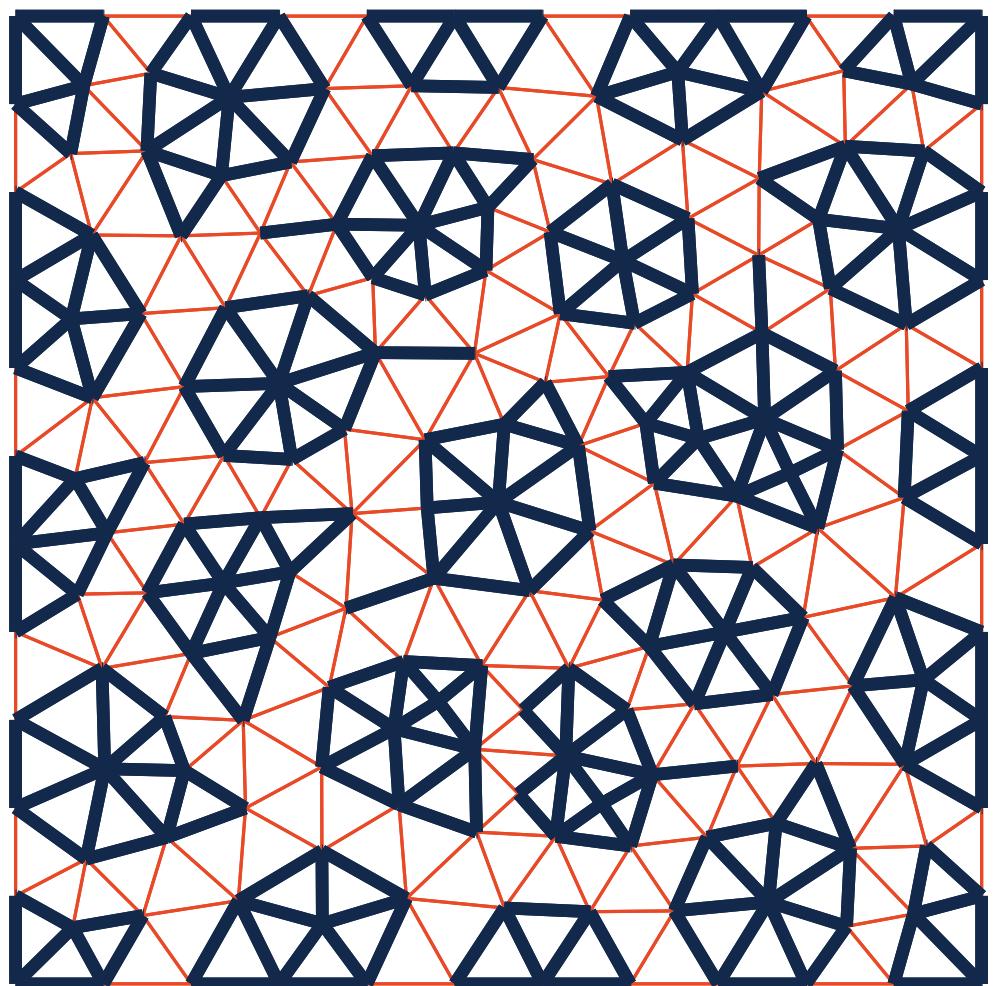


Algebraic Multigrid Tutorial

Scott MacLachlan
Department of Mathematics and Statistics
Memorial University of Newfoundland

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



Objectives – this tutorial

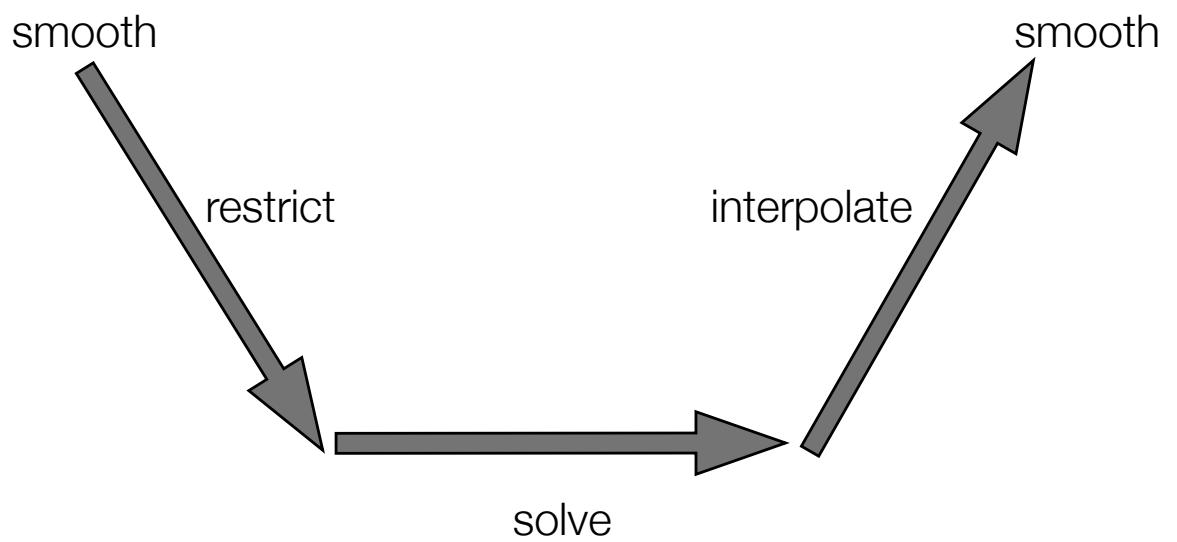
1. Algebraic multigrid

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

Algorithm: two-level multigrid

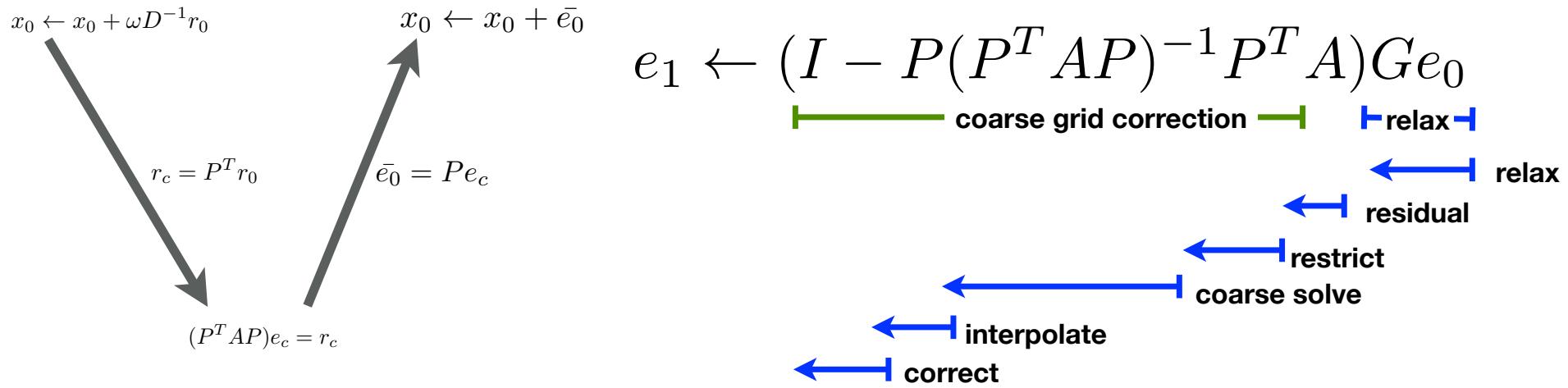
Input: initial guess

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



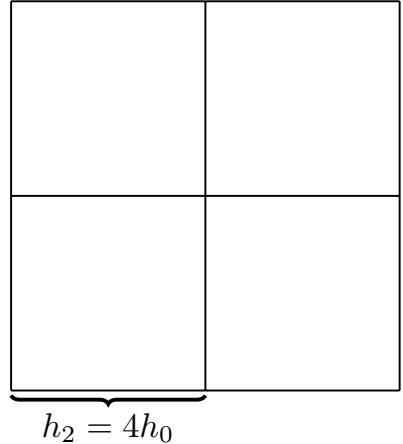
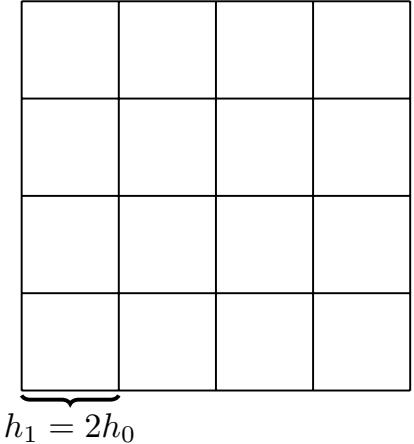
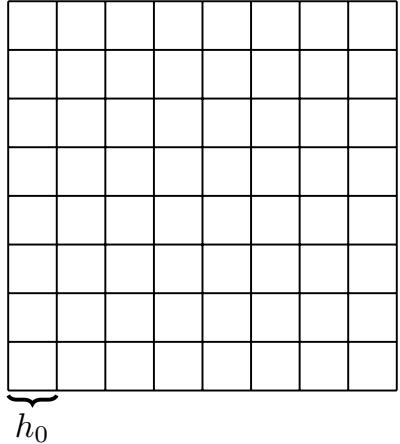
$$Ge_0 \in \mathcal{R}(P) \Rightarrow e_1 = 0$$

interpolation should capture what relaxation misses

Multigrid Basics

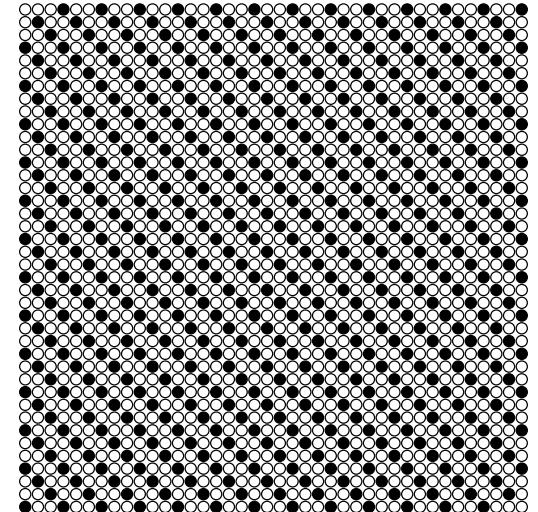
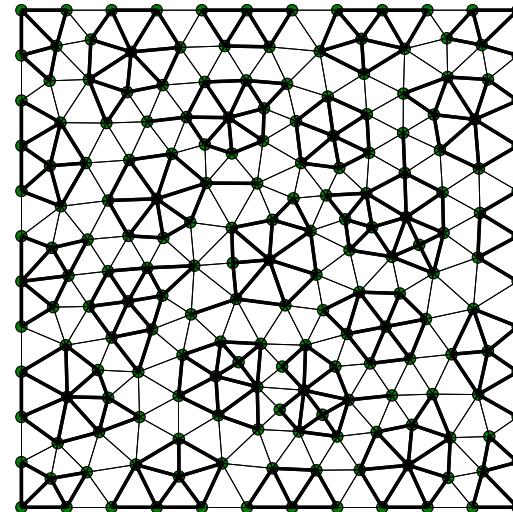
Geometric

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



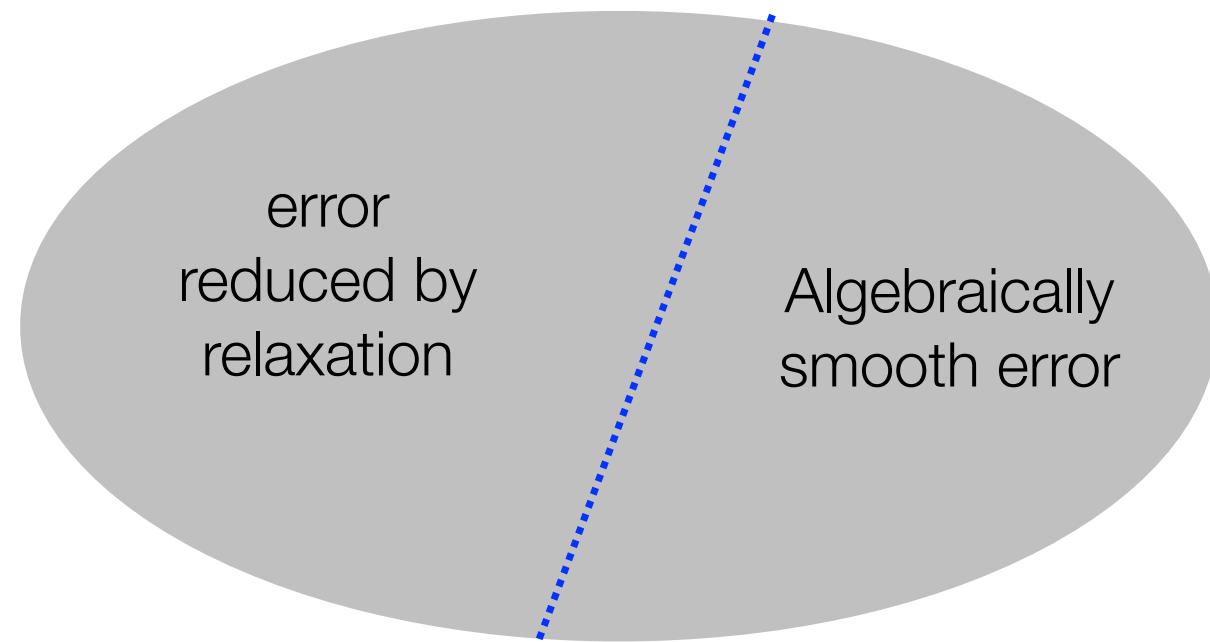
Algebraic

- 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

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

- Formally (characterized by small eigenvalues)

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

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

Main idea: Algebraically smooth error

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

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

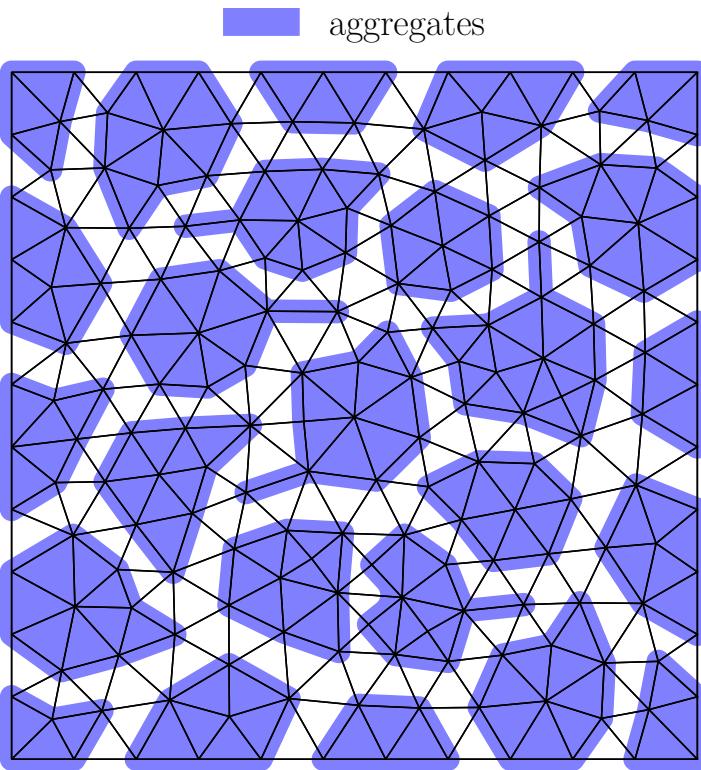
- **CF AMG:** Smooth error varies slowly in the direction of “large” matrix coefficients
- **Strength of connection:** Given a threshold $0 < \theta \leq 1$, we say that variable u_i strongly depends on variable u_j if

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

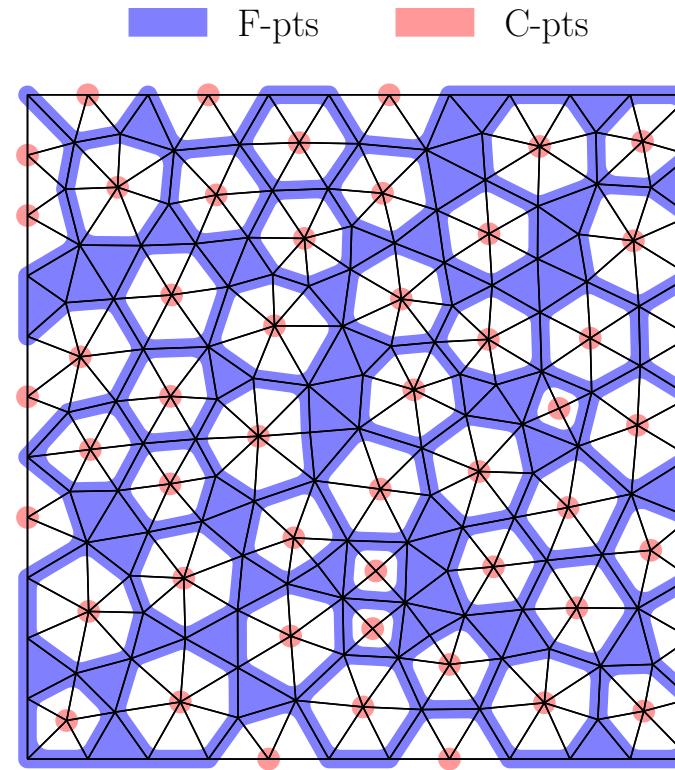
- Often positive off-diagonals are treated as **weak**
- This definition of strength of connection is not symmetric

Demo: 12-AMG-coarse-mesh.ipynb

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...

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

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

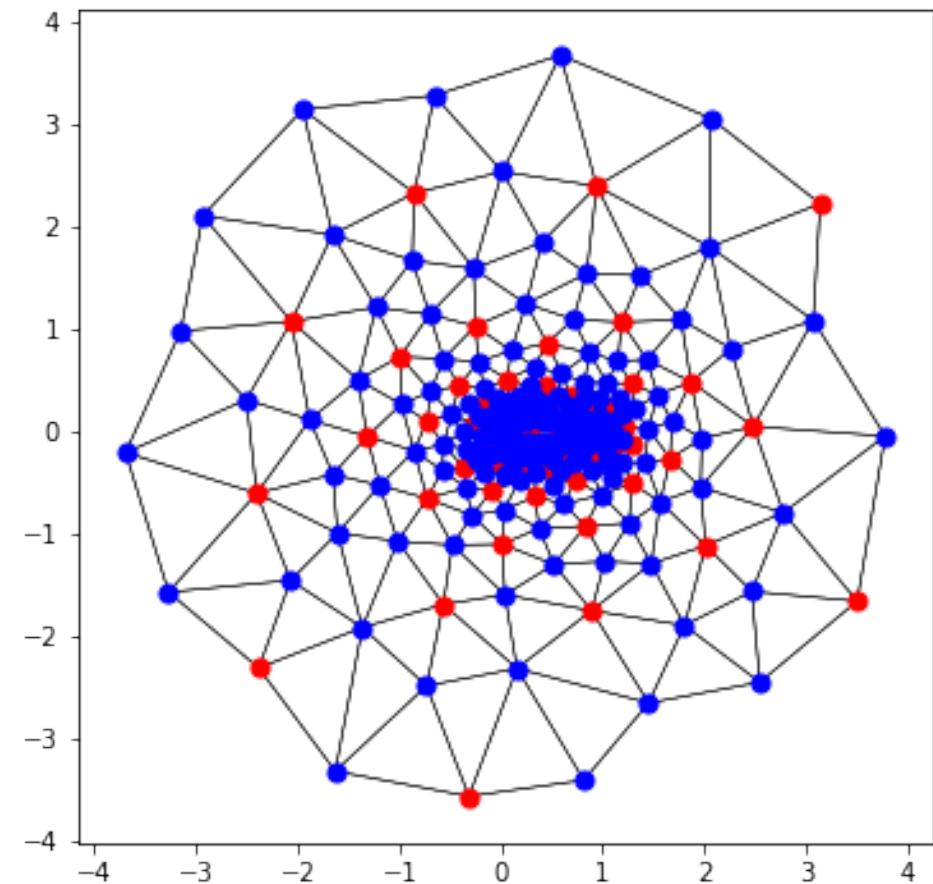
CF AMG

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

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

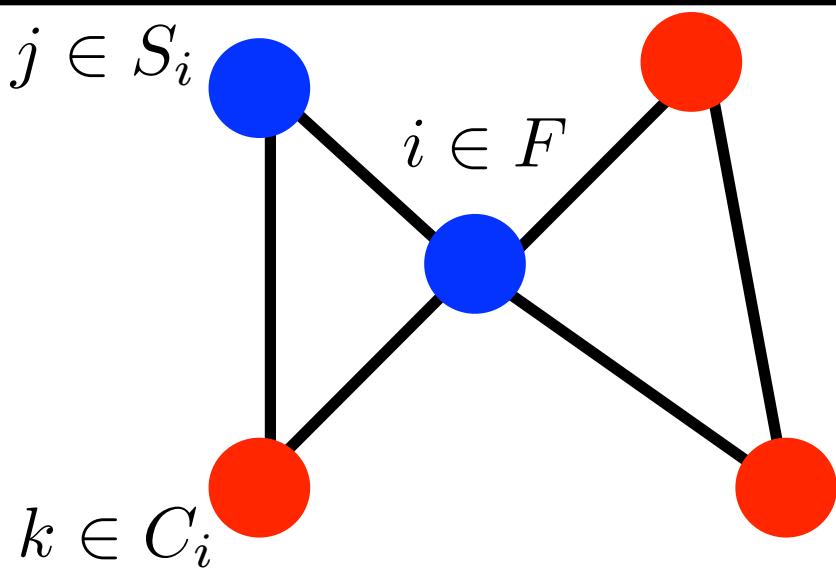
C-pts that are used to interpolated F-pt i.

$$C_i$$



CF AMG

- (C1) Every point that $i \in F$ strongly depends on is either
 - A point in C
 - A point that strongly depends on a point in C_i



