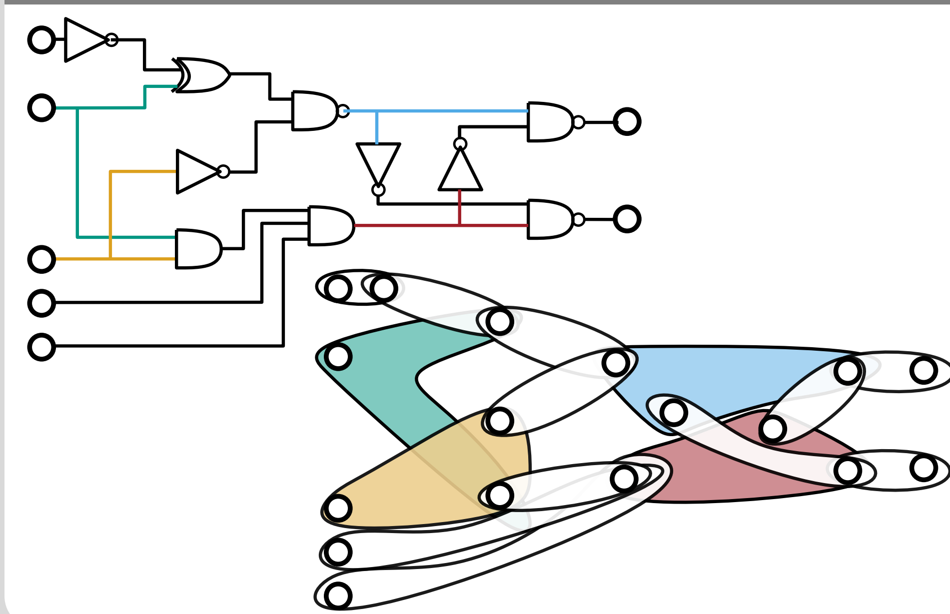
Karlsruhe Institute of Technology

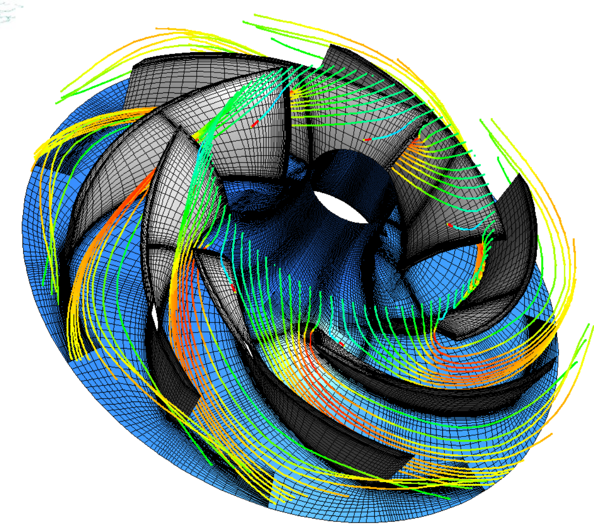
High-Quality (Hyper)Graph Partitioning

ICIAM · Combinatorial Scientific Computing · July 15th, 2019

Y. Akhremtsev, T. Heuer, P. Sanders, S. Schlag, C. Schulz, D. Seemaier, D. Strash

INSTITUTE OF THEORETICAL INFORMATICS · ALGORITHMICS GROUP





Research Areas in Peter Sanders' Group

Graph Partitioning

Shared-Memory Data Structures

Route Planning

Text Indexing

Hypergraph Partitioning

