1. Smooth ν_{pre} times on $Au = f$
2. Compute $r = f - Au$
3. Compute $r_c = Rr$
4. Solve $A_c e_c = r_c$
5. Interpolate $\hat{e} = Pe_c$
6. Correct $u \leftarrow u + \hat{e}$
7. Smooth ν_{post} times on $Au = f$



A two-level “V” cycle

Algebraic Observation



$$e_1 \leftarrow (I - P(P^T A P)^{-1} P^T A) G e_0$$

coarse grid correction

relax

relax

residual

restrict

coarse solve

interpolate

correct

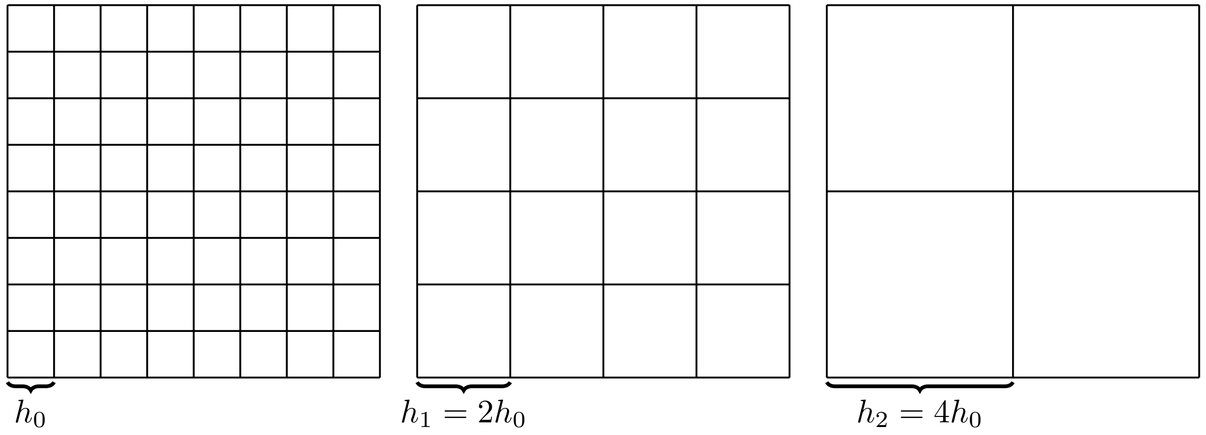
$$Ge_0 \in \mathcal{R}(P) \quad \Rightarrow \quad e_1 = 0$$

interpolation should capture what relaxation misses

Multigrid Basics

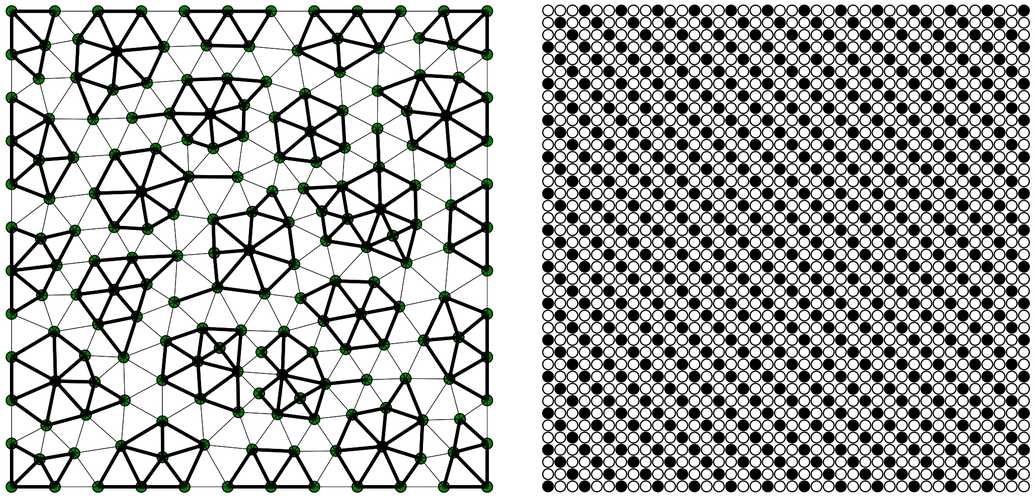
Geometric

- Grids are fixed
- Construct interpolation
- Find the best smoother



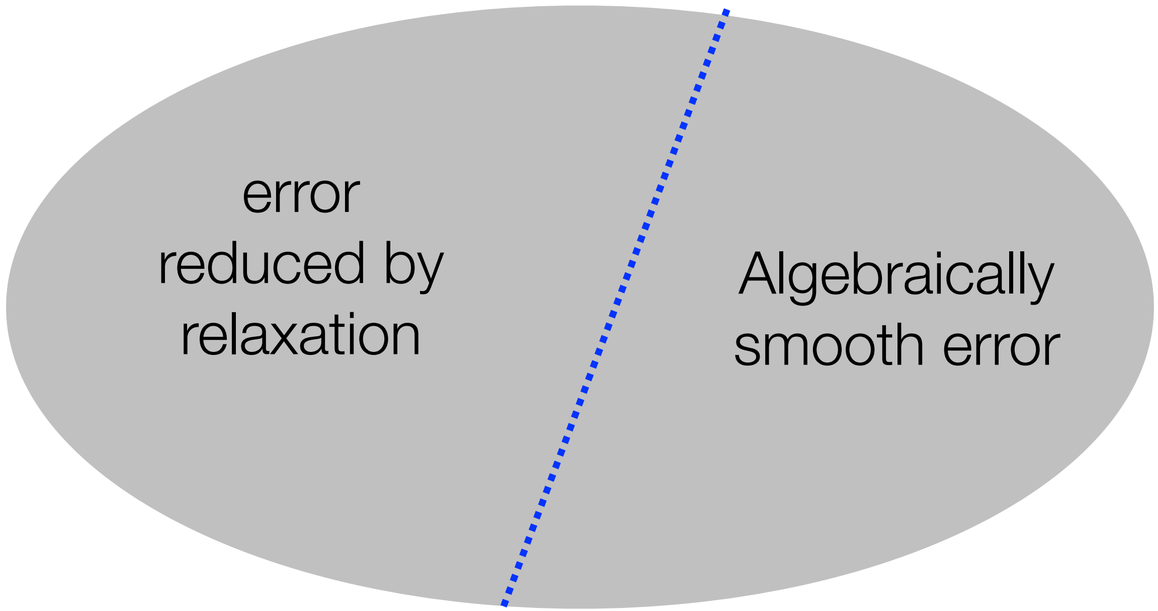
Algebraic

- Relaxation method is fixed
- Find coarse grids
- construct interpolation



Algebraically Smooth Error

- **“Algebraically smooth” error may not be geometrically smooth**



Main idea: Algebraically smooth error

- Take a relaxation scheme such as w-Jacobi

$$e \leftarrow (I - M^{-1}A)e$$

- If relaxation stagnates, then the remaining error exhibits poor convergence, so

- Formally (characterized by small eigenvalues)

$$(I - M^{-1}A)e \approx e \Rightarrow M^{-1}Ae \approx 0 \Rightarrow r \approx 0$$

$$\langle Ae, e \rangle << 1$$

Main idea: Algebraically smooth error

- We then have

$$
\begin{aligned}
\langle Ae, e\rangle &= \sum_i e_i (A_{ii} e_i + \sum_{j \neq i} A_{ij} e_j) && \text{assume zero row sum} \\
&= \sum_i e_i \left(\sum_{j \neq i} -A_{ij}(e_i - e_j) \right) \\
&= \sum_{i<j} -A_{ij} \cdot e_i \cdot (e_i - e_j) + \sum_{i>j} -A_{ij} \cdot e_i \cdot (e_i - e_j) && \text{swap i, j} \\
&= \sum_{i<j} -A_{ij} \cdot e_i \cdot (e_i - e_j) - \sum_{i<j} -A_{ij} \cdot e_i \cdot (e_i - e_j) \\
&= \sum_{i<j} -A_{ij} \cdot (e_i - e_j)^2
\end{aligned}
$$

- Ok, so smooth error varies **slowly** in the direction of large matrix coefficients

Briggs, William L. and Henson, Van Emden and McCormick, Steve F., A Multigrid Tutorial (2Nd Ed., 20000

Main idea: Algebraically smooth error

- We have assumed **geometric** smoothness to show

$$\mathbf{e}^T A \mathbf{e} = \sum_{i<j} (-a_{ij})(e_i - e_j)^2 \ll 1$$

- **CF AMG**: Smooth error varies slowly in the direction of "large" matrix coefficients
- **Strength of connection**: Given a threshold $0 < \theta \leq 1$, we say that variable u_i strongly depends on variable u_j if

$$-a_{ij} \geq \theta \max_{k \neq i} \{-a_{ik}\}$$

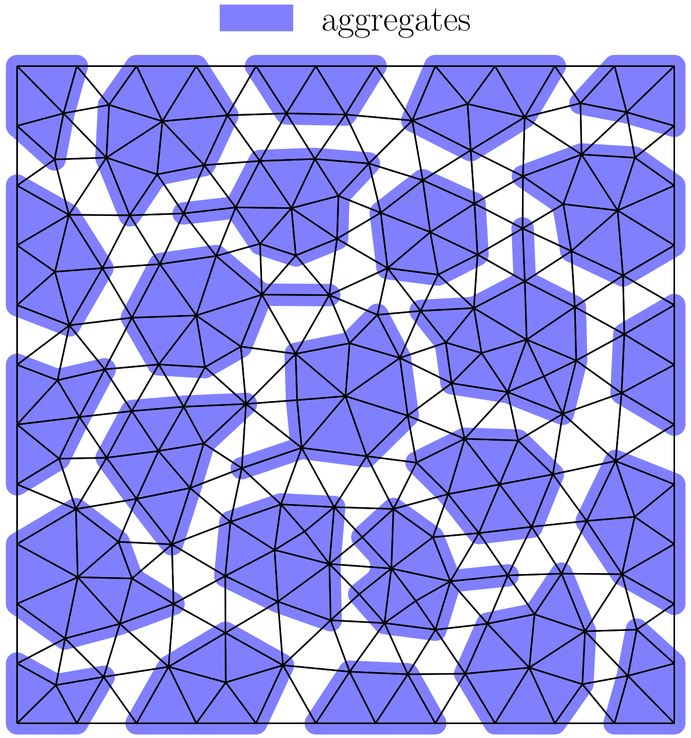
- Often positive off-diagonals are treated as weak
- This definition of strength of connection is not symmetric

Demo: `12-AMG-coarse-mesh.ipynb`

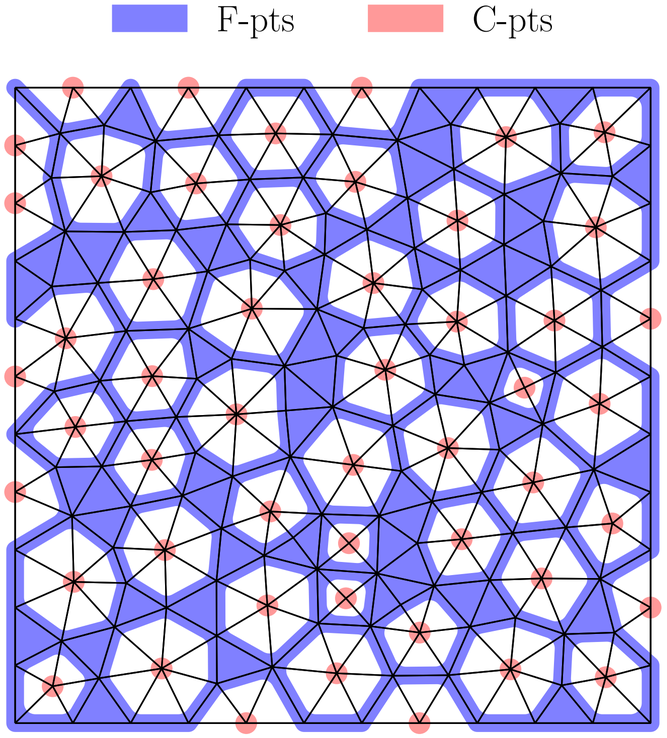
Luke Olson
http://lukeo.cs.illinois.edu/

Slide credit: R. Falgout, 2/1/2016

Two (general) forms of AMG



- Smoothed Aggregation AMG (SA-AMG)
- Interpolation constructed from candidate vectors
- Clear approach to *optimize* interpolation



- Coarse-Fine AMG (CF-AMG) or Ruge-Stüben
- Coarse grid points are a subset of the fine grid points
- Edge-wise construction of interpolation, allowing straightforward control of sparsity
- Incorporating near-nullspace is not straightforward

CF AMG

- **Goal:** select grid points to form the coarse grid where smooth error is well represented
- **Idea:** the variable at *j* would be a good **C-point** if it strongly influences the variable at *I*
- Strongly depend on…

$$S_i = \{j \ : \ -A_{ij} \geq \theta \max_{k \neq i} -A_{ik}\}$$

- Strongly influence…

$$S_i^T = \{j \ : \ i \in S_i\}$$

J. W. Ruge K. Stüben, Algebraic Multigrid, 1987

CF AMG

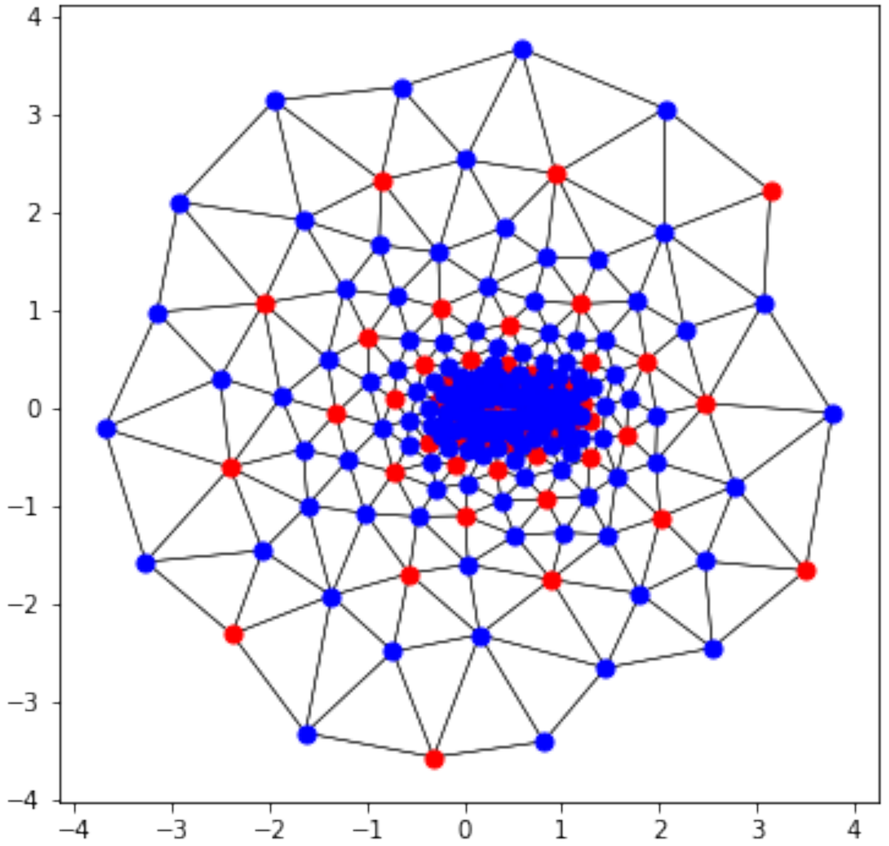
- **C-points:** coarse grid points
- **F-points:** fine grid points
- Either a C-pt or an F-pt

- Coarse interpolatory set

$$\Omega = C \cup F \quad C \cap F = \emptyset$$

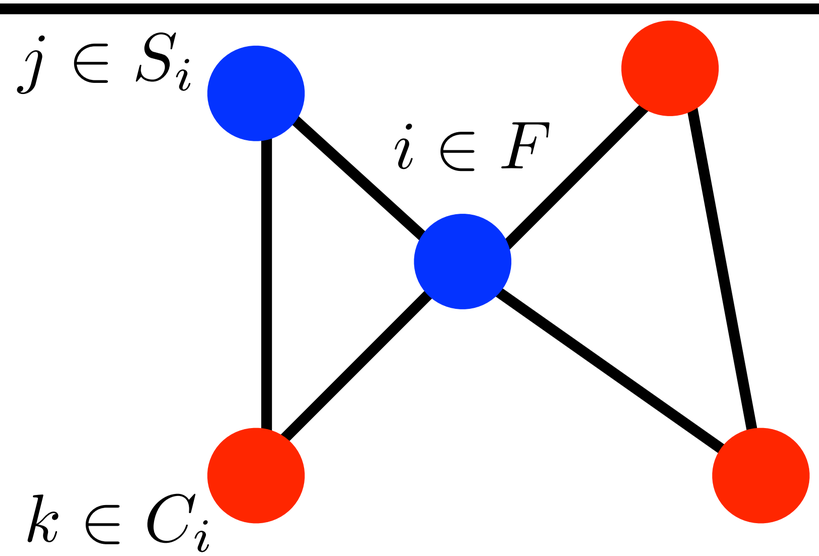
C-pts that are used to interpolated F-pt i.

$$C_i$$



CF AMG

- (C1) Each $i \in F$ should strongly depend on either
  - A point in C
  - A point that strongly depends on a point in C_i



CF AMG

- (C2) The C-points should be *maximal* with no C-point depending on another C-point

- (C1) increases the size of the coarse grid (C-points)

- (C2) puts constraints on the *size* of the coarse grid

- Must satisfy (C1) in order to construct interpolation. Use (C2) to help limit computational complexity

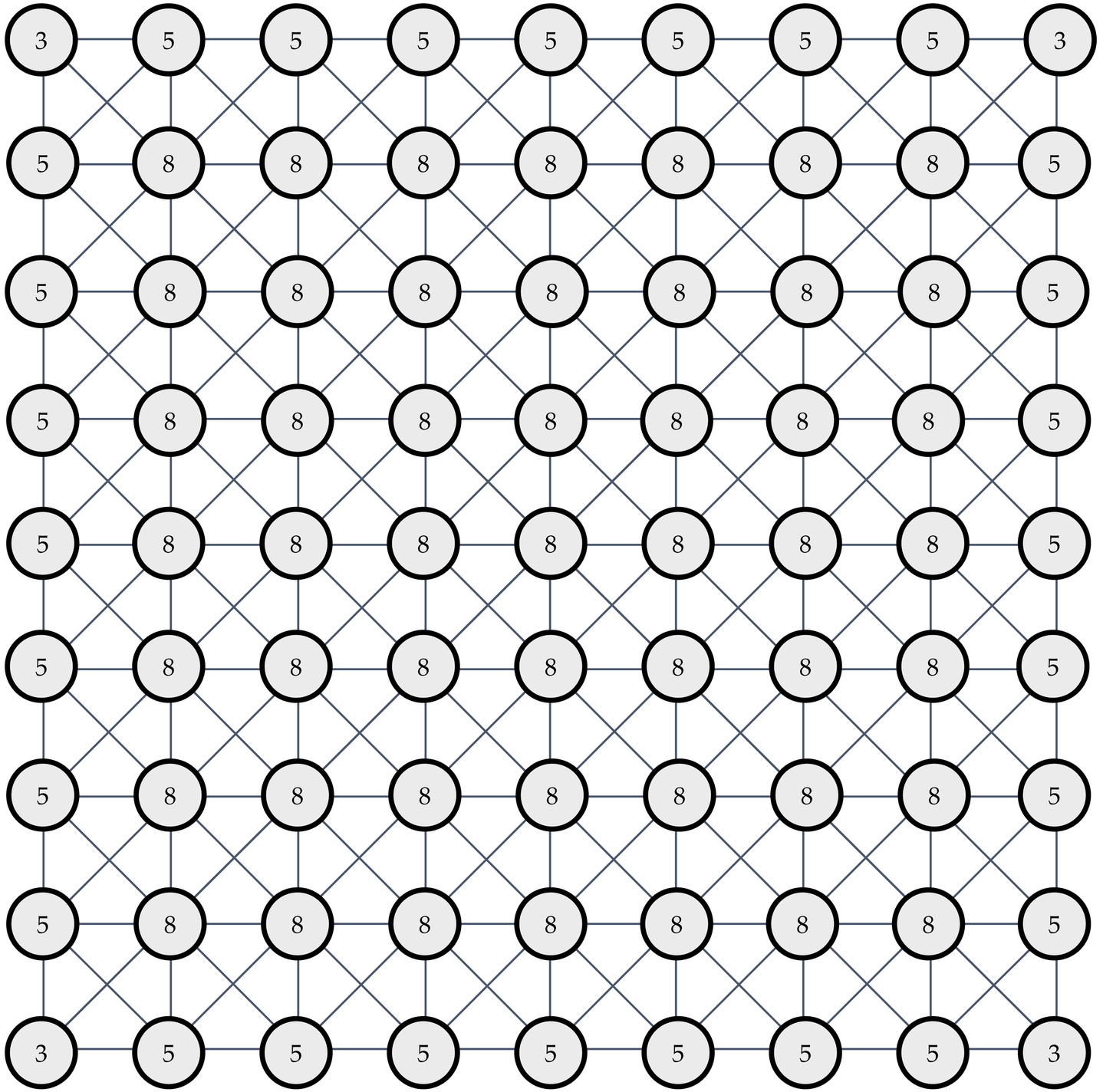
Ruge-Stüben

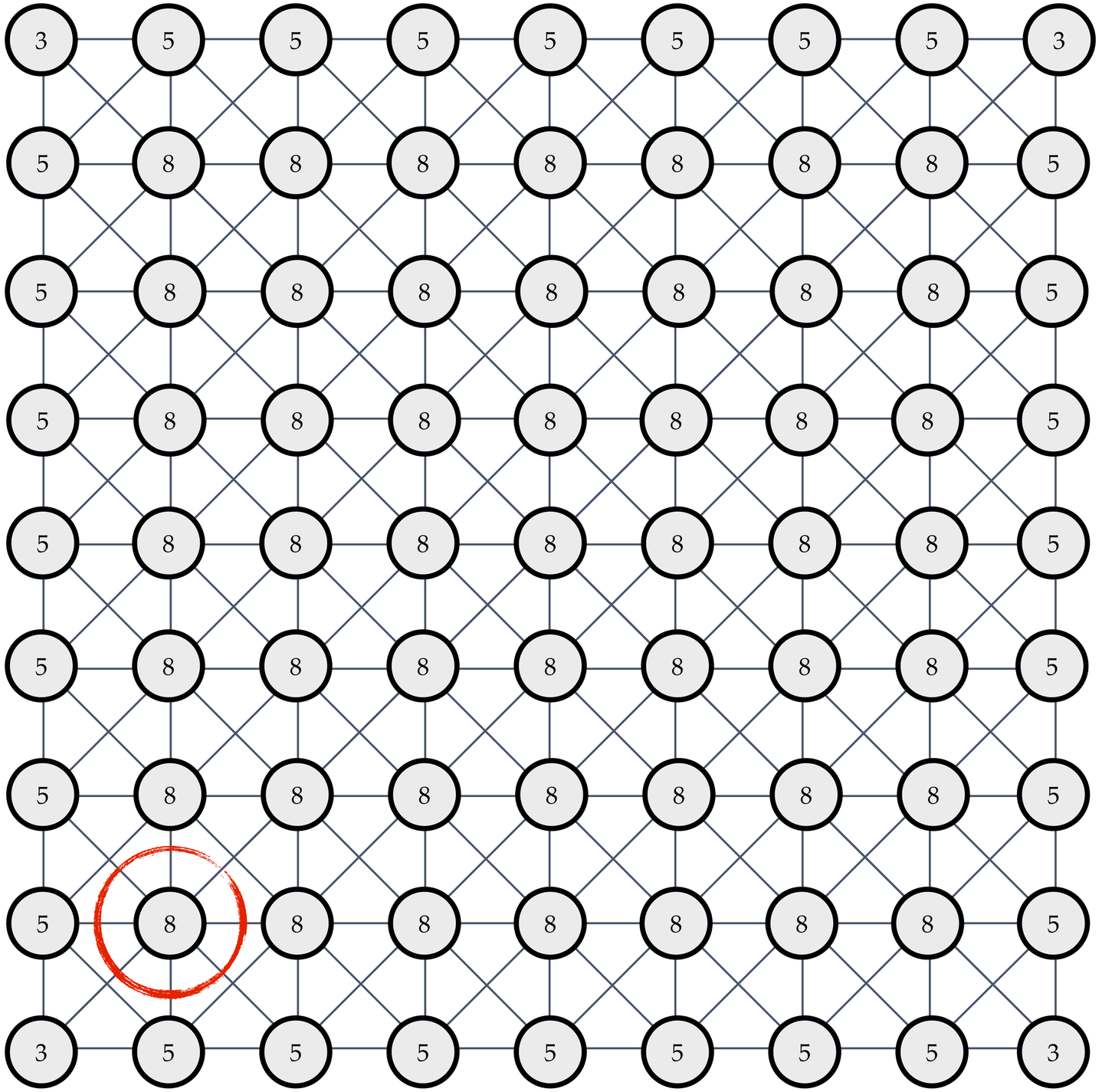
Algorithm 3. RUGE–STÜBEN.