CF AMG

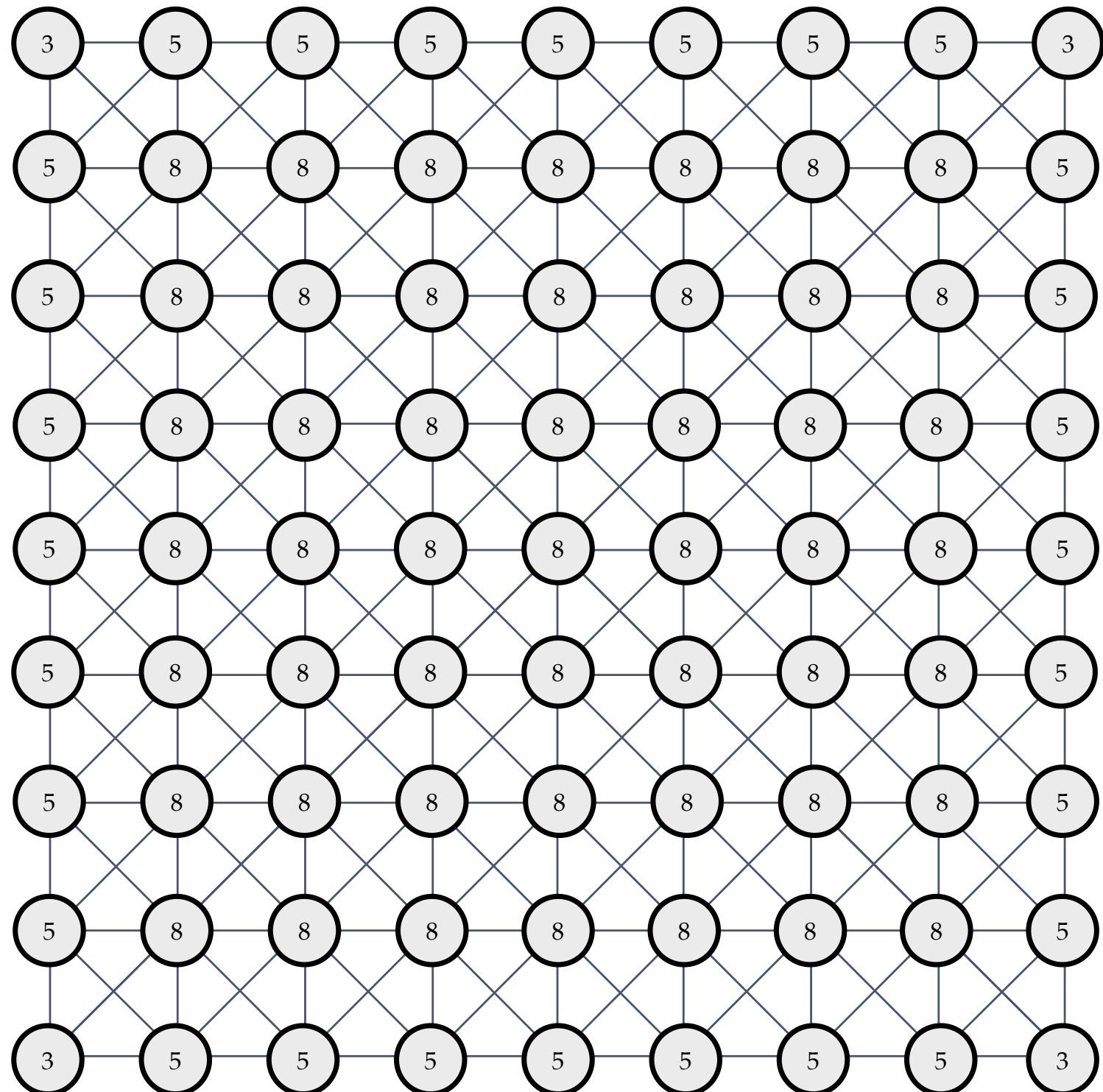
- (C2) The C-points should be *maximal* with no C-point strongly depending on another C-point
- (C1) increases the size of the coarse grid (C-points)
- (C2) puts constraints on the size of the coarse grid
- Must satisfy (C1) in order to construct interpolation. Use (C2) to help limit computational complexity

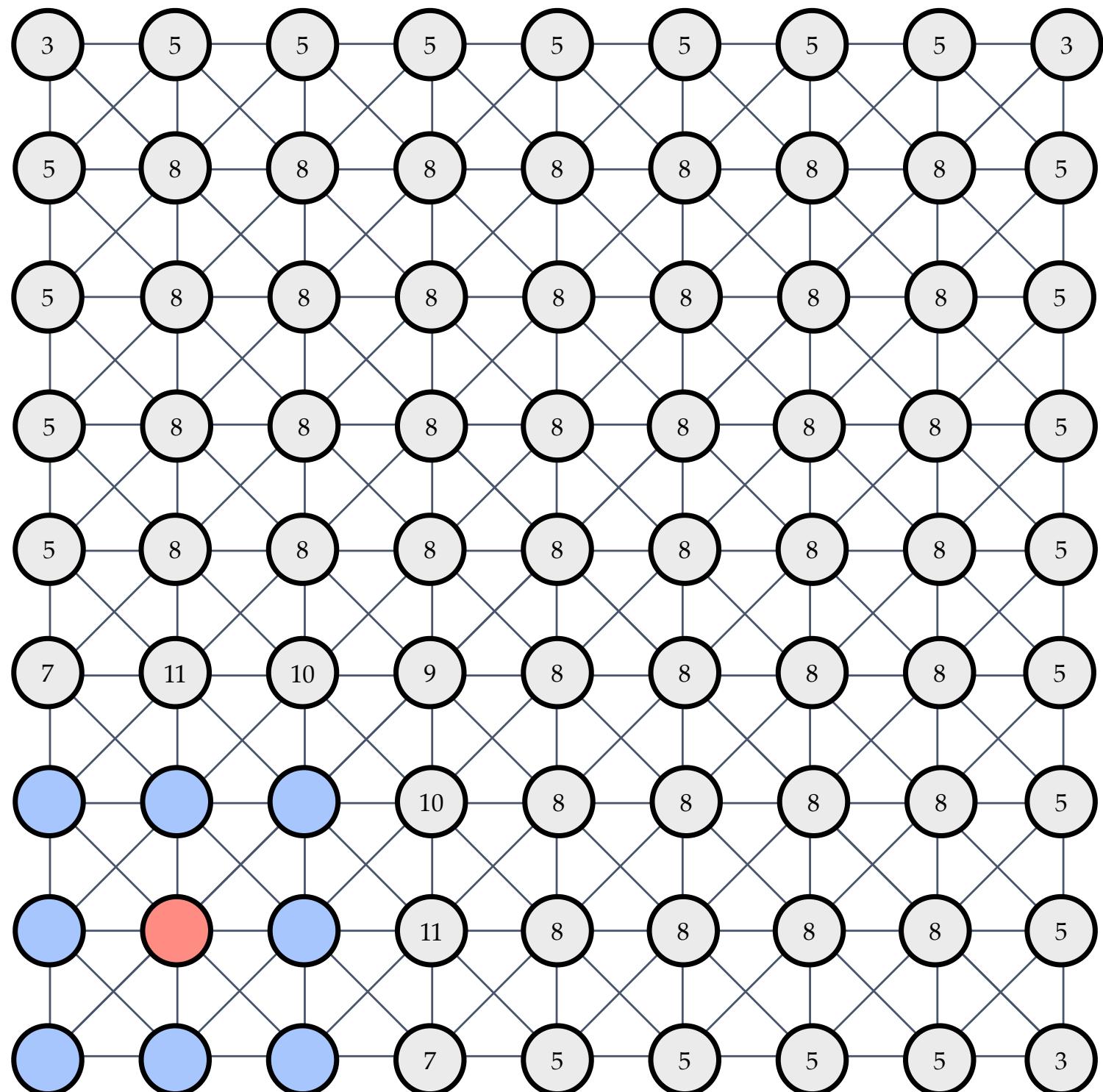
Ruge-Stüben

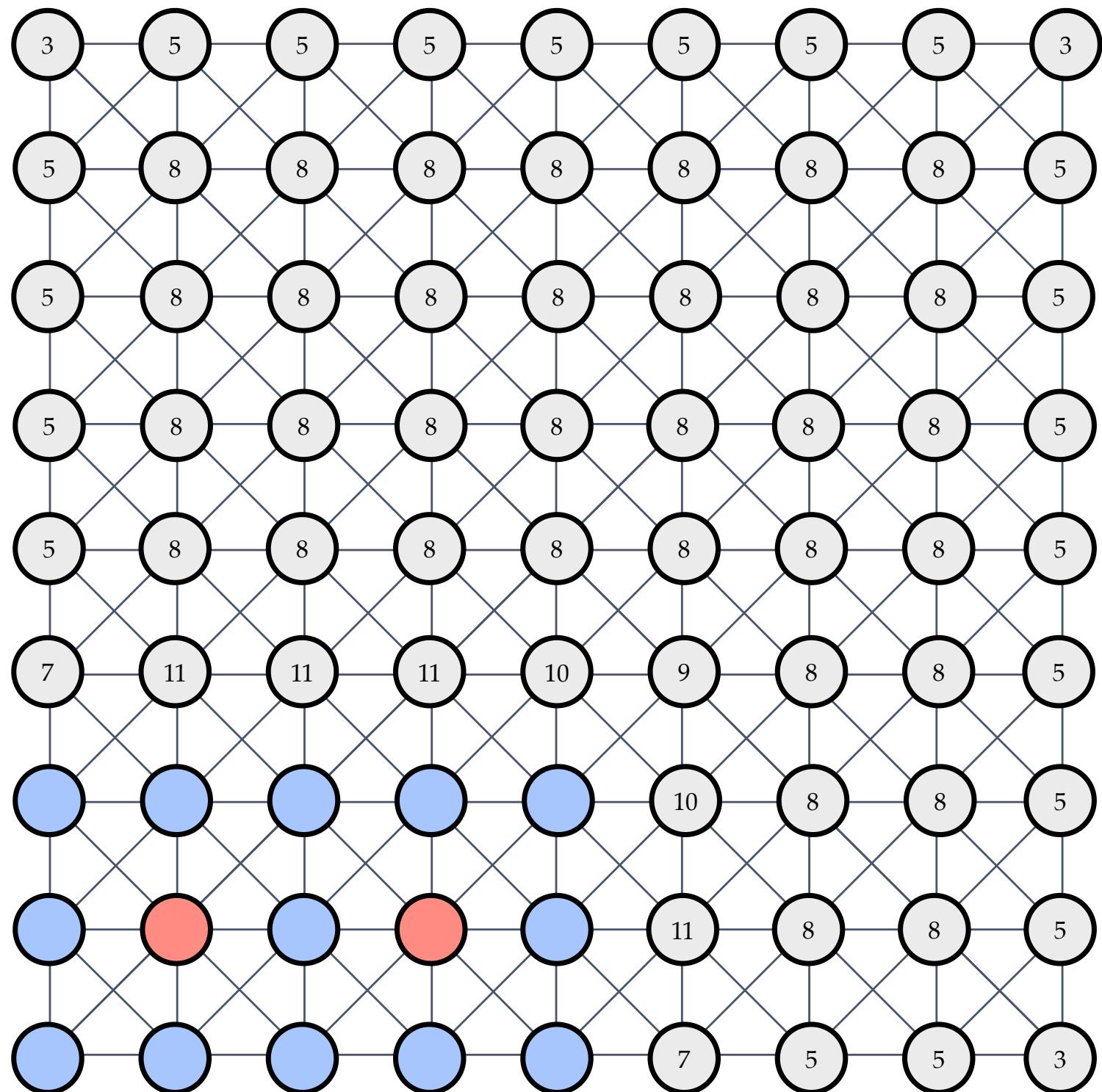
Algorithm 3. RUGE-STÜBEN.

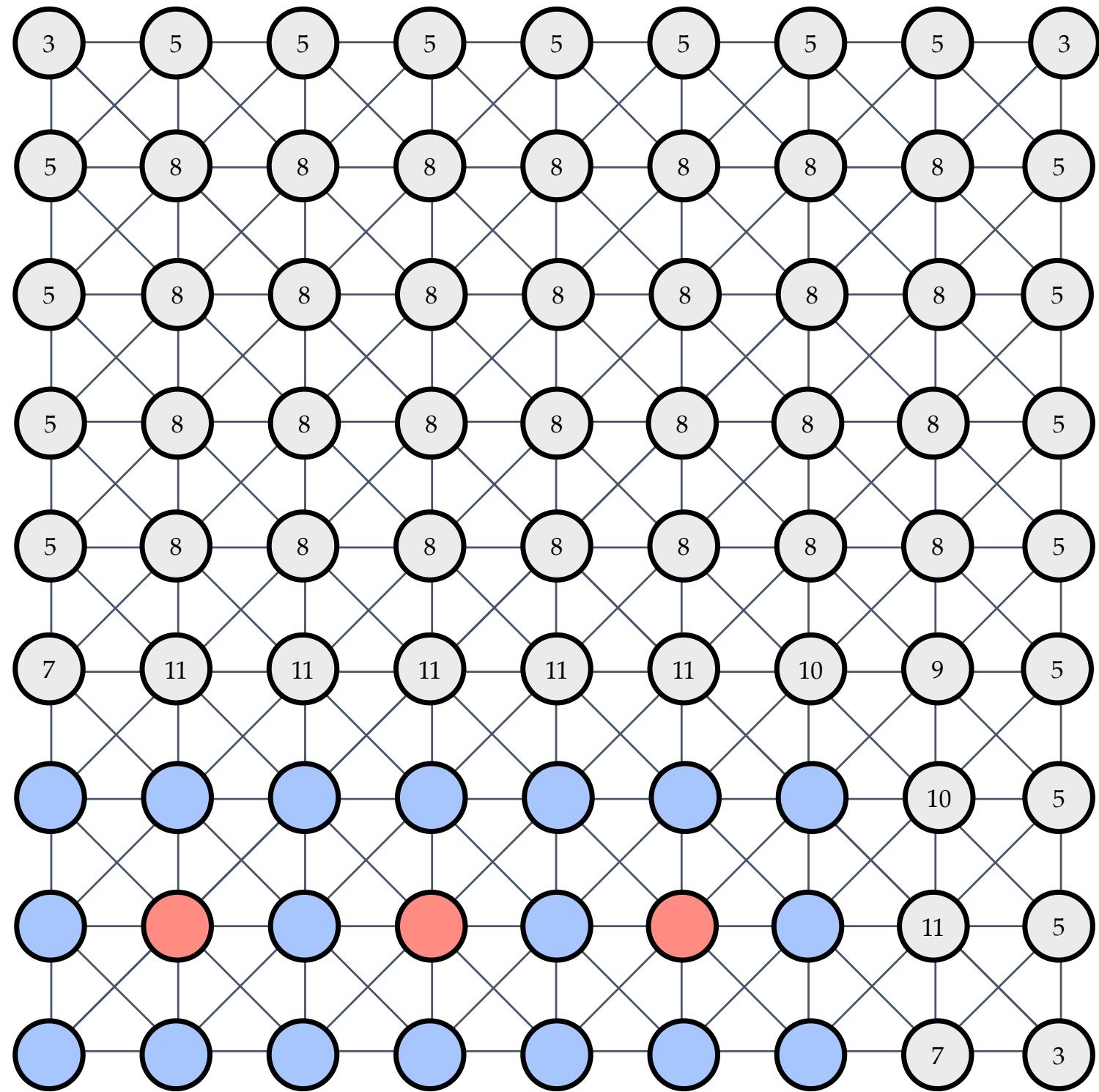
Initialize:

