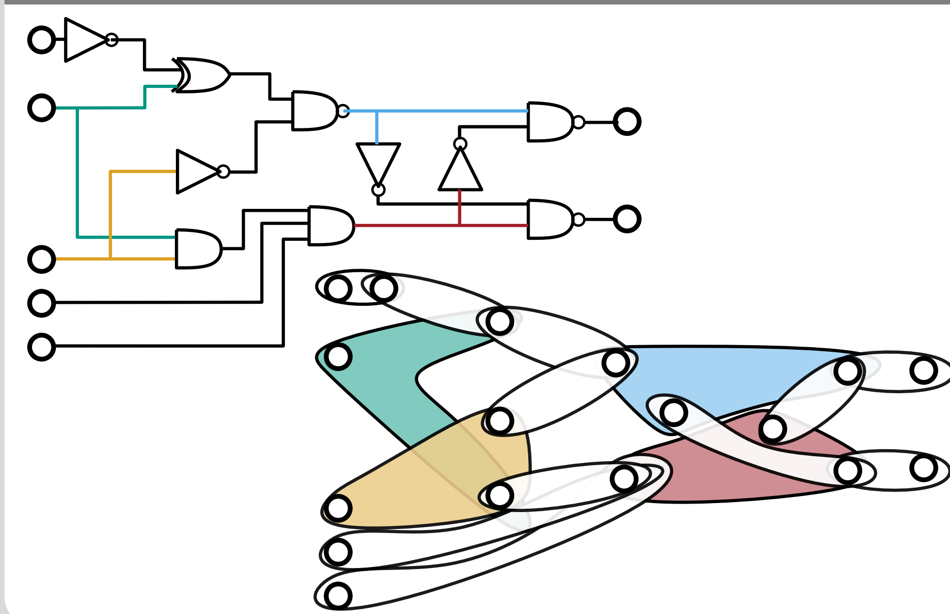
Karlsruhe Institute of Technology

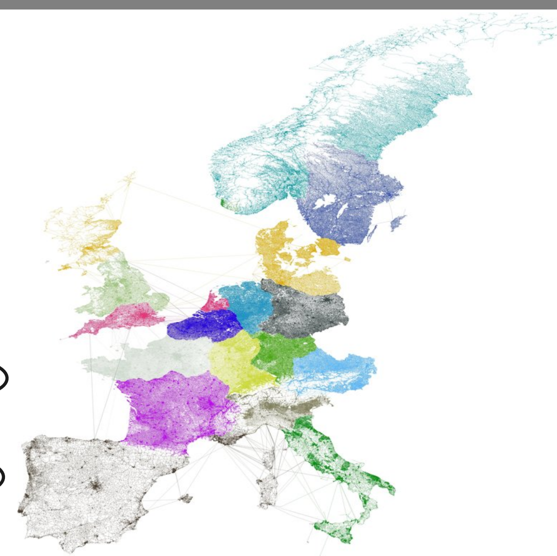
High Quality Hypergraph Partitioning

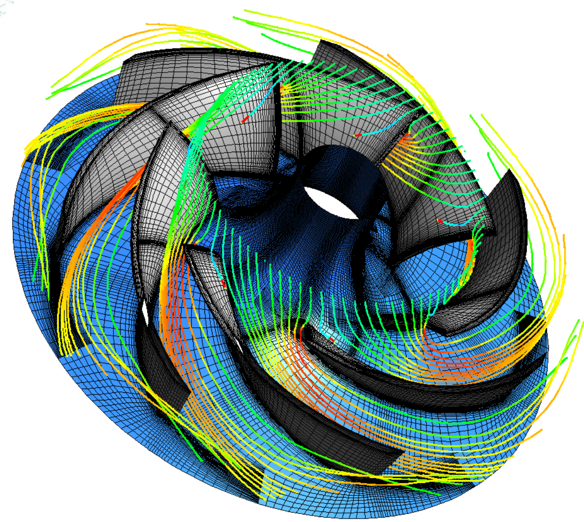
German-Israeli Winter School on Algorithms for Big Data · November 15, 2017

Robin Andre, Yaroslav Akhremtsev, Vitali Henne, Tobias Heuer, Henning Meyerhenke, Peter Sanders, Sebastian Schlag, Christian Schulz

INSTITUTE OF THEORETICAL INFORMATICS · ALGORITHMICS GROUP



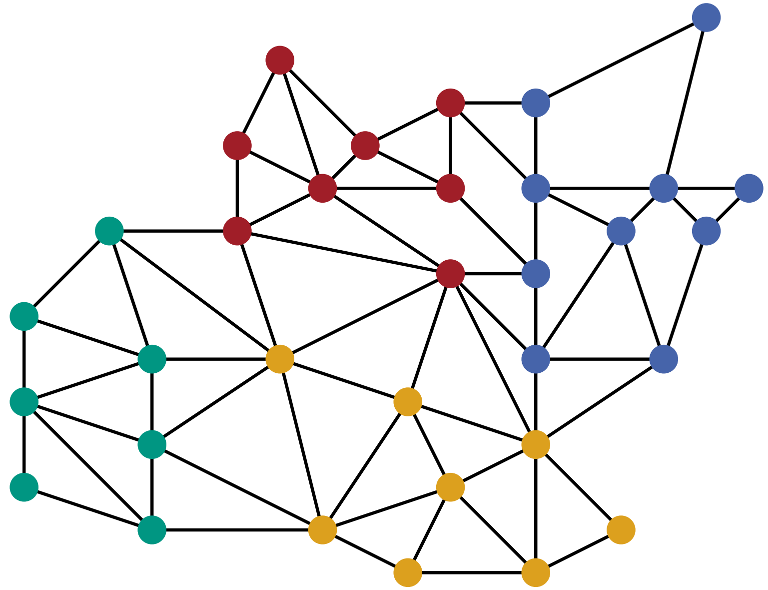




Graphs and Hypergraphs

Graph $G = (V, E)$

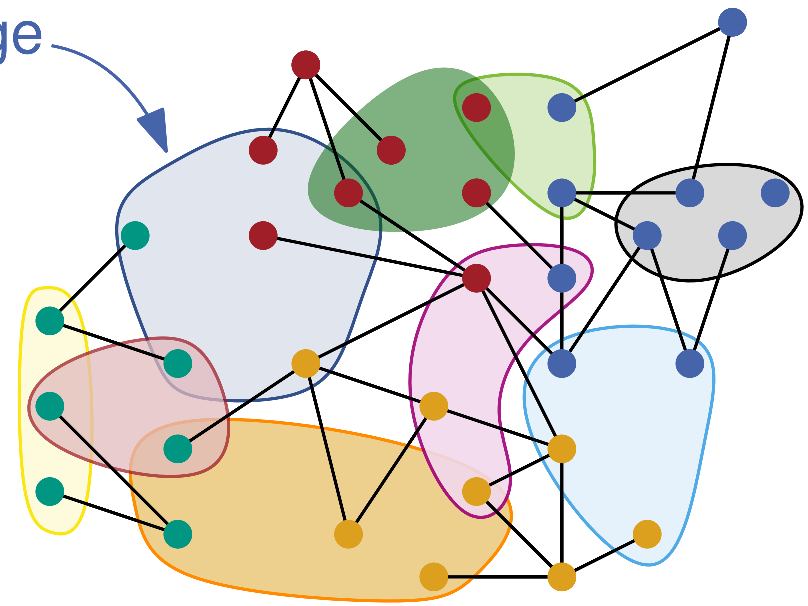
vertices → V, edges → E



- Models **relationships** between **objects**
- Dyadic ($\mathbf{2}$**-ary**) relationships

Hypergraph $H = (V, E)$

hyperedge



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

ε-Balanced Hypergraph Partitioning

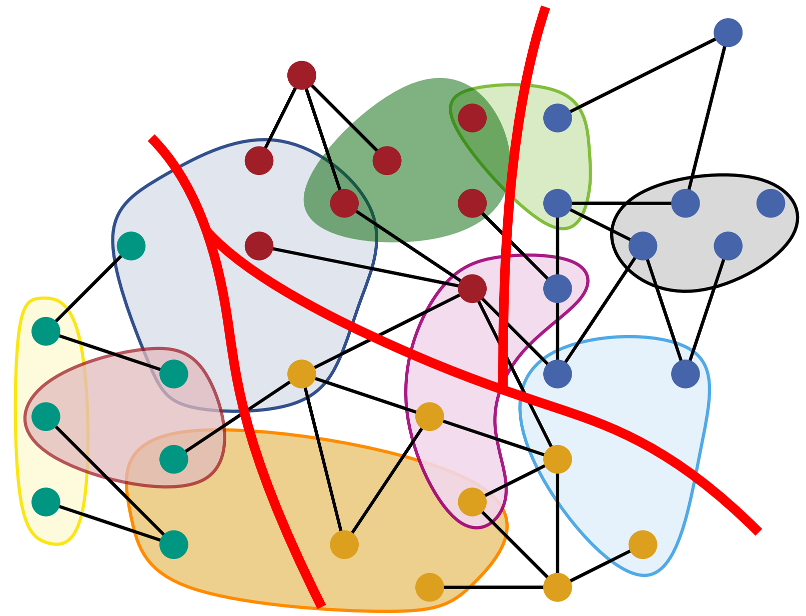
Karlsruhe Institute of Technology

Partition hypergraph $H = (V, E, c : V \rightarrow \mathrm{R}_{>0}, \omega : E \rightarrow \mathrm{R}_{>0})$ into **k** disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that

- Blocks V_i are **roughly equal-sized**:

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

- **Objective** function on hyperedges is **minimized**



ε-Balanced Hypergraph Partitioning

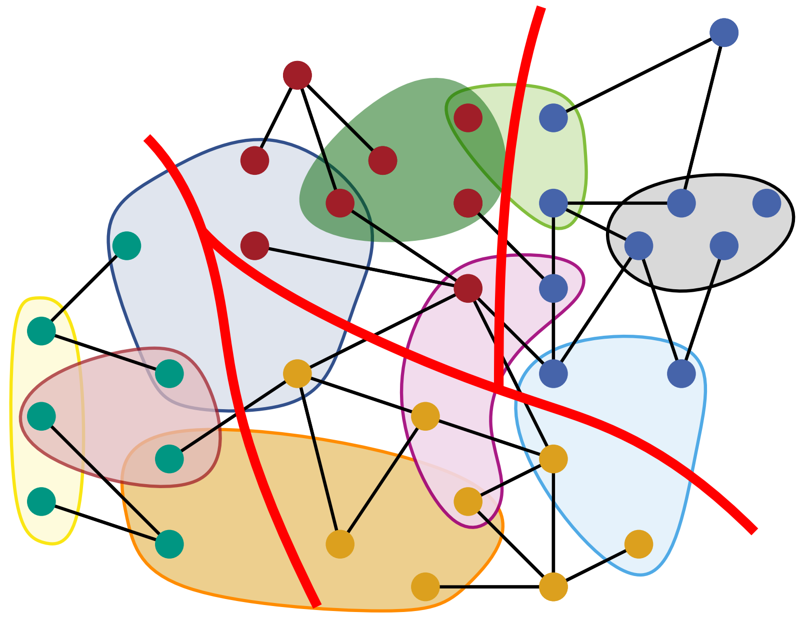
Partition hypergraph $H = (V, E, c : V \to \mathbb{R}_{>0}, \omega : E \to \mathbb{R}_{>0})$ into **k** disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that

- Blocks V_i are **roughly equal-sized**:

imbalance parameter

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

- **Objective** function on hyperedges is **minimized**



ε-Balanced Hypergraph Partitioning

Partition hypergraph $H = (V, E, c : V \to \mathbb{R}_{>0}, \omega : E \to \mathbb{R}_{>0})$ into **k** disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that

- Blocks V_i are **roughly equal-sized**:

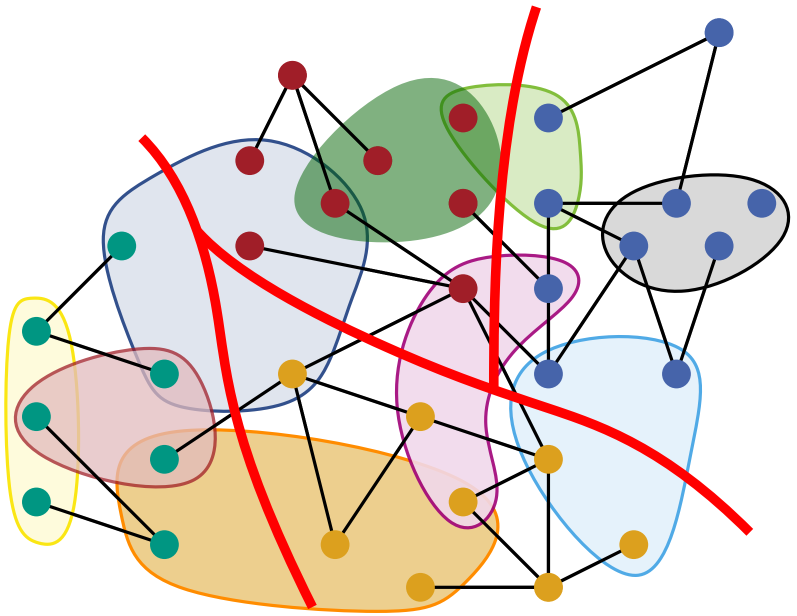
imbalance parameter

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

- **Objective** function on hyperedges is **minimized**

Common Objectives:

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



Institute of Theoretical Informatics
Algorithmics Group

ε-Balanced Hypergraph Partitioning

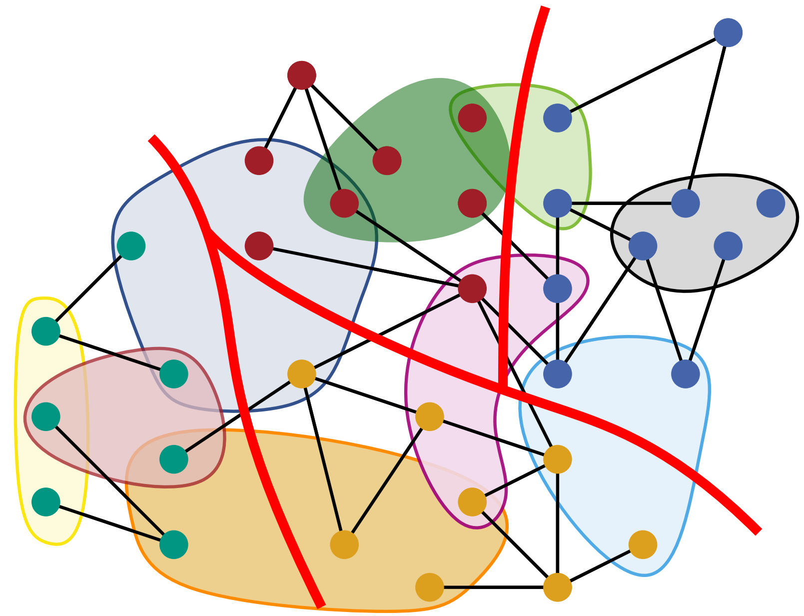
Partition hypergraph $H = (V, E, c : V \rightarrow \mathbb{R}_{>0}, \omega : E \rightarrow \mathbb{R}_{>0})$ into **k** disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that

- Blocks V_i are **roughly equal-sized**:

imbalance parameter

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

- **Objective** function on hyperedges is **minimized**



Common Objectives:

- cut: $\sum_{e \in \text{Cut}} \omega(e)$
- Connectivity: $\sum_{e \in \text{cut}} (\lambda - 1)\, \omega(e)$

ε-Balanced Hypergraph Partitioning

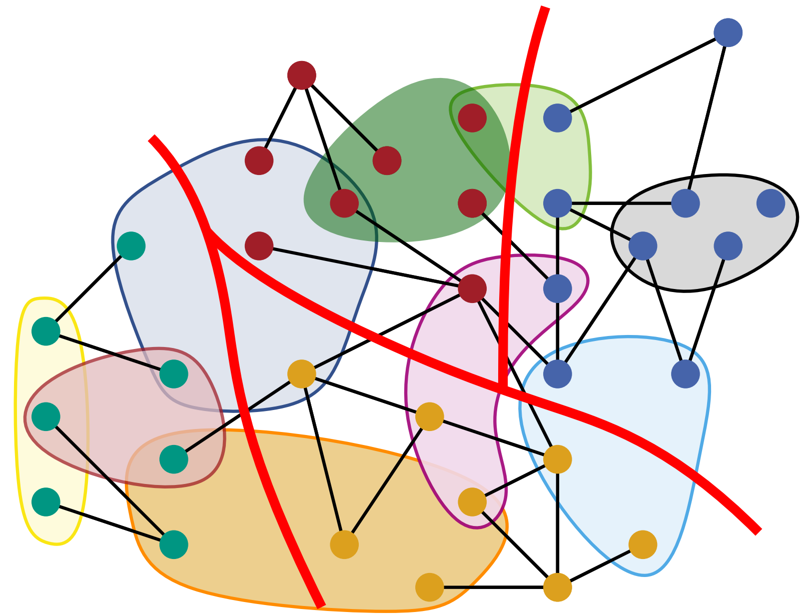
Partition hypergraph $H = (V, E, c : V \to \mathbb{R}_{>0}, \omega : E \to \mathbb{R}_{>0})$ into **k** disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that

- Blocks V_i are **roughly equal-sized**:

imbalance parameter

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

- **Objective** function on hyperedges is **minimized**



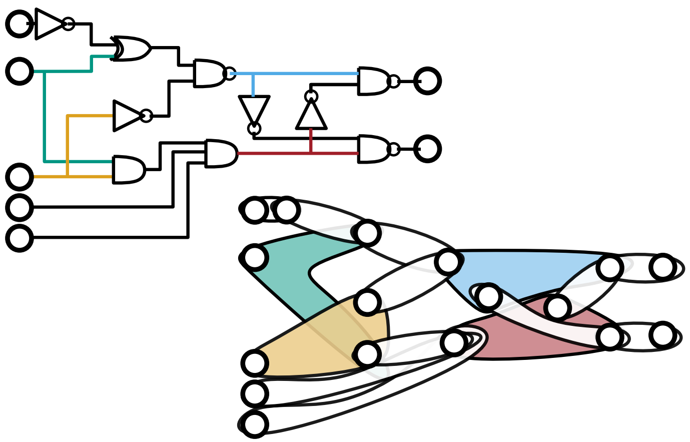
Common Objectives:

- cut: $\sum_{e \in \text{Cut}} \omega(e)$
- Connectivity: $\sum_{e \in \text{cut}} (\lambda - 1)\, \omega(e)$

 # blocks connected by *e*

Institute of Theoretical Informatics
Algorithmics Group

Applications



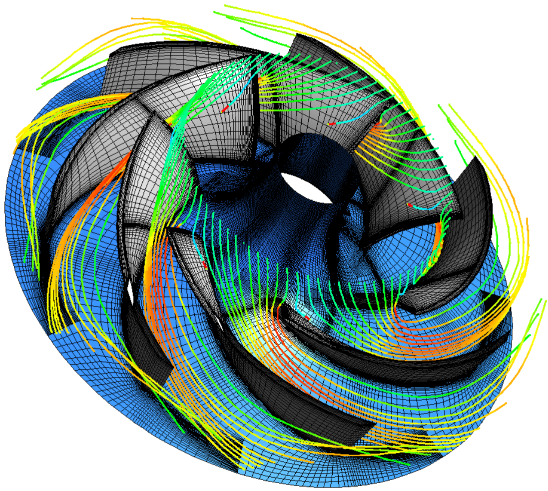
VLSI Design

Warehouse Optimization

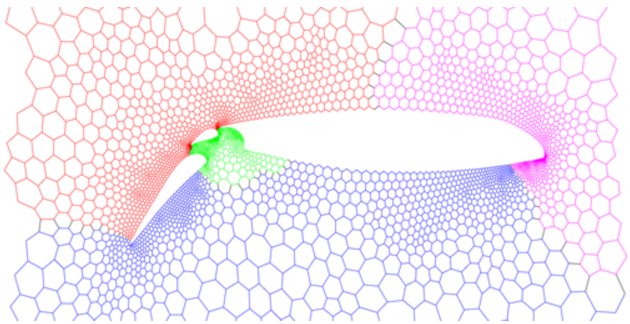
[Martin Grandjean, via Wikimedia Commons]