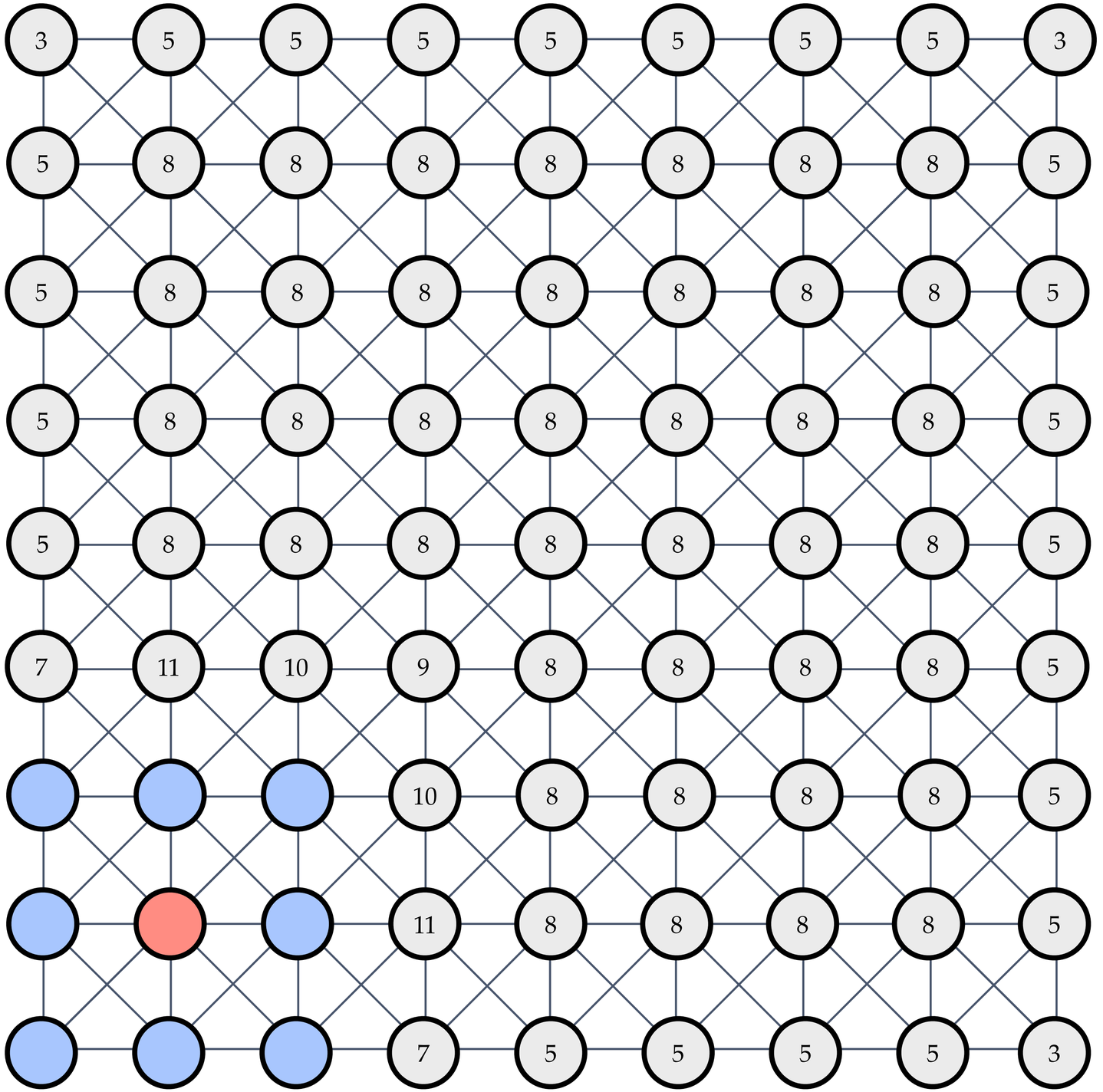
Initialize:

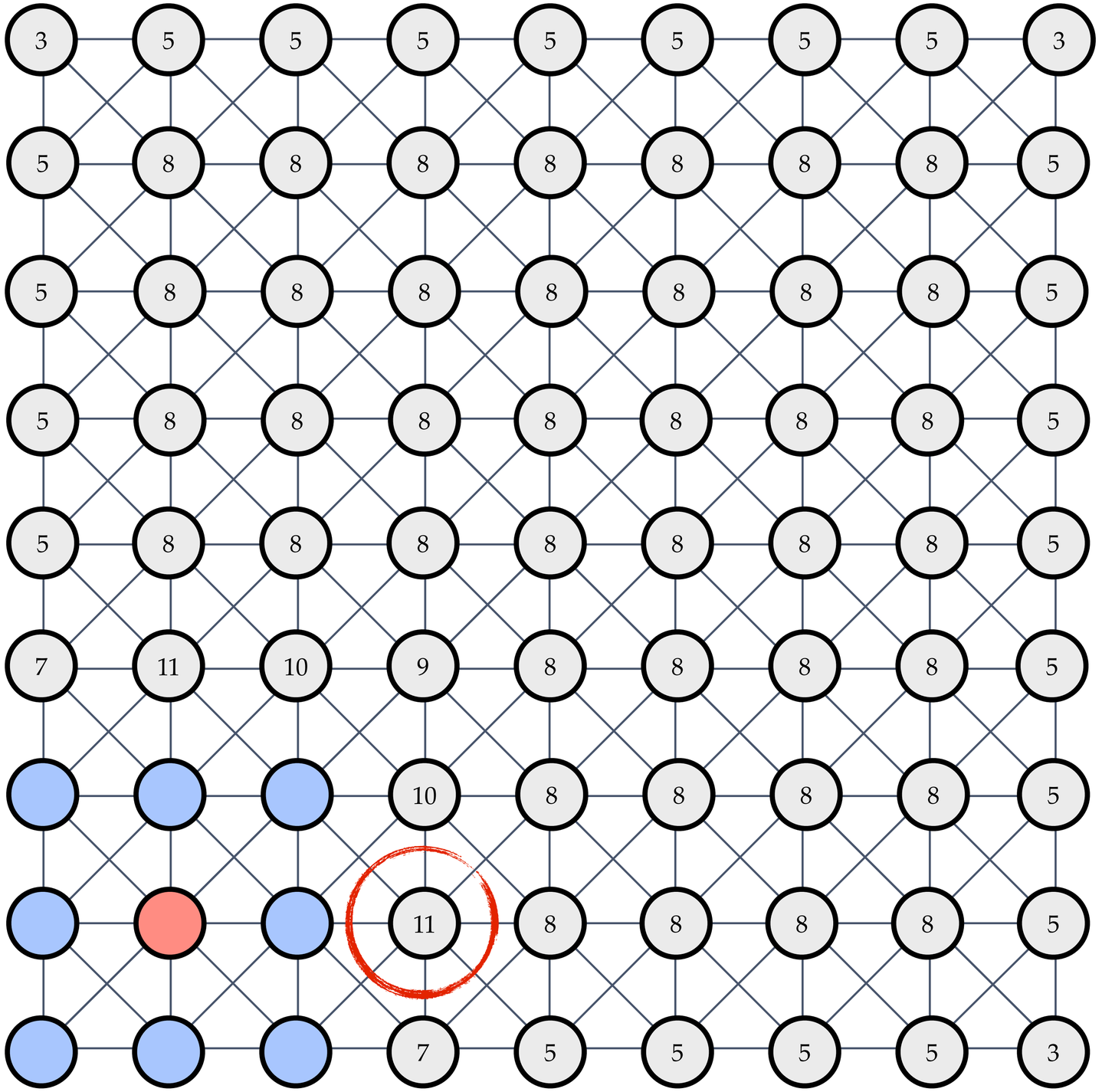
$U = \Omega,\ C = \emptyset,\ F = \emptyset$

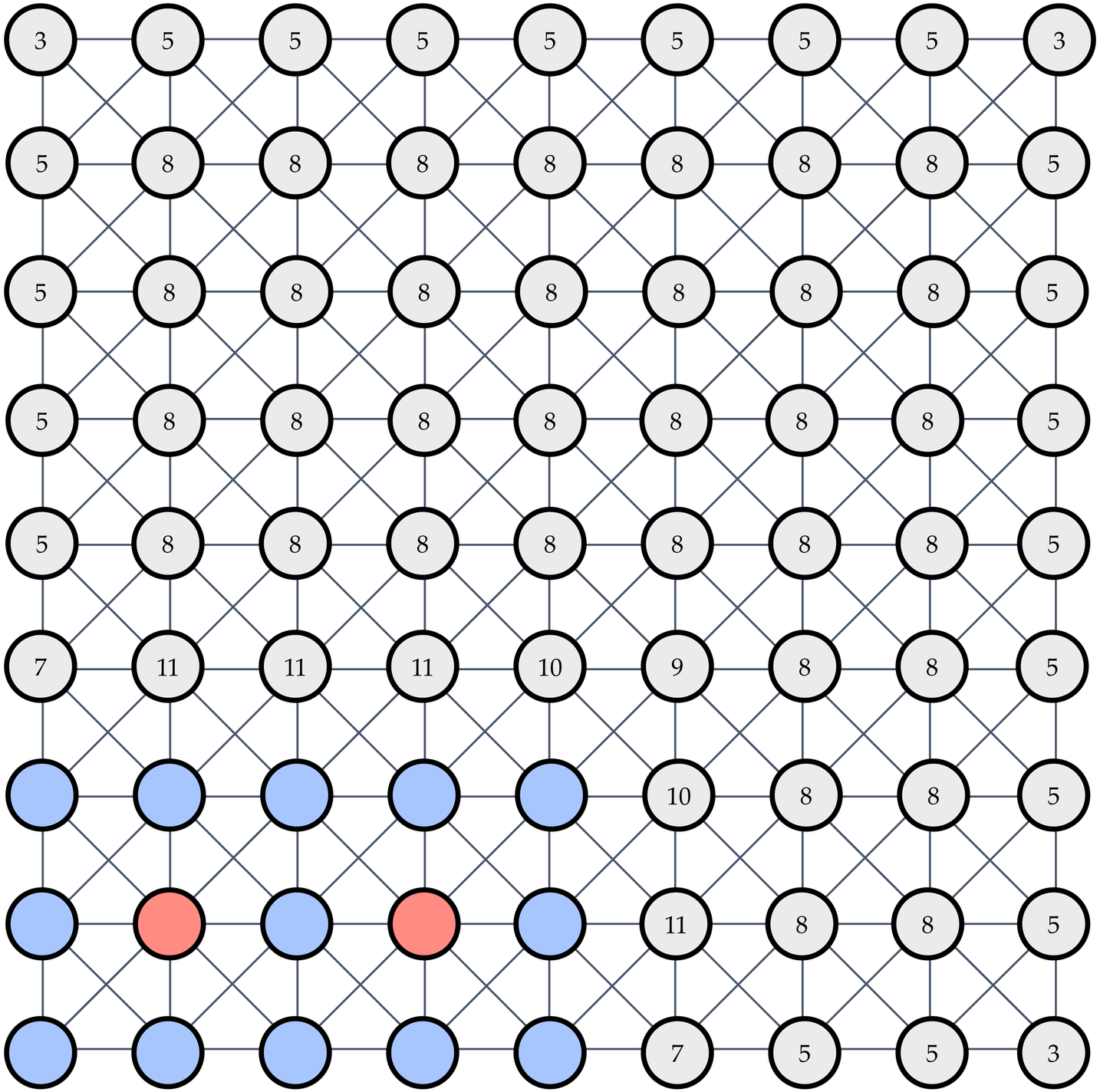
1. **for all** $i \in \Omega$ **do**
2. $\quad w_i \leftarrow |S_i^{\mathrm{T}}|$
3. **end for**
4. **while** $|U| > 0$ **do** {First pass}
5. $\quad$ select i: $w_i \geqslant w_j,\ \forall j \in U$
6. $\quad U \leftarrow U \backslash \{i\}$
7. $\quad C \leftarrow C \cup \{i\}$
8. $\quad$ **for all** $j \in S_i^{\mathrm{T}} \cap U$ **do**
9. $\quad\quad U \leftarrow U \backslash \{j\}$
10. $\quad\quad F = F \cup \{j\}$
11. $\quad\quad$ **for all** $k \in S_j \cap U$ **do**
12. $\quad\quad\quad w_k \leftarrow w_k + 1$
13. $\quad\quad$ **end for**
14. $\quad$ **end for**
15. **end while**
16. **for all** $i \in F$ **do** {Second pass}
17. $\quad$ **for all** $j \in S_i \cap S_i^{\mathrm{T}} \cap F$ **do**
18. $\quad\quad$ **if** $S_i \cap S_j \cap C = = \emptyset$ **then**
19. $\quad\quad\quad$ make i or j into C-point
20. $\quad\quad$ **end if**
21. $\quad$ **end for**
22. **end for**

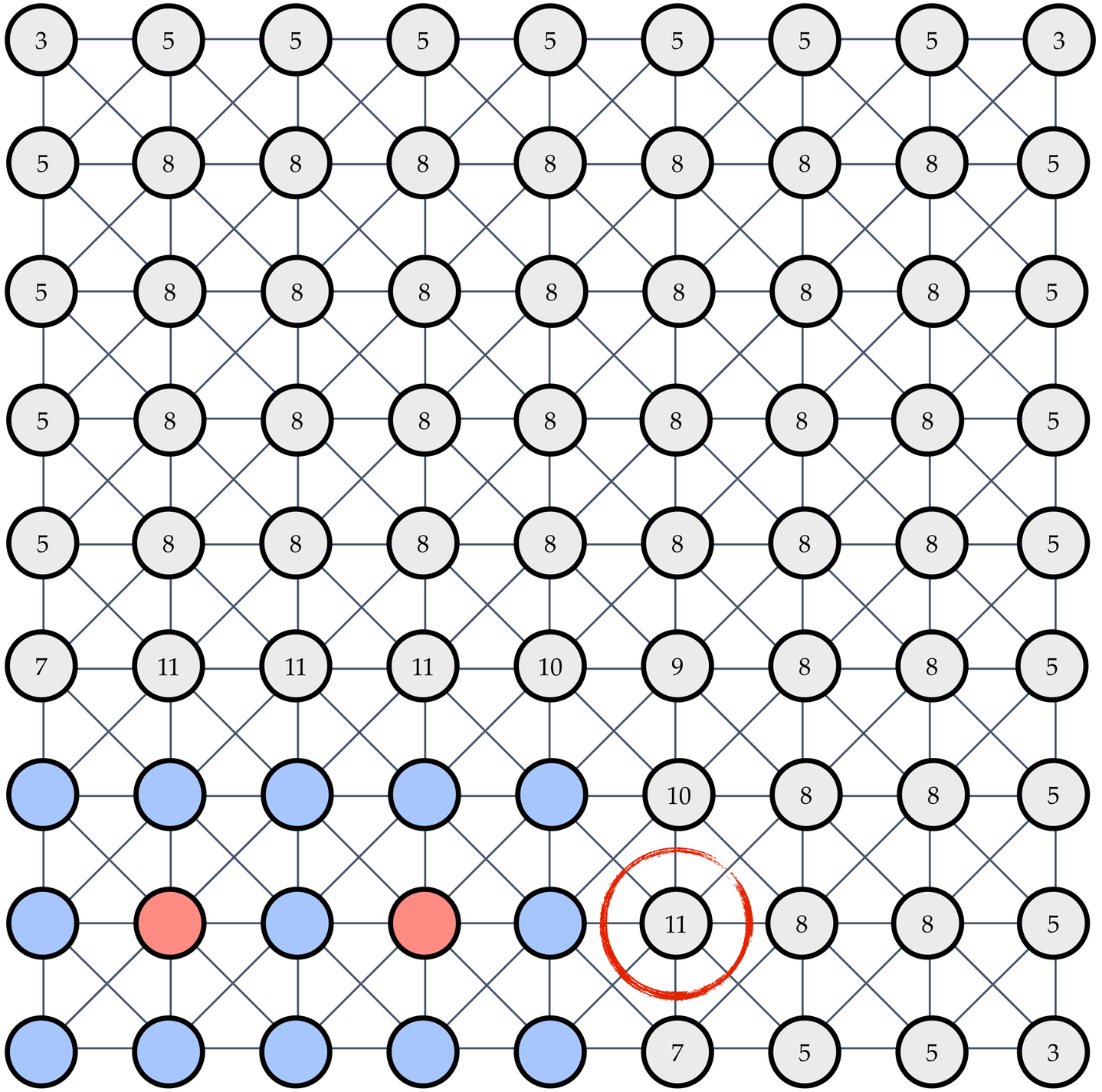

3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
3 5 5 5 5 5 5 5 3

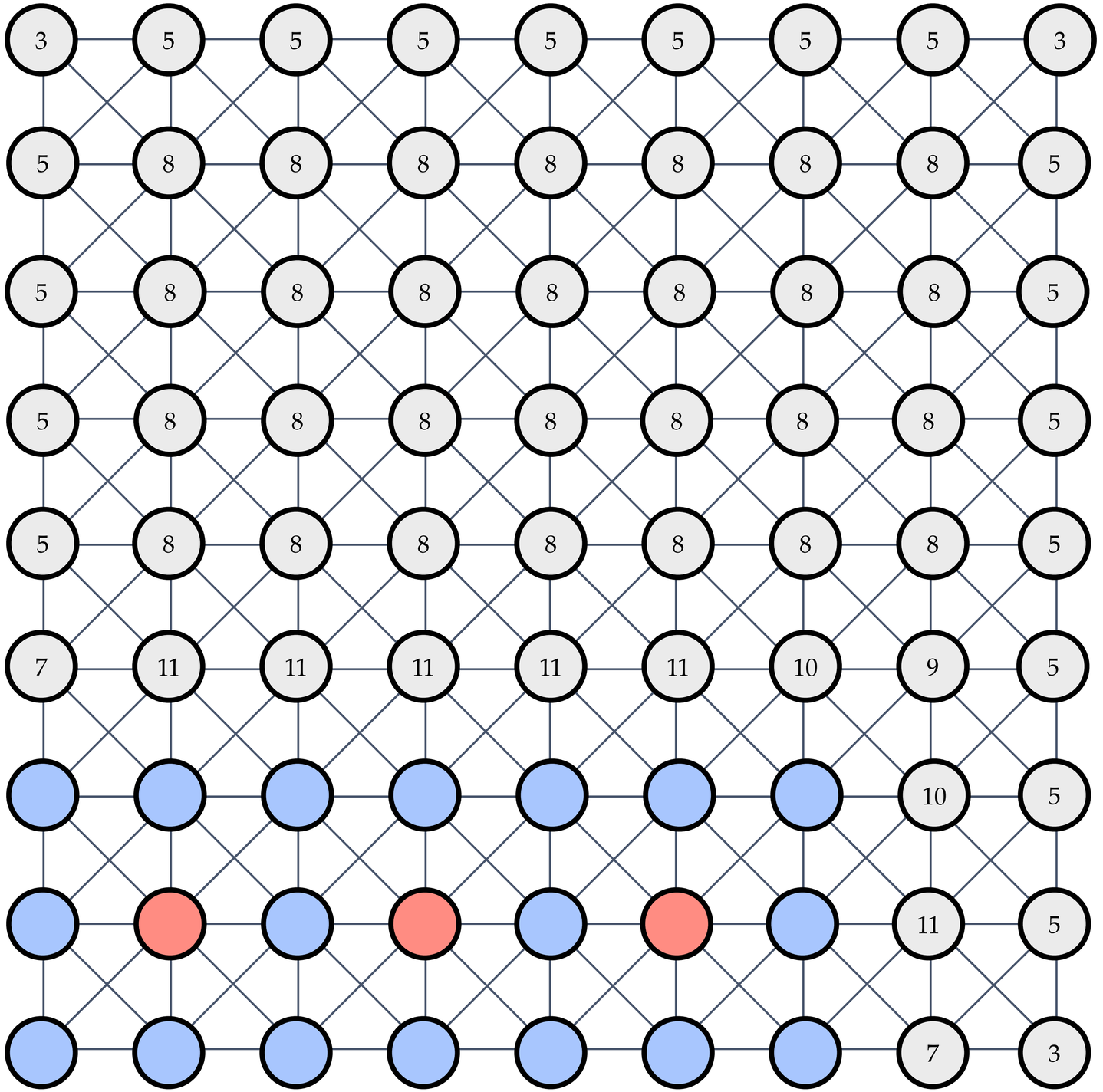

3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
3 5 5 5 5 5 5 5 3

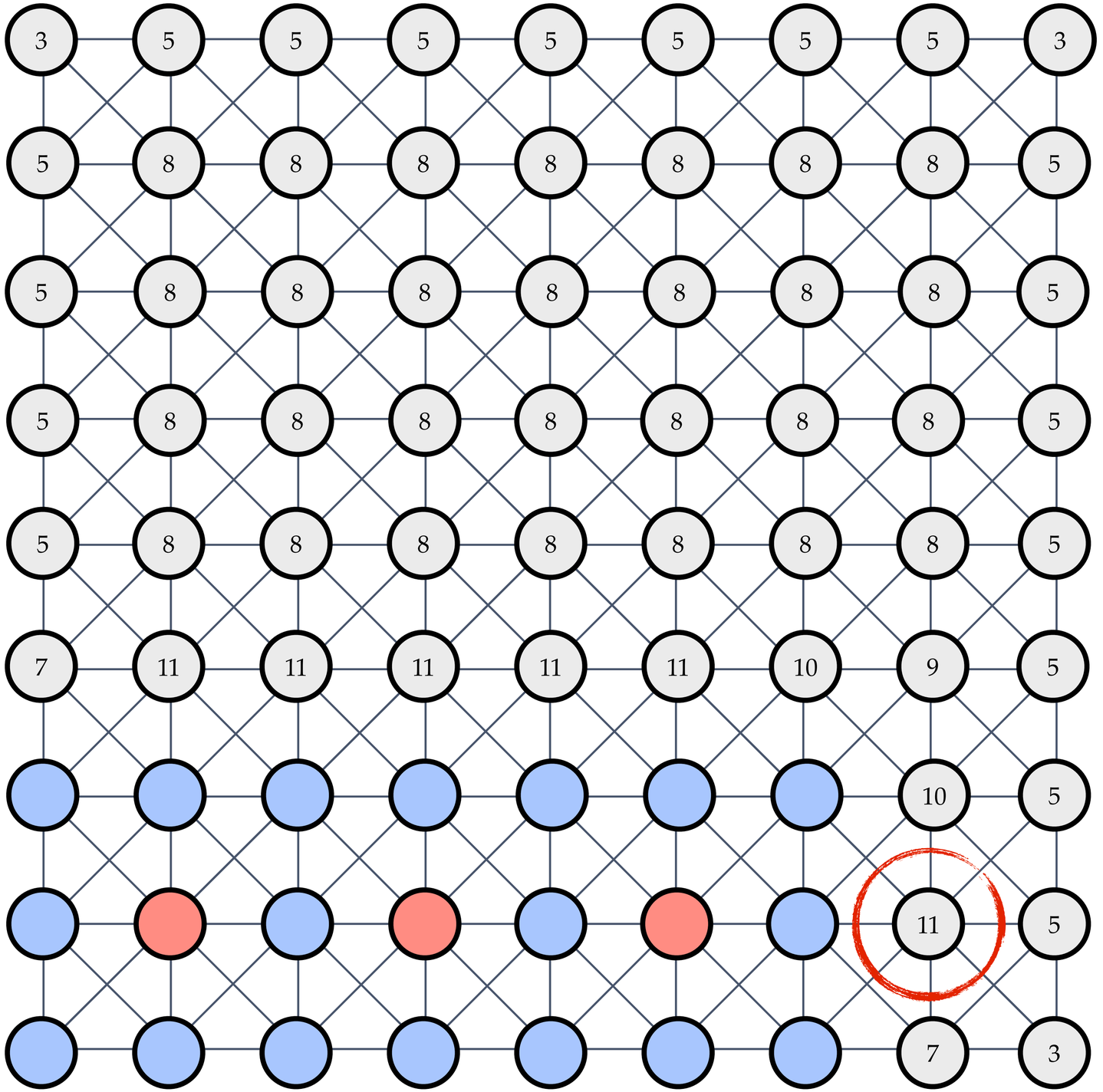

3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 10 9 8 8 8 8 5
10 8 8 8 8 5
11 8 8 8 8 5
7 5 5 5 5 3

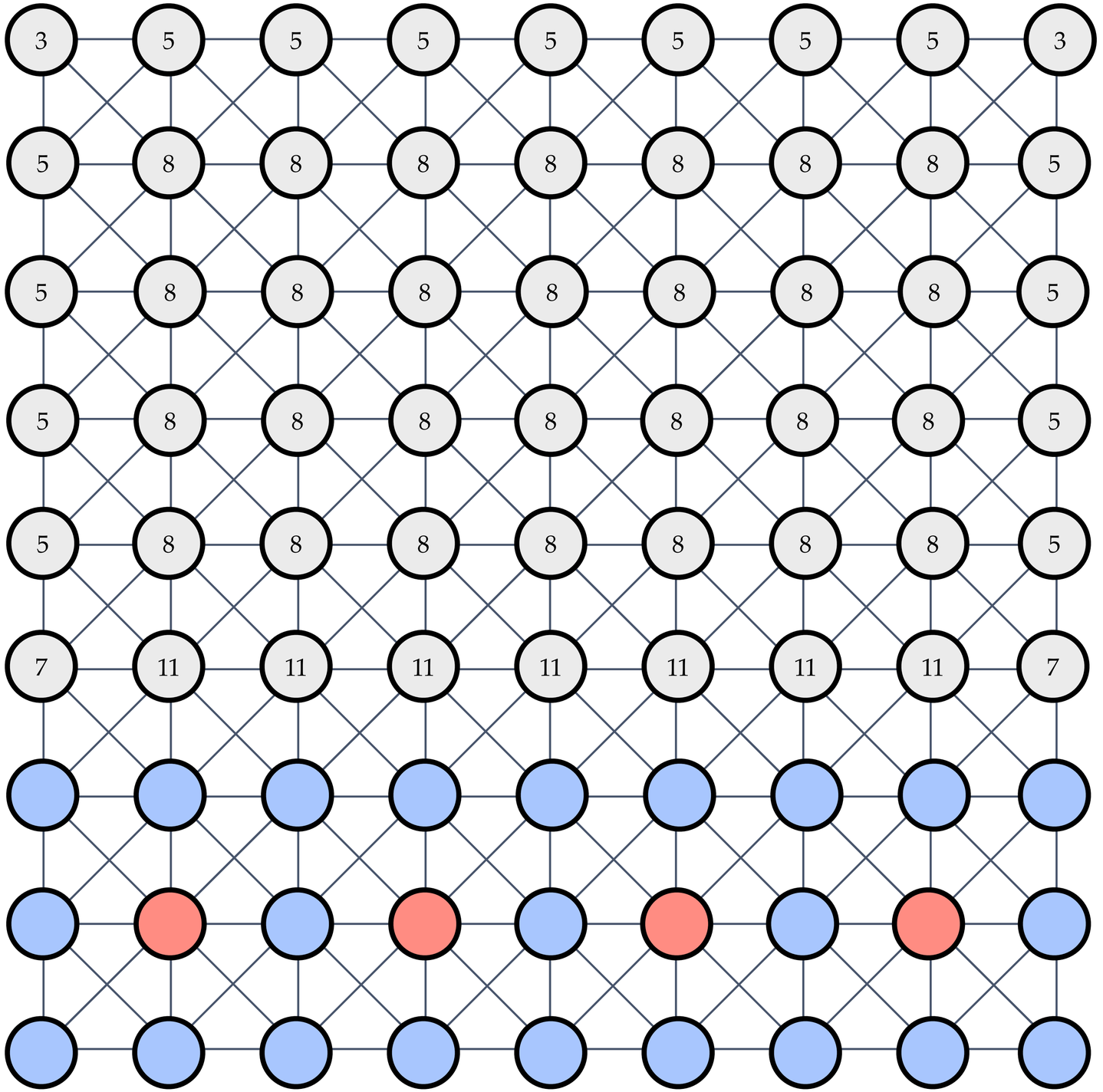

3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 10 9 8 8 8 8 5
10 8 8 8 8 5
11 8 8 8 8 5
7 5 5 5 5 3