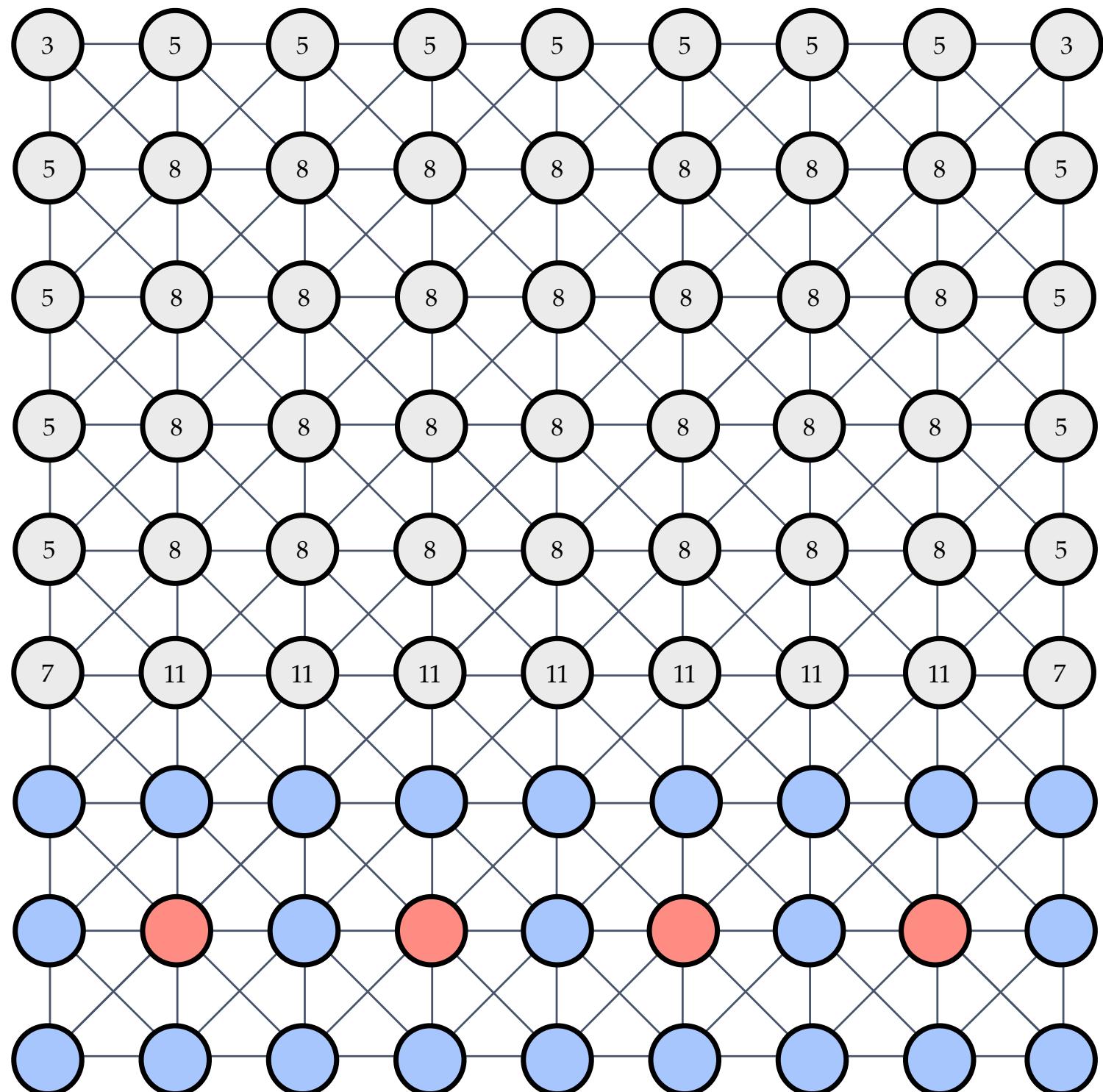
```
 $U = \Omega, C = \emptyset, F = \emptyset$ 
1. for all  $i \in \Omega$  do
2.    $w_i \leftarrow |S_i^T|$ 
3. end for
4. while  $|U| > 0$  do {First pass}
5.   select  $i: w_i \geq w_j, \forall j \in U$ 
6.    $U \leftarrow U \setminus \{i\}$ 
7.    $C \leftarrow C \cup \{i\}$ 
8.   for all  $j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

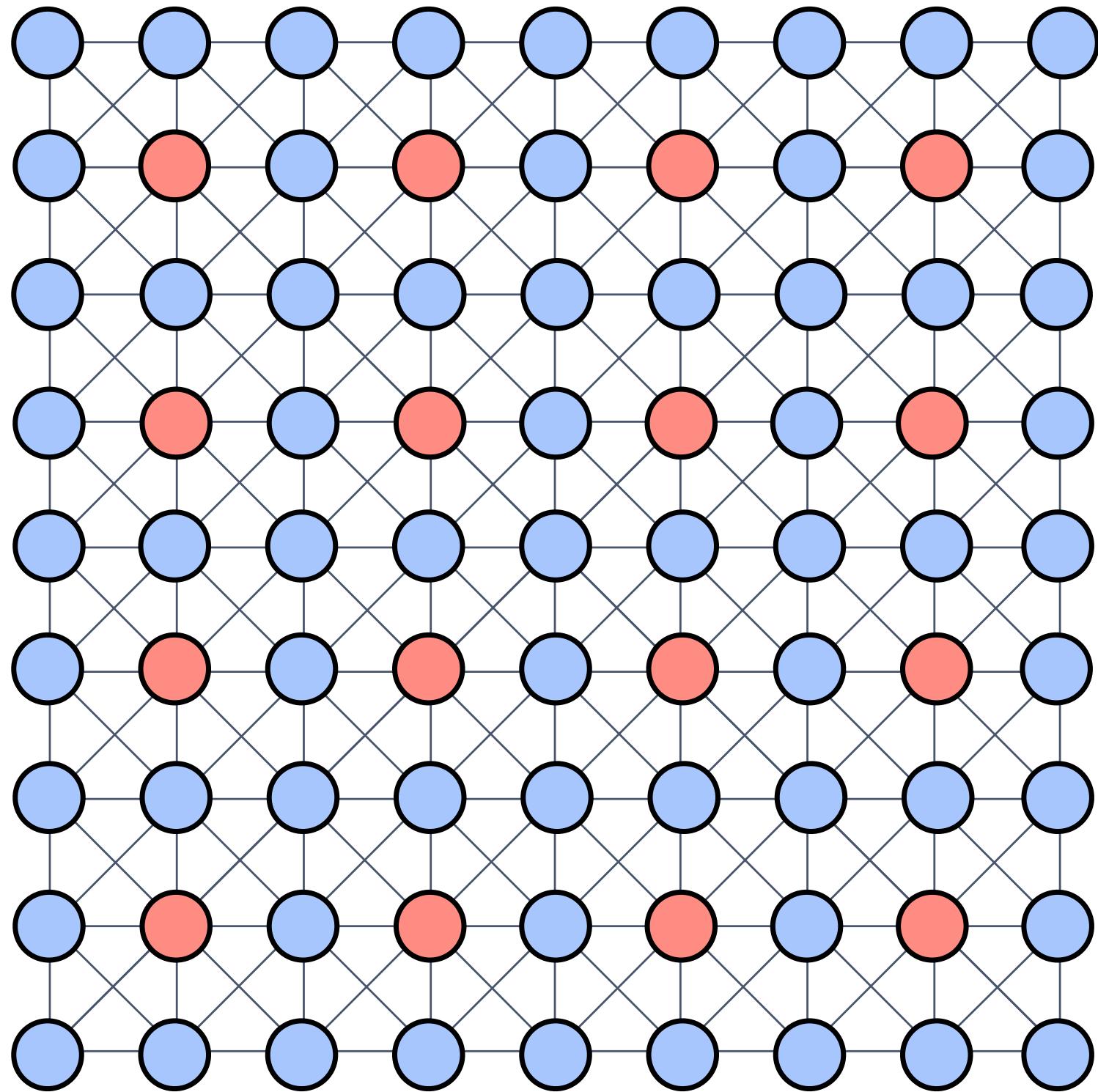












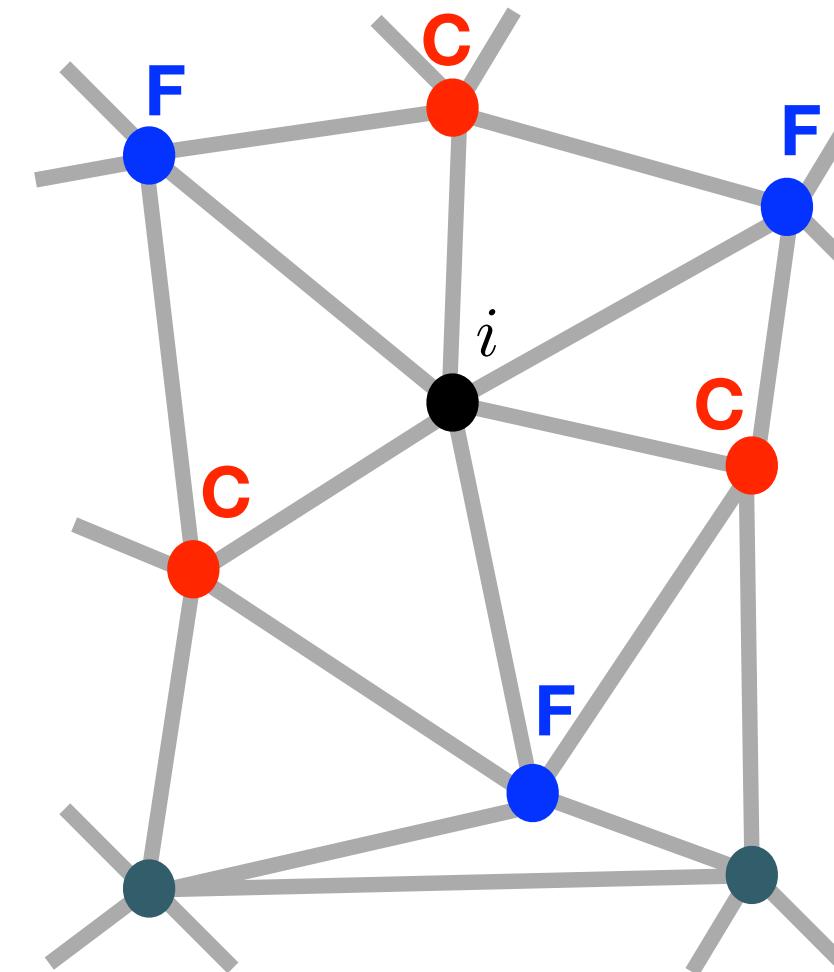
CF AMG

- 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

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

- Or

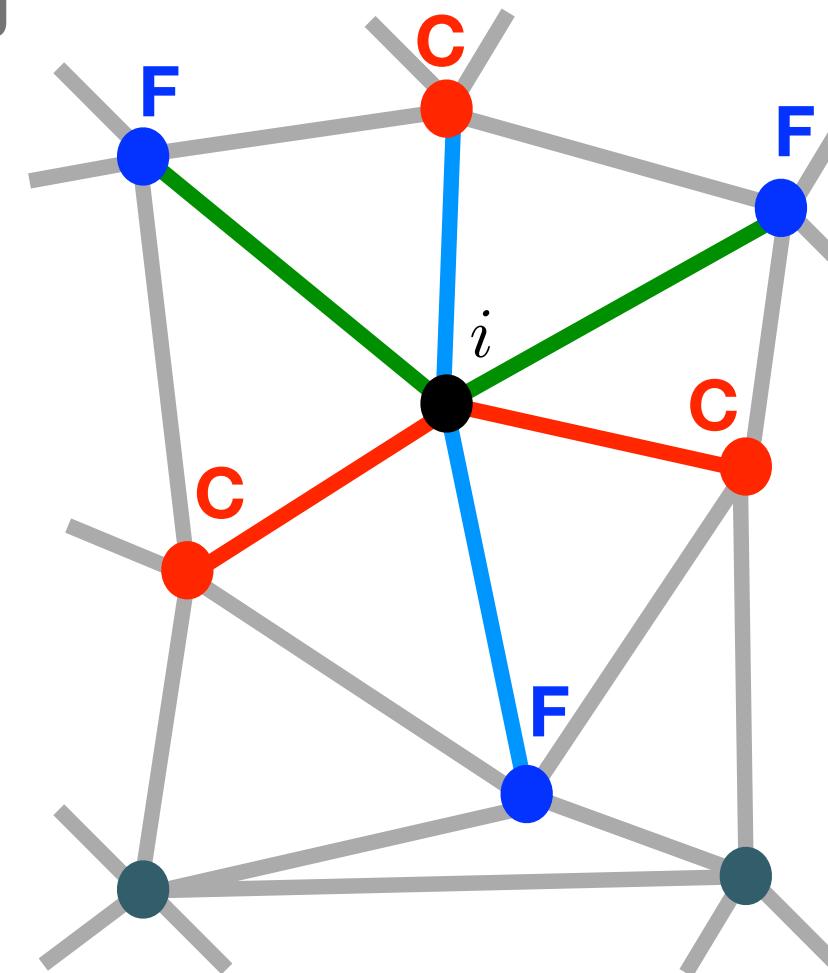
$$\vec{P} \vec{e} = P \begin{bmatrix} \vec{e}_F \\ \vec{e}_C \end{bmatrix} = \begin{bmatrix} W \\ I \end{bmatrix} \vec{e}_C$$



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

$$Ae \approx 0$$

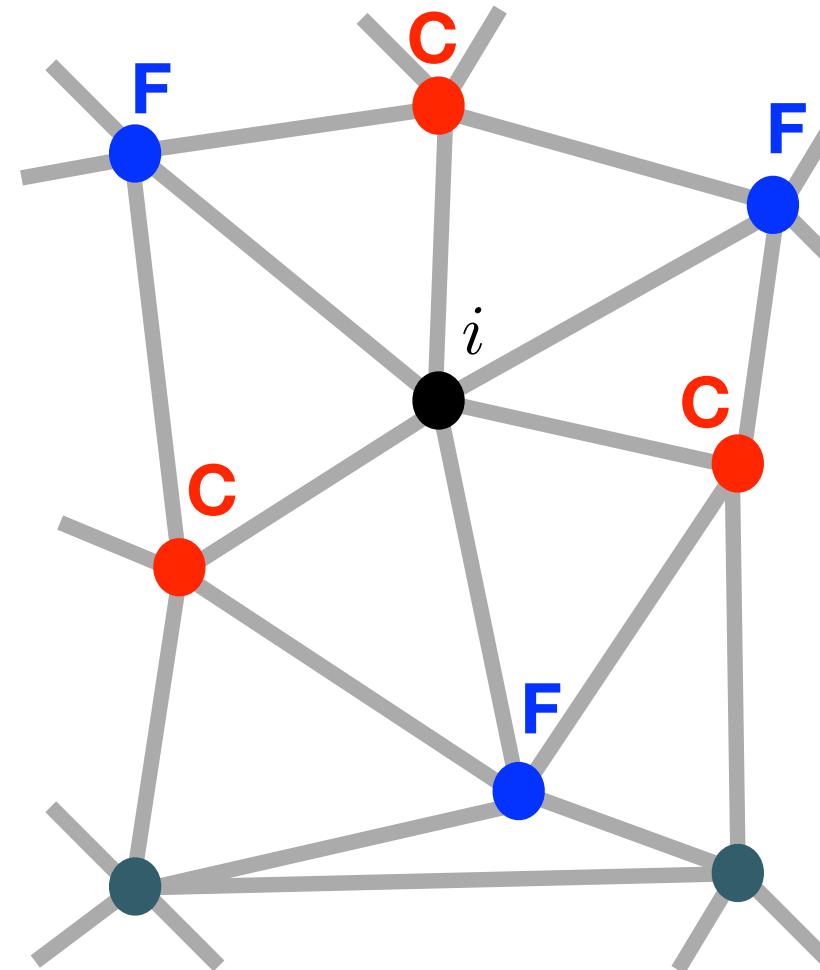
- Then “solve” for i

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

- Then split into types

$$A_{ii}e_i = - \sum_{j \in C_i} A_{ij}e_j - \sum_{k \in D_i^s} A_{ik}e_k - \sum_{m \in D_i^w} A_{im}e_m$$

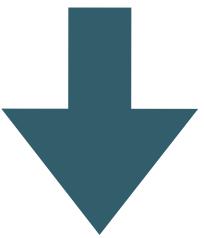
strong C **strong F** **weak**



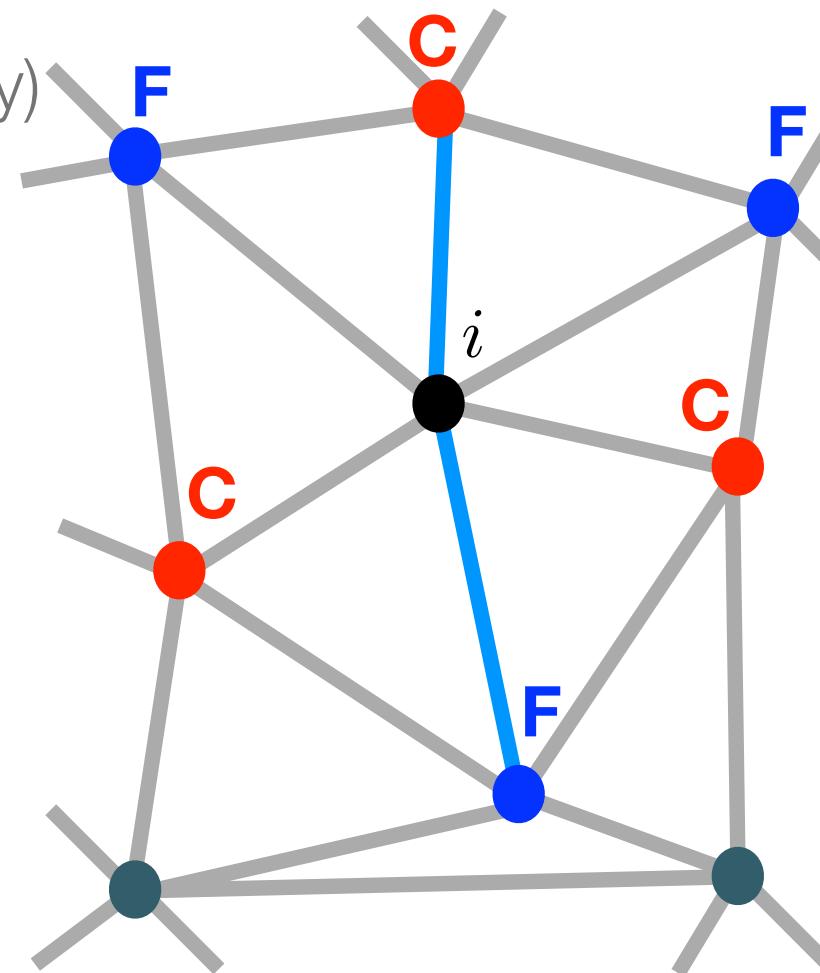
CF AMG

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

$$A_{ii}e_i = - \sum_{j \in C_i} A_{ij}e_j - \sum_{k \in D_i^s} A_{ik}e_k - \sum_{m \in D_i^w} A_{im}e_m$$



$$\left(A_{ii} + \sum_{m \in D_i^w} A_{im} \right) e_i = - \sum_{j \in C_i} A_{ij}e_j - \sum_{k \in D_i^s} A_{ik}e_k$$



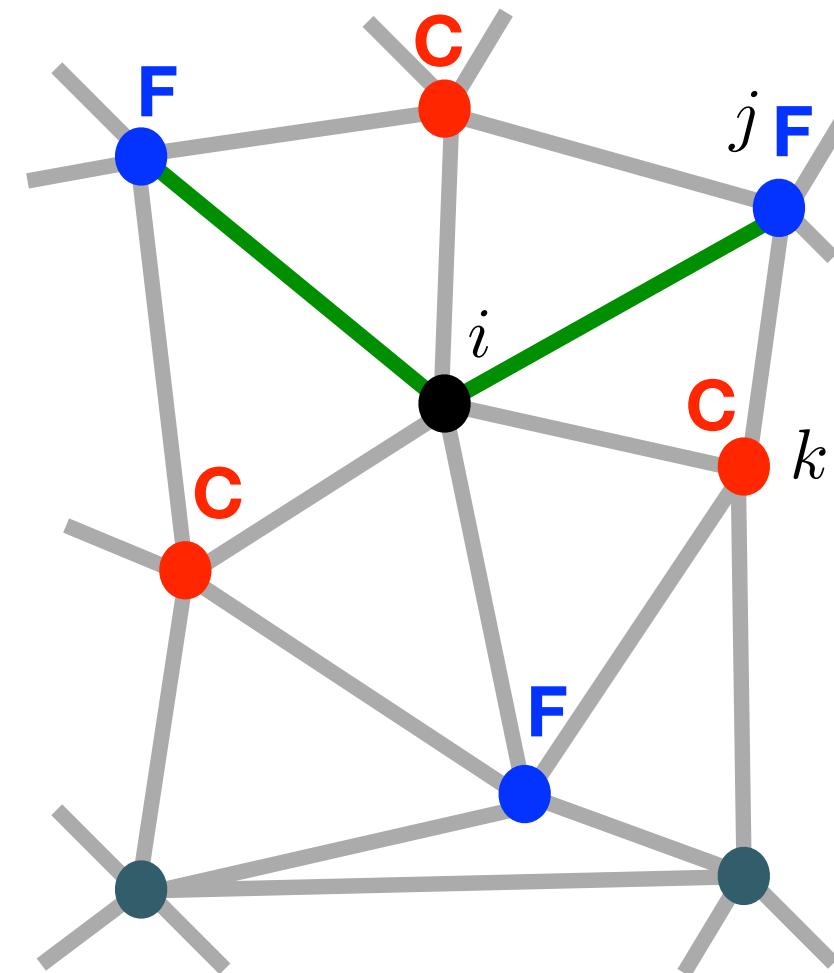
CF AMG

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

$$e_k \approx \frac{\sum_{k \in C_i} A_{kj} e_j}{\sum_{\ell \in C_i} A_{\ell j}}$$

- This gives weights

$$\omega_{ij} = -\frac{A_{ij} + \sum_{k \in D_i^s} \left(\frac{A_{ik} A_{kj}}{\sum_{\ell \in C_i} A_{\ell j}} \right)}{A_{ii} + \sum_{m \in D_i^w} A_{im}}$$



CF AMG Setup Algorithm

Algorithm 2: CF_setup()

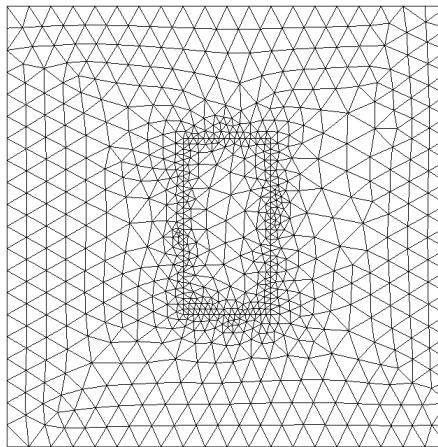
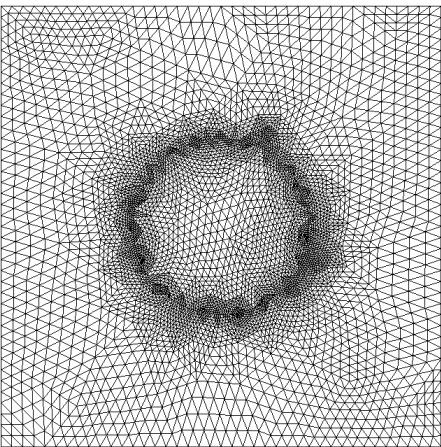
Input: A_0 : fine-grid operator
max_size: threshold for max size of coarsest problem

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

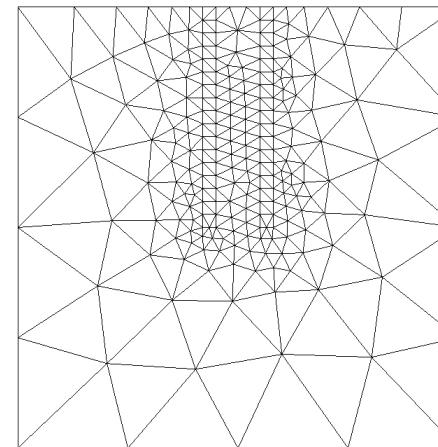
```
1  $\ell = 0$ 
2 while size( $A_\ell$ ) > max_size
3    $S_\ell = \text{strength}(A_\ell)$                                 {Strength-of-connection}
4    $\mathcal{C}_\ell, \mathcal{F}_\ell = \text{splitting}(S_\ell)$           {C/F-splitting}
5    $W = \text{weights}(S_\ell, A_\ell, \mathcal{C}_\ell,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

AMG grid hierarchies for several 2D problems

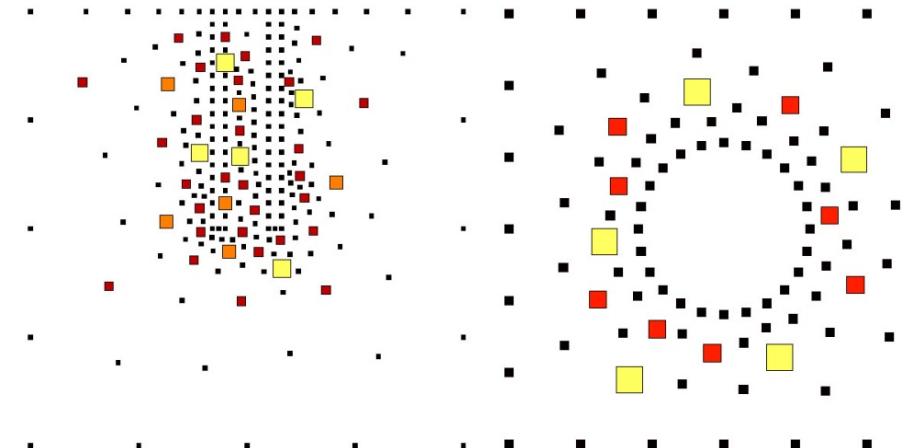
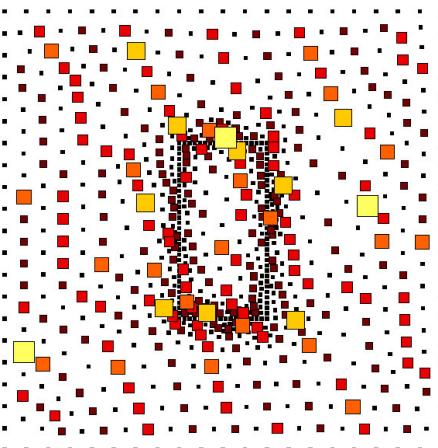
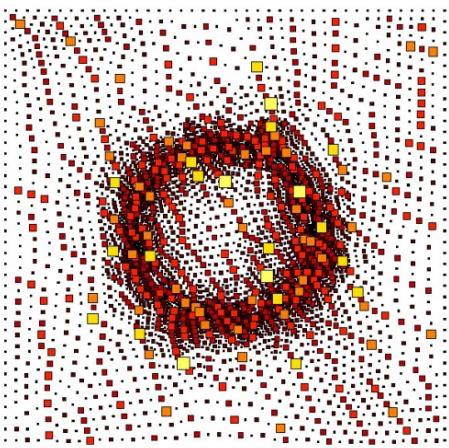
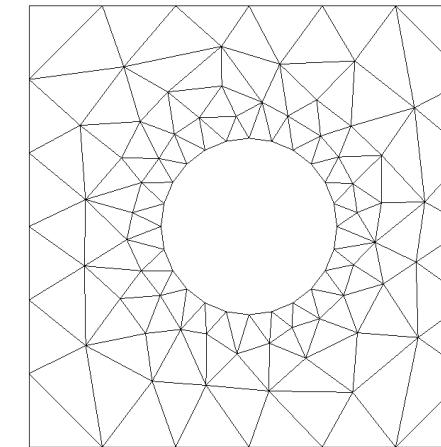
domain1 - 30° domain2 - 30°



pile



square-hole

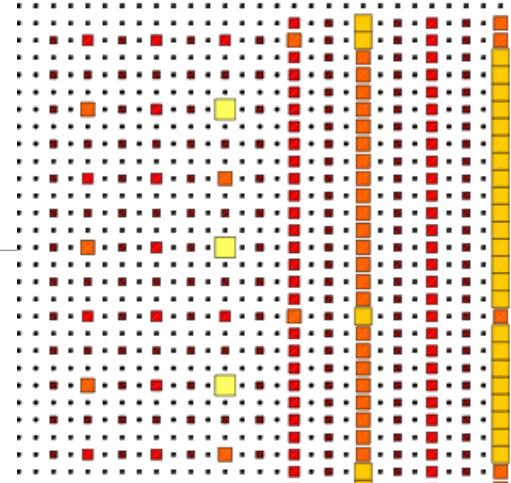


Example CF-AMG results

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

$$\boxed{a = b}$$

$$a \gg b$$



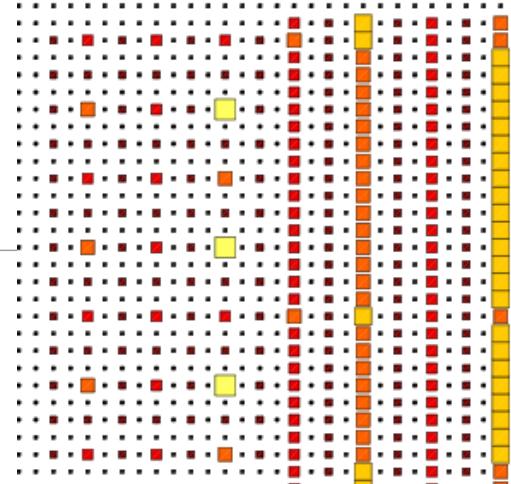
CF-AMG coarse grids

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

Example CF-AMG results

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

$$\begin{array}{c|c} a & b \\ \hline a & \gg b \end{array}$$



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



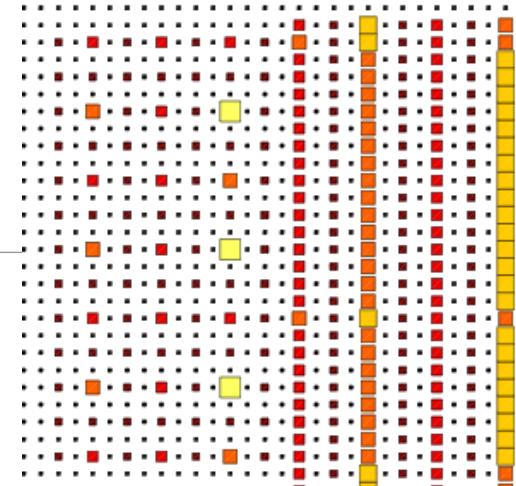
Iterations to a certain tolerance

Example CF-AMG results

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

$$\boxed{a = b}$$

$$a \gg b$$



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

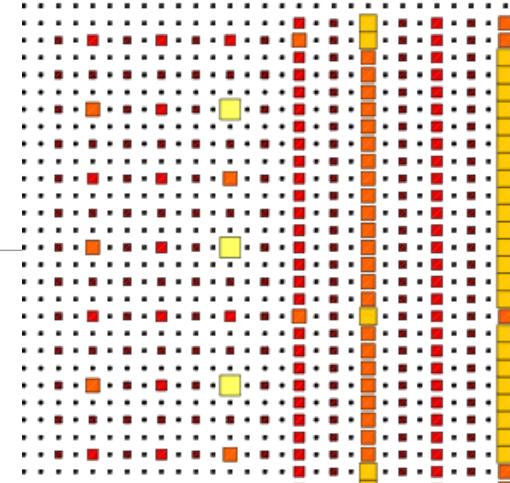


Convergence factor: factor by which the norm of the residual is reduced in each iteration

Example CF-AMG results

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

$$\begin{array}{c|c} a & b \\ \hline a & \gg b \end{array}$$



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

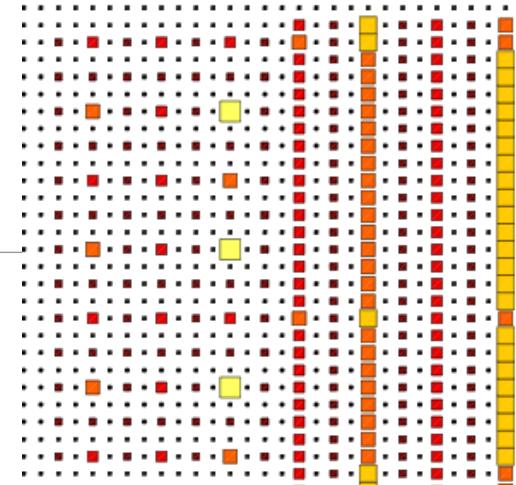


Coarse grids: number of coarse “grids”

Example CF-AMG results

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

$$\begin{array}{c|c} a & b \\ \hline a & \gg b \end{array}$$



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



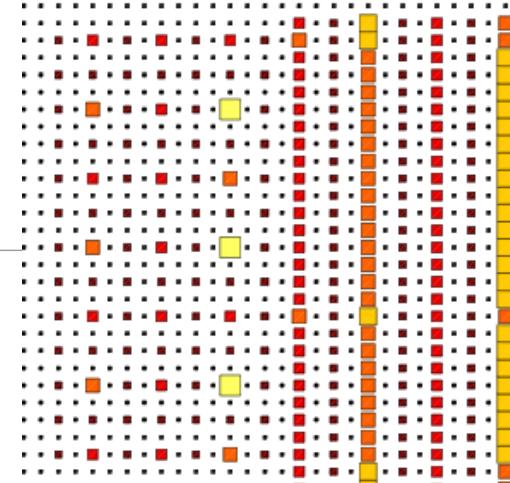
Grid complexity: total number of grid points (system size) on all levels / number of grid points on the fine level

Example CF-AMG results

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

$$\boxed{a = b}$$

$$a \gg b$$



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



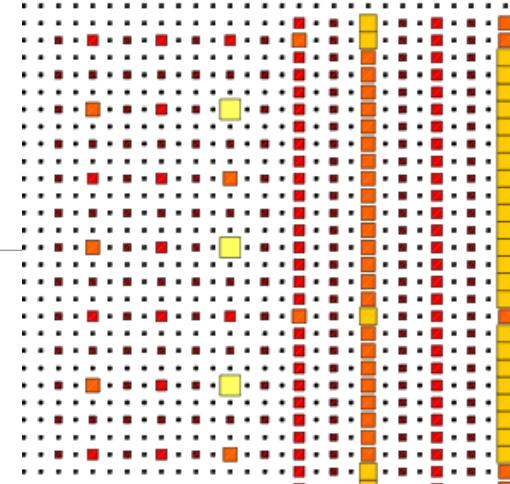
Operator complexity: total number non-zeros on all levels / number of non-zeros on the fine level

Example CF-AMG results

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

$$\boxed{a = b}$$

$$a \gg b$$



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!

