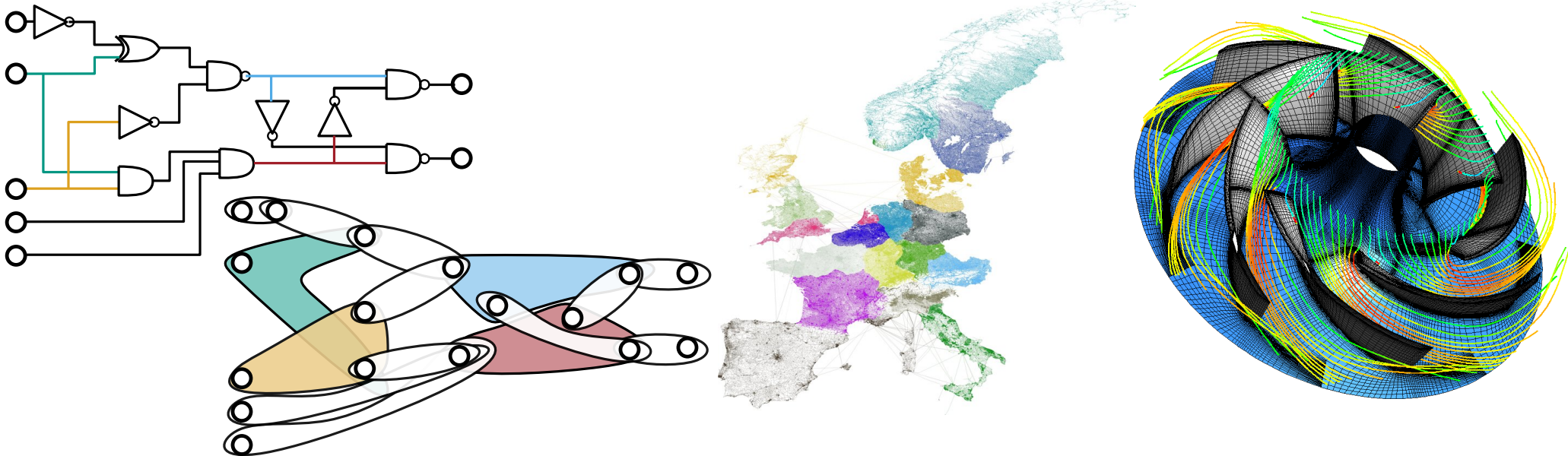


# High Quality Graph and Hypergraph Partitioning

2nd BMBF Big Data All Hands Meeting · October 11, 2017

Yaroslav Akhremtsev, Peter Sanders, Sebastian Schlag, Christian Schulz

INSTITUTE OF THEORETICAL INFORMATICS · ALGORITHMICS GROUP

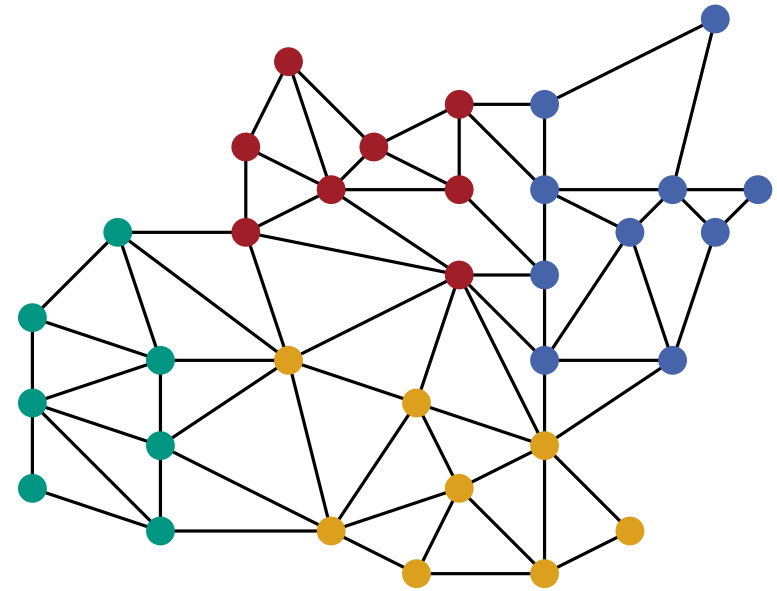


# Graphs and Hypergraphs

Graph  $G = (V, E)$

vertices  edges

- models **relationships** between **objects**
- dyadic (**2-ary**) relationships

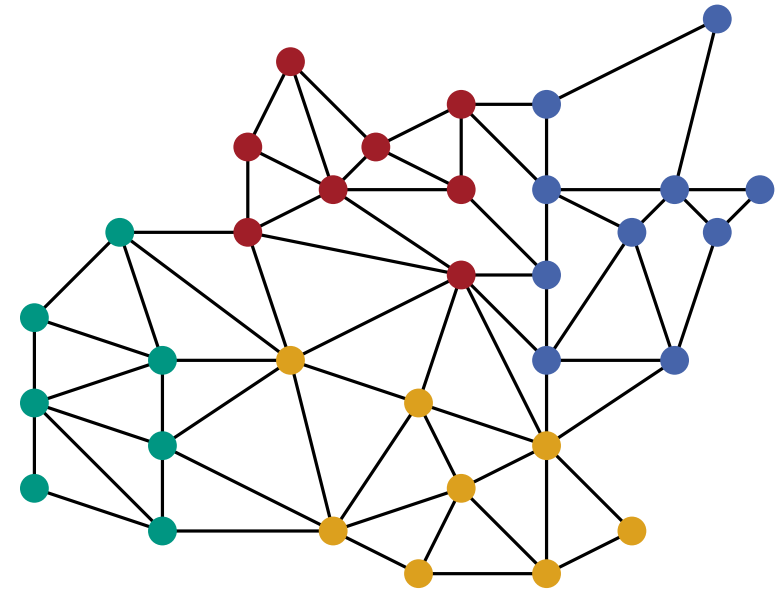


# Graphs and Hypergraphs

**Graph**  $G = (V, E)$

vertices  edges 

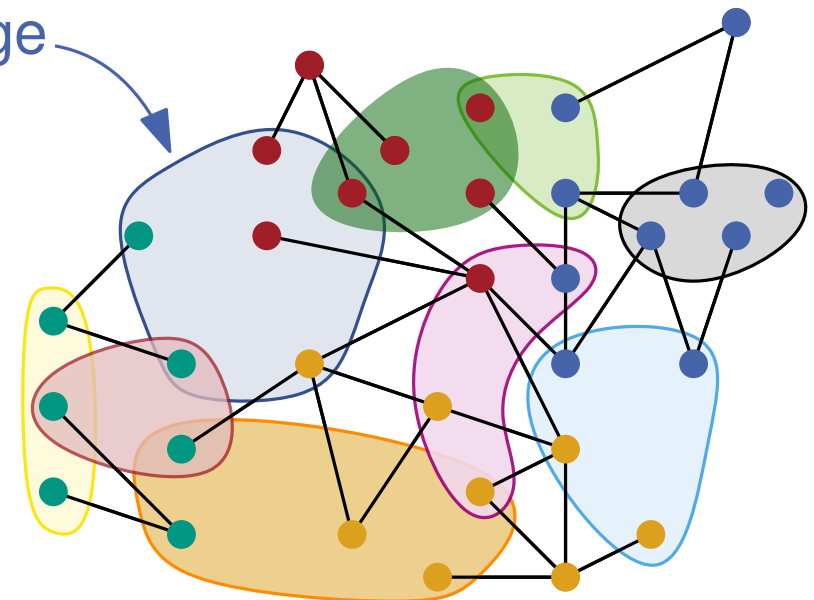
- models **relationships** between **objects**
- dyadic (**2-ary**) relationships



**Hypergraph**  $H = (V, E)$

- Generalization of a graph  
 $\Rightarrow$  hyperedges connect  $\geq 2$  nodes
- arbitrary (**d-ary**) relationships
- Edge set  $E \subseteq \mathcal{P}(V) \setminus \emptyset$

hyperedge 



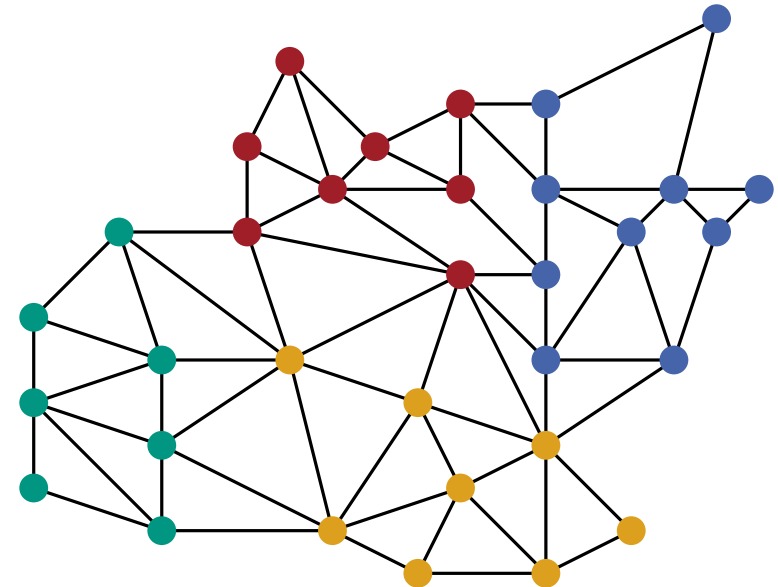
# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$   
into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

- **objective** function on edges is **minimized**



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

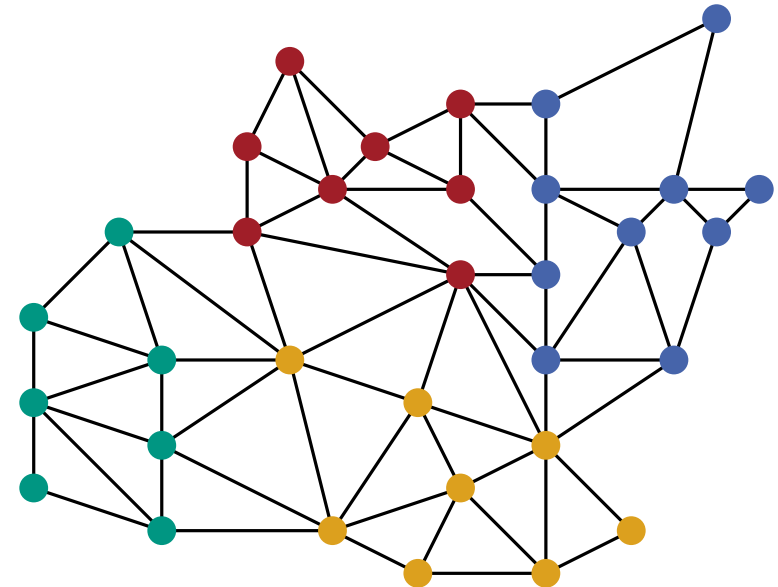
**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$   
into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

imbalance parameter

- **objective** function on edges is **minimized**



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

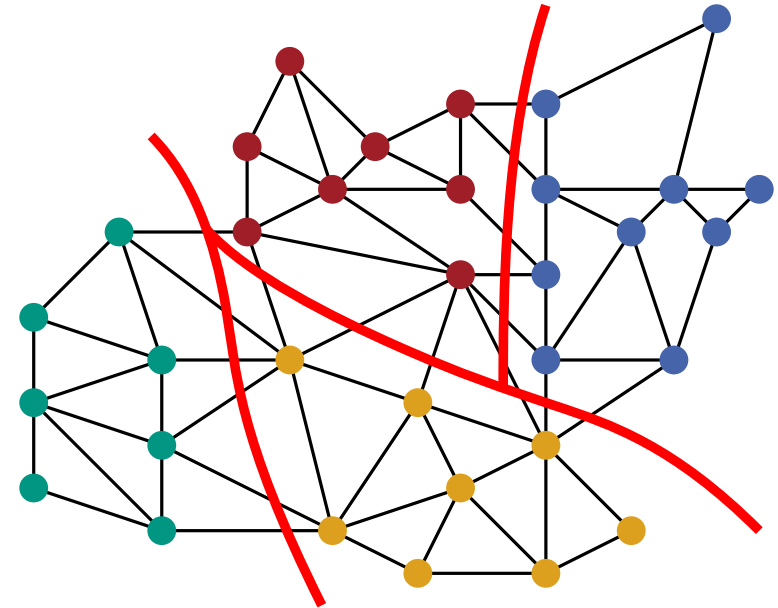
imbalance parameter

- **objective** function on edges is **minimized**

## Common Objectives:

- Graphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e)$



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

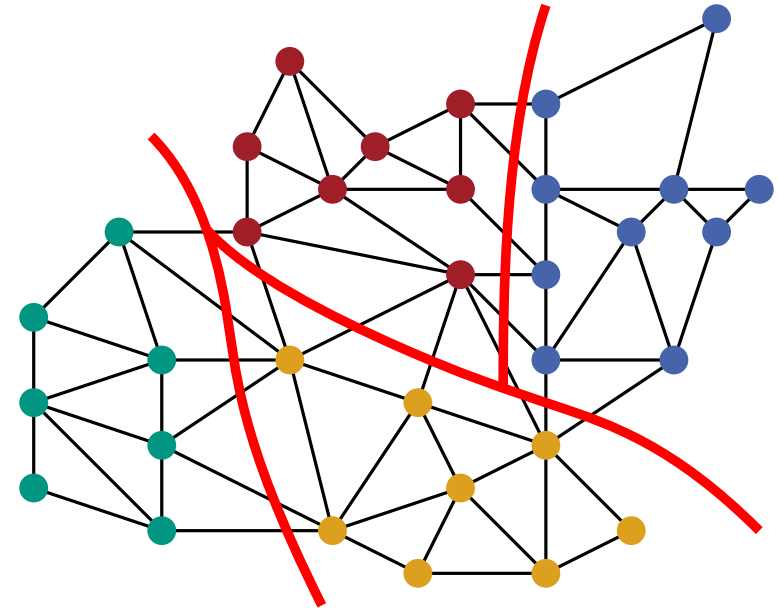
imbalance parameter

- **objective** function on edges is **minimized**

## Common Objectives:

- Graphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 17$



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

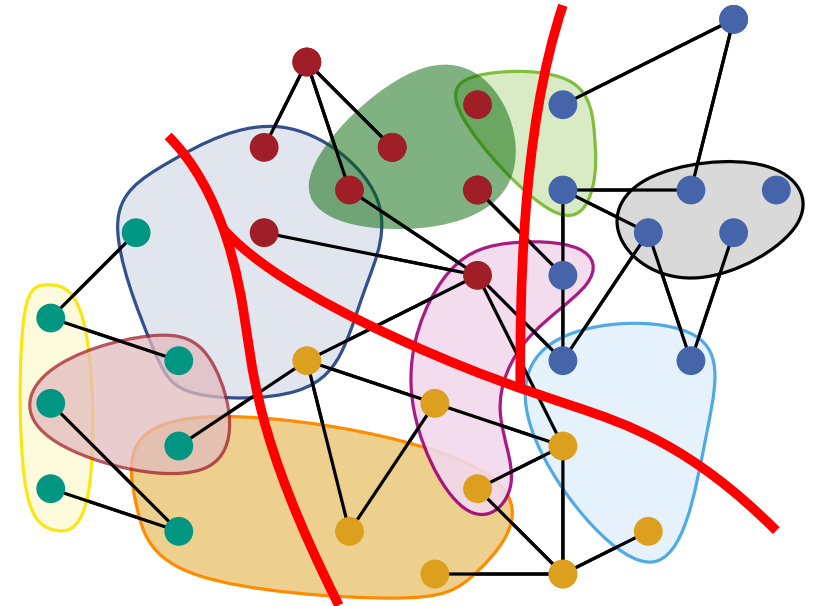
imbalance parameter

- **objective** function on edges is **minimized**

## Common Objectives:

- Graphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 17$





# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

imbalance parameter

- **objective** function on edges is **minimized**

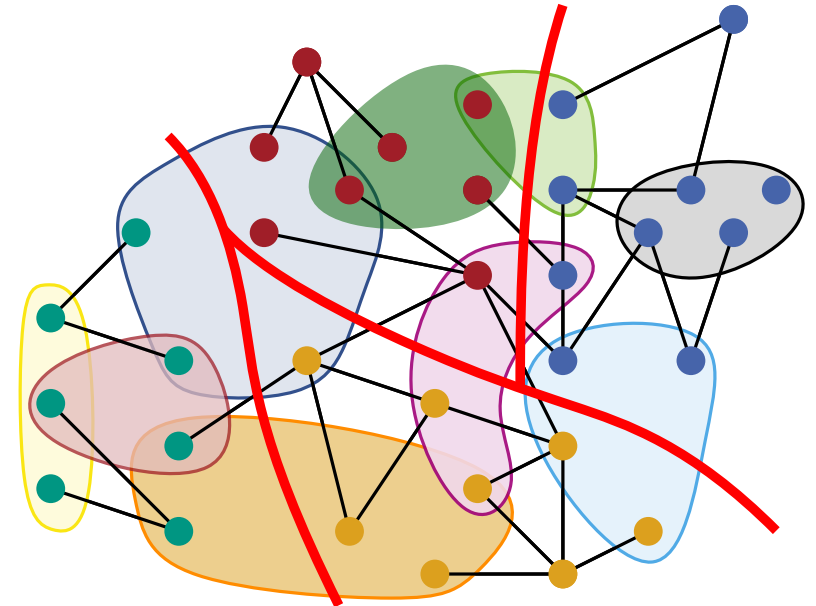
## Common Objectives:

- Graphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 17$

- Hypergraphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e)$



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

imbalance parameter

- **objective** function on edges is **minimized**

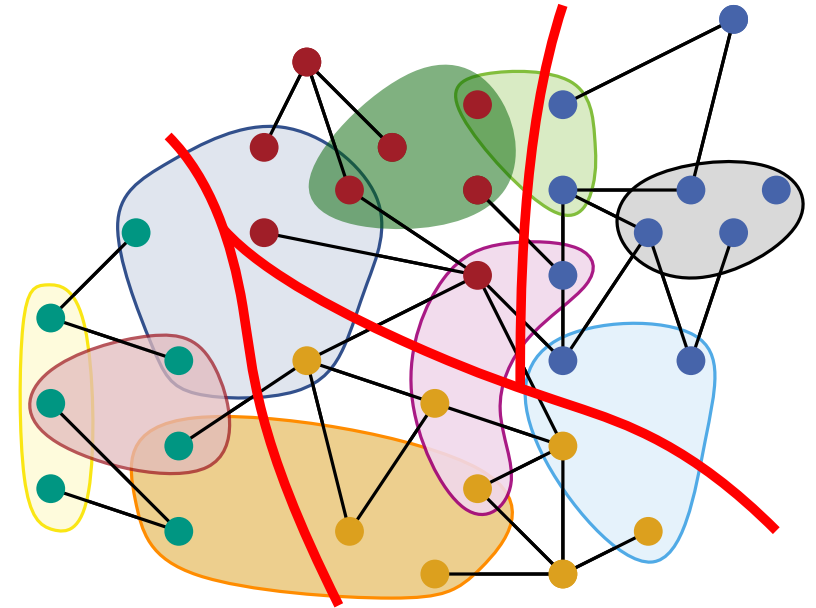
## Common Objectives:

- Graphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 17$

- Hypergraphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 5$



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

imbalance parameter

- **objective** function on edges is **minimized**

## Common Objectives:

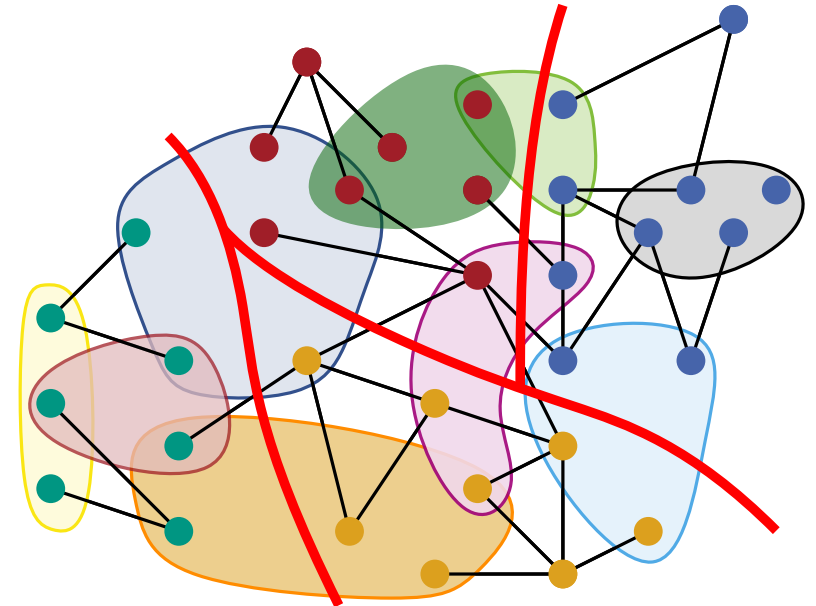
- Graphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 17$

- Hypergraphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 5$

- **connectivity**:  $\sum_{e \in \text{cut}} (\lambda - 1) \omega(e)$



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

imbalance parameter

- **objective** function on edges is **minimized**

## Common Objectives:

- Graphs:

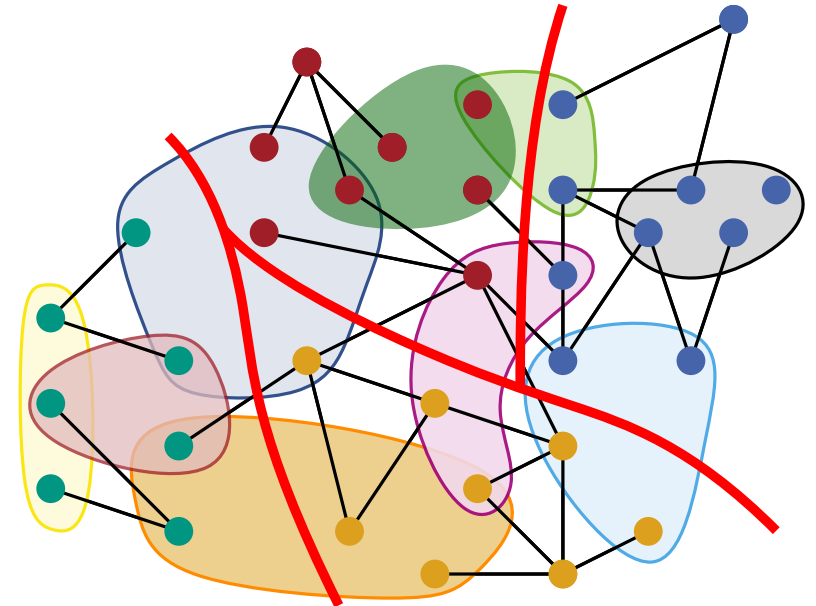
- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 17$

- Hypergraphs:

- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 5$

- **connectivity**:  $\sum_{e \in \text{cut}} (\lambda - 1) \omega(e)$

# blocks connected by  $e$



# $\varepsilon$ -Balanced Graph and Hypergraph Partitioning

**Partition** (hyper)graph  $G = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$  into  $k$  disjoint blocks  $V_1, \dots, V_k$  s.t.

- blocks  $V_i$  are **roughly equal-sized**:

$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

imbalance parameter

- **objective** function on edges is **minimized**

## Common Objectives:

- Graphs:

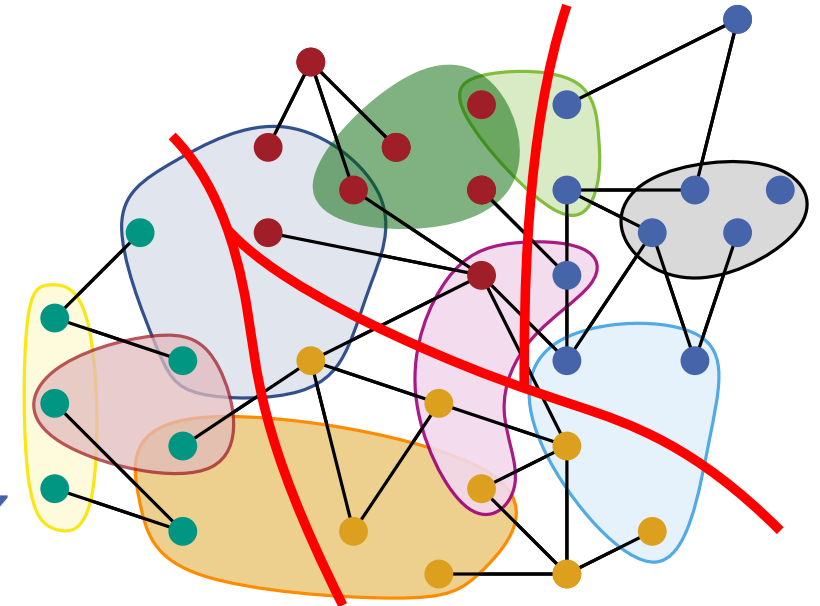
- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 17$

- Hypergraphs:

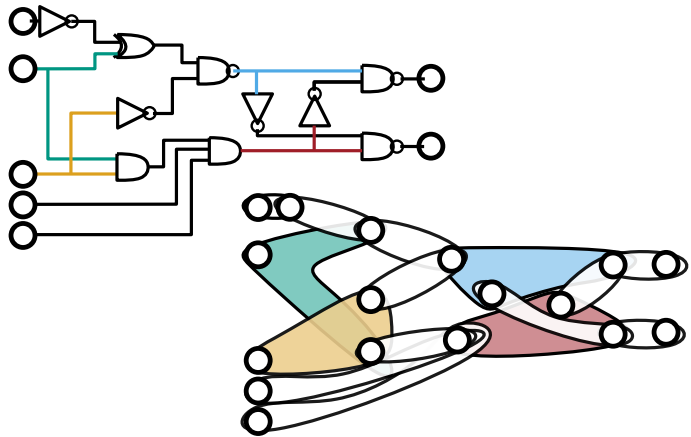
- **cut**:  $\sum_{e \in \text{cut}} \omega(e) = 5$

- **connectivity**:  $\sum_{e \in \text{cut}} (\lambda - 1) \omega(e) = 7$

# blocks connected by  $e$



# Applications



**VLSI Design**



**Warehouse Optimization**

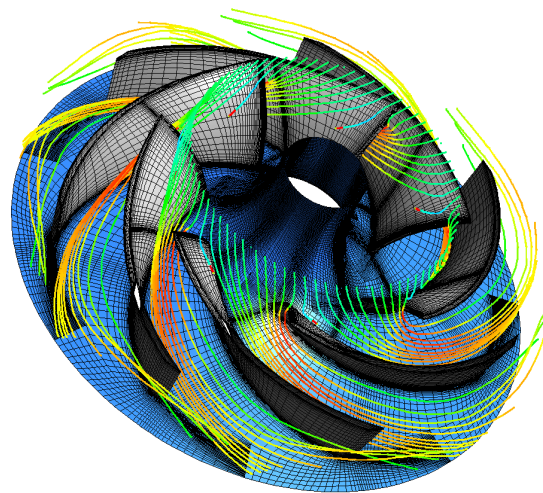
[Martin Grandjean, via Wikimedia Commons]



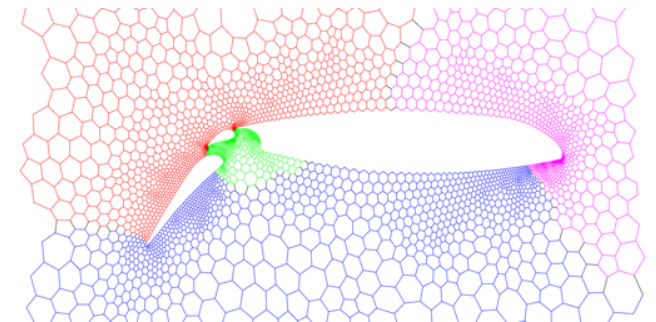
**Complex Networks**



**Route Planning**

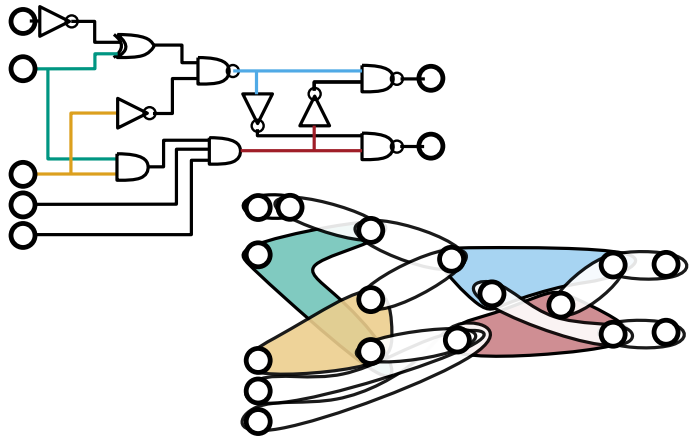


**Simulation**



$\mathbb{R}^{n \times n} \ni Ax = b \in \mathbb{R}^n$   
**Scientific Computing**

# Applications



VLSI Design



Warehouse Optimization

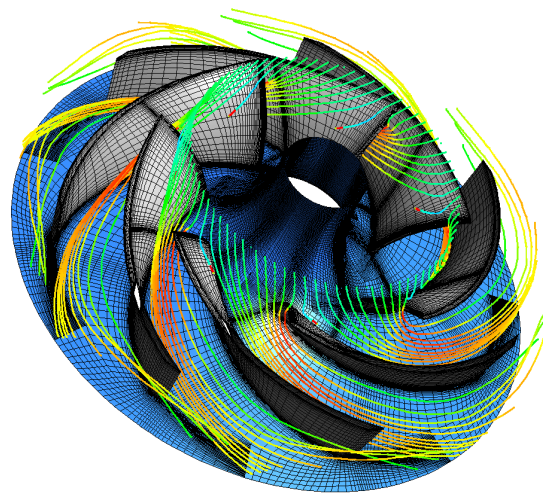
[Martin Grandjean, via Wikimedia Commons]



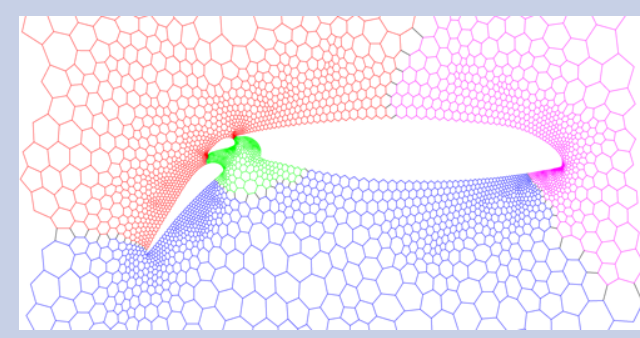
Complex Networks



Route Planning



Simulation



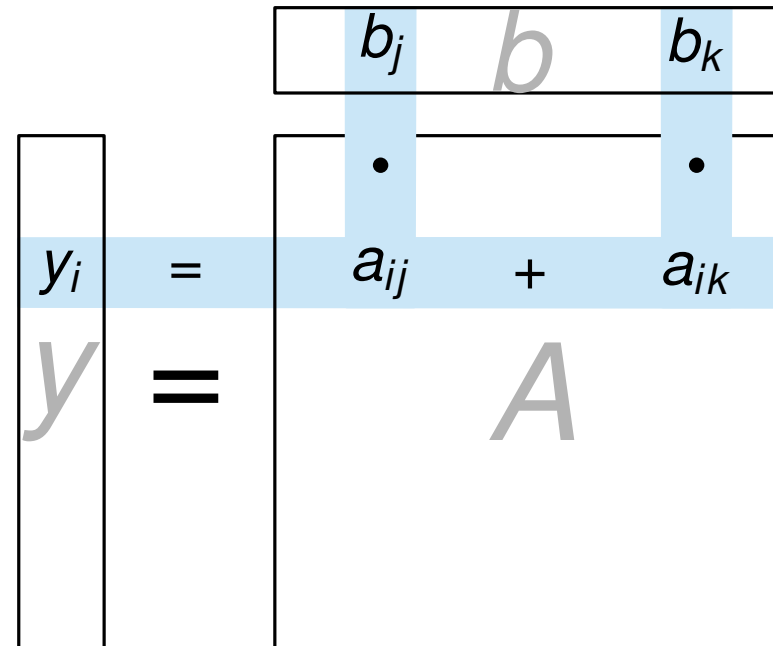
A visualization of a scientific computing problem. It shows a 2D domain with a grid of nodes and edges. The domain is colored with a gradient from blue to red, and there is a central white region. The grid is denser in some areas and sparser in others, representing a mesh refinement or adaptive meshing technique.

$$\mathbb{R}^{n \times n} \ni Ax = b \in \mathbb{R}^n$$

Scientific Computing

# Parallel Sparse-Matrix Vector Product ( $\text{SpM} \times \text{V}$ )

$$y = A b$$



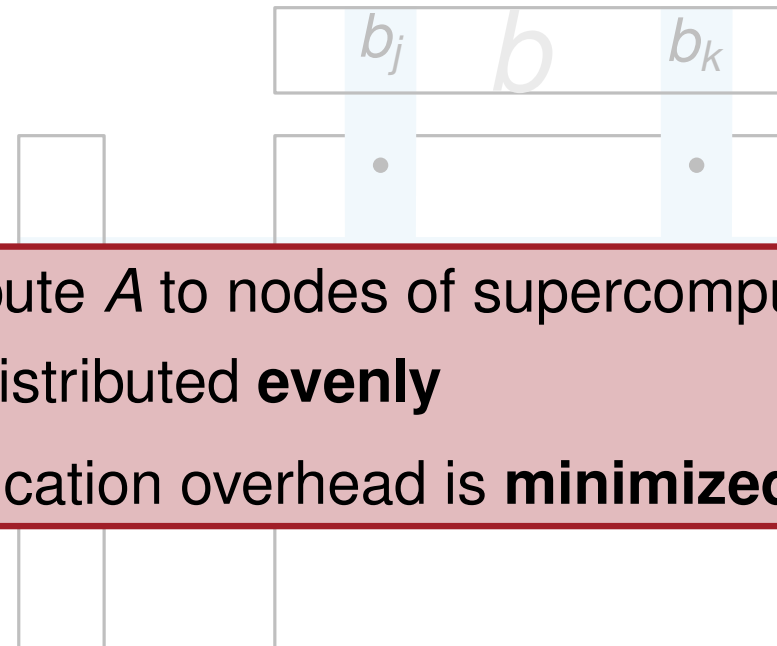
## Setting:

- repeated  $\text{SpM} \times \text{V}$  on supercomputer
- $A$  is large  $\Rightarrow$  distribute on multiple nodes
- symmetric partitioning  $\Rightarrow y$  &  $b$  divided conformally with  $A$



# Parallel Sparse-Matrix Vector Product ( $\text{SpM} \times \text{V}$ )

$$y = A b$$



**Task:** distribute  $A$  to nodes of supercomputer such that

- work is distributed **evenly**
- communication overhead is **minimized**

## Setting:

- repeated  $\text{SpM} \times \text{V}$  on supercomputer
- $A$  is large  $\Rightarrow$  distribute on multiple nodes
- symmetric partitioning  $\Rightarrow y$  &  $b$  divided conformally with  $A$

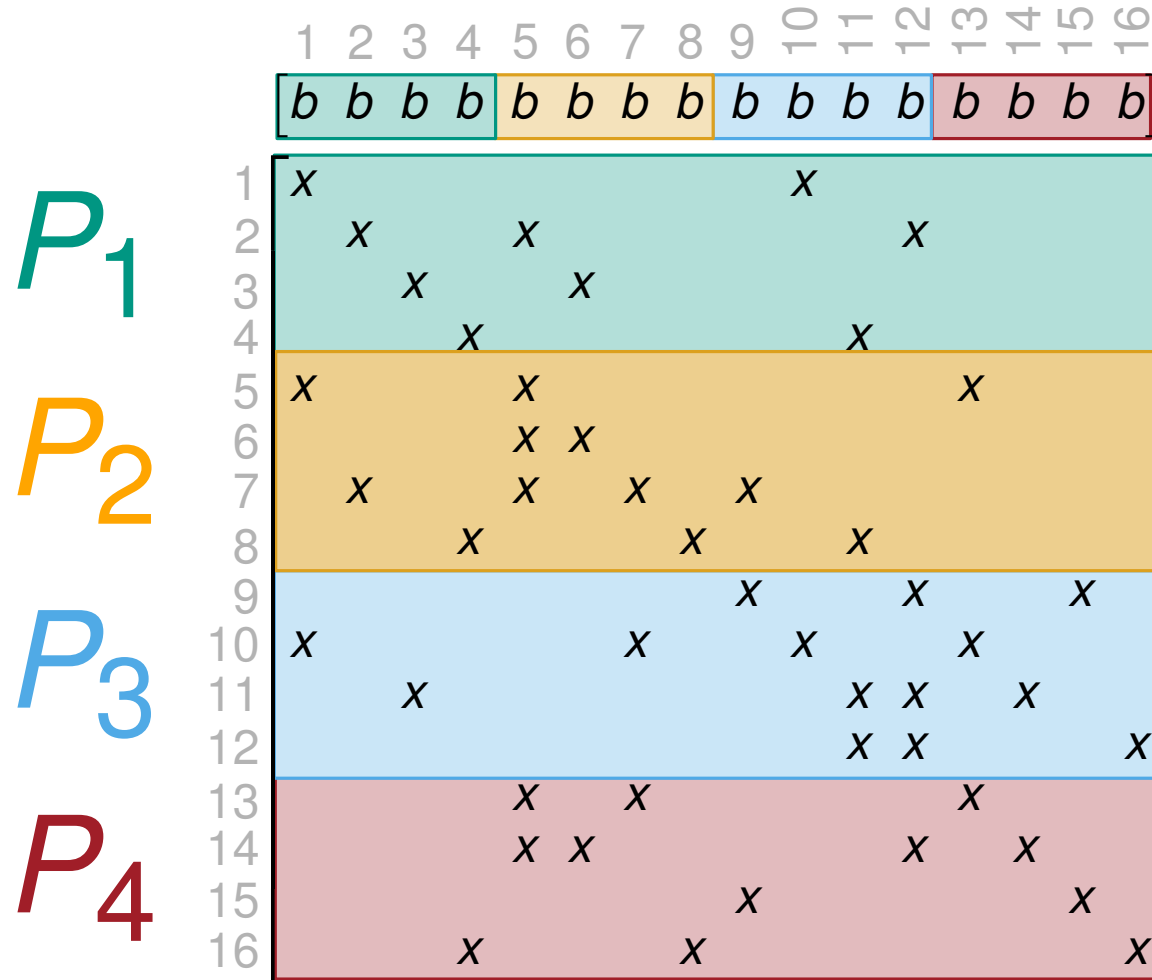
# Naive Approach: Rowwise Decomposition

$$A \in \mathbb{R}^{16 \times 16}$$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	[ b b b b b b b b b b b b b b b ]															
1	x									x						
2		x			x							x				
3			x			x										
4				x							x					
5	x				x								x			
6					x	x										
7		x			x		x		x							
8				x				x			x					
9									x			x			x	
10	x						x			x			x			
11			x								x	x		x		
12											x	x				x
13					x		x						x			
14					x	x						x		x		
15									x						x	
16				x				x								x