Complex Networks

Route Planning



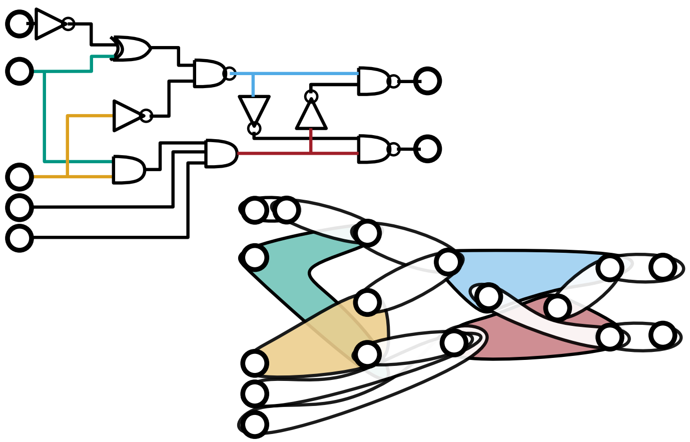
Simulation



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

Scientific Computing

Applications



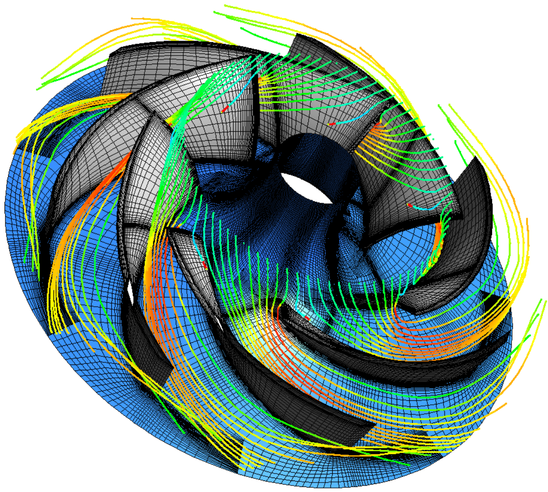
VLSI Design

Warehouse Optimization

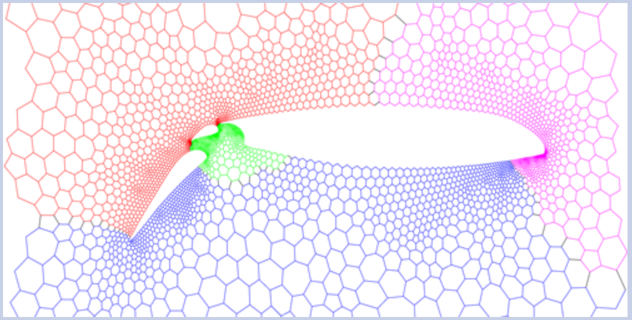
[Martin Grandjean, via Wikimedia Commons]

Complex Networks

Route Planning



Simulation



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

Scientific Computing

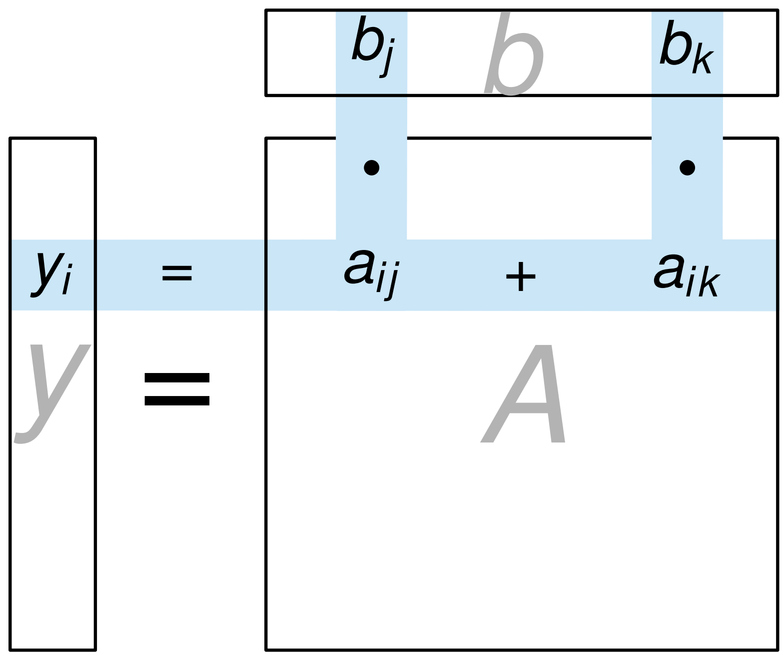
Institute of Theoretical Informatics
Algorithmics Group

Parallel Sparse-Matrix Vector Product (SpM$\times$V)

Karlsruhe Institute of Technology

[Catalyürek,Aykanat]

$$y = A\,b$$



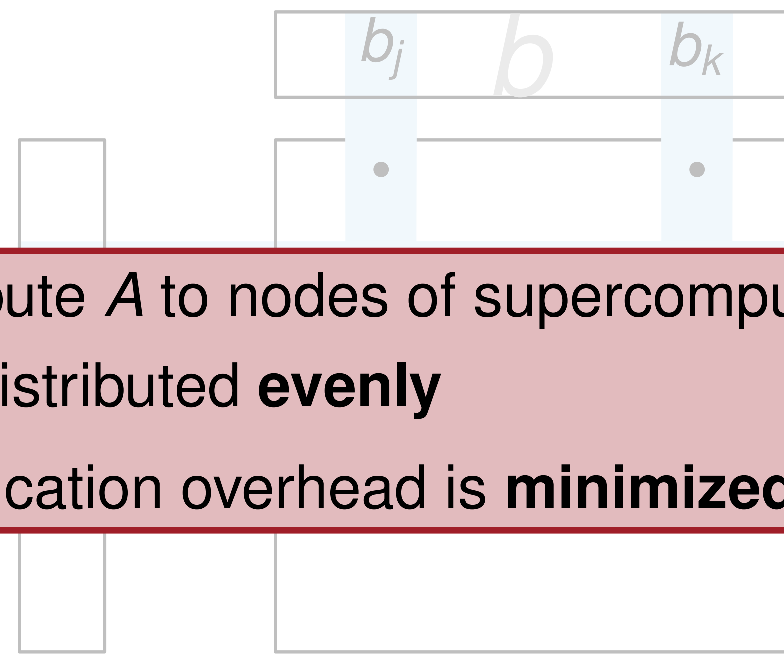
Setting:

- Repeated SpM$\times$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 (SpM×V)

[Catalyürek,Aykanat]

$$y = A\,b$$



Task: distribute A to nodes of supercomputer such that

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

Setting:

- Repeated SpM×V on supercomputer
- A is large ⇒ distribute on multiple nodes
- Symmetric partitioning ⇒ y & b divided conformally with A

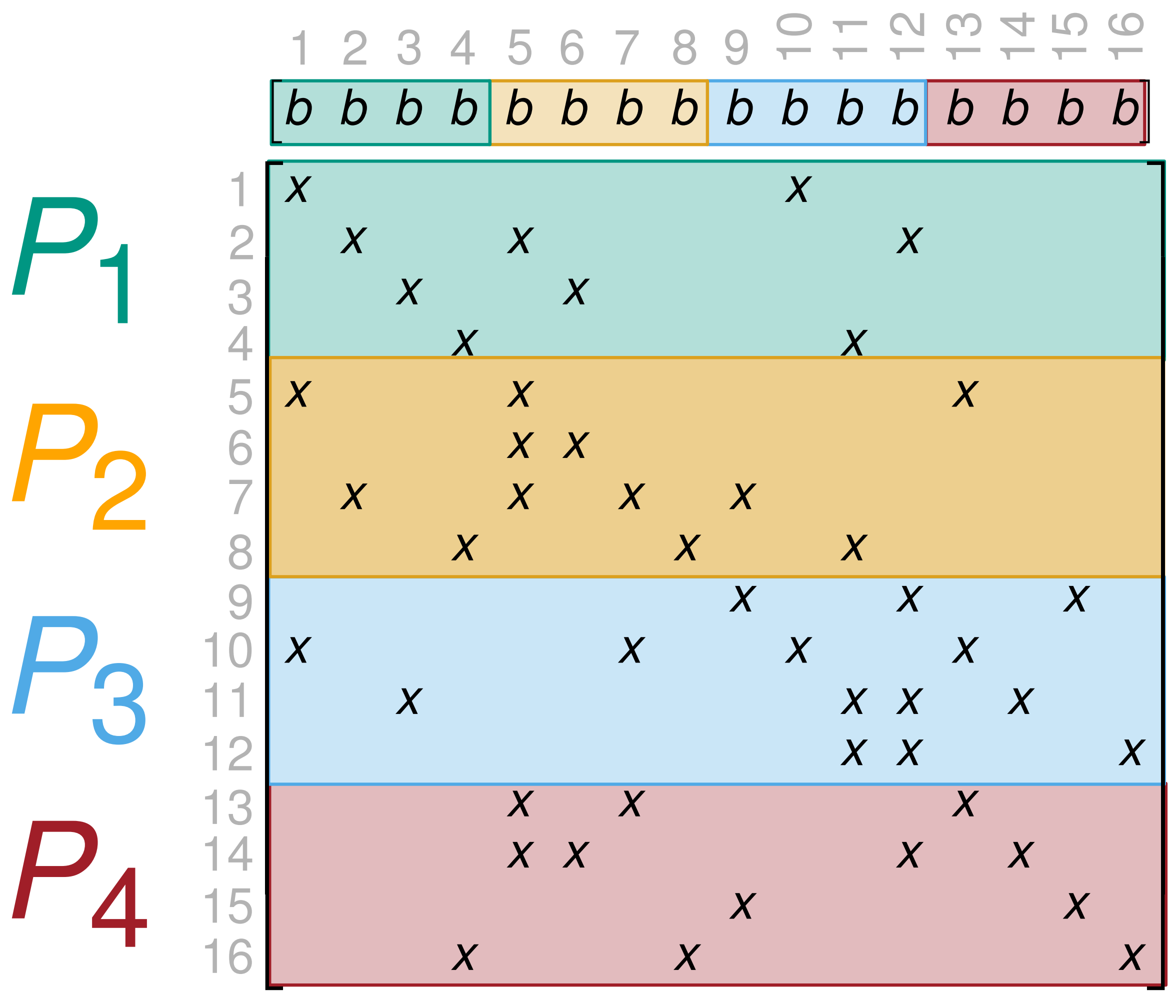
Naive Approach: Rowwise Decomposition

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

	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*	*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*

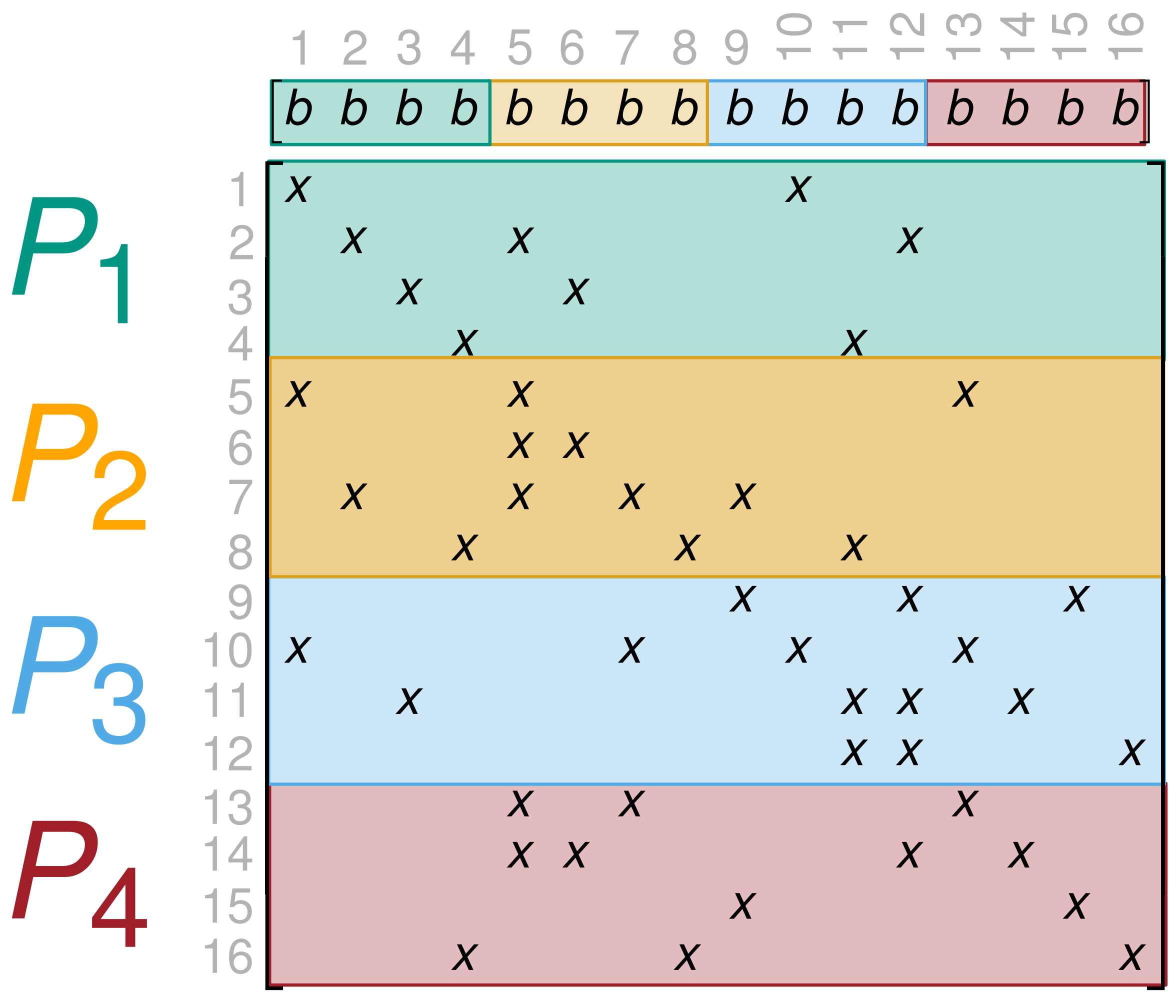
Naive Approach: Rowwise Decomposition

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



Naive Approach: Rowwise Decomposition

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



Load Balancing?

$\Rightarrow 9$

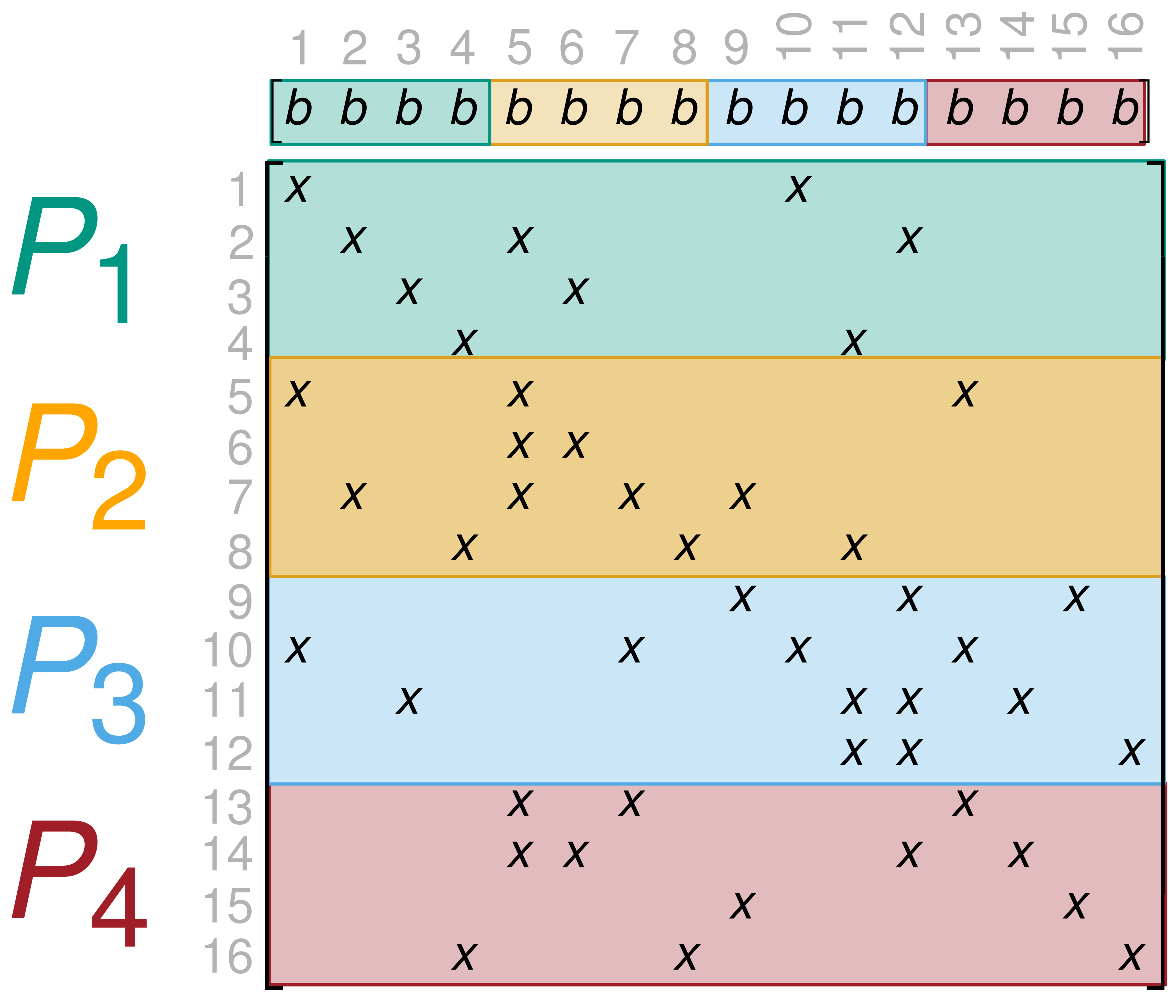
$\Rightarrow 12$

$\Rightarrow 14$

$\Rightarrow 12$

Naive Approach: Rowwise Decomposition

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



Load Balancing?