Demo

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

Some theory

- An outline of some (general) theory.

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

- Let A be SPD

$$\tilde{M} = M^T (M^T + M - A)^{-1} M$$

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

Some theory

- The two-level theory shows

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

- 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:

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

- If $R = [0 \quad I]$

$$P^T = [W^T \quad I]$$

- Then $P_{ideal} = \operatorname{argmin}_P \sup_e \frac{\|(I - PR)e\|_{\tilde{M}}^2}{\|e\|_A^2}$
$$= \begin{bmatrix} -A_{FF}^{-1} A_{FC} \\ I \end{bmatrix}$$

M

- Try to optimize this:

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

Make this small
“compatible relaxation” methods

- If $R = [0 \quad I]$

$$P^T = [W^T \quad I]$$

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

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

Impractical, but often a good guide
AMGe-type methods

Demo

- 15-CF-AMG-Interpolation.ipynb

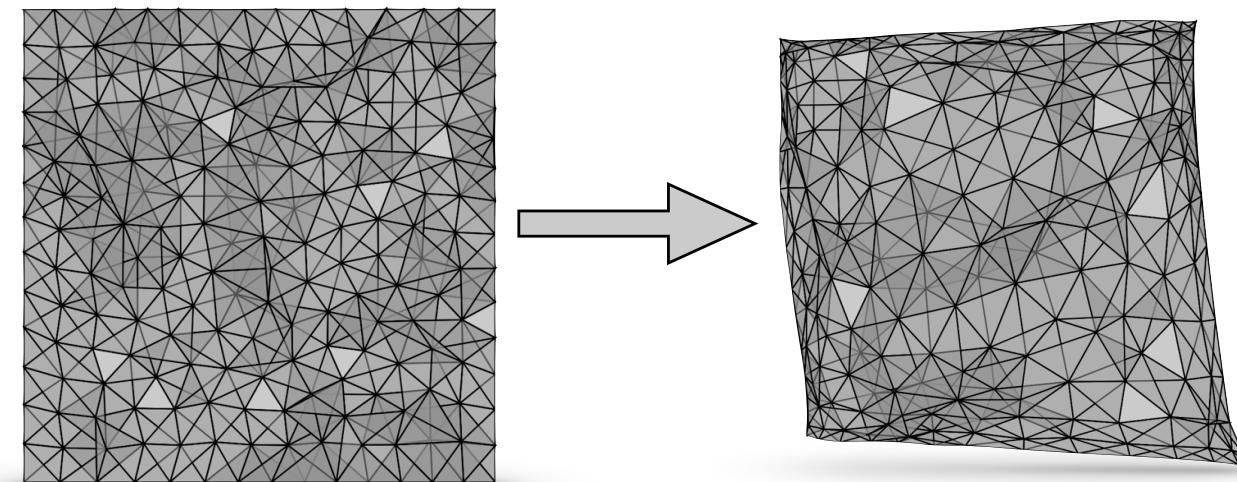
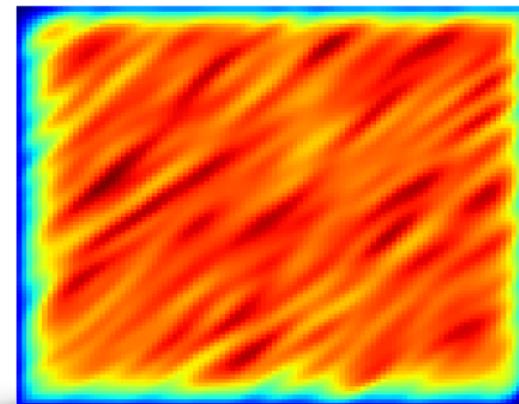
SA AMG

- 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

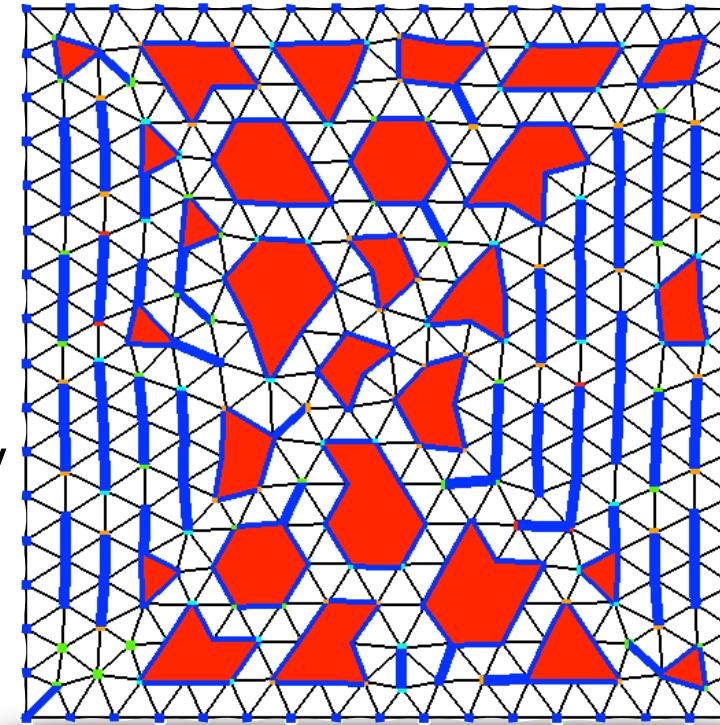
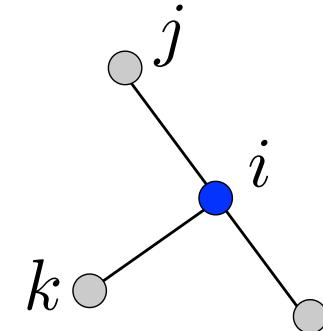
$$-A_{ij} \geq tol * \max_{k \neq i} -A_{ik}$$

- i strongly depends on j if

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

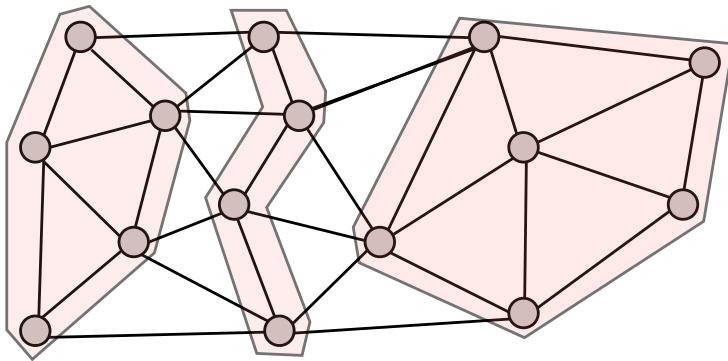
both “think” elliptic

↑
anisotropy
|



Algebraic Framework

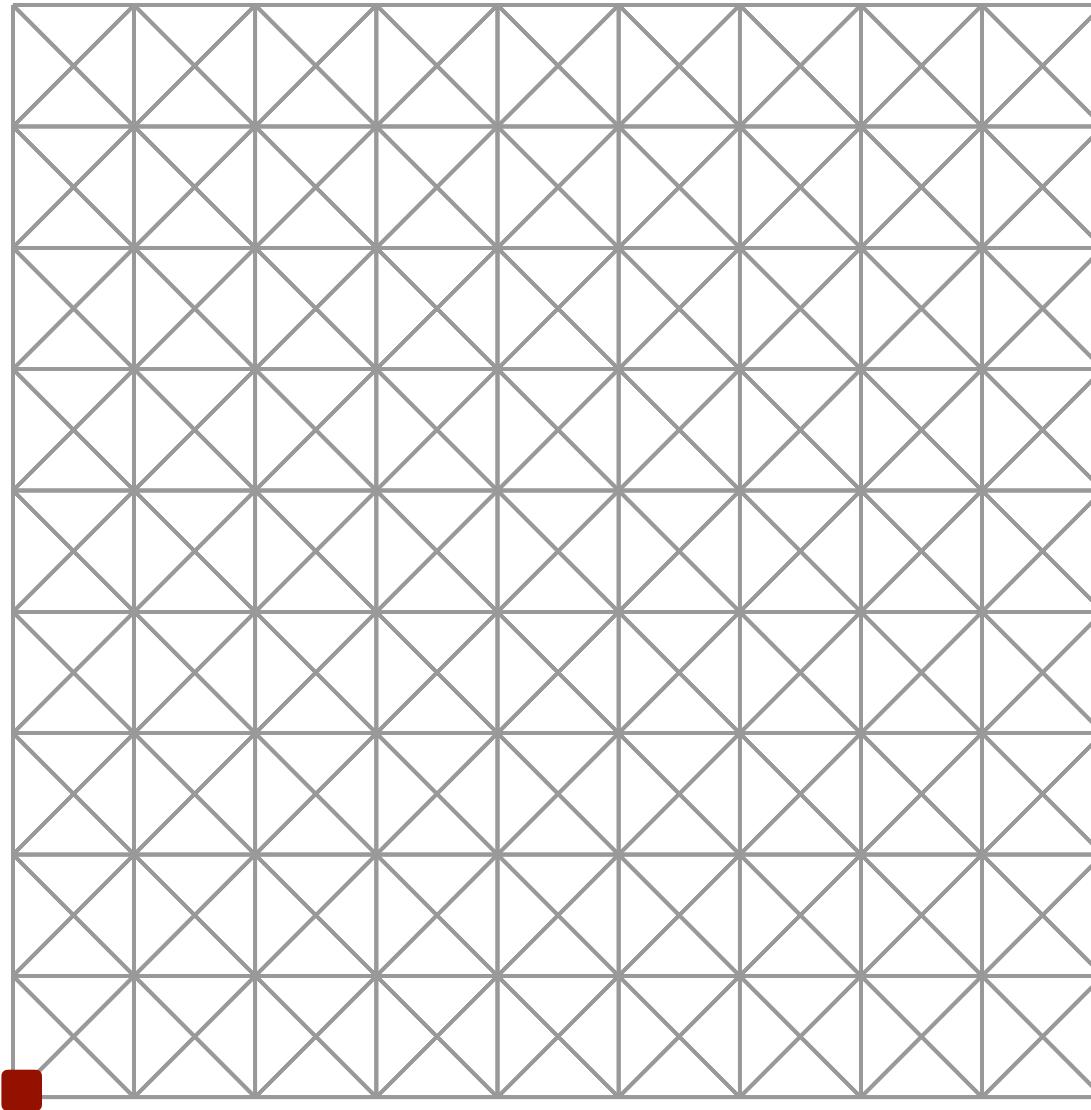
- aggregation: groups of fine nodes form coarse nodes



fine: 15
coarse: 3

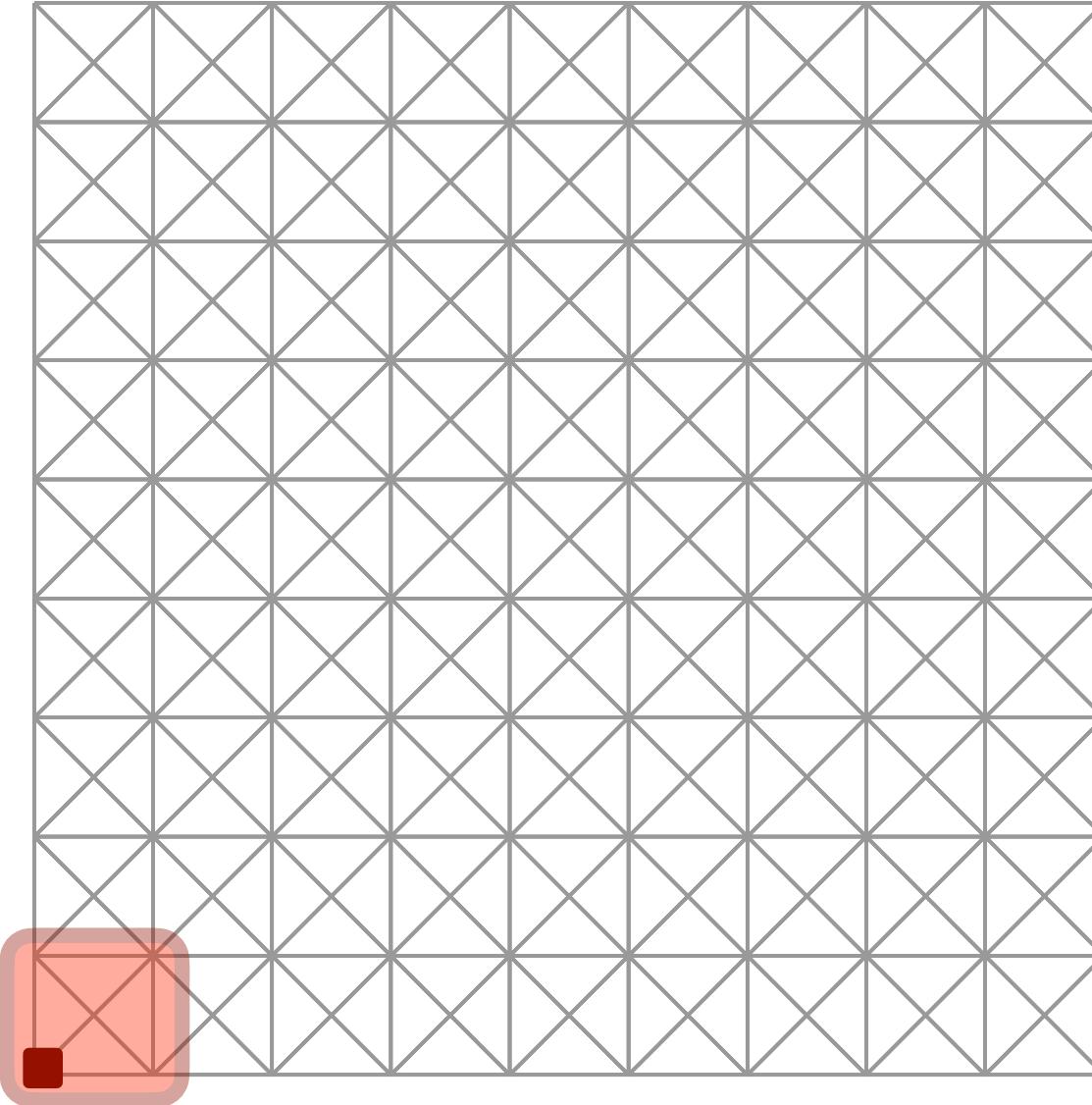
- an initial interpolation pattern
- find an optimal interpolation operator P that contains low energy

Aggregation



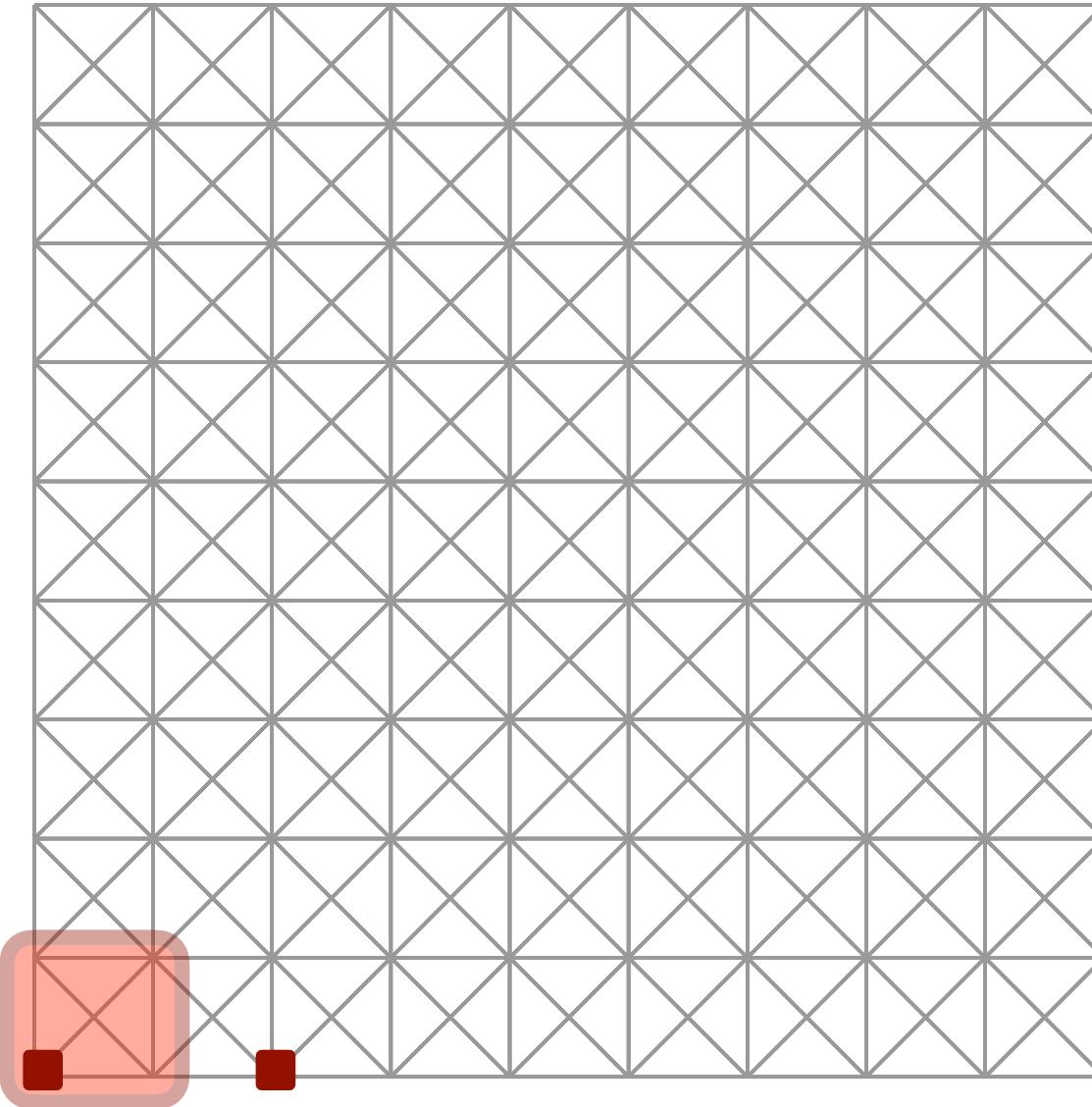
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



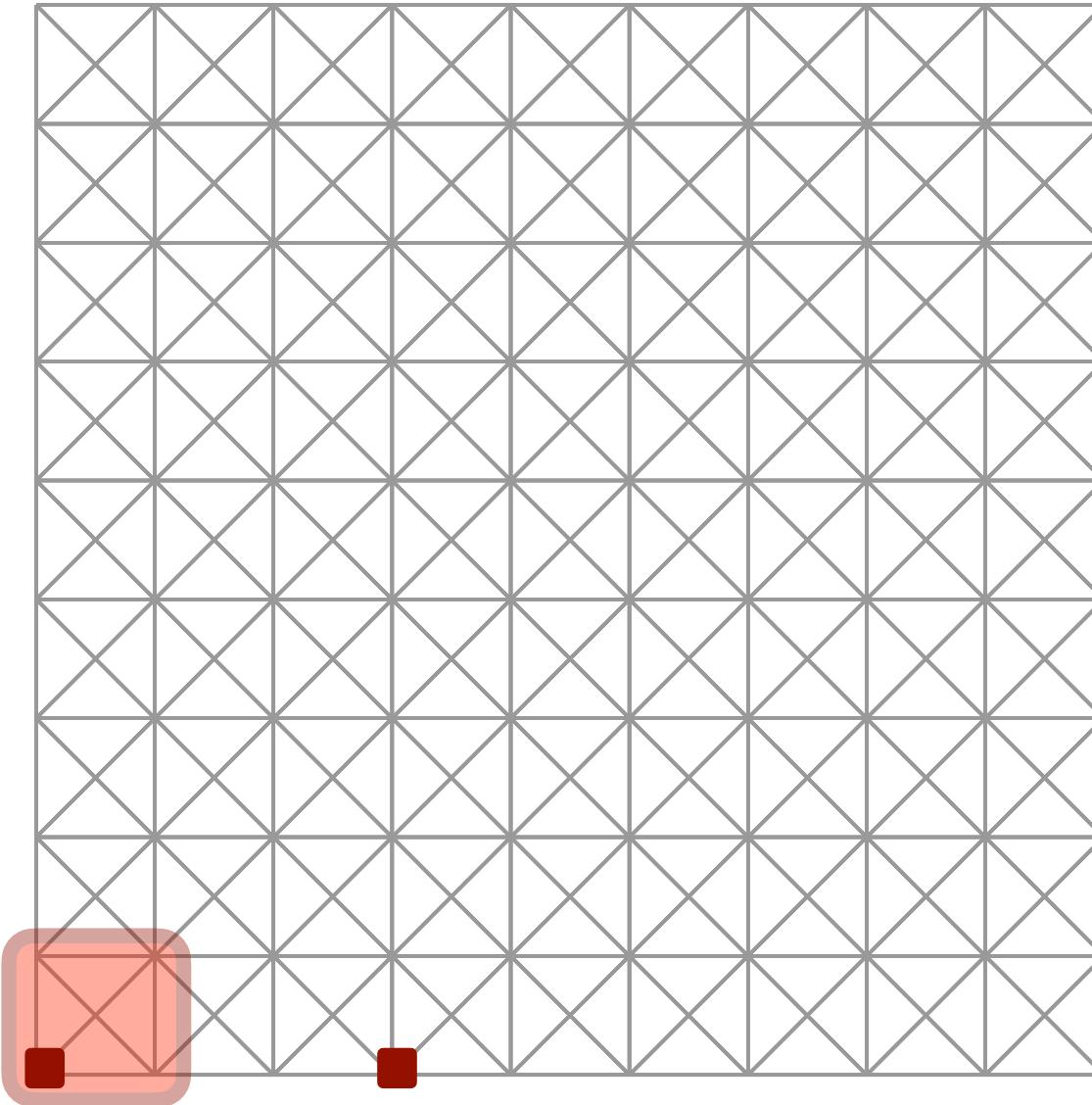
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