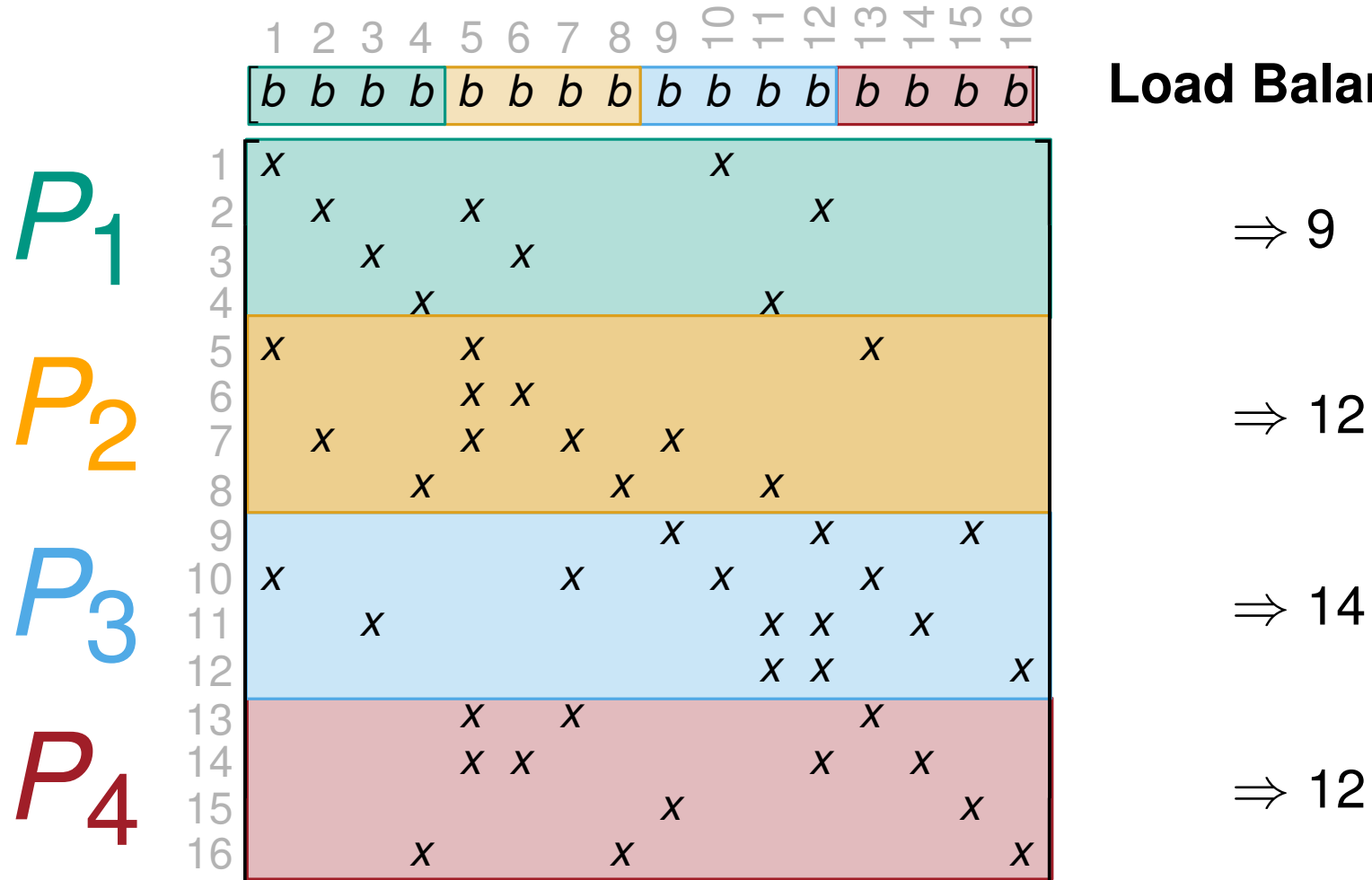
# Naive Approach: Rowwise Decomposition

$$A \in \mathbb{R}^{16 \times 16}$$



# Naive Approach: Rowwise Decomposition

$$A \in \mathbb{R}^{16 \times 16}$$



**Load Balancing?**

⇒ 9

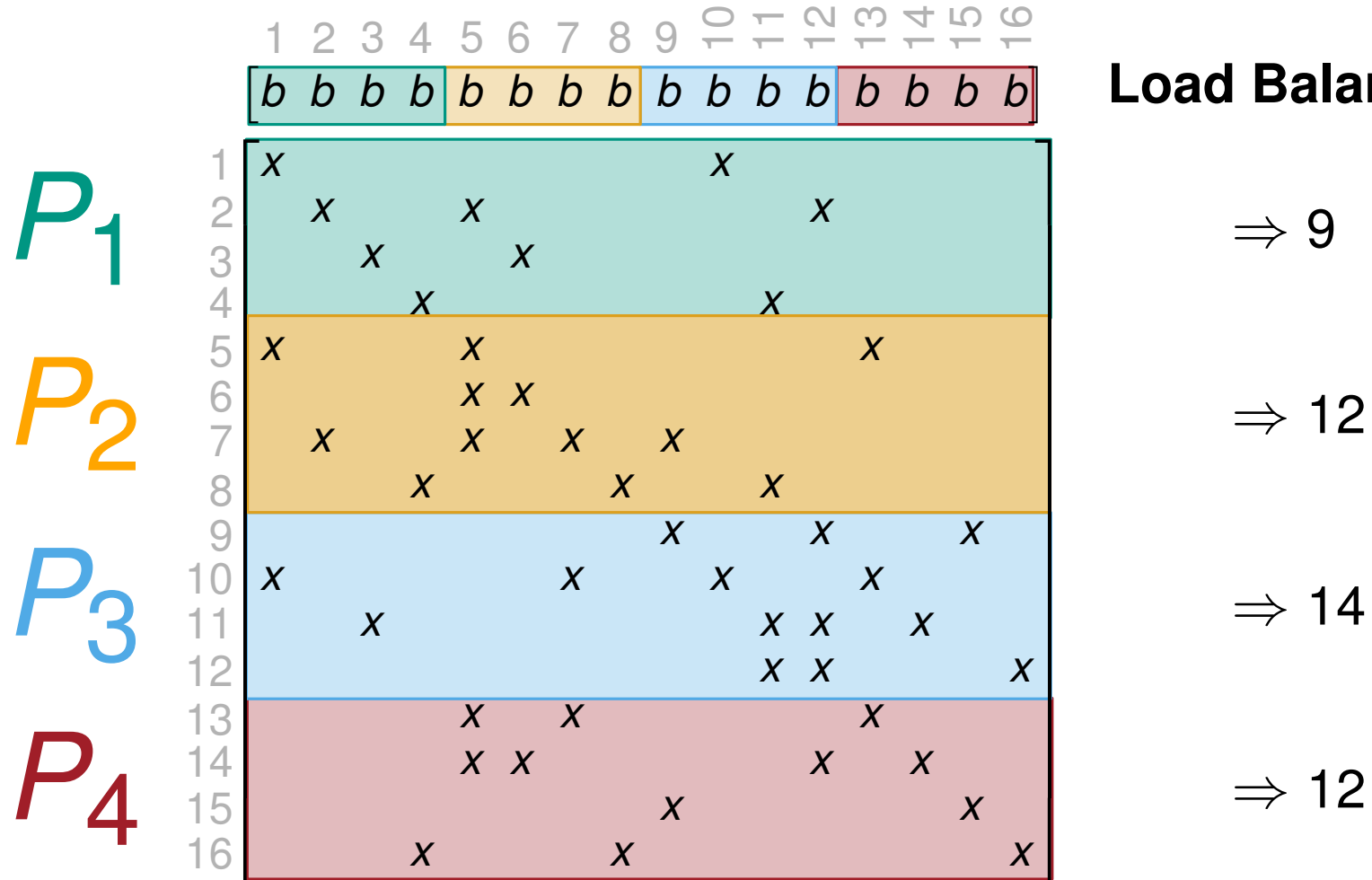
⇒ 12

⇒ 14

⇒ 12

# Naive Approach: Rowwise Decomposition

$$A \in \mathbb{R}^{16 \times 16}$$



**Load Balancing?**

⇒ 9

⇒ 12

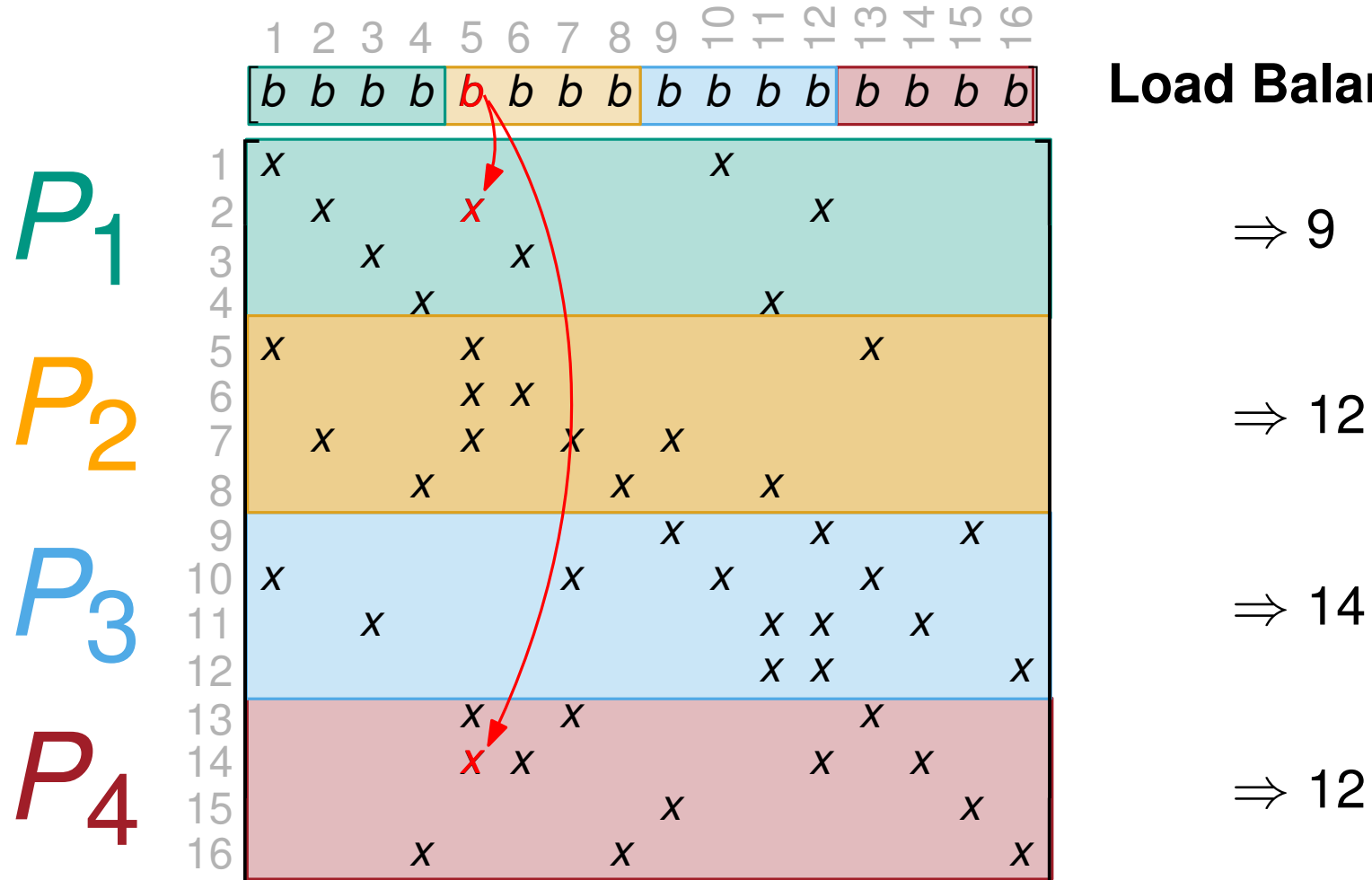
⇒ 14

⇒ 12

**Communication Volume?**

# Naive Approach: Rowwise Decomposition

$$A \in \mathbb{R}^{16 \times 16}$$



**Load Balancing?**

⇒ 9

⇒ 12

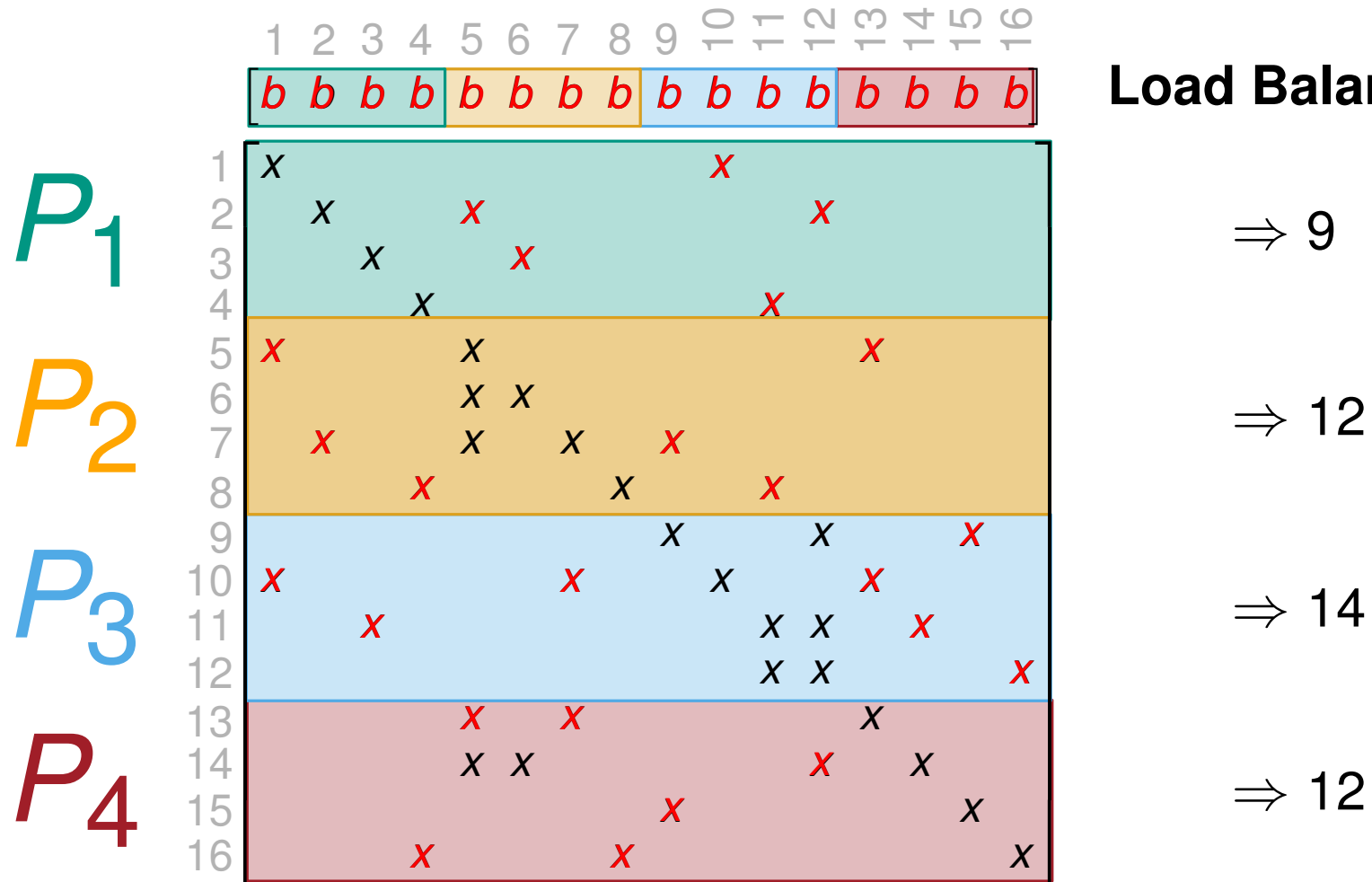
⇒ 14

⇒ 12

**Communication Volume?**

# Naive Approach: Rowwise Decomposition

$$A \in \mathbb{R}^{16 \times 16}$$



**Load Balancing?**

$\Rightarrow 9$

$\Rightarrow 12$

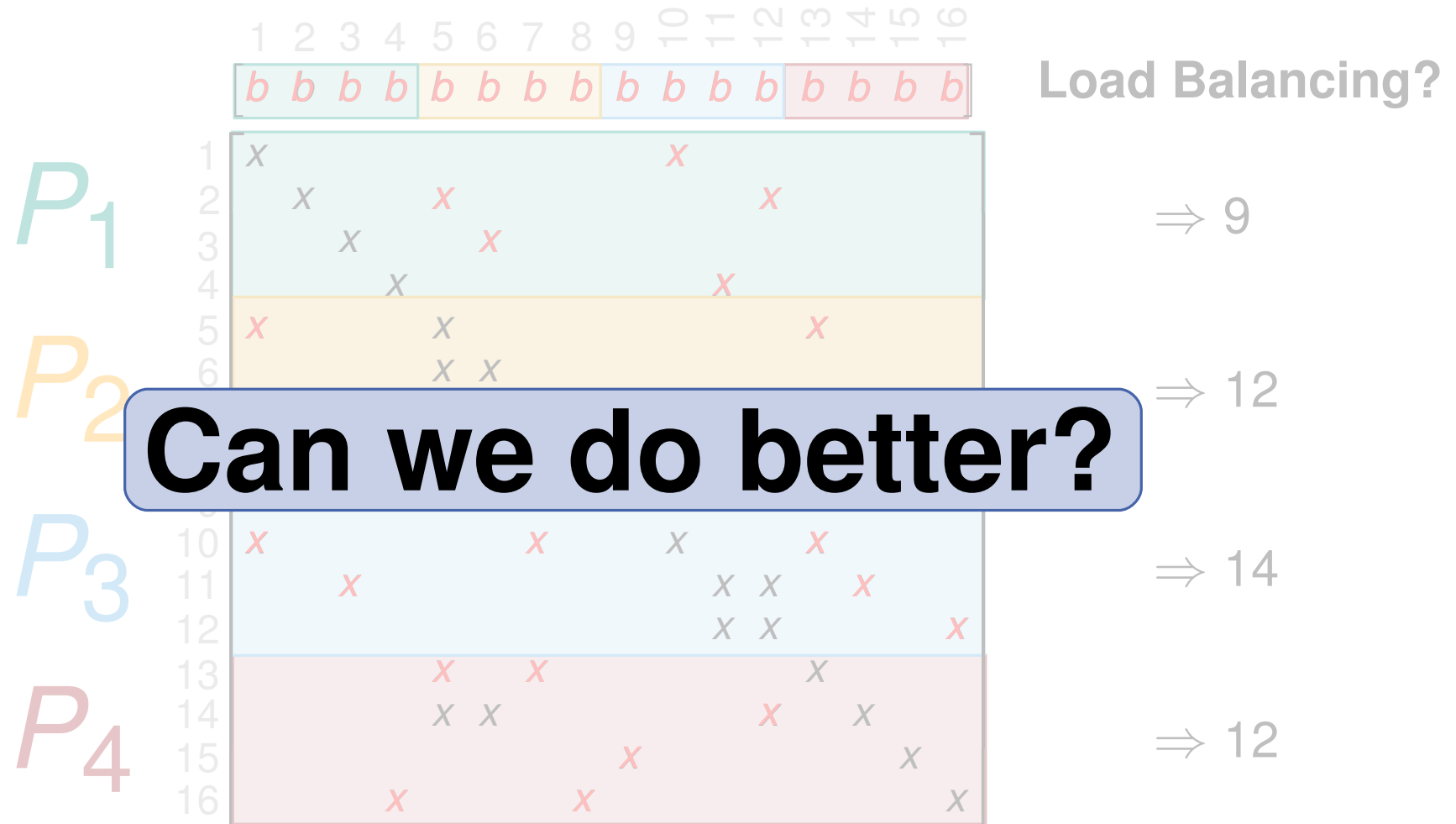
$\Rightarrow 14$

$\Rightarrow 12$

**Communication Volume?**  $\Rightarrow 24$  entries!

# Naive Approach: Rowwise Decomposition

$$A \in \mathbb{R}^{16 \times 16}$$



**Can we do better?**

Communication Volume? ⇒ 24 entries!



