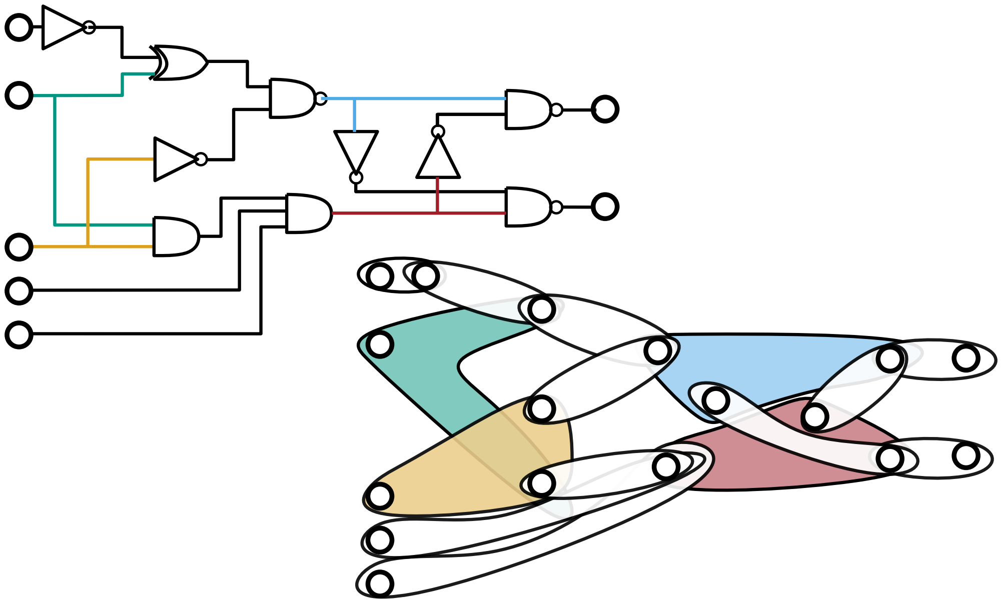
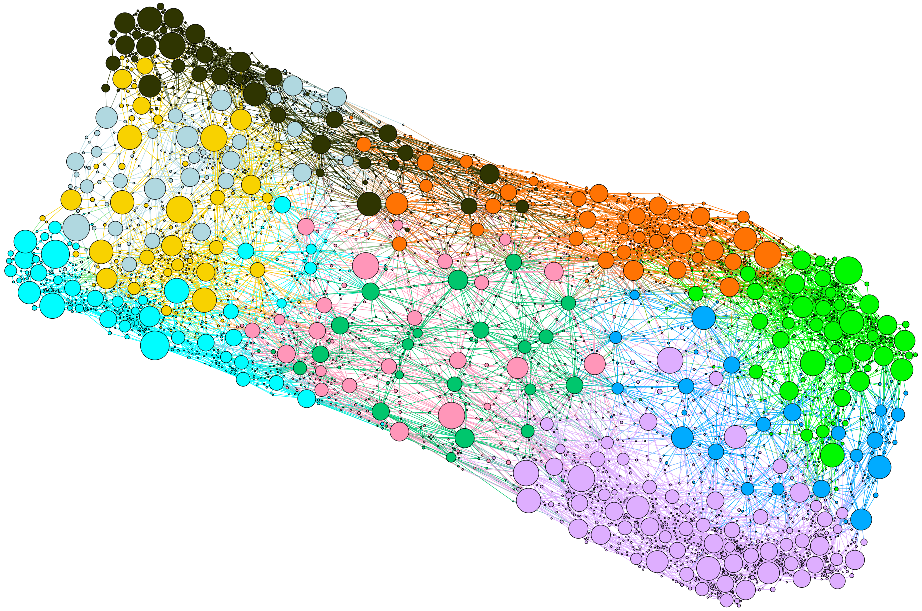
Improving Coarsening Schemes for Hypergraph Partitioning by Exploiting Community Structure

SEA’17 · June 23, 2017

Tobias Heuer and Sebastian Schlag

INSTITUTE OF THEORETICAL INFORMATICS · ALGORITHMICS GROUP



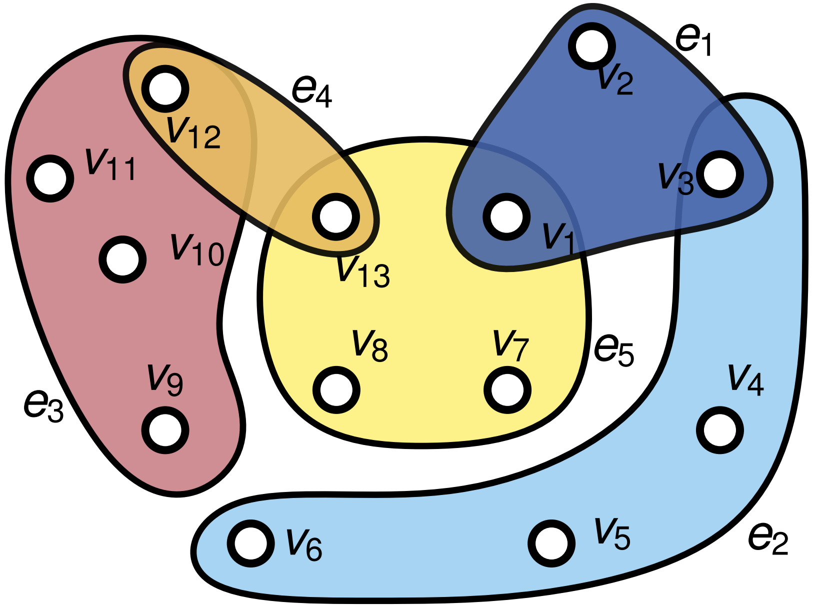


Hypergraphs

- Generalization of graphs
  $\Rightarrow$ hyperedges connect ≥ 2 nodes

- Graphs $\Rightarrow$ dyadic (**2-ary**) relationships
- Hypergraphs $\Rightarrow$ (**d-ary**) relationships

- Hypergraph $H = (V, E, c, \omega)$
  - Vertex set $V = \{1, ..., n\}$
  - Edge set $E \subseteq \mathcal{P}(V) \setminus \emptyset$
  - Node weights $c : V \rightarrow \mathbb{R}_{\geq 1}$
  - Edge weights $\omega : E \rightarrow \mathbb{R}_{\geq 1}$



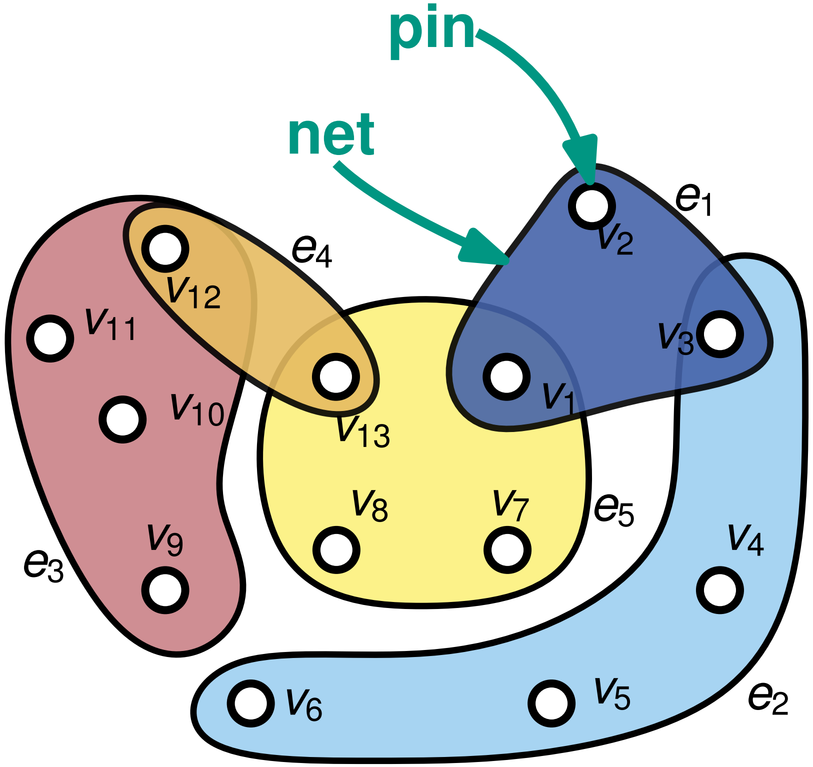
Hypergraphs

- Generalization of graphs
  $\Rightarrow$ hyperedges connect ≥ 2 nodes

- Graphs $\Rightarrow$ dyadic ($\mathbf{2}$**-ary**) relationships
- Hypergraphs $\Rightarrow$ ($\mathbf{d}$**-ary**) relationships

- Hypergraph $H = (V, E, c, \omega)$
  - Vertex set $V = \{1, ..., n\}$
  - Edge set $E \subseteq \mathcal{P}(V) \setminus \emptyset$
  - Node weights $c : V \rightarrow \mathbb{R}_{\geq 1}$
  - Edge weights $\omega : E \rightarrow \mathbb{R}_{\geq 1}$

- $|P| = \sum_{e \in E} |e| = \sum_{v \in V} d(v)$

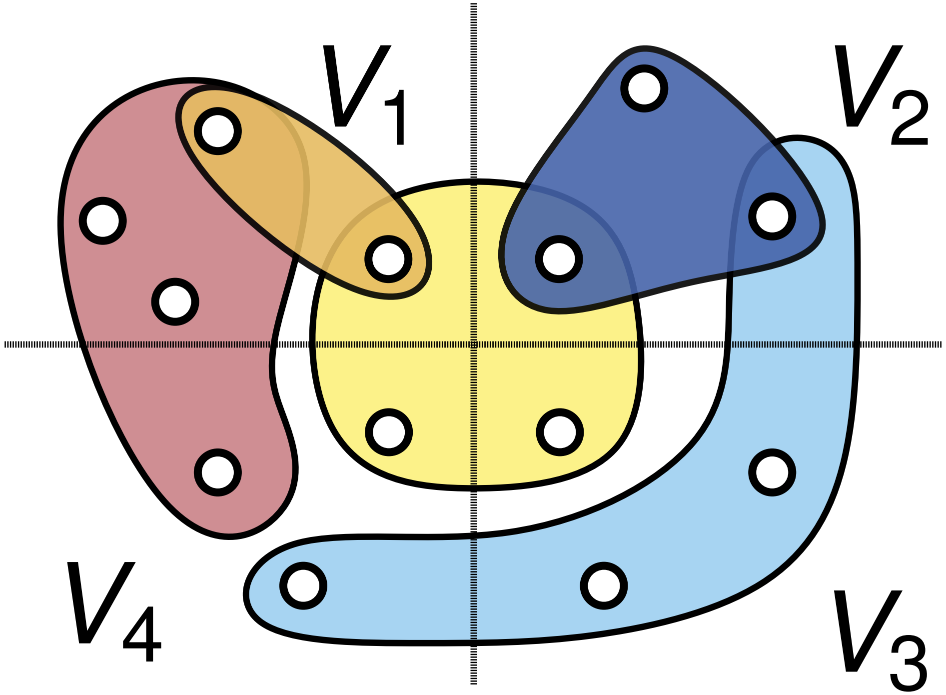


Hypergraph Partitioning Problem

Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

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

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



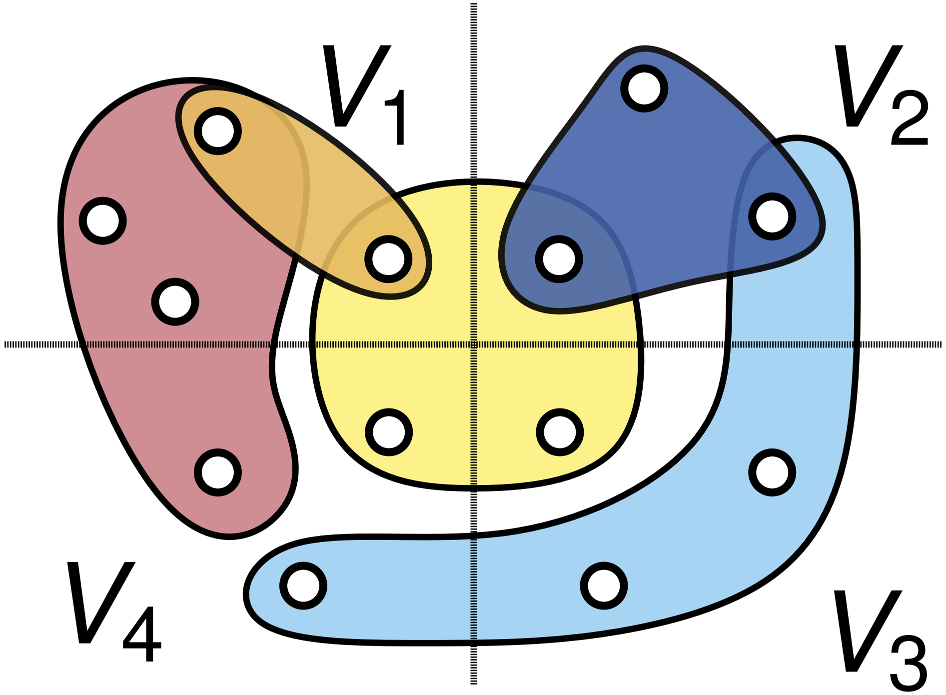
Hypergraph Partitioning Problem

Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

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

imbalance parameter

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



Hypergraph Partitioning Problem

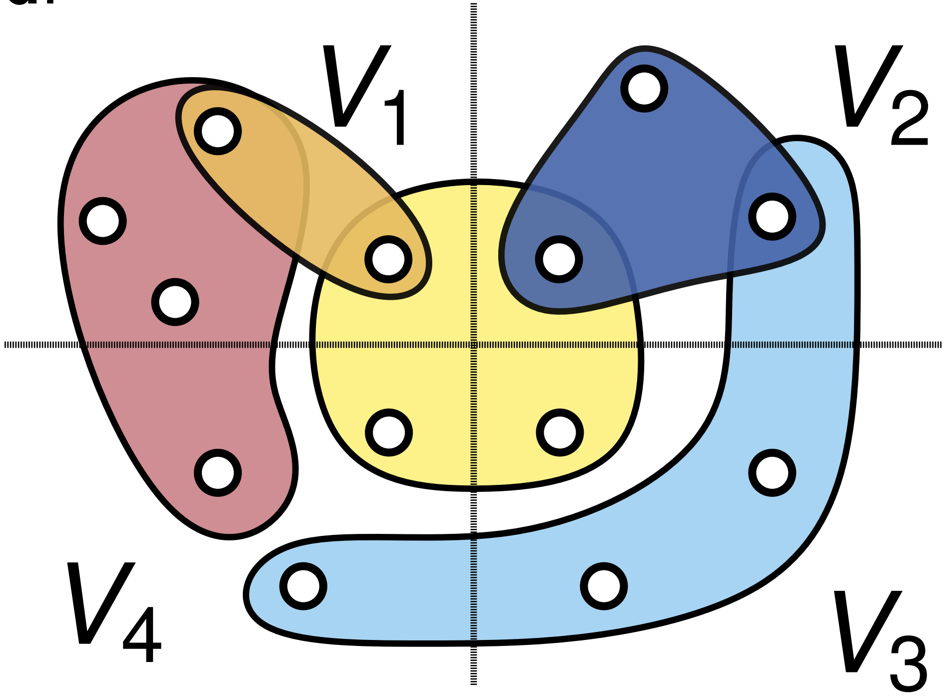
Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

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

imbalance parameter

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

- **connectivity** objective is **minimized**:



Hypergraph Partitioning Problem

Partition hypergraph $H = (V, E, c, \omega)$ into $\mathbf{k}$ disjoint blocks $\Pi = \{V_1, \dots, V_k\}$ such that:

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

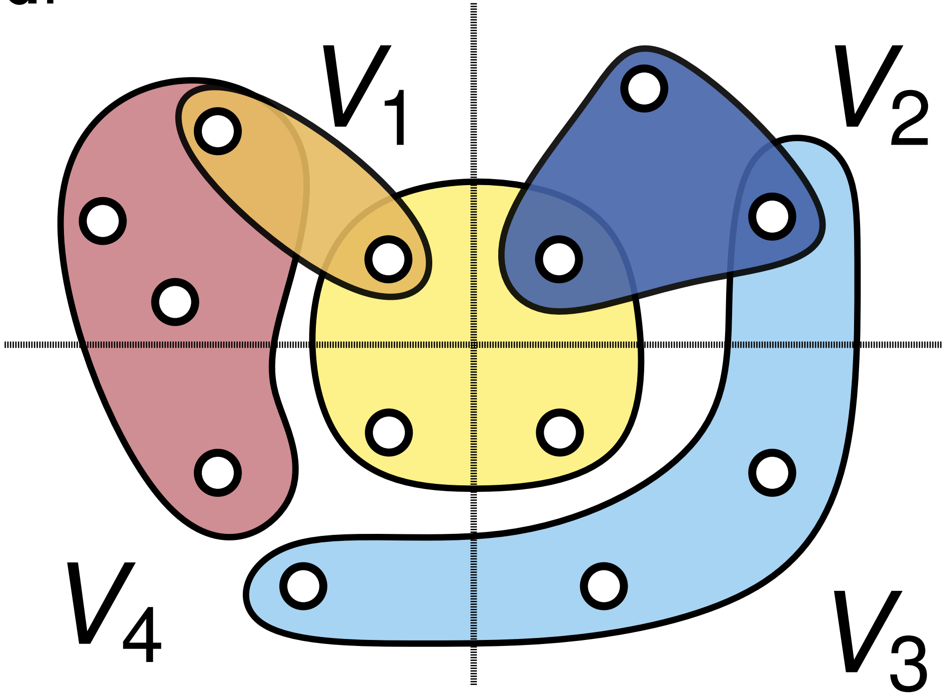
imbalance parameter

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

- **connectivity** objective is **minimized**:

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

connectivity:
 # blocks connected by net *e*



Hypergraph Partitioning Problem

Partition hypergraph $H = (V, E, c, \omega)$ into k disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that:

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

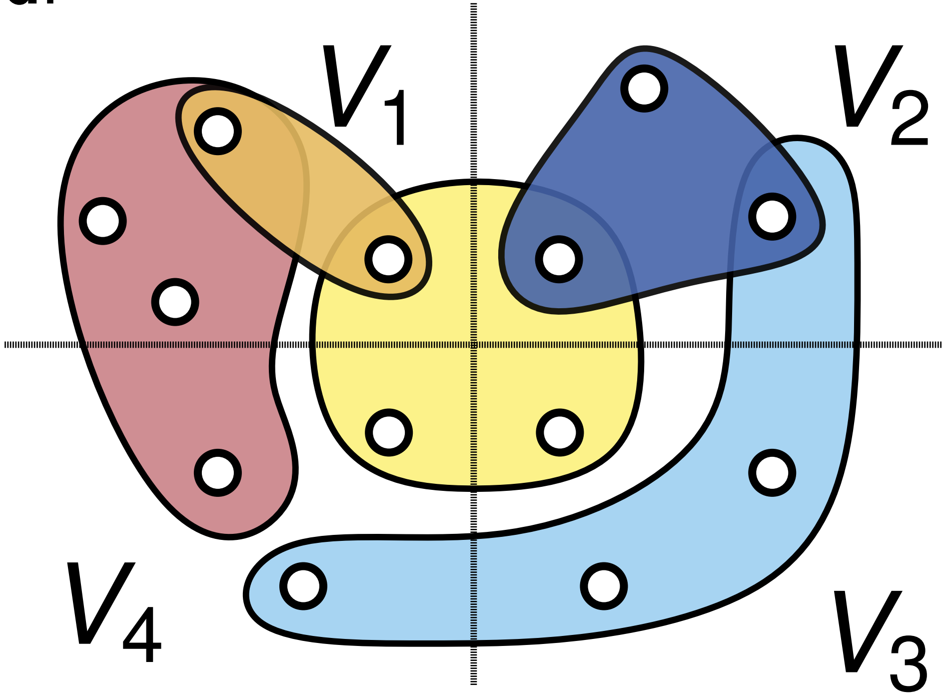
$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

imbalance
parameter

- **connectivity** objective is **minimized**:

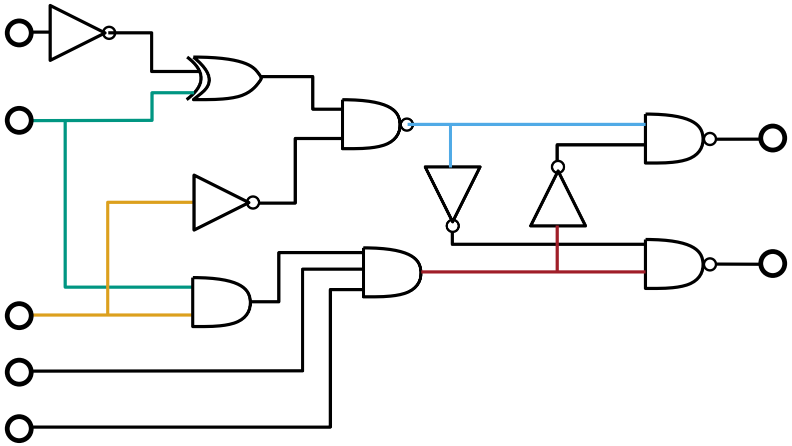
$$\sum_{e \in \text{cut}} (\lambda - 1)\, \omega(e) = 6$$

connectivity:
 # blocks connected by net *e*



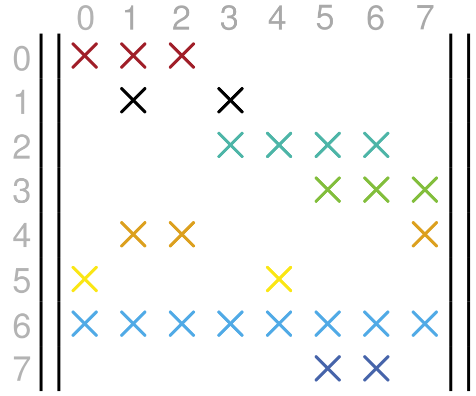
Applications

VLSI Design

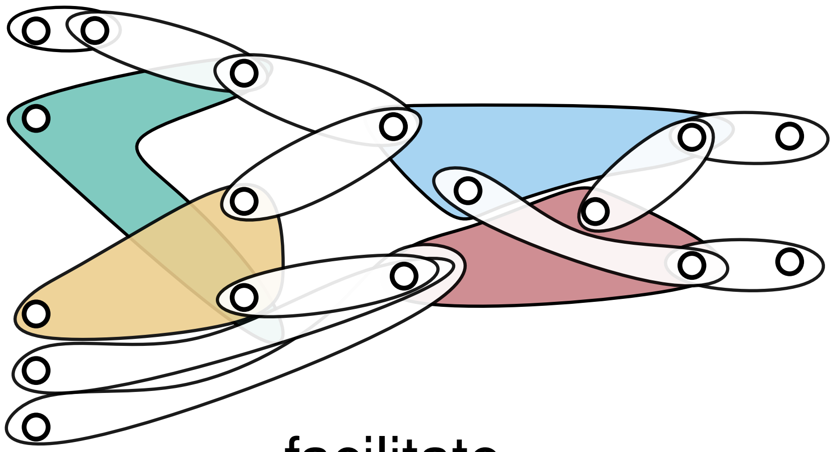


Application Domain

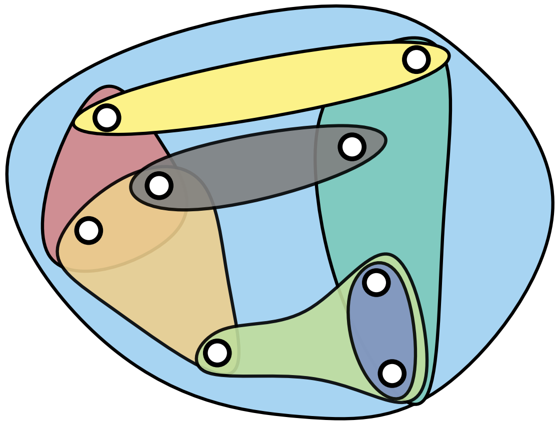
Scientific Computing



$$\begin{array}{c|cccccccc|}
 & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\
\hline
0 & \times & \times & \times & & & & & \\
1 & & \times & & \times & & & & \\
2 & & & & \times & \times & \times & \times & \\
3 & & & & & & \times & \times & \times \\
4 & & \times & \times & & & & & \times \\
5 & \times & & & & \times & & & \\
6 & \times & \times & \times & \times & \times & \times & \times & \times \\
7 & & & & & & \times & \times & \\
\end{array}$$



Hypergraph Model

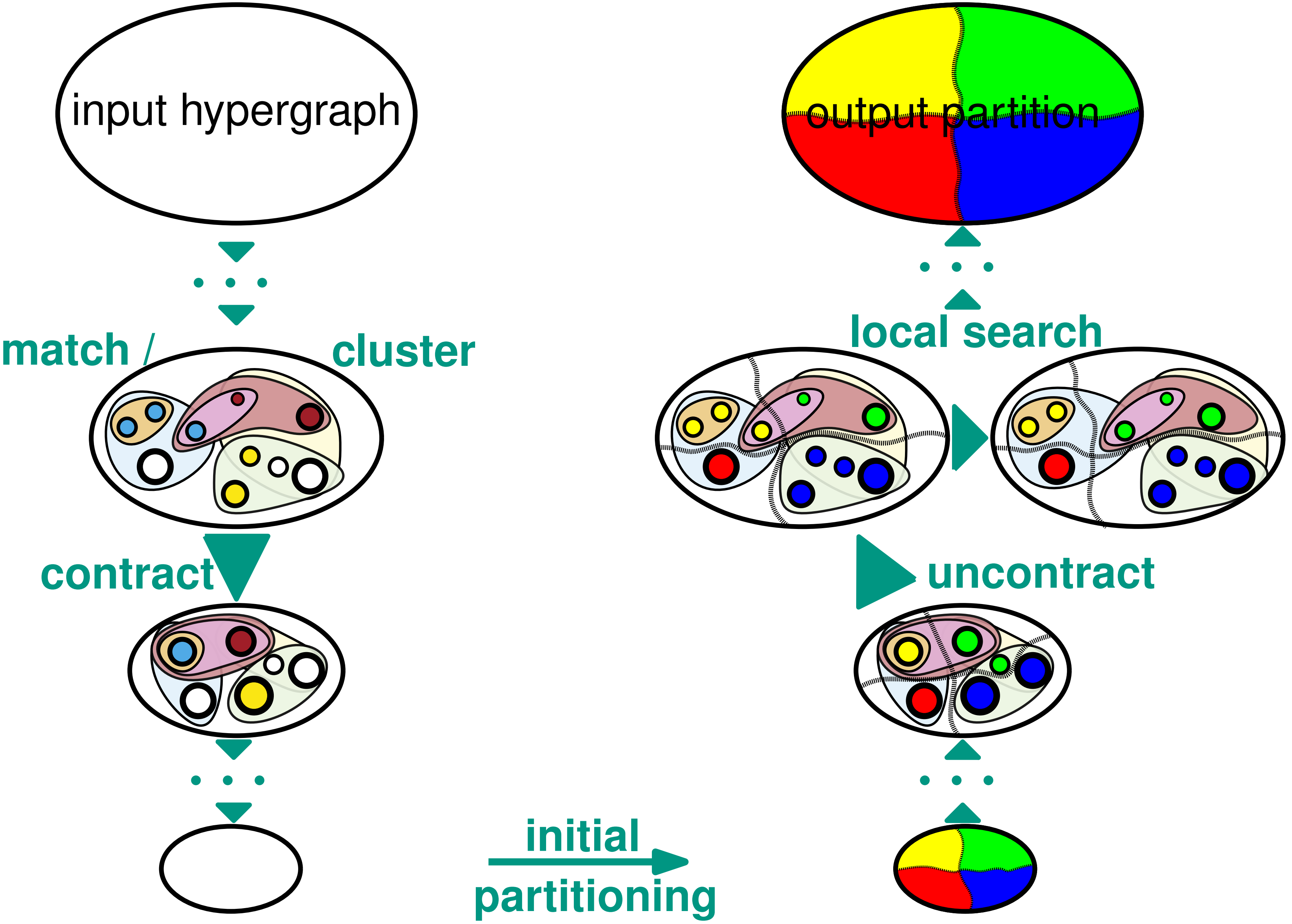


facilitate floorplanning & placement

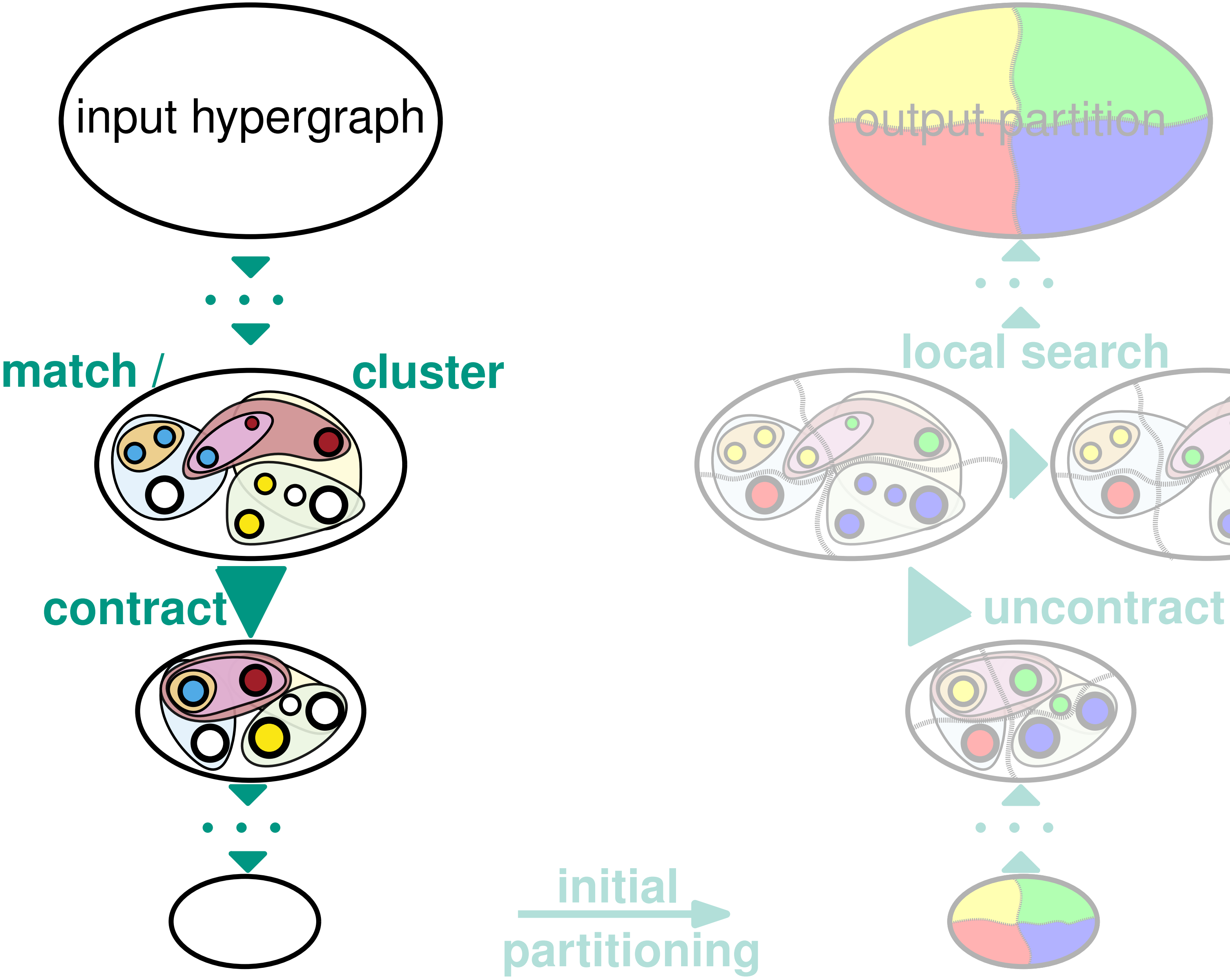
Goal

minimize communication

The Multilevel Framework



This Talk: Coarsening Phase



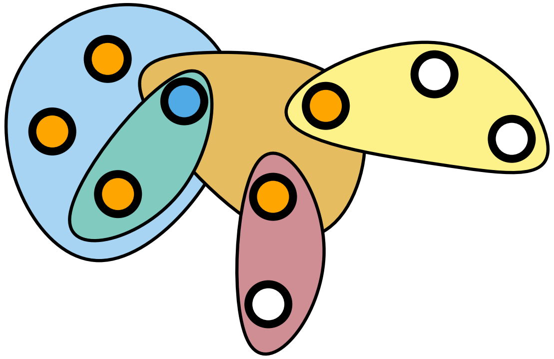
Clustering-based Coarsening

Common Strategy: **avoid** global decisions ⇝ **local**, greedy algorithms

Objective: identify highly connected vertices

using...

```
foreach vertex $v$ do
    cluster[$v$] := argmax rating($v$,$u$)
                   neighbor $u$
```



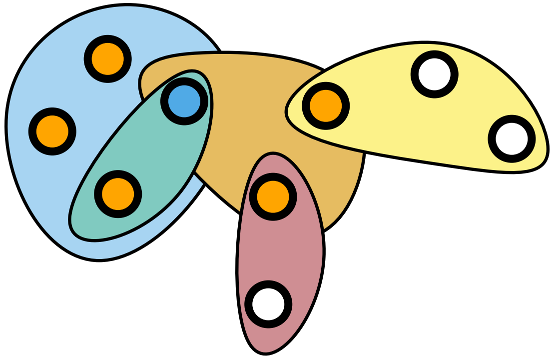
Clustering-based Coarsening

Common Strategy: **avoid** global decisions ⇝ **local**, greedy algorithms

Objective: identify highly connected vertices

using...

```
foreach vertex $v$ do
    cluster[$v$] := $\underset{\text{neighbor } u}{\operatorname{argmax}}$ rating($v$,$u$)
```



Main Design Goals:[Karypis,Kumar 99]

1: reduce **size** of nets ⇝ easier local search

2: reduce **number** of nets ⇝ easier initial partitioning

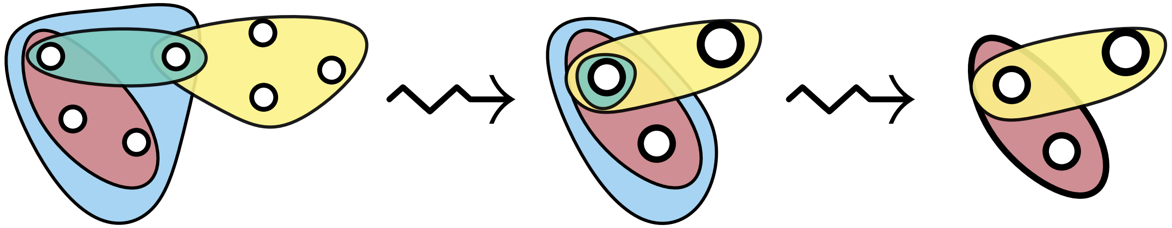
3: maintain **structural similarity** ⇝ good coarse solutions

Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets ⇝ easier local search

2: reduce **number** of nets ⇝ easier initial partitioning

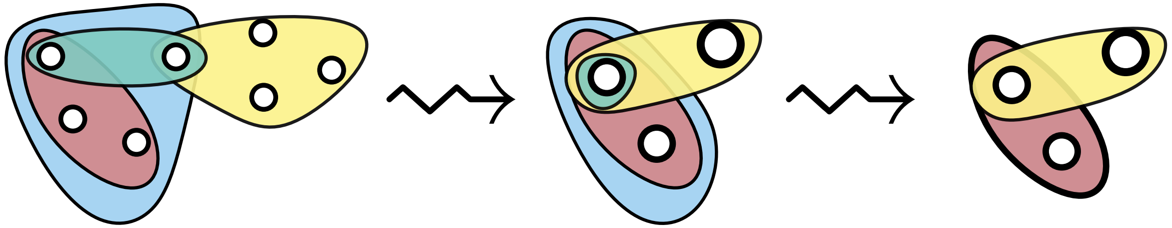


Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets ⇝ easier local search

2: reduce **number** of nets ⇝ easier initial partitioning



⟹ hypergraph-tailored rating functions:

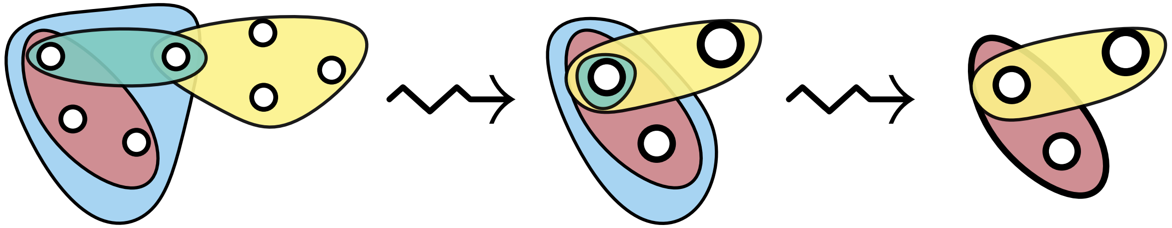
$$r(u,v) := \sum_{\substack{\text{net } e \\ \text{containing } u,v}} \frac{\omega(e)}{|e|-1}$$

Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets ⇝ easier local search

2: reduce **number** of nets ⇝ easier initial partitioning



⟹ hypergraph-tailored rating functions:

$$r(u,v) := \sum_{\substack{\text{net } e \\ \text{containing } u,v}} \frac{\omega(e)}{|e|-1}$$

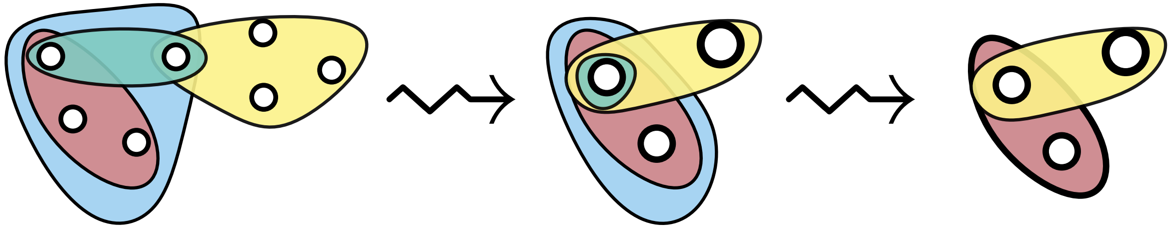
large number ... → containing u,v

Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets ⇝ easier local search

2: reduce **number** of nets ⇝ easier initial partitioning



⟹ hypergraph-tailored rating functions:

$$r(u, v) := \sum_{\substack{\text{net } e \\ \text{containing } u,v}} \frac{\omega(e)}{|e|-1}$$

of heavy nets ...

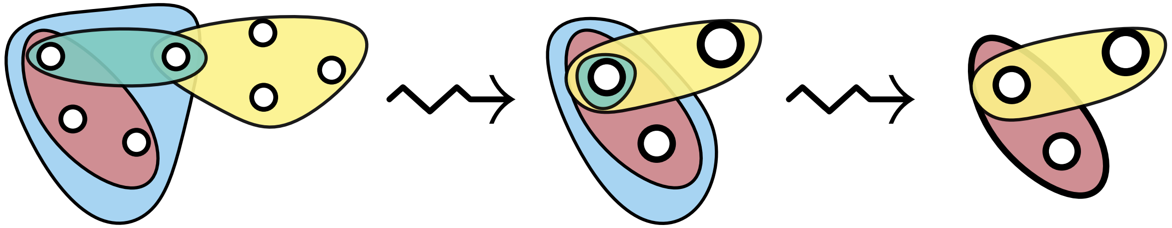
large number ...

Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets ⇝ easier local search

2: reduce **number** of nets ⇝ easier initial partitioning



⟹ hypergraph-tailored rating functions:

$$r(u,v) := \sum_{\substack{\text{net } e \\ \text{containing } u,v}} \frac{\omega(e)}{|e|-1}$$

of heavy nets ...

... with small size

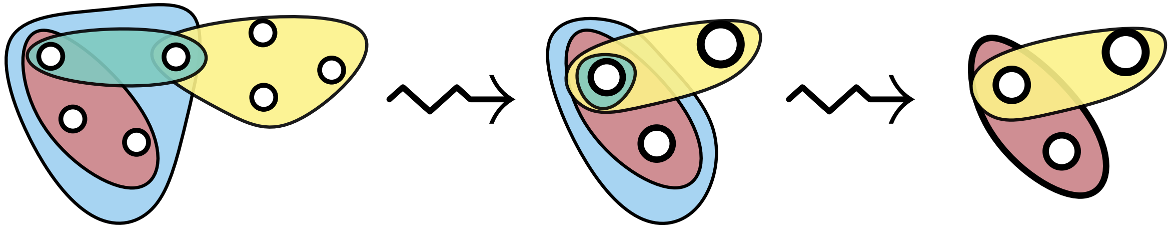
large number ...

Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets ⇝ easier local search ✓

2: reduce **number** of nets ⇝ easier initial partitioning ✓



⟹ hypergraph-tailored rating functions:

$$r(u,v) := \sum_{\substack{\text{net } e \\ \text{containing } u,v}} \frac{\omega(e)}{|e|-1}$$

large number ... → net e containing u,v

← of heavy nets ...

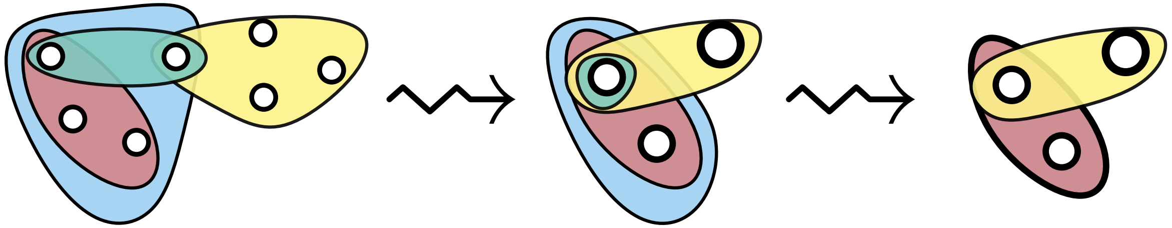
... with small size → $|e|-1$

Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets $\rightsquigarrow$ easier local search ✓

2: reduce **number** of nets $\rightsquigarrow$ easier initial partitioning ✓



$\Longrightarrow$ hypergraph-tailored rating functions:

$$r(u,v) := \sum_{\substack{\text{net } e \\ \text{containing } u,v}} \frac{\omega(e)}{|e|-1}$$

of heavy nets ...

... with small size

large number ...

3: maintain **structural similarity** $\rightsquigarrow$ good coarse solutions

$\Longrightarrow$ prefer clustering over matching

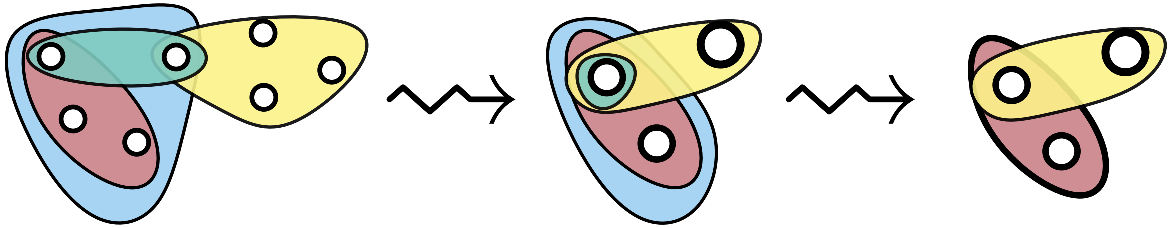
$\Longrightarrow$ ensure $\sim$balanced vertex weights

Clustering-based Coarsening

Main Design Goals:

1: reduce **size** of nets ⇝ easier local search ✓

2: reduce **number** of nets ⇝ easier initial partitioning ✓



⟹ hypergraph-tailored rating functions:

$$r(u,v) := \sum_{\substack{\text{net } e \\ \text{containing } u,v}} \frac{\omega(e)}{|e|-1}$$

of heavy nets ...

... with small size

large number ...

3: maintain **structural similarity** ⇝ good coarse solutions

⟹ prefer clustering over matching

⟹ ensure ∼balanced vertex weights

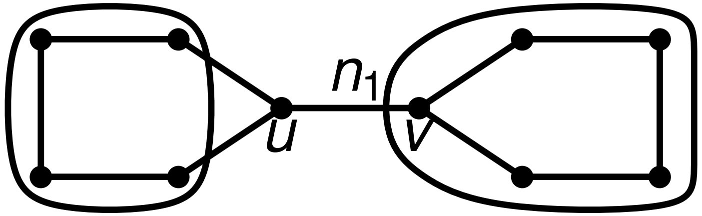
enough?

What could possibly go wrong?

... a lot:

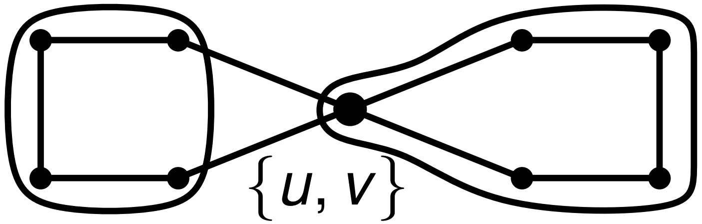
What could possibly go wrong?

... a lot:

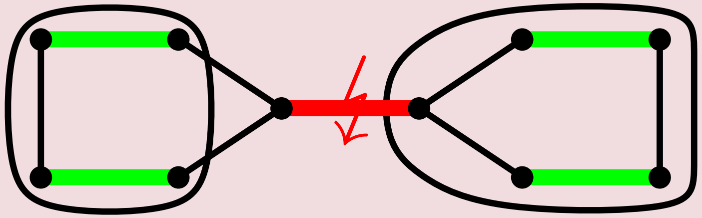


input

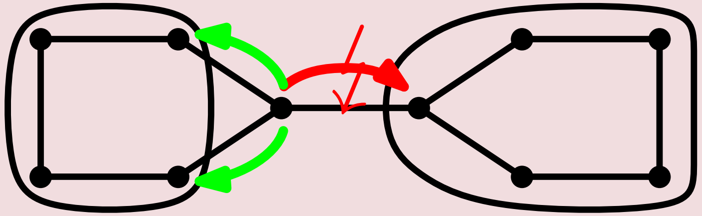
⇝



obscured clusters

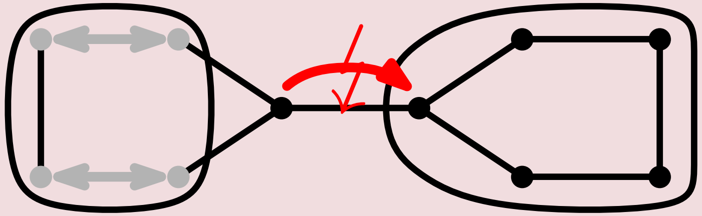


maximal matching

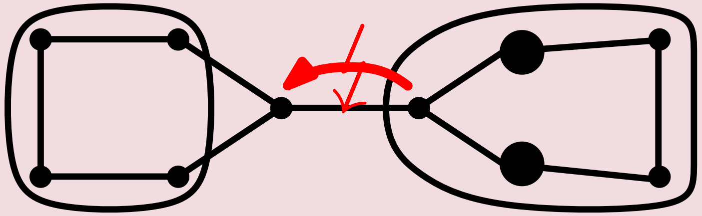


random tie-breaking

ISSUES



prefer unclustered



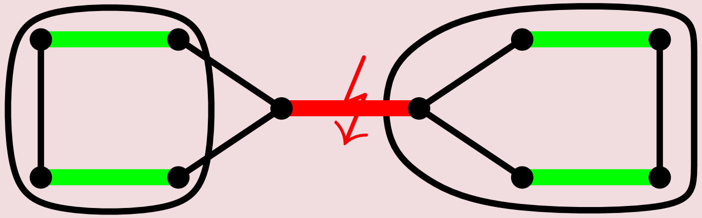
heavy neighbors

What could possibly go wrong?

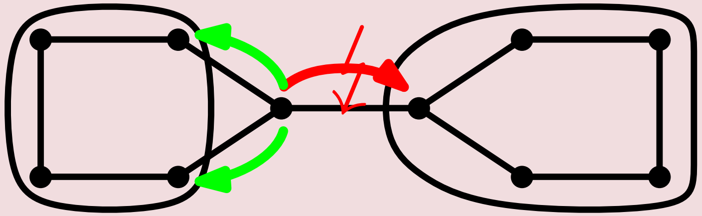
... a lot:

input

obscured clusters

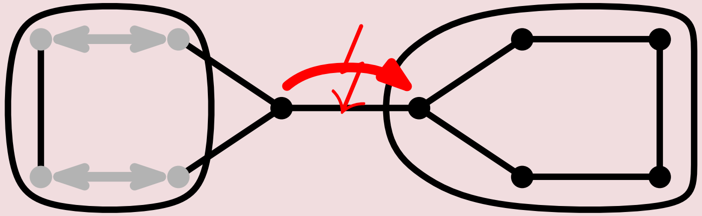


maximal matching

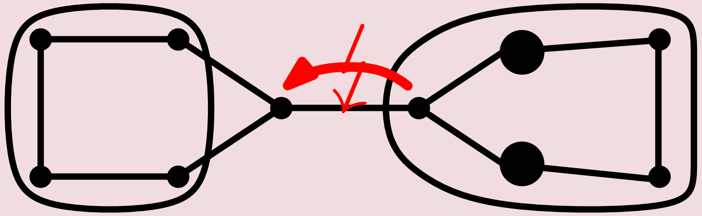


random tie-breaking

ISSUES



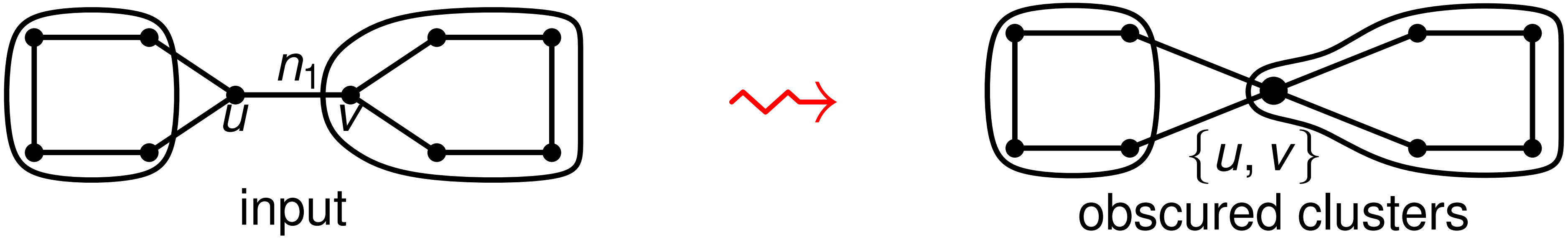
prefer unclustered



heavy neighbors

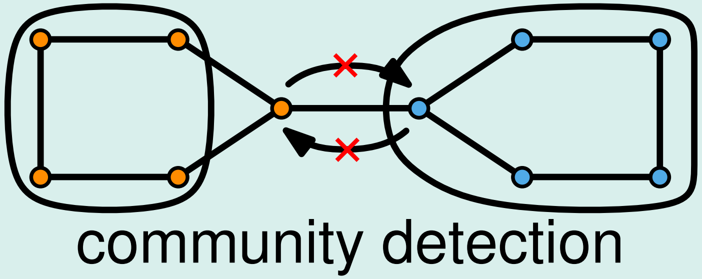
$\Longrightarrow$ **Problem:** relying **only** on **local** information!