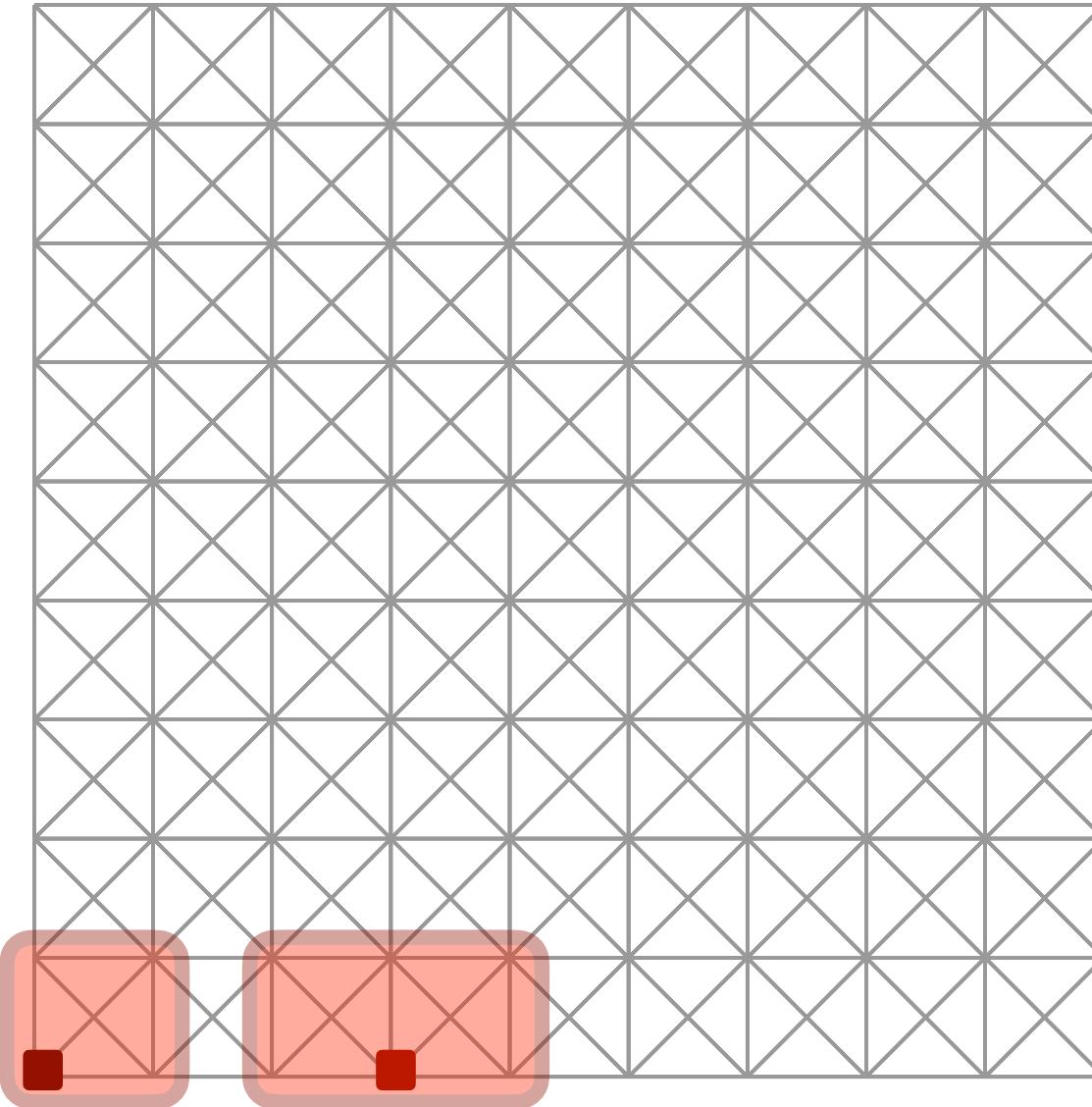
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



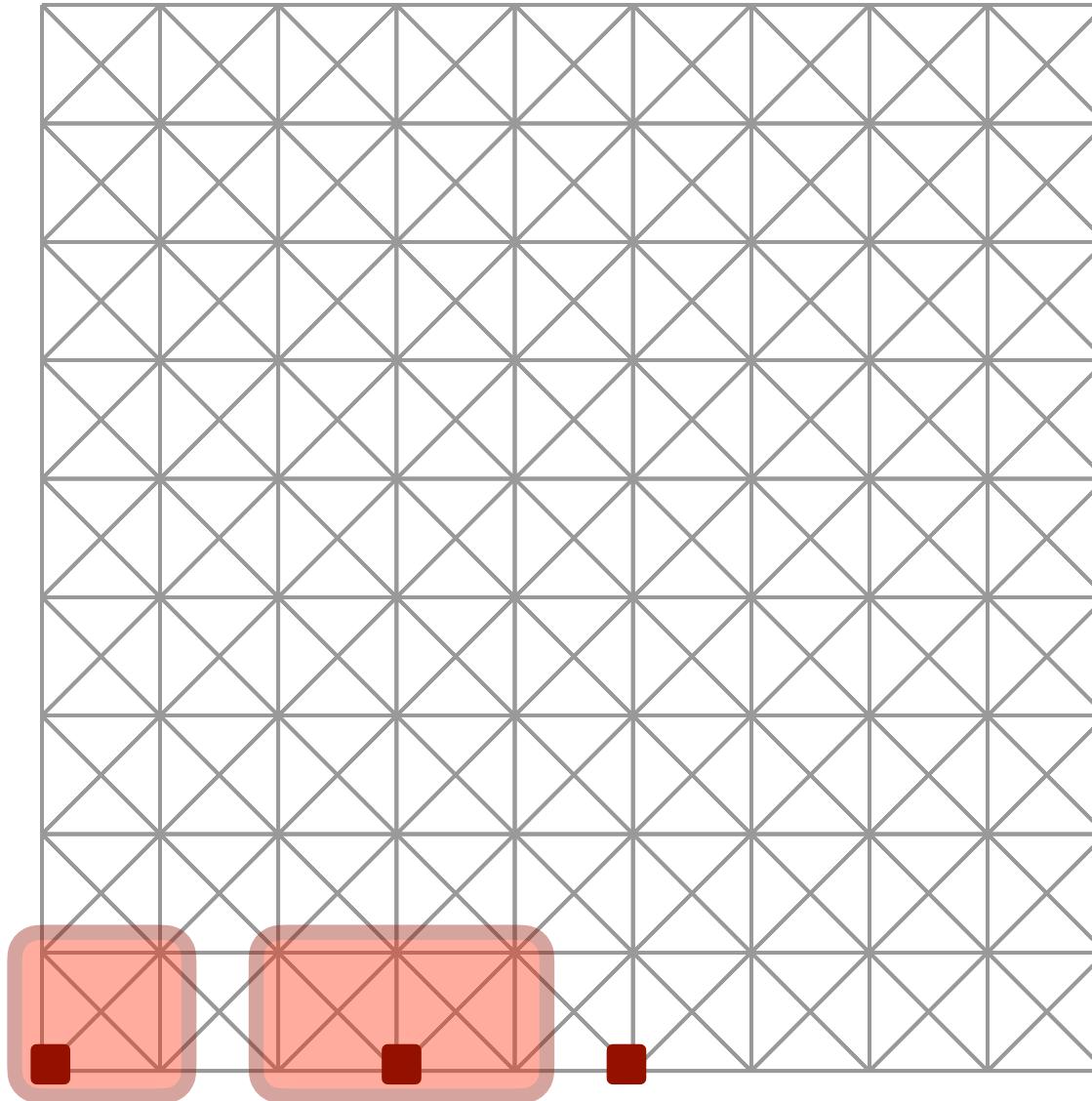
- Select next unaggregated node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



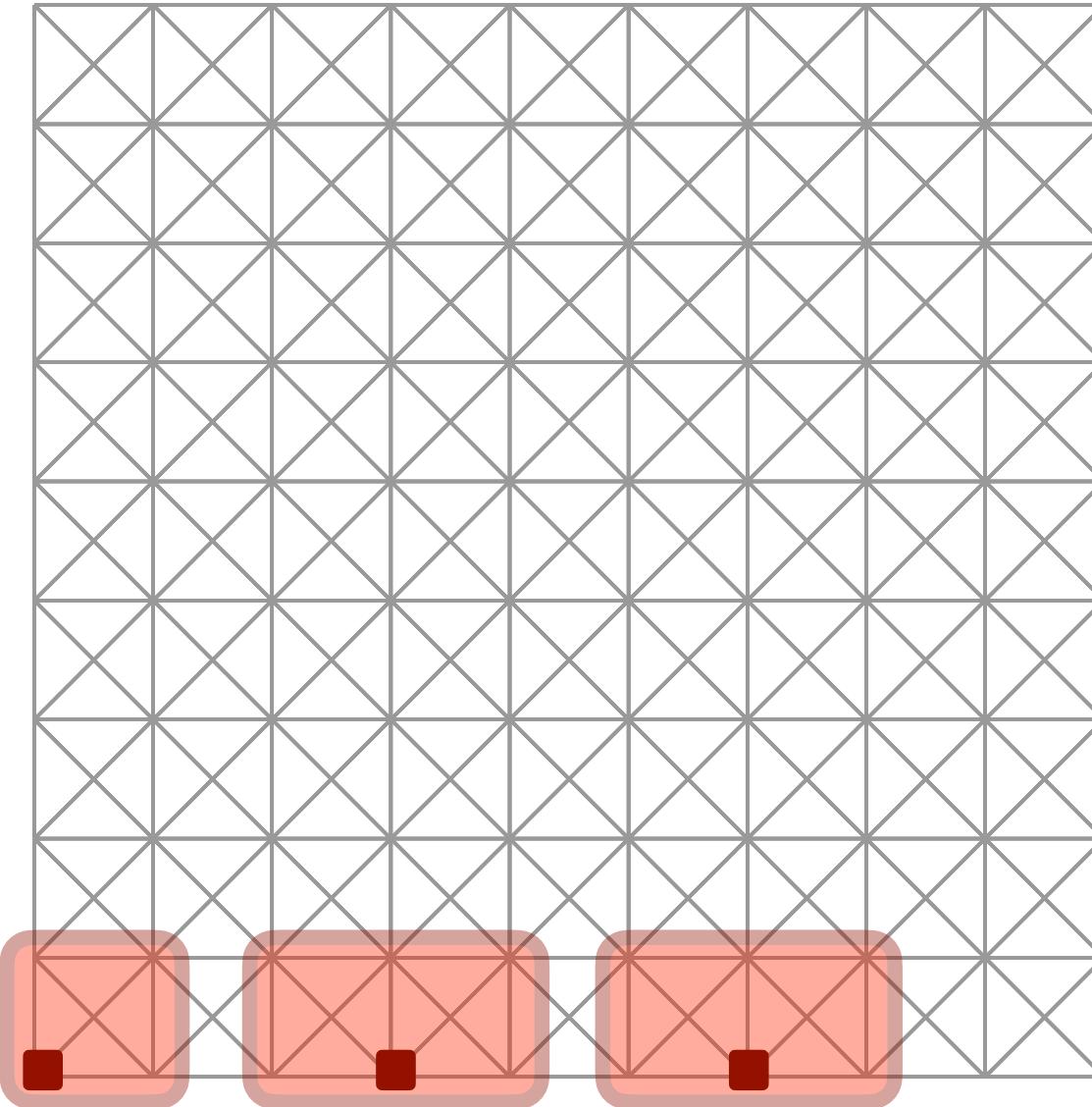
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



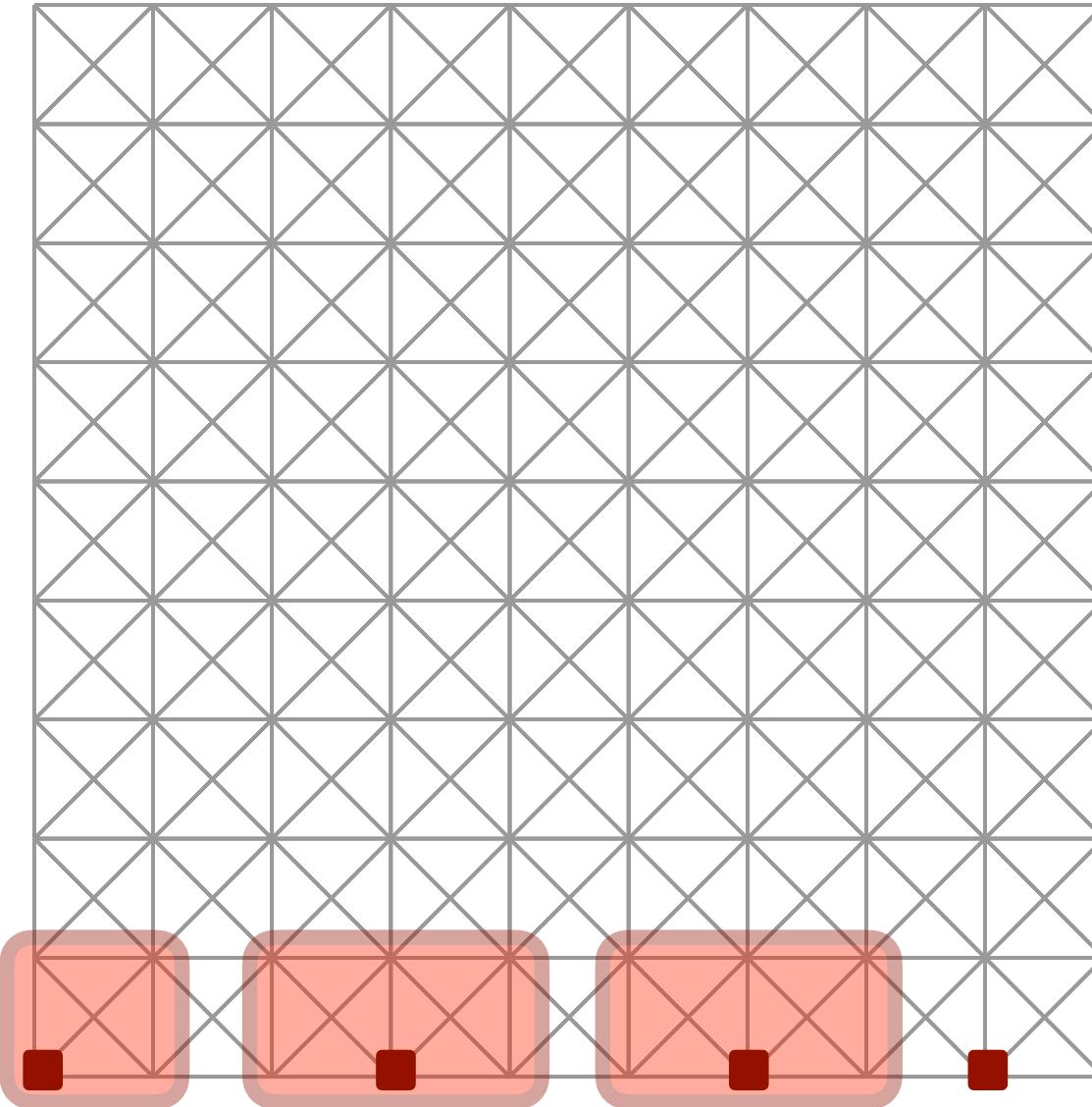
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



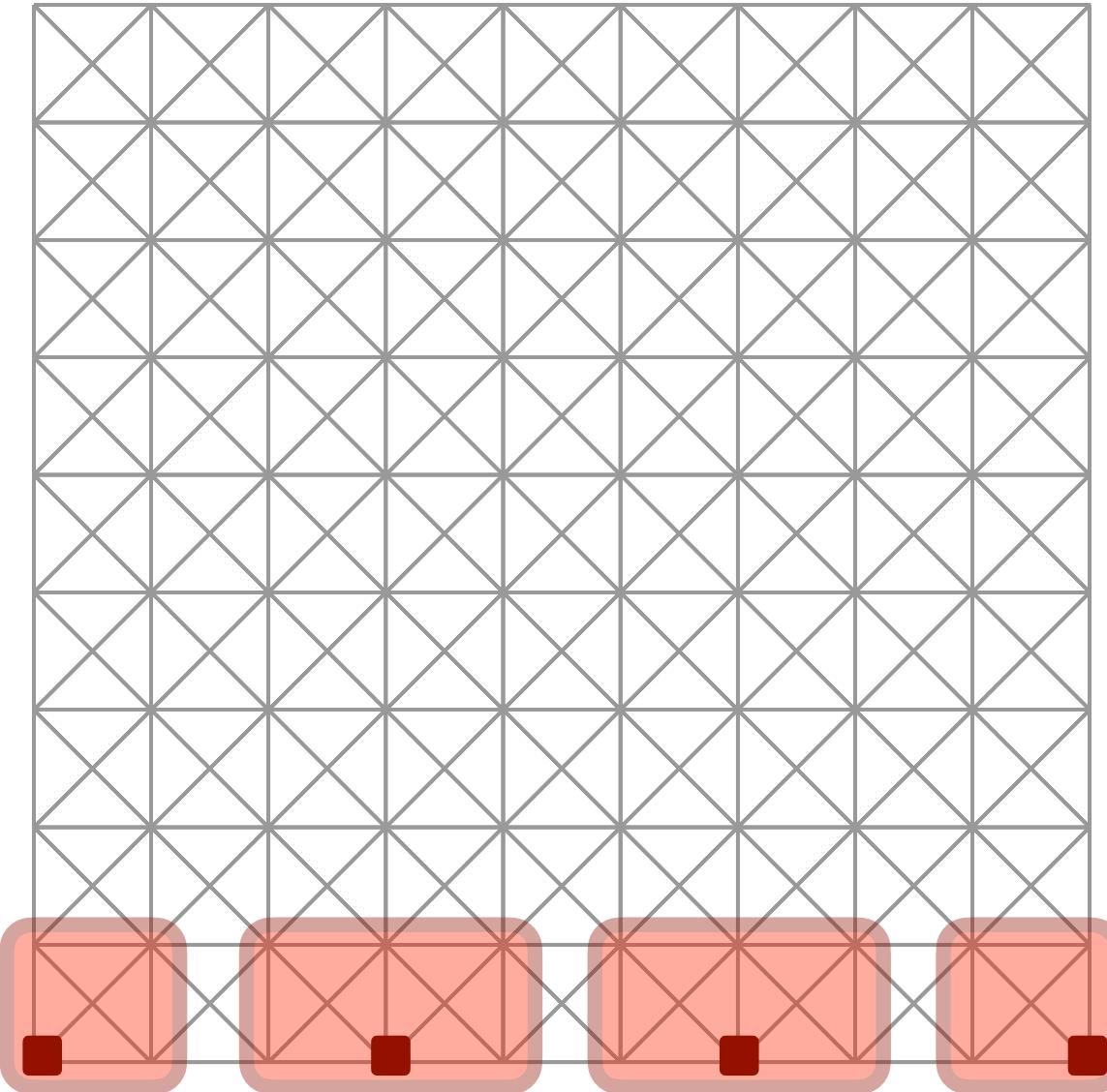
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



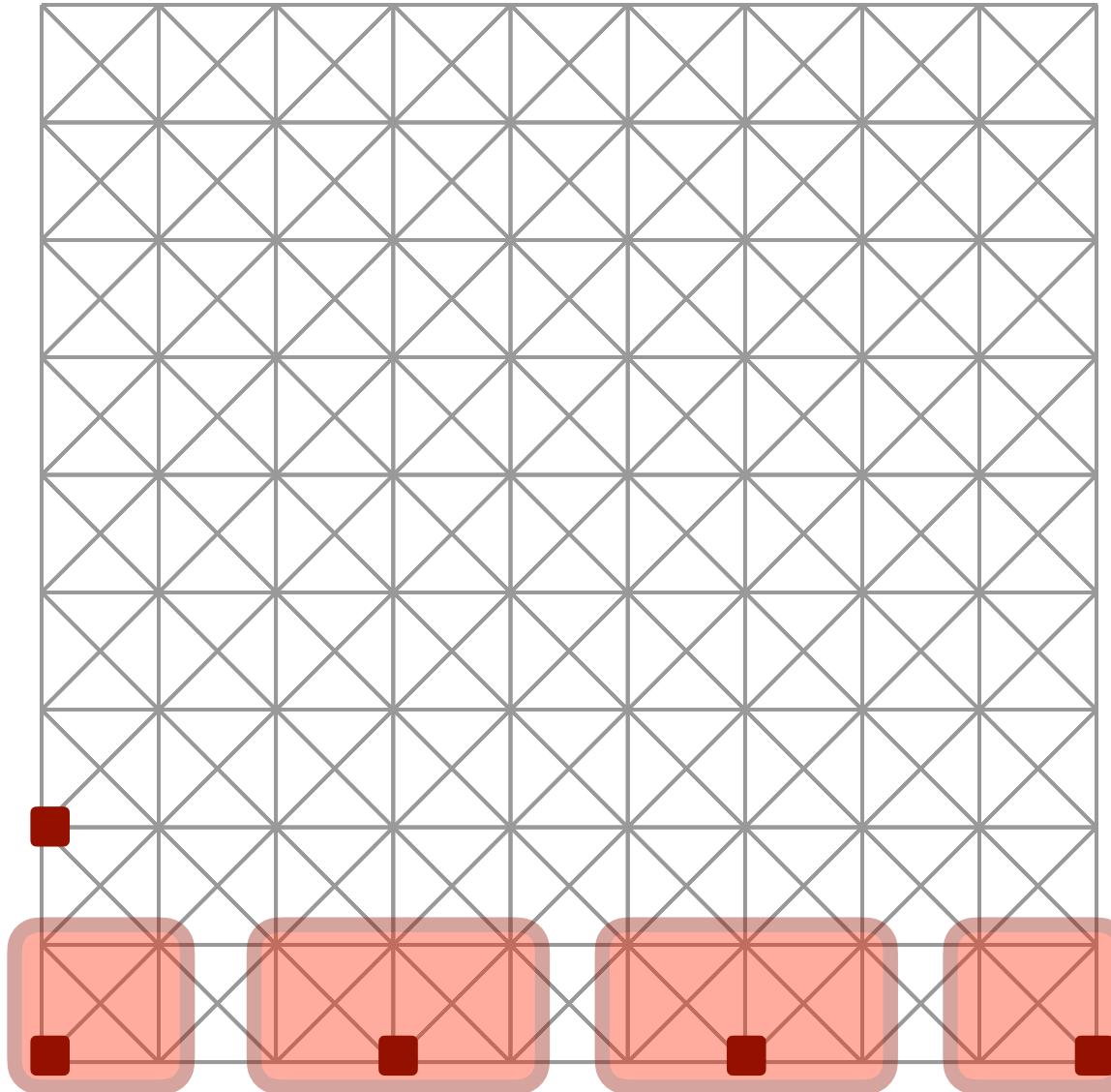
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