# From $\text{SpM} \times V$ to Hypergraph Partitioning

$$A \in \mathbb{R}^{16 \times 16} \Rightarrow H = (V_R, E_C)$$

- One vertex per row:

$$\Rightarrow V_R = \{v_1, v_2, \dots, v_{16}\}$$

- One hyperedge per column:

$$\Rightarrow E_C = \{e_1, e_2, \dots, e_{16}\}$$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$
1	x									x						
2		x			x							x				
3			x			x										
4				x							x					
5	x				x								x			
6					x	x										
7		x			x		x		x							
8				x				x			x					
9									x			x				x
10	x						x			x			x			
11			x								x	x		x		
12											x	x				x
13					x		x						x			
14					x	x						x		x		
15									x						x	
16				x				x								x

# From $\text{SpM} \times V$ to Hypergraph Partitioning

$$A \in \mathbb{R}^{16 \times 16} \Rightarrow H = (V_R, E_C)$$

- One vertex per row:  
 $\Rightarrow V_R = \{v_1, v_2, \dots, v_{16}\}$
- One hyperedge per column:  
 $\Rightarrow E_C = \{e_1, e_2, \dots, e_{16}\}$

$v_i \in V_R :$

- task to compute inner product of row  $i$  with  $b$
- $\Rightarrow c(v_i) := \# \text{ nonzeros}$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$
1	$x$									$x$						
2		$x$			$x$							$x$				
3			$x$			$x$										
4				$x$							$x$					
5	$x$				$x$								$x$			
6					$x$	$x$										
7		$x$			$x$		$x$		$x$							
8				$x$				$x$			$x$					
9									$x$		$x$				$x$	
10	$x$						$x$			$x$		$x$				
11			$x$								$x$	$x$		$x$		
12											$x$	$x$				$x$
13					$x$		$x$						$x$			
14					$x$	$x$						$x$		$x$		
15									$x$						$x$	
16				$x$				$x$								$x$

# From $\text{SpM} \times V$ to Hypergraph Partitioning

$$A \in \mathbb{R}^{16 \times 16} \Rightarrow H = (V_R, E_C)$$

- One vertex per row:  
 $\Rightarrow V_R = \{v_1, v_2, \dots, v_{16}\}$
- One hyperedge per column:  
 $\Rightarrow E_C = \{e_1, e_2, \dots, e_{16}\}$

$v_i \in V_R$ :

- task to compute inner product of row  $i$  with  $b$
- $\Rightarrow c(v_i) := \# \text{ nonzeros}$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$
1	$x$				$x$					$x$						
2		$x$			$x$							$x$				
3			$x$			$x$										
4				$x$							$x$					
5	$x$				$x$								$x$			
6					$x$	$x$										
7		$x$			$x$		$x$		$x$							
8				$x$				$x$			$x$					
9									$x$			$x$			$x$	
10	$x$						$x$			$x$			$x$			
11			$x$								$x$	$x$		$x$		
12											$x$	$x$				$x$
13					$x$		$x$						$x$			
14					$x$	$x$						$x$		$x$		
15									$x$						$x$	
16				$x$				$x$								$x$

$e_j \in E_C$ : set of vertices that need  $b_j$

# From $\text{SpM} \times V$ to Hypergraph Partitioning

$$A \in \mathbb{R}^{16 \times 16} \Rightarrow H = (V_R, E_C)$$

- One vertex per row:

$$\Rightarrow V_R = \{v_1, v_2, \dots, v_{16}\}$$

- One hyperedge per column:

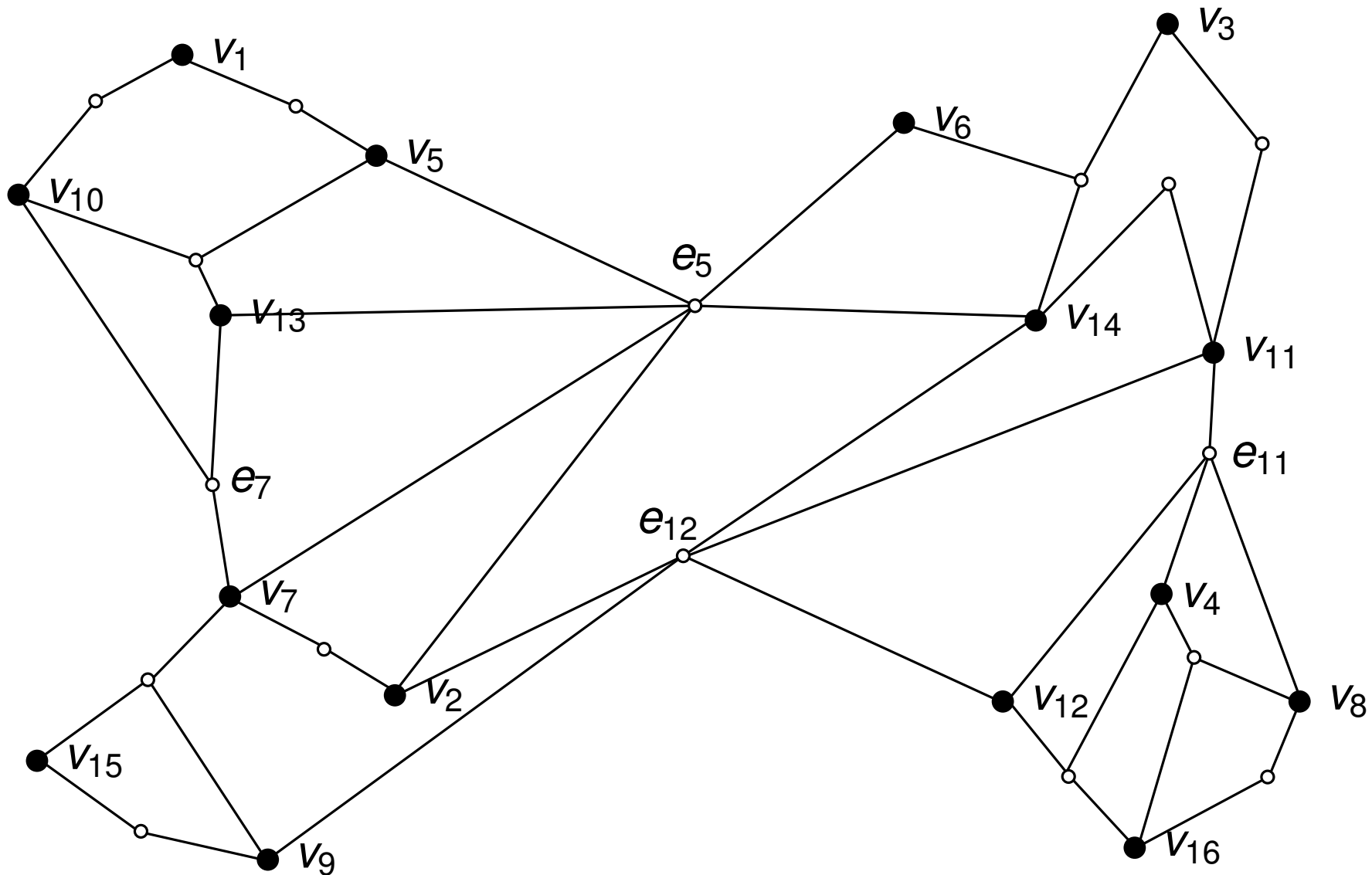
$$\Rightarrow E_C = \{e_1, e_2, \dots, e_{16}\}$$

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$	$b$
1	$x$									$x$						
2		$x$			$x$							$x$				
3			$x$			$x$										
4				$x$							$x$					
5	$x$				$x$								$x$			
6					$x$	$x$										
7																
8																
9																
10																
11																
12										$x$	$x$					$x$
13					$x$		$x$						$x$			
14					$x$	$x$					$x$			$x$		
15									$x$						$x$	
16				$x$				$x$								$x$

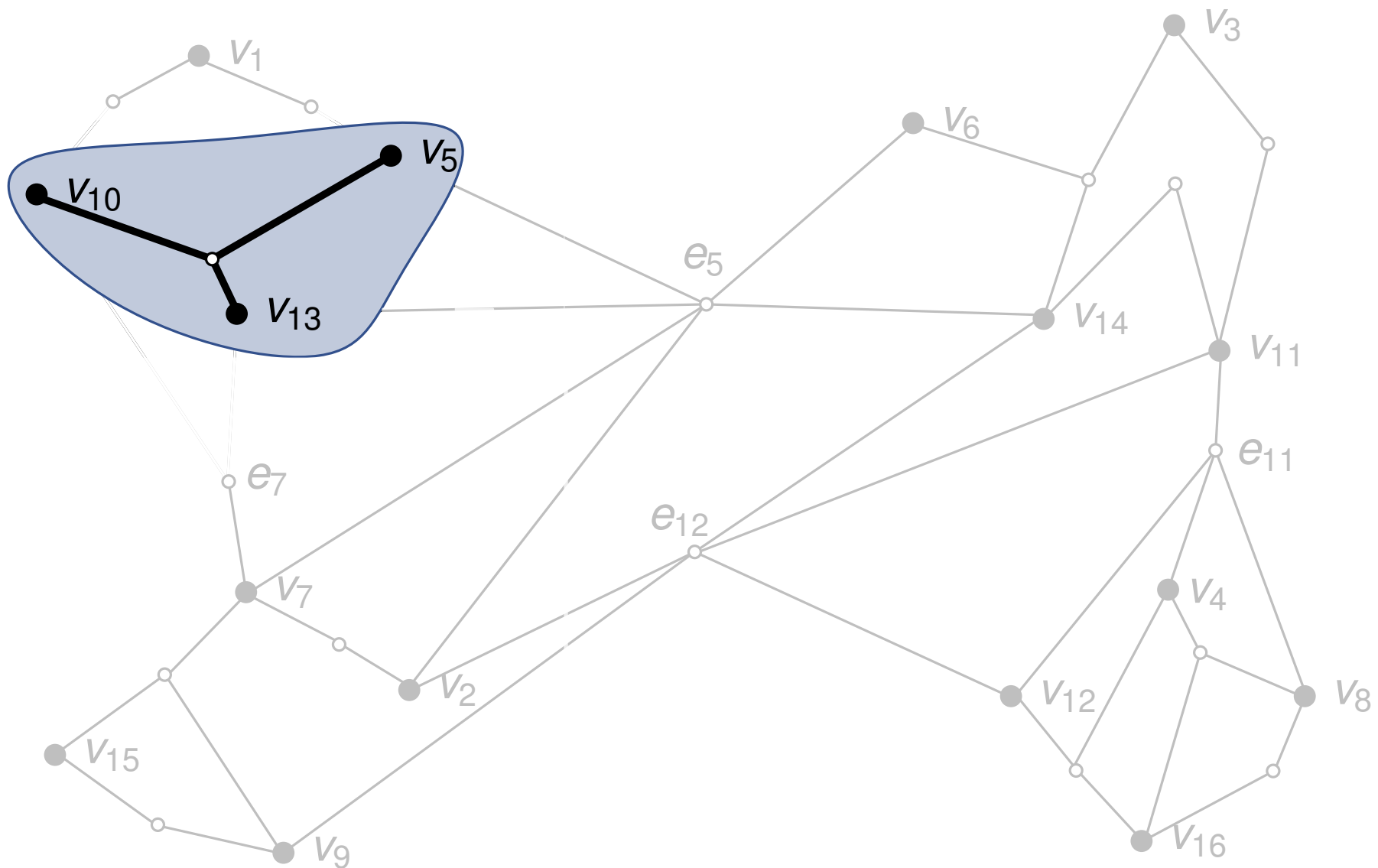
**Solution:**  $\varepsilon$ -balanced partition of  $H$

- balanced partition  $\rightsquigarrow$  computational load balance
- small  $(\lambda - 1)$ -cutsizes  $\rightsquigarrow$  minimizing communication volume

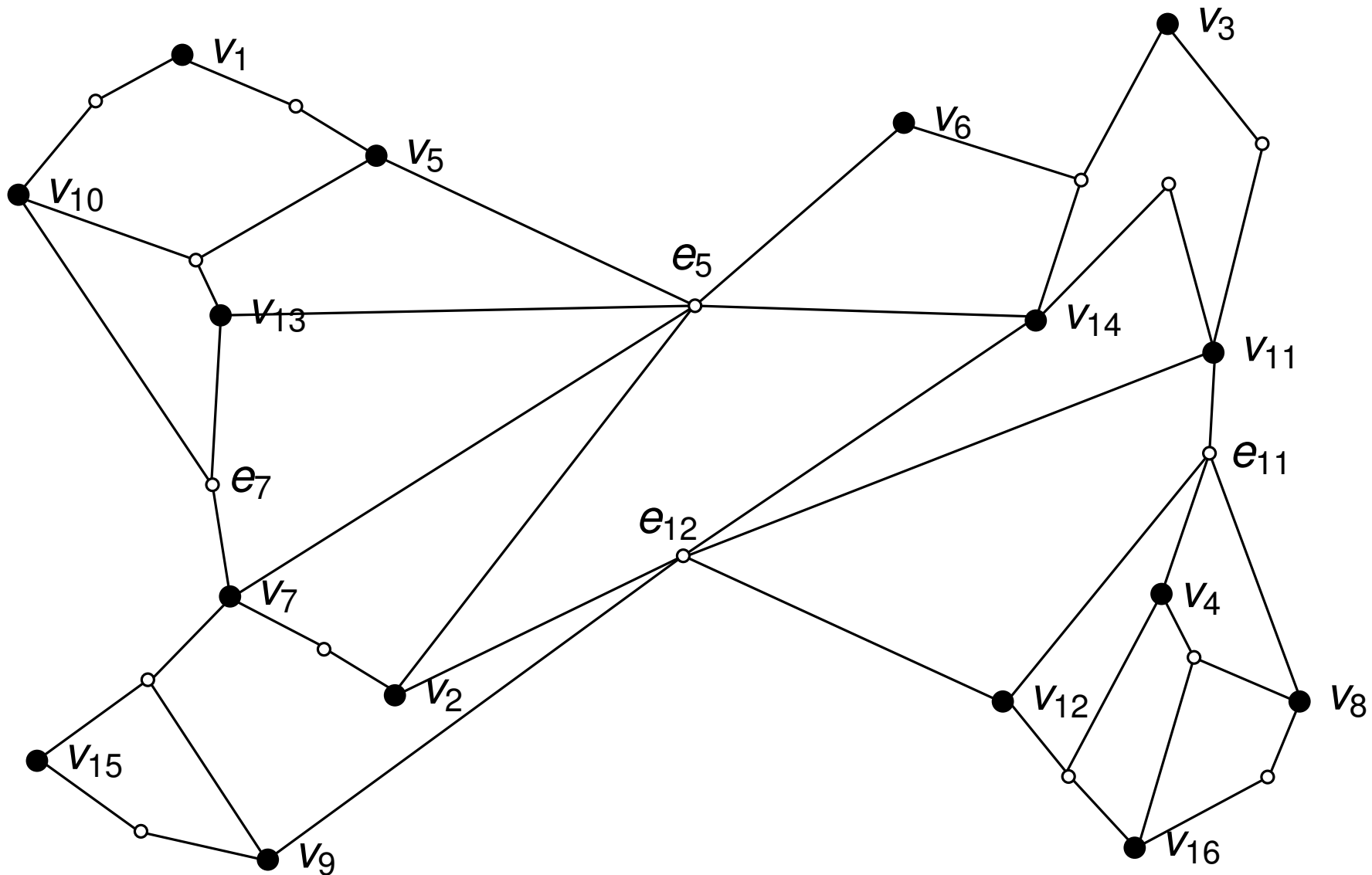
# From $\text{SpM} \times V$ to Hypergraph Partitioning



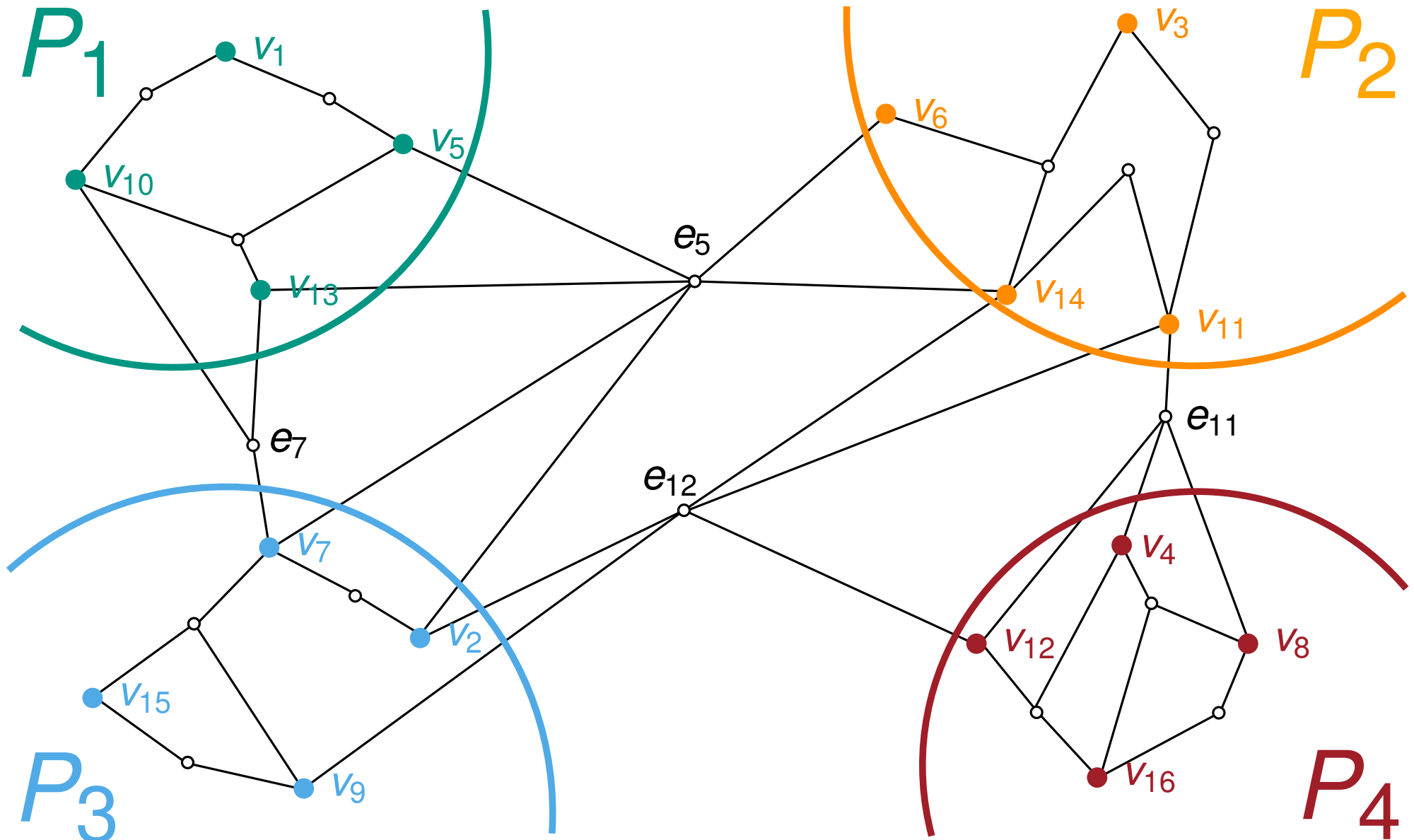
# From $\text{SpM} \times V$ to Hypergraph Partitioning