Our Approach: Community-aware Coarsening



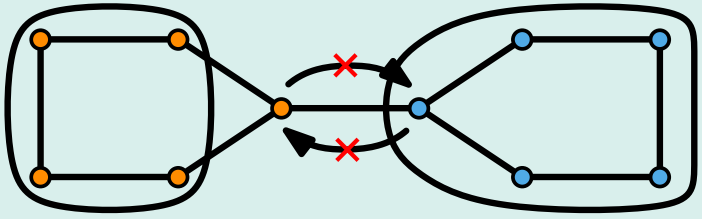
Our Approach: Community-aware Coarsening

SOLUTION



Our Approach: Community-aware Coarsening

SOLUTION

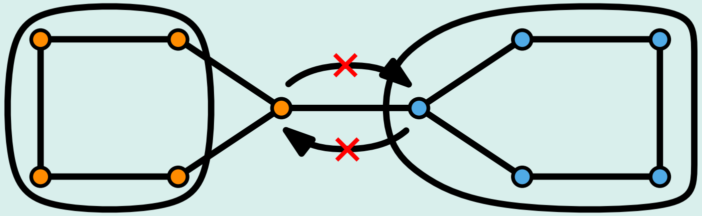


Framework:

- preprocessing: determine **community structure**
- only allow **intra-community** contractions

Our Approach: Community-aware Coarsening

SOLUTION



community detection

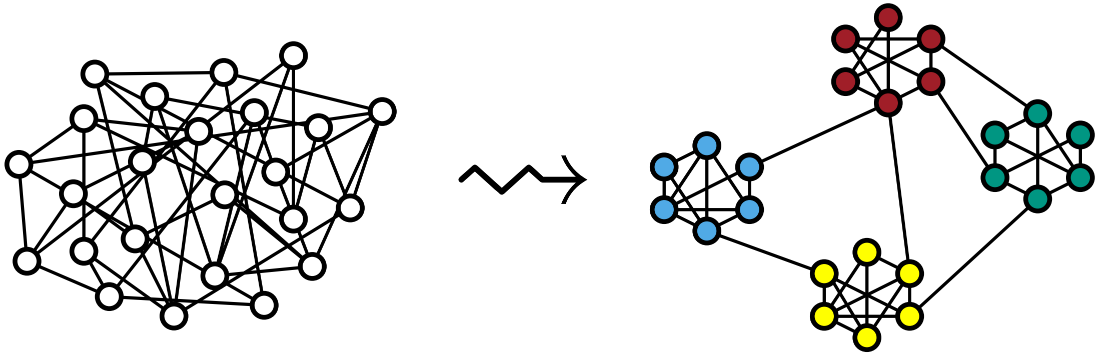
Framework:

- preprocessing: determine **community structure**
- only allow **intra-community** contractions

How?

Detecting Community Structure

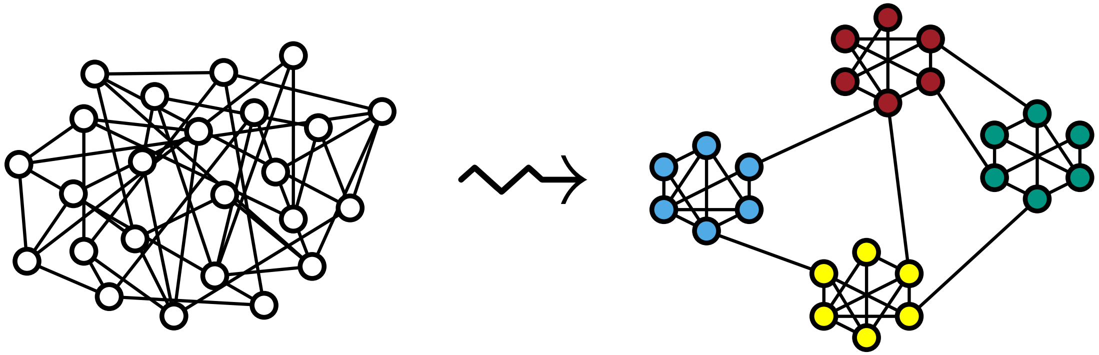
Goal: partition graph into **natural** groups $\mathcal{C}$



Community:
internally **dense**,
externally **sparse**
subgraph

Detecting Community Structure

Goal: partition graph into **natural** groups $\mathcal{C}$



Community:
internally **dense**,
externally **sparse**
subgraph

(One) **Formalization:** [Newman,Girvan 04]

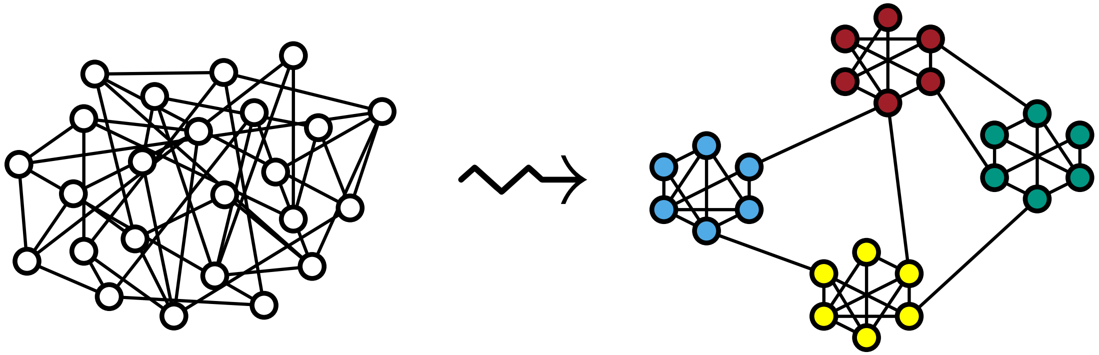
Modularity $\texttt{mod}(G,\mathcal{C}) := \texttt{cov}(G,\mathcal{C}) - \mathbb{E}[\texttt{cov}(G,\mathcal{C})]$

fraction of
intra-cluster edges

Coverage $\texttt{cov}(G,\mathcal{C}) := \sum_{C\in\mathcal{C}} \frac{|E(C)|}{|E|}$

Detecting Community Structure

Goal: partition graph into **natural** groups $\mathcal{C}$



Community:
internally **dense**,
externally **sparse**
subgraph

(One) **Formalization:** [Newman,Girvan 04]

Modularity $\texttt{mod}(G,\mathcal{C}) := \texttt{cov}(G,\mathcal{C}) - \mathbb{E}[\texttt{cov}(G,\mathcal{C})]$

Coverage $\texttt{cov}(G,\mathcal{C}) := \sum_{C\in\mathcal{C}} \frac{|E(C)|}{|E|}$

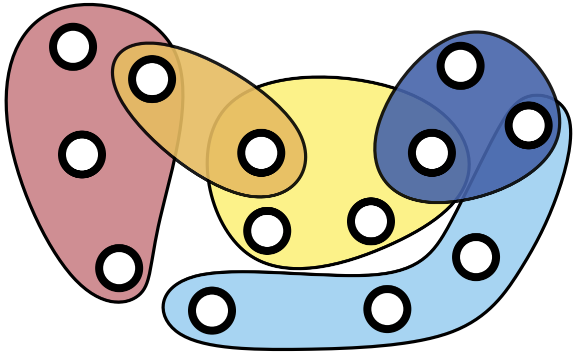
fraction of intra-cluster edges

Efficient Heuristic: Louvain Method (multilevel, local, greedy) [Blondel et al. 08]

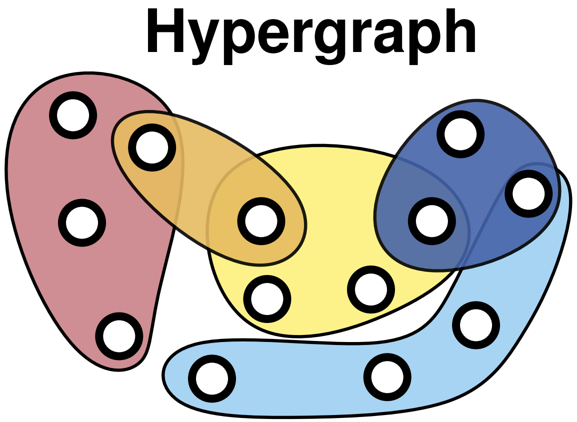
- repeatedly move nodes to neighbor communities
- coarsen graph & repeat

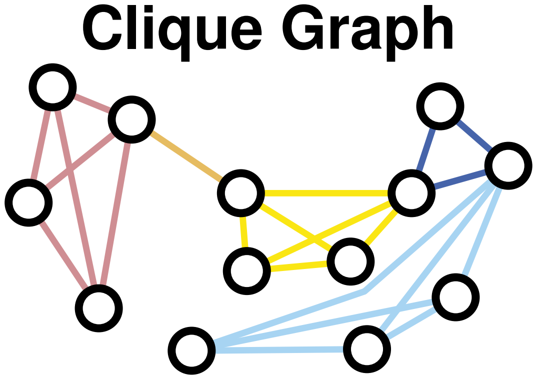
Graph Representations of Hypergraphs

Hypergraph



Graph Representations of Hypergraphs

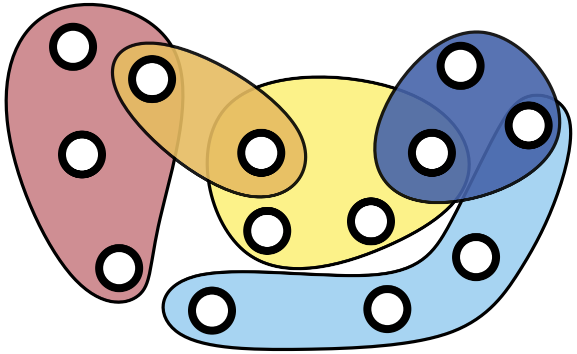




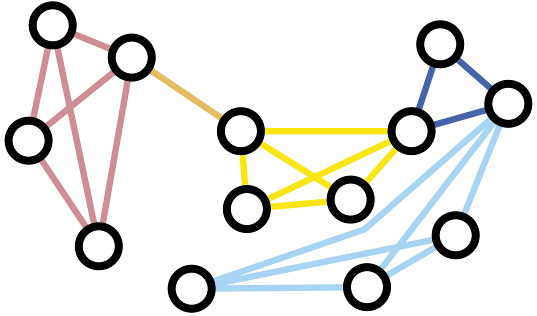
- destroys natural sparsity: $\binom{|e|}{2}$ edges per net e
- exaggerates importance of large nets

Graph Representations of Hypergraphs

Hypergraph

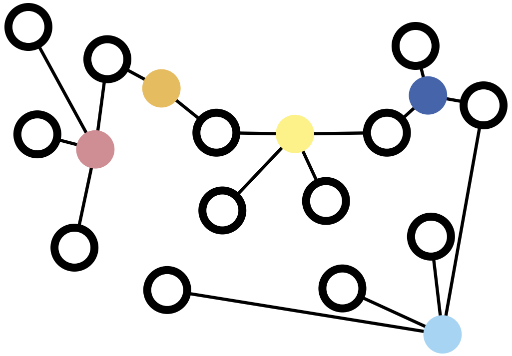


Clique Graph



- − destroys natural sparsity: $\binom{|e|}{2}$ edges per net e
- − exaggerates importance of large nets

Bipartite (Star) Graph



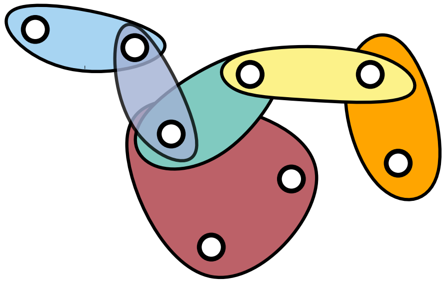
- + compact representation: $\mathcal{O}(|P|)$ space
- − nets become nodes & part of clustering

Bipartite Graphs: Modeling Pecularities

Density: $d := \frac{m}{n} = \frac{|P|/n}{|P|/m} = \frac{\overline{d(v)}}{\overline{|e|}}$

Bipartite Graphs: Modeling Pecularities

Density: $d := \frac{m}{n} = \frac{|P|/n}{|P|/m} = \frac{\overline{d(v)}}{\overline{|e|}}$

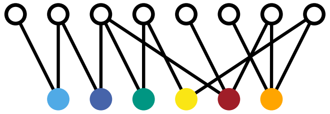


$$\mathbf{d} \approx 1$$

$$|V| \simeq |E|$$

Hypernodes ($\top$-nodes)

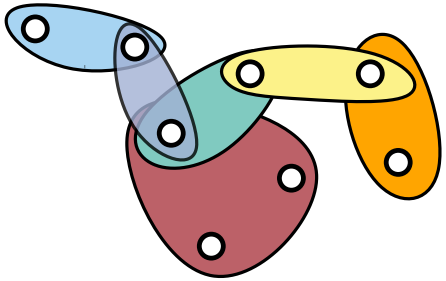
Hyperedges ($\perp$-nodes)

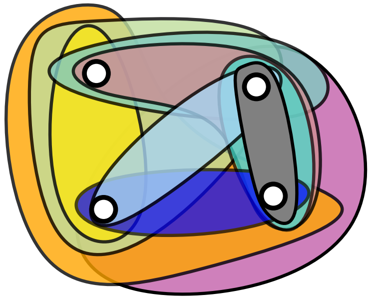


$$\overline{d(v)} \simeq \overline{|e|}$$

Bipartite Graphs: Modeling Pecularities

Density: $d := \frac{m}{n} = \frac{|P|/n}{|P|/m} = \frac{\overline{d(v)}}{\overline{|e|}}$



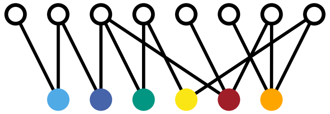


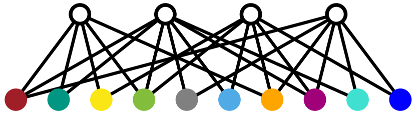
$\mathbf{d} \approx 1$ $\mathbf{d} \gg 1$

$|V| \simeq |E|$ $|V| \ll |E|$

Hypernodes ($\top$-nodes)

Hyperedges ($\perp$-nodes)

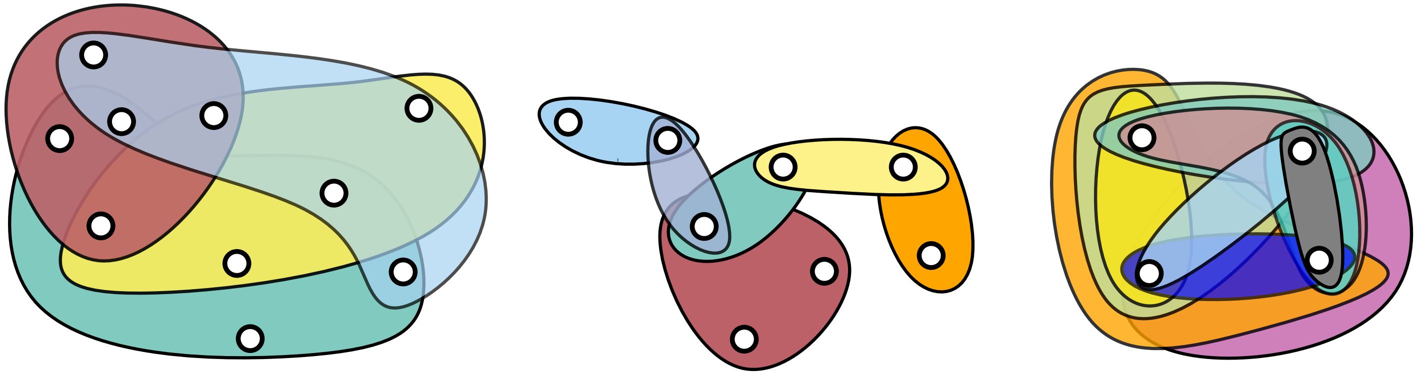




$\overline{d(v)} \simeq \overline{|e|}$ $\overline{d(v)} \gg \overline{|e|}$

Bipartite Graphs: Modeling Pecularities

Density: $d := \frac{m}{n} = \frac{|P|/n}{|P|/m} = \frac{\overline{d(v)}}{\overline{|e|}}$

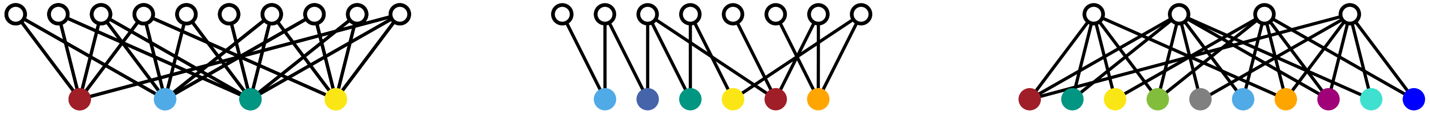


$\mathbf{d} \ll 1$ $\mathbf{d} \approx 1$ $\mathbf{d} \gg 1$

$|V| \gg |E|$ $|V| \simeq |E|$ $|V| \ll |E|$

Hypernodes ($\top$-nodes)

Hyperedges ($\perp$-nodes)



$\overline{d(v)} \ll \overline{|e|}$ $\overline{d(v)} \simeq \overline{|e|}$ $\overline{d(v)} \gg \overline{|e|}$

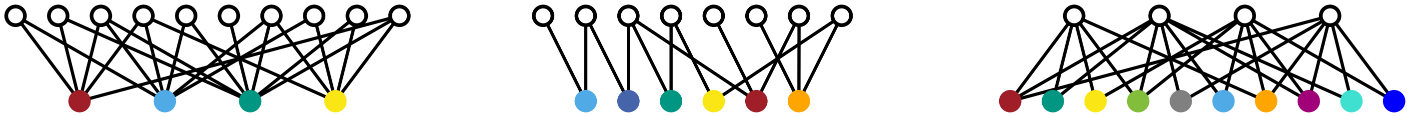
Bipartite Graphs: Modeling Pecularities

$\mathbf{d} \ll 1$ $\quad$ $\mathbf{d} \approx 1$ $\quad$ $\mathbf{d} \gg 1$

$|V| \gg |E|$ $\quad$ $|V| \simeq |E|$ $\quad$ $|V| \ll |E|$

Hypernodes ($\top$-nodes)

Hyperedges ($\perp$-nodes)



$\overline{d(v)} \ll \overline{|e|}$ $\quad$ $\overline{d(v)} \simeq \overline{|e|}$ $\quad$ $\overline{d(v)} \gg \overline{|e|}$