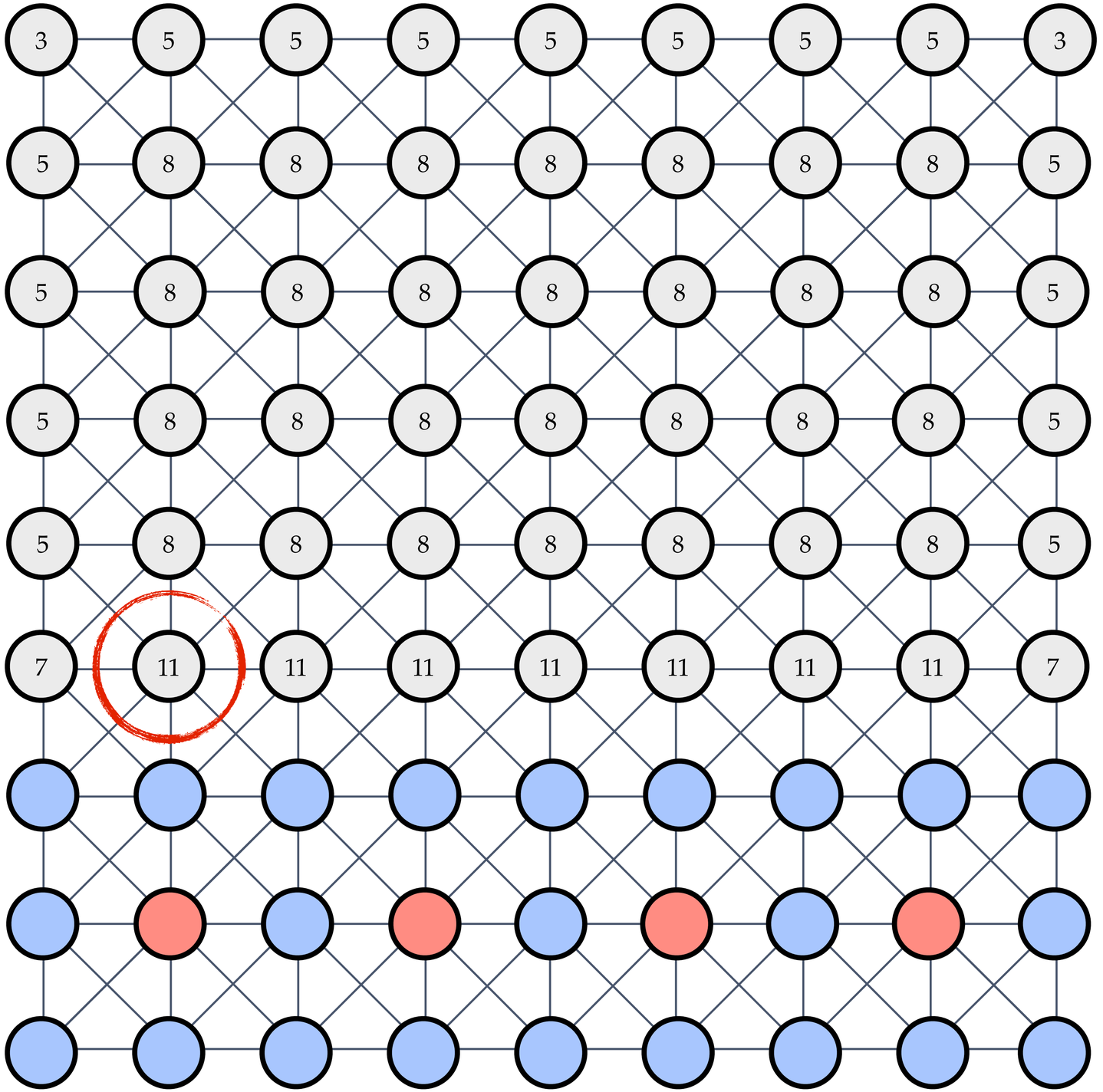

3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 11 11 10 9 8 8 5
10 8 8 5
11 8 8 5
7 5 5 3

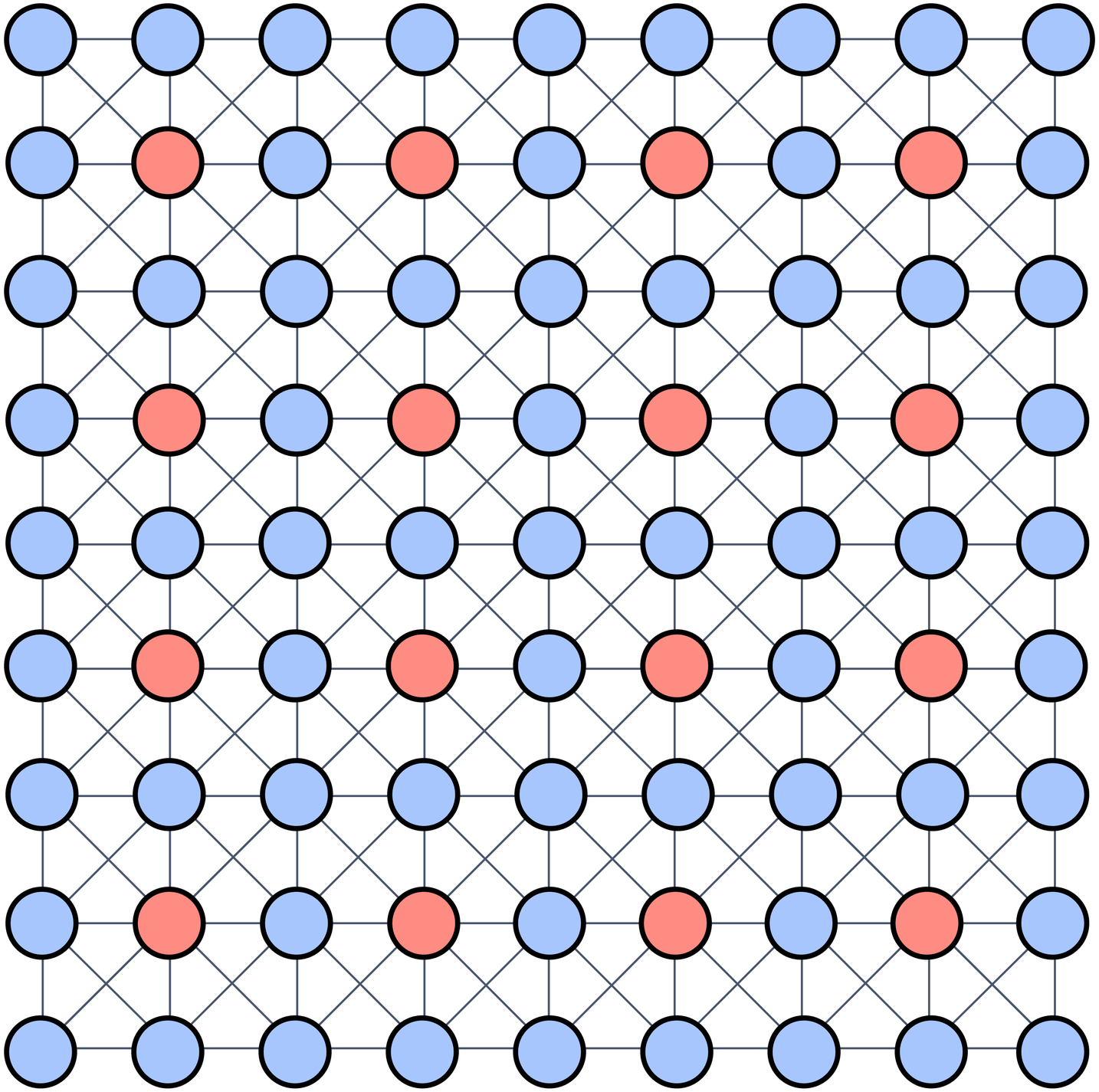

3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 11 11 10 9 8 8 5
10 8 8 5
11 8 8 5
7 5 5 3


3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 11 11 11 11 10 9 5
10 5
11 5
7 3


3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 11 11 11 11 10 9 5
10 5
11 5
7 3


3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 11 11 11 11 11 11 7


3 5 5 5 5 5 5 5 3
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
5 8 8 8 8 8 8 8 5
7 11 11 11 11 11 11 11 7



CF AMG

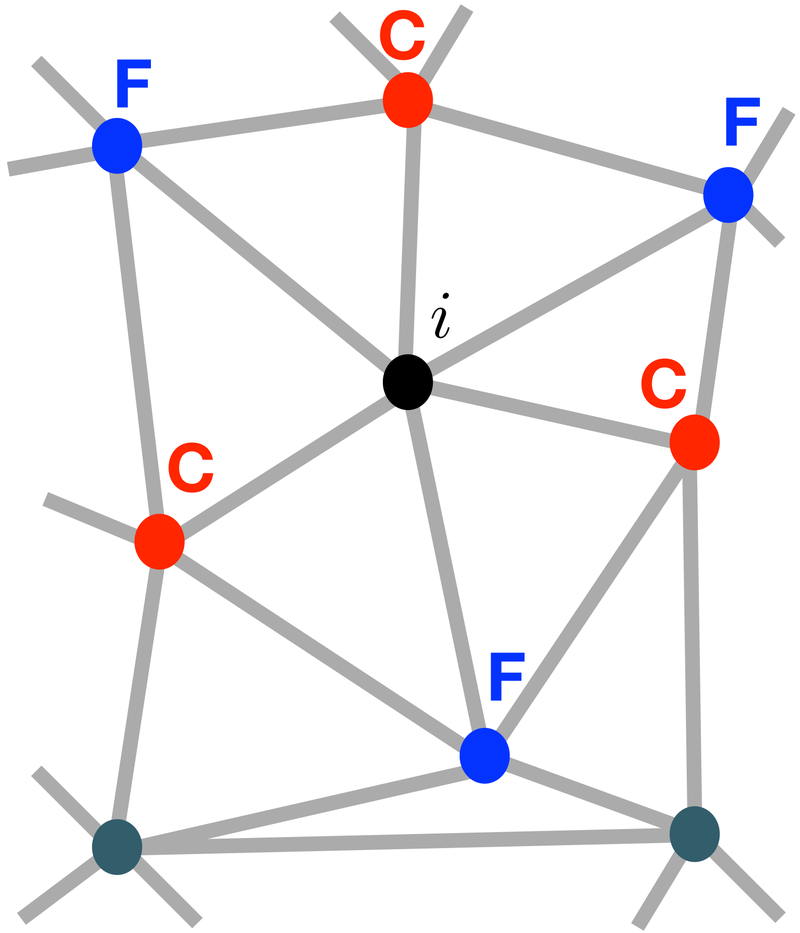
- With a coarse grid (C-points) defined, we can turn to interpolation.

- Write interpolation as a weighted sum of points in the coarse interpolatory set

$$(P\vec{e})_i = \begin{cases} e_i & i \in C \\ \sum_{j \in C_i} \omega_{ij} e_j & i \in F \end{cases}$$

- Or

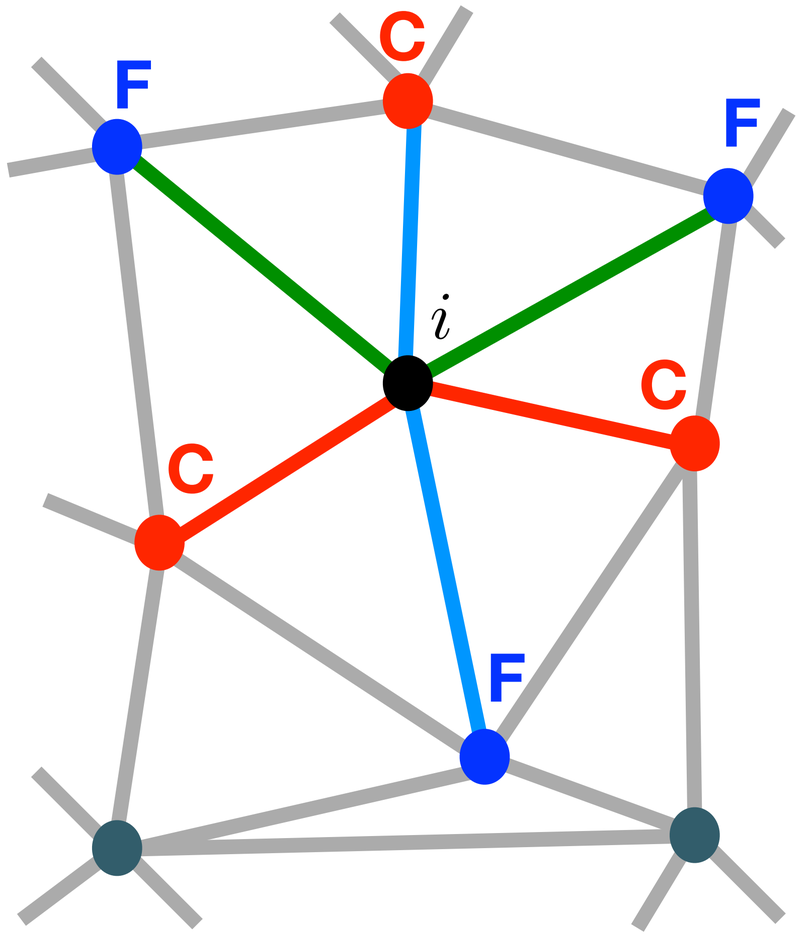
$$P\vec{e} = P \begin{bmatrix} \vec{e}_C \\ \vec{e}_F \end{bmatrix} = \begin{bmatrix} I \\ W \end{bmatrix} \begin{bmatrix} \vec{e}_C \\ \vec{e}_F \end{bmatrix}$$



Example from MG Tutorial

CF AMG

- To interpolate, distinguish **strong** and **weak** connections to interpolate from
- C_i are **strong C-points**
- D_i^s are **strong F-points**
- D_i^w are **weak C/F-points**



CF AMG

- Start with smooth error

$$Ae \approx 0$$

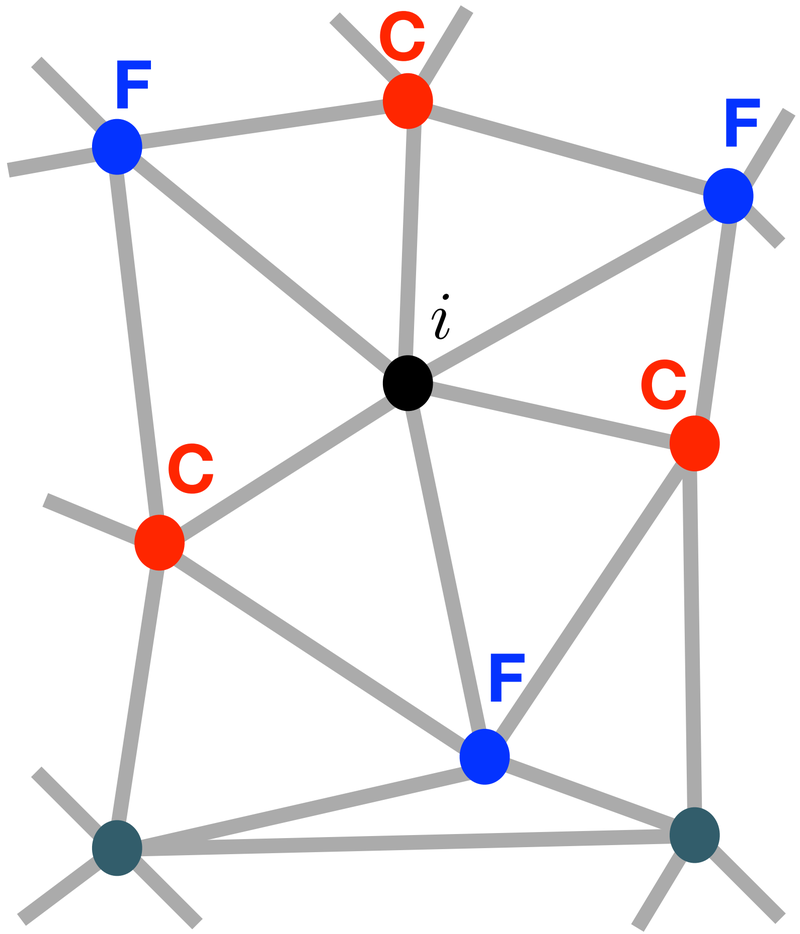
- Then “solve” for i

$$A_{ii}e_i = \sum_{j \neq i} A_{ij}e_j$$

- Then split into types

$$A_{ii}e_i = \sum_{j \in C_i} A_{ij}e_j + \sum_{j \in D_i^s} A_{ij}e_j + \sum_{j \in D_i^w} A_{ij}e_j$$

strong C **strong F** **weak**

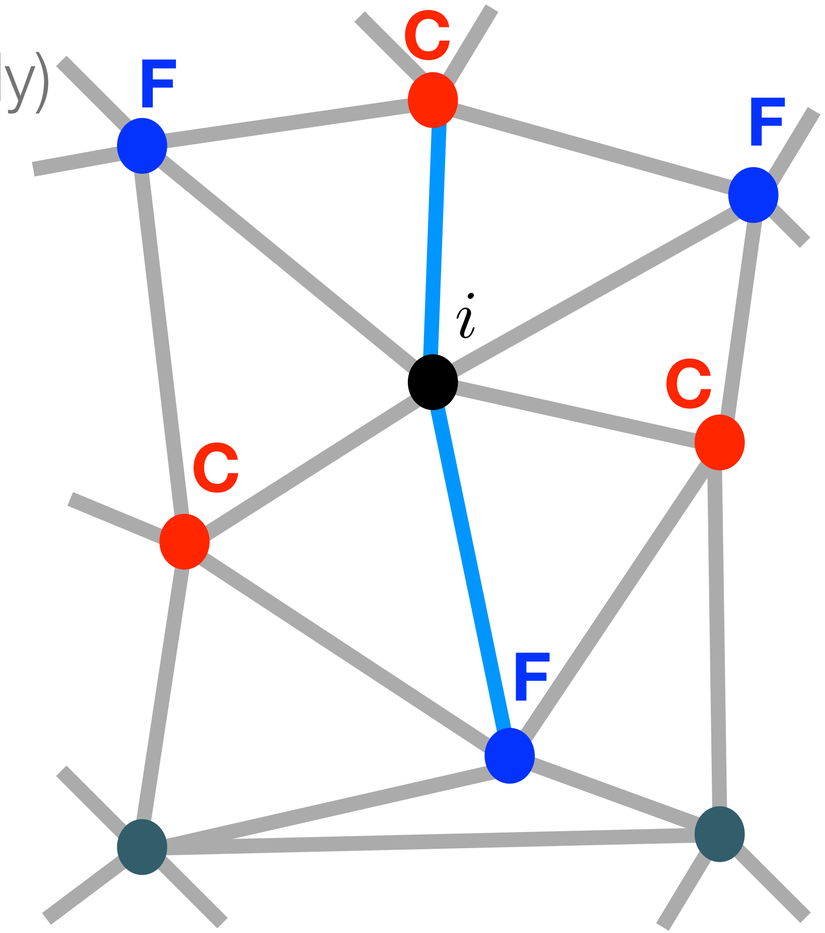


CF AMG

- **Weak:** assume $e_j \approx e_i$ in case there is dependence (=vary slowly)

$$A_{ii}e_i = \sum_{j \in C_i} A_{ij}e_j + \sum_{j \in D_i^s} A_{ij}e_j + \sum_{j \in D_i^w} A_{ij}e_j$$

$$\left(A_{ii} + \sum_{j \in D_i^w} A_{ij} \right) e_i = \sum_{j \in C_i} A_{ij}e_j + \sum_{j \in D_i^s} A_{ij}e_j$$



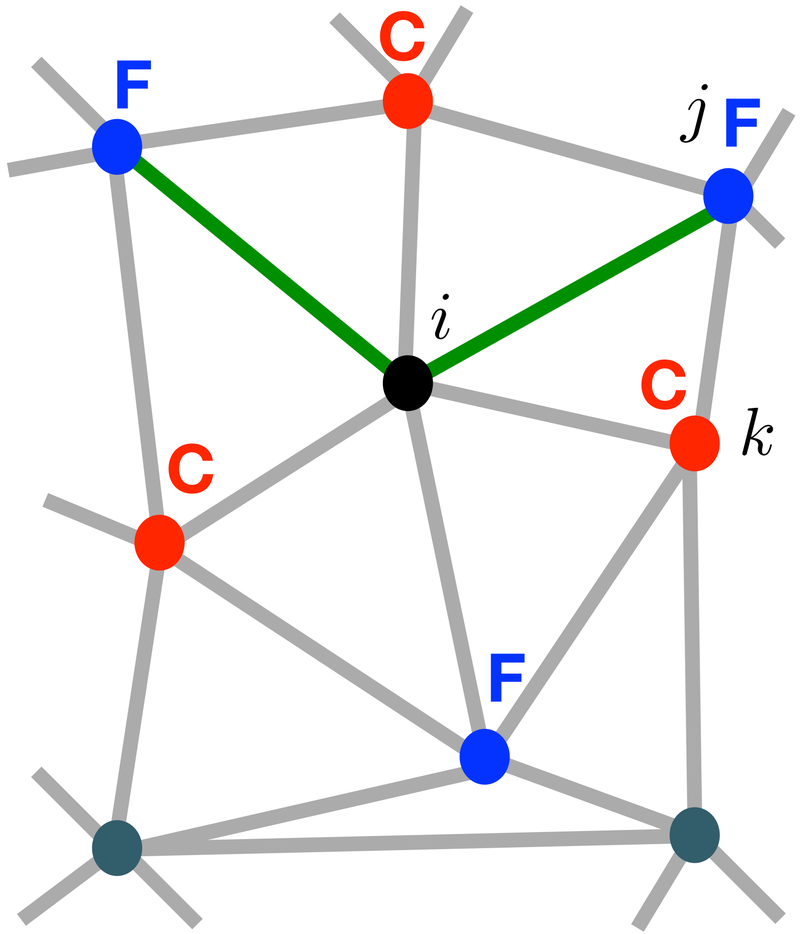
CF AMG

- **Strong F:** approximate e_j by points in $C_i \cap C_j$

$$e_j \approx \frac{\sum_{k \in C_i} A_{jk} e_k}{\sum_{k \in C_i} A_{jk}}$$

- This gives weights

$$\omega_{ij} = -\frac{A_{ij} + \sum_{j \in D_i^s} \frac{A_{ik} A_{kj}}{\sum_{m \in C_i} A_{km}}}{A_{ii} + \sum_{n \in D_i^w} A_{in}}$$



CF AMG Setup Algorithm

Algorithm 2: `CF_setup()`

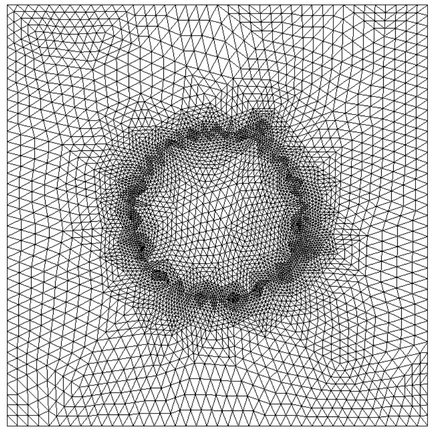
Input: A_0: fine-grid operator
`max_size`: threshold for max size of coarsest problem

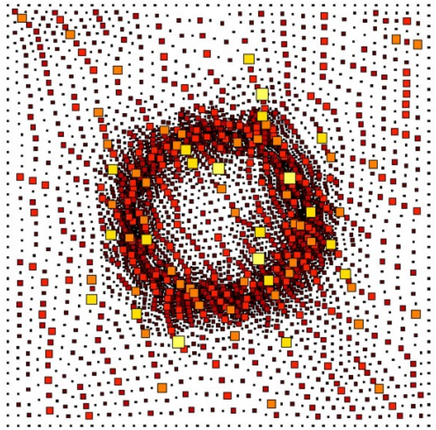
Output: $A_1, \ldots, A_L$,
$P_0, \ldots, P_{L-1}$

1 $\ell = 0$
2 **while** `size`$(A_\ell) >$ `max_size`
3 $\quad S_\ell =$ `strength`(A_ℓ) {Strength-of-connection}
4 $\quad \mathcal{C}_\ell, \mathcal{F}_\ell =$ `splitting`(S_ℓ) {C/F-splitting}
5 $\quad W =$ `weights`$(S_\ell,\ A_\ell,\ \mathcal{C}_\ell,\ \mathcal{F}_\ell)$ {Interpolation weights}
6 $\quad P_\ell = \begin{bmatrix} W \\ I \end{bmatrix}$ {Form interpolation}
7 $\quad A_{\ell+1} = P_\ell^T A_\ell P_\ell$ {Coarse-grid operator}
8 $\quad \ell \ = \ell \ + 1$

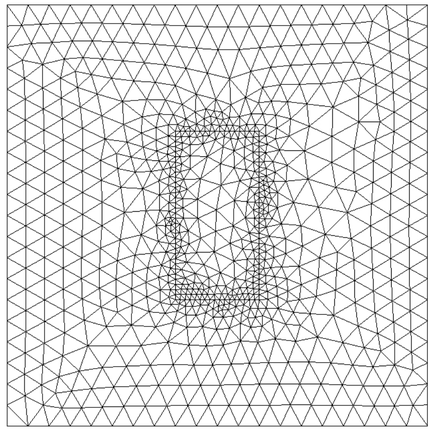
AMG grid hierarchies for several 2D problems