# From $\text{SpM} \times V$ to Hypergraph Partitioning

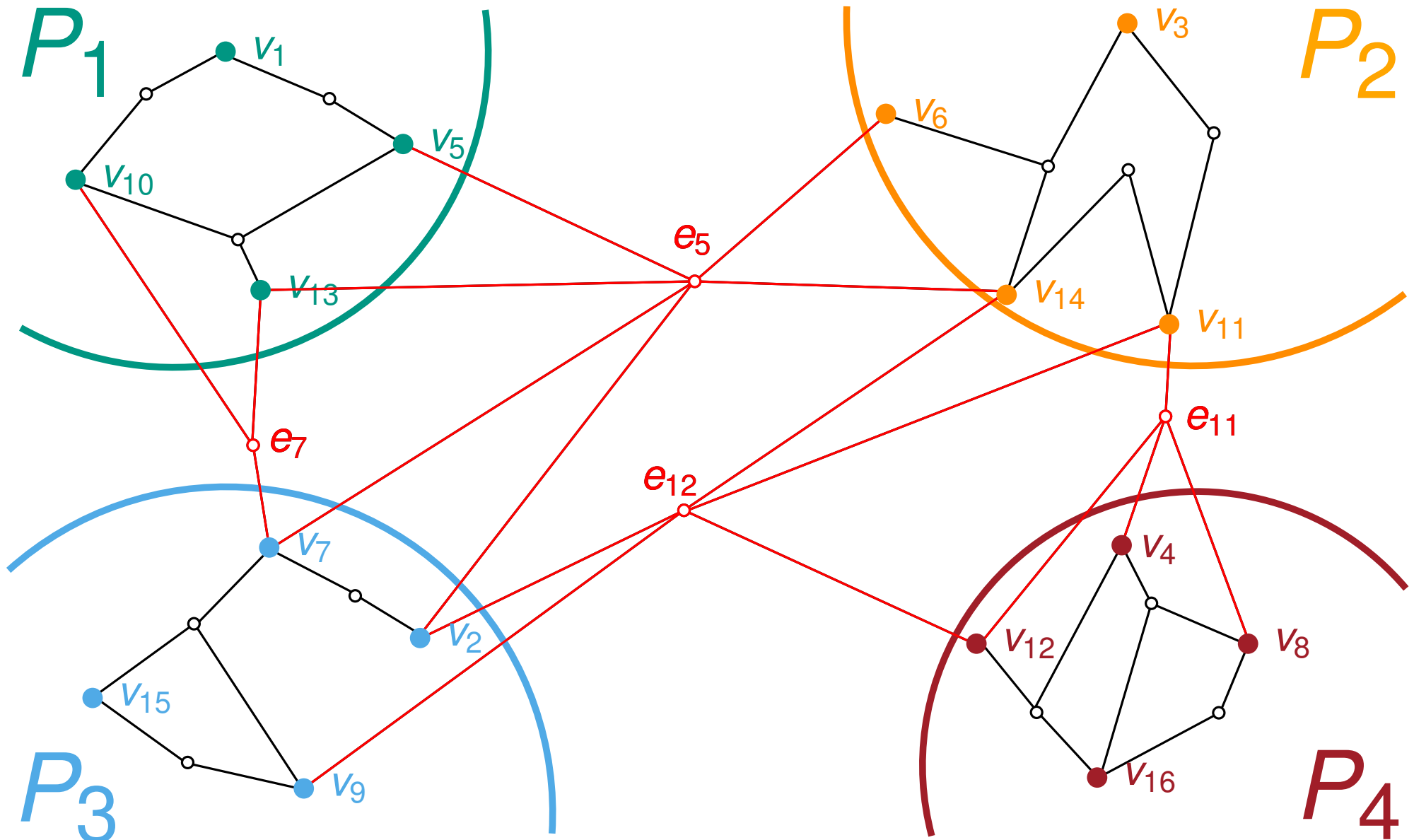


# From $\text{SpM} \times V$ to Hypergraph Partitioning





# From $\text{SpM} \times V$ to Hypergraph Partitioning



# From Hypergraph Partitioning to $\text{SpM} \times \mathbf{V}$

	0	3	5	1	6	14	11	3	2	15	7	9	8	16	12	4
	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>
$P_1$	10	x	x		x							x				
	13		x	x								x				
	5		x	x	x											
	1	x			x											
$P_2$	6			x		x										
	14			x		x	x									x
	11						x	x	x							x
	3					x			x							
$P_3$	2			x						x						x
	15										x		x			
	7			x						x		x	x			
	9										x		x			x
$P_4$	8							x						x		x
	16													x	x	x
	12							x							x	x
	4							x							x	x

# From Hypergraph Partitioning to $\text{SpM} \times \text{V}$

	0	3	5	1	6	14	11	3	2	15	7	9	8	16	12	4
	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>	<i>b</i>
$P_1$	x	x		x							x					
		x	x								x					
			x	x	x											
	x				x											
$P_2$			x		x											
			x		x	x										x
						x	x	x								x
					x			x								
$P_3$			x						x							x
										x		x				
			x						x		x	x				
										x		x				x
$P_4$							x						x			x
													x	x		x
							x							x	x	
							x							x		x

**Load Balancing?**

# From Hypergraph Partitioning to $\text{SpM} \times \mathbf{V}$

	0	3	5	1	6	14	11	3	2	15	7	9	8	16	12	4
	b				b				b				b			
$P_1$	x	x		x							x					
		x	x								x					
			x	x	x											
	x				x											
$P_2$			x		x											
			x		x	x									x	
						x	x	x							x	
					x			x								
$P_3$			x						x							x
										x		x				
			x						x		x	x				
										x		x				x
$P_4$							x						x			x
													x	x		x
							x							x	x	
							x							x	x	

**Load Balancing?**

$\Rightarrow 12$

$\Rightarrow 12$

$\Rightarrow 12$

$\Rightarrow 12$

# From Hypergraph Partitioning to $\text{SpM} \times \text{V}$

Where are the cut-hyperedges?

	0	3	5	1	6	4	7	3	2	5	7	9	8	6	2	4
	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b	b
$P_1$	10	x	x		x							x				
	13		x	x								x				
	5		x	x	x											
	1	x			x											
$P_2$	6			x		x										
	14		x		x	x										x
	11					x	x	x								x
	3				x			x								
$P_3$	2		x						x							x
	15									x		x				
	7		x						x		x	x				
	9									x		x				x
$P_4$	8							x					x			x
	16												x	x		x
	12							x						x	x	
	4							x						x		x

Load Balancing?

$\Rightarrow 12$

$\Rightarrow 12$

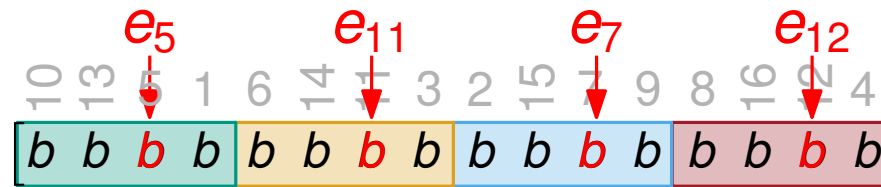
$\Rightarrow 12$

$\Rightarrow 12$

Communication Volume?

# From Hypergraph Partitioning to $\text{SpM} \times V$

Where are the cut-hyperedges?



$P_1$

$P_2$

$P_3$

$P_4$

10	x	x	x						x					
13		x	x						x					
5		x	x	x										
1	x			x										
6		x			x									
14		x			x	x							x	
11						x	x	x						x
3					x			x						
2		x							x					x
15									x		x			
7		x							x		x	x		
9									x		x			x
8										x			x	x
16										x	x			x
12											x	x		
4												x	x	

Load Balancing?

$\Rightarrow 12$

$\Rightarrow 12$

$\Rightarrow 12$

$\Rightarrow 12$

Communication Volume?  $\Rightarrow 6$  entries!

# How does (Hyper)Graph Partitioning work?

# How does

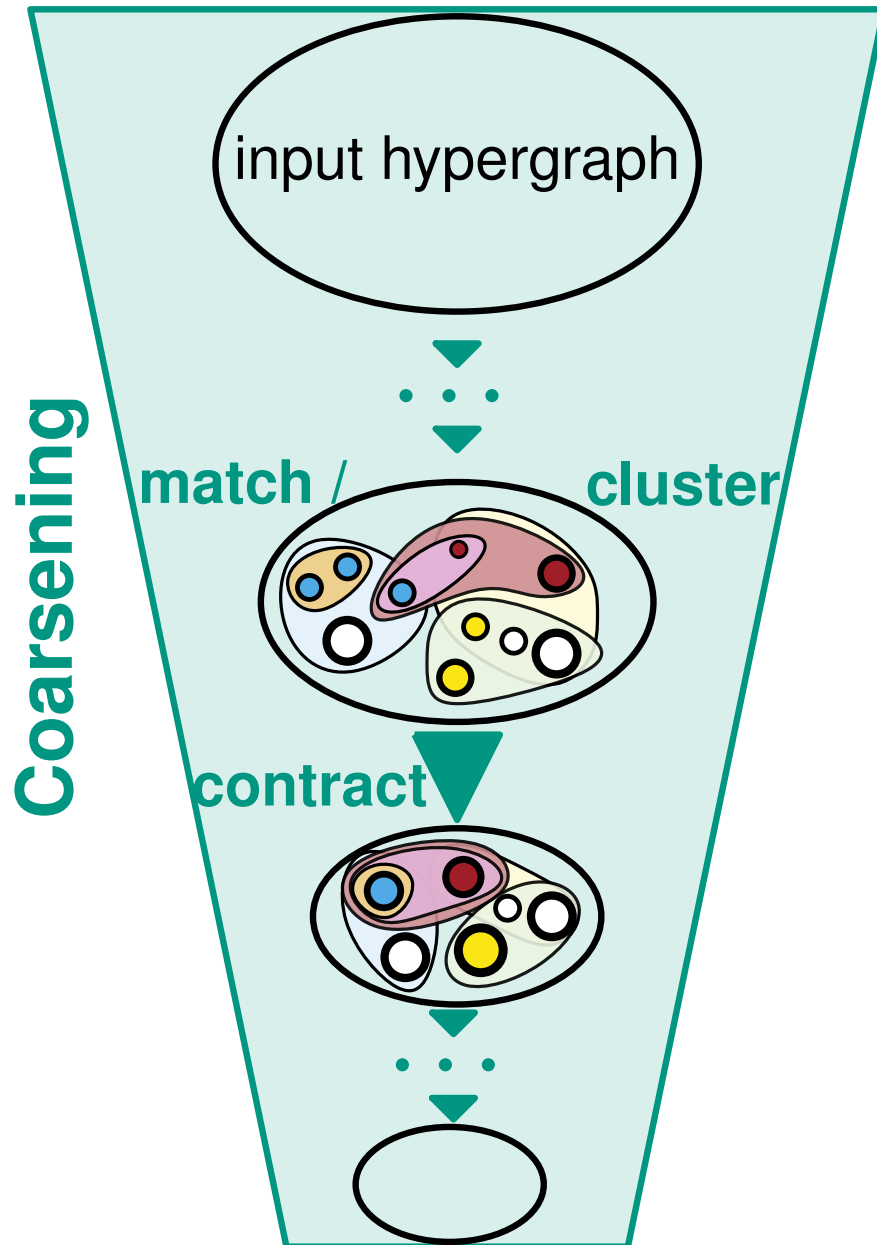
## Bad News:

- Hypergraph Partitioning is **NP**-hard
- even finding **good approximate** solutions for graphs is **NP**-hard

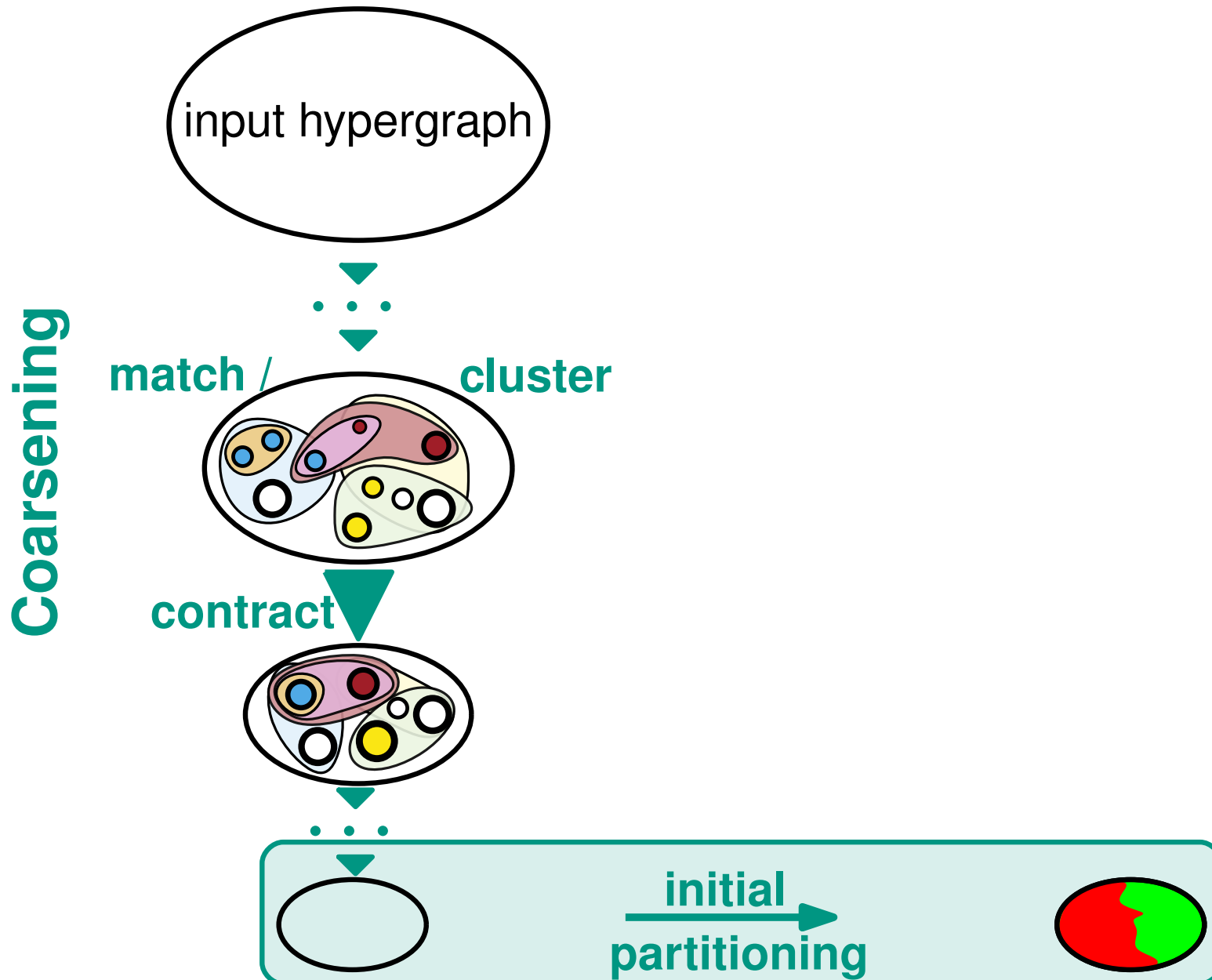
# work?