$\Longrightarrow$ addressed via **edge weights:**

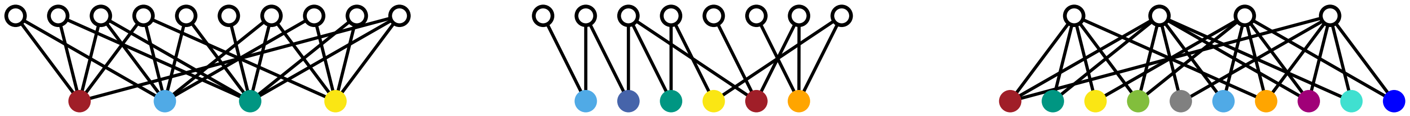
Bipartite Graphs: Modeling Pecularities

$\mathbf{d} \ll 1$ $\quad$ $\mathbf{d} \approx 1$ $\quad$ $\mathbf{d} \gg 1$

$\left|V\right| \gg \left|E\right|$ $\quad$ $\left|V\right| \simeq \left|E\right|$ $\quad$ $\left|V\right| \ll \left|E\right|$

Hypernodes ($\top$-nodes)

Hyperedges ($\bot$-nodes)

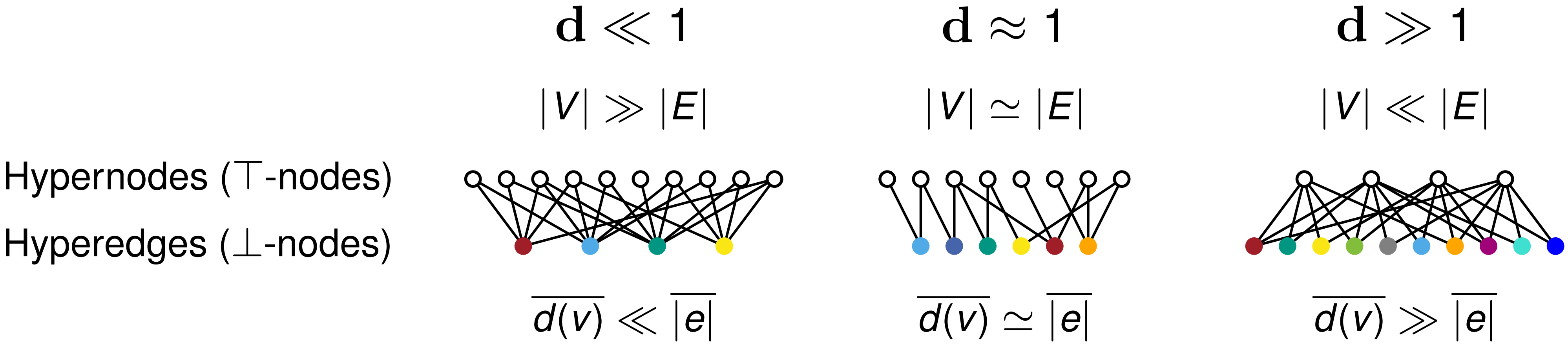


$\overline{d(v)} \ll \overline{\left|e\right|}$ $\quad$ $\overline{d(v)} \simeq \overline{\left|e\right|}$ $\quad$ $\overline{d(v)} \gg \overline{\left|e\right|}$

$\Longrightarrow$ addressed via **edge weights:**

- $\omega(v, e) := 1$ baseline

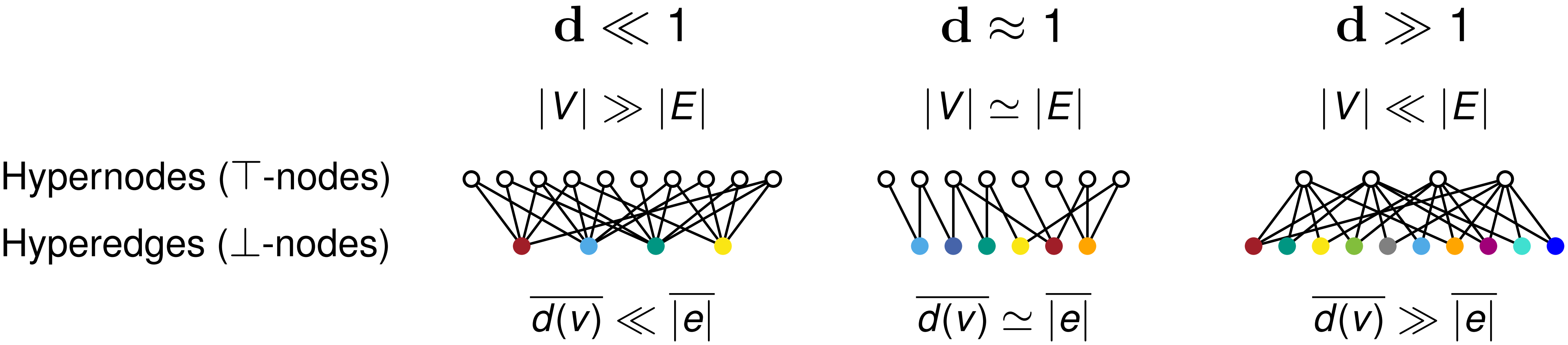
Bipartite Graphs: Modeling Pecularities



$\Longrightarrow$ addressed via **edge weights:**

- $\omega(v, e) := 1$ baseline
- $\omega_e(v, e) := \frac{1}{|e|}$ small nets $\rightsquigarrow$ higher influence

Bipartite Graphs: Modeling Pecularities



$\Longrightarrow$ addressed via **edge weights:**

- $\omega(v, e) := 1$ baseline
- $\omega_e(v, e) := \frac{1}{|e|}$ small nets ⇝ higher influence
- $\omega_{de}(v, e) := \frac{d(v)}{|e|}$ + high degree ⇝ higher influence

Experiments – Benchmark Setup

- System: 1 core of 2 Intel Xeon E5-2670 @ 2.6 Ghz, 64 GB RAM
- Hypergraphs:[1]

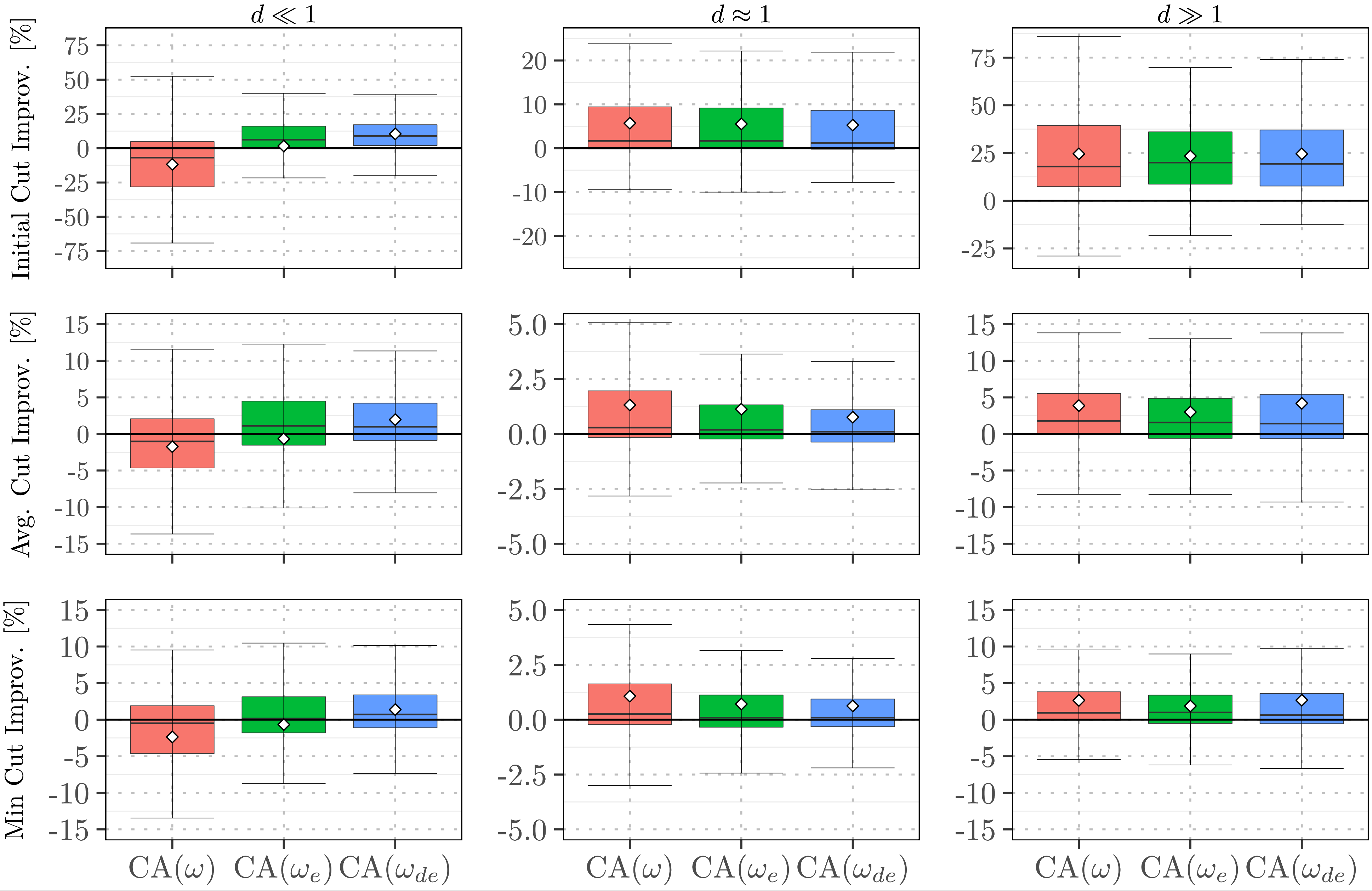
Application	VLSI		Sparse Matrix	SAT Solving		
Benchmark Set	ISPD98	DAC2012	UF-SPM	SAT14		
Representation	direct	direct	row-net	literal	primal	dual
Density Class	$d \approx 1$	$d \approx 1$	$d \ll 1, d \approx 1, d \gg 1$	$d \gg 1$	$d \gg 1$	$d \ll 1$
Community Str.	✓	✓	some instances	✓	✓	✓
# Hypergraphs	18	10	184	92	92	92

- $k \in \{2, 4, 8, 16, 32, 64, 128\}$ with imbalance: $\varepsilon = 3\%$
- 8 hours time limit / instance
- Comparing **KaHyPar-CA** with:
  - KaHyPar-K
  - hMetis-R & hMetis-K
  - PaToH-Default & PaToH-Quality