$\Rightarrow 9$

$\Rightarrow 12$

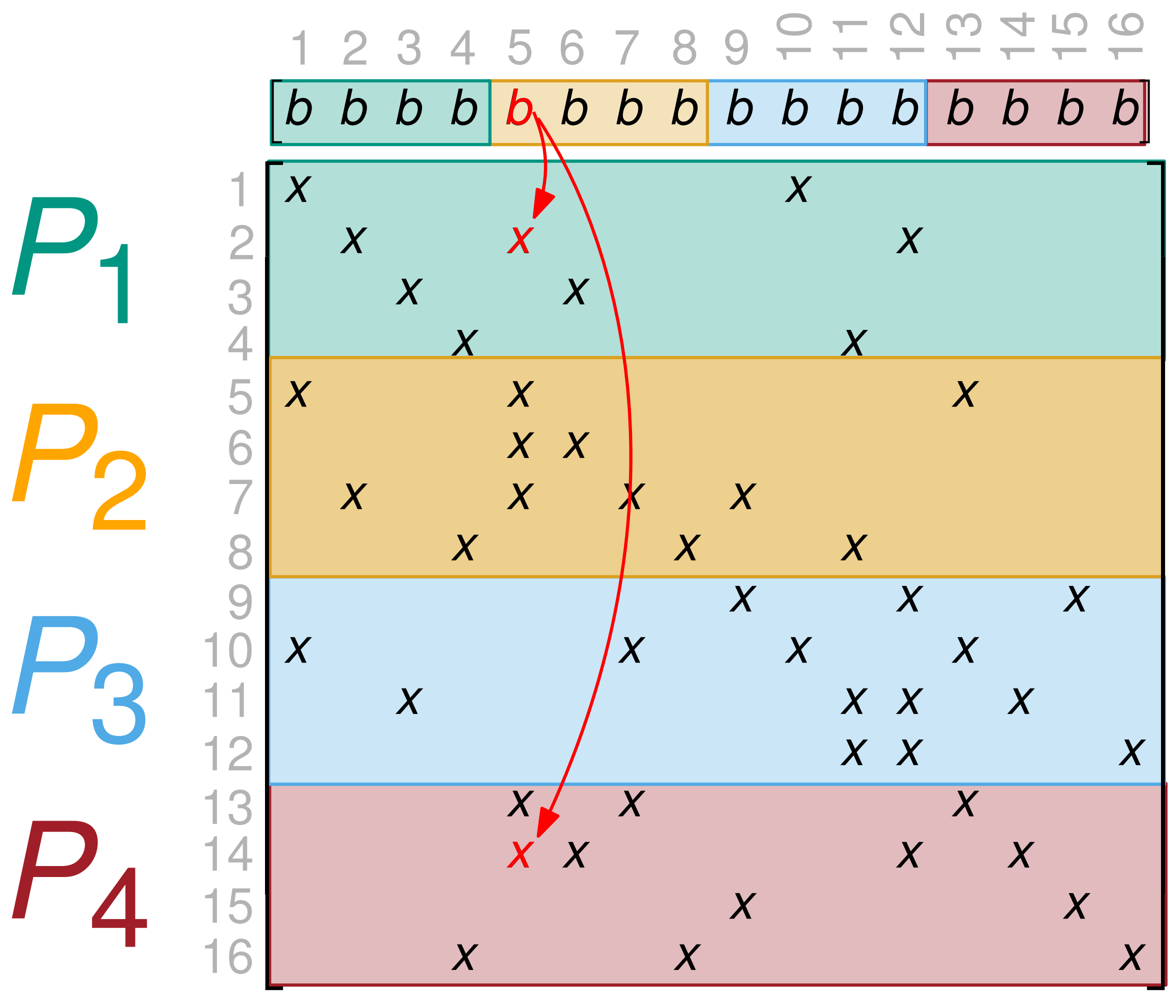
$\Rightarrow 14$

$\Rightarrow 12$

Commuication Volume?

Naive Approach: Rowwise Decomposition

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



Load Balancing?

$\Rightarrow 9$

$\Rightarrow 12$

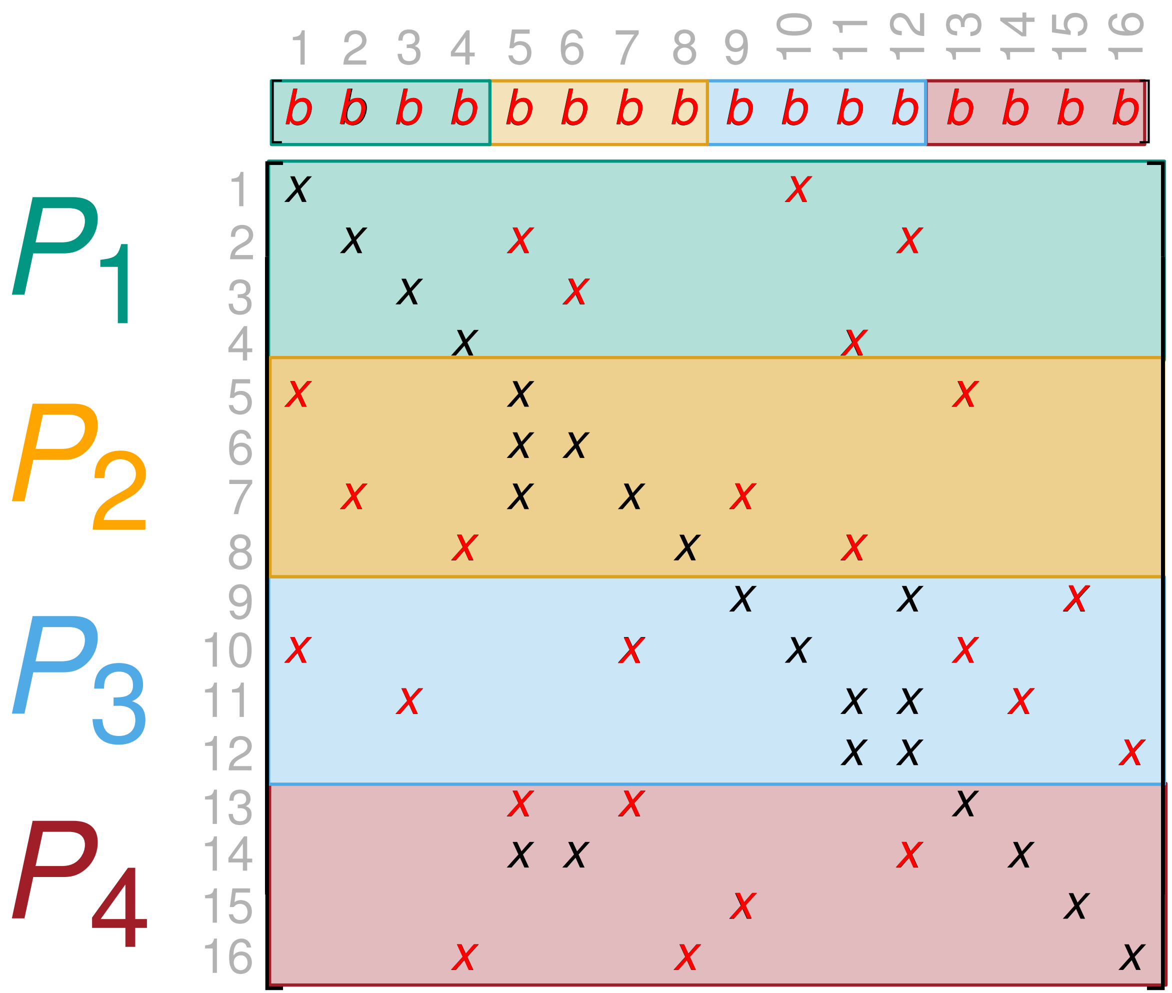
$\Rightarrow 14$

$\Rightarrow 12$

Commuication Volume?

Naive Approach: Rowwise Decomposition

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



Load Balancing?

$\Rightarrow$ 9

$\Rightarrow$ 12

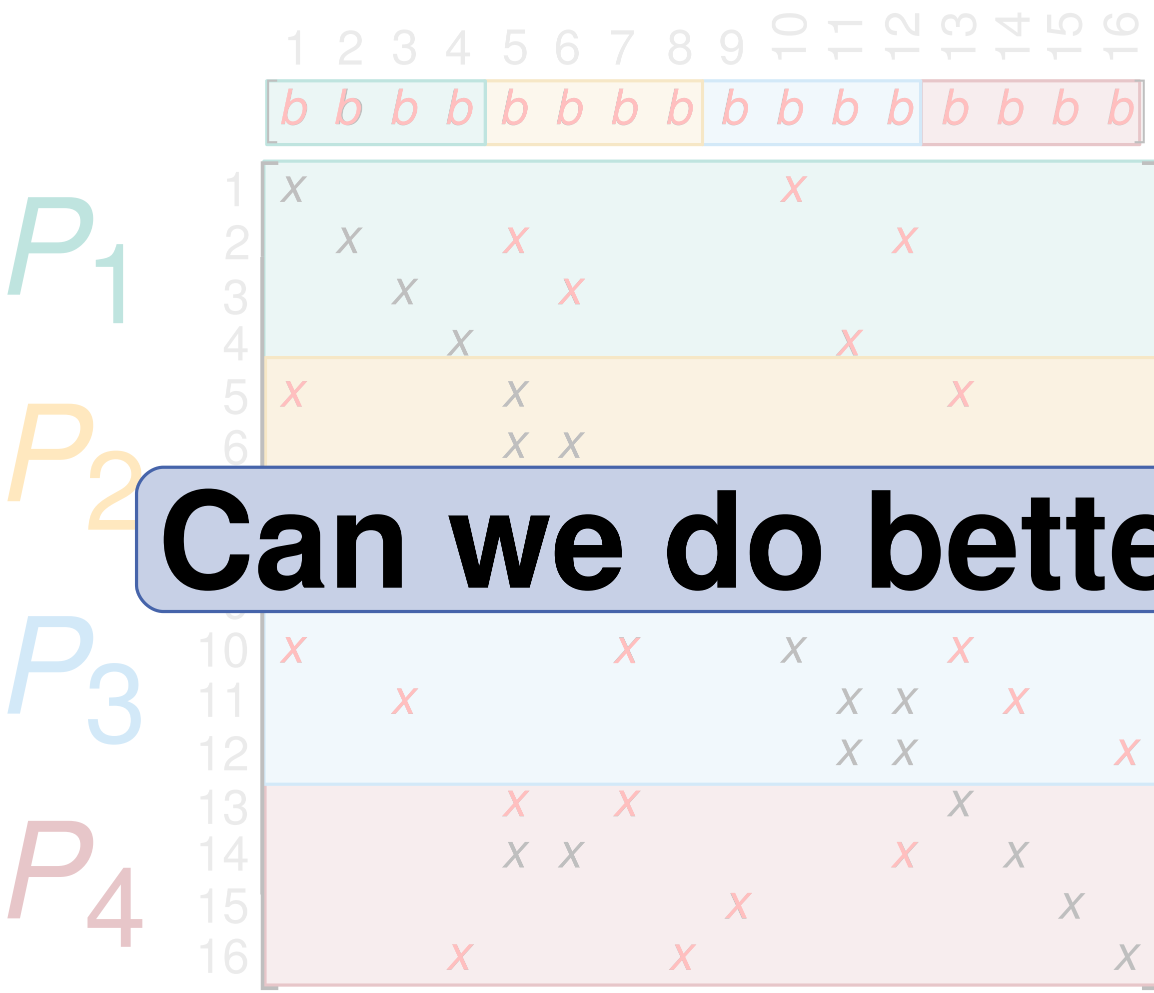
$\Rightarrow$ 14

$\Rightarrow$ 12

Commuication Volume? $\Rightarrow$ 24 entries!

Naive Approach: Rowwise Decomposition

$A \in \mathbb{R}^{16\times16}$



Load Balancing?

$\Rightarrow$ 9

$\Rightarrow$ 12

$\Rightarrow$ 14

$\Rightarrow$ 12

Can we do better?

Commuication Volume? $\Rightarrow$ 24 entries!

From SpM×V to Hypergraph Partitioning

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

- One vertex per row:

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

- One hyperedge per column:

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

	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 SpM×V to Hypergraph Partitioning

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

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

$v_i \in V_R$ **:**

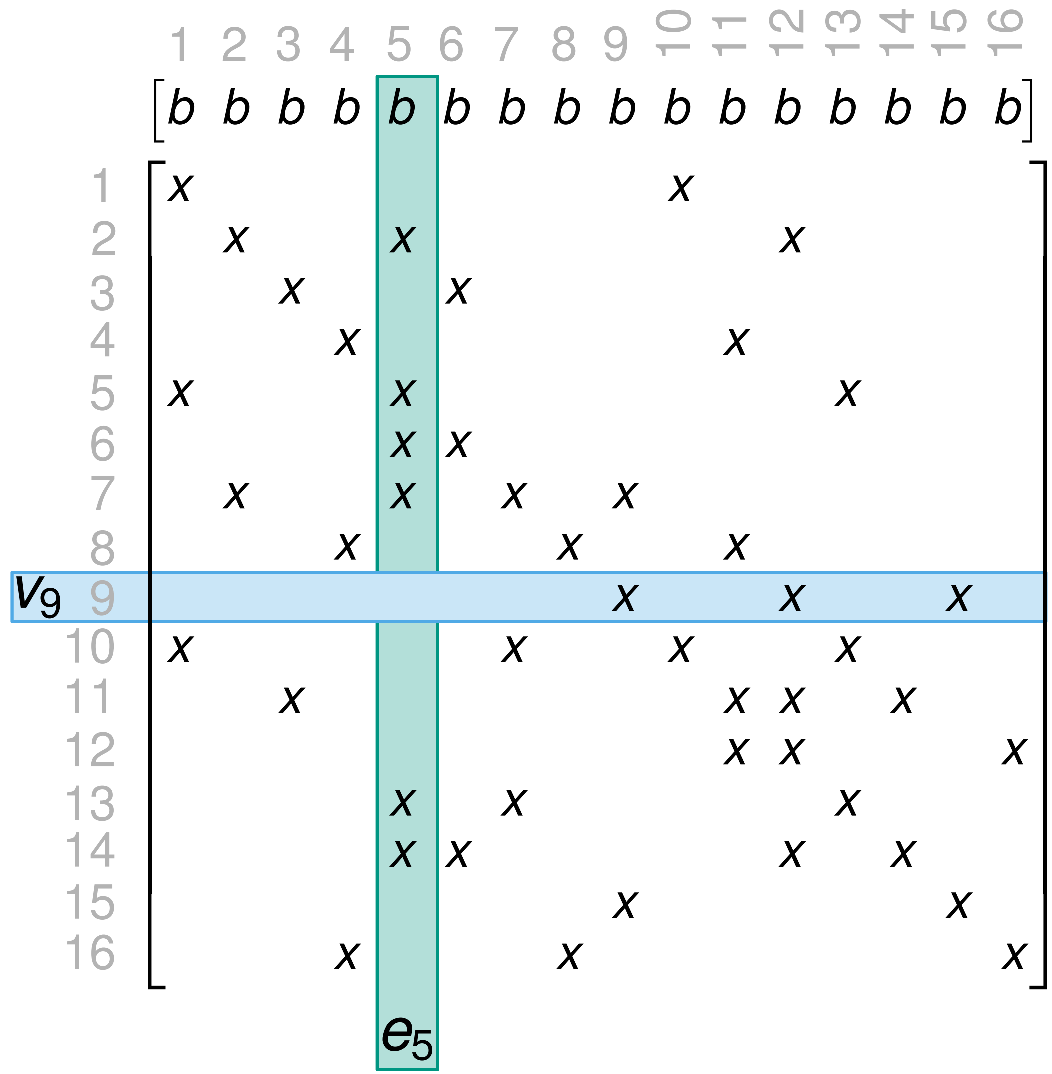
- Inner product of row *i* with *b*
- $\Rightarrow c(v_i) :=$ # nonzeros

	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					
v_9 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 SpM×V to Hypergraph Partitioning

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

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



$v_i \in V_R$:

- Inner product of row i with b
- $\Rightarrow c(v_i) :=$ # nonzeros

$e_j \in E_C$:

- Set of vertices that need b_j

From SpM×V to Hypergraph Partitioning

$A \in \mathbb{R}^{16\times16} \Rightarrow H = (V_R, E_C)$

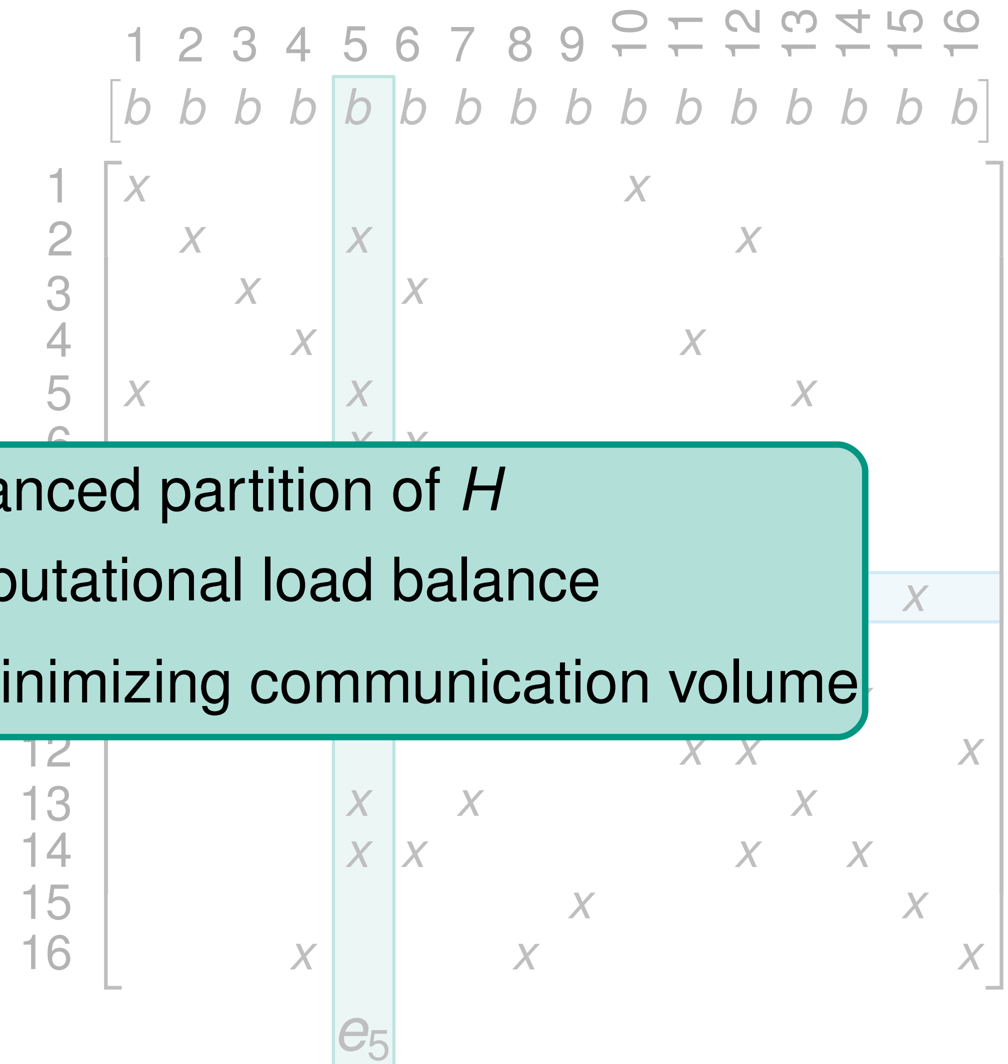
- One vertex per row:
  $\Rightarrow V_R = \{v_1, v_2, \ldots, v_{16}\}$
- One hyperedge per column:
  $\Rightarrow E_C = \{e_1, e_2, \ldots, e_{16}\}$

$v_i \in$

- In
- $\Rightarrow c(v_i) :=$ # nonzeros

$e_j \in E_C$:

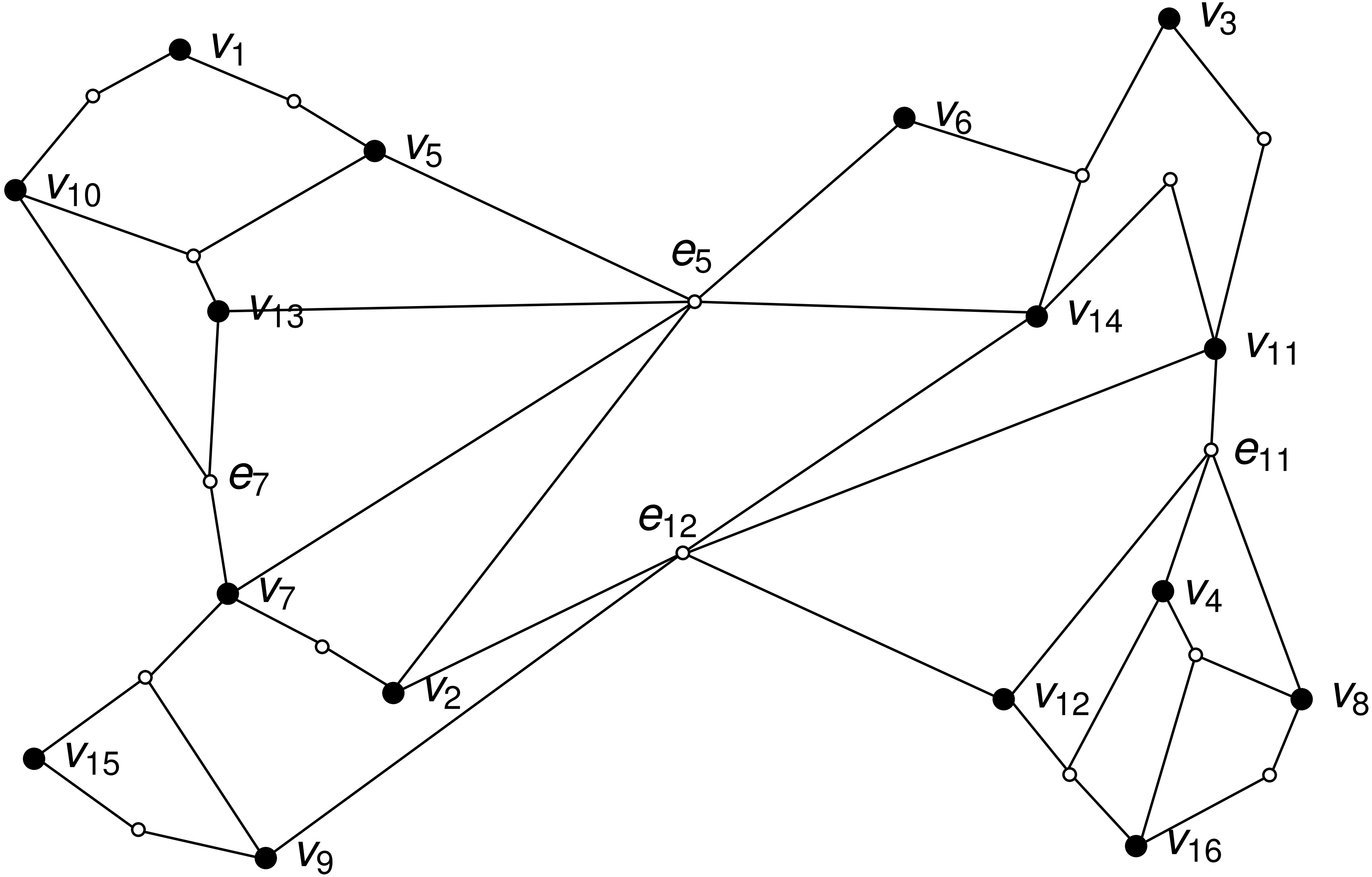
- Set of vertices that need b_j



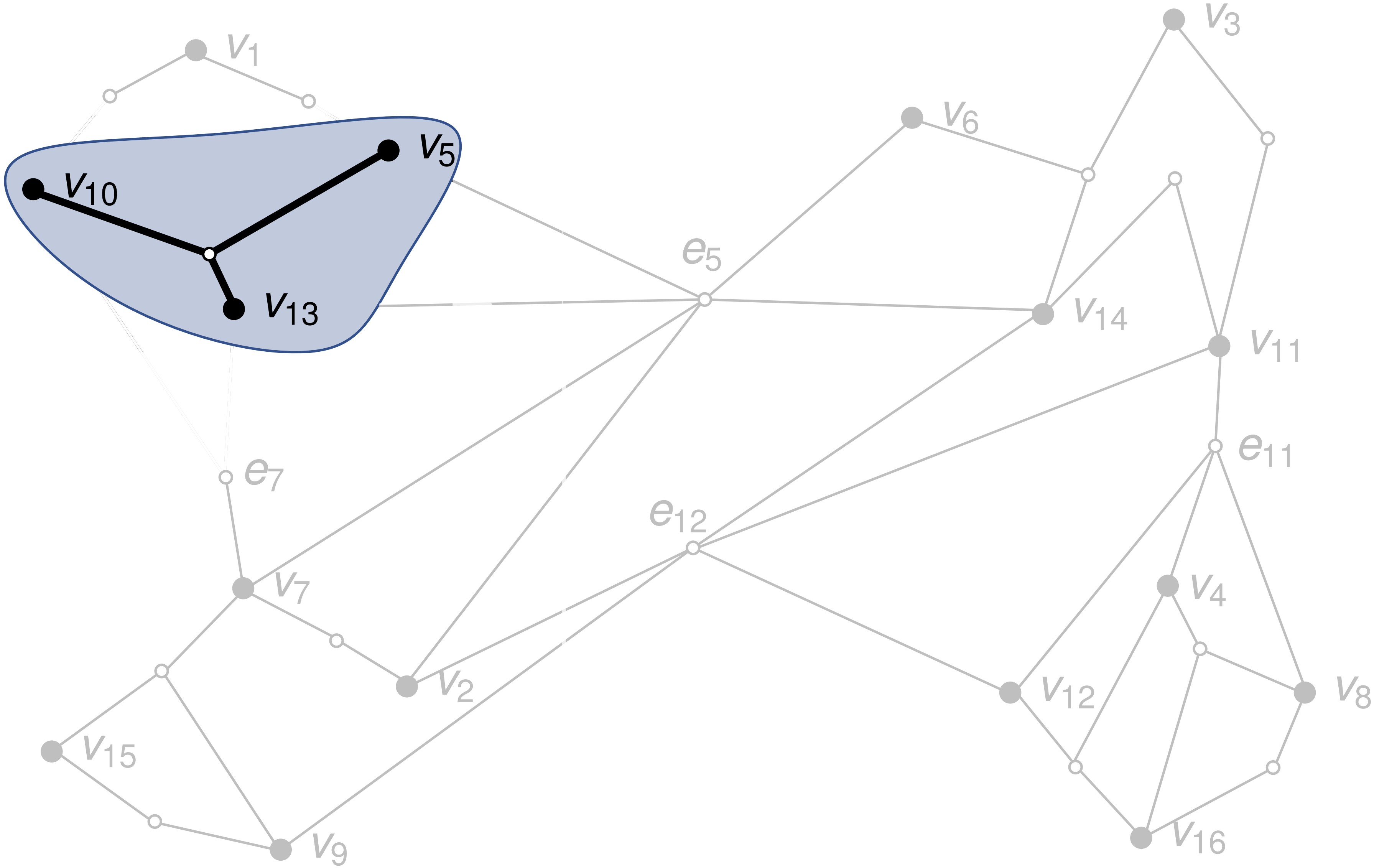
Solution: ε-balanced partition of H

- Balanced partition ⇝ computational load balance
- Small $(\lambda - 1)$-cutsize ⇝ minimizing communication volume