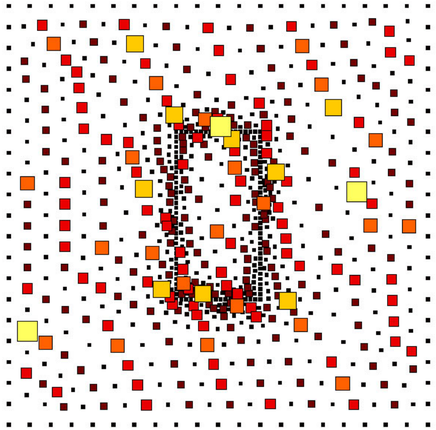
domain1 - 30º



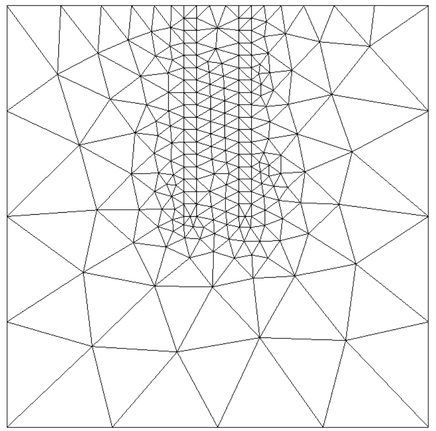


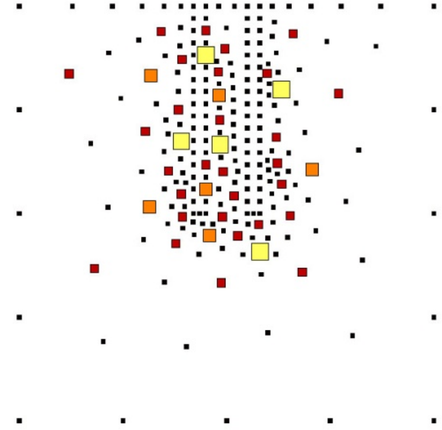
domain2 - 30º



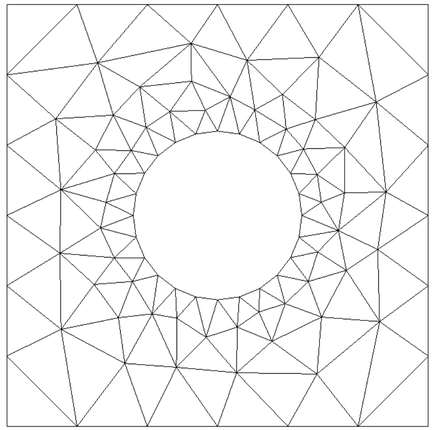


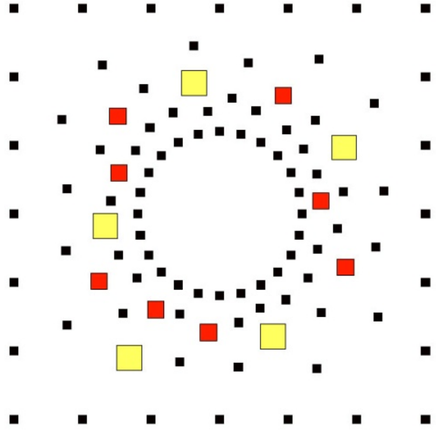
pile





square-hole

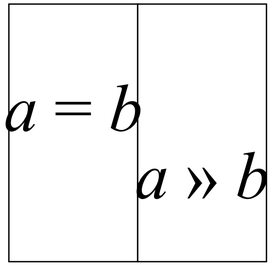


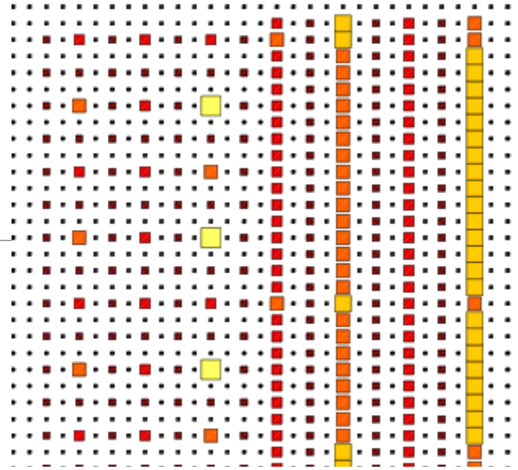


Slide credit: R. Falgout, 2/1/2016

Example CF-AMG results

$$-au_{xx} - bu_{yy} = f$$





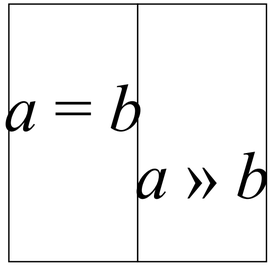
CF-AMG coarse grids

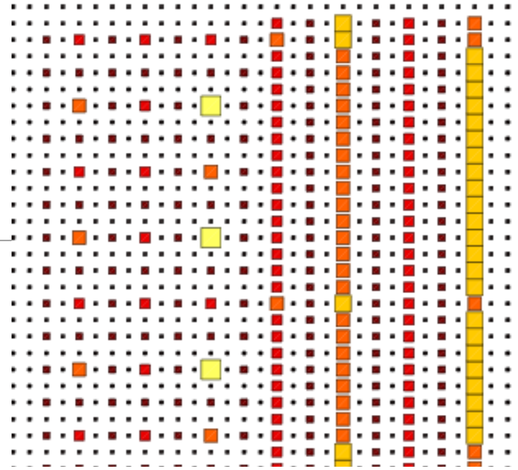
N	Iters	Conv factor	Coarse grids	Grid comp	Oper comp	Setup time	Solve time
61×61	10	0.23	6	1.6	1.6	0.01	0.02
121×121	9	0.23	8	1.6	1.7	0.05	0.07
241×241	9	0.23	9	1.6	1.7	0.25	0.32
481×481	9	0.23	12	1.7	1.7	1.02	1.27
961×961	11	0.29	13	1.7	1.7	4.42	6.28

Slide credit: R. Falgout, 2/1/2016

Example CF-AMG results

$$-au_{xx} - bu_{yy} = f$$





CF-AMG coarse grids

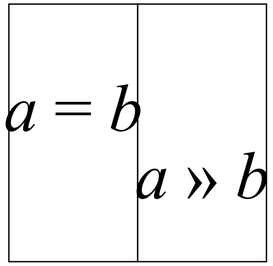
N	Iters	Conv factor	Coarse grids	Grid comp	Oper comp	Setup time	Solve time
61×61	10	0.23	6	1.6	1.6	0.01	0.02
121×121	9	0.23	8	1.6	1.7	0.05	0.07
241×241	9	0.23	9	1.6	1.7	0.25	0.32
481×481	9	0.23	12	1.7	1.7	1.02	1.27
961×961	11	0.29	13	1.7	1.7	4.42	6.28

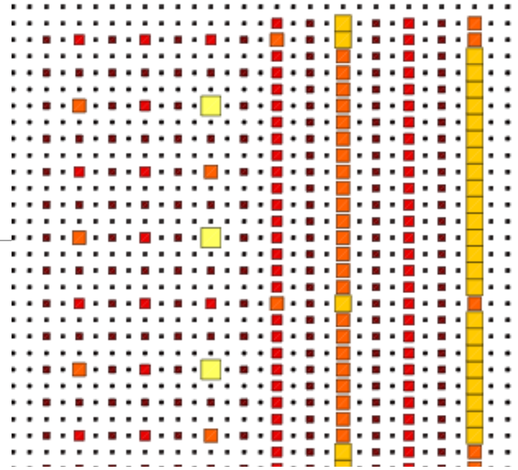
Iterations to a certain tolerance

Slide credit: R. Falgout, 2/1/2016

Example CF-AMG results

$$-au_{xx} - bu_{yy} = f$$





CF-AMG coarse grids

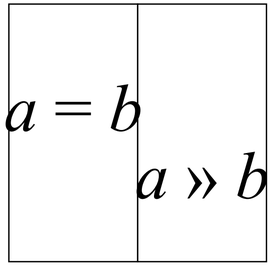
N	Iters	Conv factor	Coarse grids	Grid comp	Oper comp	Setup time	Solve time
61×61	10	0.23	6	1.6	1.6	0.01	0.02
121×121	9	0.23	8	1.6	1.7	0.05	0.07
241×241	9	0.23	9	1.6	1.7	0.25	0.32
481×481	9	0.23	12	1.7	1.7	1.02	1.27
961×961	11	0.29	13	1.7	1.7	4.42	6.28

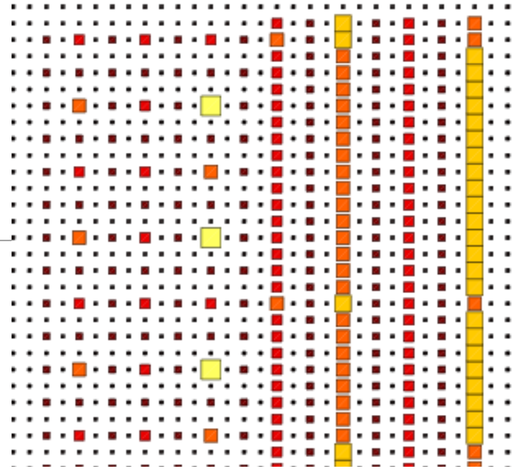
Convergence factor: factor by which the norm of the residual is reduced in each iteration

Slide credit: R. Falgout, 2/1/2016

Example CF-AMG results

$$-au_{xx} - bu_{yy} = f$$





CF-AMG coarse grids

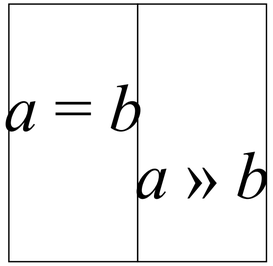
N	Iters	Conv factor	Coarse grids	Grid comp	Oper comp	Setup time	Solve time
61×61	10	0.23	6	1.6	1.6	0.01	0.02
121×121	9	0.23	8	1.6	1.7	0.05	0.07
241×241	9	0.23	9	1.6	1.7	0.25	0.32
481×481	9	0.23	12	1.7	1.7	1.02	1.27
961×961	11	0.29	13	1.7	1.7	4.42	6.28

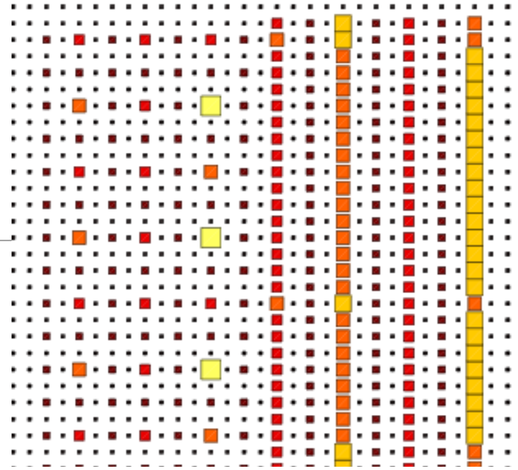
Coarse grids: number of coarse “grids”

Slide credit: R. Falgout, 2/1/2016

Example CF-AMG results

$$-au_{xx} - bu_{yy} = f$$





CF-AMG coarse grids

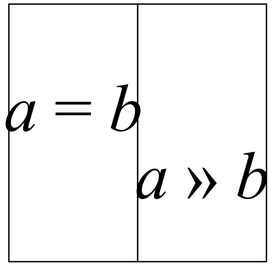
N	Iters	Conv factor	Coarse grids	Grid comp	Oper comp	Setup time	Solve time
61×61	10	0.23	6	1.6	1.6	0.01	0.02
121×121	9	0.23	8	1.6	1.7	0.05	0.07
241×241	9	0.23	9	1.6	1.7	0.25	0.32
481×481	9	0.23	12	1.7	1.7	1.02	1.27
961×961	11	0.29	13	1.7	1.7	4.42	6.28

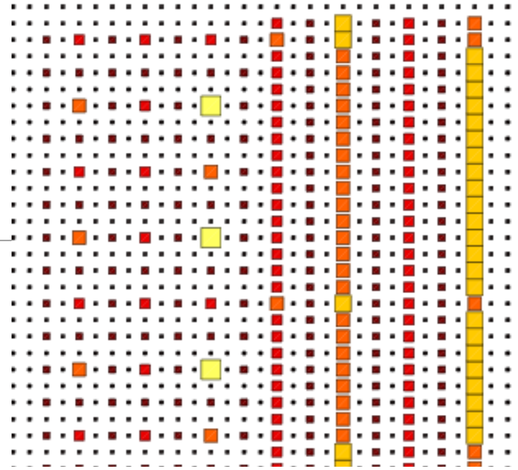
Grid complexity: total number of grid points (system size) on all levels / number of grids points on the fine level

Slide credit: R. Falgout, 2/1/2016

Example CF-AMG results

$$-au_{xx} - bu_{yy} = f$$





CF-AMG coarse grids

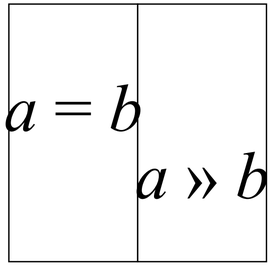
N	Iters	Conv factor	Coarse grids	Grid comp	Oper comp	Setup time	Solve time
61×61	10	0.23	6	1.6	1.6	0.01	0.02
121×121	9	0.23	8	1.6	1.7	0.05	0.07
241×241	9	0.23	9	1.6	1.7	0.25	0.32
481×481	9	0.23	12	1.7	1.7	1.02	1.27
961×961	11	0.29	13	1.7	1.7	4.42	6.28

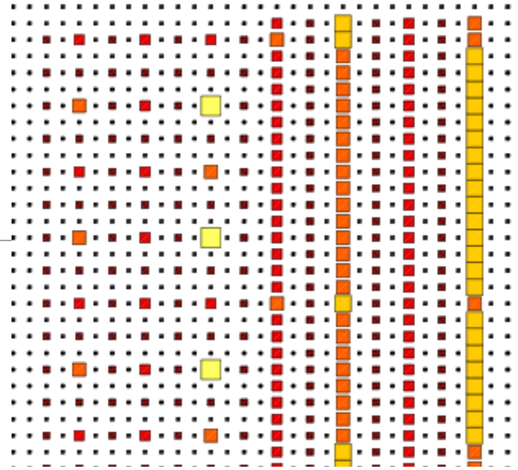
Operator complexity: total number non-zeros on all levels / number of non-zeros on the fine level

Slide credit: R. Falgout, 2/1/2016

Example CF-AMG results

$$-au_{xx} - bu_{yy} = f$$





CF-AMG coarse grids

N	Iters	Conv factor	Coarse grids	Grid comp	Oper comp	Setup time	Solve time
61×61	10	0.23	6	1.6	1.6	0.01	0.02
121×121	9	0.23	8	1.6	1.7	0.05	0.07
241×241	9	0.23	9	1.6	1.7	0.25	0.32
481×481	9	0.23	12	1.7	1.7	1.02	1.27
961×961	11	0.29	13	1.7	1.7	4.42	6.28

Setup/solve times: can vary *a lot* depending on the problem, but the setup is significant!

Slide credit: R. Falgout, 2/1/2016

Demo

- `14-anisotropy-cf-amg.ipynb`

Some theory

- An outline of some (general) theory.

On Generalizing the Algebraic Multigrid Framework, Falgout, Vassilevski, 2004, SINUM.

-

- Consider smoothing of the form

$$u \leftarrow u + M^{-1} r$$

or

$$e \leftarrow e - M^{-1} A e$$

or

$$e \leftarrow (I - M^{-1} A) e$$

Some theory

- Let interpolation be P

- Consider some “restriction” operation (not the multigrid operator) such that

$$RP = I$$

- *Note:* PR is a projection onto P

- Example: $R = \begin{bmatrix} 0 & I \end{bmatrix}$

On Generalizing the Algebraic Multigrid Framework, Falgout, Vassilevski, 2004, SINUM.

Some theory

- The two-level theory shows (more Friday)

$$\|(I - M^{-1}A)(I - P(P^TAP)^{-1}P^TA)\|_A^2 \leq 1 - \frac{1}{K}$$

- where

$$K = \sup_e \frac{\|(I - PR)e\|_{\tilde{M}}^2}{\|e\|_A^2}$$

an approximation property

- General interpretation: higher quality interpolation (measured with M) leads to improved convergence.

Some theory

- Try to optimize this:

$$K = \sup_e \frac{\|(I - PR)e\|^2_{\tilde{M}}}{\|e\|^2_A}$$

- If $R = \begin{bmatrix} 0 & I \end{bmatrix}$

$$P^T = \begin{bmatrix} W^T & I \end{bmatrix}$$

- Then $P_{ideal} = \operatorname{argmin}_P \sup_e \frac{\|(I - PR)e\|^2_{\tilde{M}}}{\|e\|^2_A}$

$$= \begin{bmatrix} -A_{FF}^{-1} A_{FC} \\ I \end{bmatrix}$$

On Generalizing the Algebraic Multigrid Framework, Falgout, Vassilevski, 2004, SINUM.

M

- Try to optimize this:

$$K = \sup_e \frac{\|(I - PR)e\|^2_{\tilde{M}}}{\|e\|^2_A}$$

Make this small
"compatible relaxation" methods

- If $R = \begin{bmatrix} 0 & I \end{bmatrix}$

$$P^T = \begin{bmatrix} W^T & I \end{bmatrix}$$

- Then $P_{ideal} = \operatorname{argmin}_P \sup_e \frac{\|(I - PR)e\|^2_{\tilde{M}}}{\|e\|^2_A}$

$$= \begin{bmatrix} -A_{FF}^{-1} A_{FC} \\ I \end{bmatrix}$$

Impractical, but often a good guide
AMGe-type methods

Demo

- `15-CF-AMG-Interpolation.ipynb`

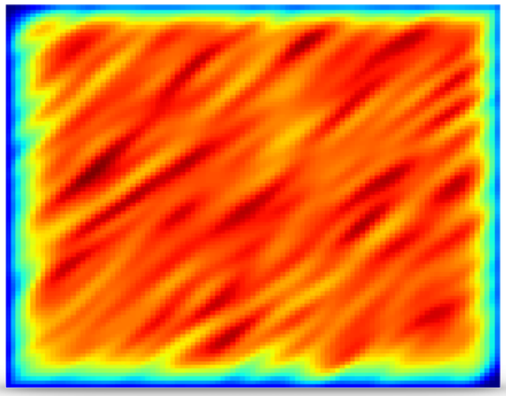
SA AMG

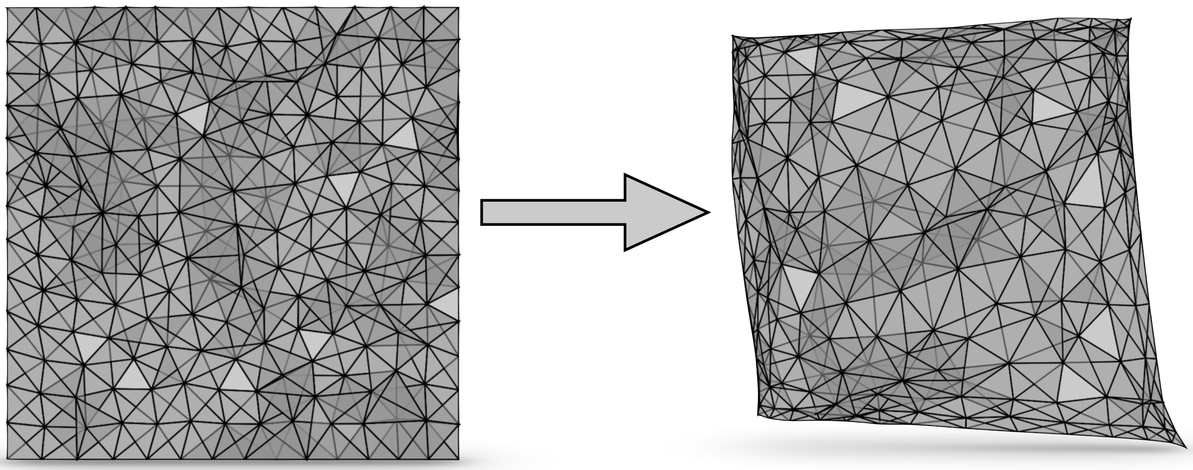
- Smoothed Aggregation based AMG takes a *different* approach
- Still, the same steps:
  1. strength between points
  2. find a coarse grid (this time **aggregates** of points)
  3. define interpolation
  4. compute the coarse grid operator

P. Vaněk, J. Mandel, M. Brezina, Algebraic multigrid by smoothed aggregation for second and fourth order elliptic problems, Computing, 1996

SA AMG

- **SA AMG relies on “candidate” vectors or the near null space or smooth error**
- vector of ones
- pre-smooth guesses
- adapting a cycle
- *a priori* knowledge
- topological inference





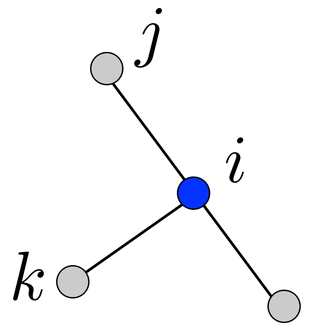
Symmetric Strength

- i strongly depends on j if

$$-A_{ij} \geq tol * \max_{k \neq i} -A_{ik}$$

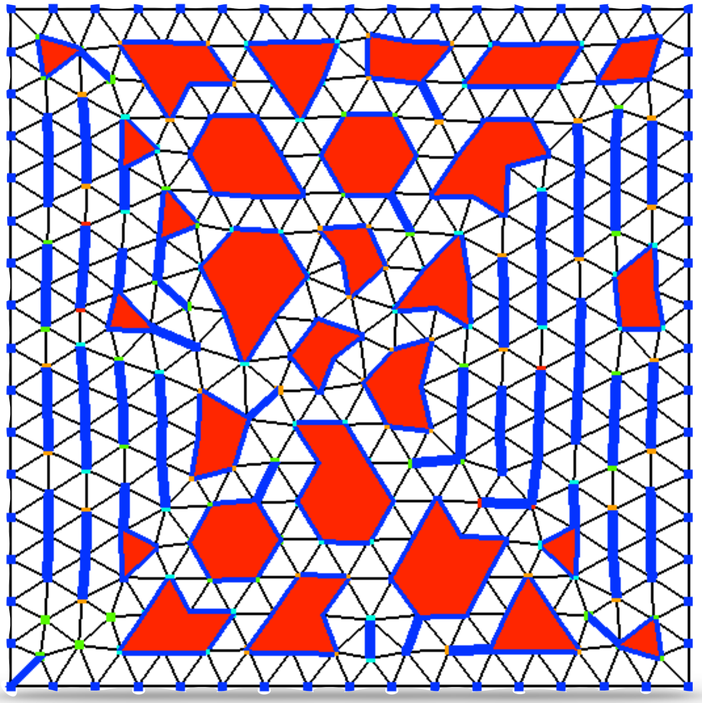
- i strongly depends on j if

$$\frac{|A_{ij}|}{\sqrt{A_{ii}A_{jj}}} \geq tol$$



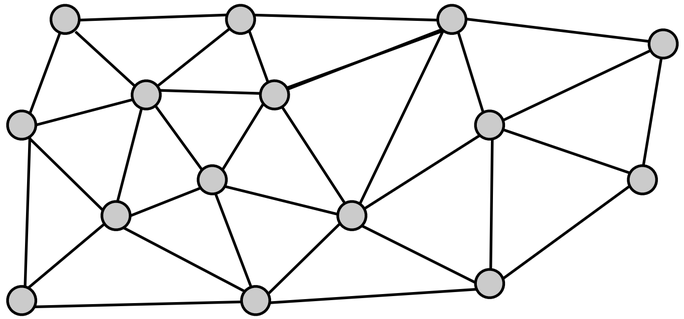
both “think” elliptic

anisotropy



Algebraic Framework

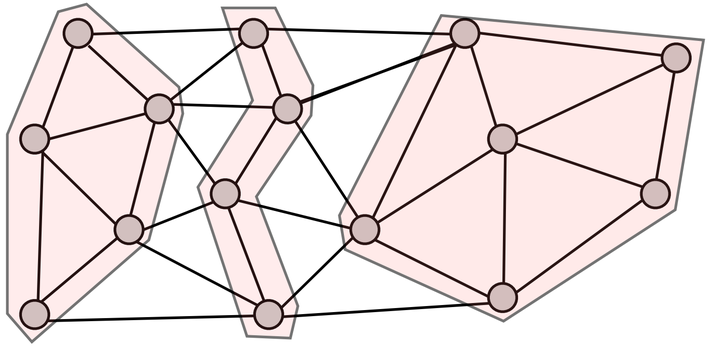
- aggregation: groups of fine nodes form coarse nodes



- an initial interpolation pattern
- find an optimal interpolation operator P that contains low energy

Algebraic Framework

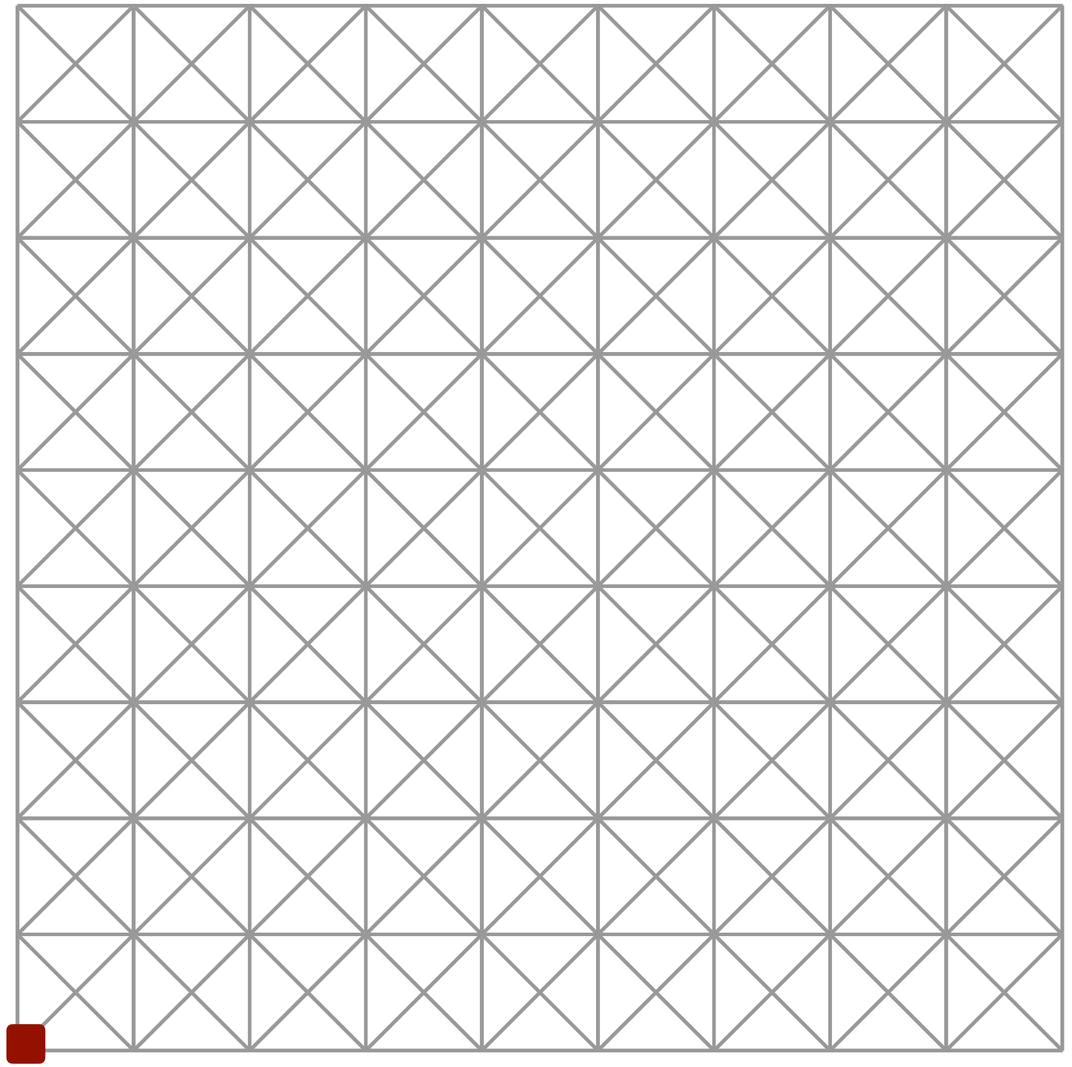
- aggregation: groups of fine nodes form coarse nodes



fine: 15
coarse: 3

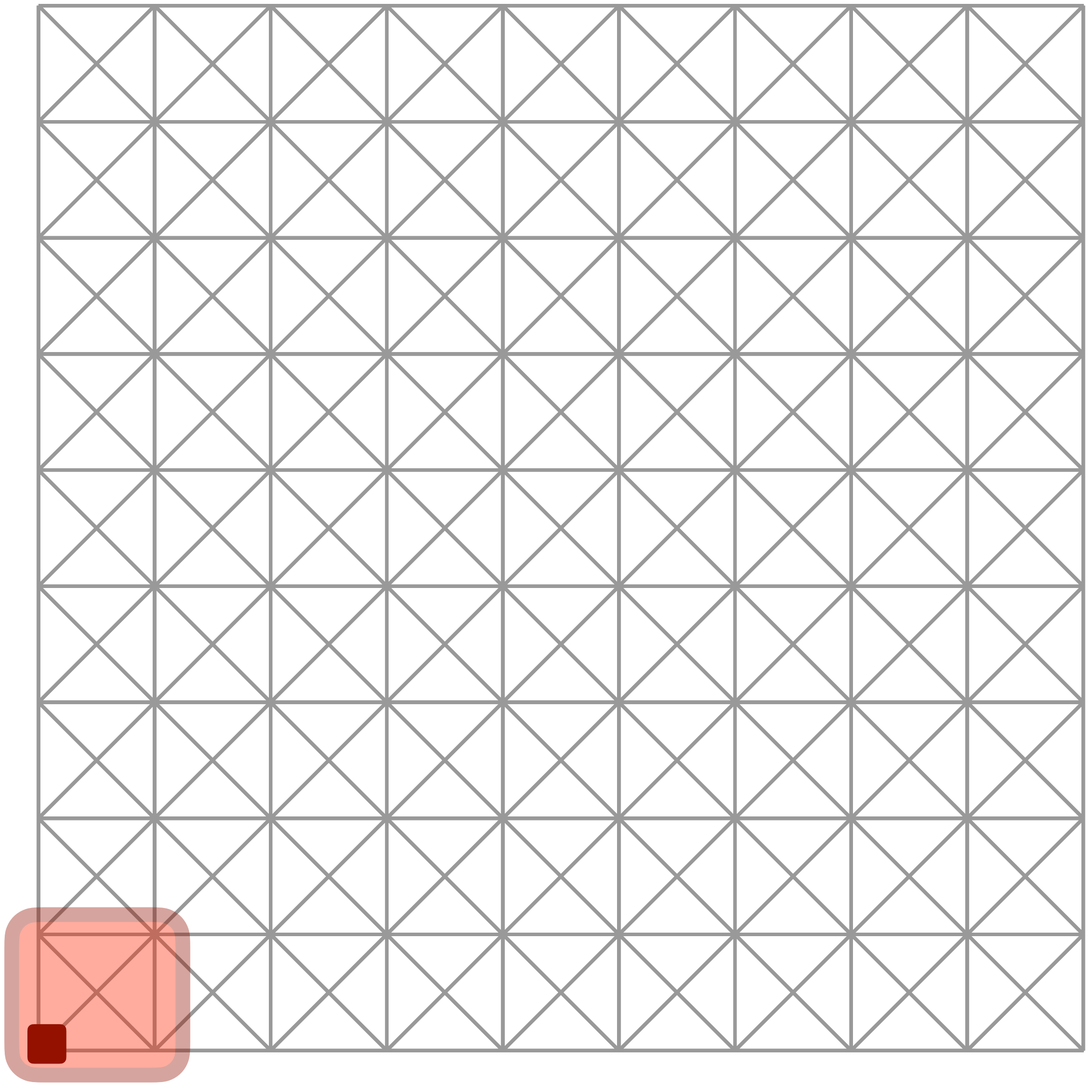
- an initial interpolation pattern
- find an optimal interpolation operator P that contains low energy