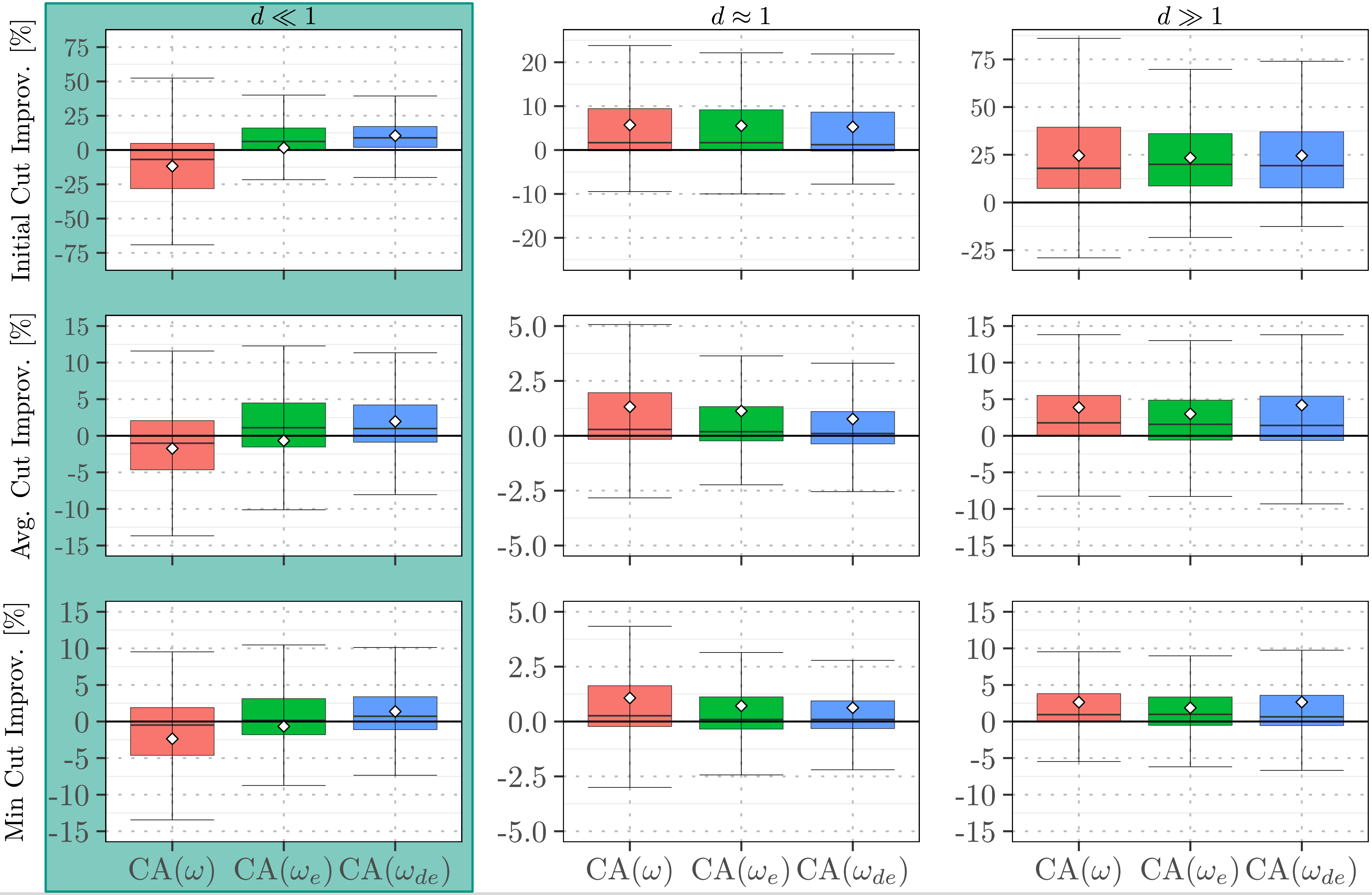
---

[1] available @ https://algo2.iti.kit.edu/schlag/sea2017/

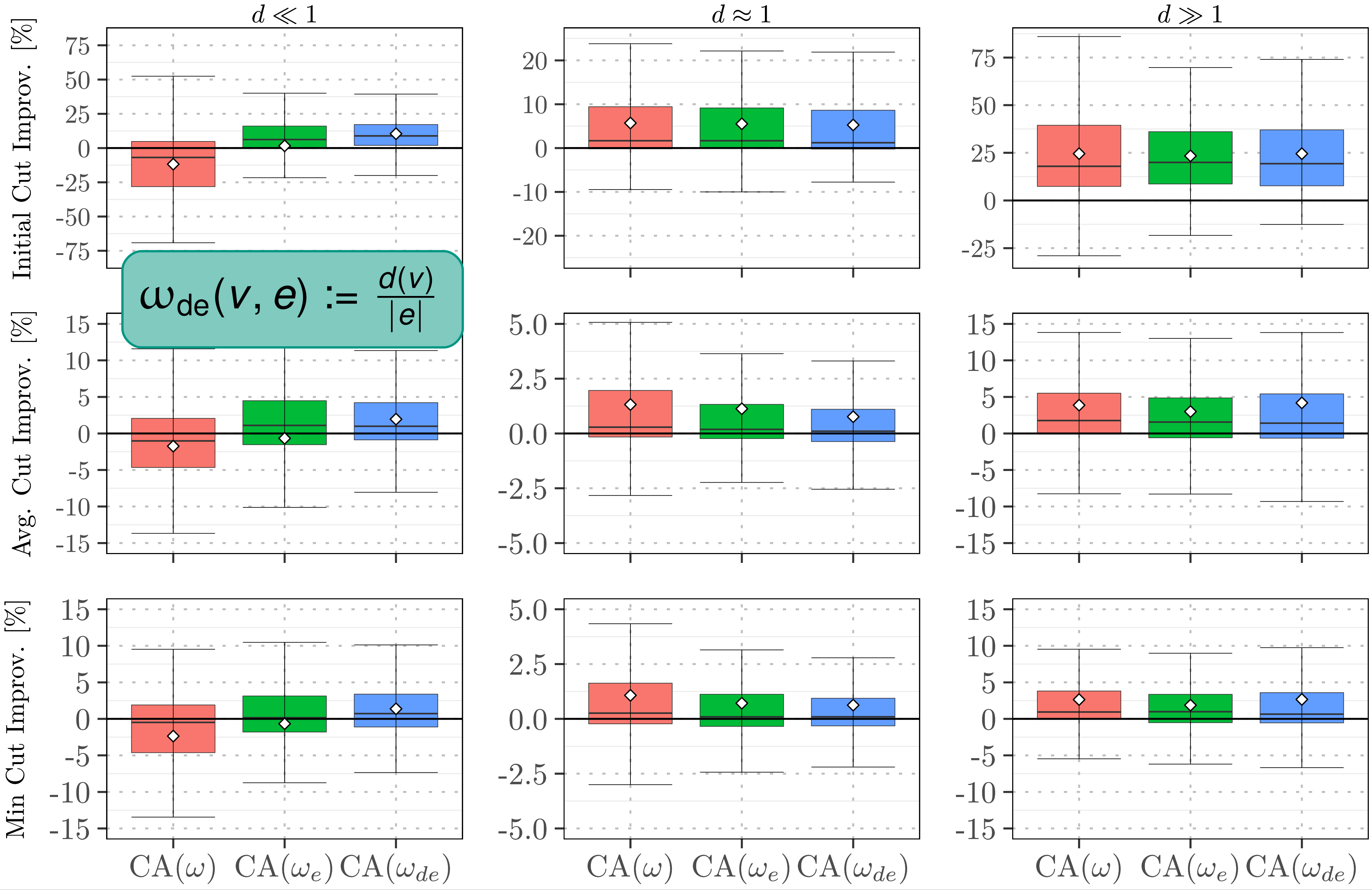
Comparison of Edge Weighting Schemes



Comparison of Edge Weighting Schemes

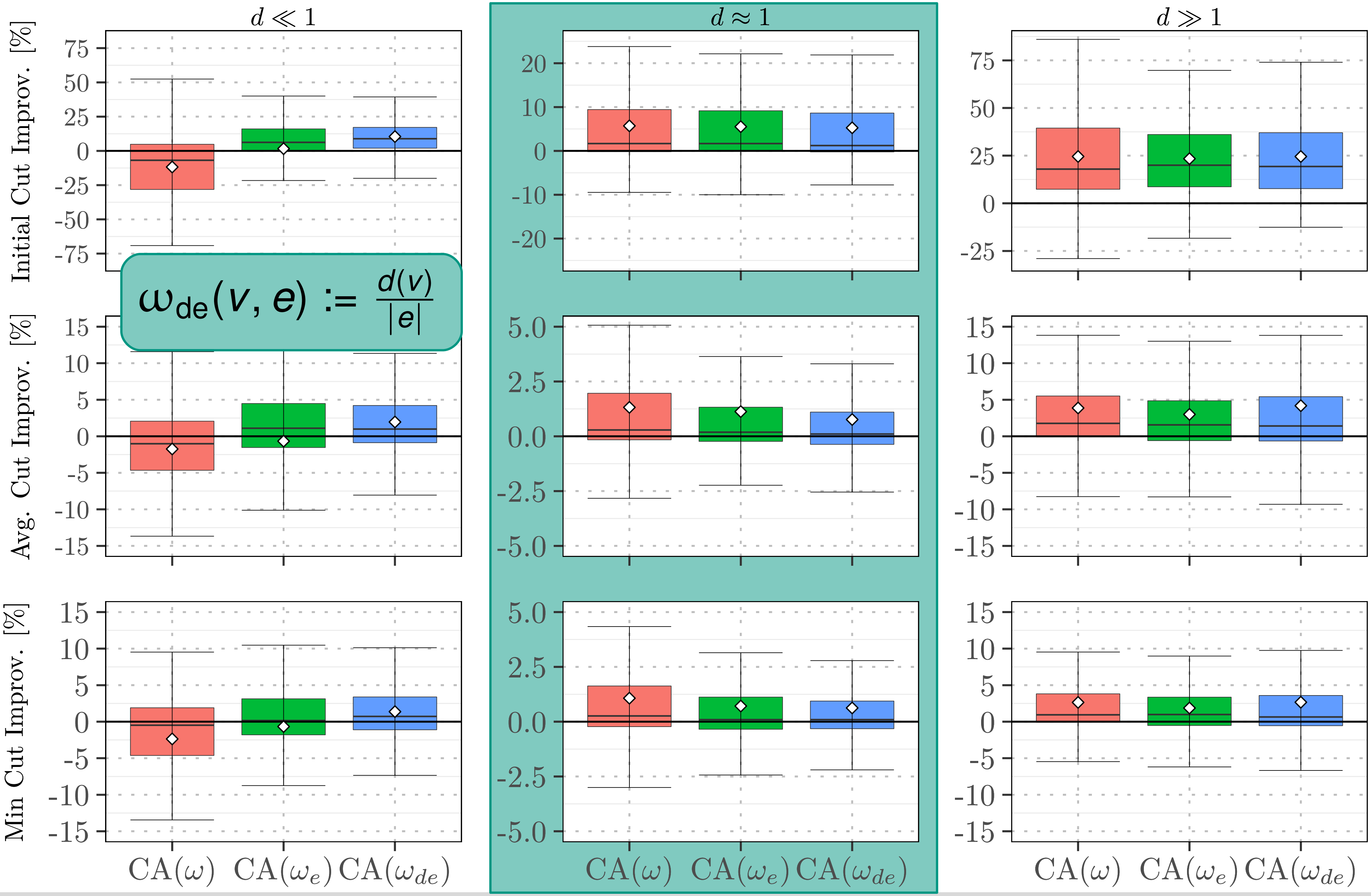


Comparison of Edge Weighting Schemes



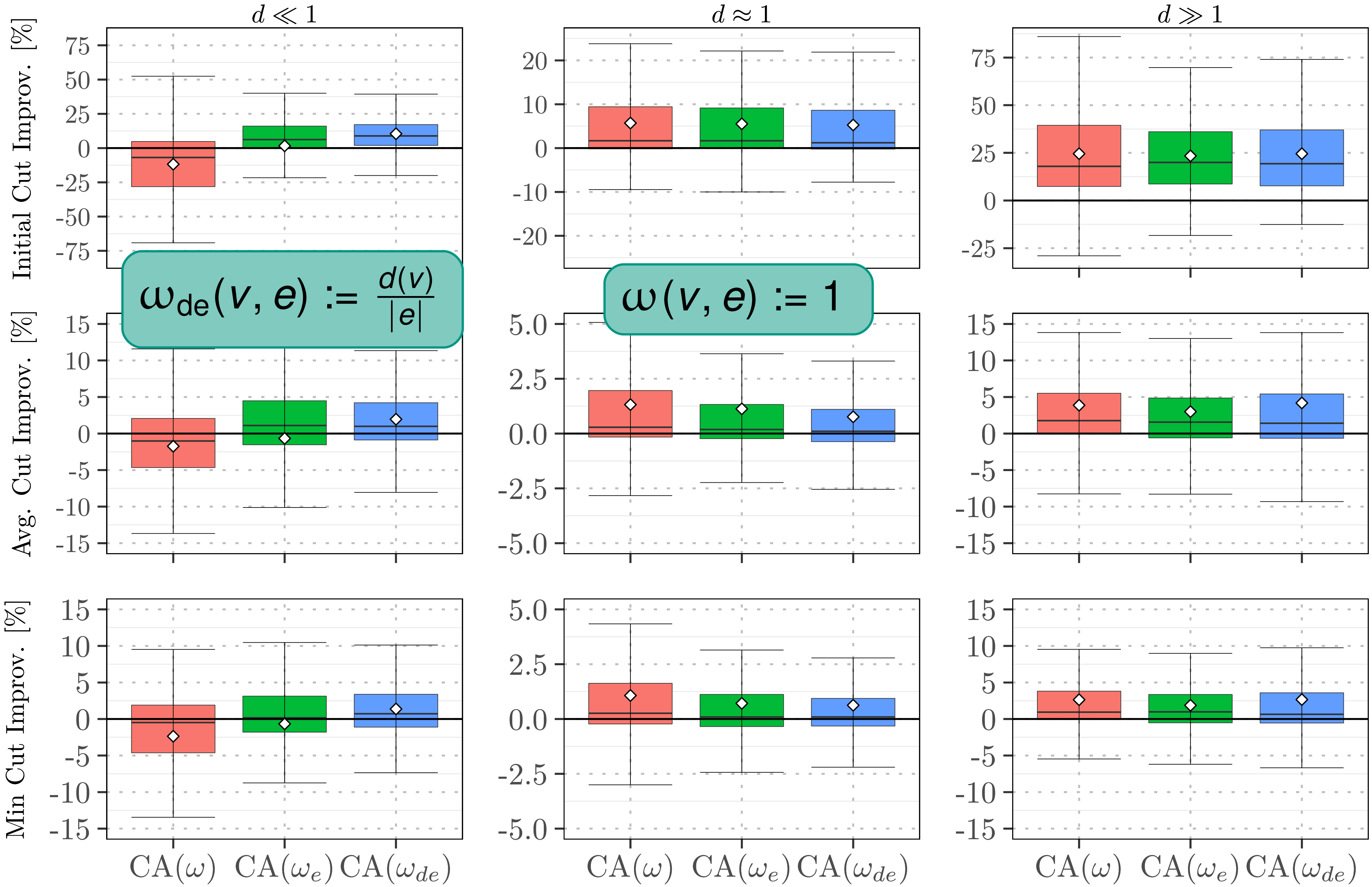
$$\omega_{\mathrm{de}}(v, e) := \frac{d(v)}{|e|}$$