From SpM×V to Hypergraph Partitioning

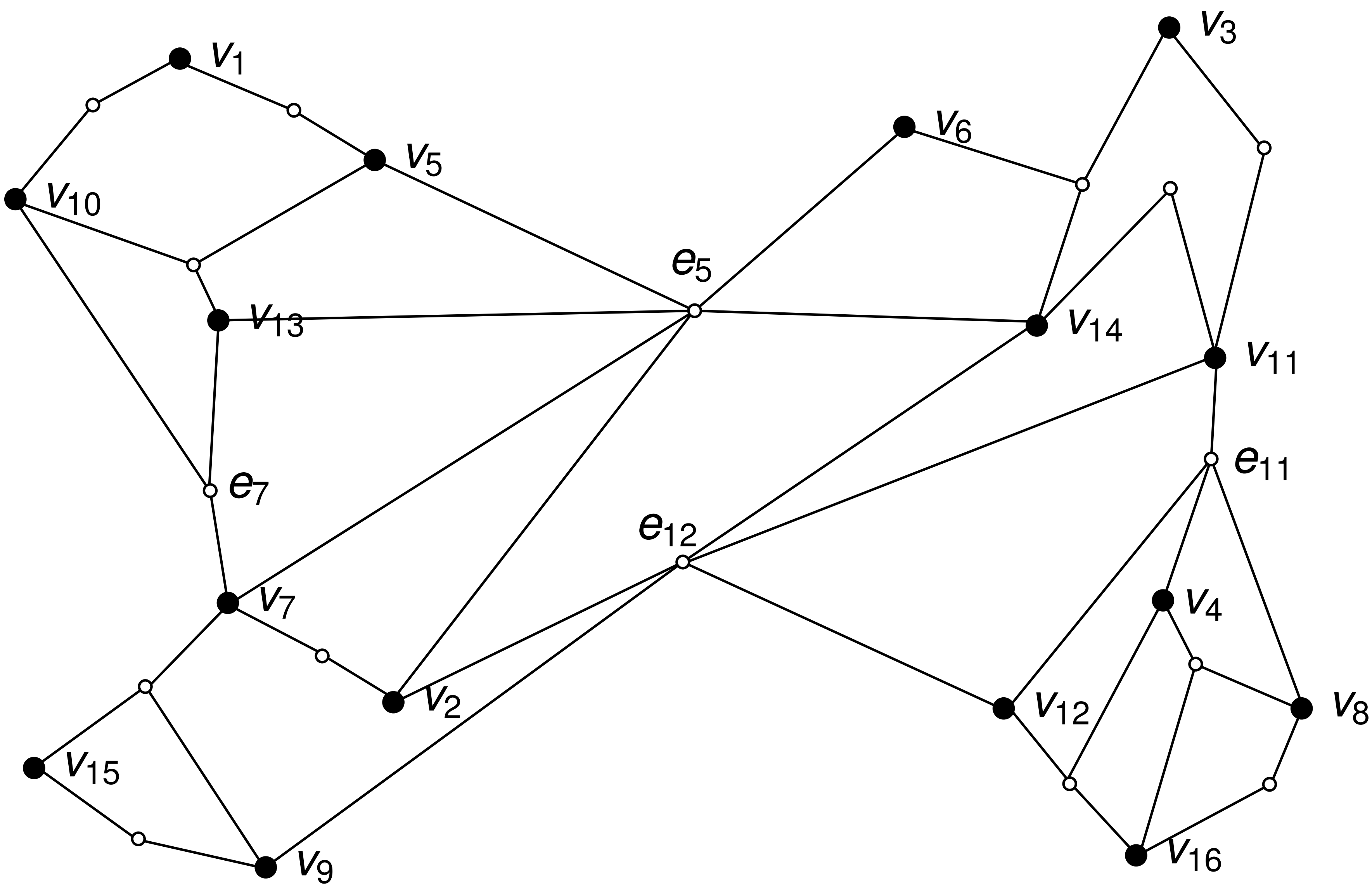


From SpM×V to Hypergraph Partitioning

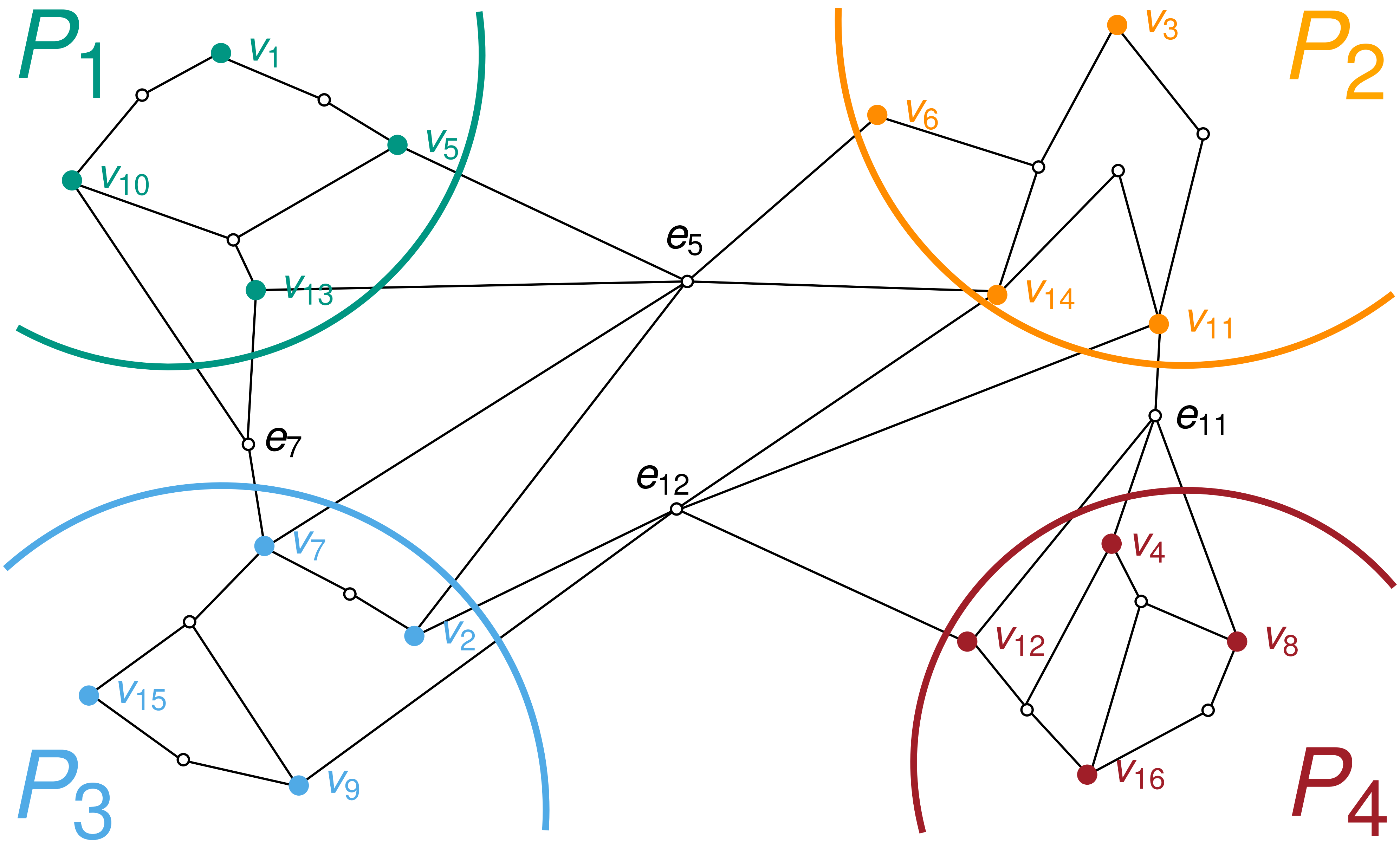


Institute of Theoretical Informatics
Algorithmics Group

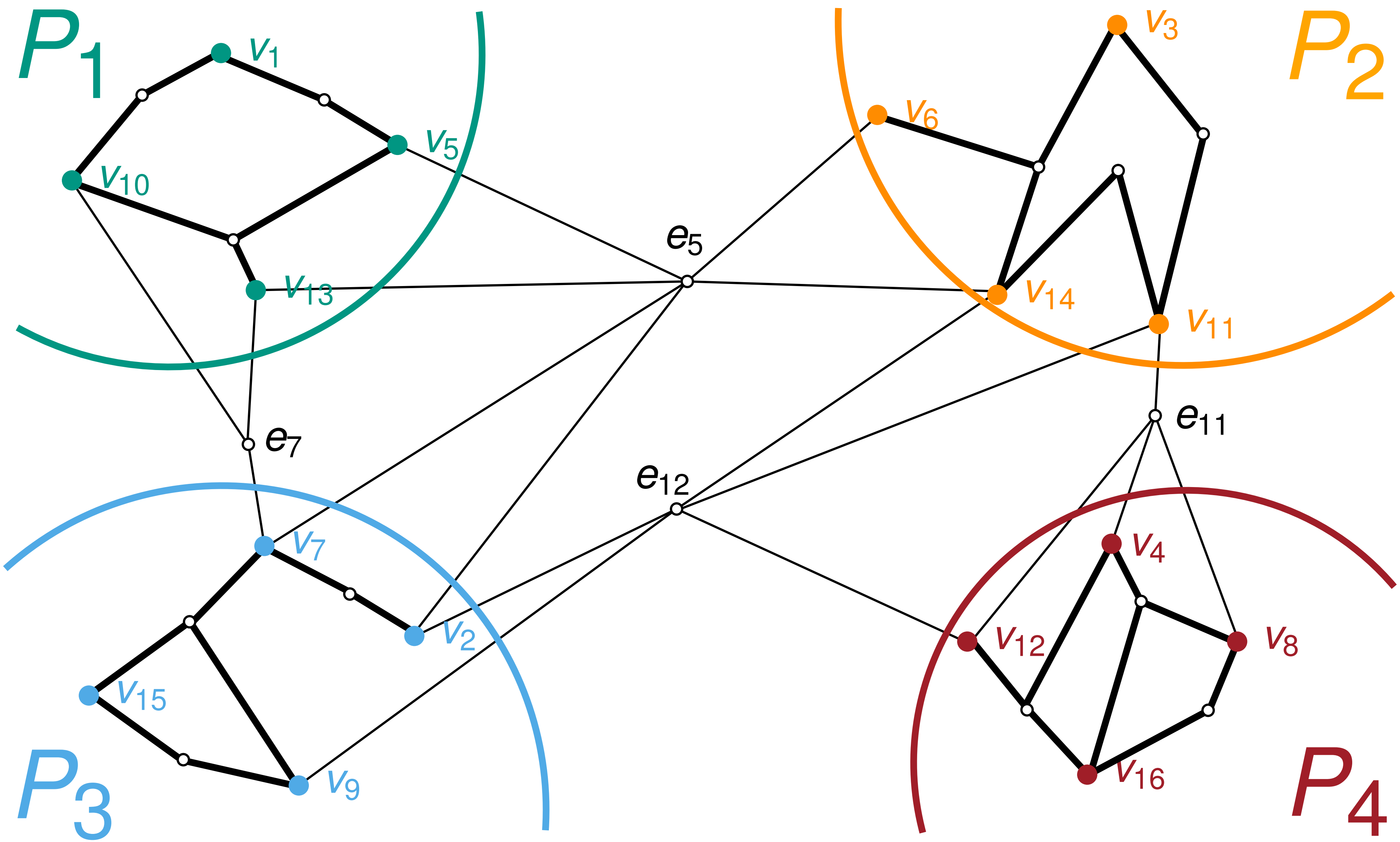
From SpM×V to Hypergraph Partitioning



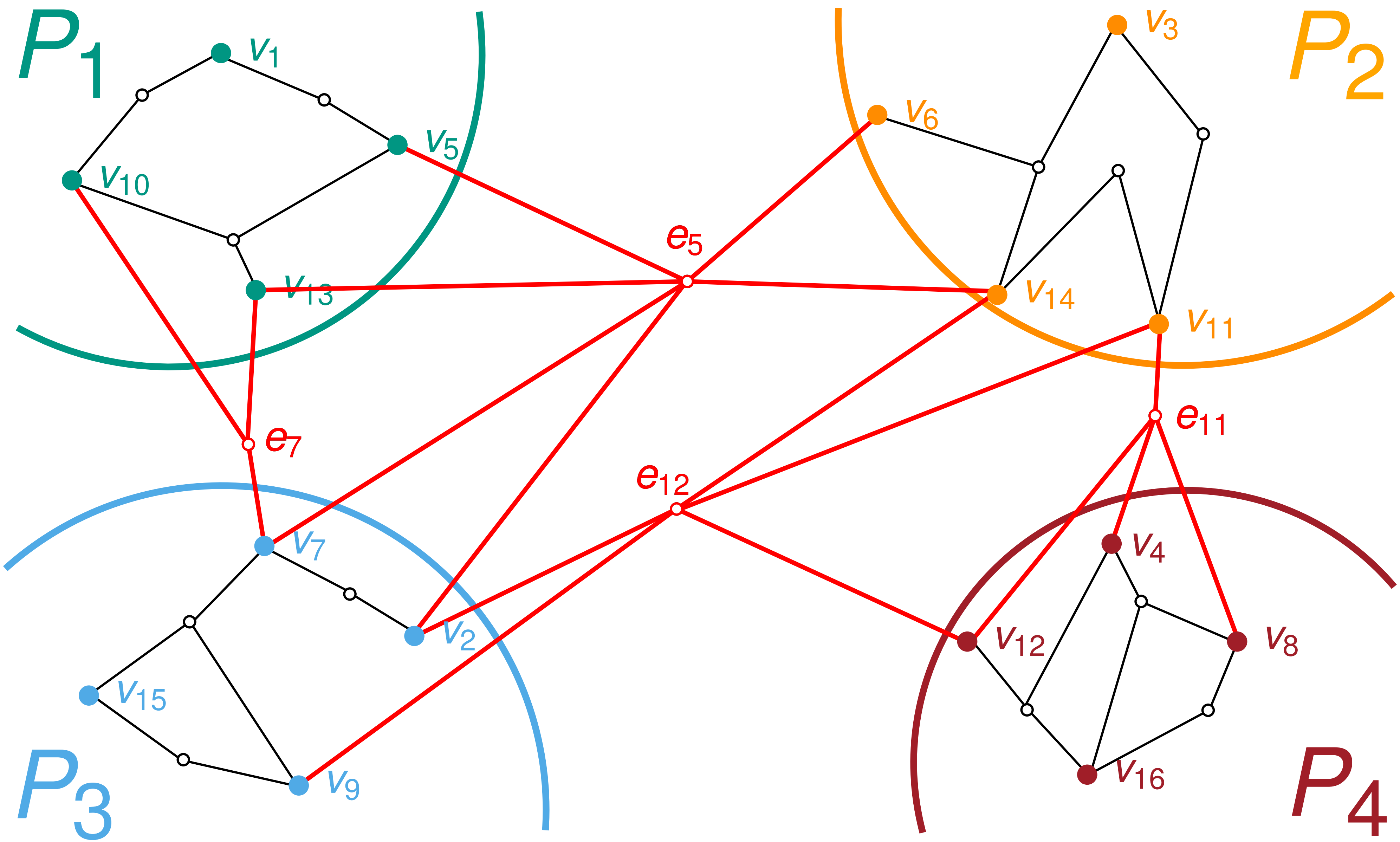
From SpM×V to Hypergraph Partitioning



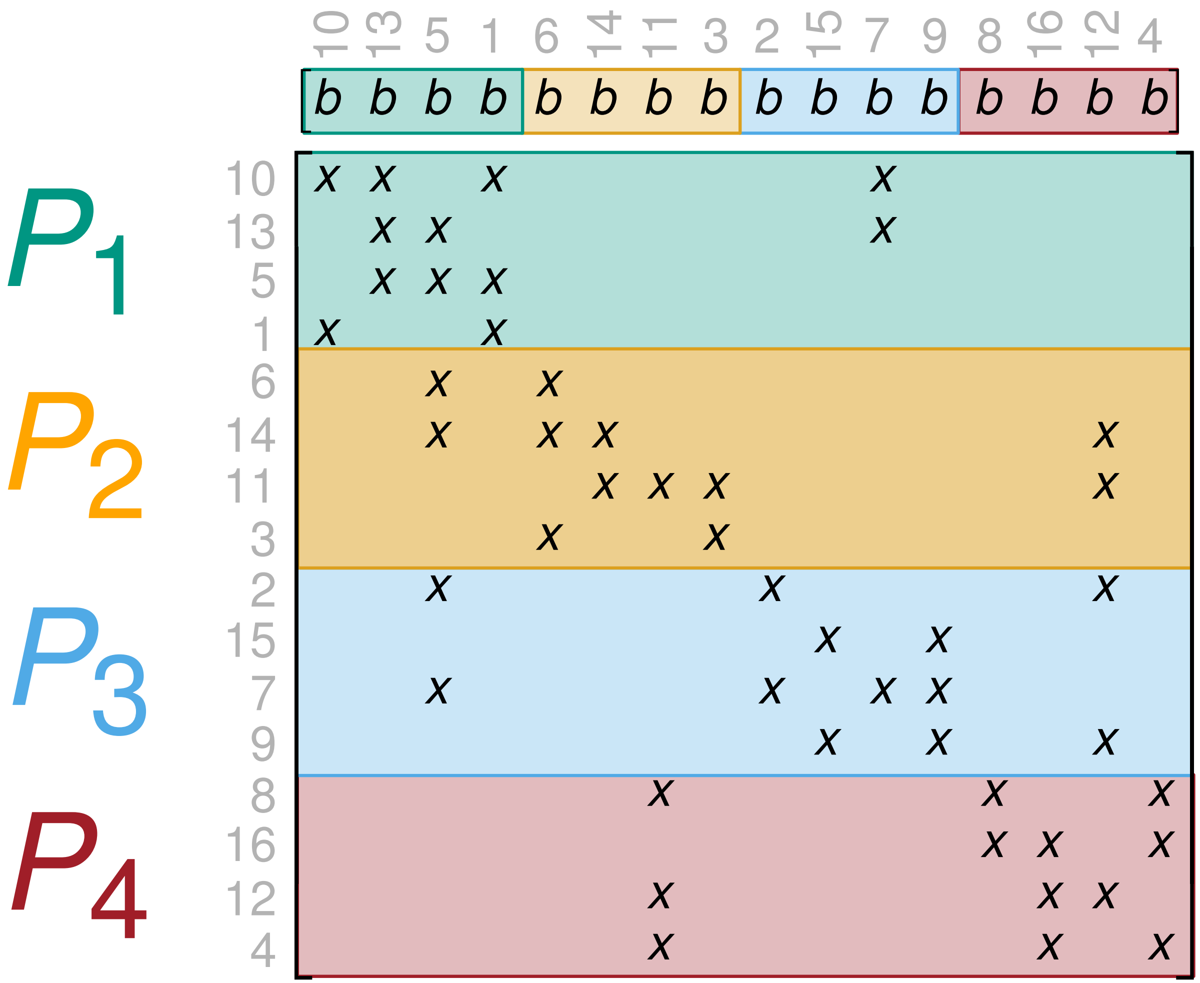
From SpM×V to Hypergraph Partitioning



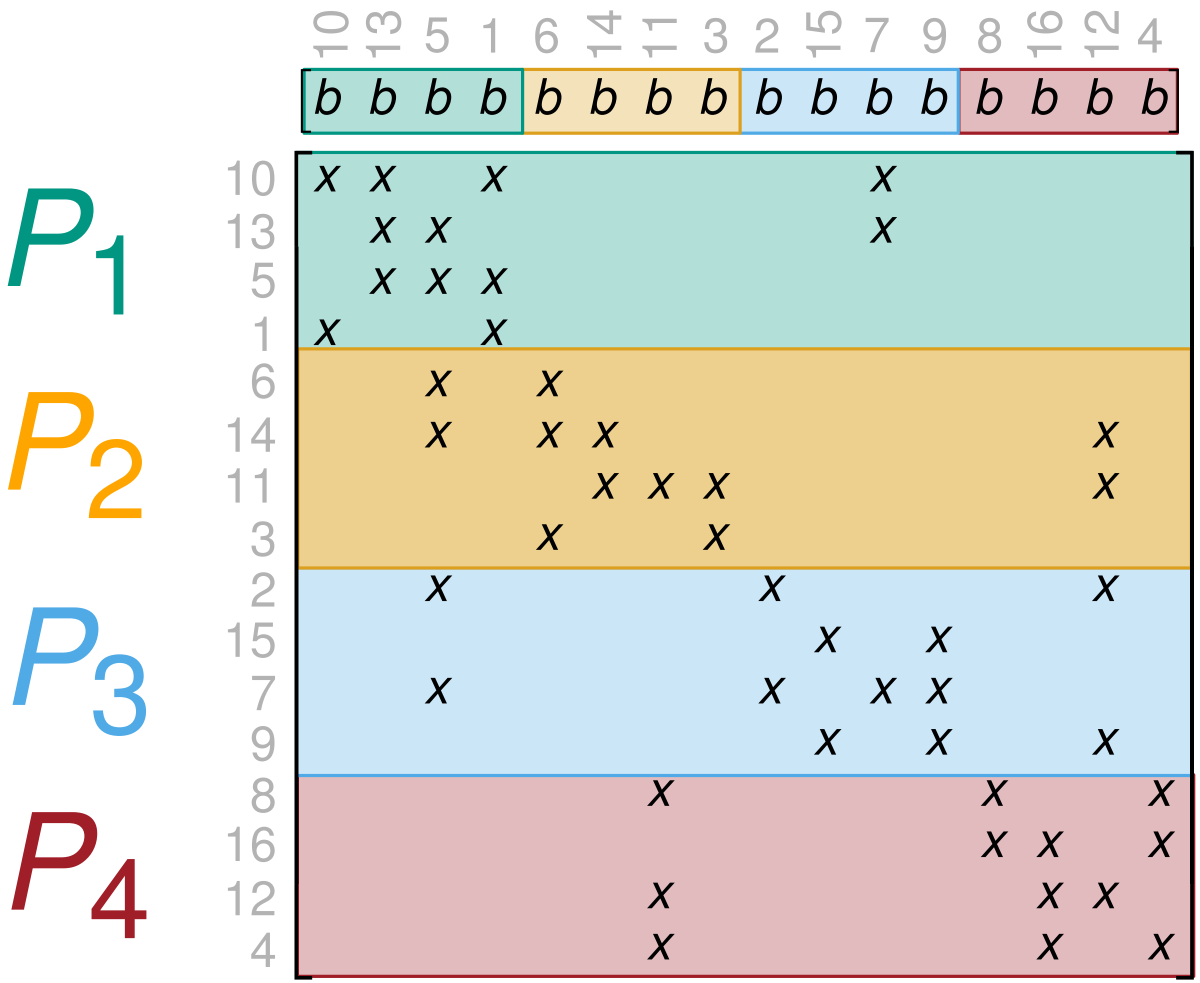
From SpM×V to Hypergraph Partitioning



From Hypergraph Partitioning to SpM×V



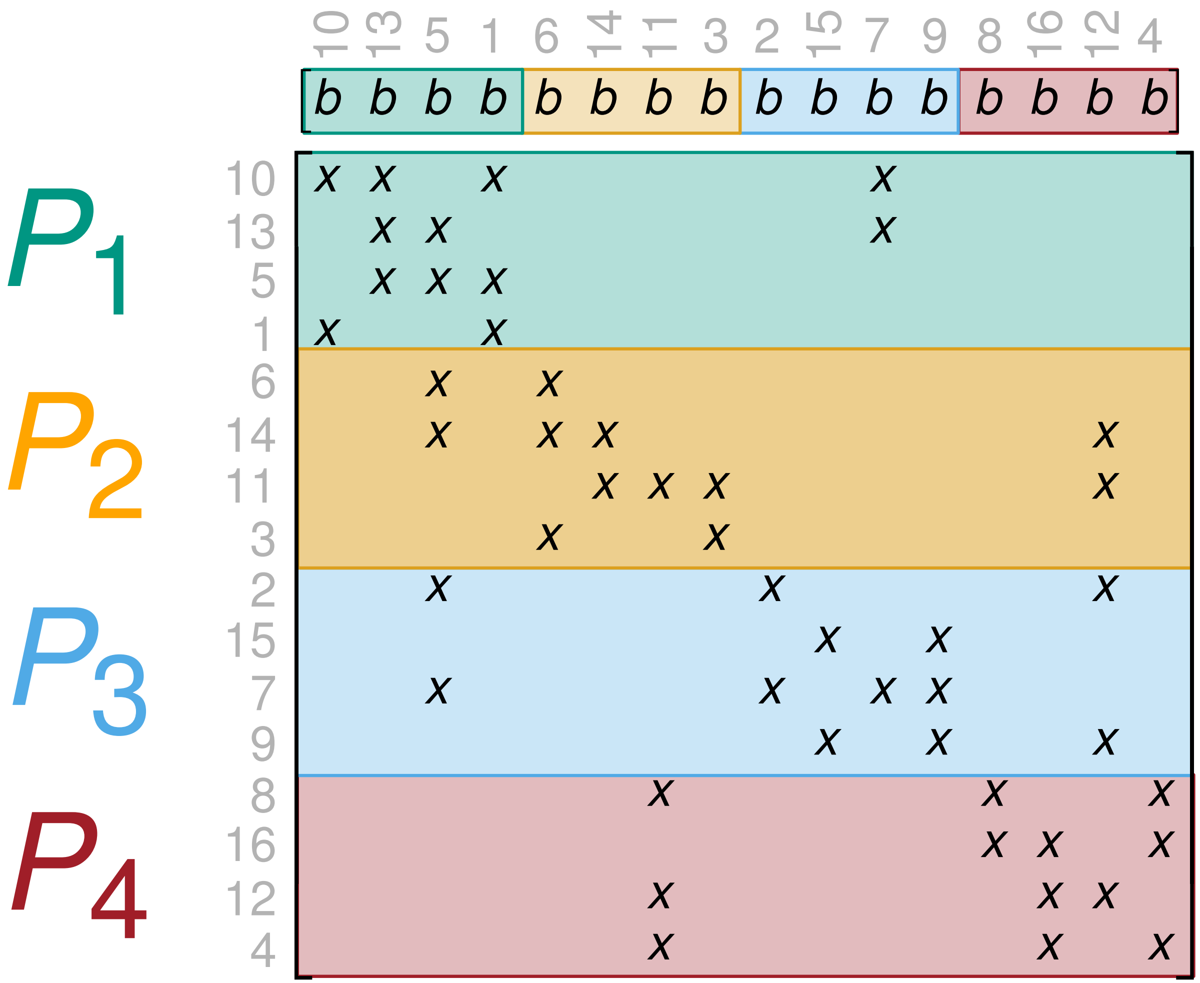
From Hypergraph Partitioning to SpM×V



Load Balancing?

Institute of Theoretical Informatics
Algorithmics Group

From Hypergraph Partitioning to SpM×V



Load Balancing?

$\Rightarrow$ 12

$\Rightarrow$ 12

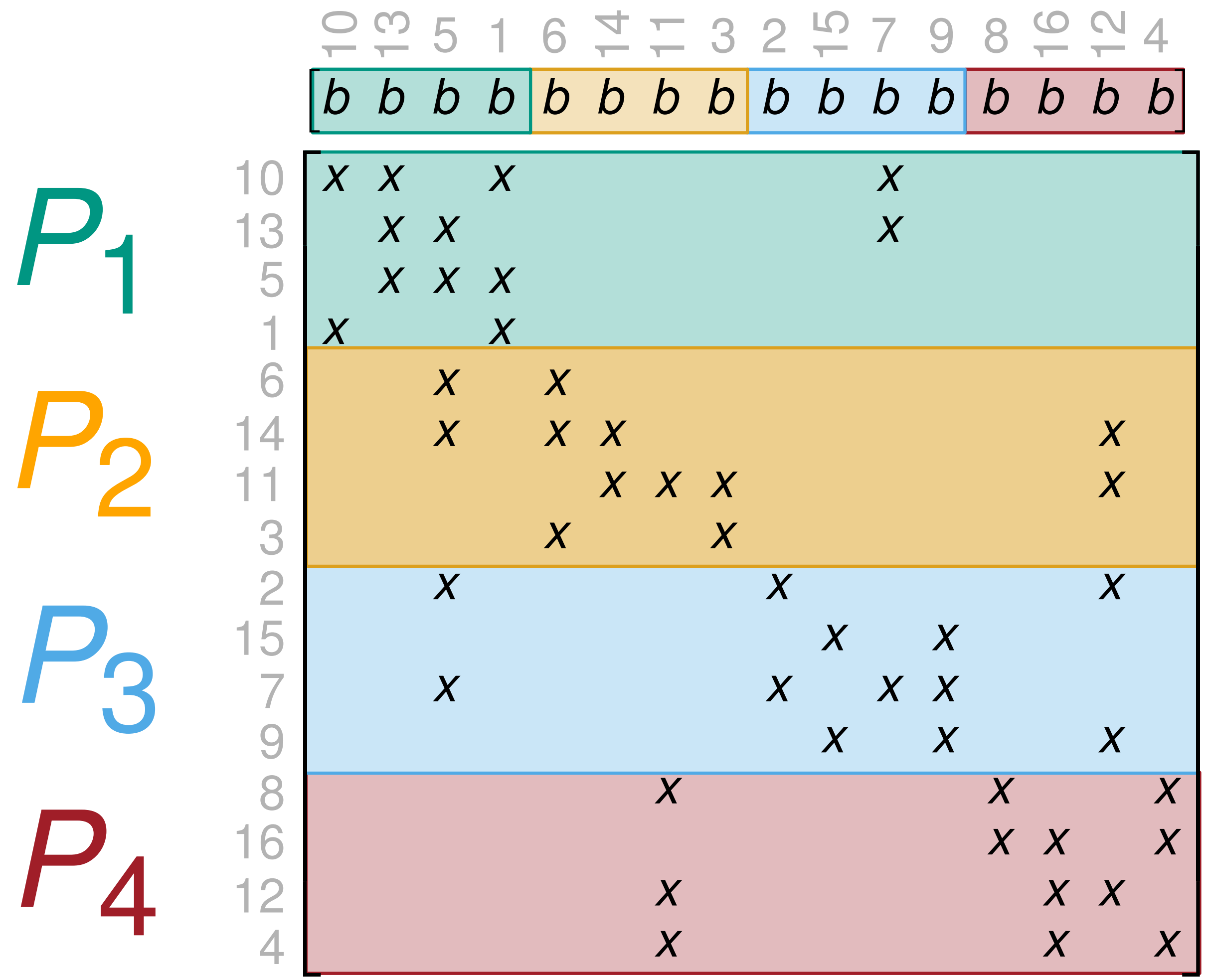
$\Rightarrow$ 12

$\Rightarrow$ 12

From Hypergraph Partitioning to SpM×V

Karlsruhe Institute of Technology

Where are the cut-hyperedges?



Load Balancing?

⇒ 12

⇒ 12

⇒ 12

⇒ 12

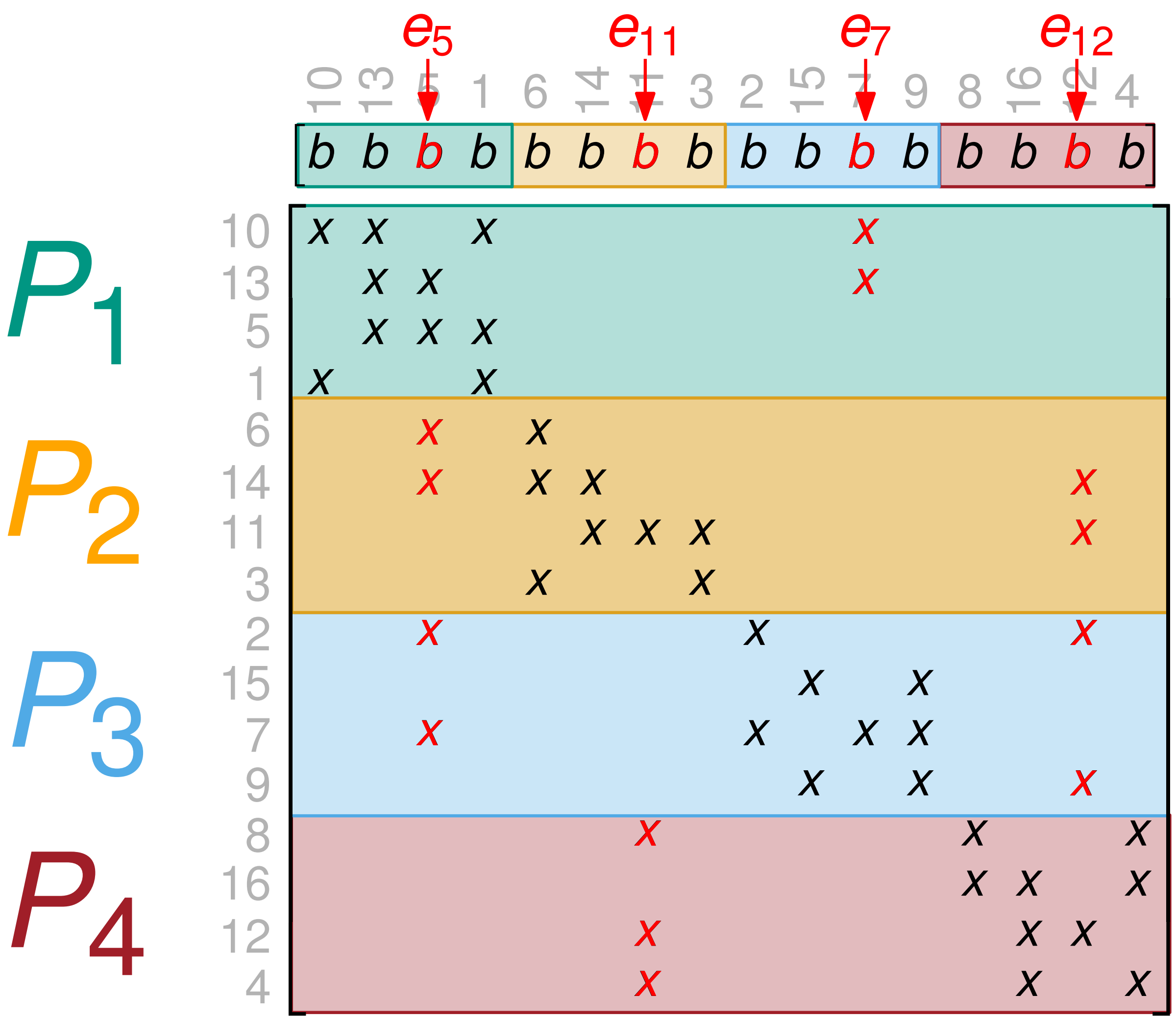
Commuication Volume?