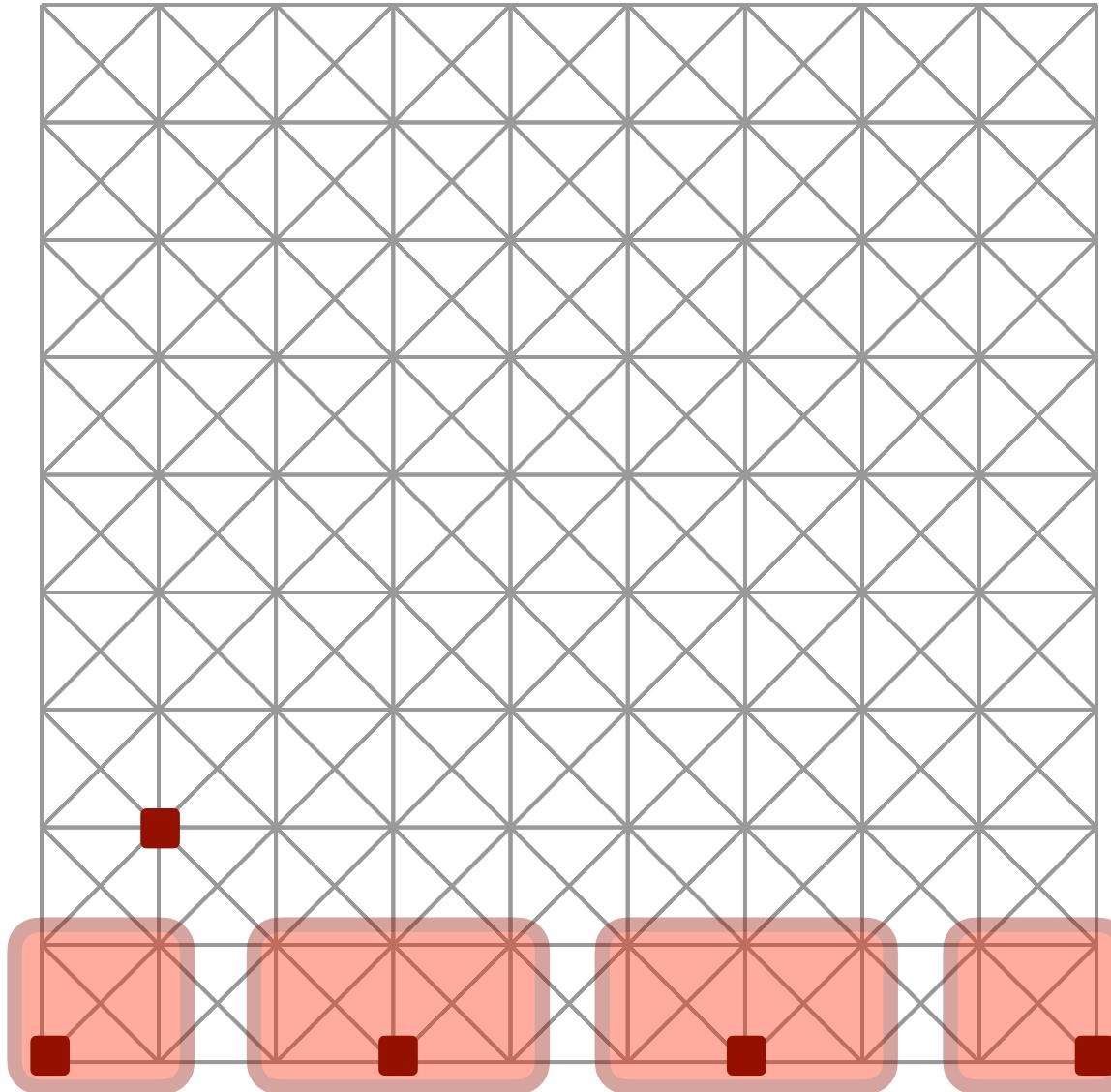
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



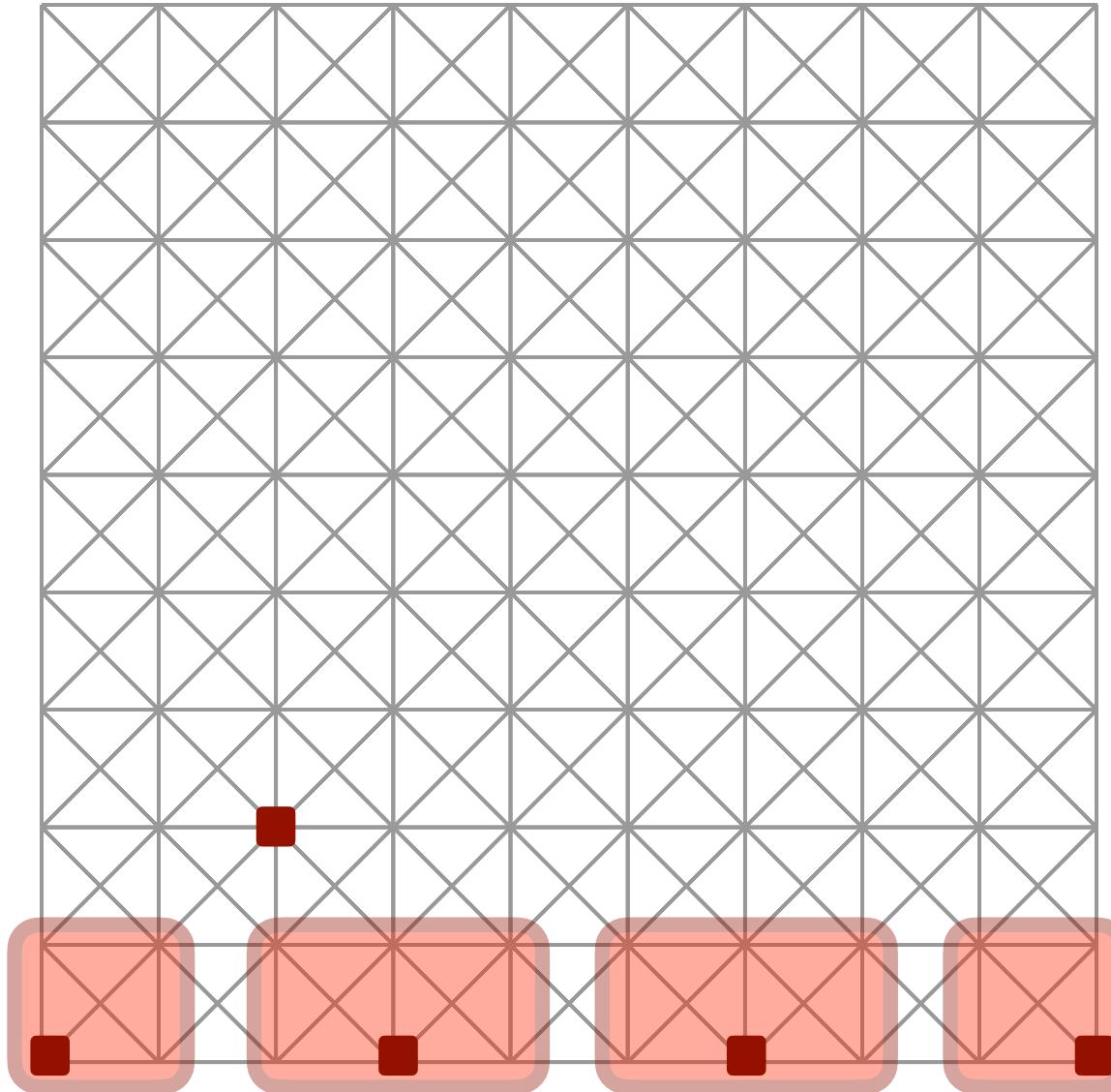
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



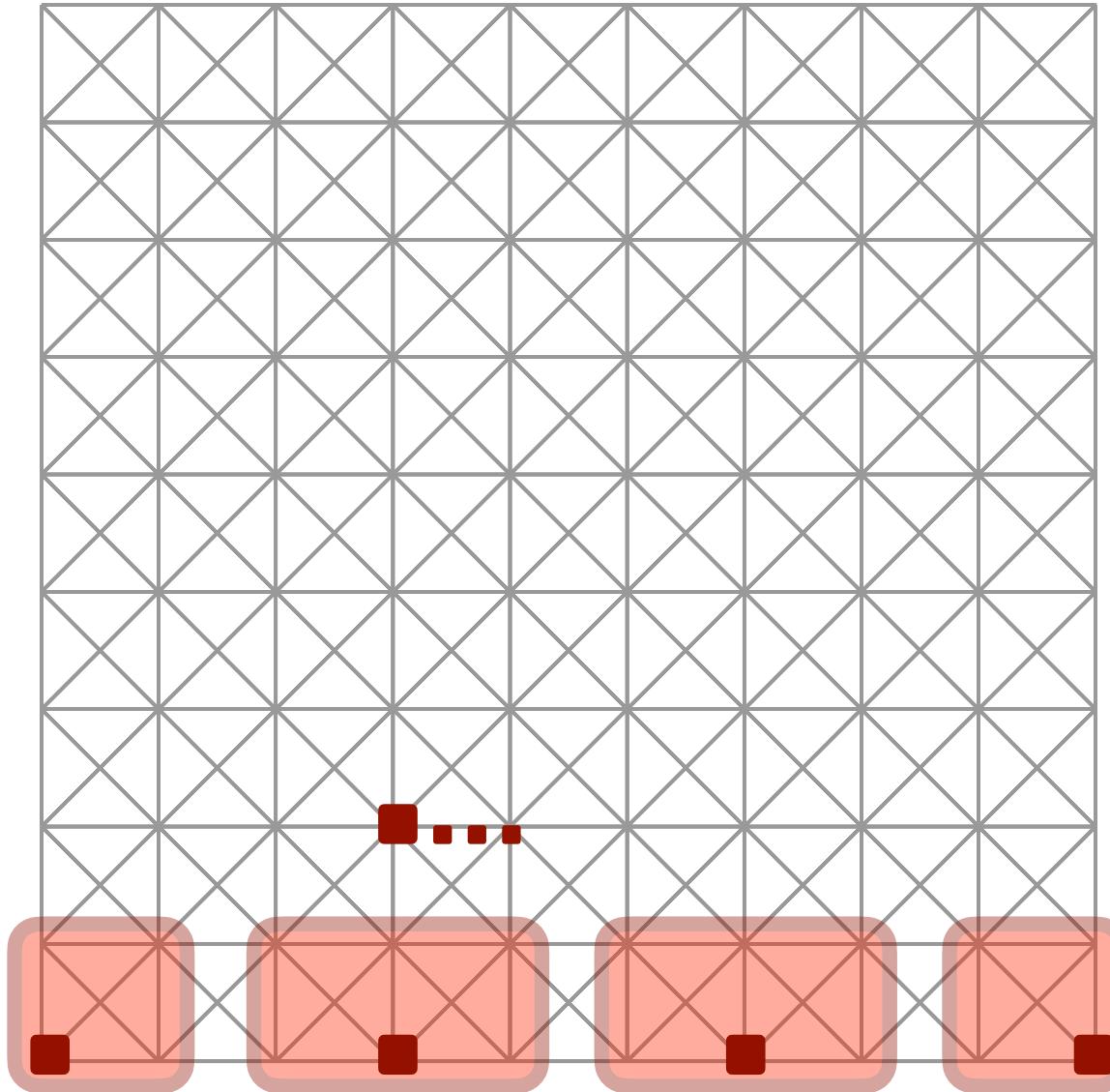
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



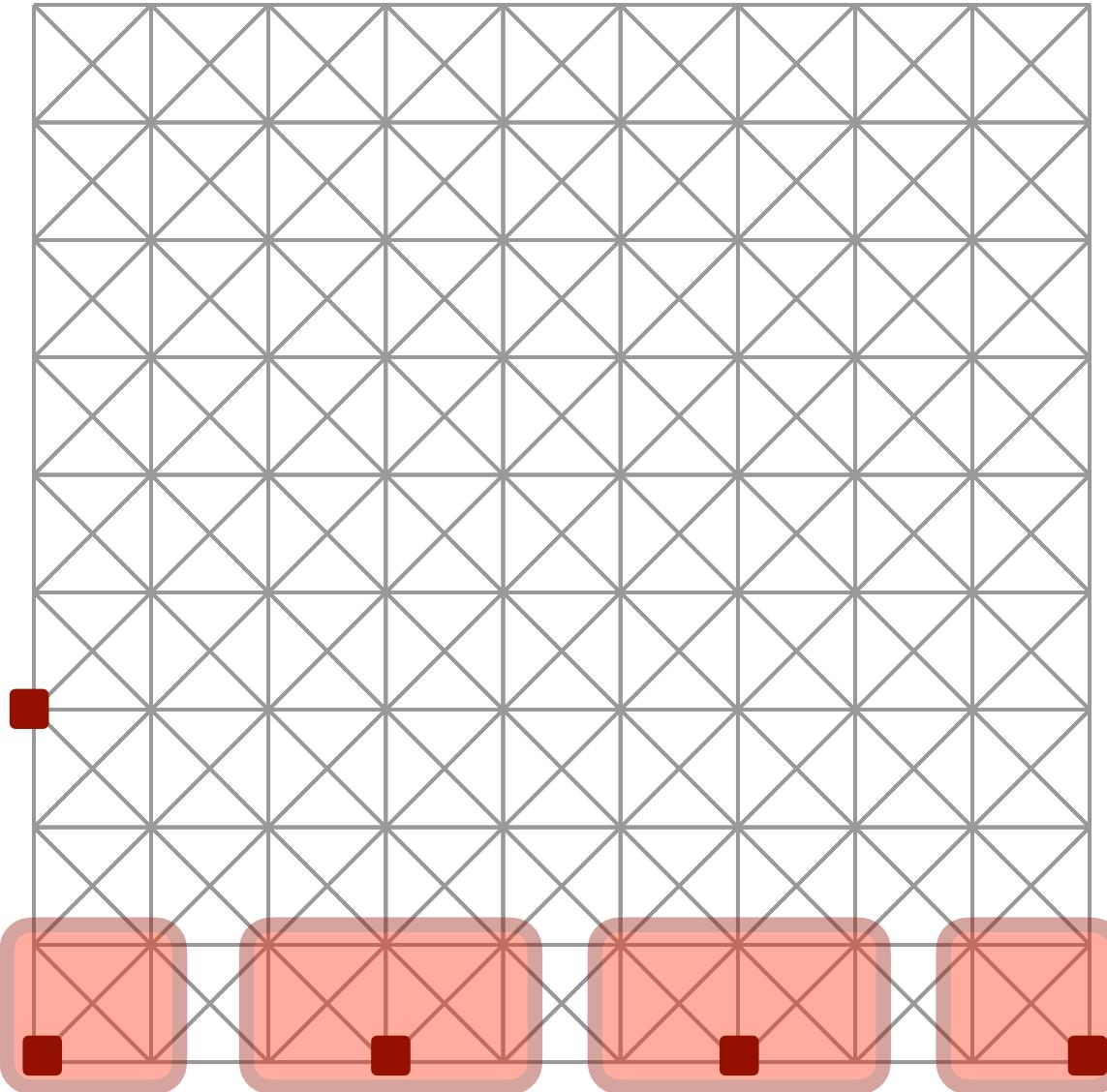
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



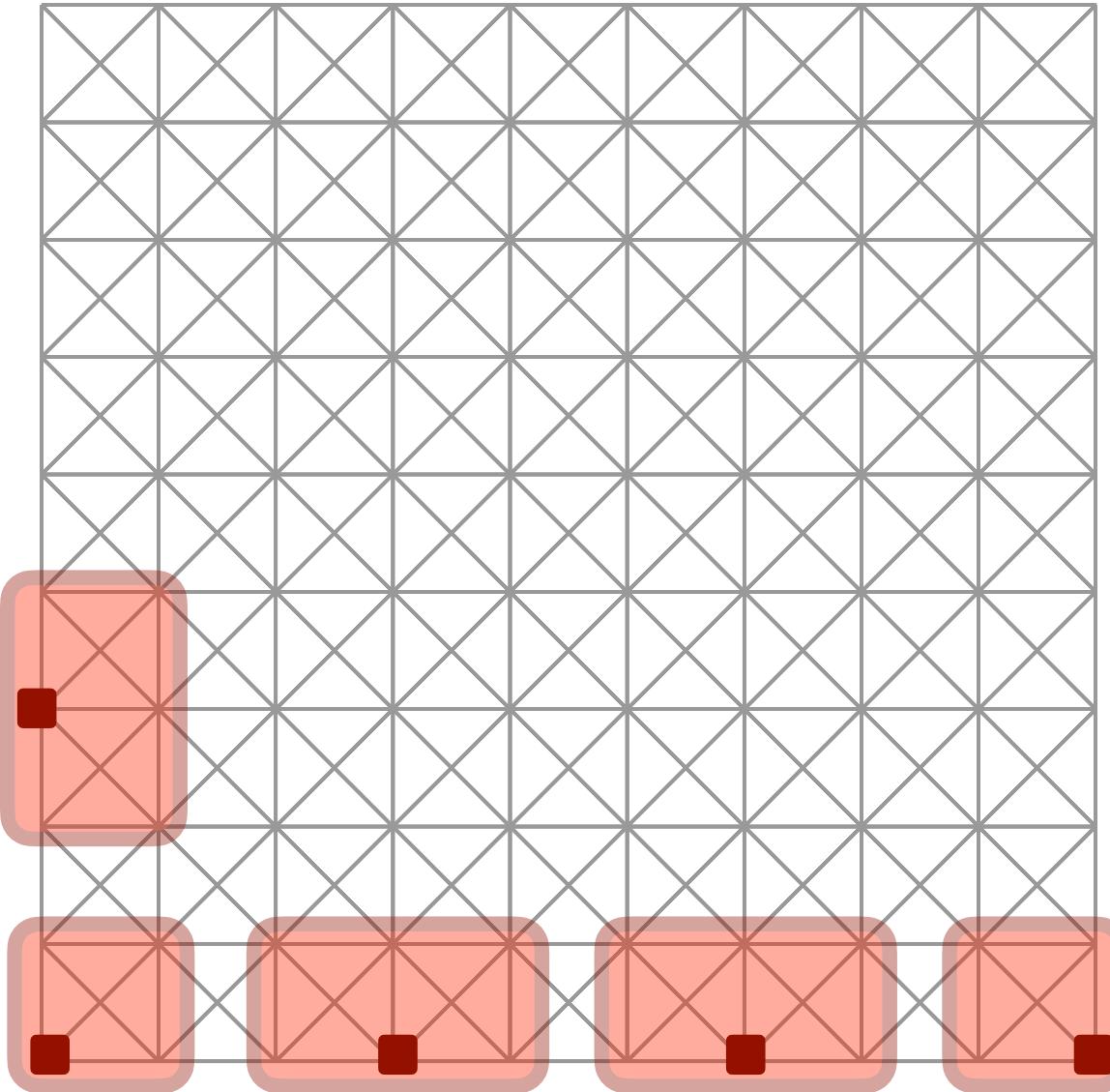
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



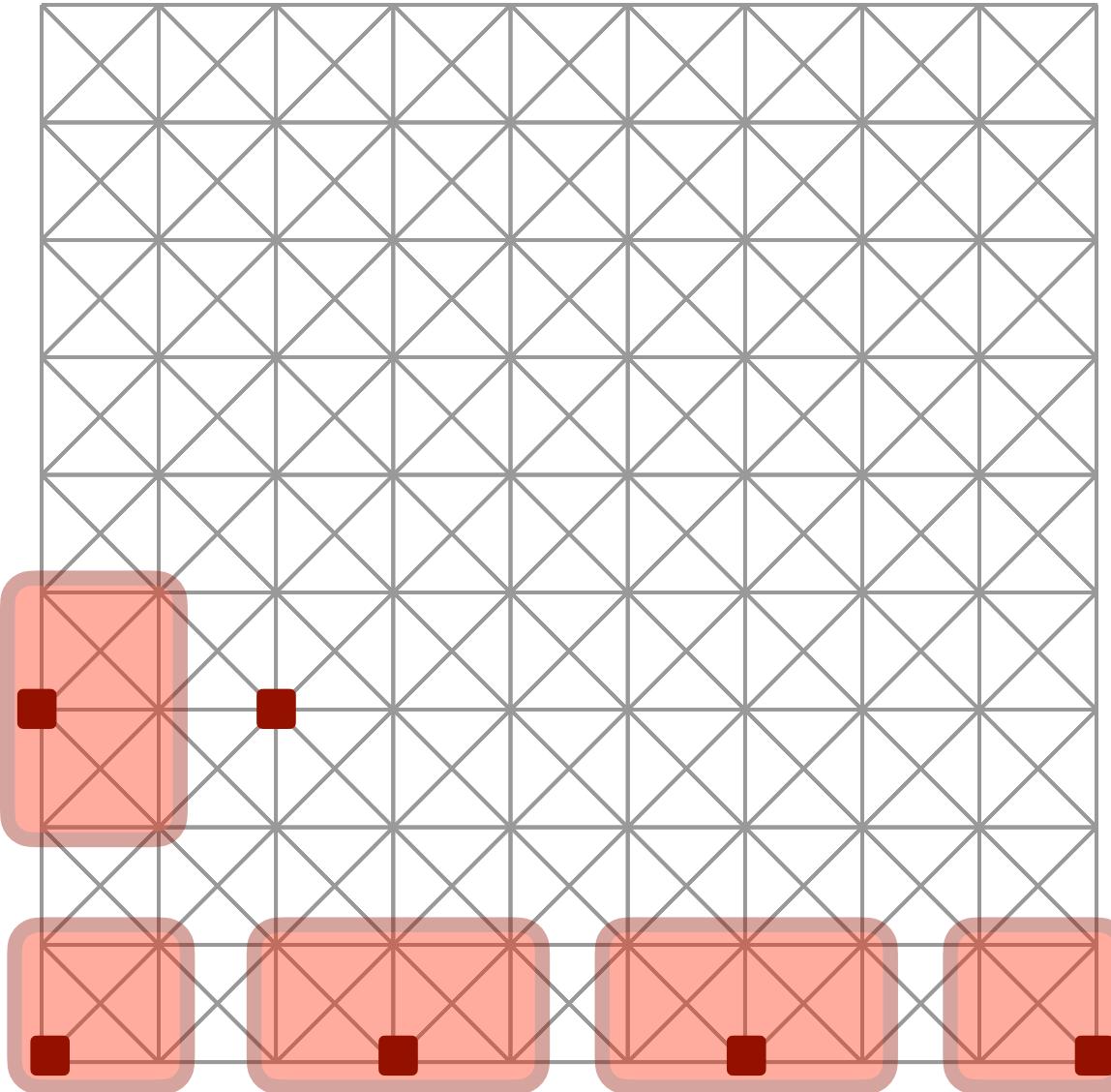
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