Aggregation



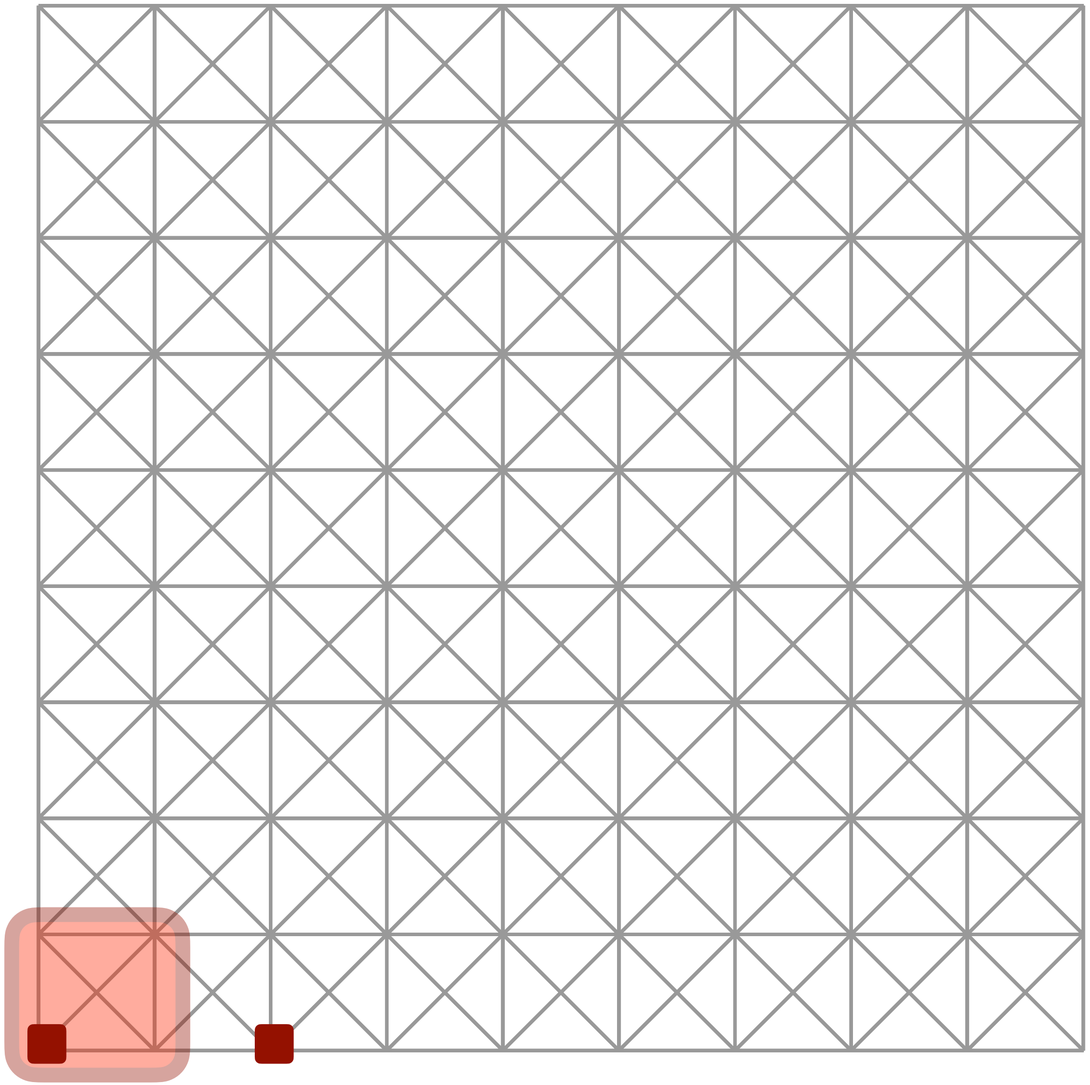
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



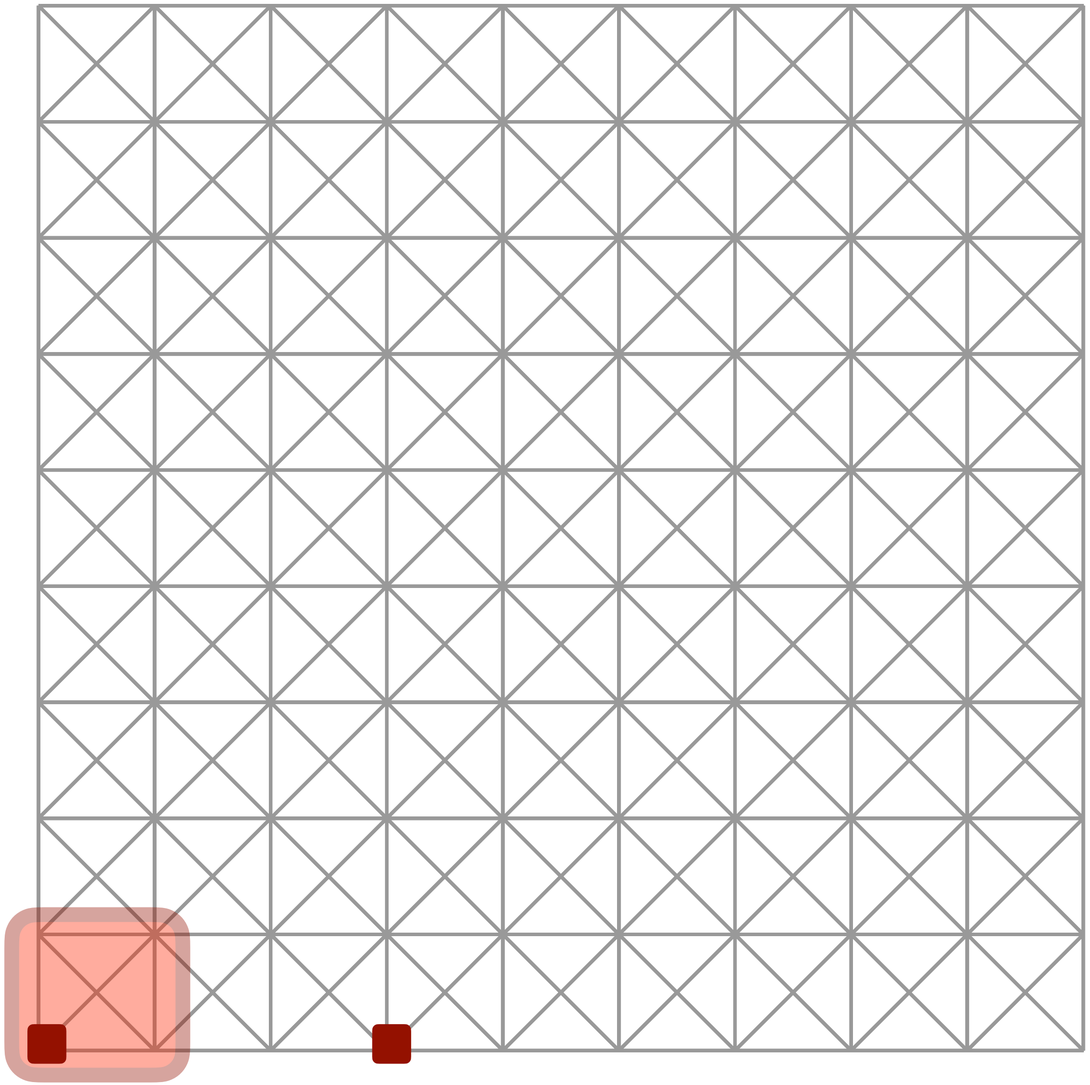
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



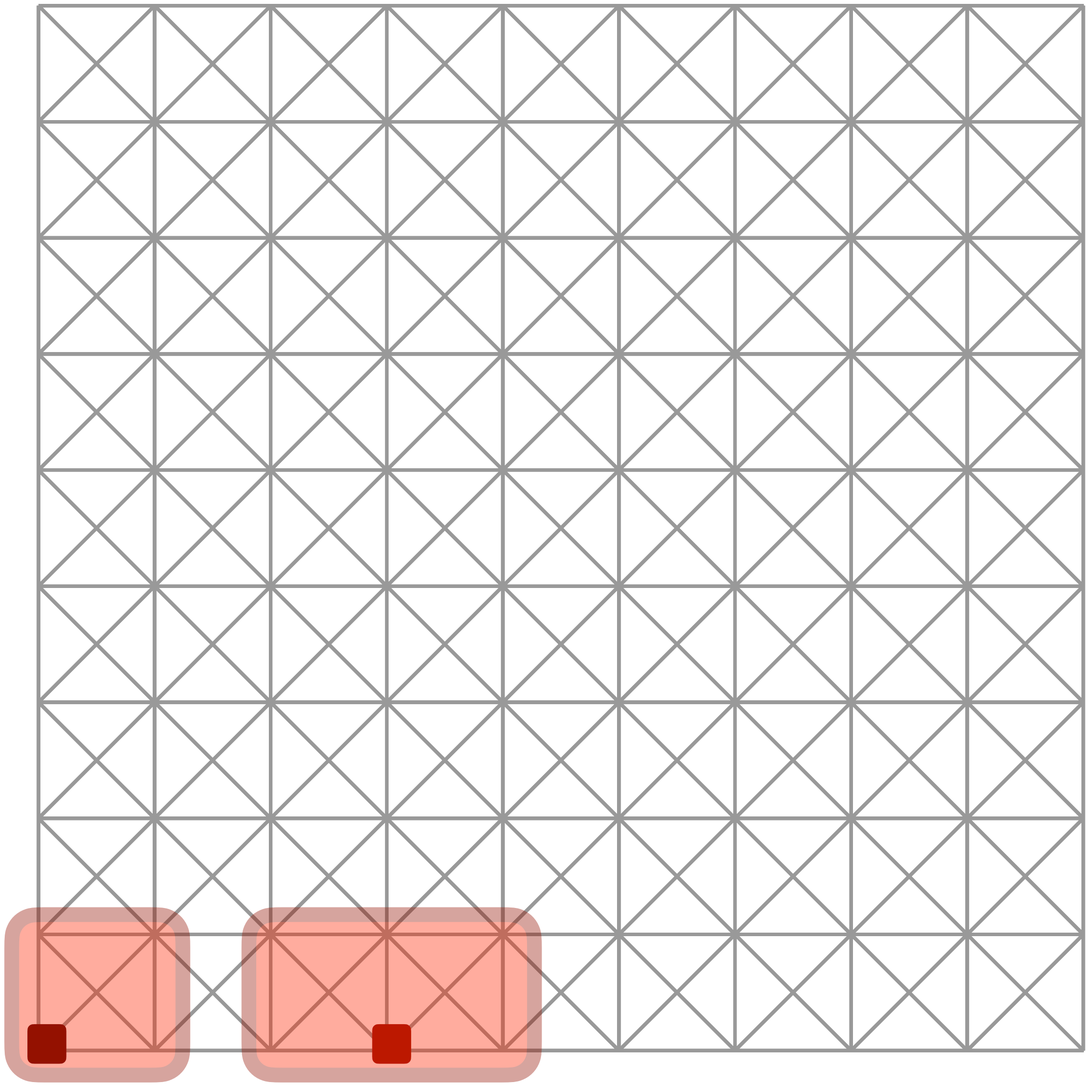
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



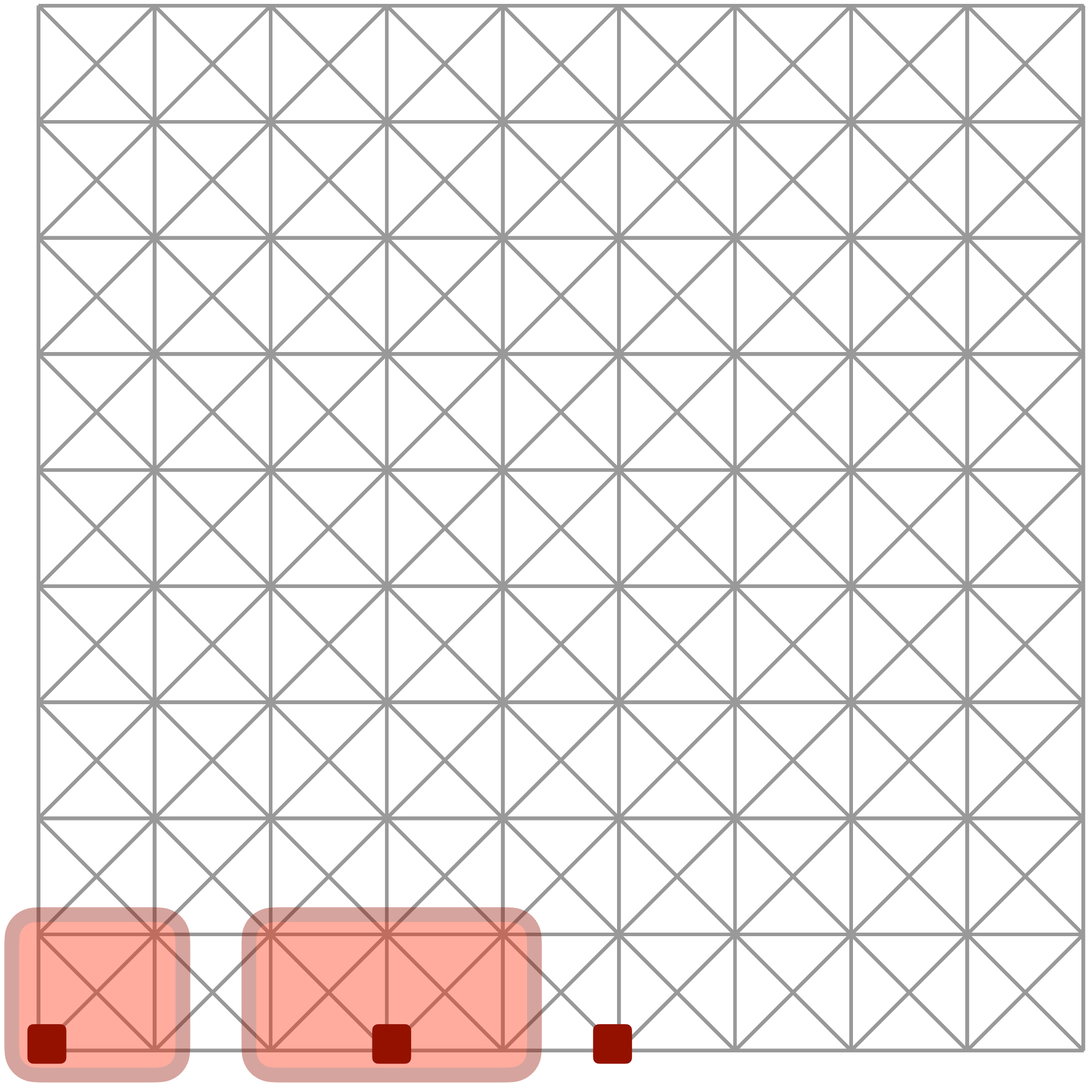
- Select next unaggregated node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



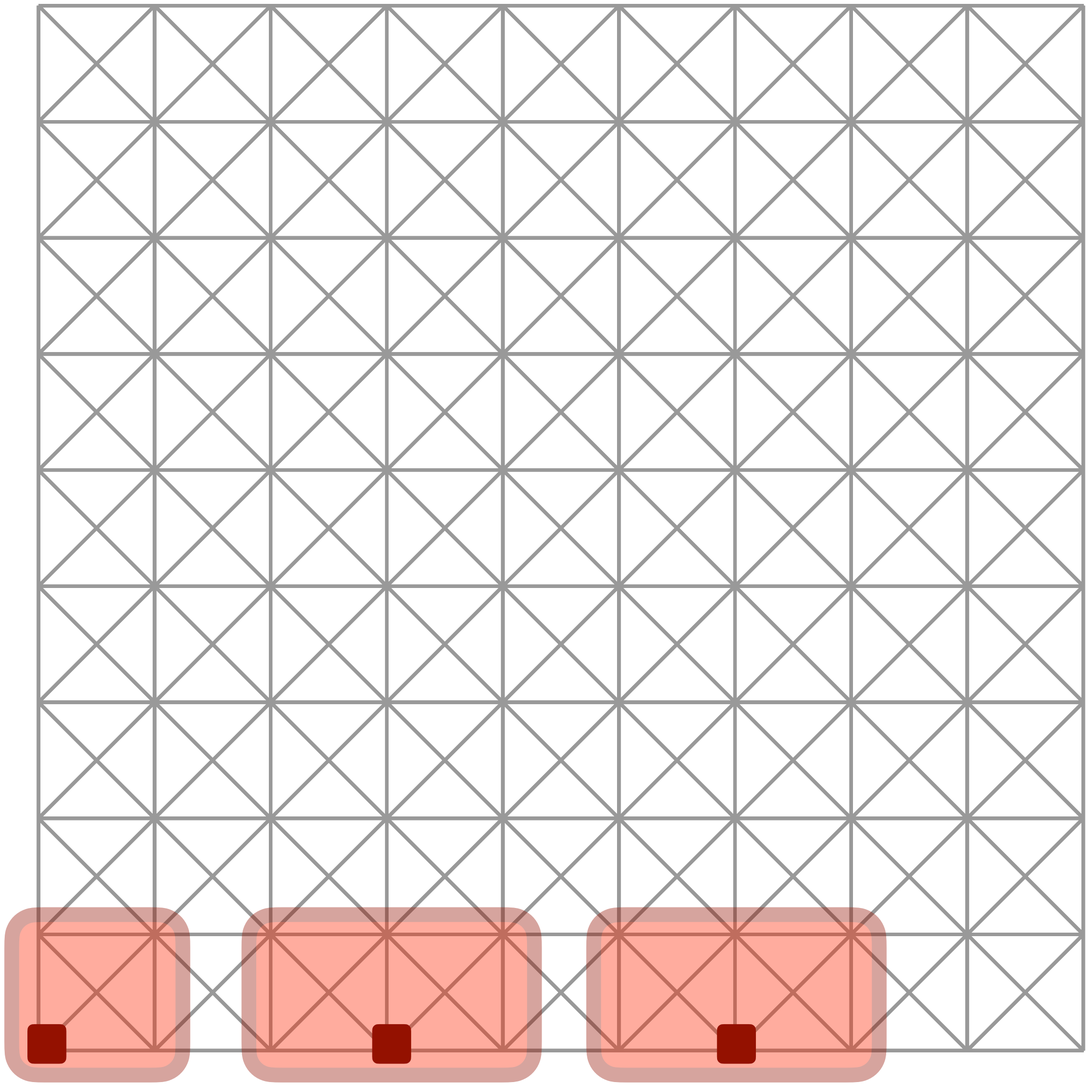
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



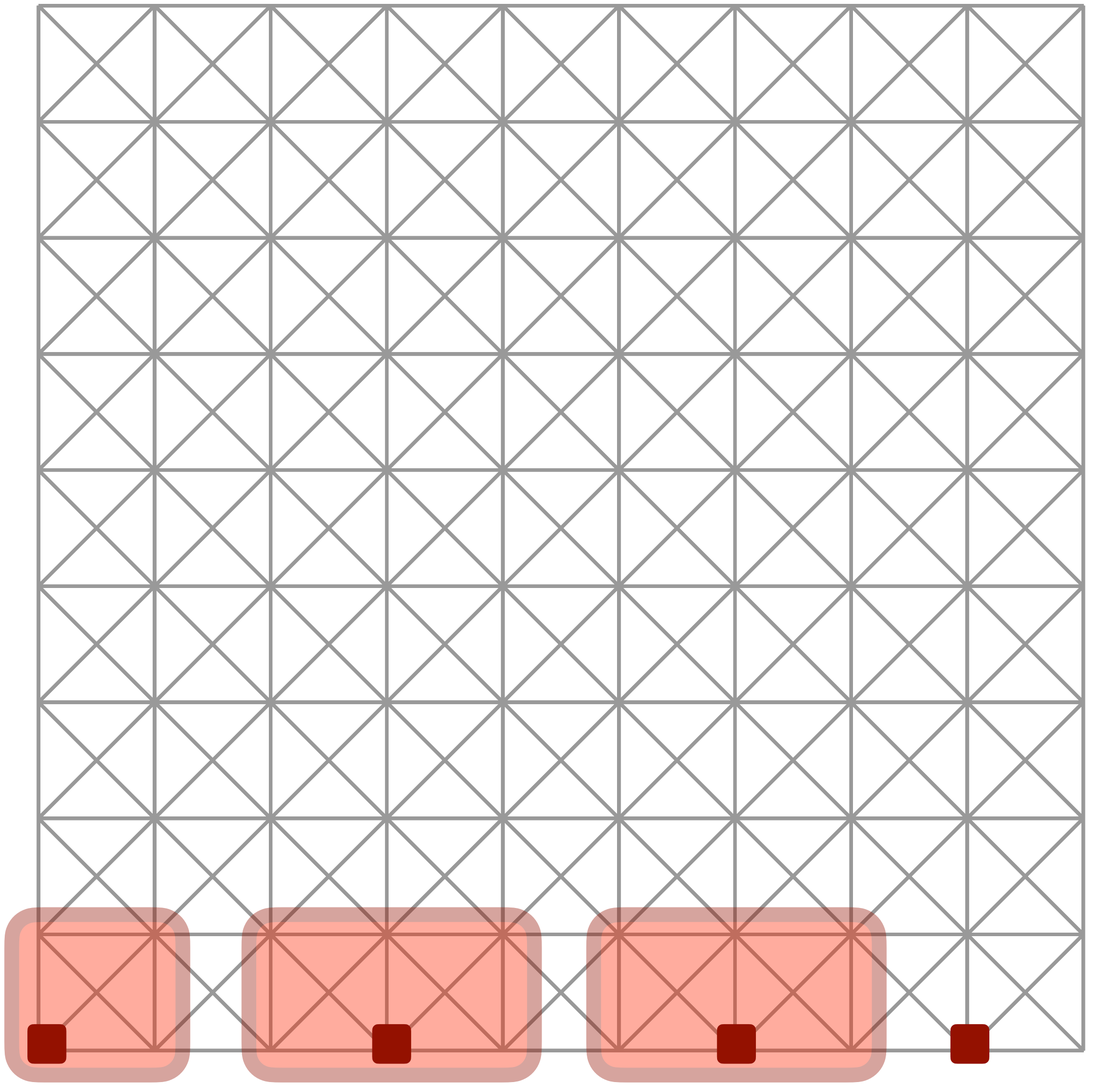
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



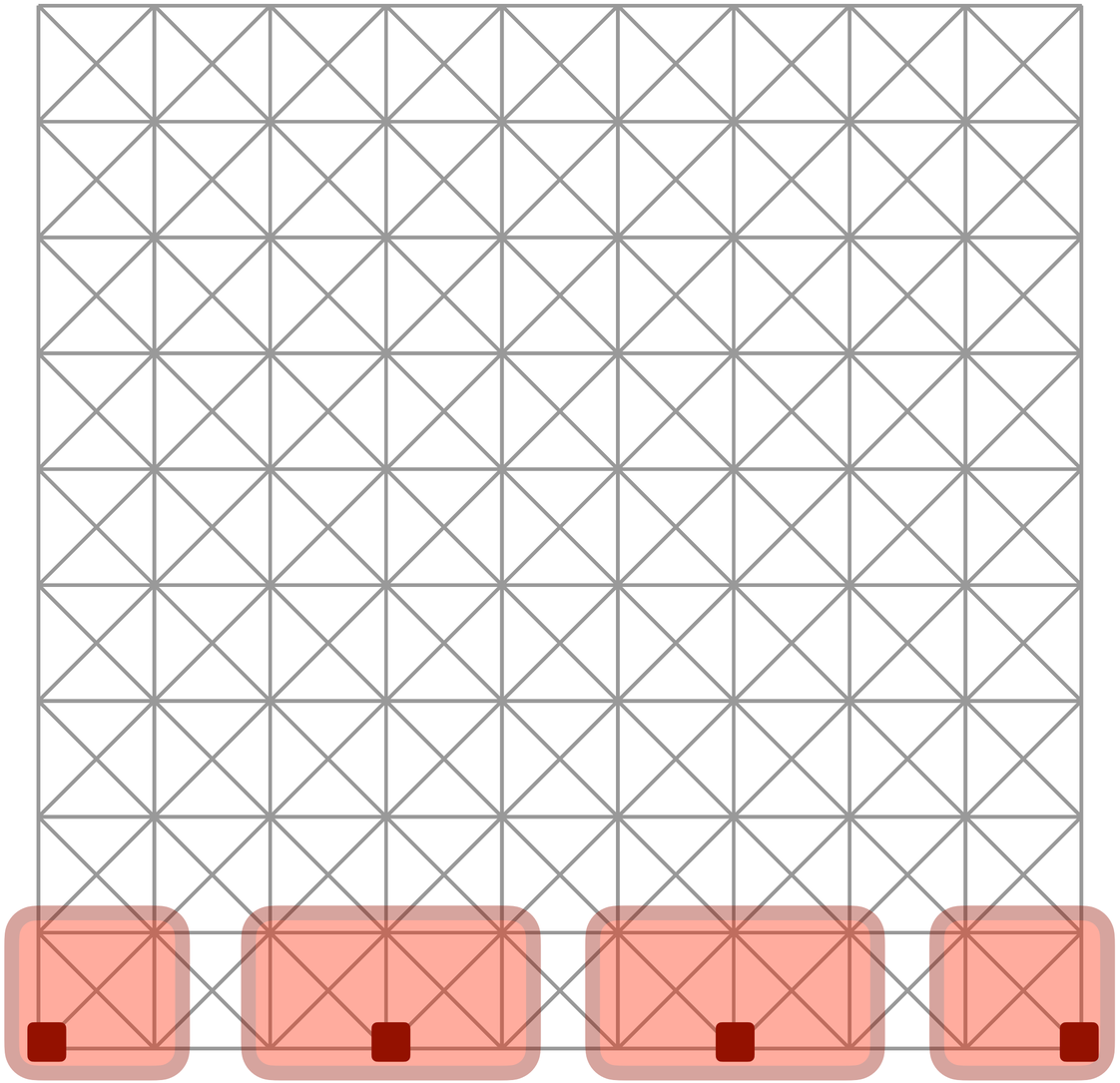
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