Comparison of Edge Weighting Schemes



$$\omega_{\text{de}}(v, e) := \frac{d(v)}{|e|}$$

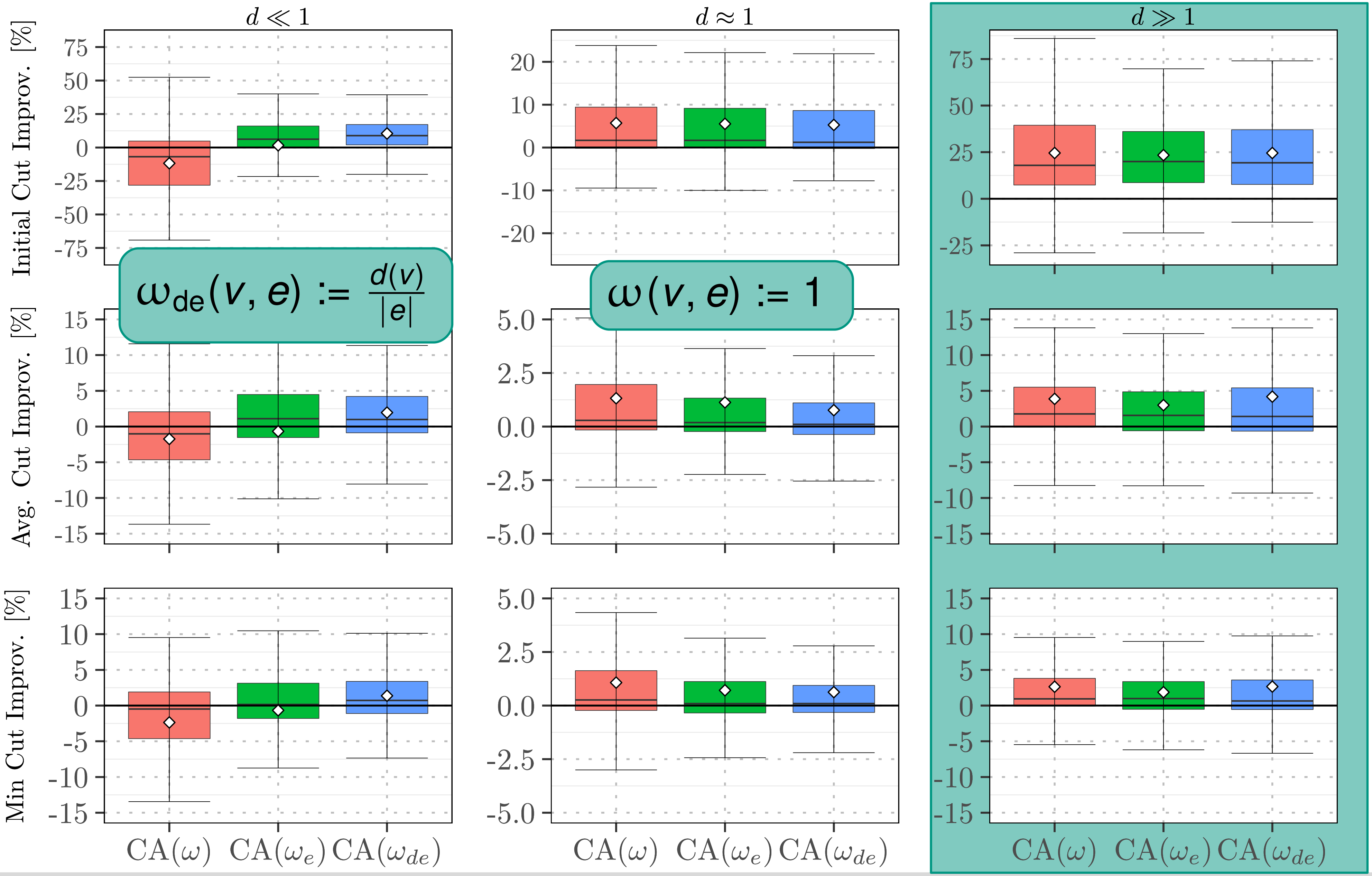
Comparison of Edge Weighting Schemes



$\omega_{\mathrm{de}}(v, e) := \frac{d(v)}{|e|}$

$\omega(v, e) := 1$

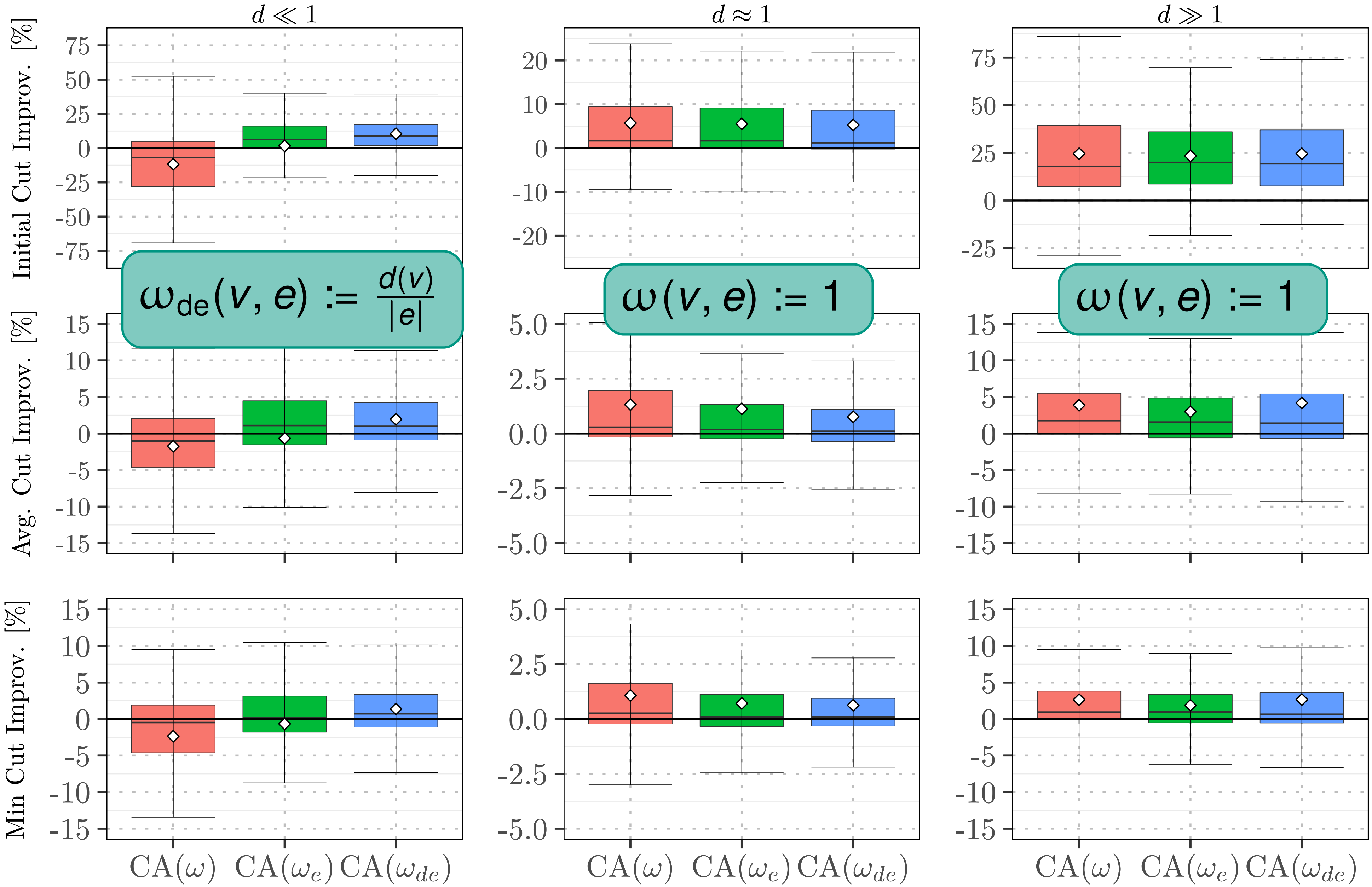
Comparison of Edge Weighting Schemes



$\omega_{\mathrm{de}}(v, e) := \frac{d(v)}{|e|}$

$\omega(v, e) := 1$

Comparison of Edge Weighting Schemes

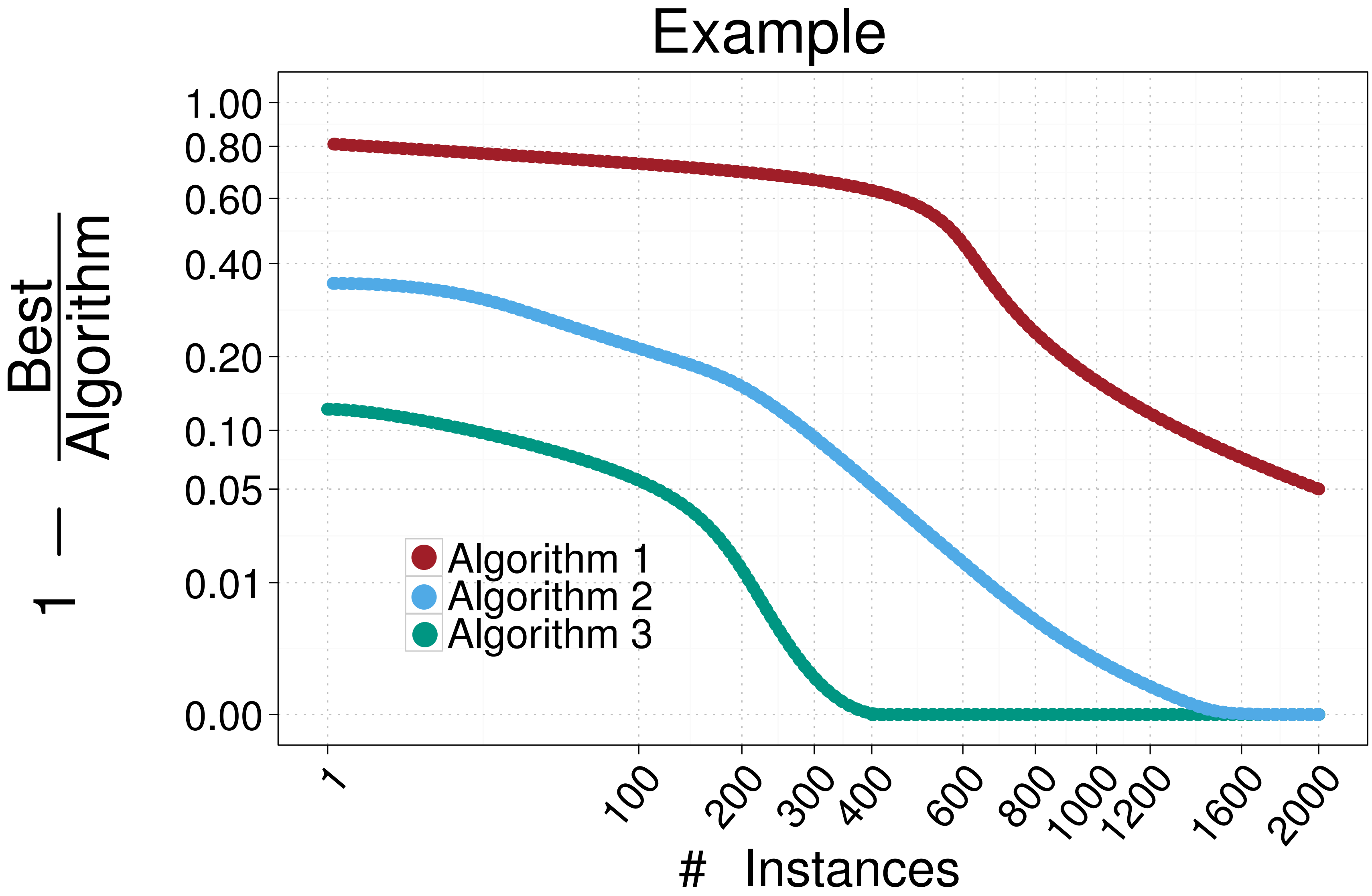


$\omega_{\text{de}}(v, e) := \frac{d(v)}{|e|}$

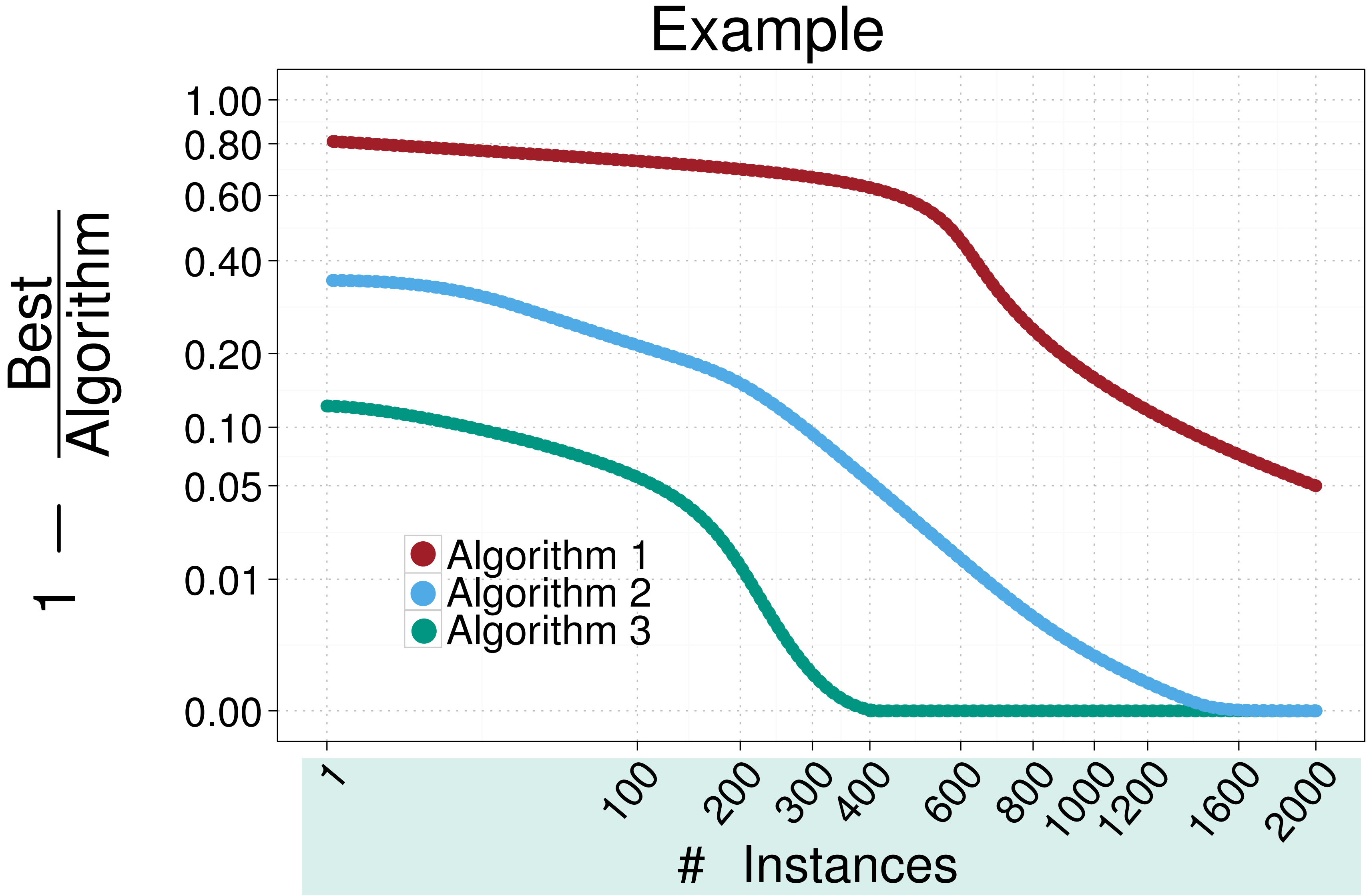
$\omega(v, e) := 1$

$\omega(v, e) := 1$

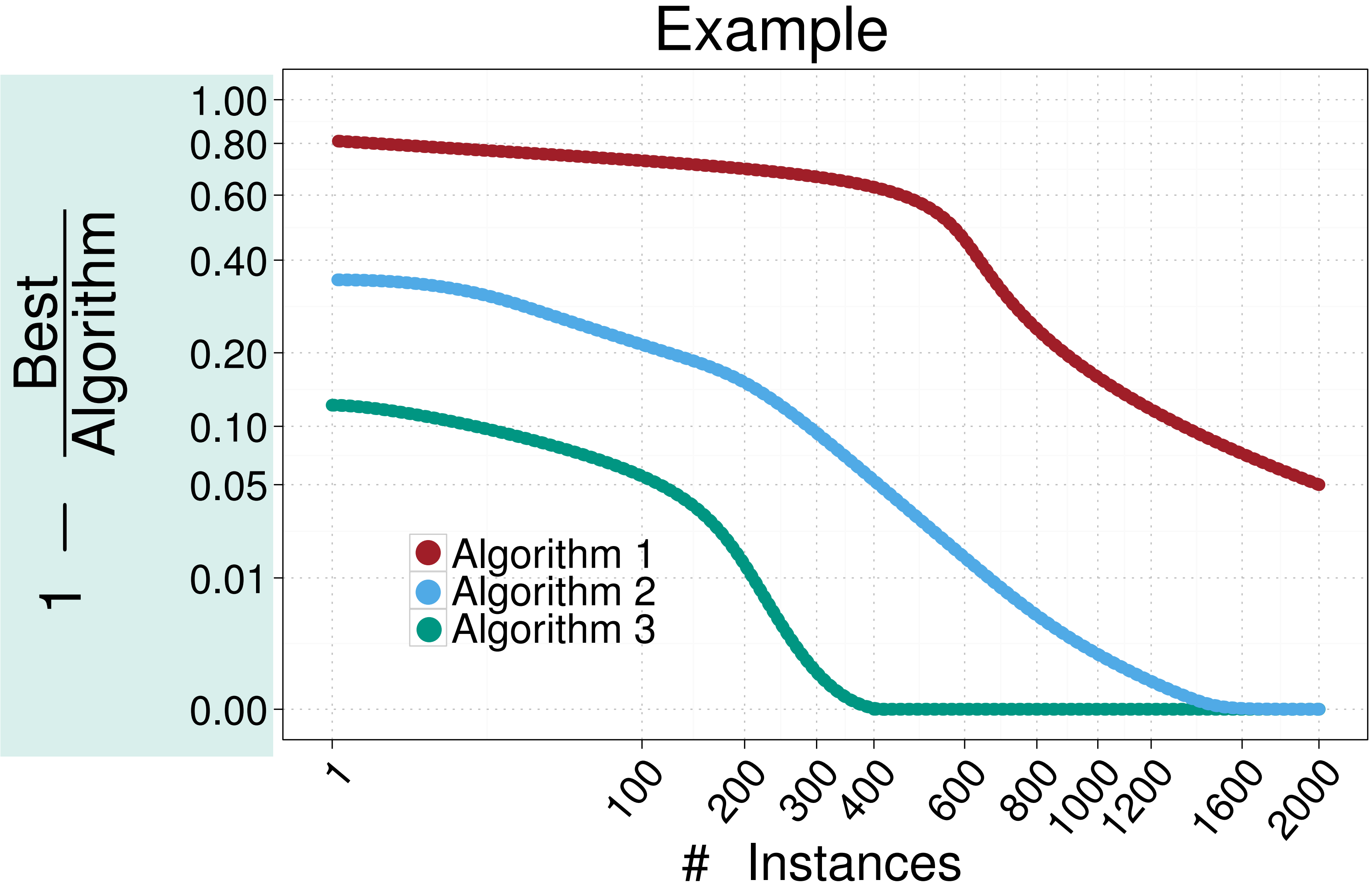
Experimental Results – Partitioning Quality



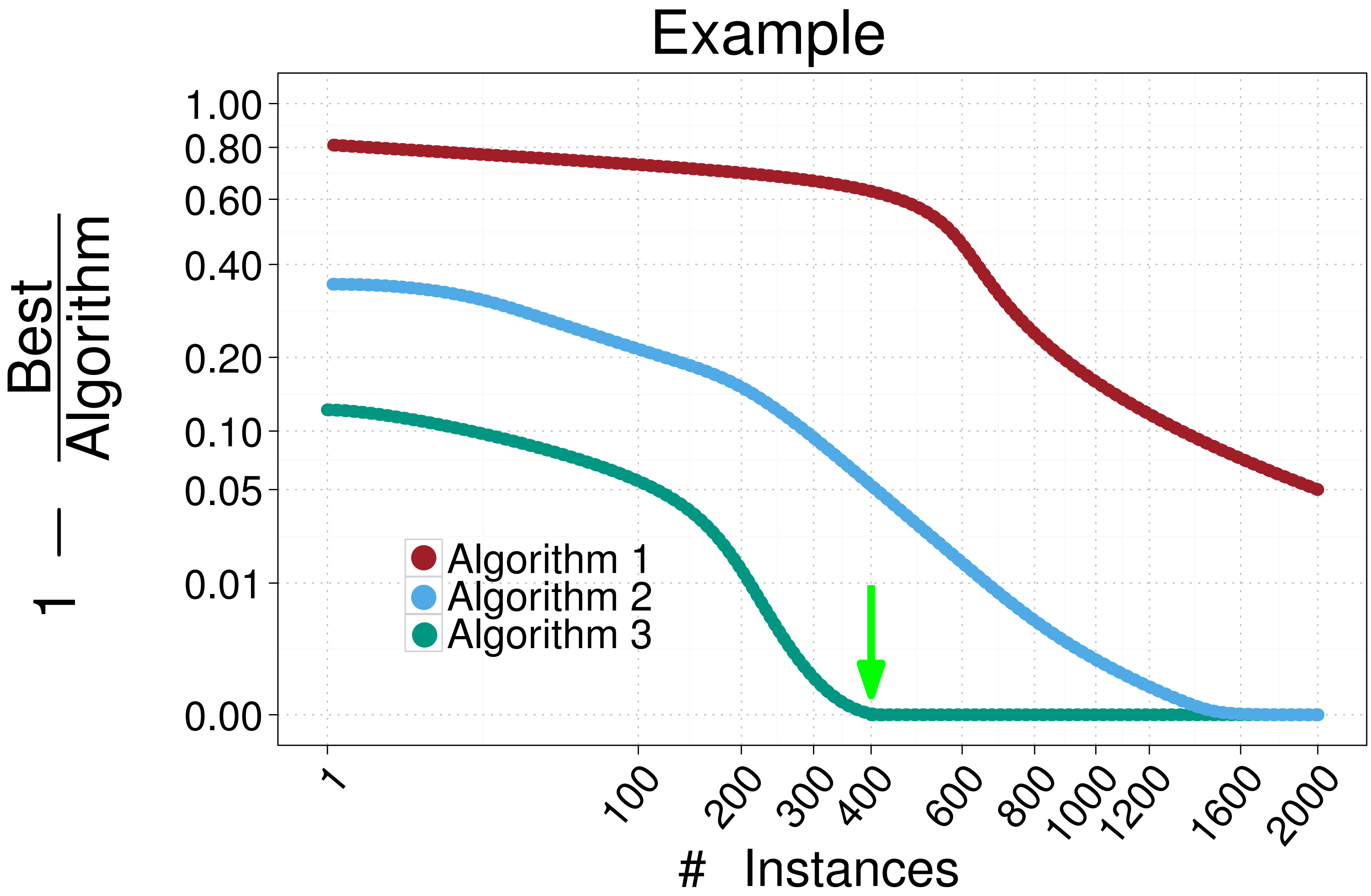
Experimental Results – Partitioning Quality



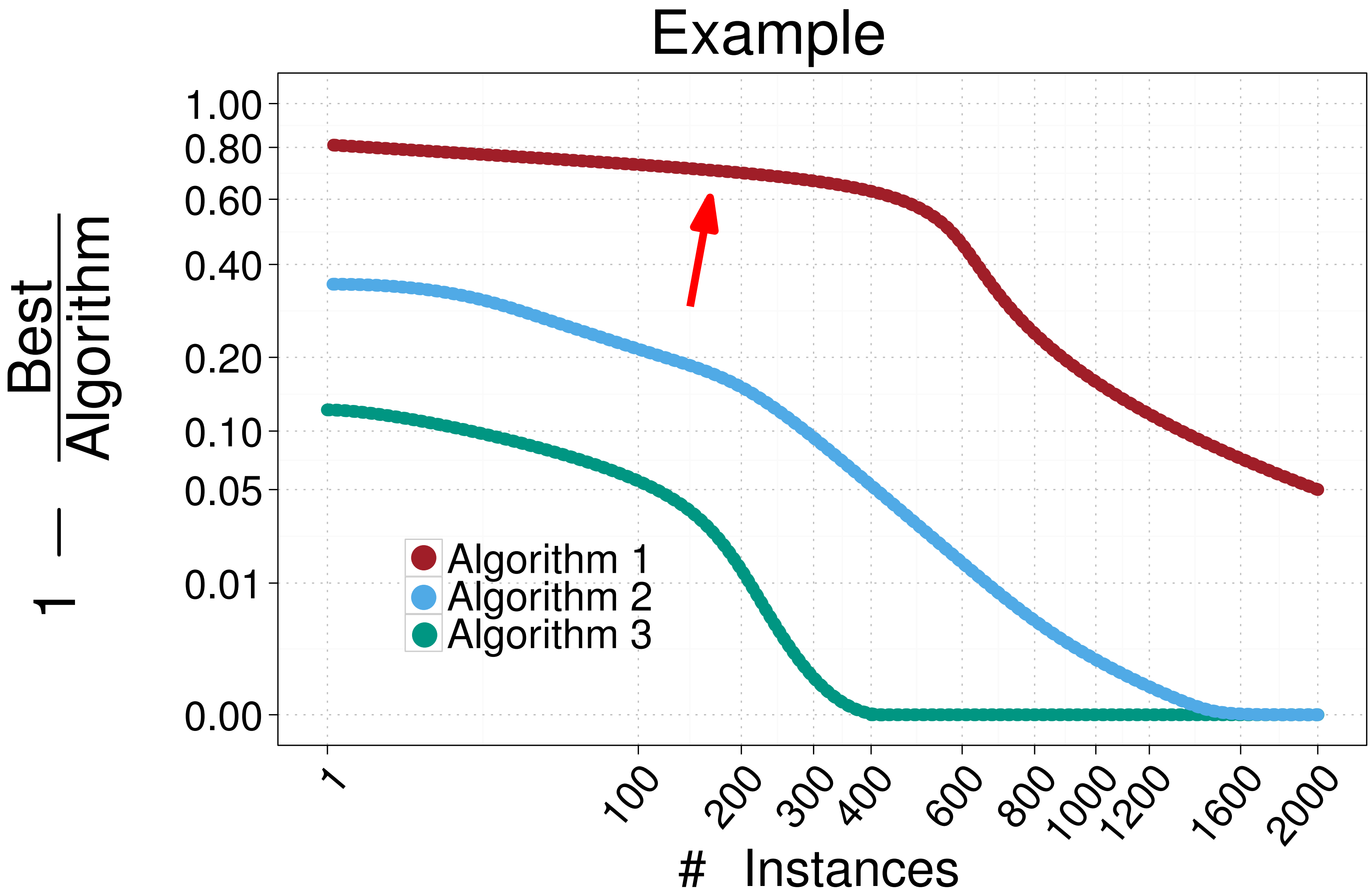
Experimental Results – Partitioning Quality