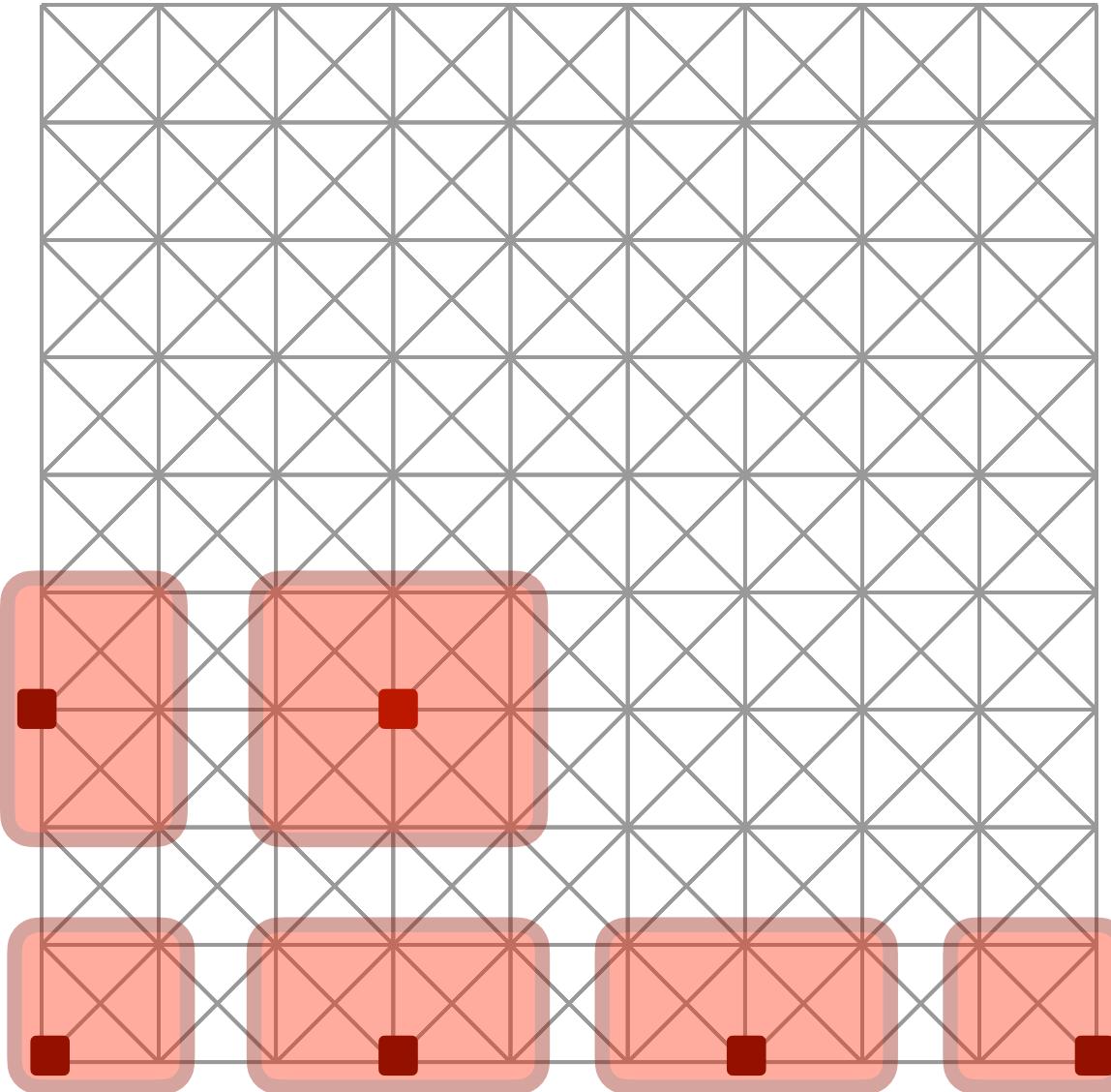
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



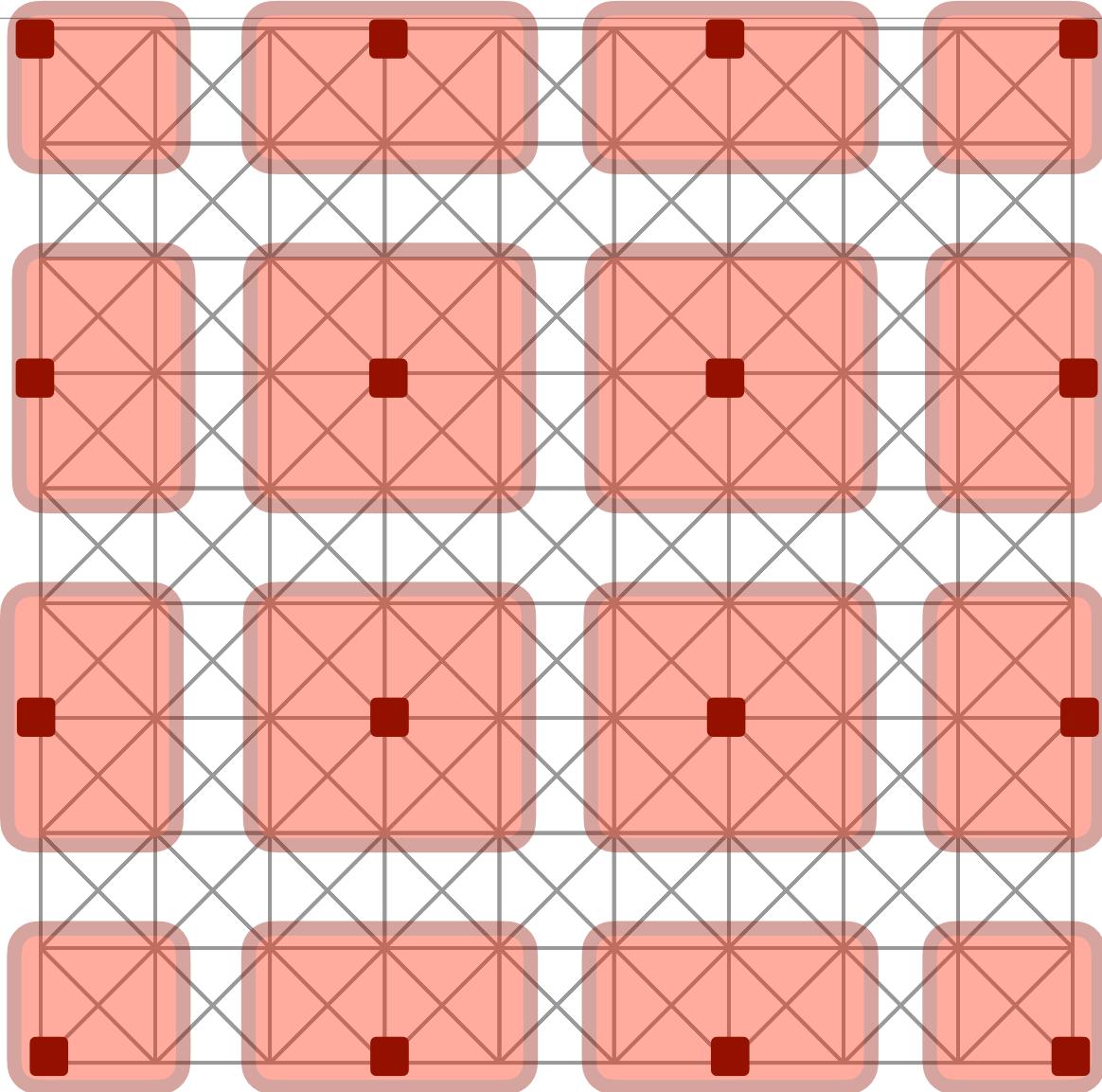
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

Aggregation



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

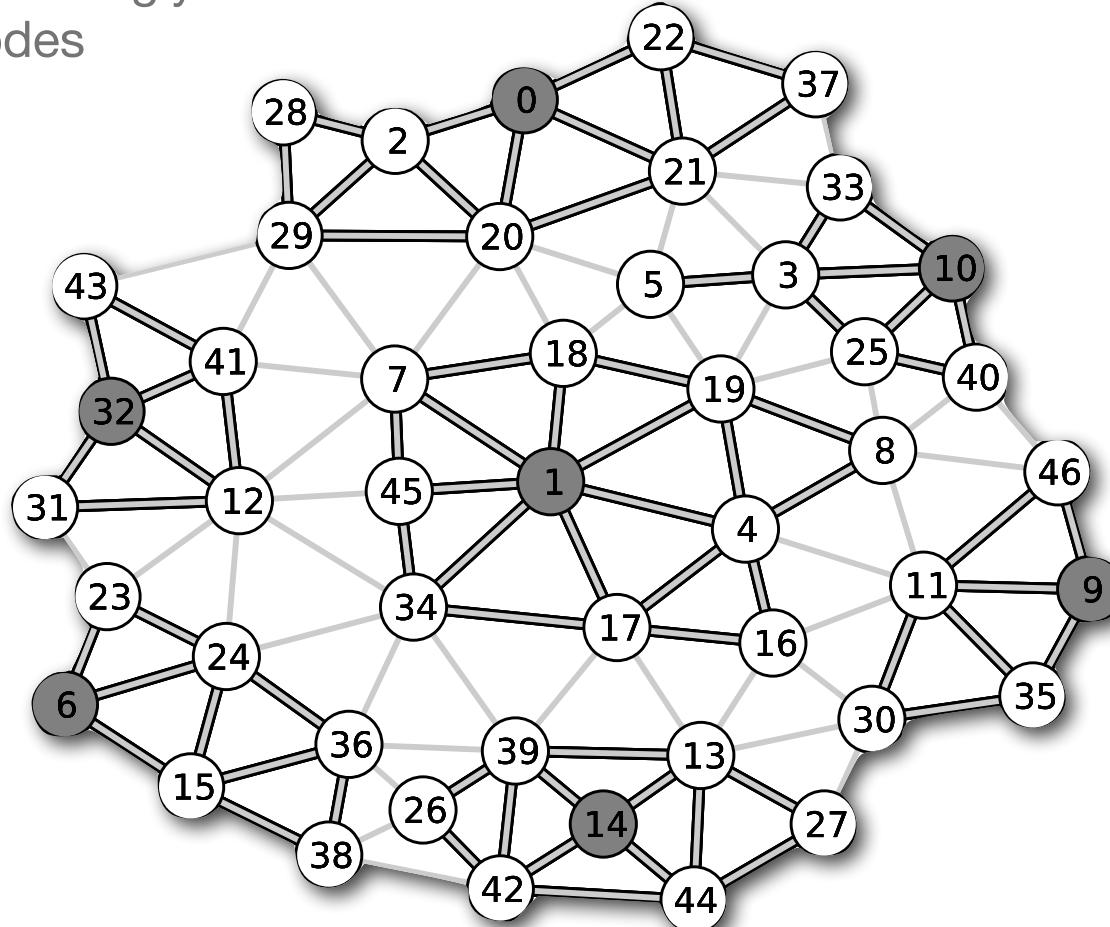
Aggregation



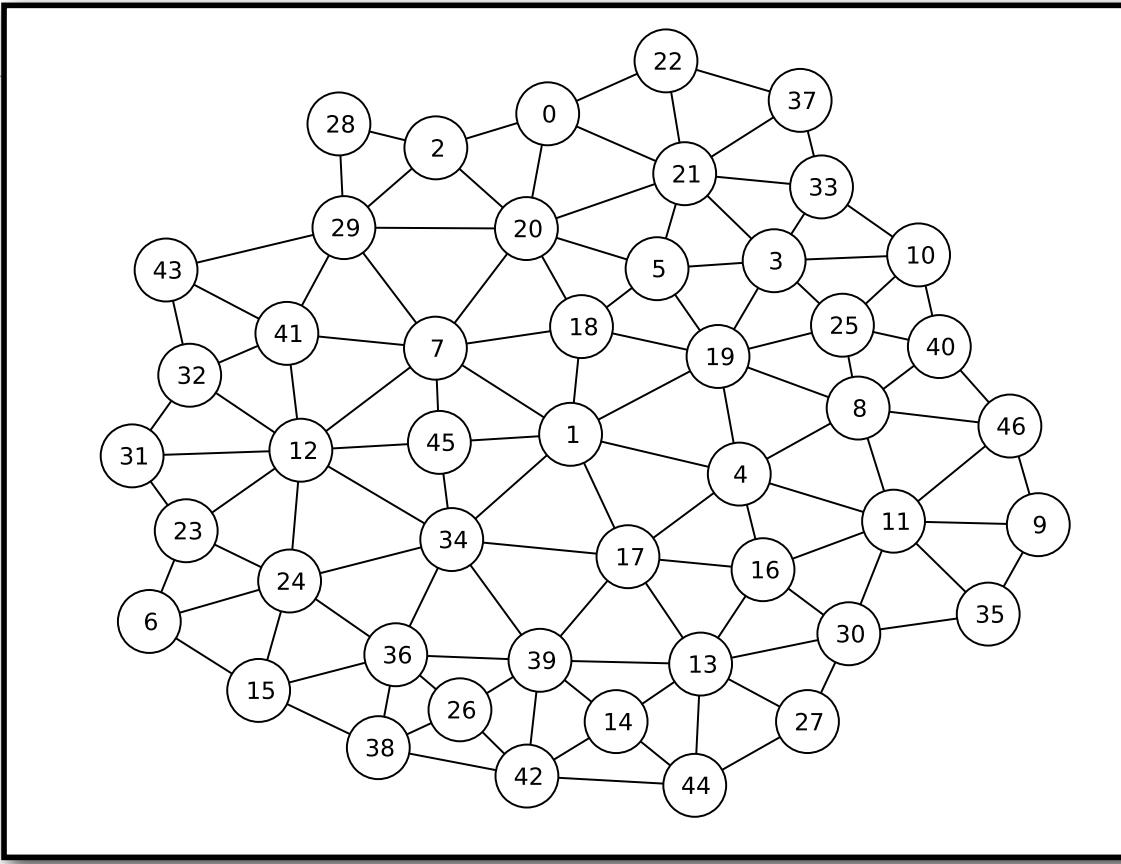
- Select next node
- If all strongly connected neighbors are unaggregated, then aggregate.

MIS-based Graph Aggregation

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



MIS(2)



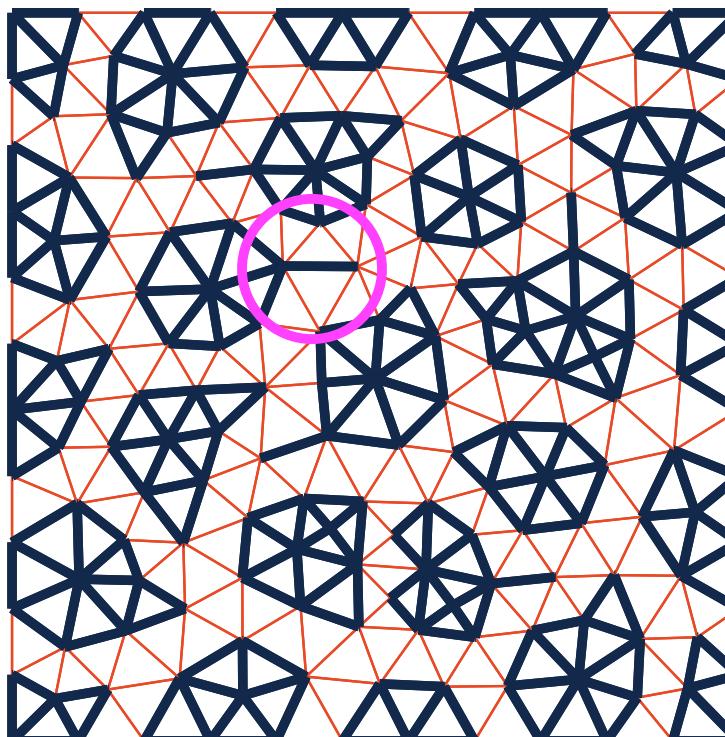
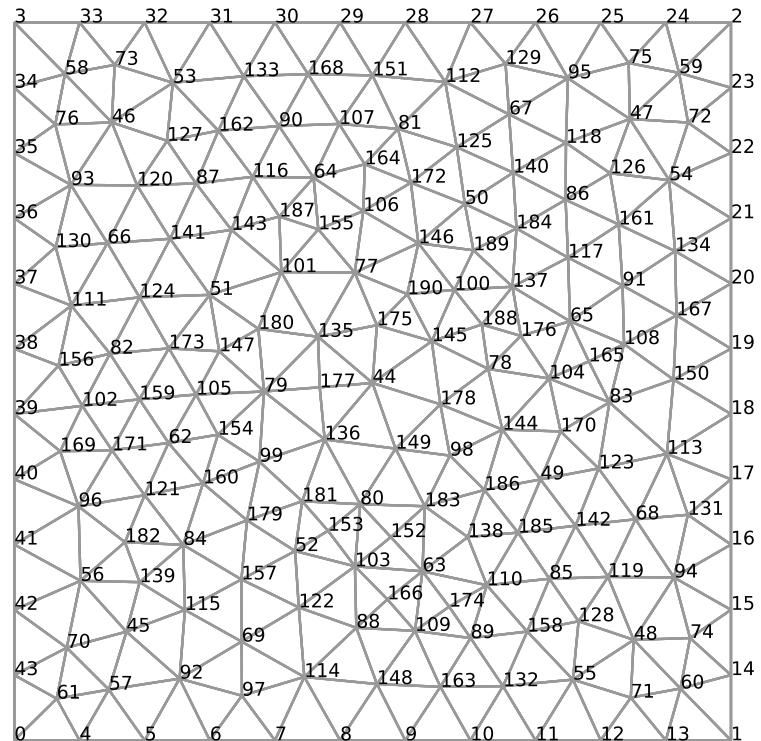
- root nodes more than 2 edges apart ($>$ distance-2)
- an unaggregated node more than 2 edges from a root can become a root

MIS(2)

independent

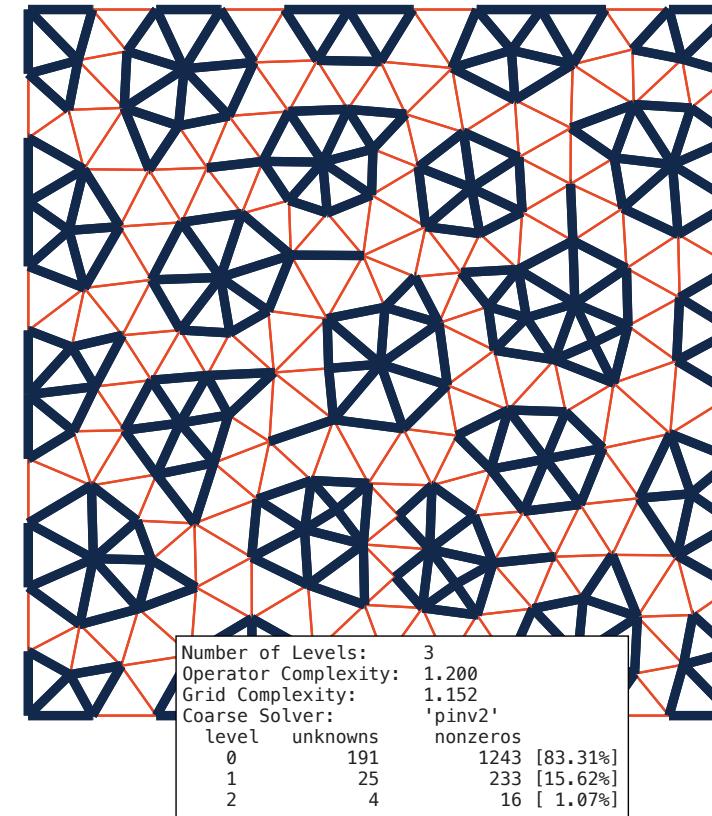
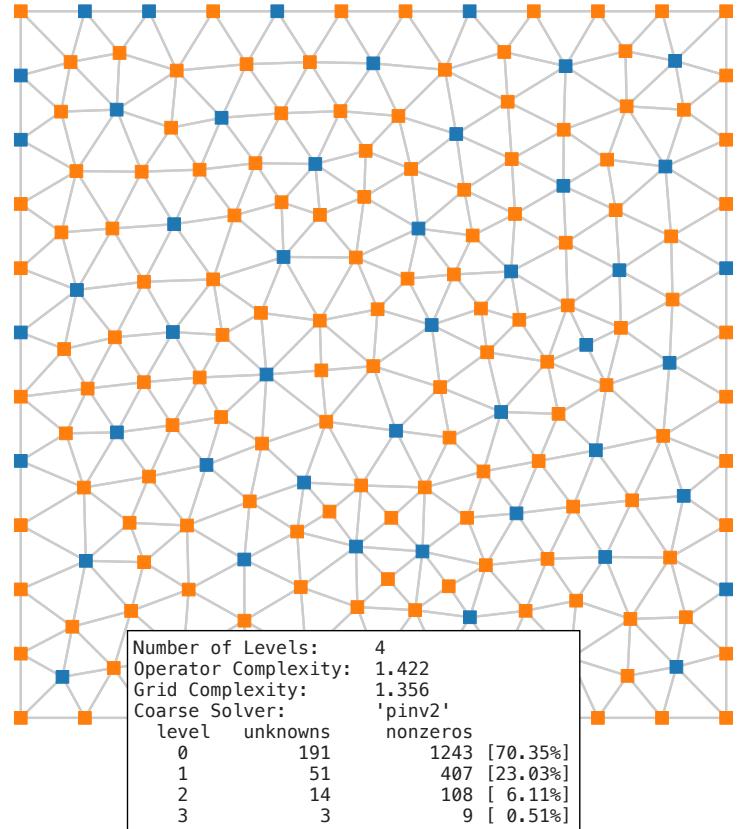
maximal