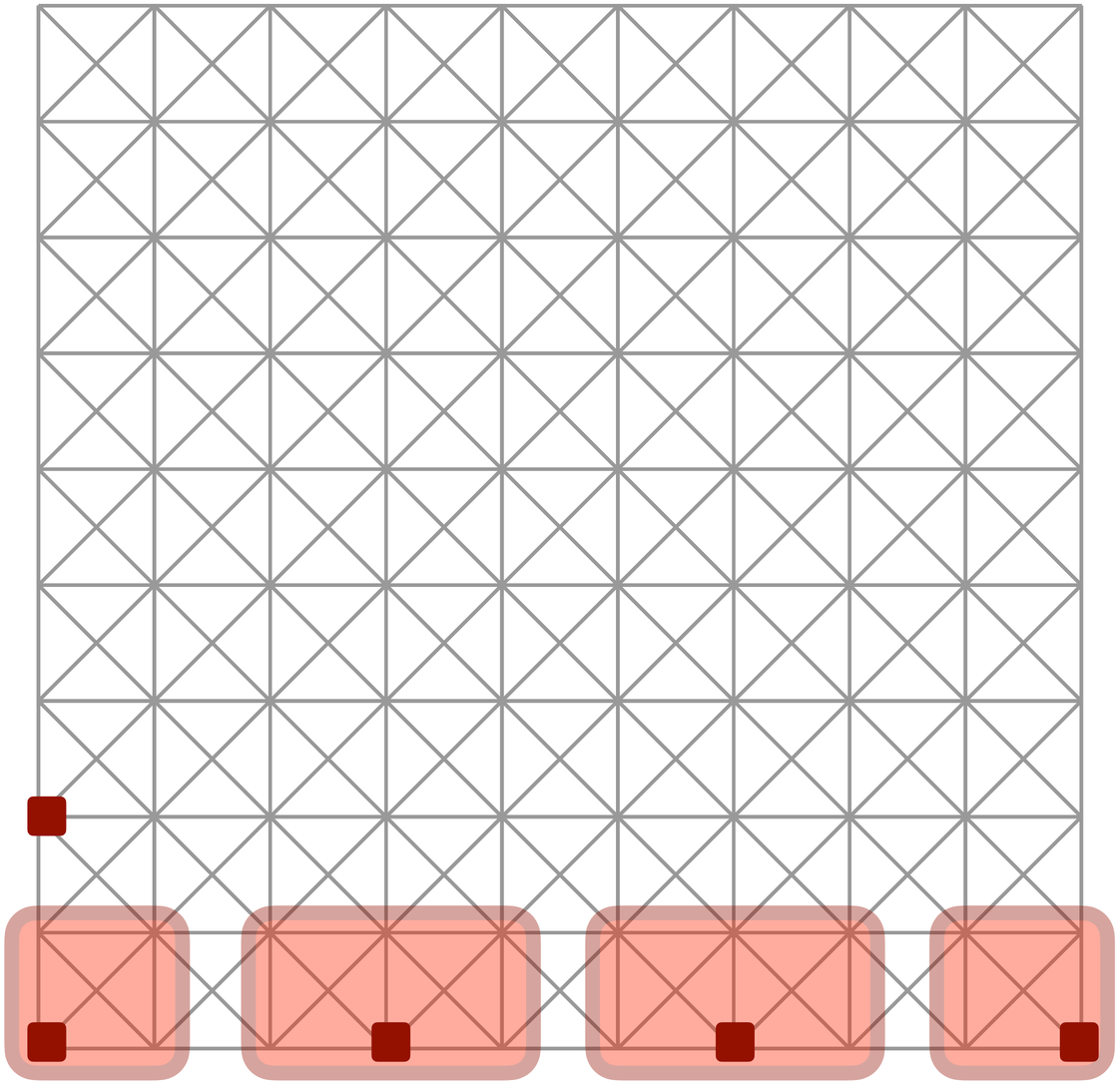
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



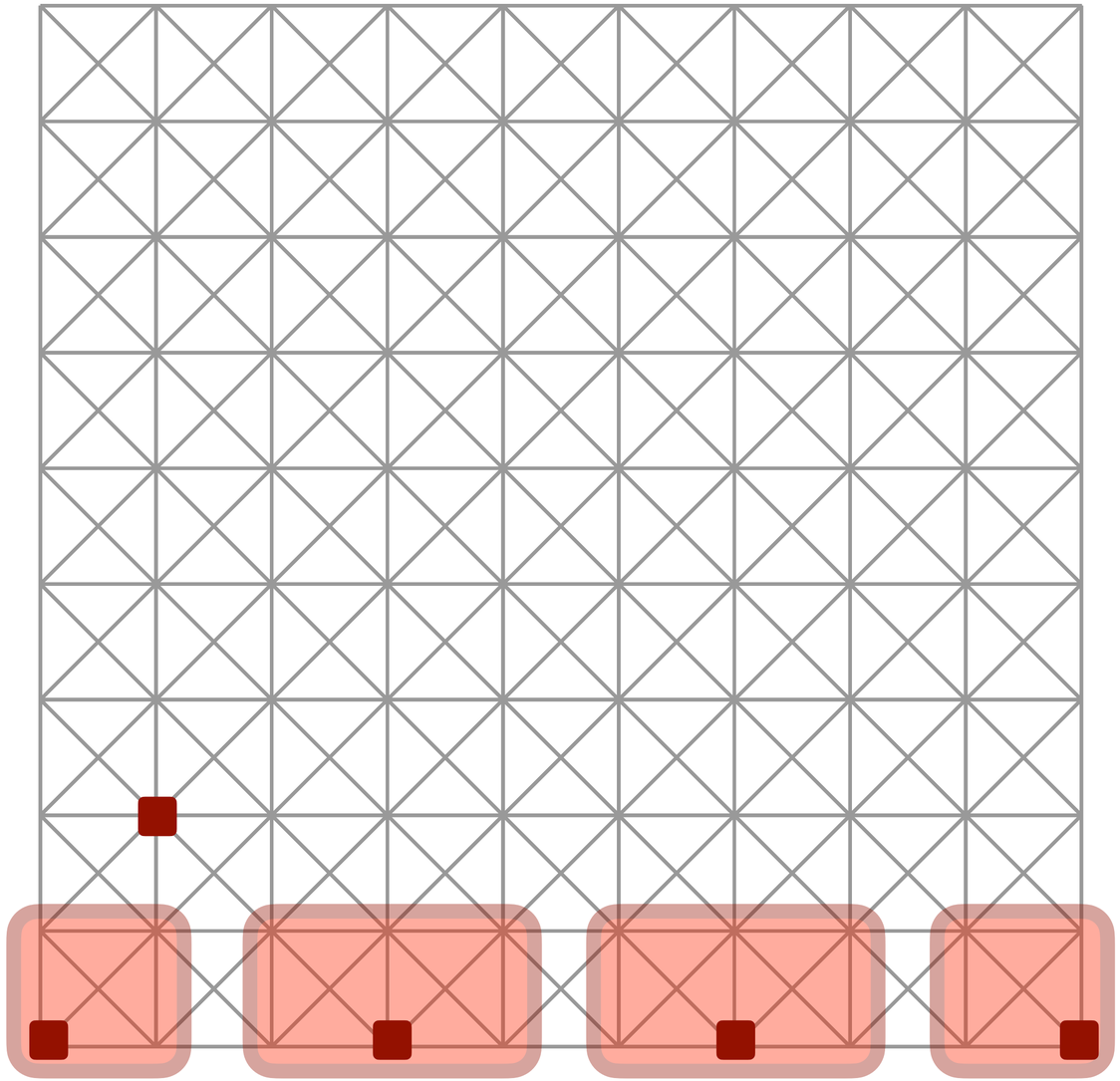
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



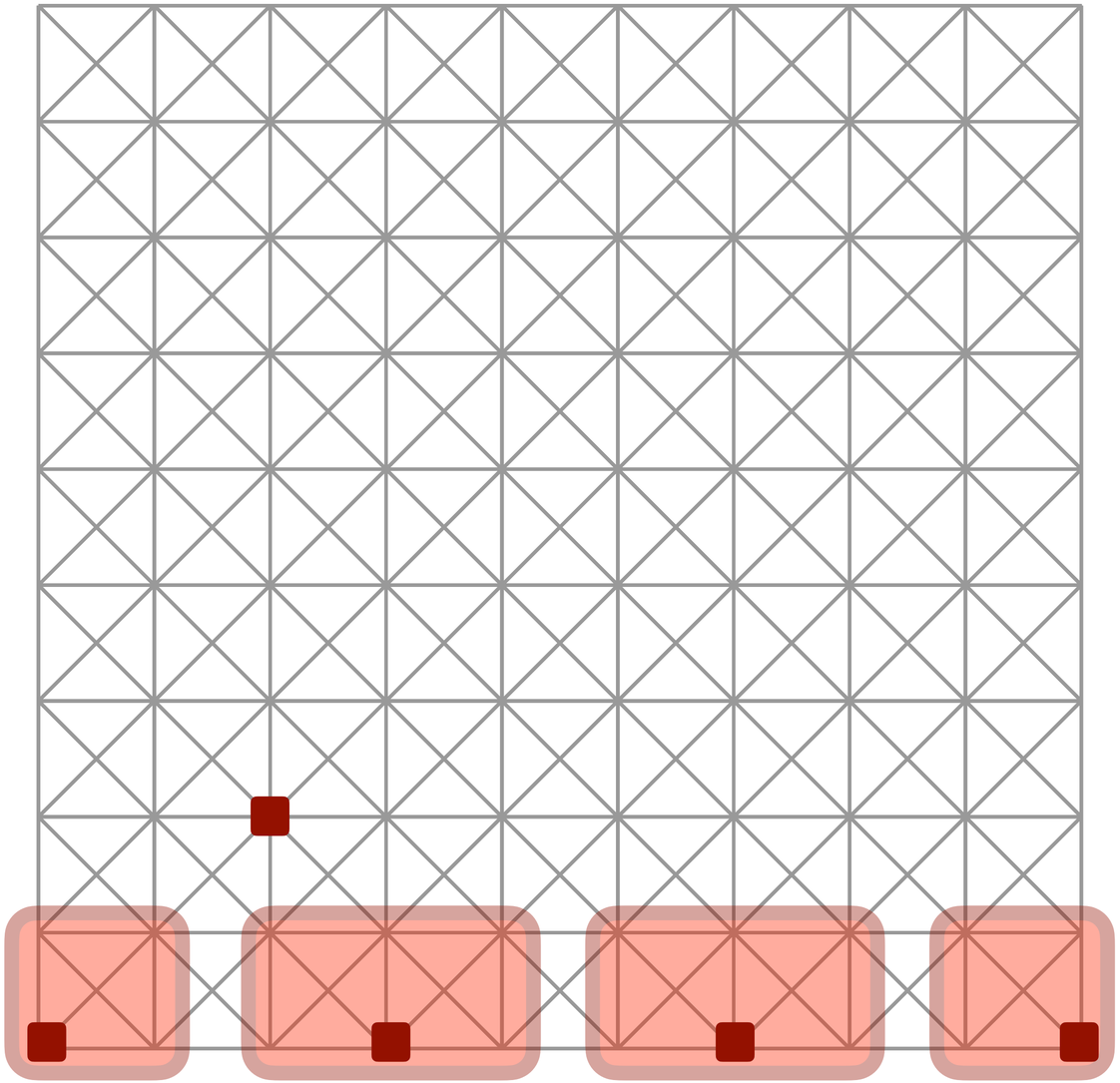
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



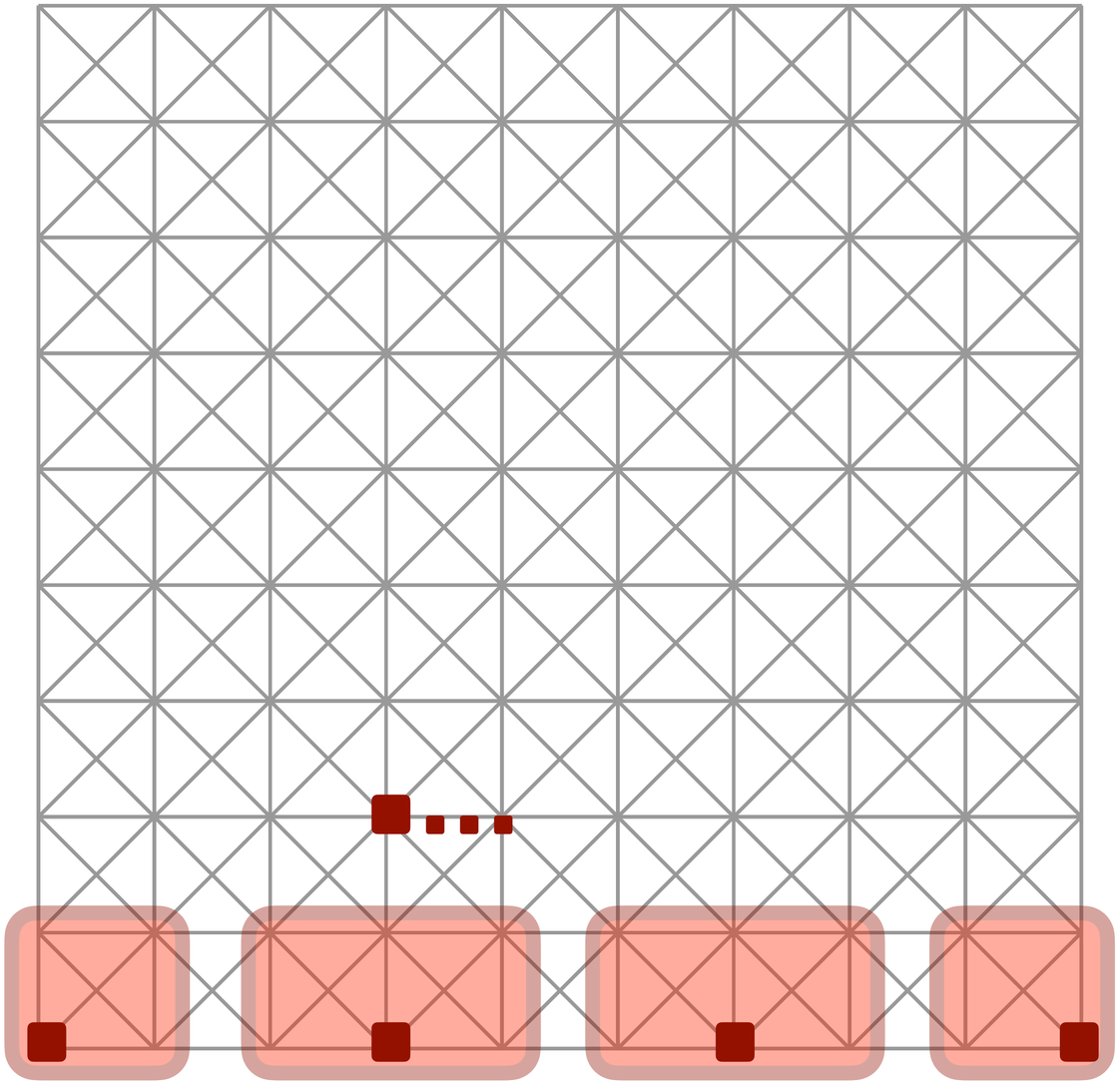
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



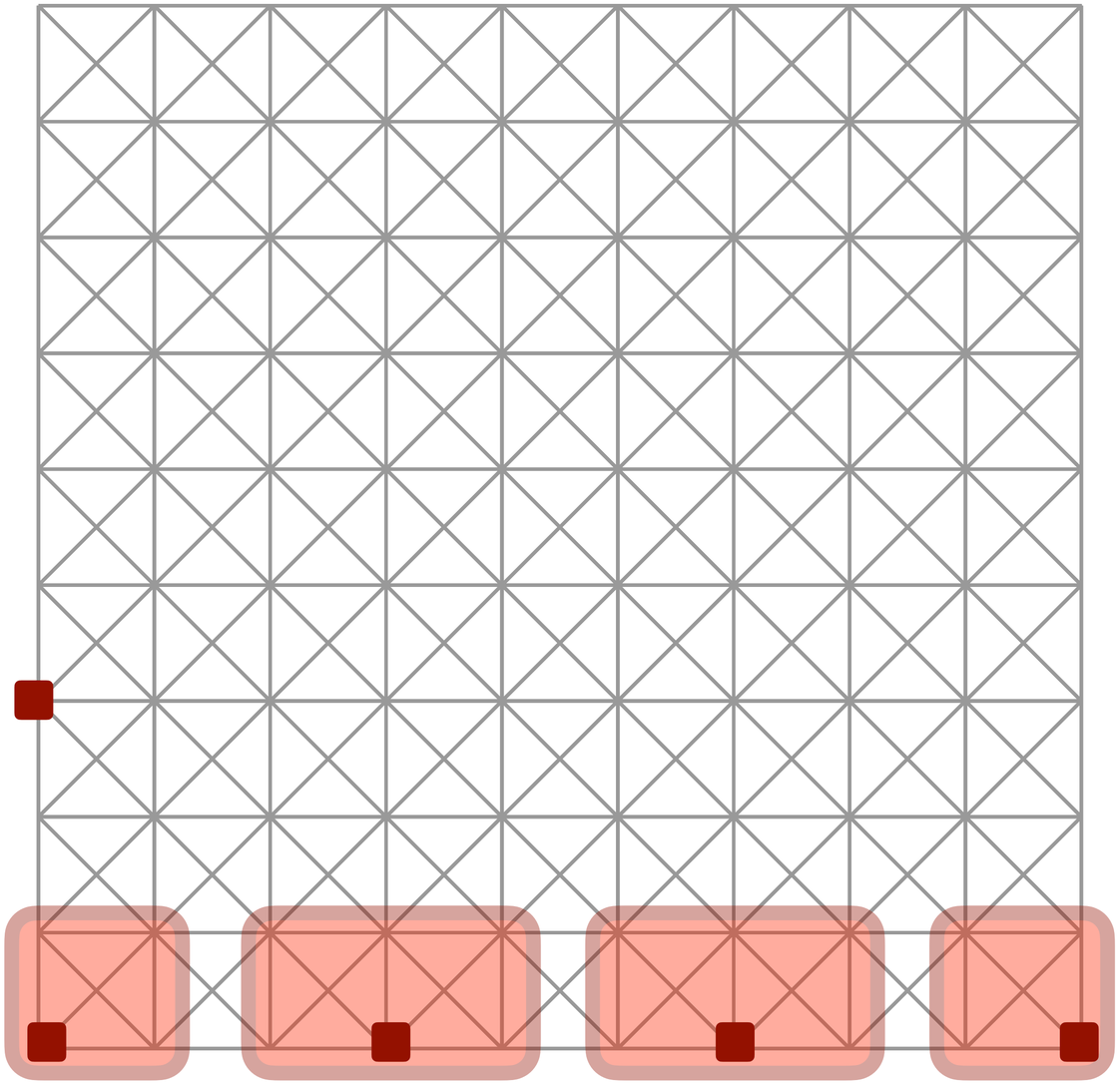
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



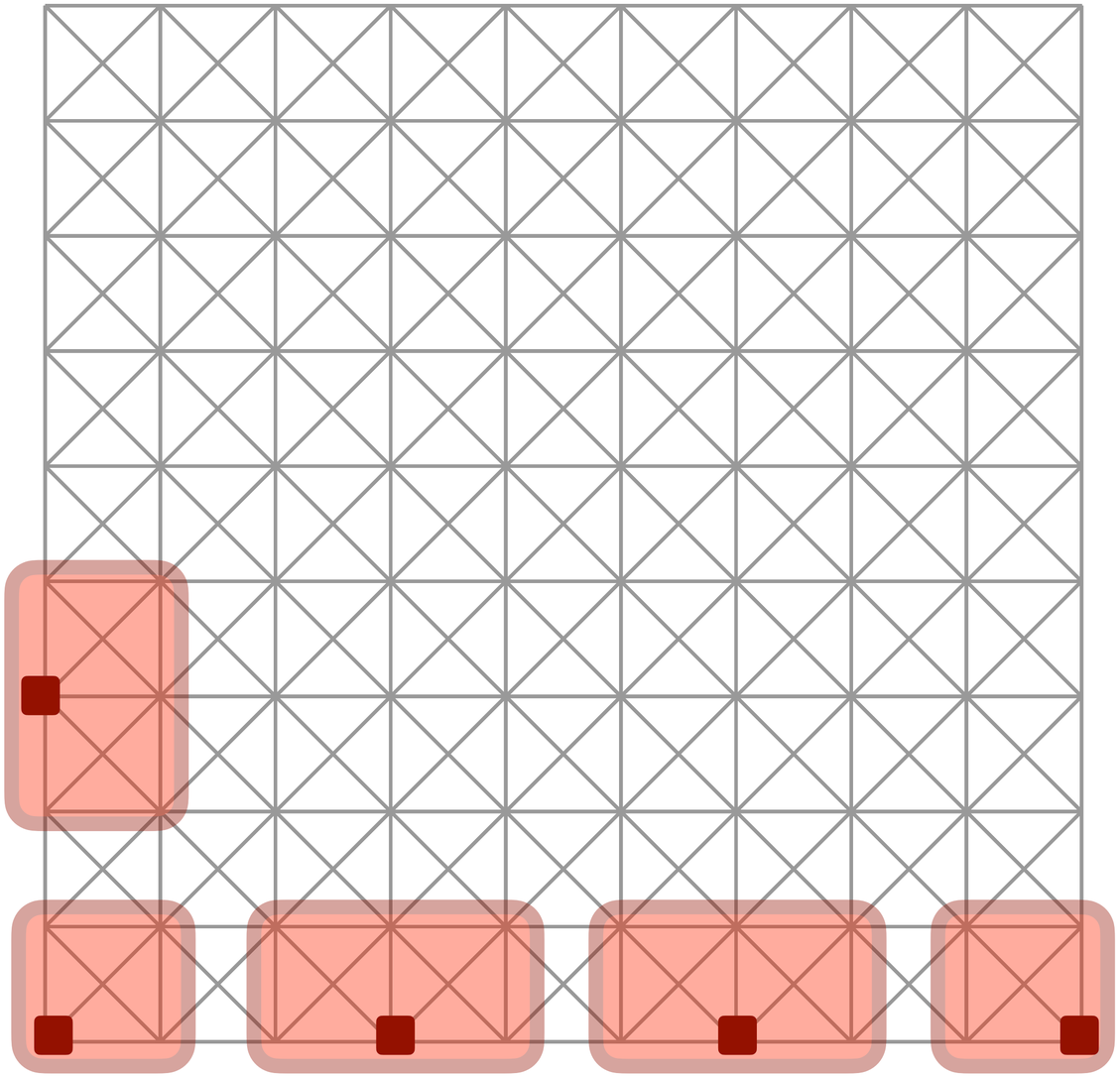
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



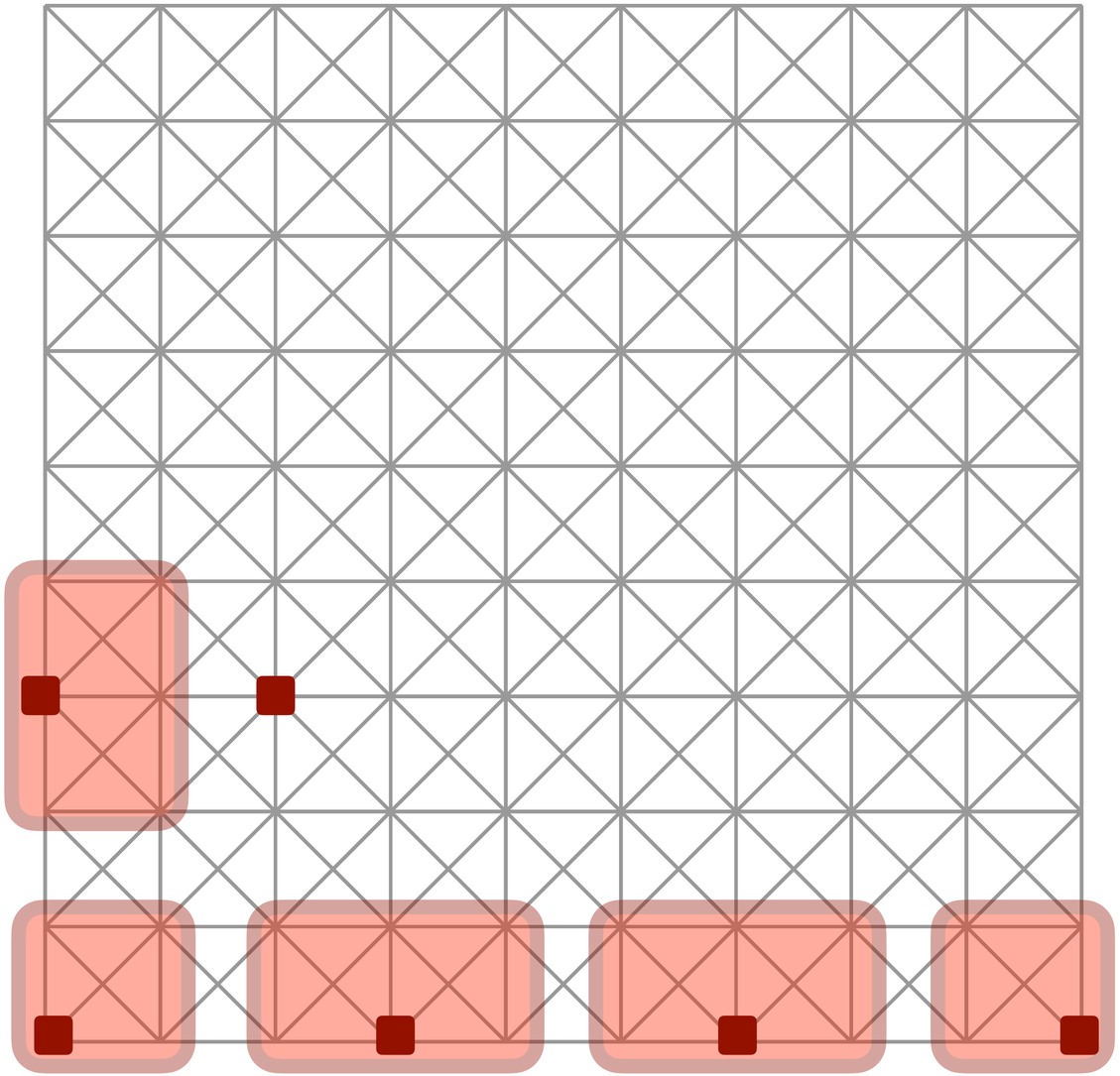
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