Institute of Theoretical Informatics
Algorithmics Group

From Hypergraph Partitioning to SpM×V

Karlsruhe Institute of Technology

Where are the cut-hyperedges?



Load Balancing?

⇒ 12

⇒ 12

⇒ 12

⇒ 12

Commuication Volume? ⇒ 6 entries!

How does Hypergraph Partitioning work?

How does

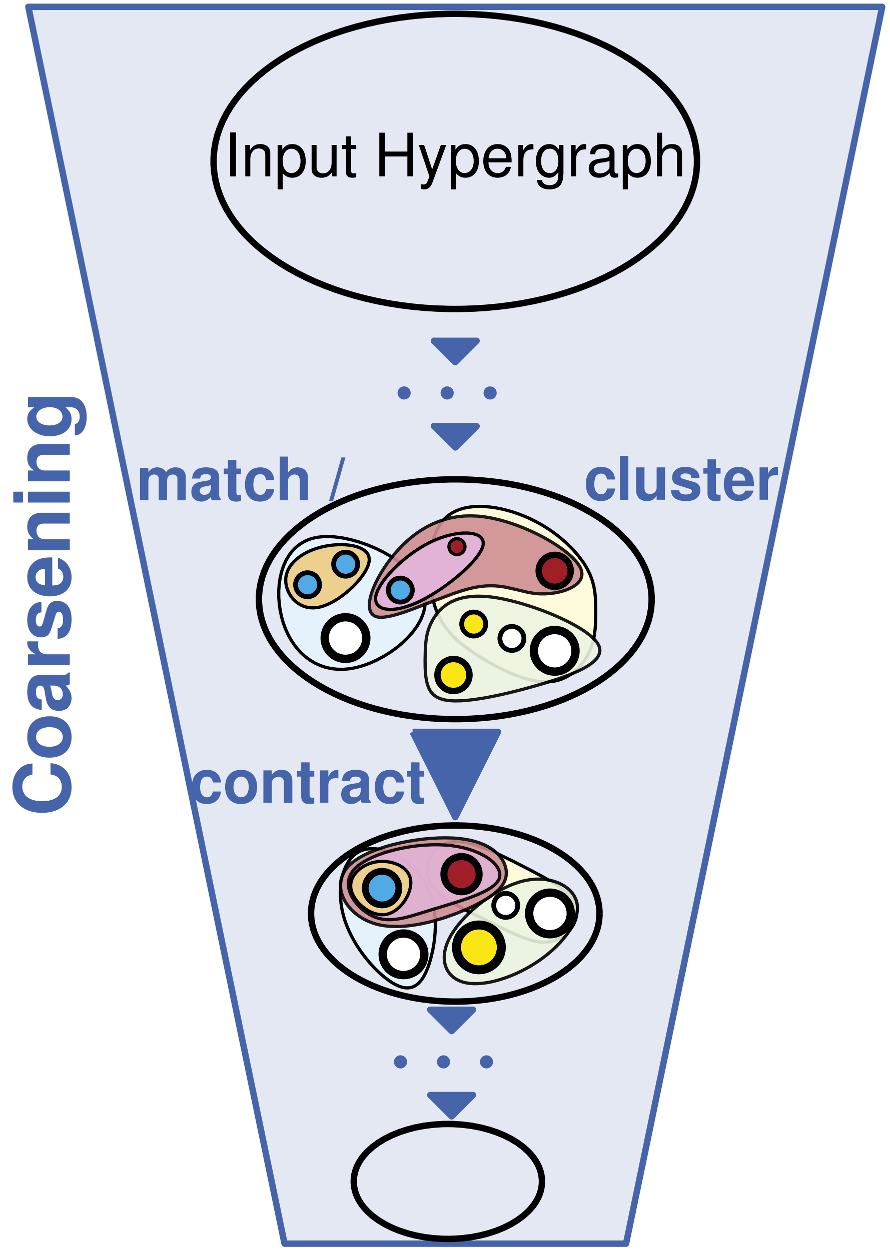
Bad News:

- Hypergraph Partitioning is $\mathbf{NP}$-hard
- Even finding good approximate solutions for graphs is $\mathbf{NP}$-hard

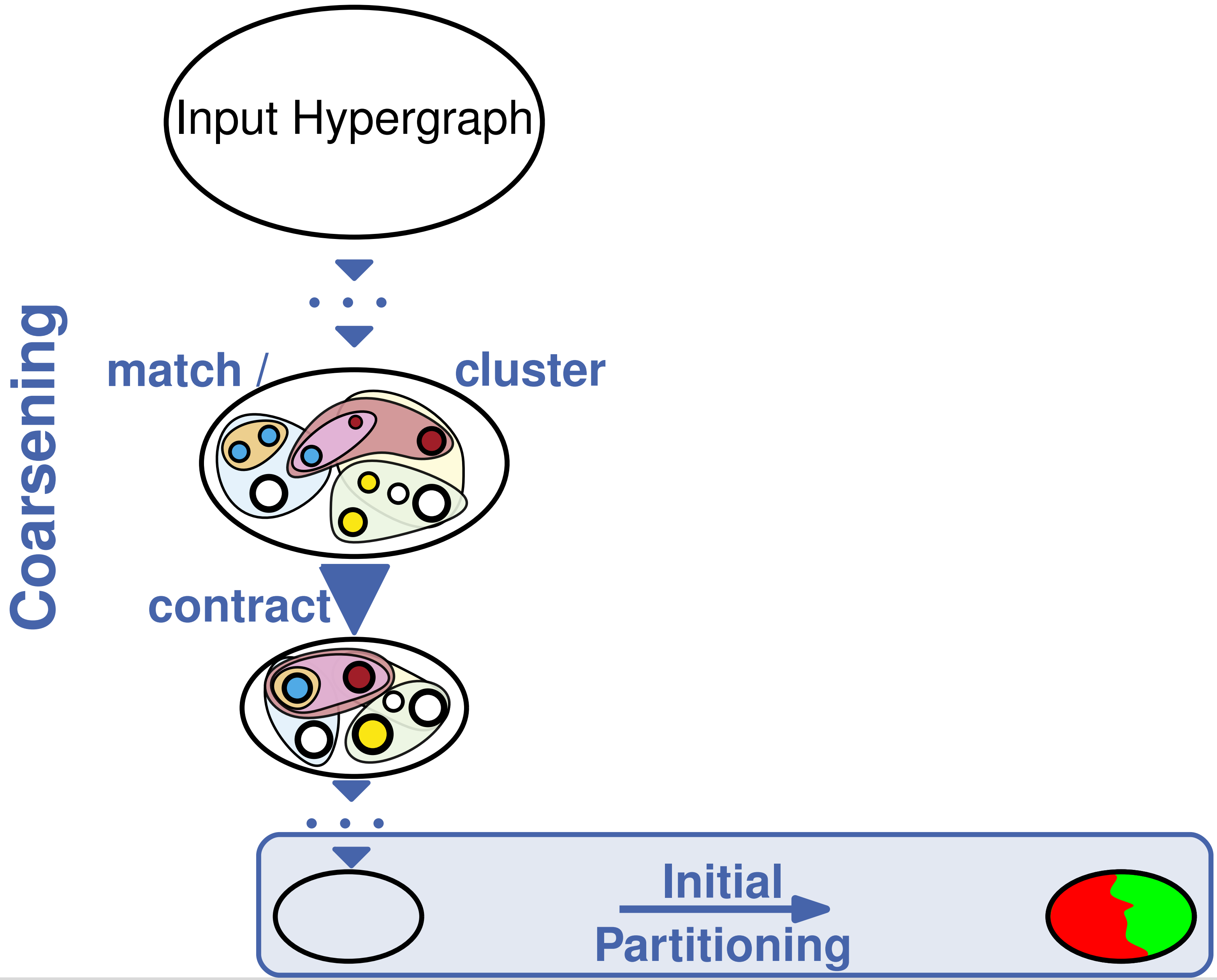
work?

Institute of Theoretical Informatics
Algorithmics Group

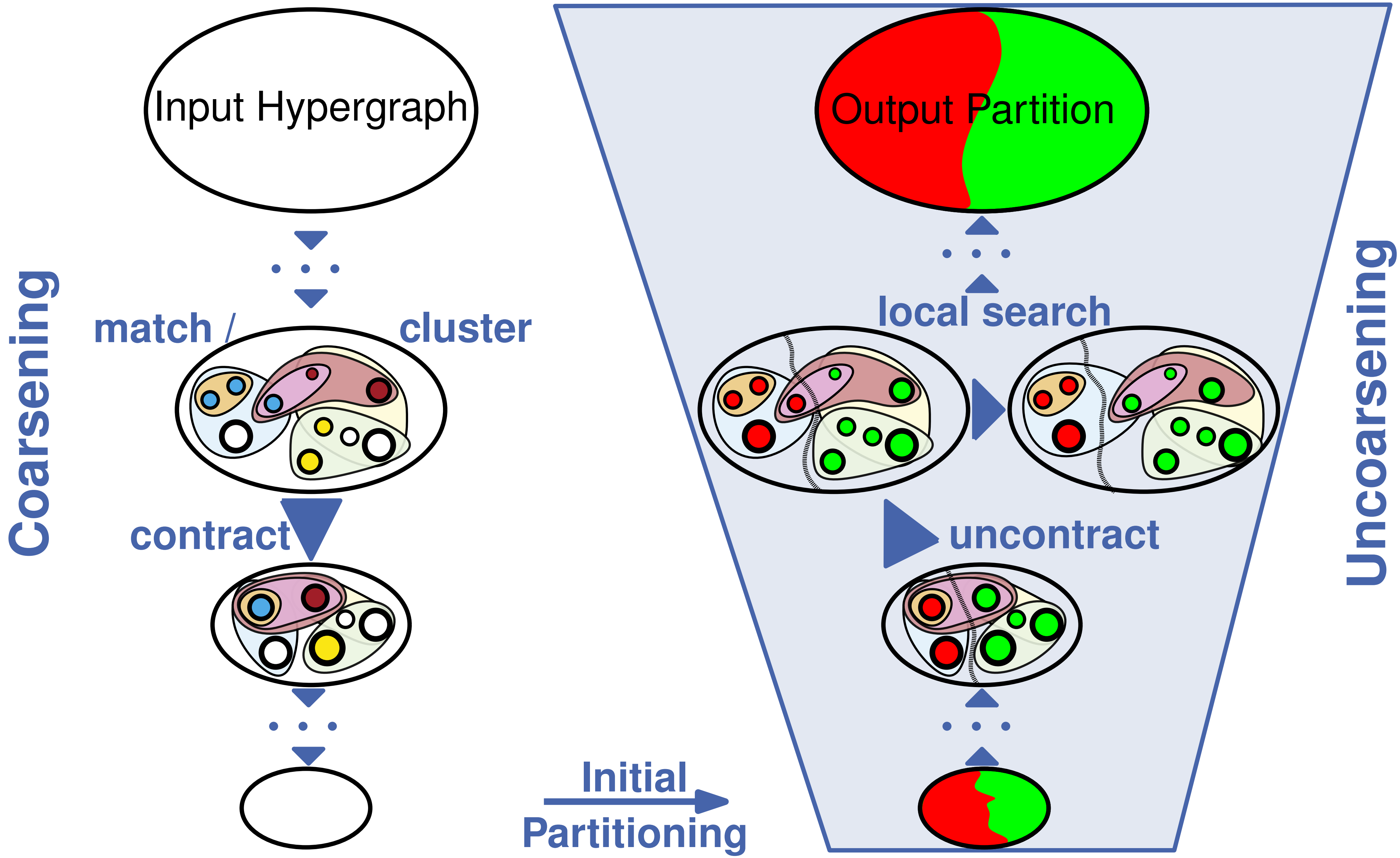
Successful Heuristic: Multilevel Paradigm



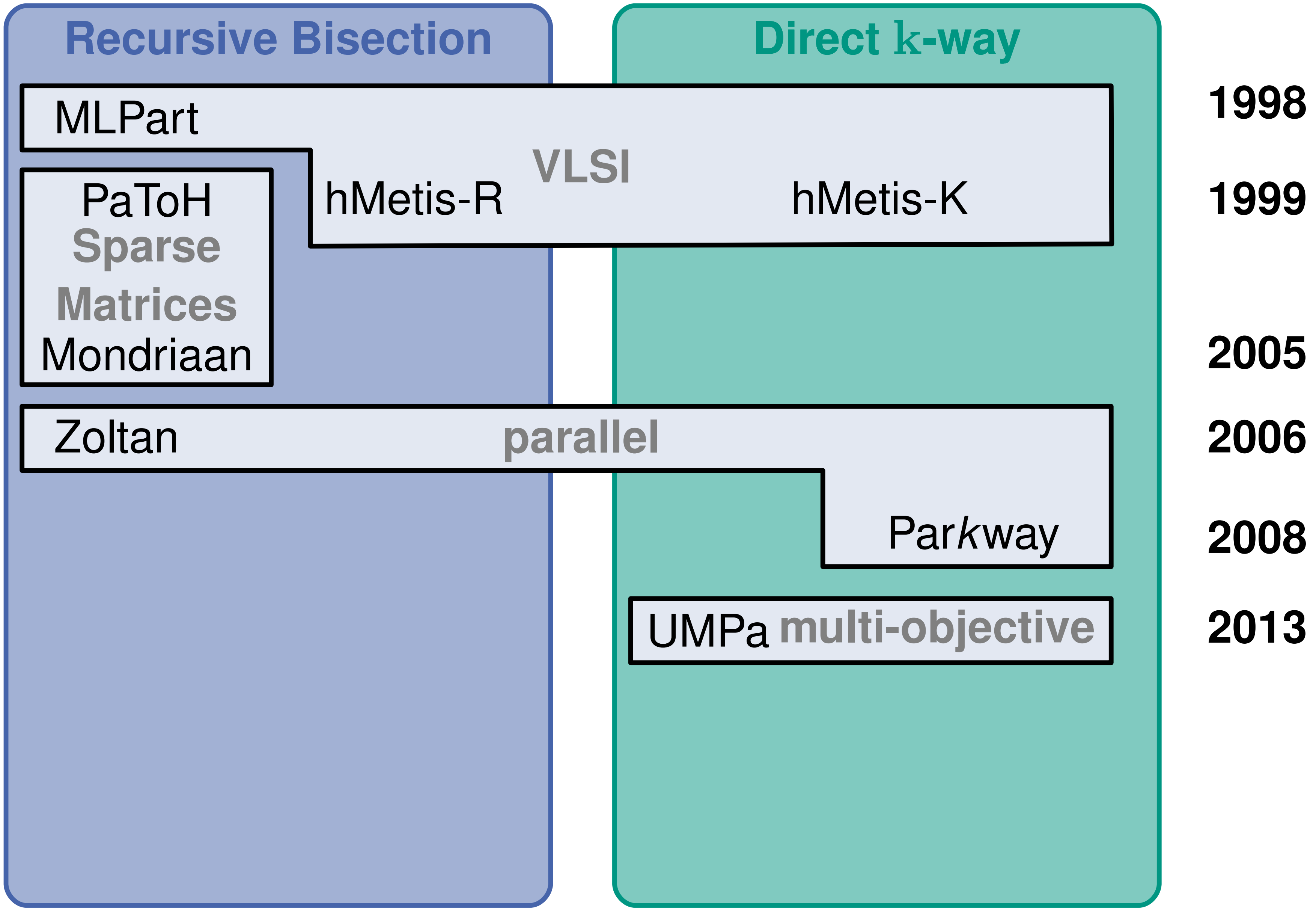
Successful Heuristic: Multilevel Paradigm



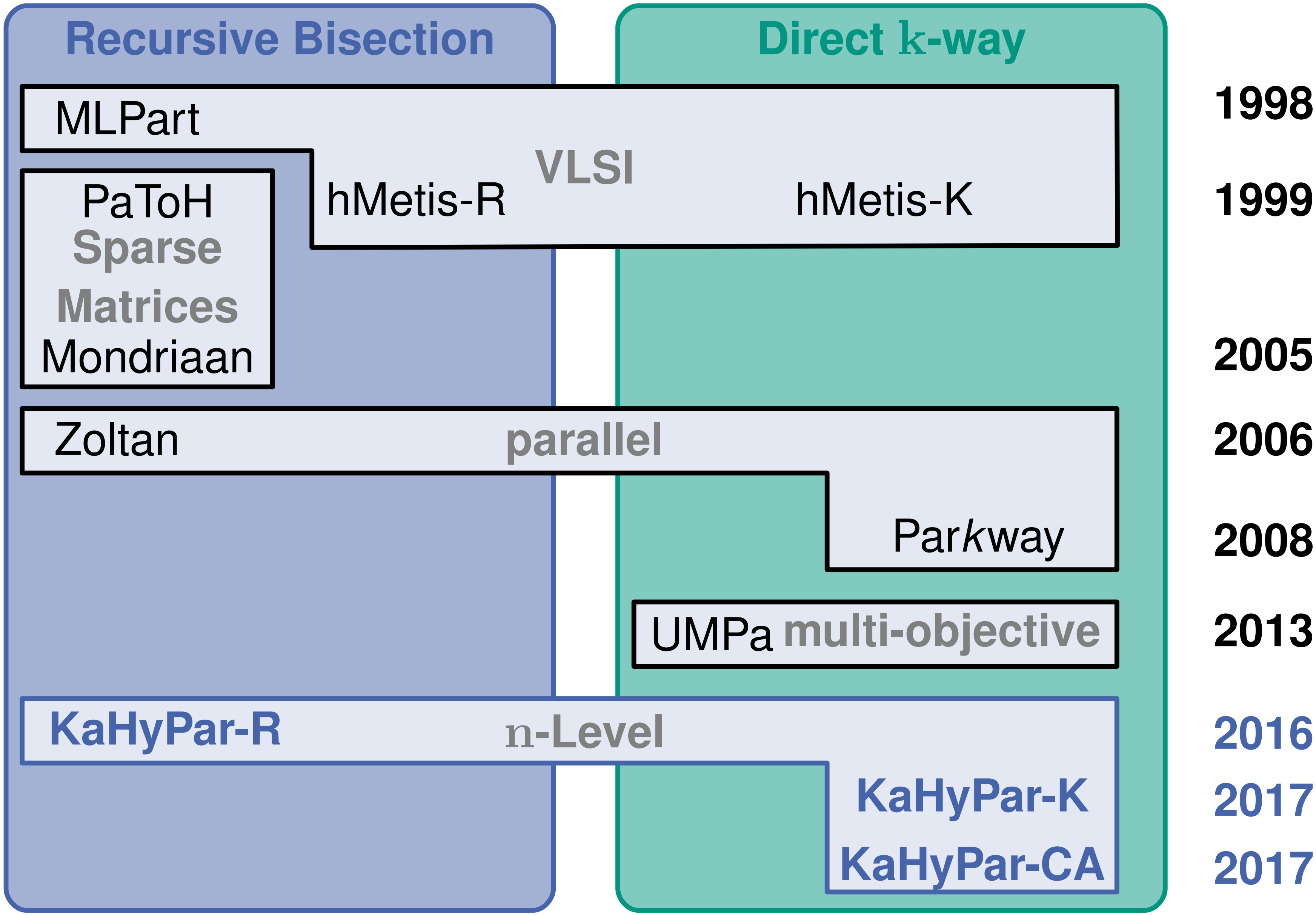
Successful Heuristic: Multilevel Paradigm