Aggregation



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

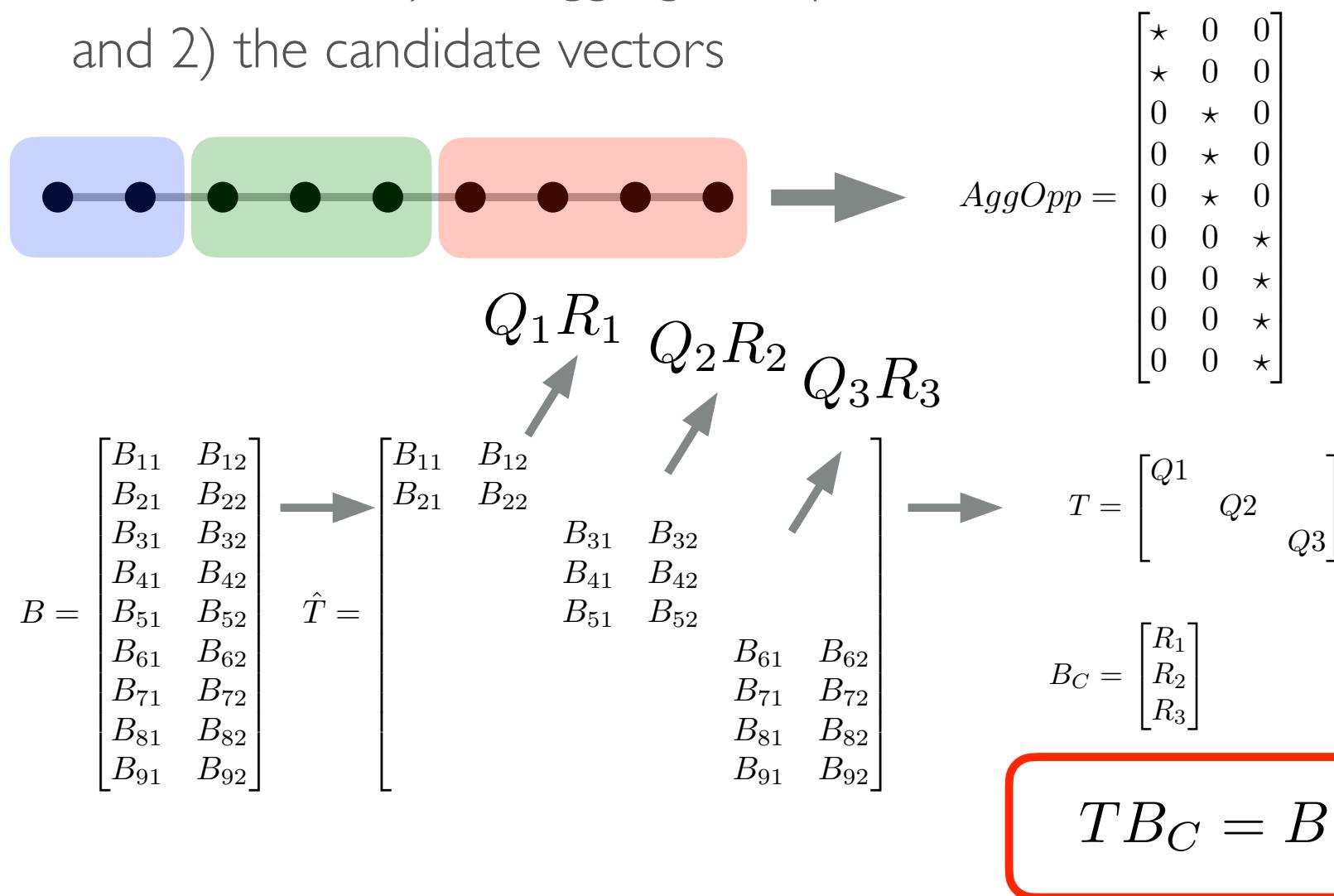
Aggregation vs splitting



- Aggregation generally more aggressive coarsening

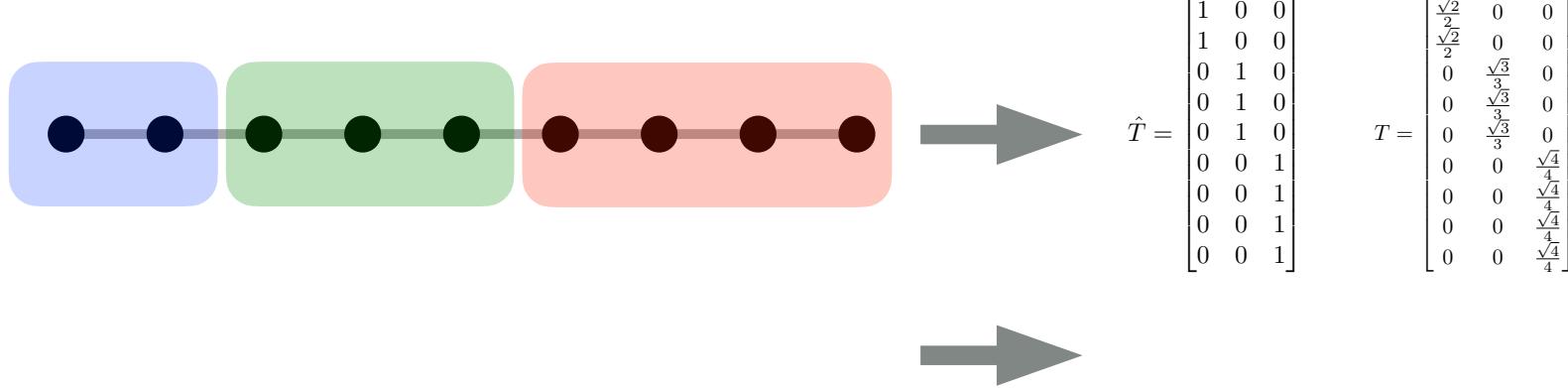
SA AMG Interpolation

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

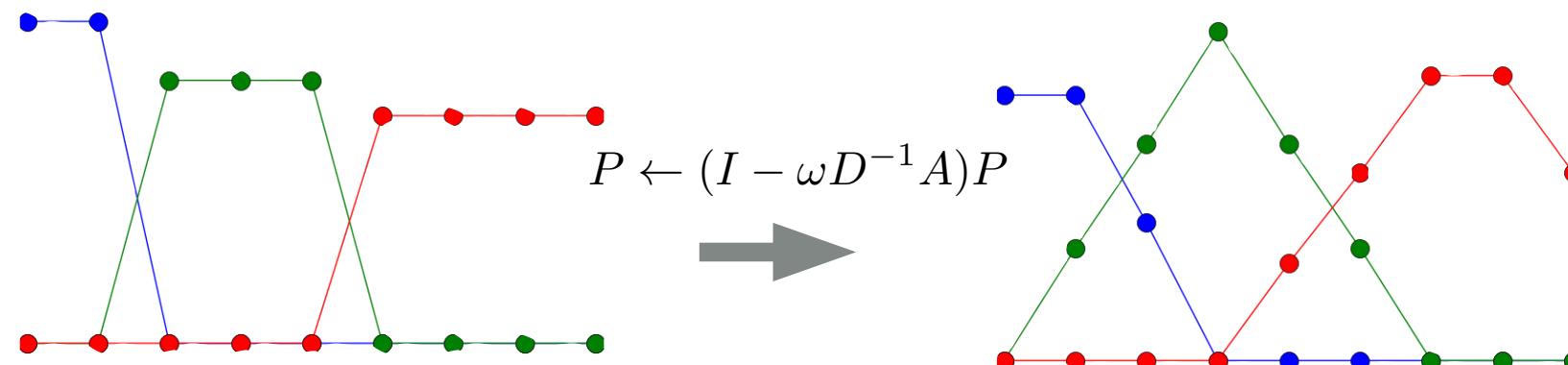


SA AMG Interpolation

- Example



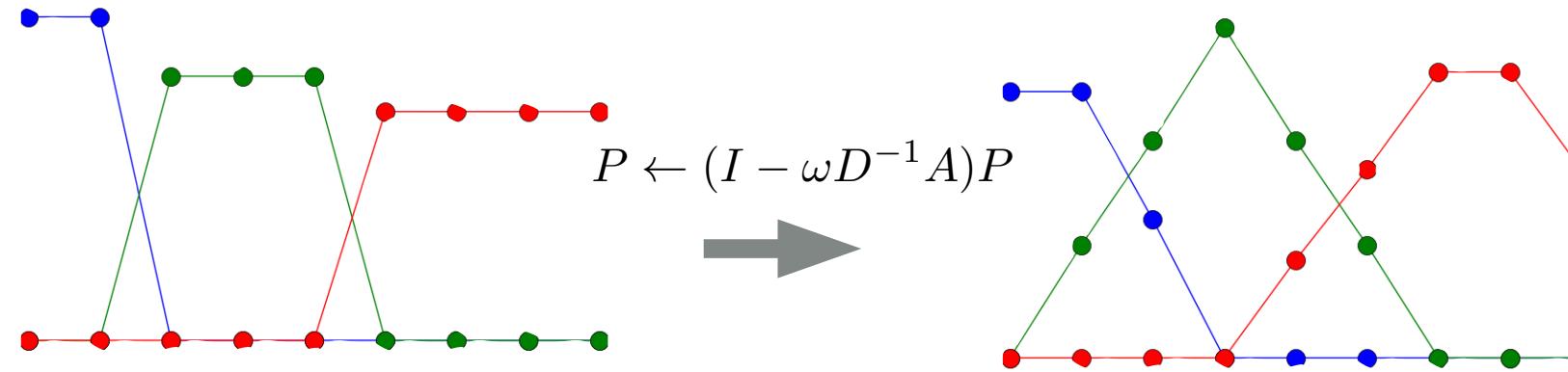
- Now make interpolation **better**



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

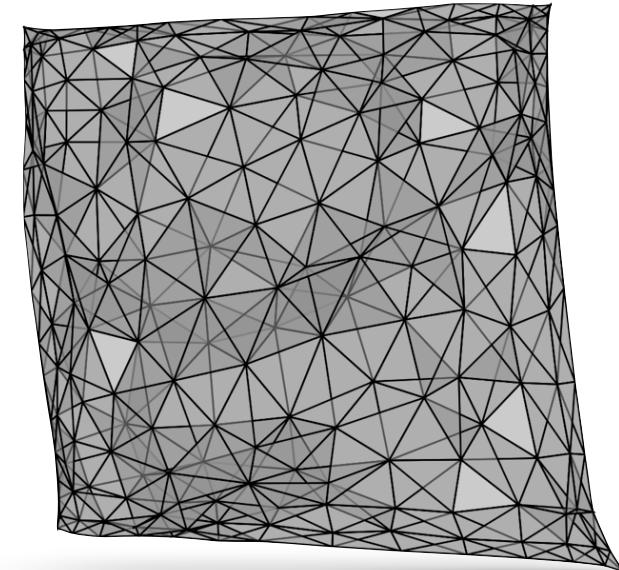
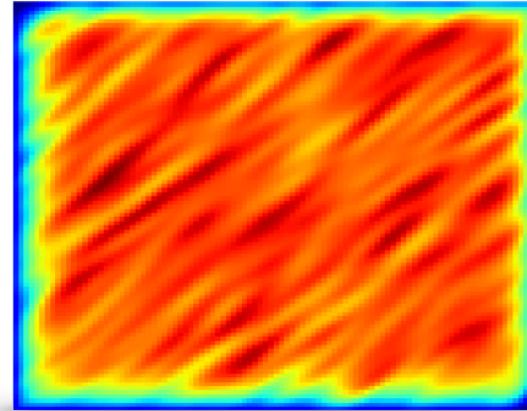
$S \leftarrow \text{strength}(A, B)$

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

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

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

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



Aggregation Based Setup

B

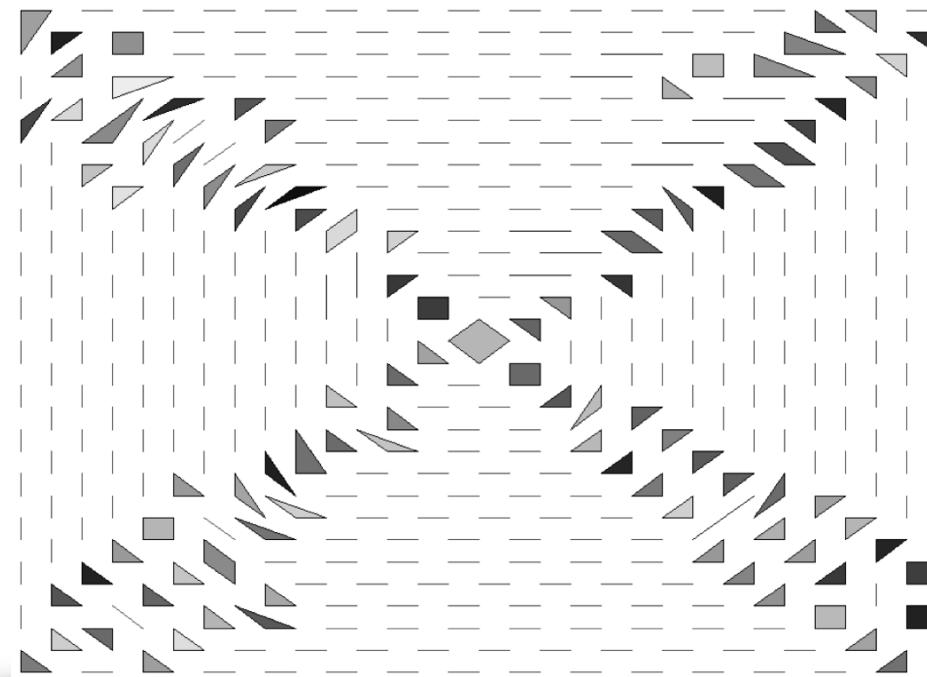
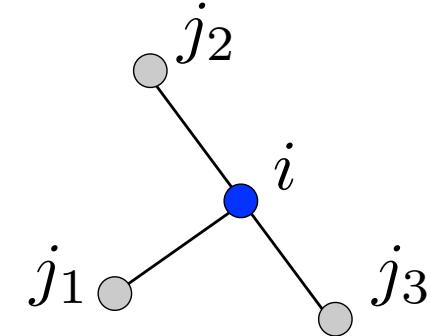
$S \leftarrow \text{strength}(A, B)$

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

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

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

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



Aggregation Based Setup

B

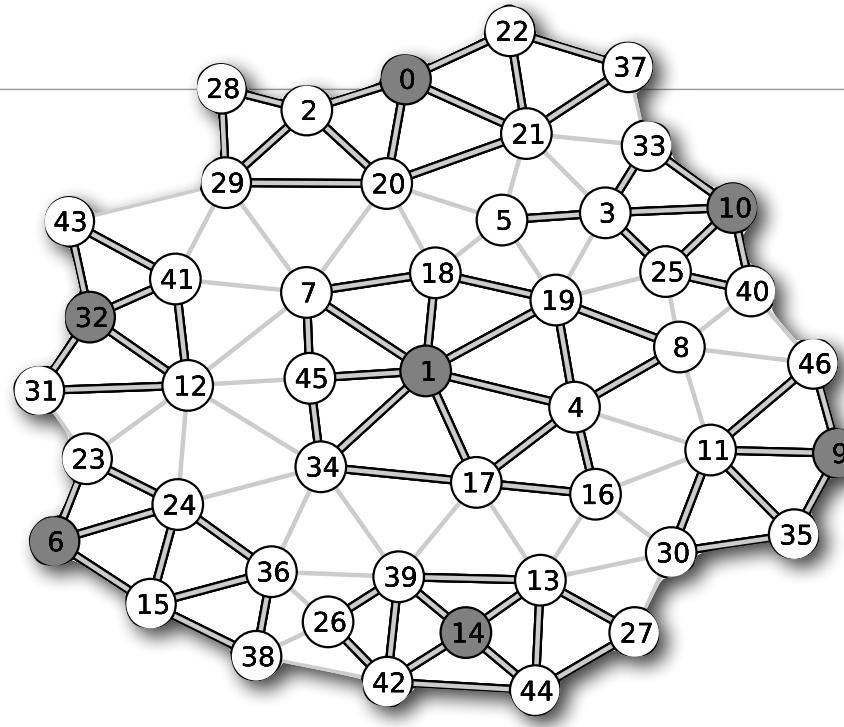
$S \leftarrow \text{strength}(A, B)$

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

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

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

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



- Group nodes
- based on strong connections
- variable size

Aggregation Based Setup

B

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

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

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

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

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

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

\downarrow

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

Aggregation Based Setup

B

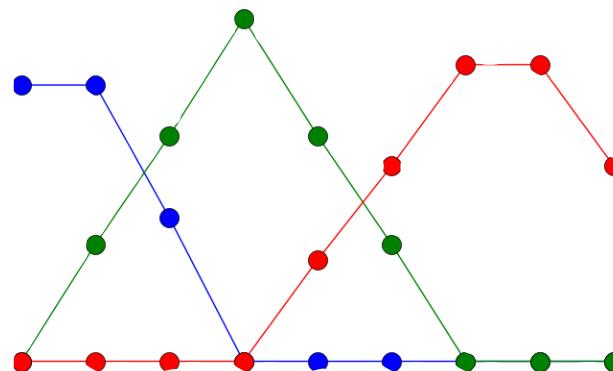
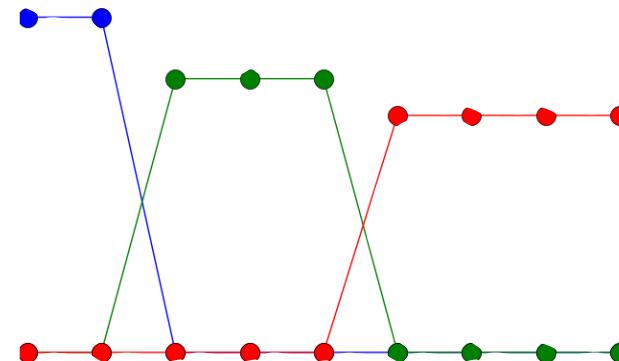
$$S \leftarrow \text{strength}(A, B)$$

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

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

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

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



SA AMG Setup Algorithm

Algorithm 1: SA_setup()

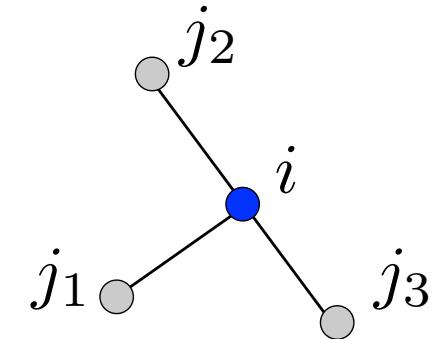
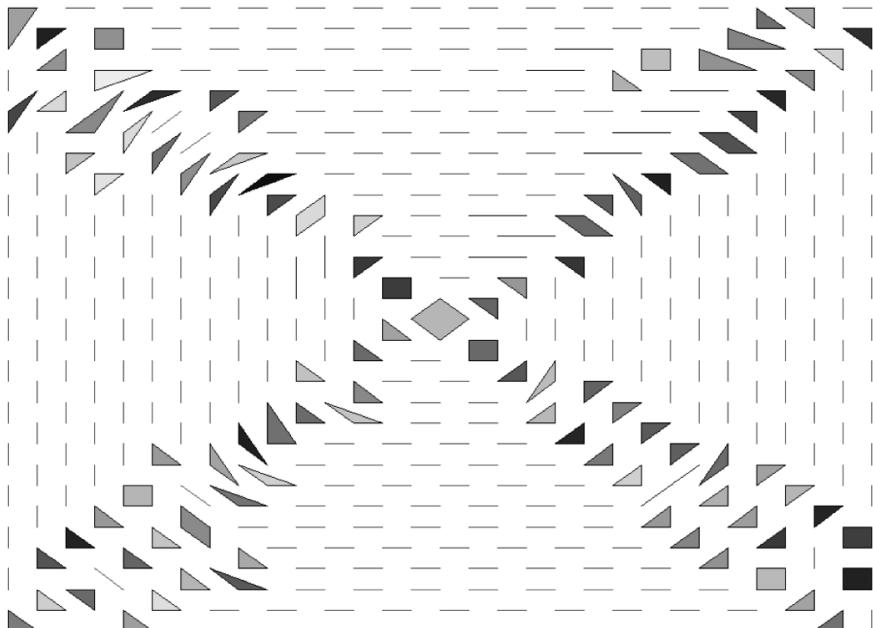
Input: A_0 : fine-grid operator
 B_0 : fine-grid candidate vectors
`max_size`: threshold for max size of coarsest problem

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

```
1  $\ell = 0$ 
2 while size( $A_\ell$ ) > max_size
3    $S_\ell = \text{strength}(A_\ell)$                                 {Strength-of-connection}
4    $\mathcal{A}_\ell = \text{aggregate}(S_\ell)$                       {Aggregation}
5    $T_\ell, B_{\ell+1} = \text{inject}(\mathcal{A}_\ell, B_\ell)$     {Form tentative interpolation and coarse candidates}
6    $P_\ell = \text{smooth}(A_\ell, T_\ell)$                       {Smooth  $T_\ell$ }
7    $A_{\ell+1} = P_\ell^T A_\ell P_\ell$                           {Coarse-grid operator}
8    $\ell = \ell + 1$ 
```

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

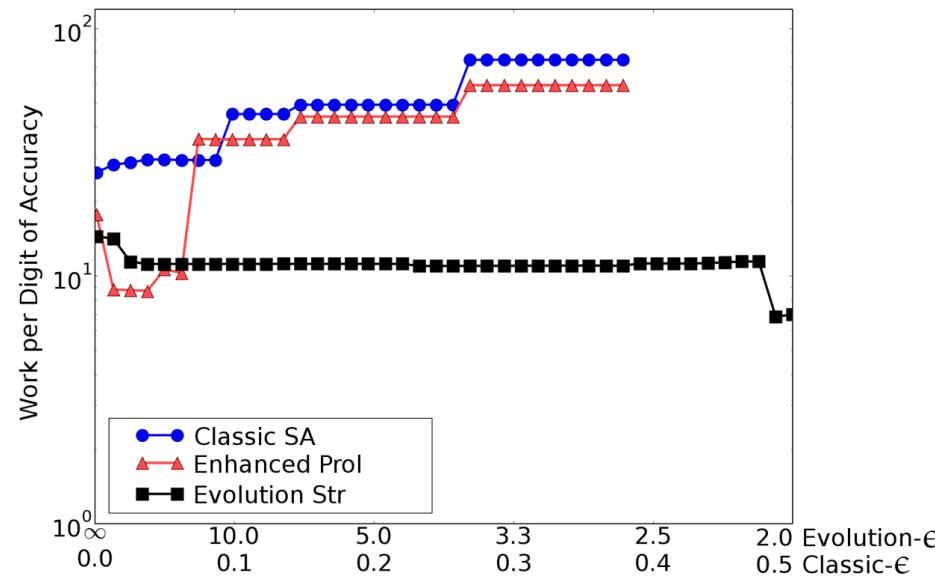


- efficient
- parameter insensitive
- Euler flow
- wave problems
- high-order
- discontinuous elements

Luke N. Olson , Jacob Schroder , Raymond S. Tuminaro, A new perspective on strength measures in algebraic multigrid, NLA, 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		with P_{opt}		Evol
	0	$\frac{1}{4}$	0	$\frac{1}{4}$	4.00
Linear Tets					
$p = 1$	h	18	48	15	27
	$h/2$	17	74	14	62
	$h/4$	26	130	20	92
$p = 2$	h	41	85	24	46
	$h/2$	21	44	26	33
	$h/4$	45	88	67	67
Quad Tets					
$p = 1$	h	63	148	129	129
	$h/2$	49	86	61	61
	$h/4$	49	86	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

$$G e_0 \in \mathcal{R}(P) \quad \Rightarrow \quad e_1 = 0$$

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

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

- Non-Hermitian: use GMRES $R = P^*$

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

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

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

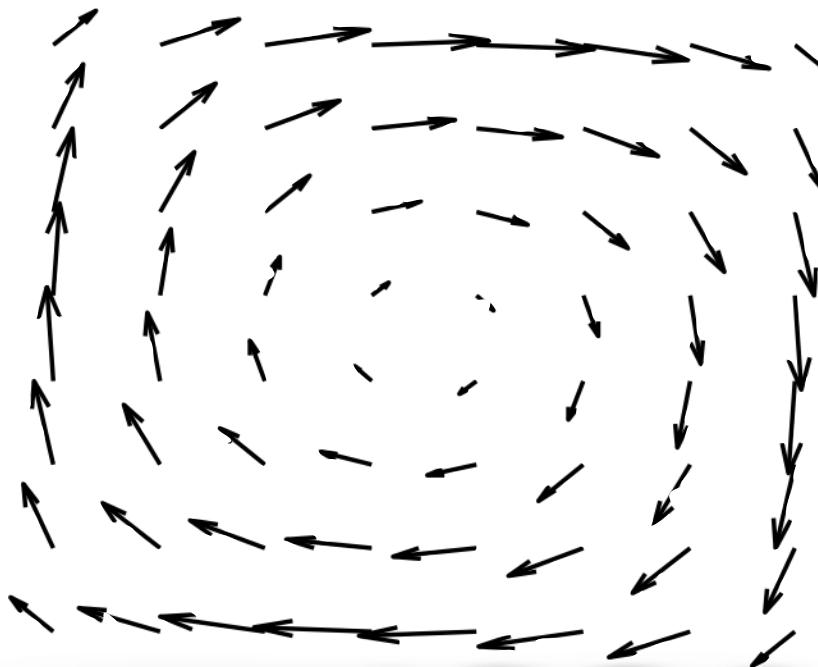
W. L. Wan, T. F. Chan, and B. Smith, An energy-minimizing interpolation for robust multi-grid methods, SIAM J. Sci. Comput., 2000

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