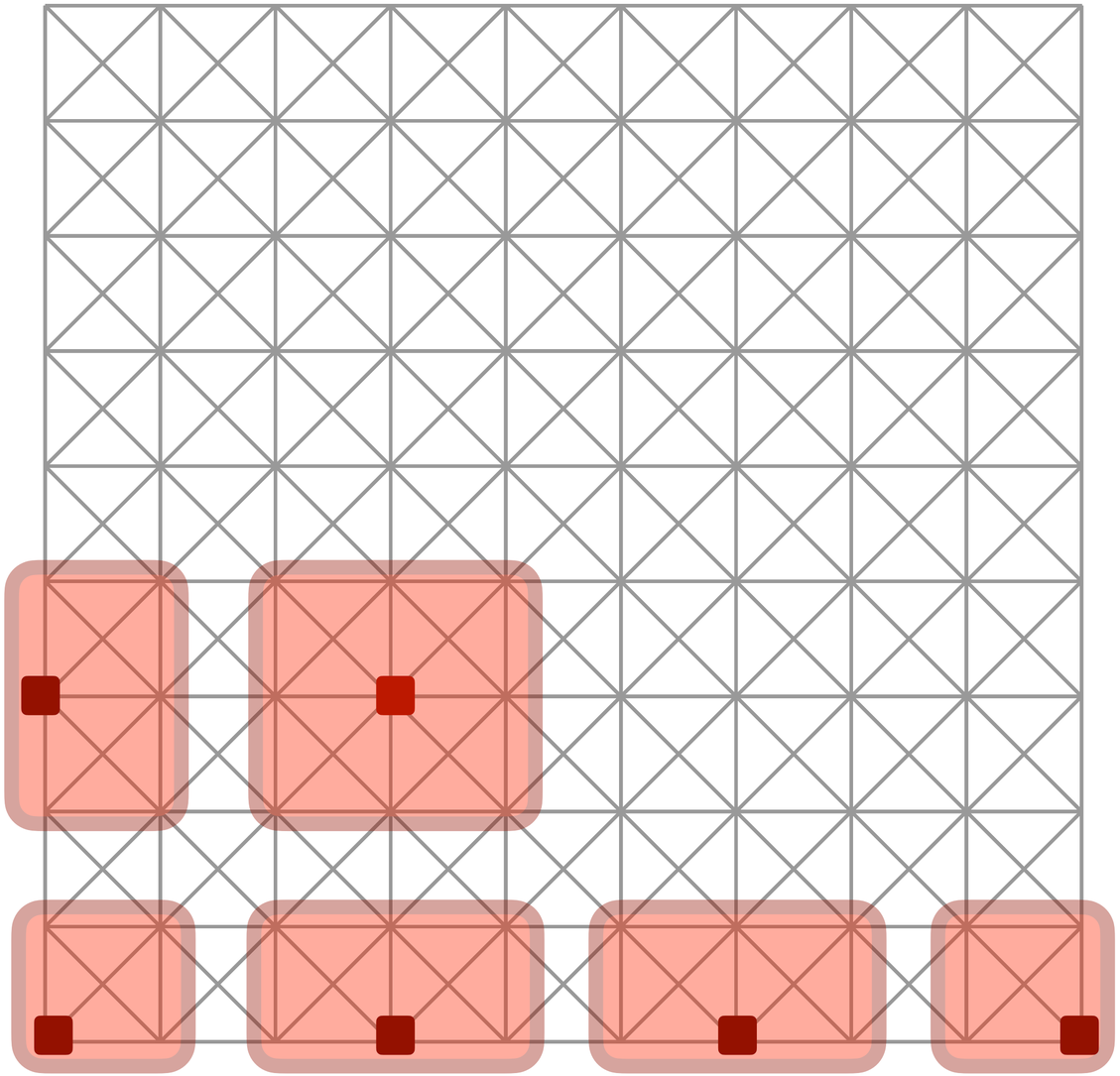
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



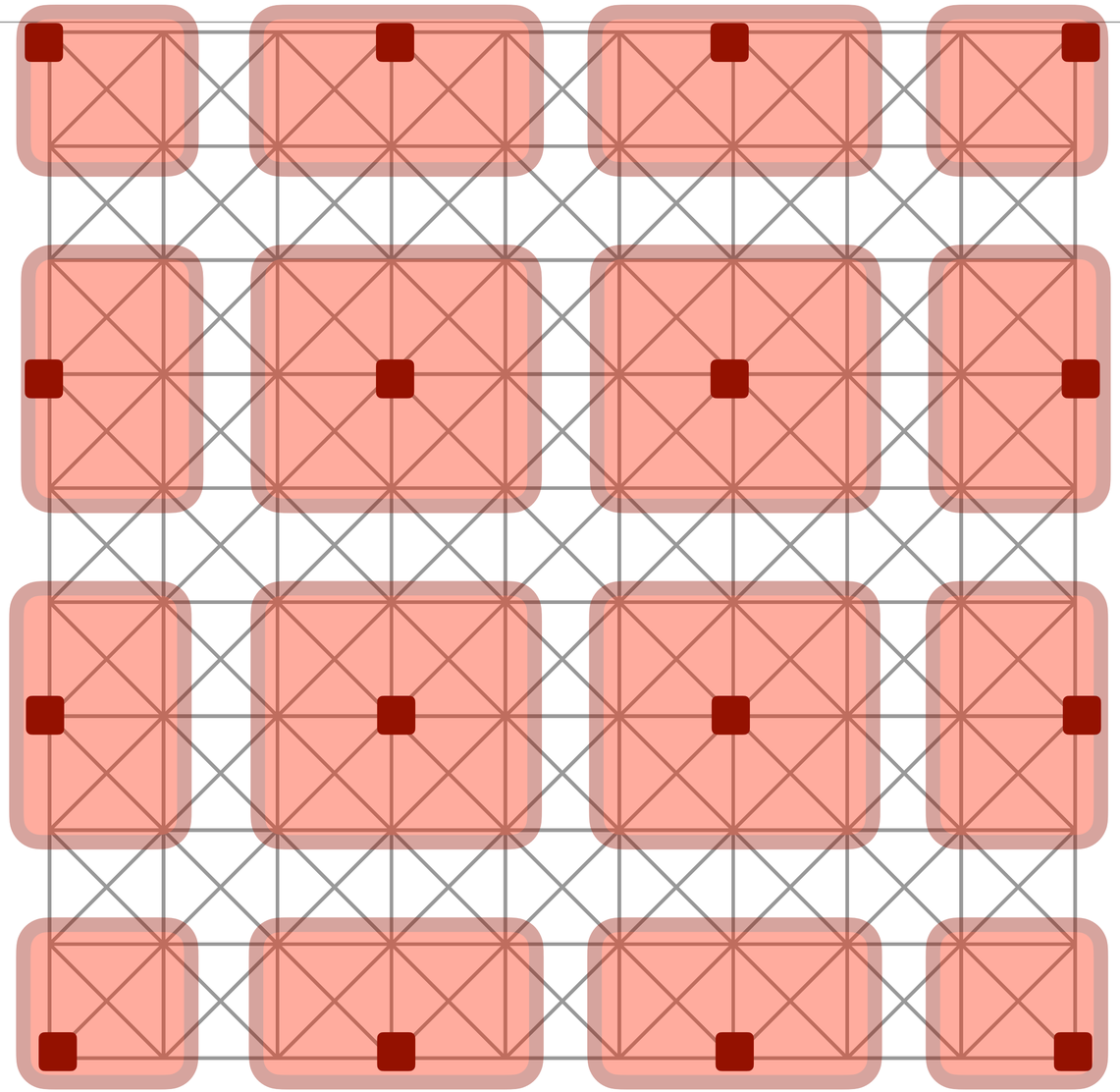
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

Aggregation



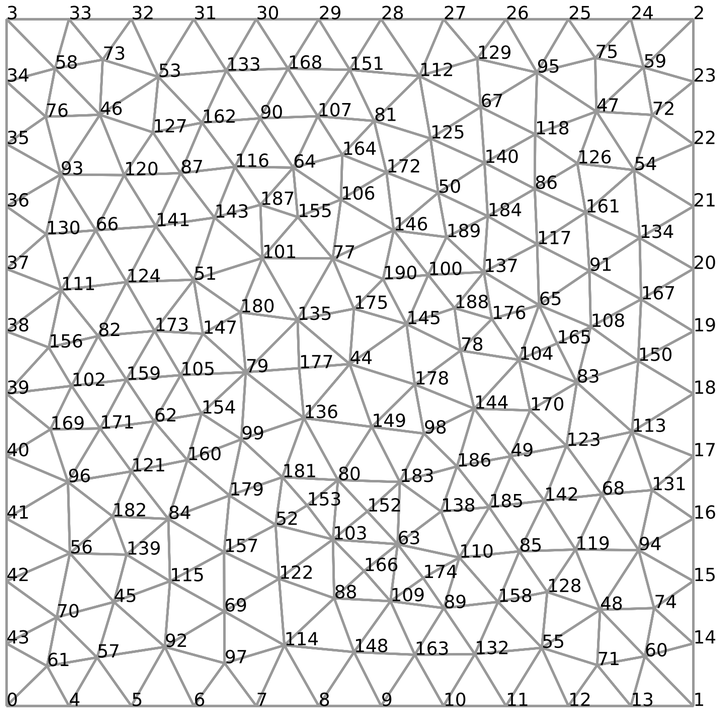
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.
-

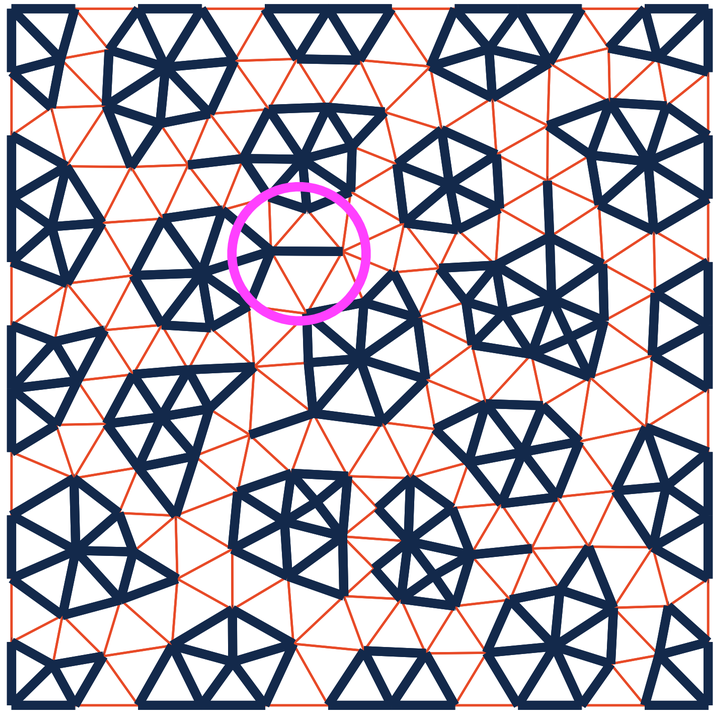
Aggregation



- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

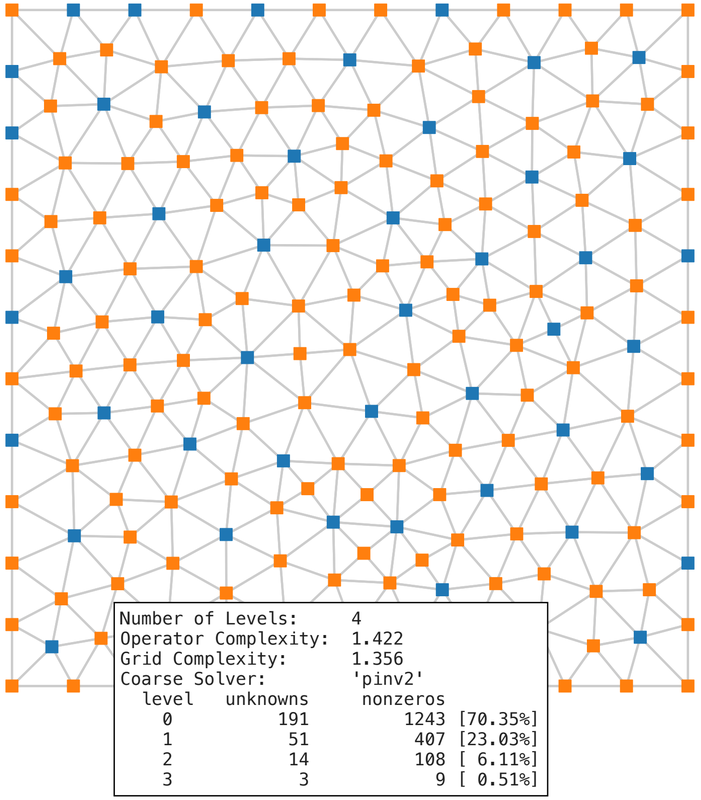
Aggregation



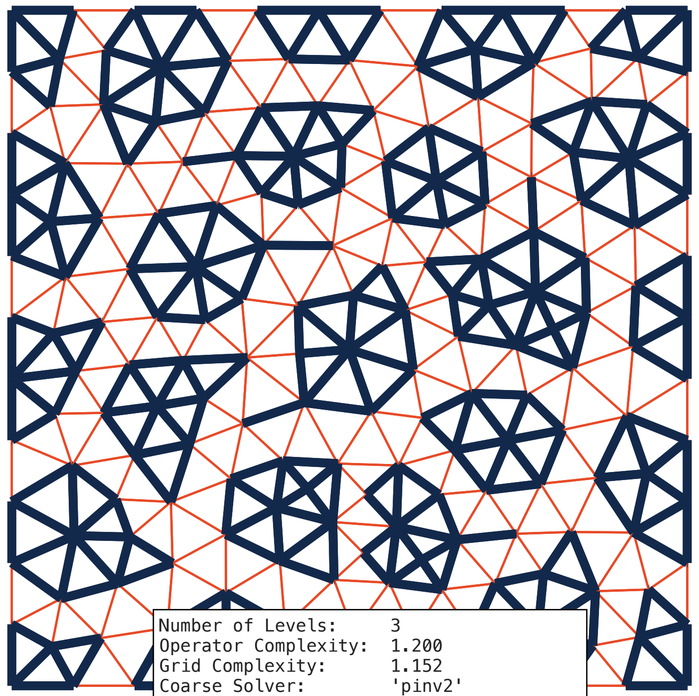


- Second passes (often) needed
- Options:
  - group with largest
  - regroup
  - group with strongest

Aggregation vs splitting



```
Number of Levels:     4
Operator Complexity:  1.422
Grid Complexity:      1.356
Coarse Solver:        'pinv2'
  level   unknowns     nonzeros
    0          191         1243 [70.35%]
    1           51          407 [23.03%]
    2           14          108 [ 6.11%]
    3            3            9 [ 0.51%]
```

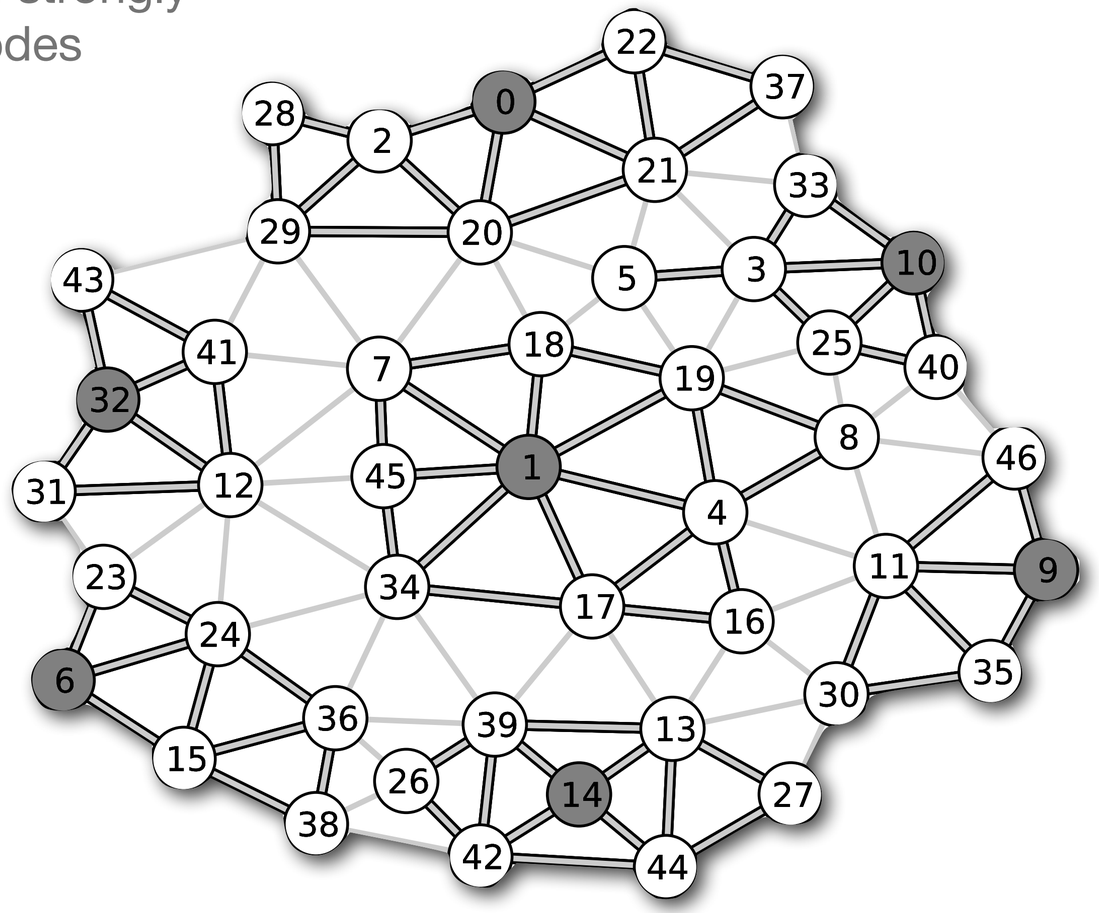


```
Number of Levels:     3
Operator Complexity:  1.200
Grid Complexity:      1.152
Coarse Solver:        'pinv2'
  level   unknowns     nonzeros
    0          191         1243 [83.31%]
    1           25          233 [15.62%]
    2            4           16 [ 1.07%]
```

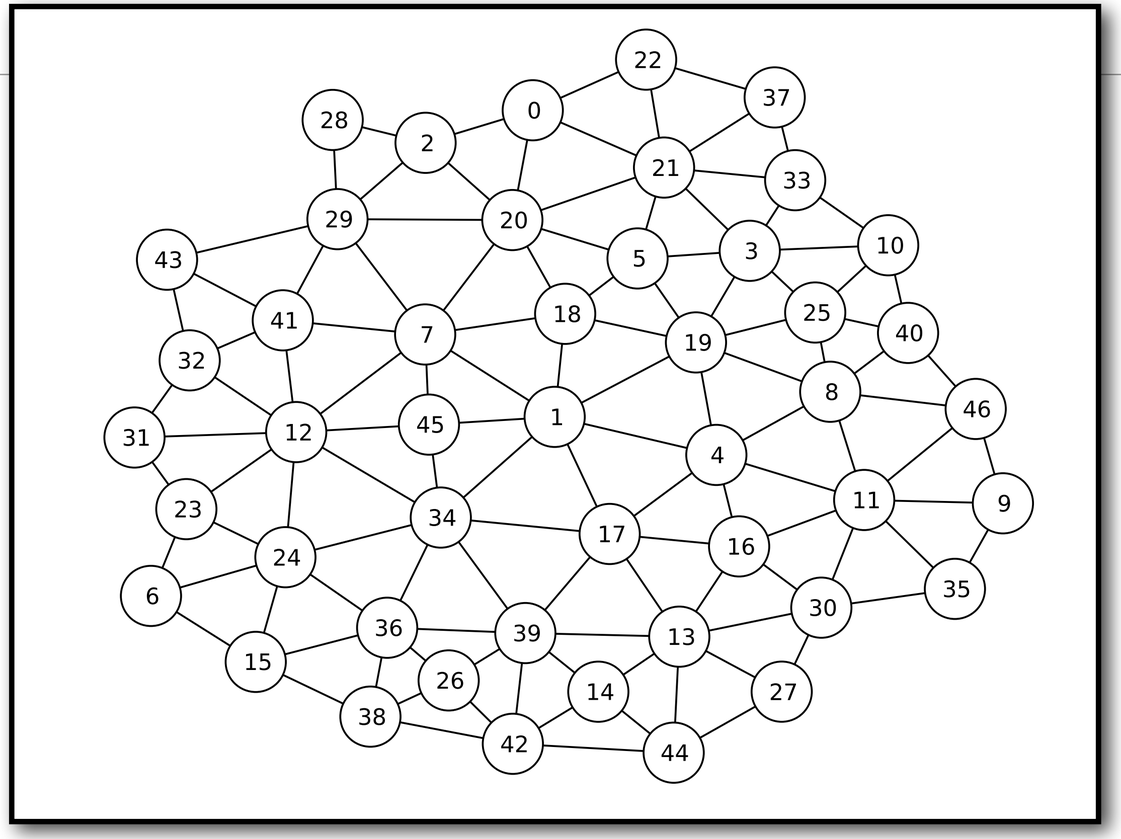
- Aggregation generally more aggressive coarsening

MIS-based Graph Aggregation

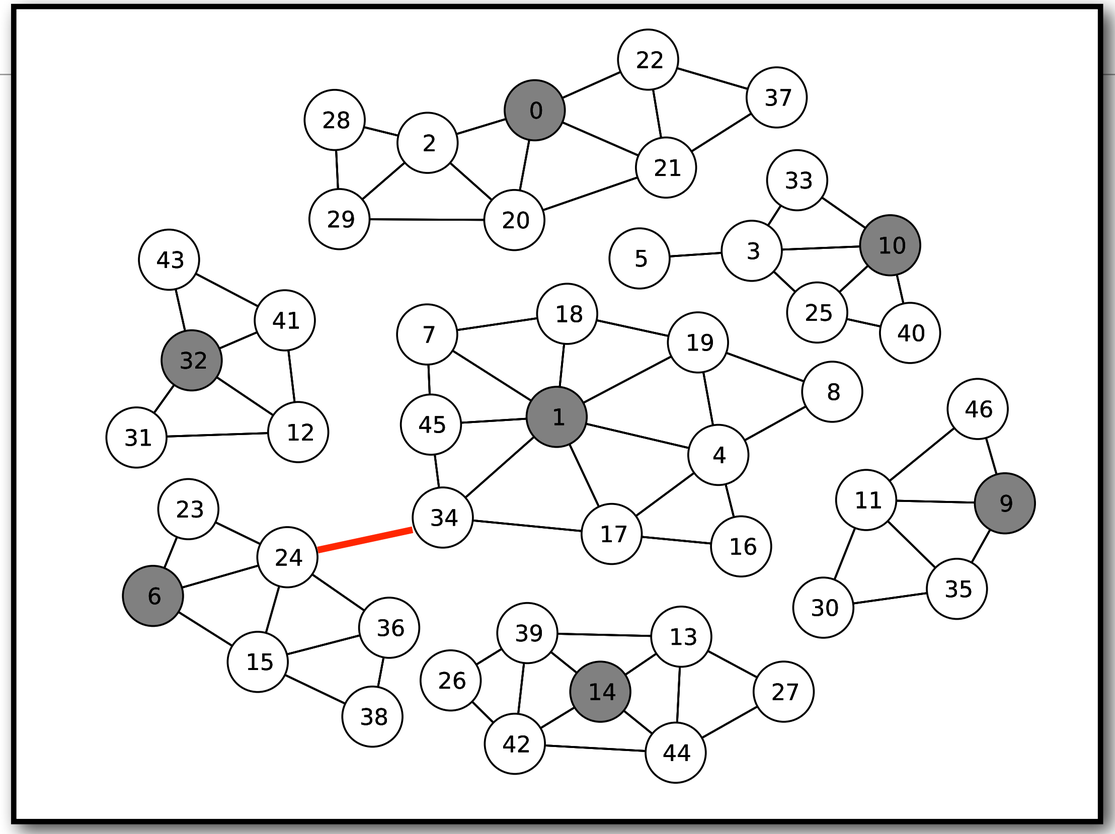
- Greedy: group collections of strongly connected unaggregated nodes
- Problems:
  1. size fixed
  2. sequential (greedy)



MIS(2)


22
37
0
28
2
21
33
29
20
43
5
3
10
41
18
25
40
7
19
32
8
46
31
12
45
1
4
23
11
9
34
17
16
24
6
35
30
36
39
13
15
26
14
27
38
42
44

MIS(2)



- root nodes more than 2 edges apart (> distance-2)
- an unaggregated node more than 2 edges from a root can become a root

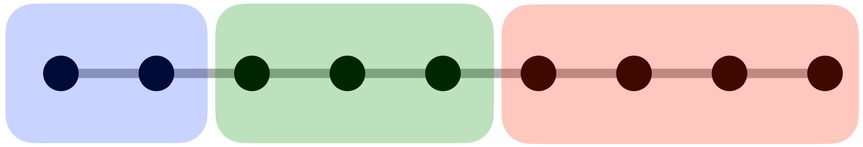
independent

MIS(2)

maximal

SA AMG Interpolation

- Here we use 1) the aggregation pattern and 2) the candidate vectors



$$AggOpp = \begin{bmatrix} \star & 0 & 0 \\ \star & 0 & 0 \\ 0 & \star & 0 \\ 0 & \star & 0 \\ 0 & \star & 0 \\ 0 & 0 & \star \\ 0 & 0 & \star \\ 0 & 0 & \star \\ 0 & 0 & \star \end{bmatrix}$$

$$B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \\ B_{31} & B_{32} \\ B_{41} & B_{42} \\ B_{51} & B_{52} \\ B_{61} & B_{62} \\ B_{71} & B_{72} \\ B_{81} & B_{82} \\ B_{91} & B_{92} \end{bmatrix} \rightarrow \hat{T} = \begin{bmatrix} B_{11} & B_{12} & & \\ B_{21} & B_{22} & & \\ & & B_{31} & B_{32} \\ & & B_{41} & B_{42} \\ & & B_{51} & B_{52} \\ & & & \\ & & & \\ & & & \\ & & & \\ & & B_{61} & B_{62} \\ & & B_{71} & B_{72} \\ & & B_{81} & B_{82} \\ & & B_{91} & B_{92} \end{bmatrix}$$