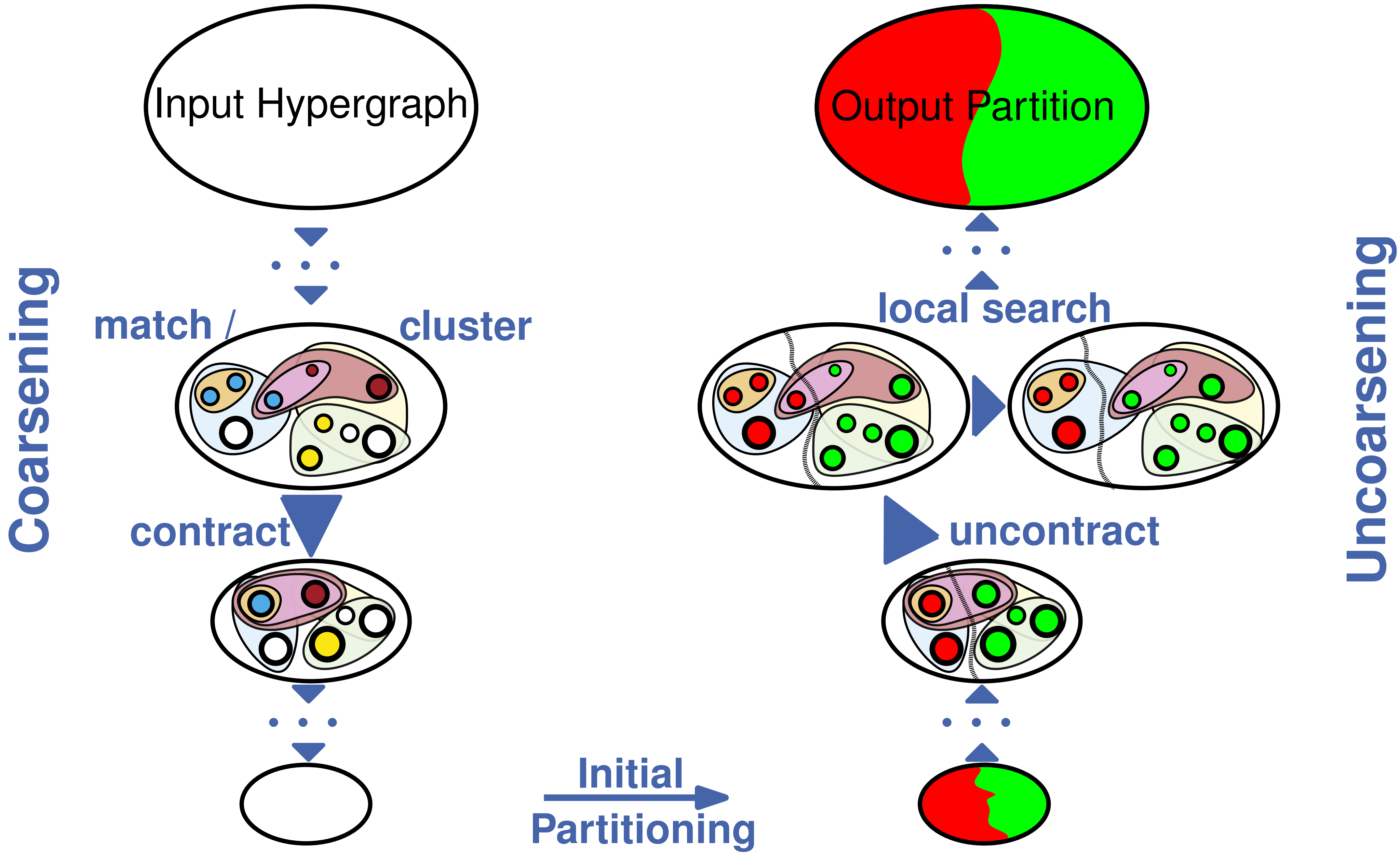
Taxonomy of Hypergraph Partitioning Tools



Taxonomy of Hypergraph Partitioning Tools

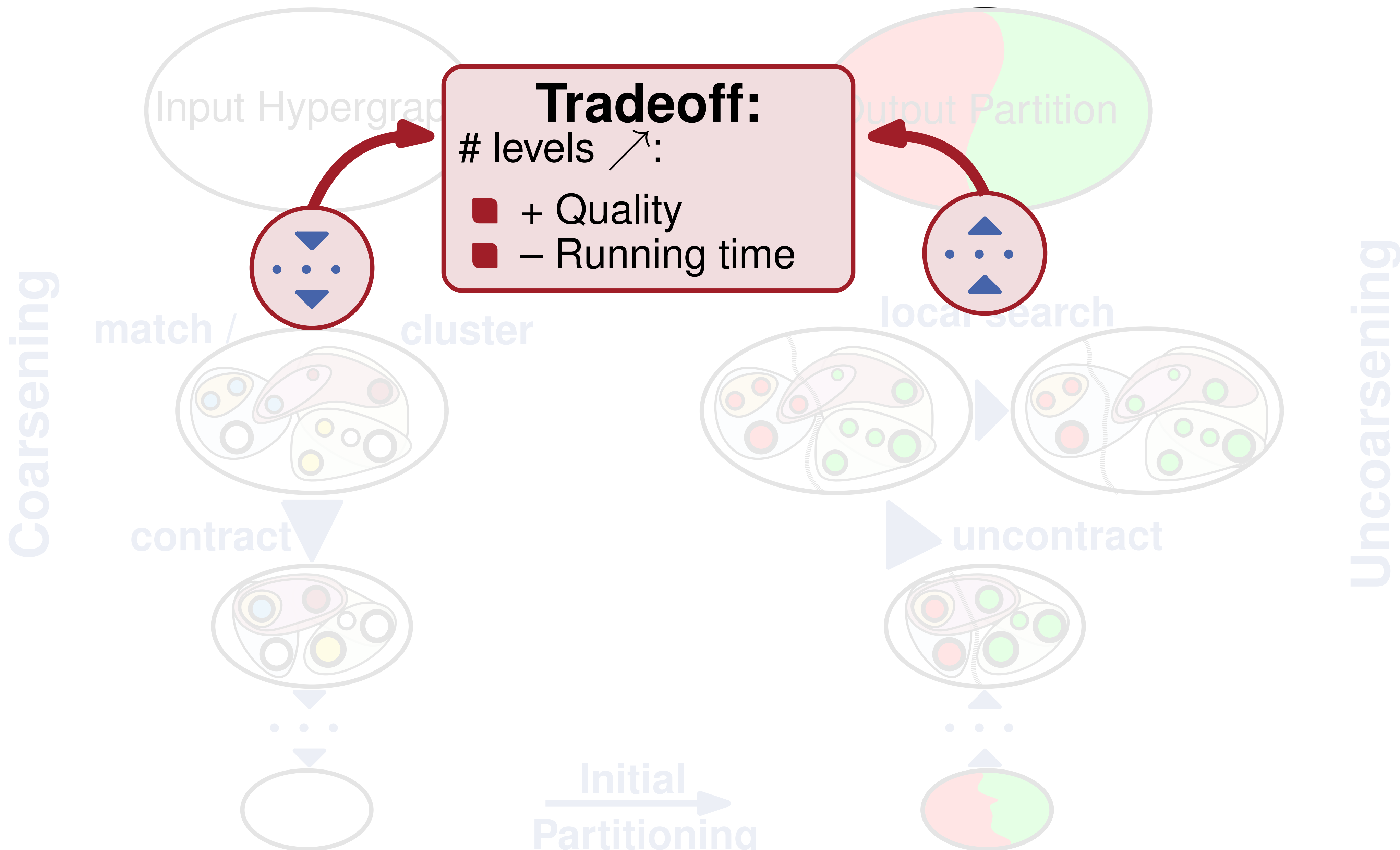


Why Yet Another Multilevel Algorithm?

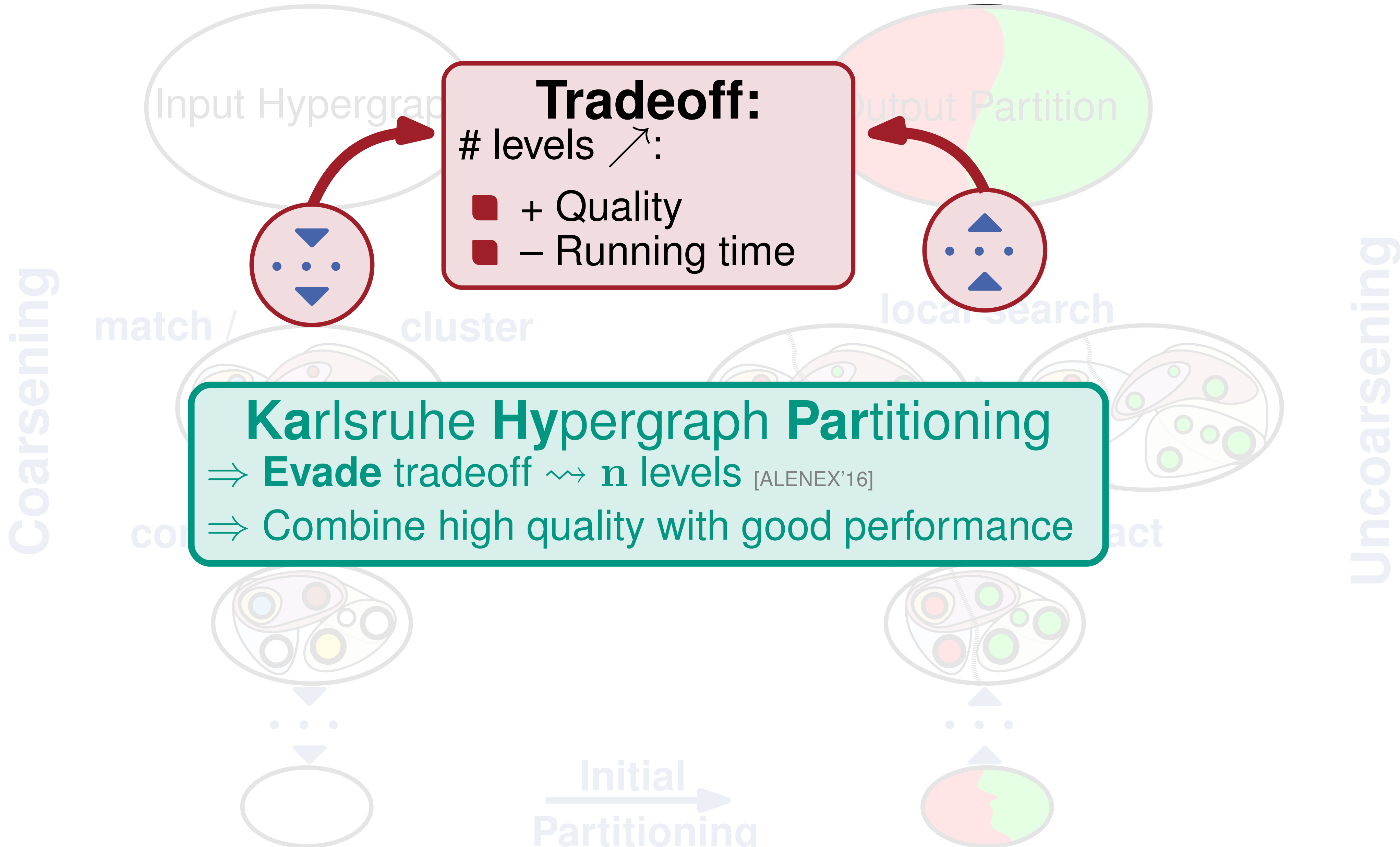


Institute of Theoretical Informatics
Algorithmics Group

Why Yet Another Multilevel Algorithm?



Why Yet Another Multilevel Algorithm?



Tradeoff:

levels ↗:

- + Quality
- – Running time

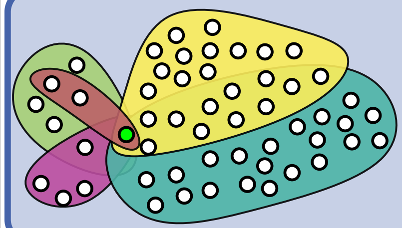
Karlsruhe **Hy**pergraph **Par**titioning

⇒ **Evade** tradeoff ⇝ n levels [ALENEX'16]

⇒ Combine high quality with good performance

KaHyPar: Novel Algorithmic Ingredients

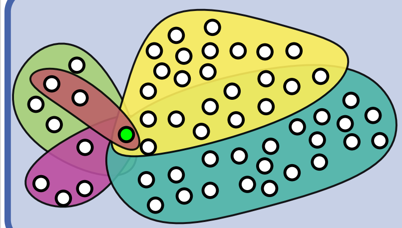
KaHyPar: Novel Algorithmic Ingredients



Min-Hash Based Sparsification

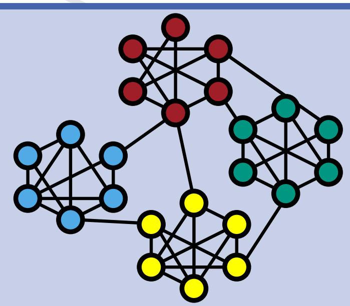
[ALENEX'17]

KaHyPar: Novel Algorithmic Ingredients



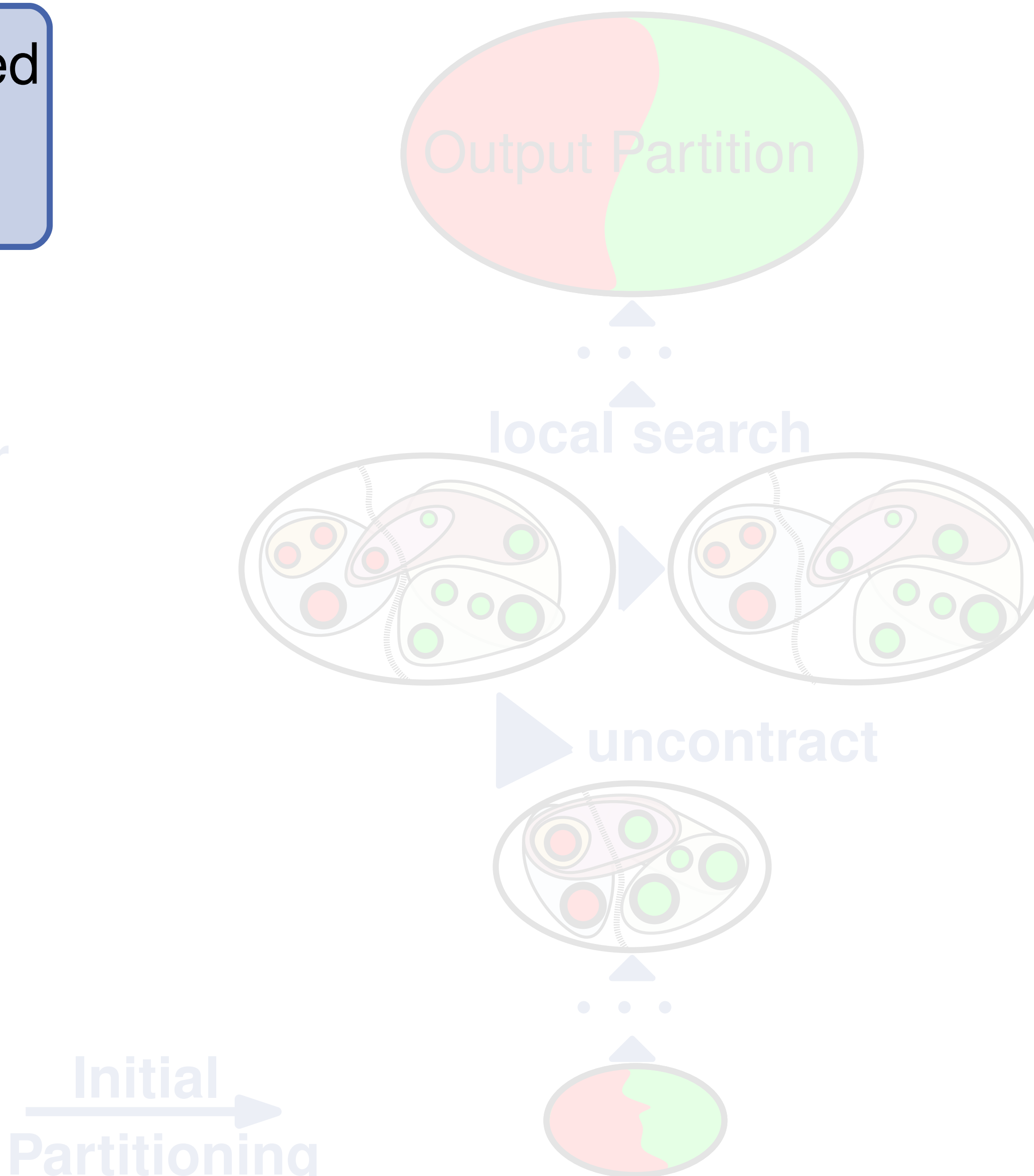
Min-Hash Based Sparsification

[ALENEX'17]

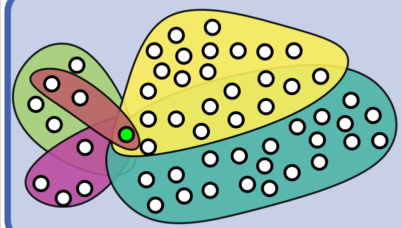


Community-Aware Coarsening

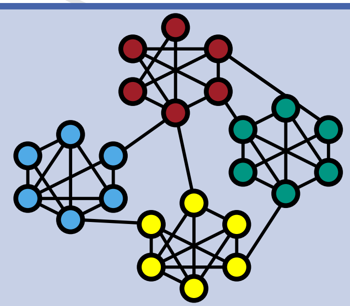
[SEA'17]



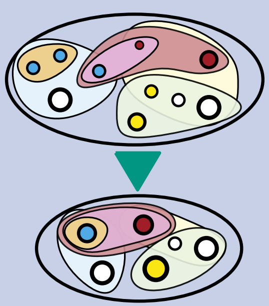
KaHyPar: Novel Algorithmic Ingredients



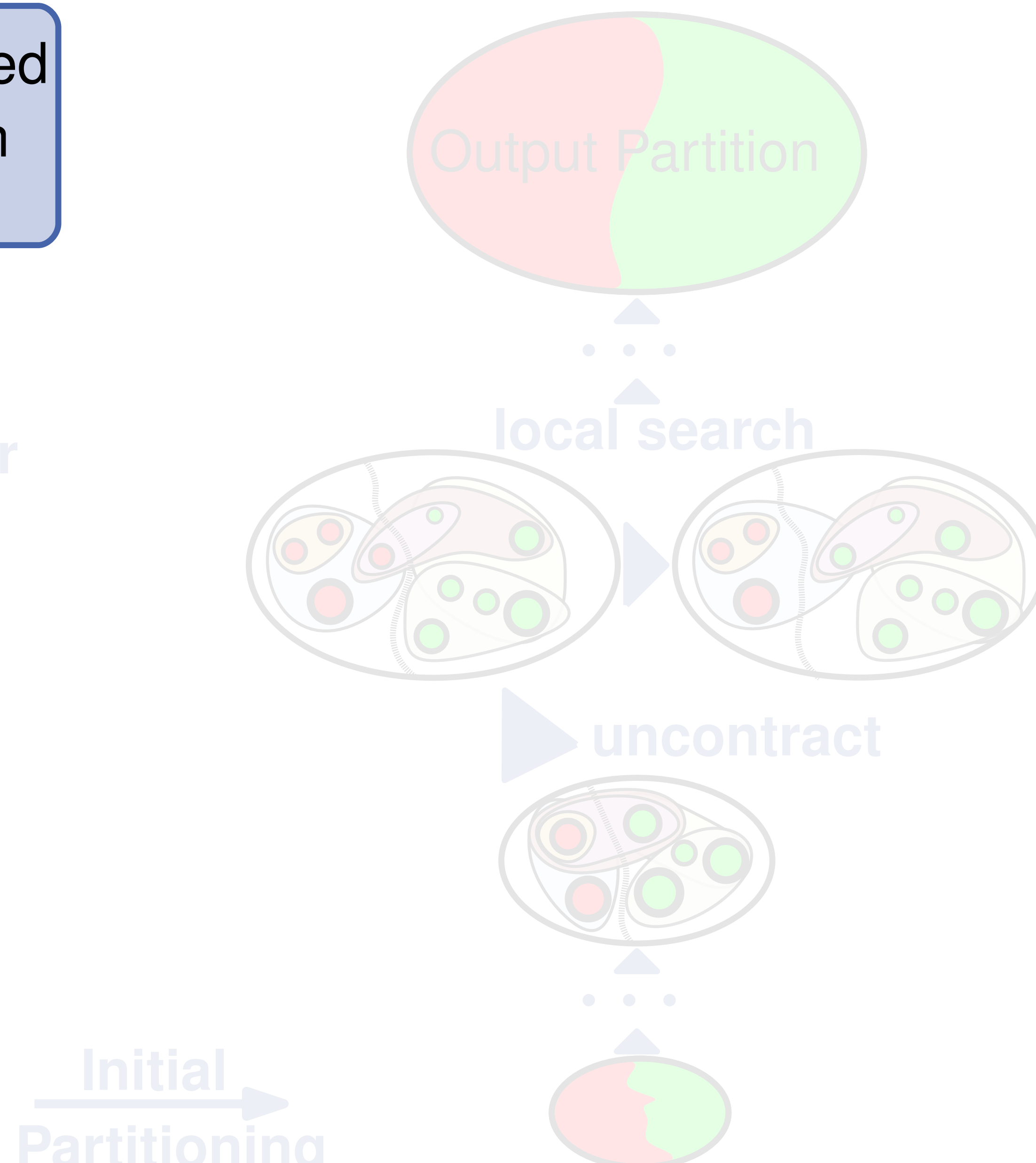
Min-Hash Based Sparsification
[ALENEX'17]