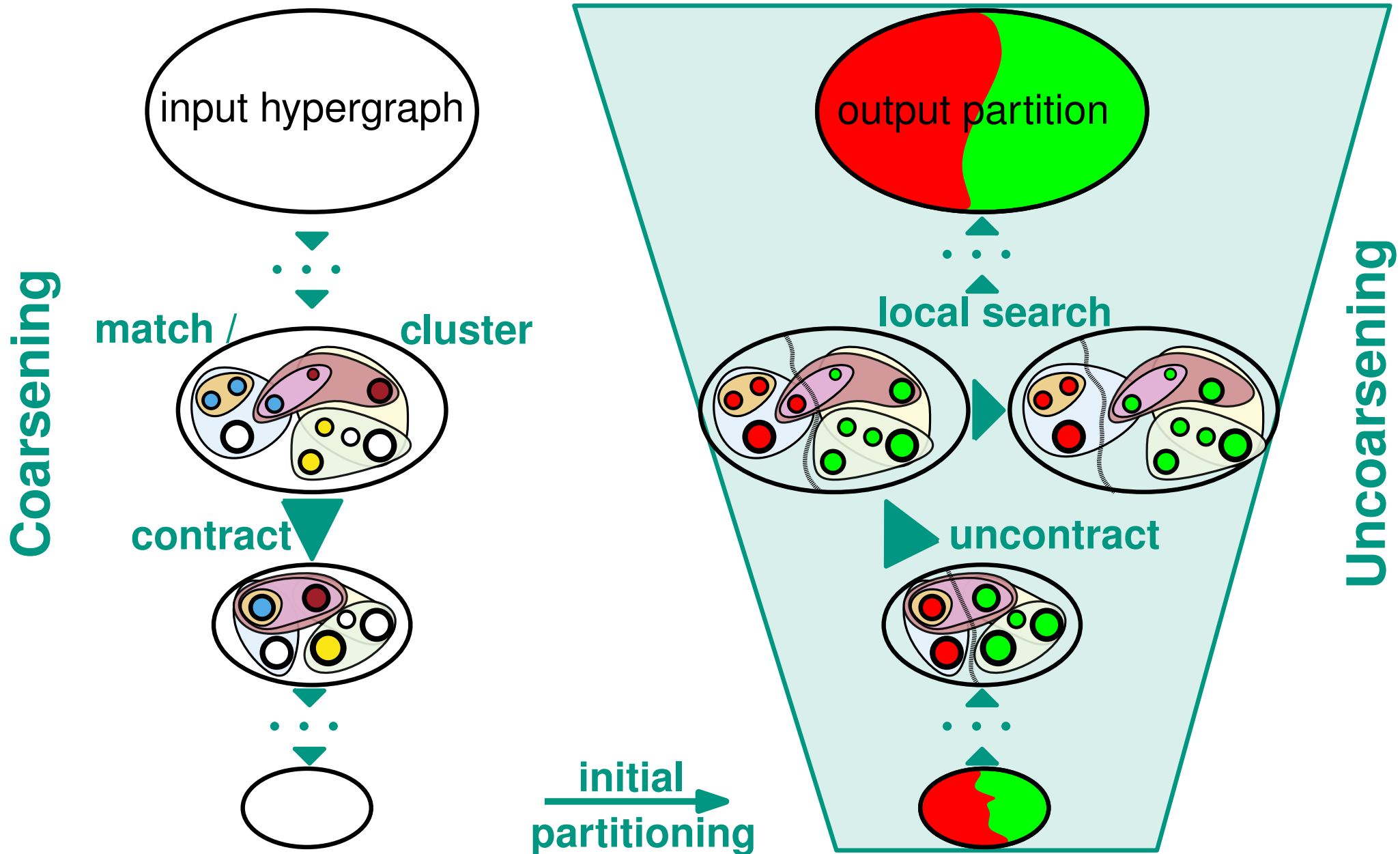
# Successful Heuristic: Multilevel Paradigm



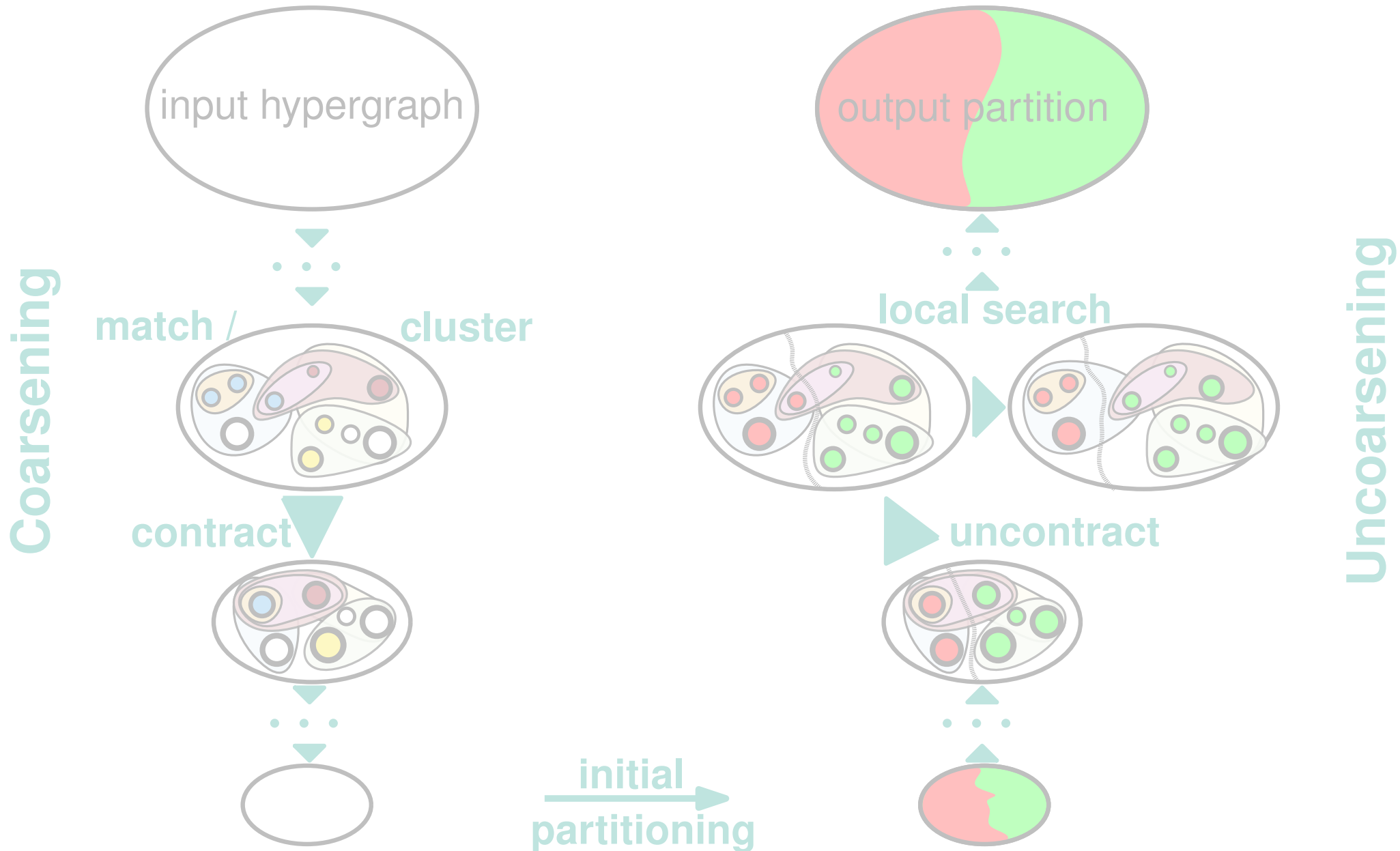
# Successful Heuristic: Multilevel Paradigm



# Successful Heuristic: Multilevel Paradigm



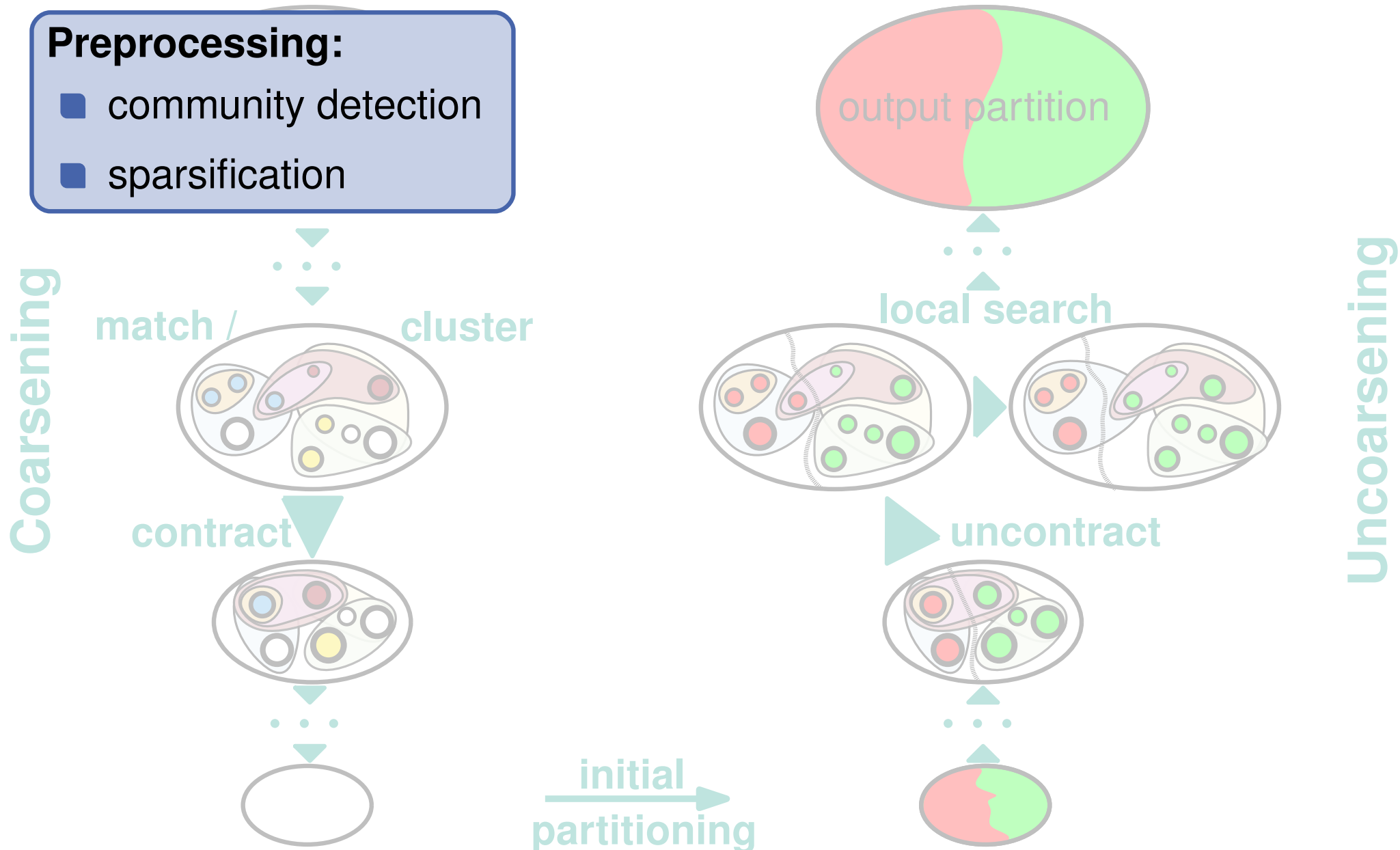
# Multilevel Paradigm - Algorithmic Ingredients



# Multilevel Paradigm - Algorithmic Ingredients

## Preprocessing:

- community detection
- sparsification



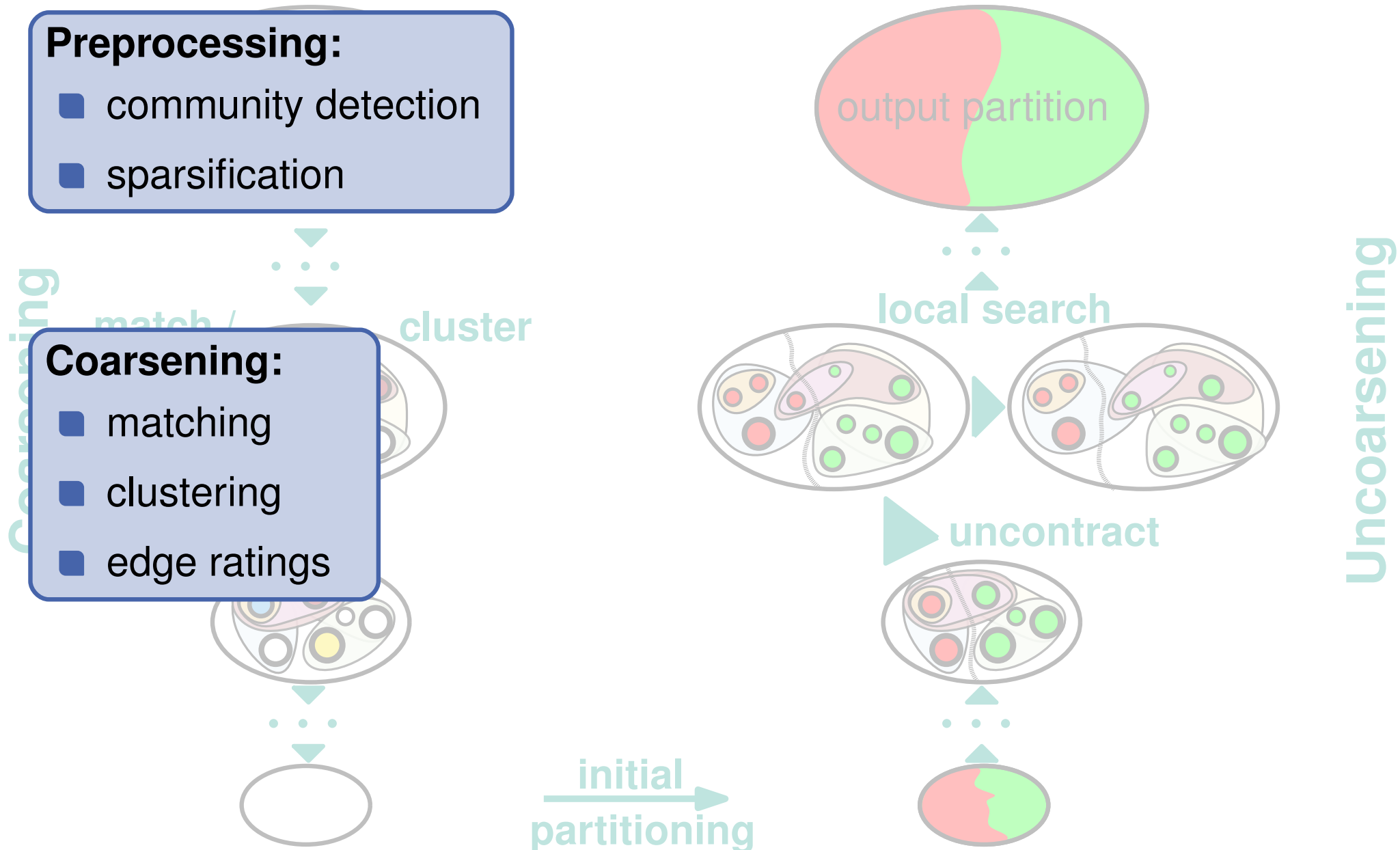
# Multilevel Paradigm - Algorithmic Ingredients

## Preprocessing:

- community detection
- sparsification

## Coarsening:

- matching
- clustering
- edge ratings



# Multilevel Paradigm - Algorithmic Ingredients

## Preprocessing:

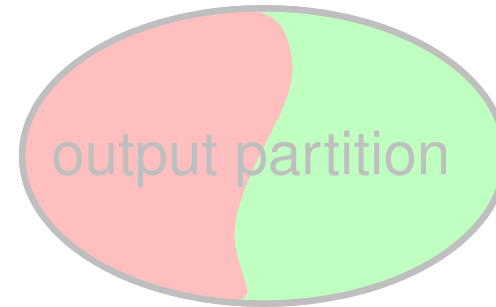
- community detection
- sparsification

## Coarsening:

- matching
- clustering
- edge ratings

## Initial Partitioning:

- portfolio of various algorithms  $\rightsquigarrow$  diversification



local search



A diagram illustrating the local search step. It shows two ovals representing graph partitions. The left oval contains several smaller colored ovals (red, green, yellow) representing clusters. The right oval shows a different configuration of these clusters, with some moved or merged, indicating a search for a better partition.

uncontract



A diagram illustrating the uncontract step. It shows a single oval containing several smaller colored ovals (red, green, yellow) representing clusters. An arrow points from this oval to the local search step, indicating that the clusters are being expanded or refined.

Coarsening

Uncoarsening

# Multilevel Paradigm - Algorithmic Ingredients

## Preprocessing:

- community detection
- sparsification

## Coarsening:

- matching
- clustering
- edge ratings

## Initial Partitioning:

- portfolio of various algorithms  $\rightsquigarrow$  diversification

## Local Search:

- Kernighan-Lin
- Fiduccia-Mattheyses
- Max-Flow Min-Cut

output partition

local search

coarsening

match /

cluster

refining



# Multilevel Paradigm - Algorithmic Ingredients

## Preprocessing:

- community detection
- sparsification

## Metaheuristics:

- Global Search
- Evolutionary Algorithms

## Coarsening:

- matching
- clustering
- edge ratings

## Local Search:

- Kernighan-Lin
- Fiduccia-Mattheyses
- Max-Flow Min-Cut

## Initial Partitioning:

- portfolio of various algorithms  $\rightsquigarrow$  diversification

# Multilevel Paradigm - Algorithmic Ingredients

## Preprocessing:

- community detection
- sparsification

## Metaheuristics:

- Global Search
- Evolutionary Algorithms

## Coarsening:

- matching
- clustering
- edge ratings

## Parallelization:

- shared memory
- distributed memory

## Local Search:

- Kernighan-Lin
- Fiduccia-Mattheyses
- Max-Flow Min-Cut

## Initial Partitioning:

- portfolio of various algorithms  $\rightsquigarrow$  diversification

# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

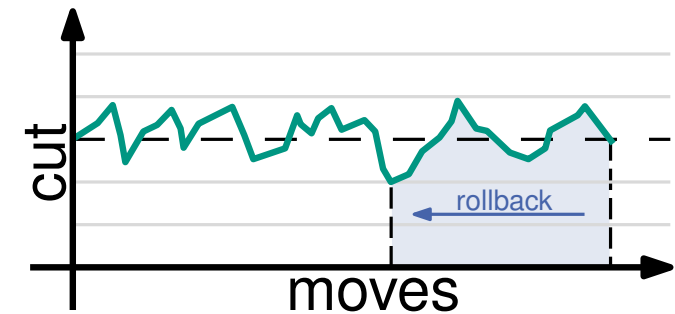
**while**  $\neg$  *done* **do**

    find best move

    perform best move

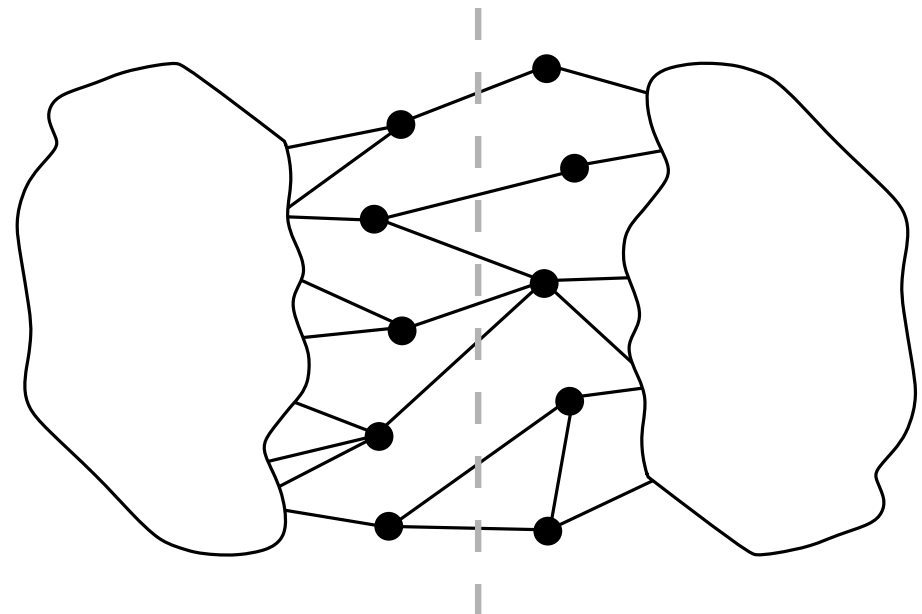
rollback to best solution

---



can worsen solution

- compute gain  $g(v) = d_{\text{ext}}(v) - d_{\text{int}}(v)$
- alternate between blocks
- edge-cut: 7