$Q_1 R_1$ $Q_2 R_2$ $Q_3 R_3$

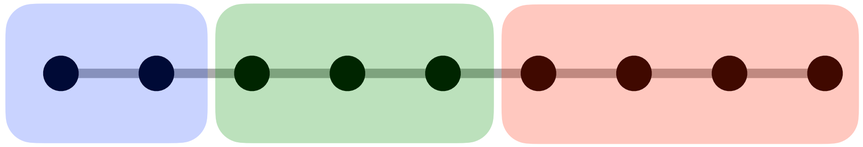
$$T = \begin{bmatrix} Q1 & & \\ & Q2 & \\ & & Q3 \end{bmatrix}$$

$$B_C = \begin{bmatrix} R_1 \\ R_2 \\ R_3 \end{bmatrix}$$

$$TB_C = B$$

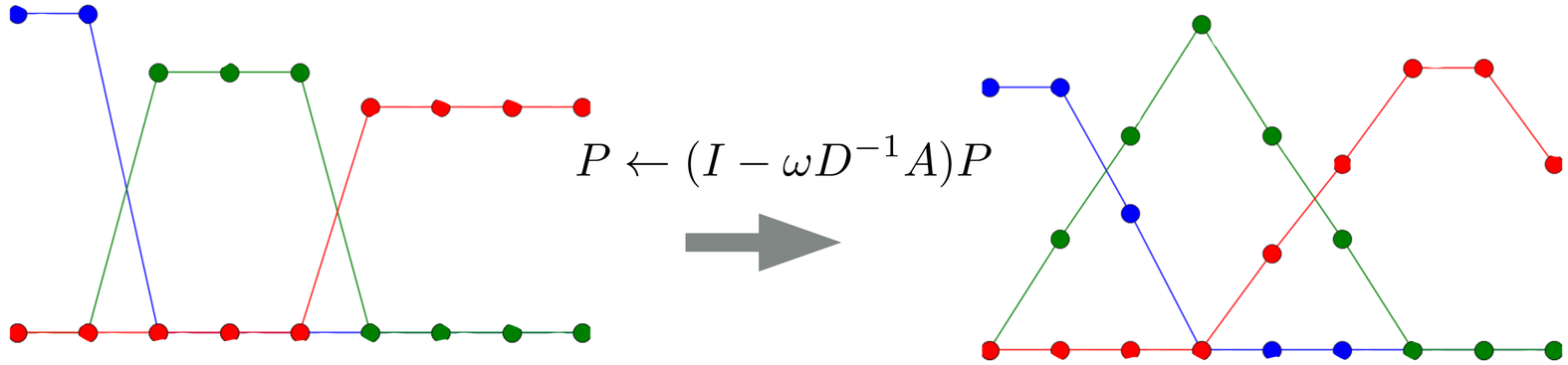
SA AMG Interpolation

- Example



$$\hat{T} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 1 \end{bmatrix} \qquad T = \begin{bmatrix} \frac{\sqrt{2}}{2} & 0 & 0 \\ \frac{\sqrt{2}}{2} & 0 & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 \\ 0 & \frac{\sqrt{3}}{3} & 0 \\ 0 & 0 & \frac{\sqrt{4}}{4} \\ 0 & 0 & \frac{\sqrt{4}}{4} \\ 0 & 0 & \frac{\sqrt{4}}{4} \\ 0 & 0 & \frac{\sqrt{4}}{4} \end{bmatrix}$$

- Now make interpolation **better**

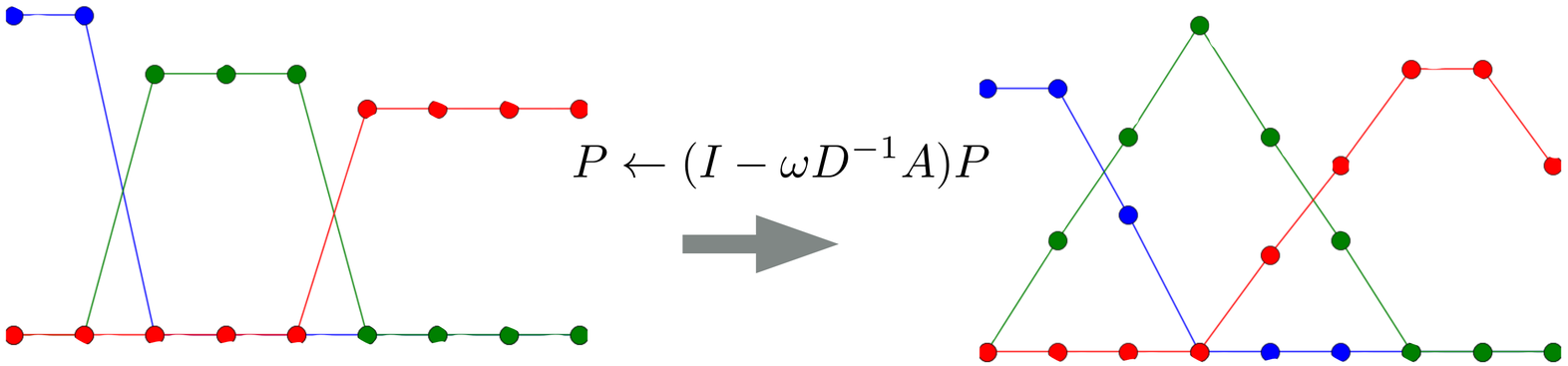


$$P \leftarrow (I - \omega D^{-1} A) P$$

SA AMG Interpolation

- reduce energy
- improve accuracy
- increase complexity

- Improving interpolation



- Makes the columns of P ***smoother***

- Makes the sparsity of P **denser**

Demo: 17-smoothed-aggregation-1d.ipynb

Aggregation Based Setup

B

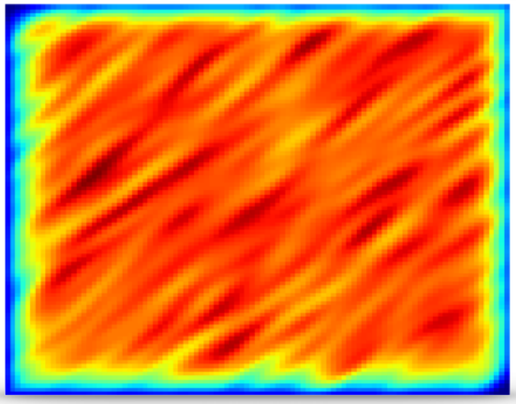
$S \leftarrow \text{strength}(A, B)$

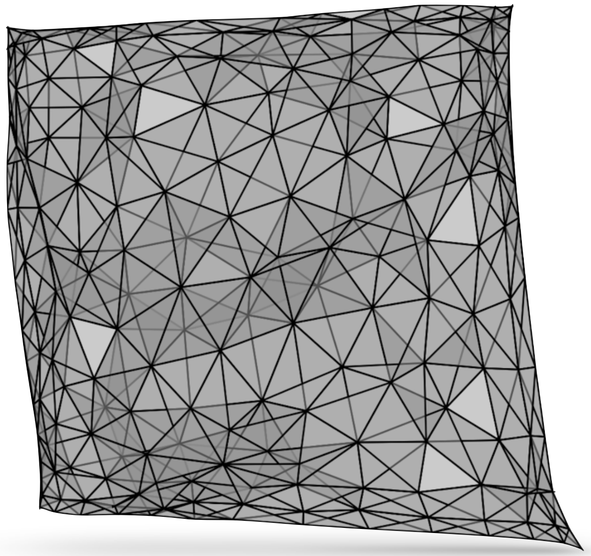
$C \leftarrow \text{aggregate}(S)$

$P^{(0)}, B^C \leftarrow \text{inject}(C, B)$

$P \leftarrow \text{improve}(A, P^{(0)})$

$R = P^* \qquad A^C = RAP$





Aggregation Based Setup

$$B$$

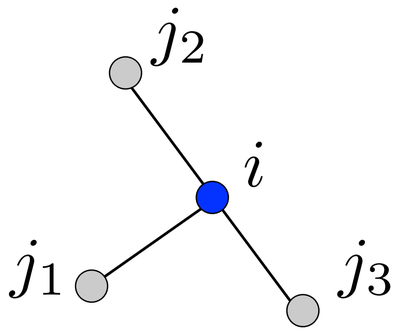
$$S \leftarrow \text{strength}(A, B)$$

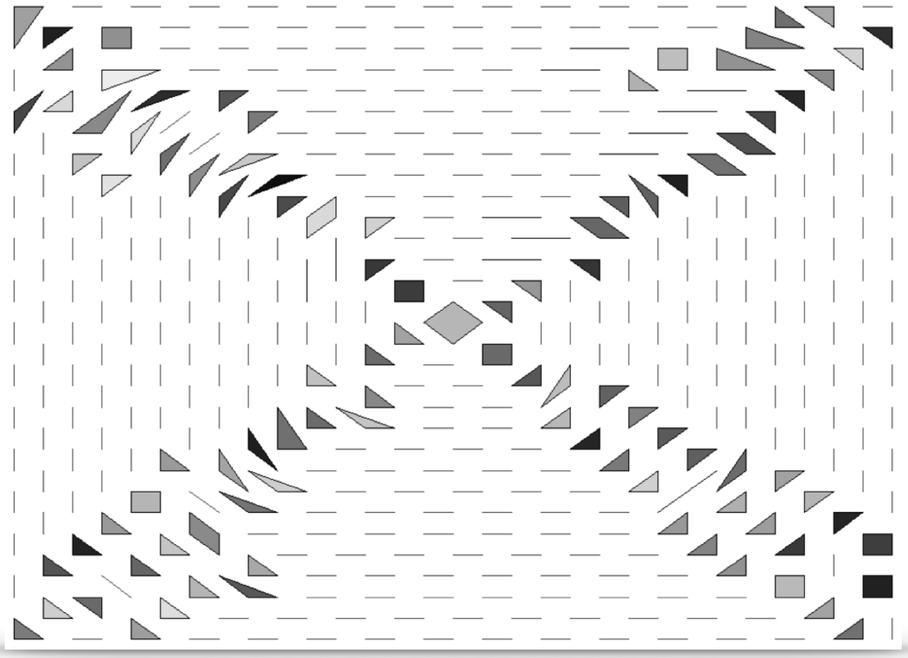
$$C \leftarrow \text{aggregate}(S)$$

$$P^{(0)}, B^C \leftarrow \text{inject}(C, B)$$

$$P \leftarrow \text{improve}(A, P^{(0)})$$

$$R = P^* \qquad A^C = RAP$$





Aggregation Based Setup

$$B$$

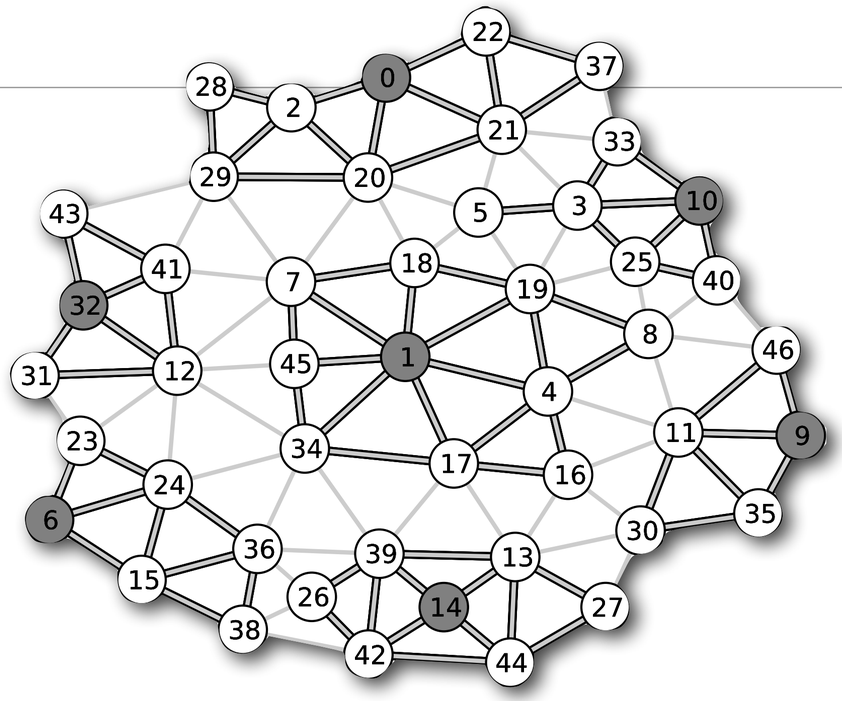
$$S \leftarrow \text{strength}(A, B)$$

$$C \leftarrow \text{aggregate}(S)$$

$$P^{(0)}, B^C \leftarrow \text{inject}(C, B)$$

$$P \leftarrow \text{improve}(A, P^{(0)})$$

$$R = P^* \qquad A^C = RAP$$



- Group nodes
- based on strong connections
- variable size

Aggregation Based Setup

B

$S \leftarrow \text{strength}(A, B)$

$C \leftarrow \text{aggregate}(S)$

$P^{(0)}, B^C \leftarrow \text{inject}(C, B)$

$P \leftarrow \text{improve}(A, P^{(0)})$

$R = P^* \qquad A^C = RAP$

$$B = \begin{bmatrix} b_{00} & b_{01} \\ b_{10} & b_{11} \\ b_{20} & b_{21} \\ b_{30} & b_{31} \\ b_{40} & b_{41} \\ b_{50} & b_{51} \\ b_{60} & b_{61} \\ b_{70} & b_{71} \\ b_{80} & b_{81} \\ b_{90} & b_{91} \end{bmatrix}$$

$$P_{tent} = \begin{bmatrix} b_{00} & b_{01} & & & & & & \\ b_{10} & b_{11} & & & & & & \\ & & b_{20} & b_{21} & & & & \\ & & b_{30} & b_{31} & & & & \\ & & b_{40} & b_{41} & & & & \\ & & & & b_{50} & b_{51} & & \\ & & & & b_{60} & b_{61} & & \\ & & & & b_{70} & b_{71} & & \\ & & & & & & b_{80} & b_{81} \\ & & & & & & b_{90} & b_{91} \end{bmatrix}$$

Aggregation Based Setup

$$B$$

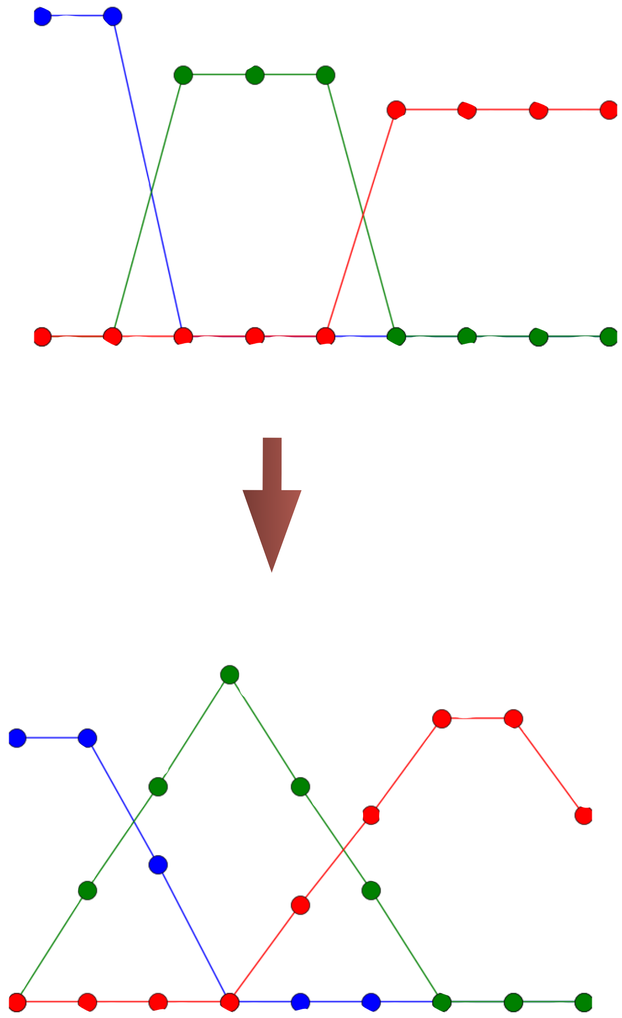
$$S \leftarrow \text{strength}(A, B)$$

$$C \leftarrow \text{aggregate}(S)$$

$$P^{(0)}, B^C \leftarrow \text{inject}(C, B)$$

$$P \leftarrow \text{improve}(A, P^{(0)})$$

$$R = P^* \qquad A^C = RAP$$



SA AMG Setup Algorithm

Algorithm 1: `SA_setup()`

Input: A_0: fine-grid operator
B_0: fine-grid candidate vectors
`max_size`: threshold for max size of coarsest problem

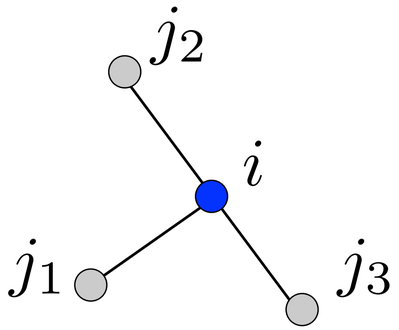
Output: $A_1, \ldots, A_L$,
$P_0, \ldots, P_{L-1}$

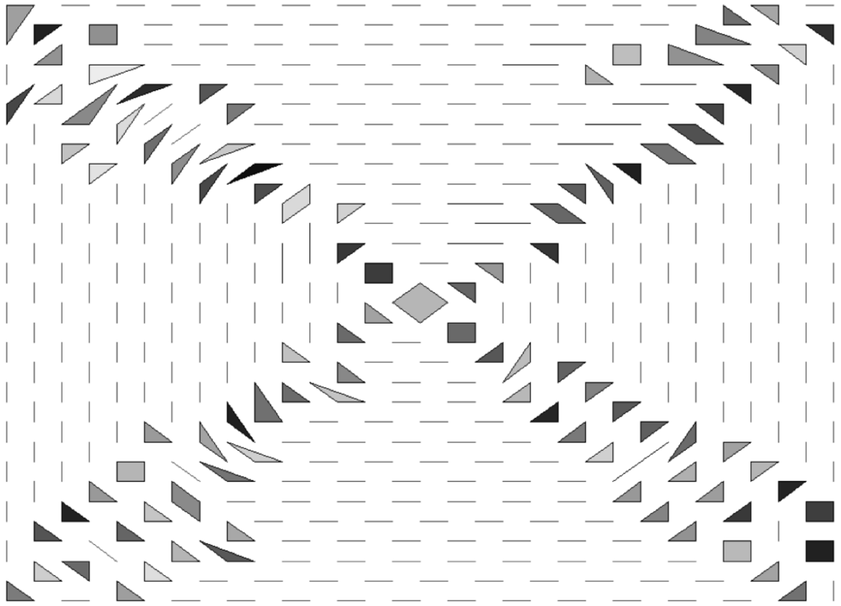
$\ell = 0$
while $\texttt{size}(A_\ell) > \texttt{max\_size}$
  $S_\ell = \texttt{strength}(A_\ell)$ {Strength-of-connection}
  $\mathcal{A}_\ell = \texttt{aggregate}(S_\ell)$ {Aggregation}
  $T_\ell,\, B_{\ell+1} = \texttt{inject}(\mathcal{A}_\ell,\, B_\ell)$ {Form tentative interpolation and coarse candidates}
  $P_\ell = \texttt{smooth}(A_\ell,\, T_\ell)$ {Smooth T_ℓ}
  $A_{\ell+1} = P_\ell^T A_\ell P_\ell$ {Coarse-grid operator}
  $\ell = \ell + 1$

Demo: 18-components-of-smoothed-aggregation.ipynb

Advanced feature: evolution measure

1. drop point source at a node
2. evolve point source with A
3. evaluate diffusivity at neighbors in comparison to known low energy



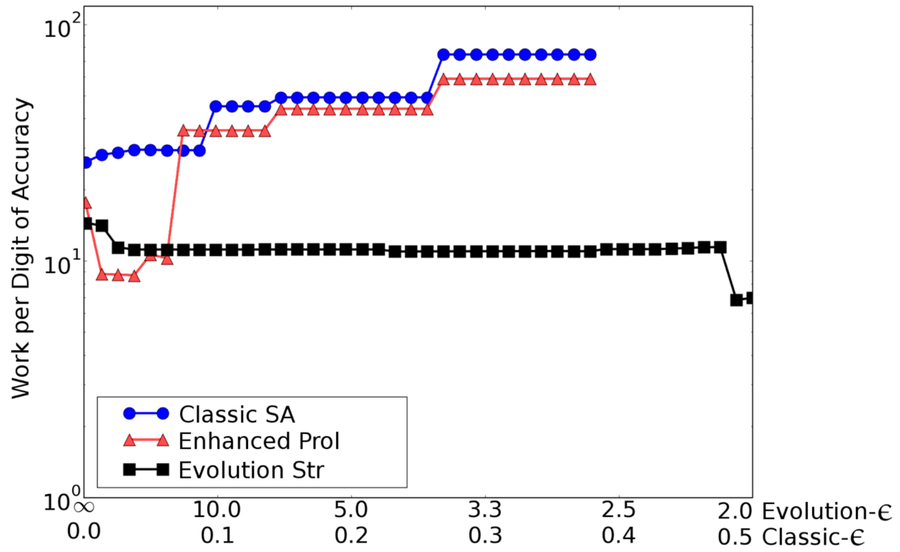


- efficient
- parameter insensitive
- Euler flow
- wave problems
- high-order
- discontinuous elements

Luke N. Olson , Jacob Schroder , Raymond S. Tuminaro, A new perspective on strength measures in algebraic multigrid, NLAA, 2010

Advanced feature: evolution measure

- less sensitive to drop tolerances
- single largest improvement we see in practice
- necessary for any complicated physics or discretization



ϵ		Classic SA 0	Classic SA $\frac{1}{4}$	with P_{opt} 0	with P_{opt} $\frac{1}{4}$	Evol 4.00
Linear Tets						
$p=1$	h	18	48	15	27	15
	$h/2$	17	74	14	62	14
	$h/4$	26	130	20	92	17
$p=2$	h	41	85	24	46	18
Quad Tets						
$p=1$	h	21	44	26	33	16
	$h/2$	45	88	67	67	14
	$h/4$	63	148	129	129	17
$p=2$	h	49	86	61	61	17

SA AMG Interpolation

$$e_1 \leftarrow (I - P(P^T A P)^{-1} P^T A) G e_0$$

coarse grid correction | relax

$$Ge_0 \in \mathcal{R}(P) \quad \Rightarrow \quad e_1 = 0$$

interpolation should capture what relaxation misses

- P should have low energy (low A-norm or A^*A-norm)
  1. determine sparsity pattern
  2. minimize energy column-wise (parallel)

SA AMG Interpolation

- Want P so that $u_{low} \in \mathcal{R}(P)$

1. Grow and fix sparsity pattern as $S^k P_{tent}$

2. Minimize residual of

$$AP_j = 0 \quad \text{for each column } j$$

3. Constraint the minimization with

$$PB_c = B$$

SA AMG General Interpolation

- Hermitian (and positive definite): use CG

$$AP_j = 0 \Leftrightarrow \min \|P_j\|_A$$

$$R = P^*$$

- Non-Hermitian: use GMRES

$$AP_j = 0 \Leftrightarrow \min \|P_j\|_{A^*A}$$

$$A^*R_j^* = 0 \Leftrightarrow \min \|R_j^*\|_{AA^*}$$

- Range of interpolation targets “right” low-energy
- Range of restriction* targets “left” low-energy
- Cost is comparable to that of standard smoothing

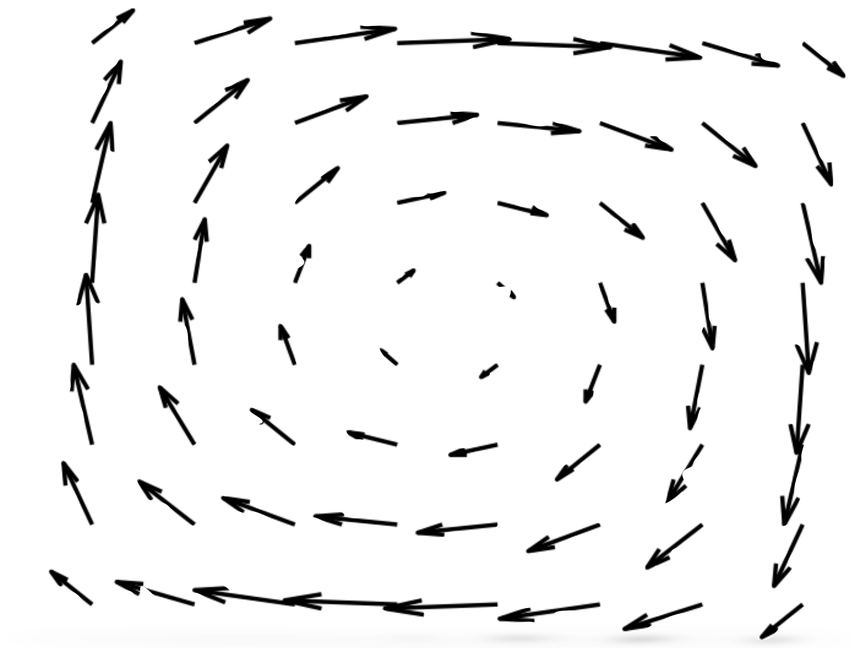
W. L. Wan, T. F. Chan, and B. Smith, An energy-minimizing interpolation for robust multi-grid methods, SIAM J. Sci. Comput., 2000

J. Xu and L. Zikatanov, On an energy minimizing basis for algebraic multigrid methods, Comput. Vis. Sci., 2004

Luke N. Olson , Jacob Schroder , Raymond S. Tuminaro, A General Interpolation Strategy for Algebraic Multigrid Using Energy Minimization, SISC, 2011

SA AMG General Interpolation

- P should have low energy
  (low A-norm or A^*A-norm)
  1. determine sparsity pattern
  2. minimize energy column-wise (parallel)



h	std.	opt.
1/64	>150	24
1/128	>150	28
1/256	>150	33
1/512	>150	33

Advanced Demos

Demo: `19-AMG-advanced-options-anisotropy.ipynb`

Demo: `20-AMG-advanced-options-nonsymmetric-flow.ipynb`

Demo: `21-AMG-advanced-options-systems-elasticity.ipynb`

More questions…

- Why does this work (at all)?
- What do you do if XYZ happens?
- How do you develop a new AMG method for a new problem?
- Where does AMG tend to work well and where does it not?