Experimental Results – Partitioning Quality

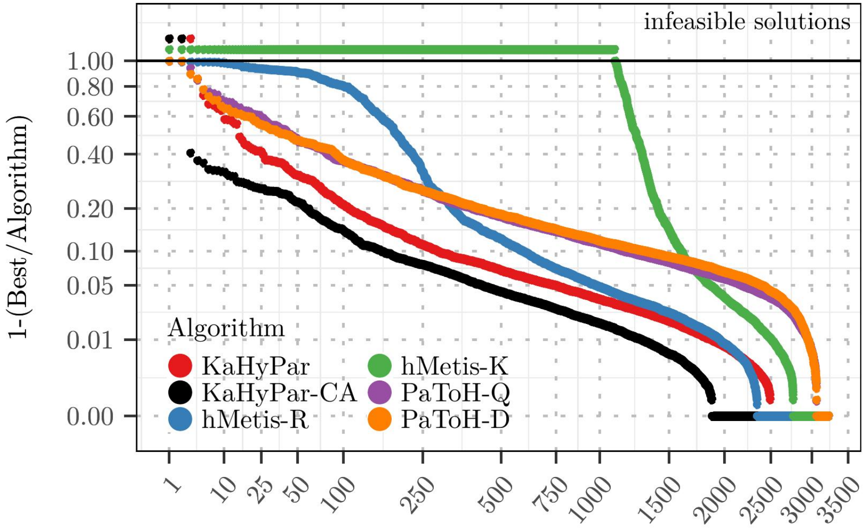


Experimental Results – Partitioning Quality

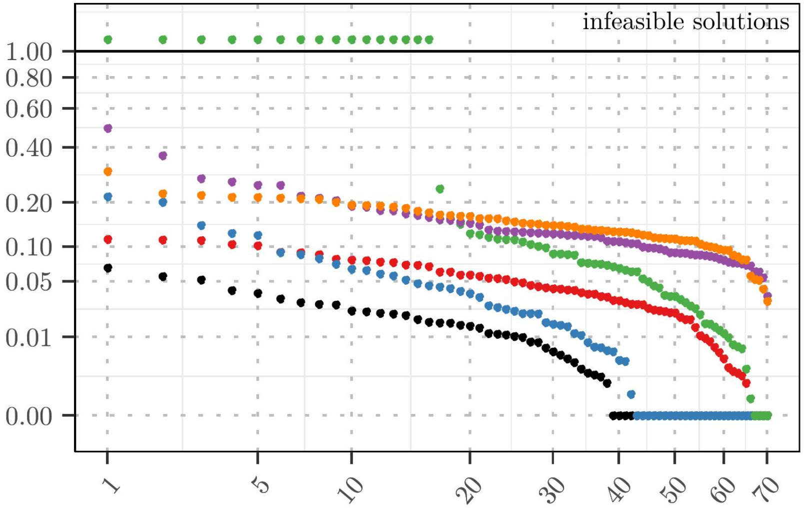


Experimental Results - Quality

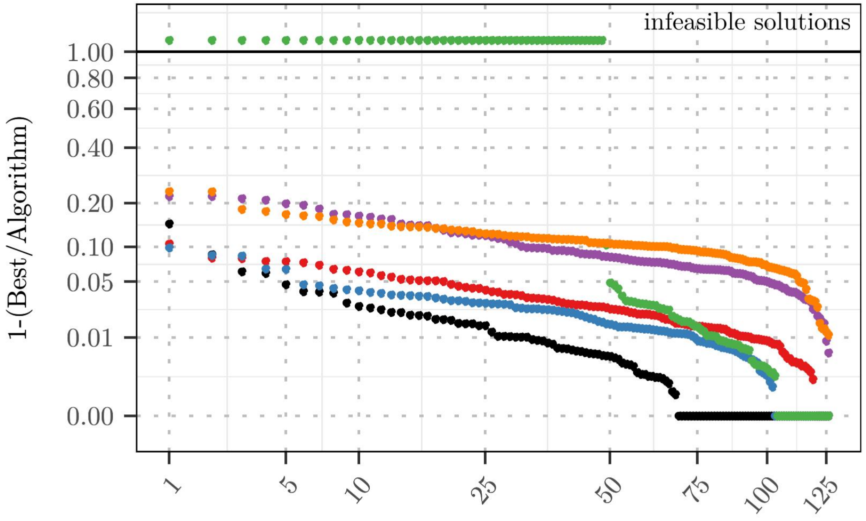
All Instances



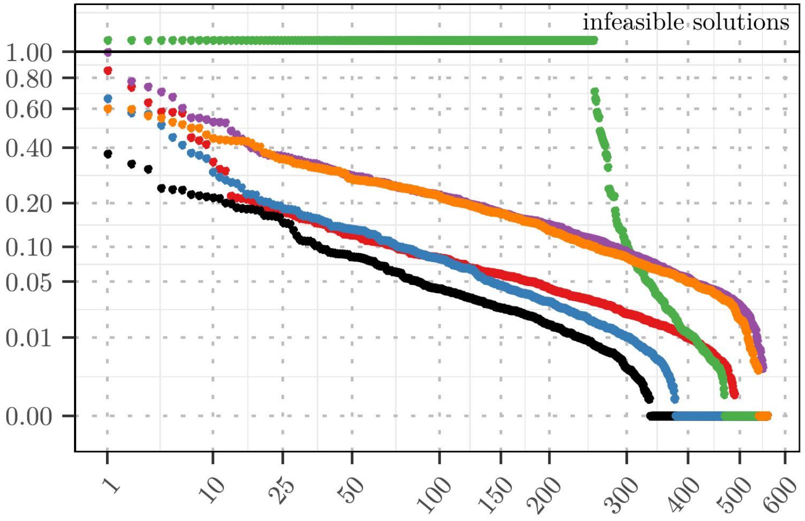
$d \approx 1$: DAC2012



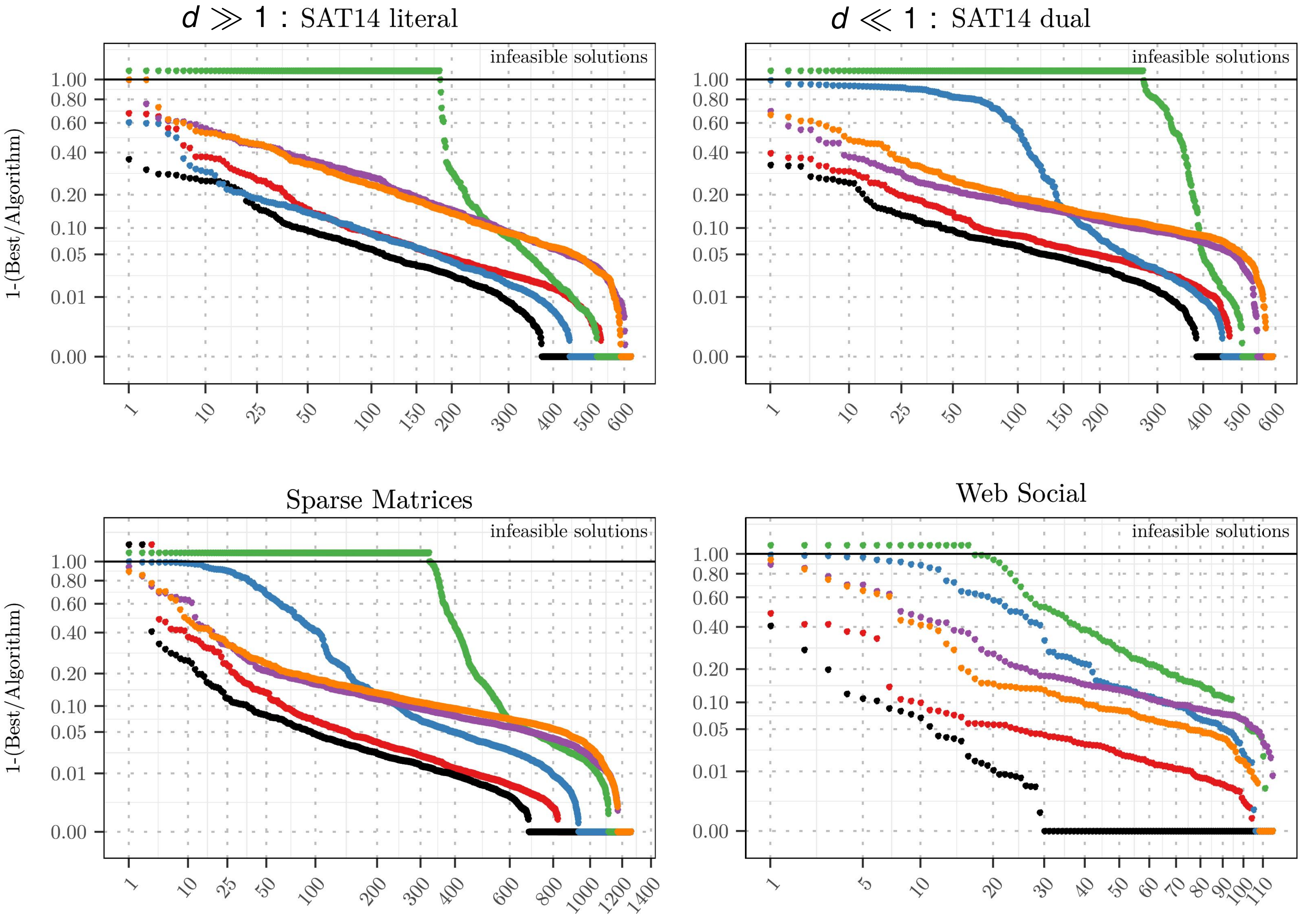
$d \approx 1$: ISPD98



$d \gg 1$: SAT14 primal



Experimental Results - Quality



Experimental Results - Running Time

Algorithm	Running Time [s]							
	All	DAC2012	ISPD98	Primal	Literal	Dual	SPM	WebSocial
KaHyPar	20.4	289.5	8.1	15.6	30.6	57.8	10.9	66.7
KaHyPar-CA	31.0	369.0	12.3	32.9	64.7	68.3	13.9	67.1
hMetis-R	79.2	446.4	29.0	66.2	142.1	200.4	41.8	89.7
hMetis-K	57.9	240.9	23.2	44.2	94.9	125.6	36.0	111.9
PaToH-Q	5.9	28.3	1.9	6.9	9.2	10.6	3.4	4.7
PaToH-D	1.2	6.5	0.4	1.1	1.6	2.9	0.8	0.9

Experimental Results - Running Time

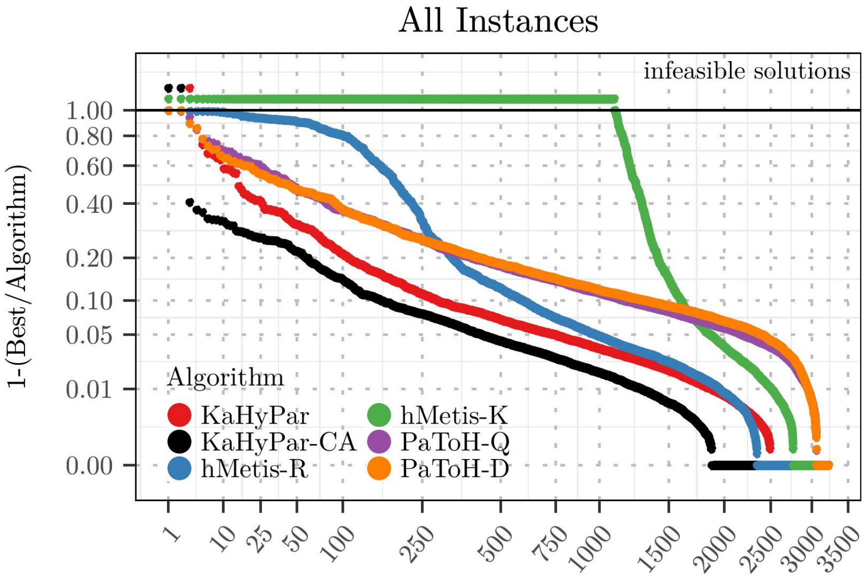
Algorithm	Running Time [s]							
	All	DAC2012	ISPD98	Primal	Literal	Dual	SPM	WebSocial
KaHyPar	20.4	289.5	8.1	15.6	30.6	57.8	10.9	66.7
KaHyPar-CA	31.0	369.0	12.3	32.9	64.7	68.3	13.9	67.1
hMetis-R	79.2	446.4	29.0	66.2	142.1	200.4	41.8	89.7
hMetis-K	57.9	240.9	23.2	44.2	94.9	125.6	36.0	111.9
PaToH-Q	5.9	28.3	1.9	6.9	9.2	10.6	3.4	4.7
PaToH-D	1.2	6.5	0.4	1.1	1.6	2.9	0.8	0.9

Conclusion & Discussion

KaHyPar-CA - Community-aware Coarsening

Community Detection via:

- modularity maximization
- Louvain Method (LM)
- bipartite, weighted graph



Future Work:

- speedup preprocessing: parallel LM
- resolution limit ⇝ multi-resolution modularity
- other formalizations:
  - Infomap
  - Surprise

KaHyPar-Framework
Open-Source on Github:
`https://git.io/vMBaR`

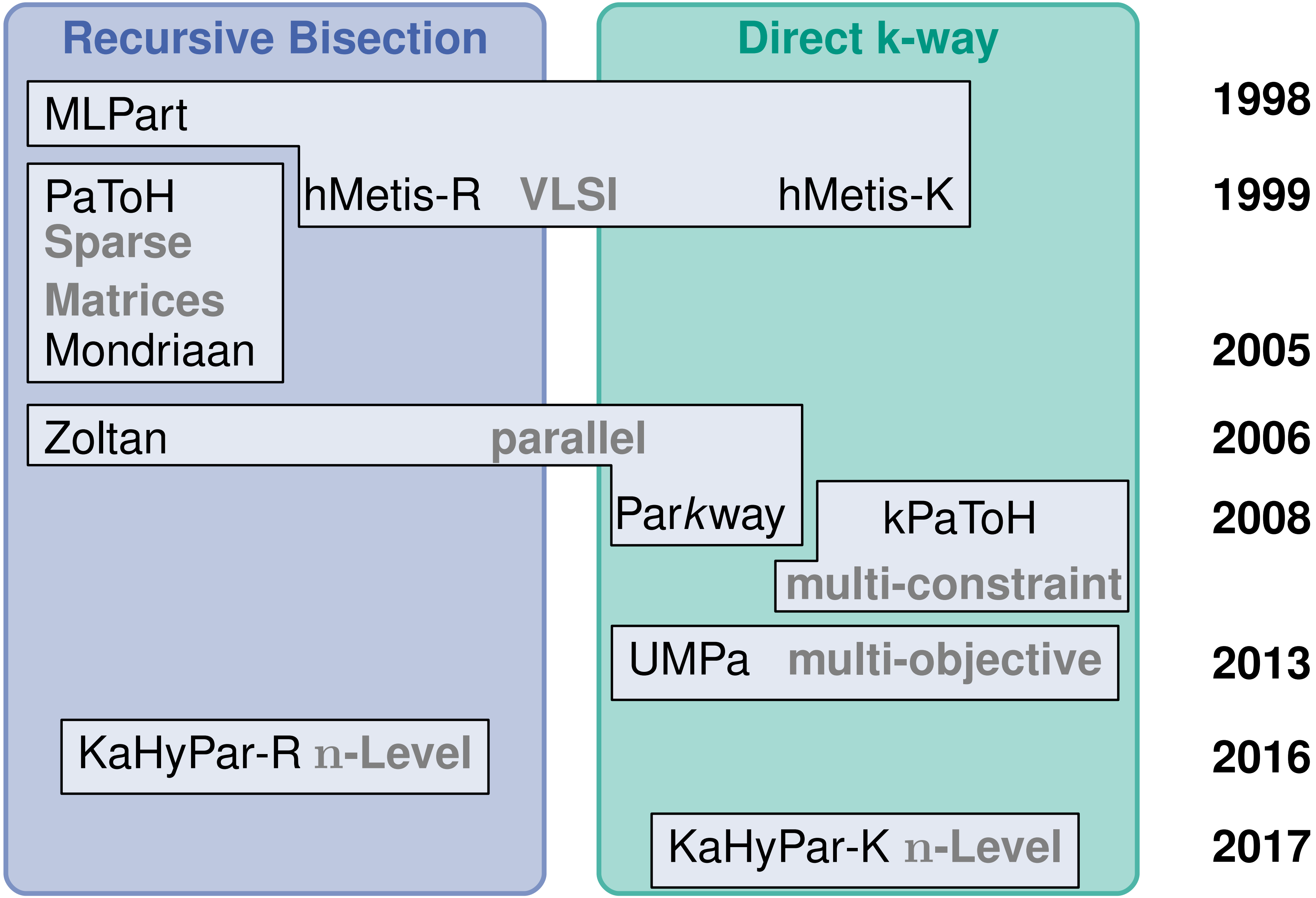
References

[Newman, Girvan 04]
M. E. J. Newman and M. Girvan. Finding and evaluating community structure in networks. Physical Review E, 69:026113, Feb 2004.

[Blondel et al. 08]
V. D. Blondel, J. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. Journal of Statistical Mechanics: Theory and Experiment, 2008(10):P10008, 2008.

[Karypis, Kumar 99]
G. Karypis and V. Kumar. Multilevel K-way Hypergraph Partitioning. In Proceedings of the 36th ACM/IEEE Design Automation Conference, pages 343–348. ACM, 1999.

Taxonomy of Hypergraph Partitioning Tools



Recursive Bisection

Direct k-way

1998 MLPart

1999 hMetis-R **VLSI** hMetis-K

PaToH **Sparse Matrices** Mondriaan

2005

2006 Zoltan **parallel**

2008 Parkway, kPaToH **multi-constraint**

2013 UMPa **multi-objective**

2016 KaHyPar-R **n-Level**

2017 KaHyPar-K **n-Level**