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

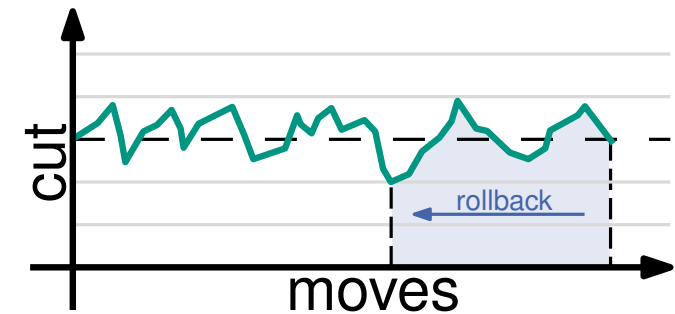
**while**  $\neg$  *done* **do**

    find best move

    perform best move

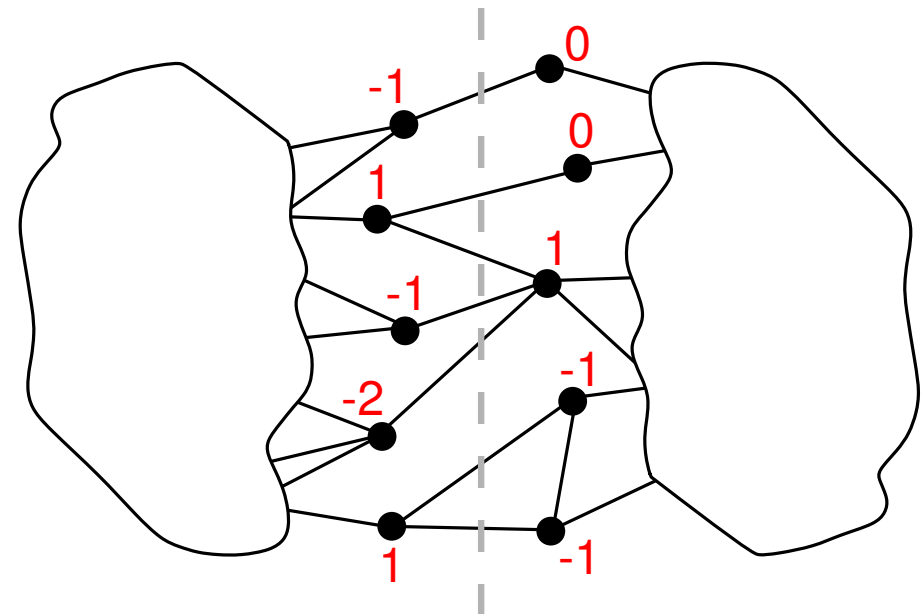
rollback to best solution

---



can worsen solution

- compute **gain**  $g(v) = d_{\text{ext}}(v) - d_{\text{int}}(v)$
- alternate between blocks
- edge-cut: 7



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

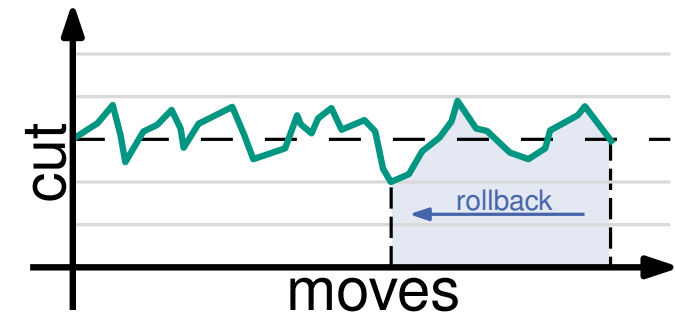
**while**  $\neg$  *done* **do**

    find best move

    perform best move

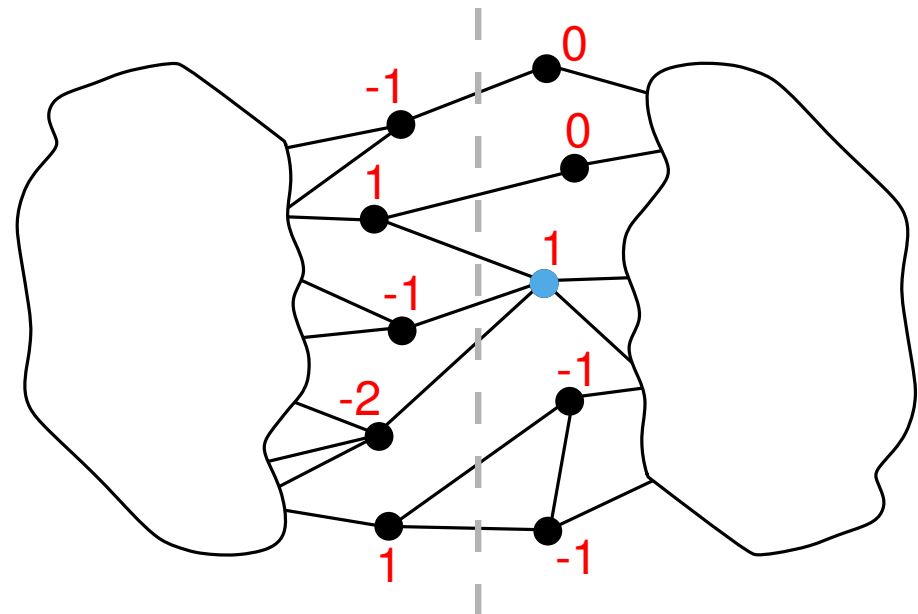
rollback to best solution

---



can worsen solution

- compute **gain**  $g(v) = d_{\text{ext}}(v) - d_{\text{int}}(v)$
- alternate between blocks
- edge-cut: 7



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

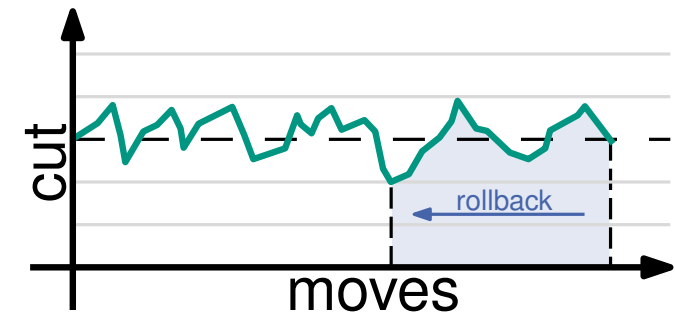
**while**  $\neg$  *done* **do**

    find best move

    perform best move

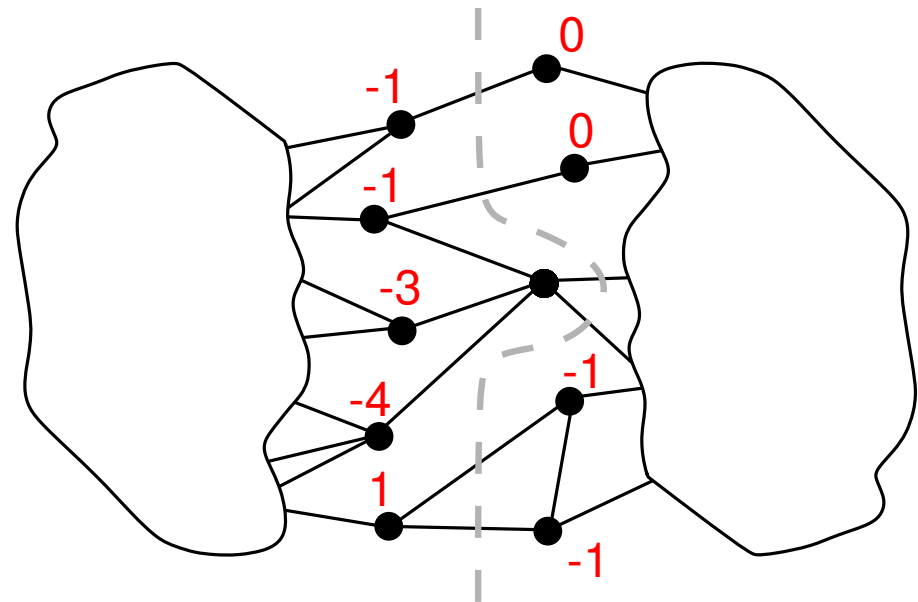
rollback to best solution

---



can worsen solution

- recalculate gain  $g(v)$  of neighbors
- move each node at most once
- edge-cut: 7, 6



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

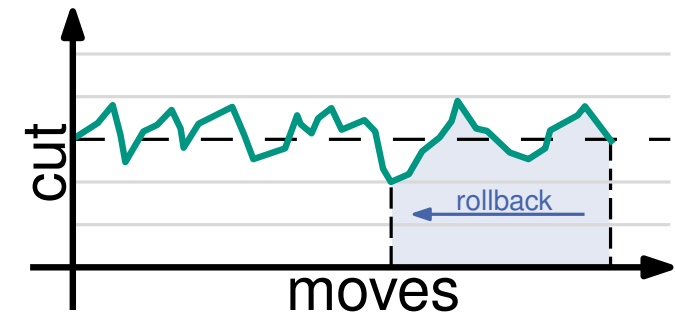
**while**  $\neg$  *done* **do**

    find best move

    perform best move

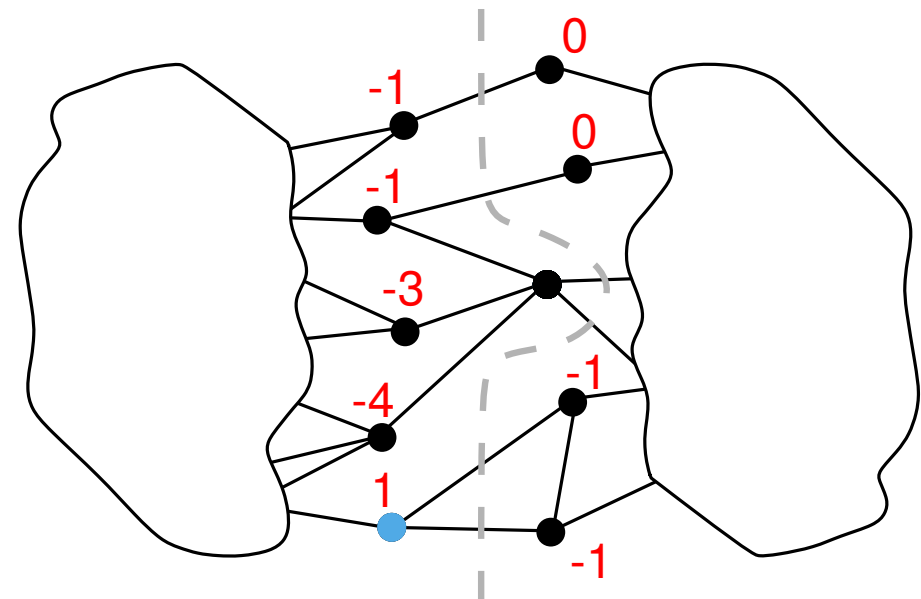
rollback to best solution

---



can worsen solution

- recalculate gain  $g(v)$  of neighbors
- move each node at most once
- edge-cut: 7, 6



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

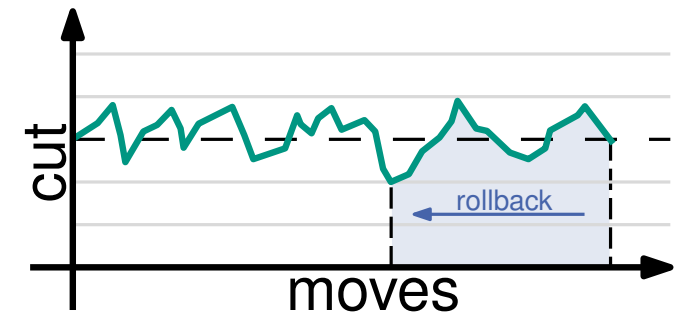
**while**  $\neg$  *done* **do**

    find best move

    perform best move

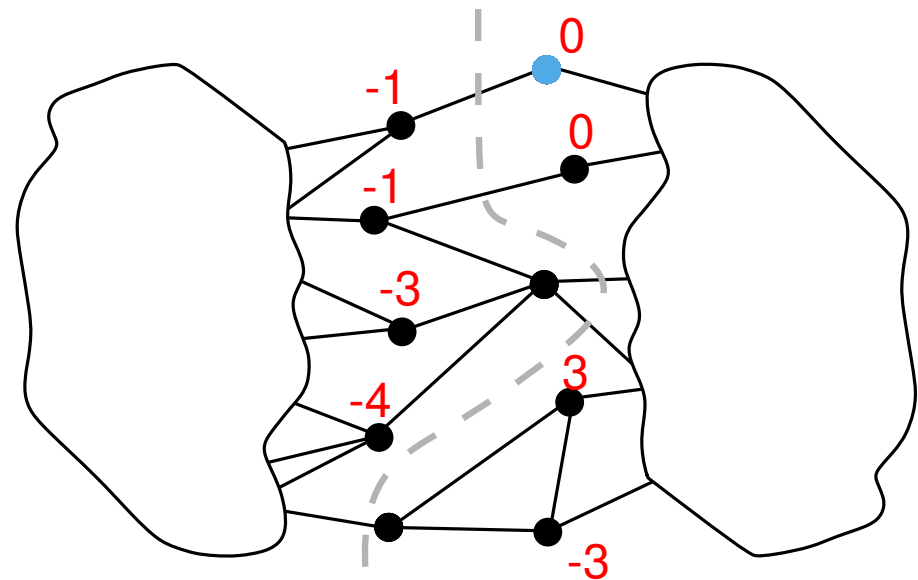
rollback to best solution

---



can worsen solution

- recalculate gain  $g(v)$  of neighbors
- move each node at most once
- edge-cut: 7, 6,5





# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

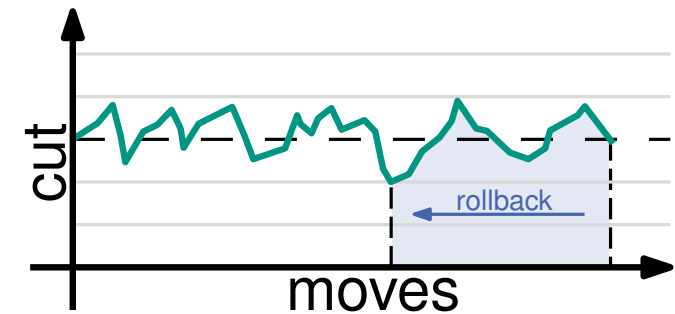
**while**  $\neg$  *done* **do**

    find best move

    perform best move

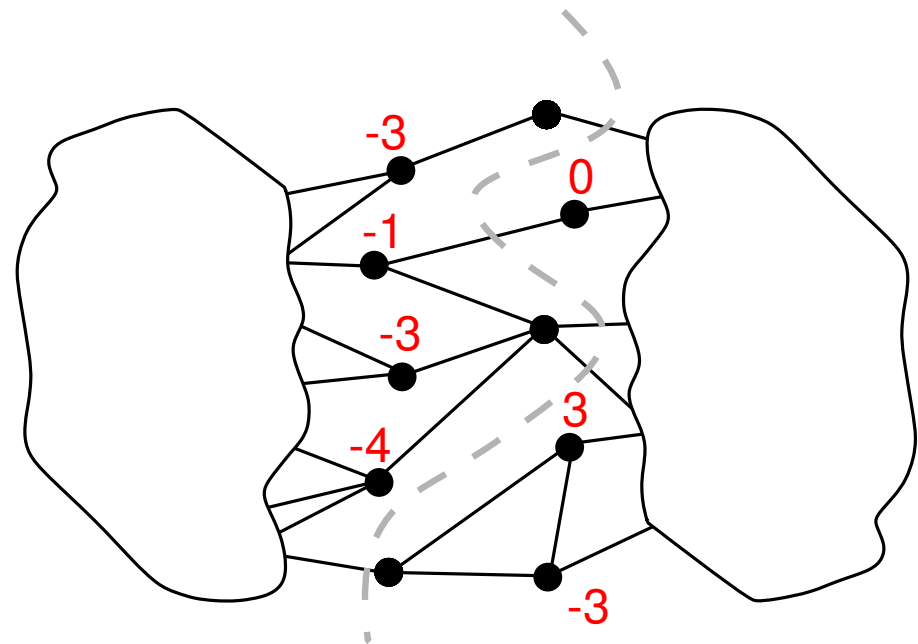
rollback to best solution

---



can worsen solution

- **recalculate** gain  $g(v)$  of neighbors
- move each node at most once
- edge-cut: **7, 6,5,5**



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

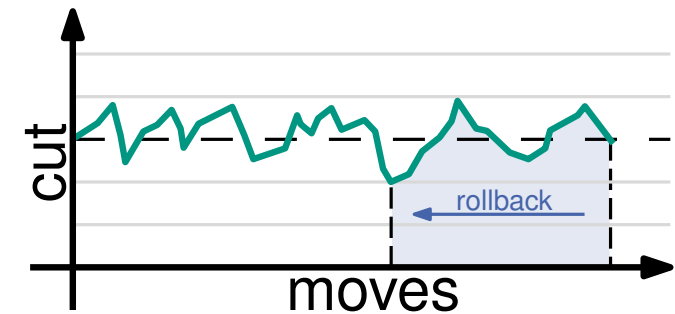
**while**  $\neg$  *done* **do**

    find best move

    perform best move

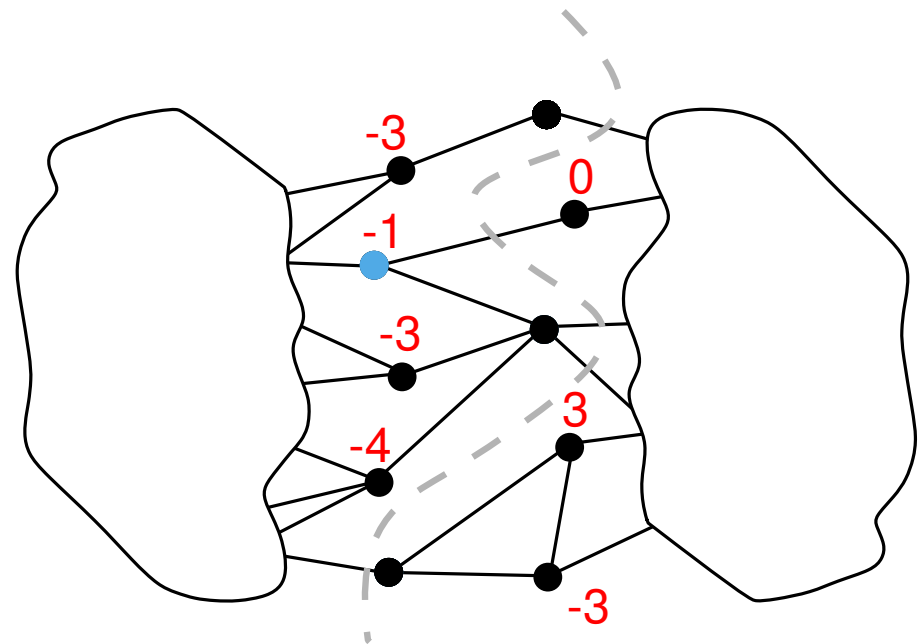
rollback to best solution

---



can worsen solution

- **recalculate** gain  $g(v)$  of neighbors
- move each node at most once
- edge-cut: 7, 6, 5, 5



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

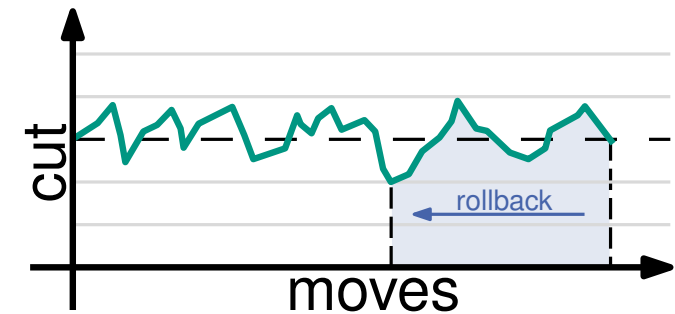
**while**  $\neg$  *done* **do**

    find best move

    perform best move

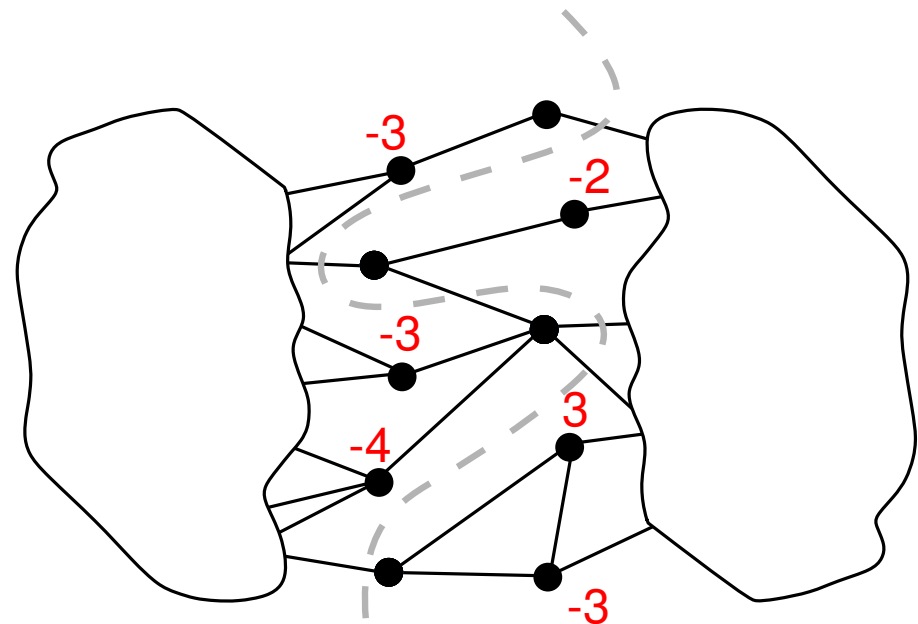
rollback to best solution

---



can worsen solution

- **recalculate** gain  $g(v)$  of neighbors
- move each node at most once
- edge-cut: **7, 6, 5, 5, 6**