Community-Aware Coarsening
[SEA'17]

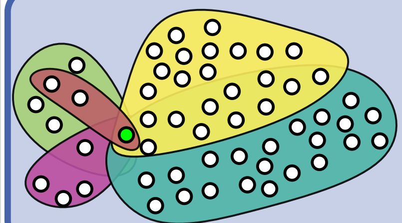


Fast *n*-Level Coarsening
[ALENEX'17]



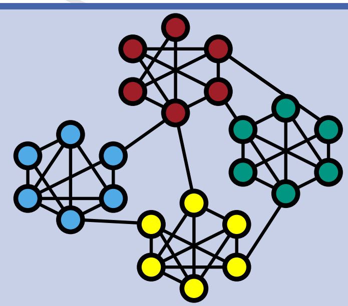
KaHyPar: Novel Algorithmic Ingredients

Karlsruhe Institute of Technology



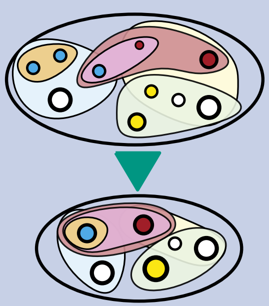
Min-Hash Based Sparsification

[ALENEX'17]



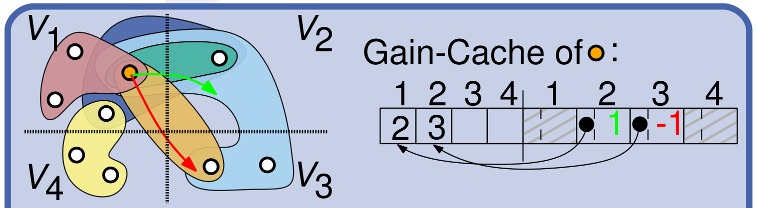
Community-Aware Coarsening

[SEA'17]



Fast n-Level Coarsening

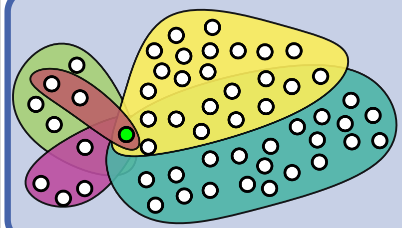
[ALENEX'17]



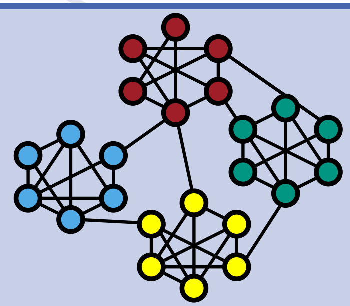
Engineered k-way FM

[ALENEX'17]

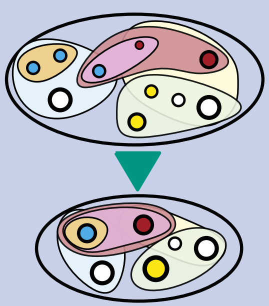
KaHyPar: Novel Algorithmic Ingredients



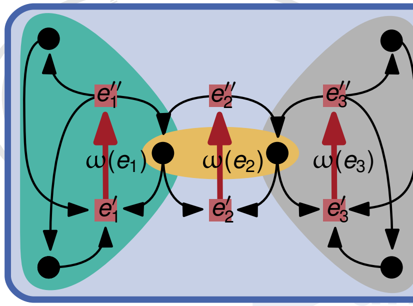
Min-Hash Based Sparsification
[ALENEX'17]



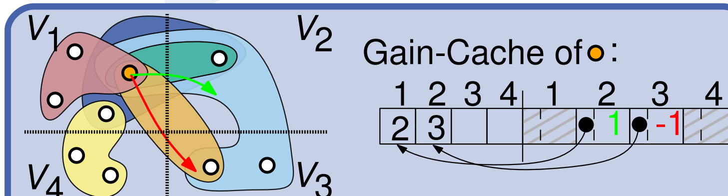
Community-Aware Coarsening
[SEA'17]



Fast *n*-Level Coarsening
[ALENEX'17]

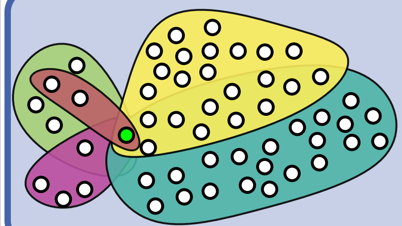


Max-Flow Min-Cut Refinement
[Heuer, Master's Thesis (upcoming)]



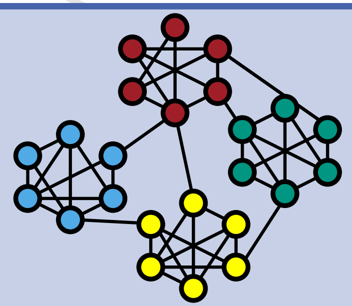
Engineered *k*-way FM
[ALENEX'17]

KaHyPar: Novel Algorithmic Ingredients



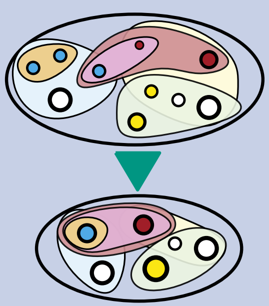
Min-Hash Based Sparsification

[ALENEX'17]



Community-Aware Coarsening

[SEA'17]



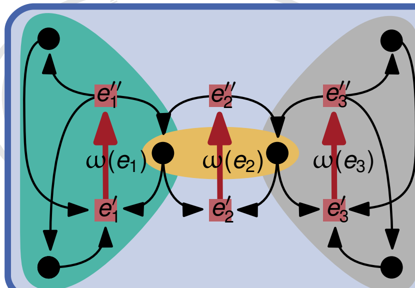
Fast n-Level Coarsening

[ALENEX'17]

$$\text{Algorithm } A \leftarrow \begin{cases} \text{Config } \mathcal{C}_1 \\ \text{Config } \mathcal{C}_2 \end{cases}$$

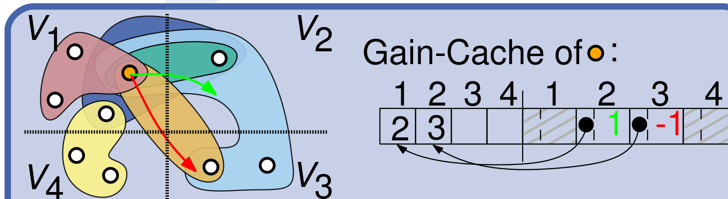
Algorithm Configuration

[Öhl, Bachelor's Thesis (upcoming)]



Max-Flow Min-Cut Refinement

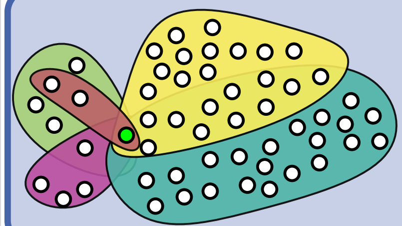
[Heuer, Master's Thesis (upcoming)]



Engineered k-way FM

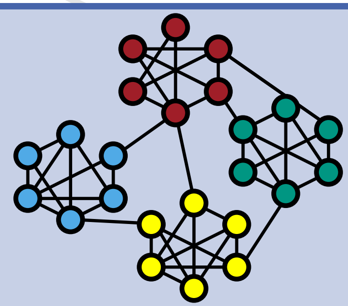
[ALENEX'17]

KaHyPar: Novel Algorithmic Ingredients



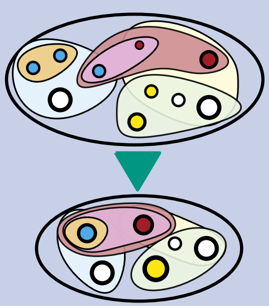
Min-Hash Based Sparsification

[ALENEX'17]



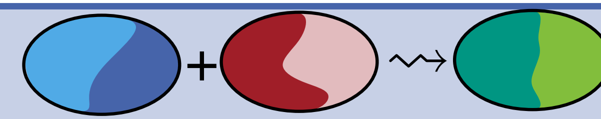
Community-Aware Coarsening

[SEA'17]



Fast n-Level Coarsening

[ALENEX'17]



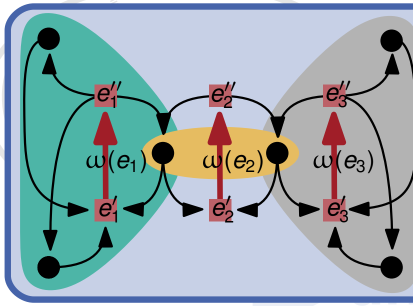
Memetic Multilevel Algorithm

[arXiv]

Algorithm $A \leftarrow \begin{cases} \text{Config } \mathcal{C}_1 \\ \text{Config } \mathcal{C}_2 \end{cases}$

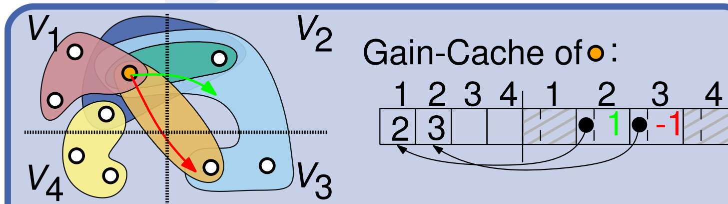
Algorithm Configuration

[Öhl, Bachelor's Thesis (upcoming)]



Max-Flow Min-Cut Refinement

[Heuer, Master's Thesis (upcoming)]

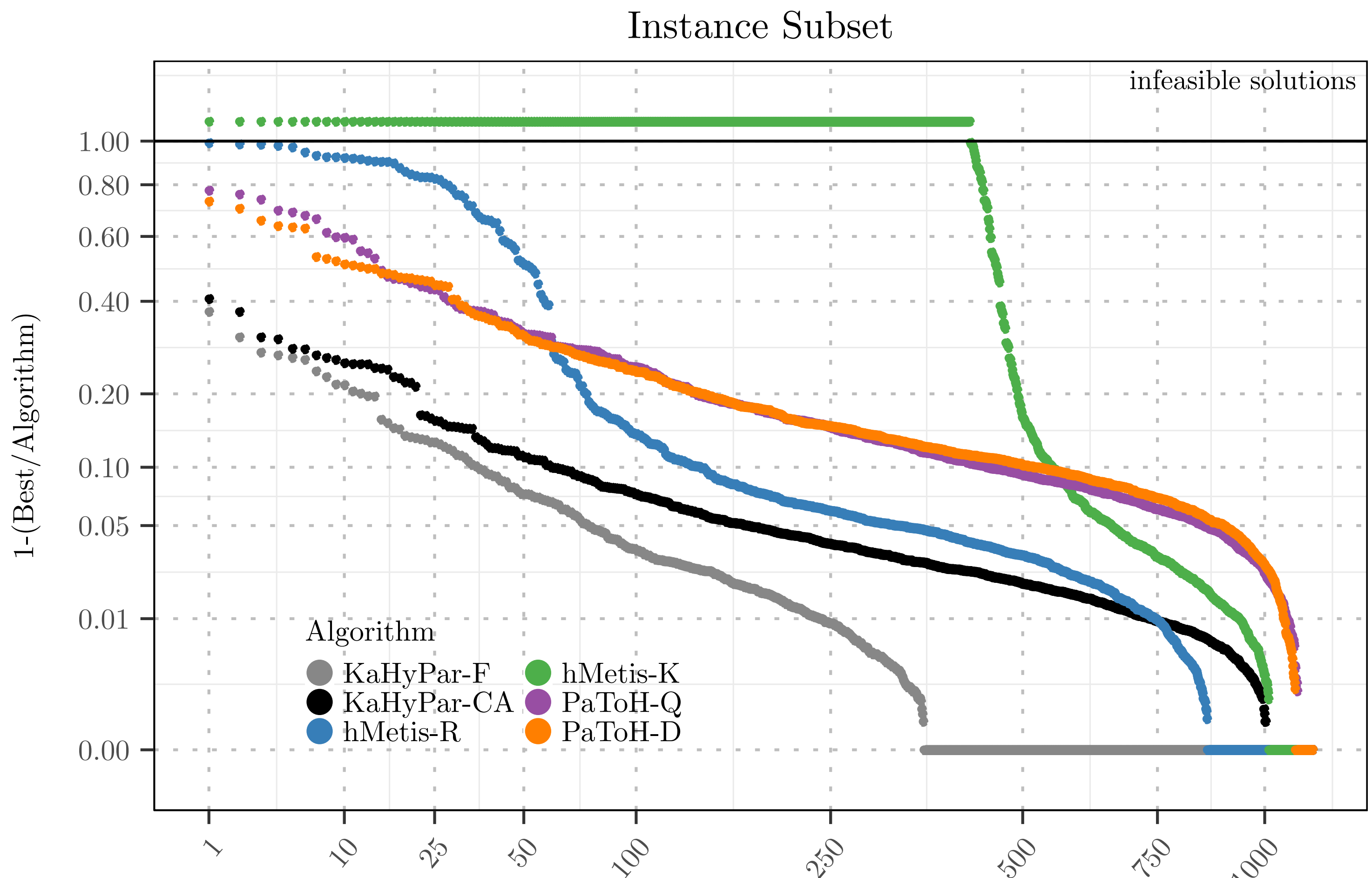


Engineered k-way FM

[ALENEX'17]

Institute of Theoretical Informatics
Algorithmics Group

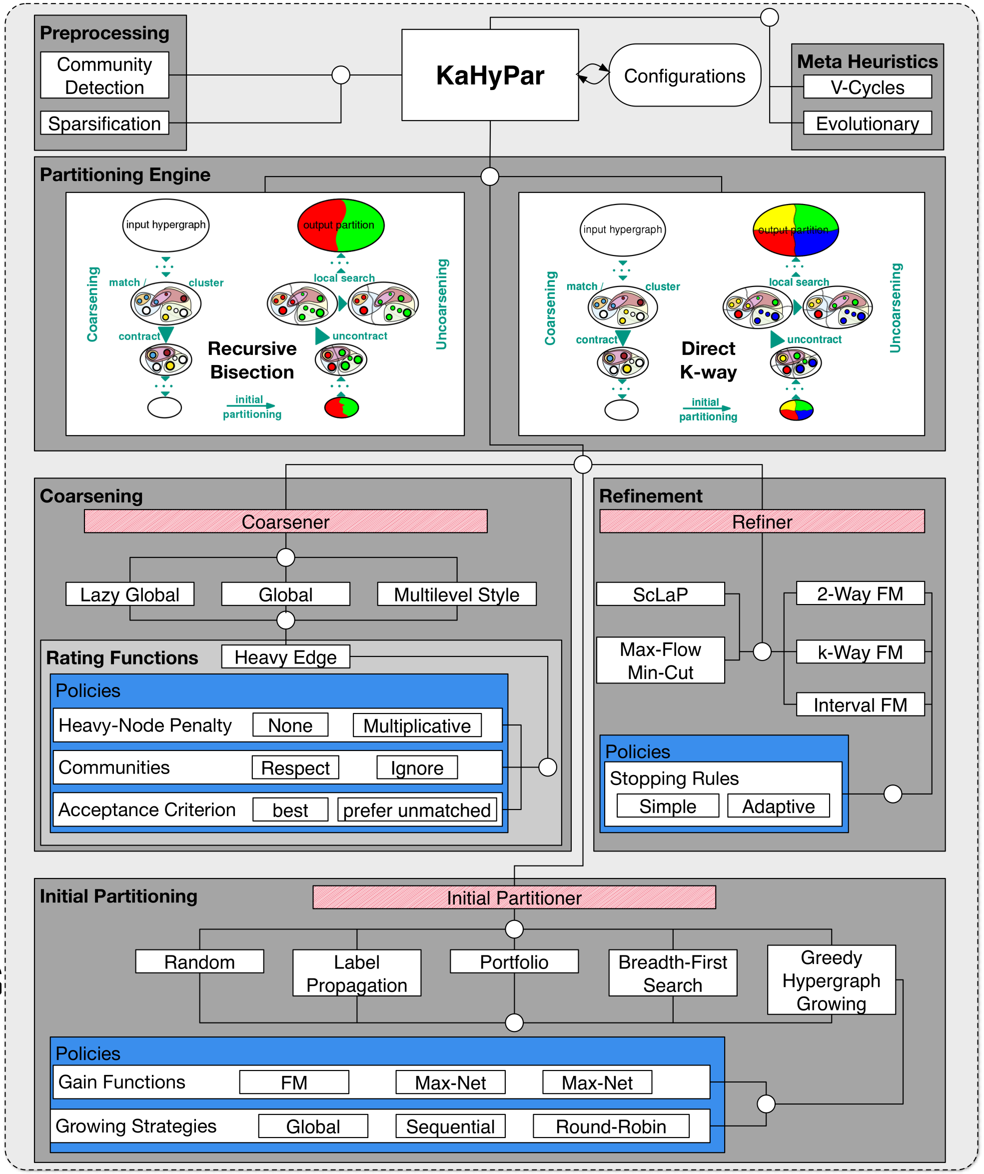
Latest Experimental Results



Algorithm	$t[s]$
KaHyPar-CA	21.0
KaHyPar-F	46.8
hMetis-R	63.1

KaHyPar

- *n*-Level Partitioning Framework
- Objectives:
  - 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`



References

Schlag et. al (ALENEX'16): S. Schlag, V. Henne, T. Heuer, H. Meyerhenke, P. Sanders, Christian Schulz. *k*-way Hypergraph Partitioning via *n*-Level Recursive Bisection. In 18th Workshop on Algorithm Engineering and Experiments, (ALENEX), pages 53–67, 2016.

Schlag et. al (ALENEX'17): Y. Akhremtsev, T. Heuer, P. Sanders, and S. Schlag. Engineering a direct k-way hypergraph partitioning algorithm. In 19th Workshop on Algorithm Engineering and Experiments, (ALENEX), pages 28–42, 2017.

Heuer, Schlag (SEA'17): T. Heuer and S. Schlag. Improving Coarsening Schemes for Hypergraph Partitioning by Exploiting Community Structure. In 16th International Symposium on Experimental Algorithms, (SEA), page 21:121:19, 2017.

Andre, Schlag, Schulz (arXiv): R. Andre, S. Schlag, and C. Schulz. Memetic Multilevel Hypergraph Partitioning. arXiv preprint arXiv:1710.01968 (2017).