# Fiduccia-Mattheyses Algorithm

---

## Algorithm 1: FM Local Search

---

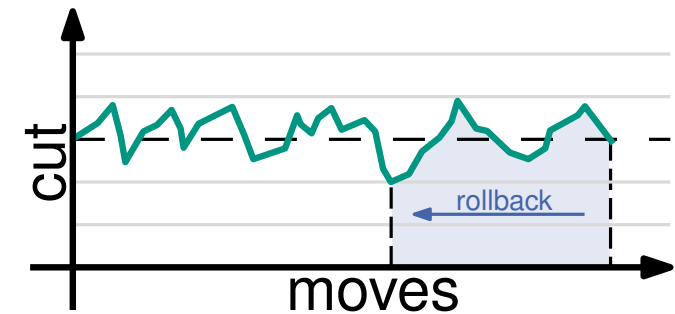
**while**  $\neg$  *done* **do**

    find best move

    perform best move

rollback to best solution

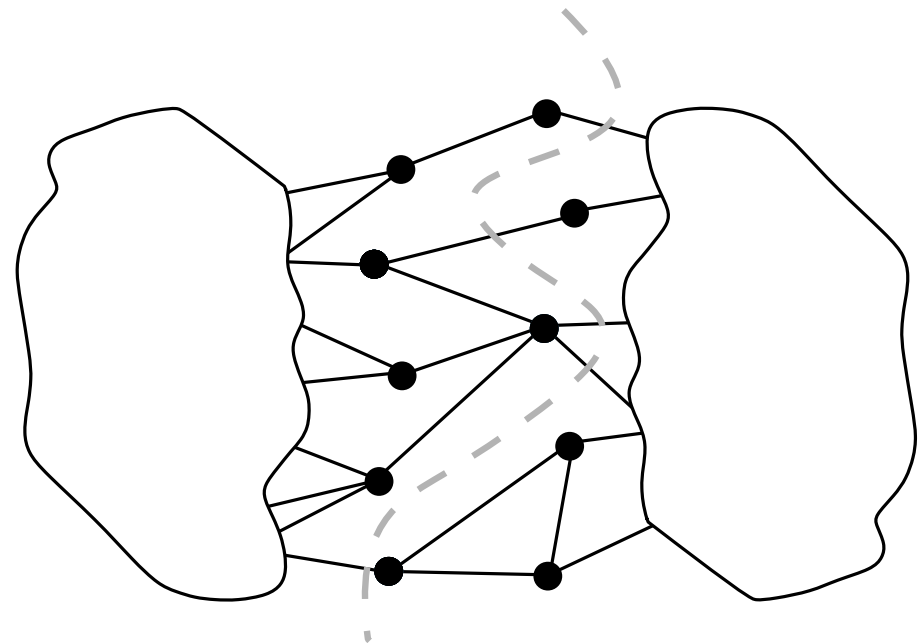
---



can worsen solution

- **recalculate** gain  $g(v)$  of neighbors
- move each node at most once
- edge-cut: 7, 6, 5, 5, 6

rollback



# KaHIP - Karlsruhe High Quality Partitioning

mapping [SEA17]

highly parallel [TPDPS17]

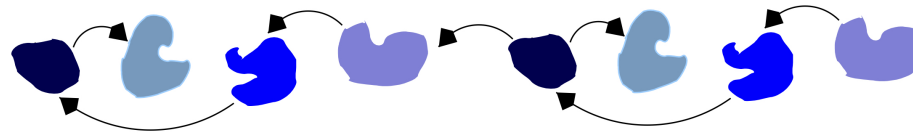
separators [SEA16,GECCO17]

road [ALX12,GIS15]

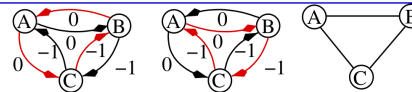
social [SEA14,IPDPS15,ALX15]

hypergraphs [ALX16]

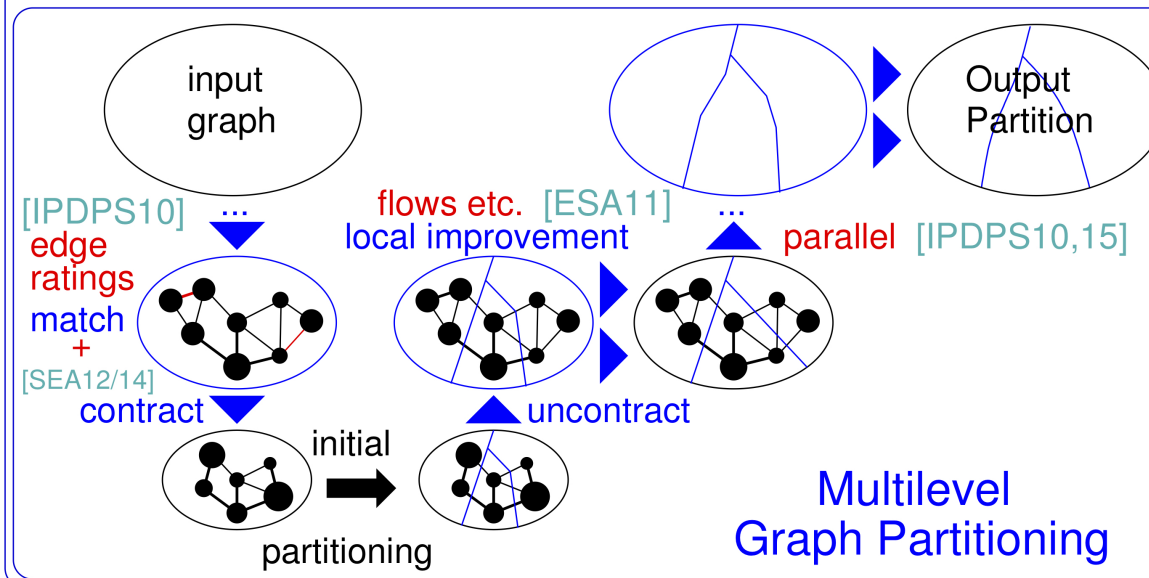
distr.  
evol. alg.  
[ALENEX12]  
[DIMACS12]



highly balanced:  
[SEA13]

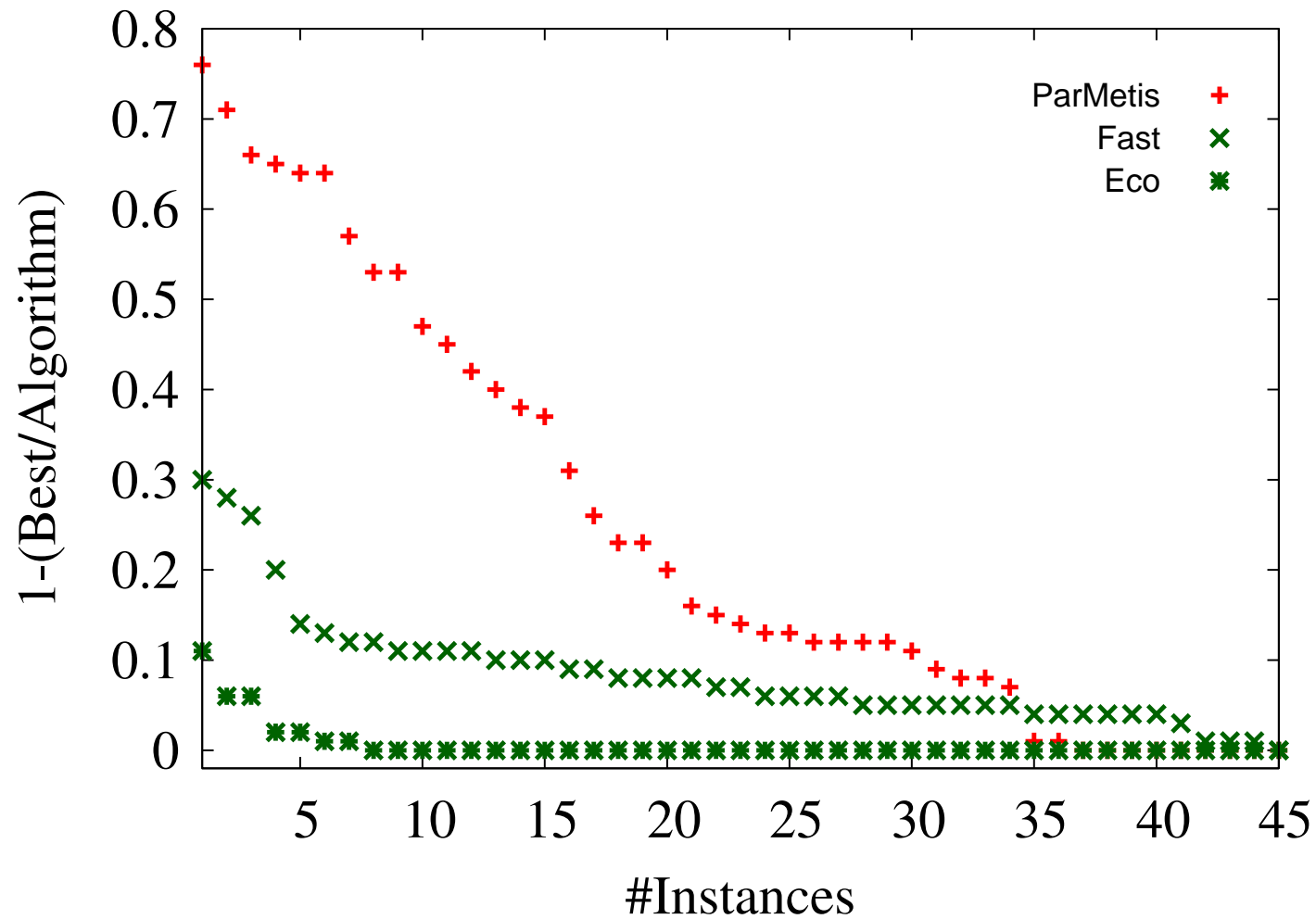


V- F- W- cycles a la multigrid [ESA11]



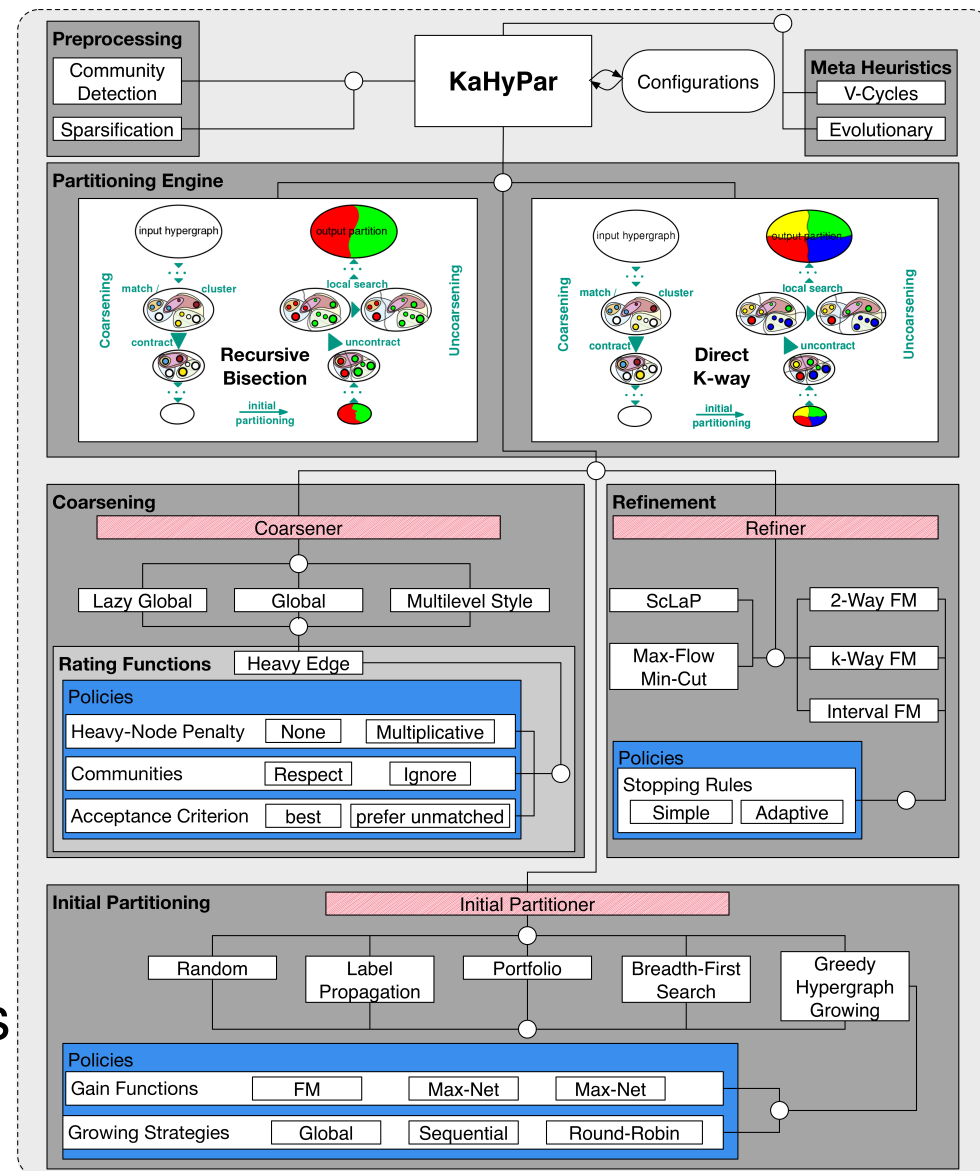
<http://algo2.iti.kit.edu/kahip/>

# Experimental Results – KaHIP (ParHIP)

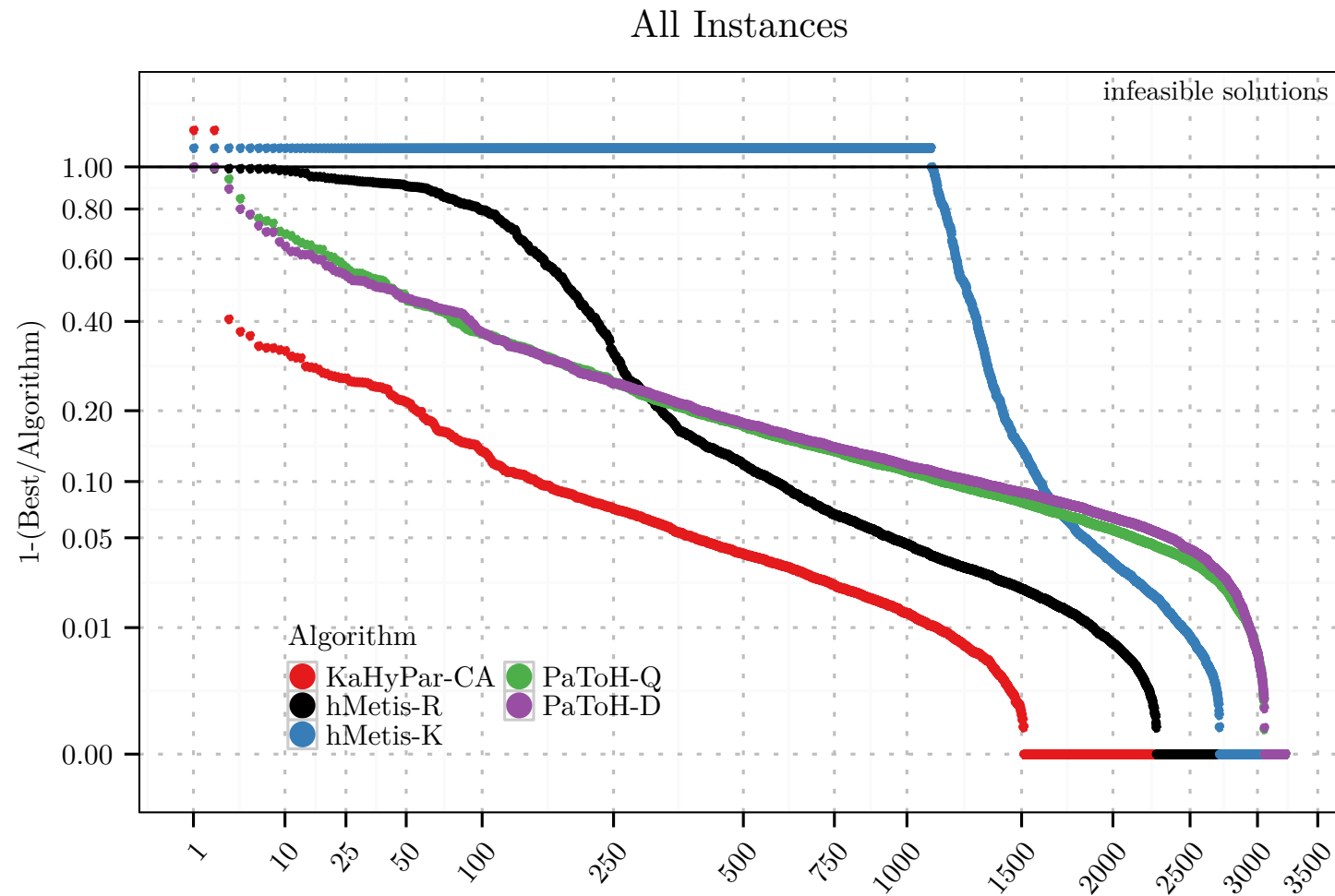


# KaHyPar - Karlsruhe Hypergraph Partitioning

- *n*-Level Partitioning Framework
- Objectives:
  - hyperedge cut
  - connectivity ( $\lambda - 1$ )
- Partitioning Modes:
  - recursive bisection
  - direct *k*-way
- Upcoming Features:
  - evolutionary algorithm
  - flow-based refinement
  - advanced local search algorithms
- <http://www.kahypar.org>



# Experimental Results – KaHyPar





# Conclusion

## (Hyper)Graph Partitioning:

- fundamental graph problem with **many** application areas
- successful heuristic: **multilevel** approach + **local search**
- Graphs: **KaHIP** – <http://algo2.iti.kit.edu/kahip/>
- Hypergraphs: **KaHyPar** – <http://www.kahypar.org>