SAT Solving

Parallel Sorting

Communication-Efficient Algorithms

Graph Generators

Institute of Theoretical Informatics
Algorithmics Group

This Talk: Hypergraph & Graph Partitioning

Graph Partitioning

Shared-Memory
Data Structures

Route Planning

Text Indexing

Hypergraph Partitioning

SAT Solving

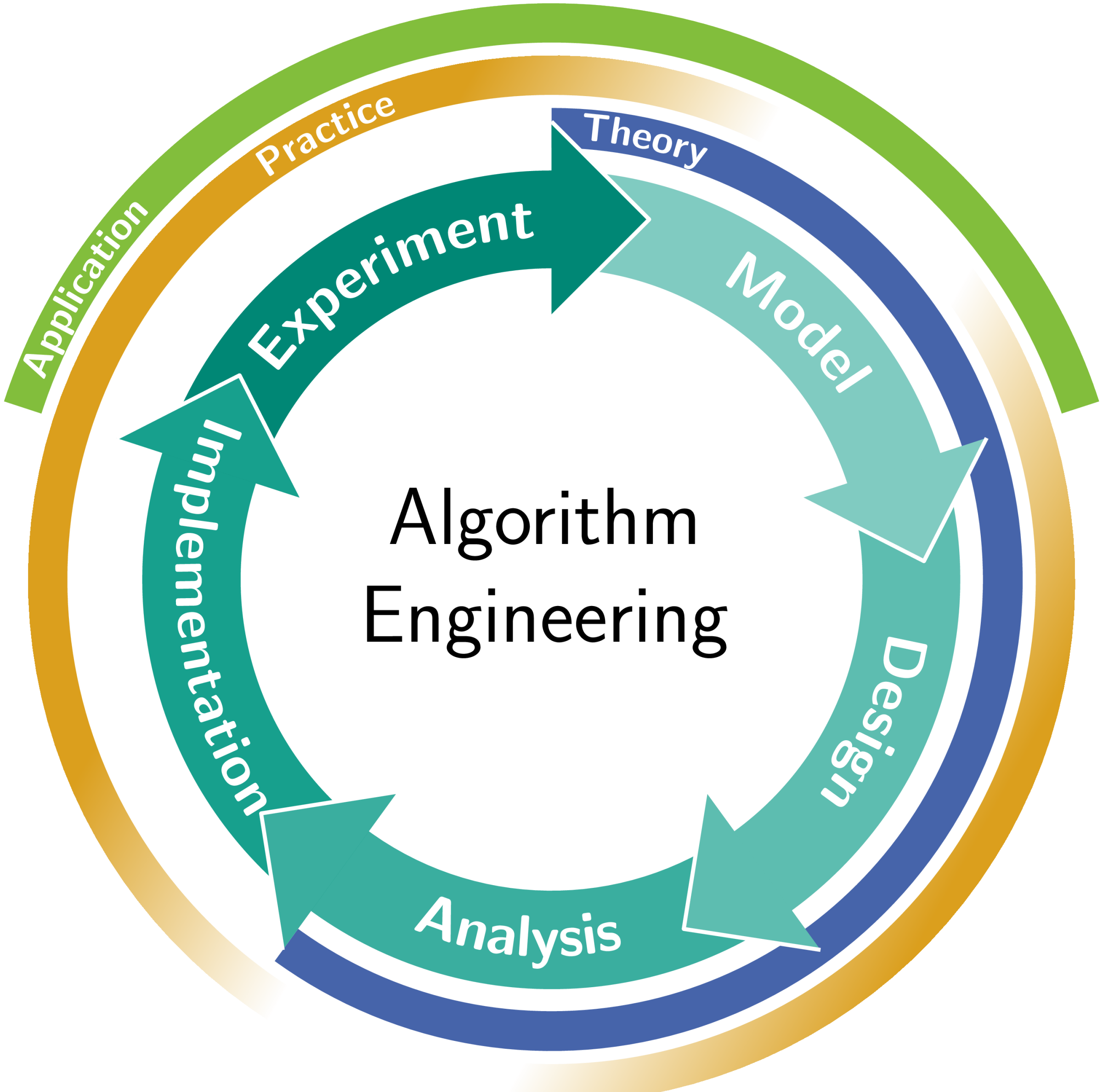
Parallel Sorting

Communication-Efficient
Algorithms

Graph Generators

Institute of Theoretical Informatics
Algorithmics Group

Research Methodology

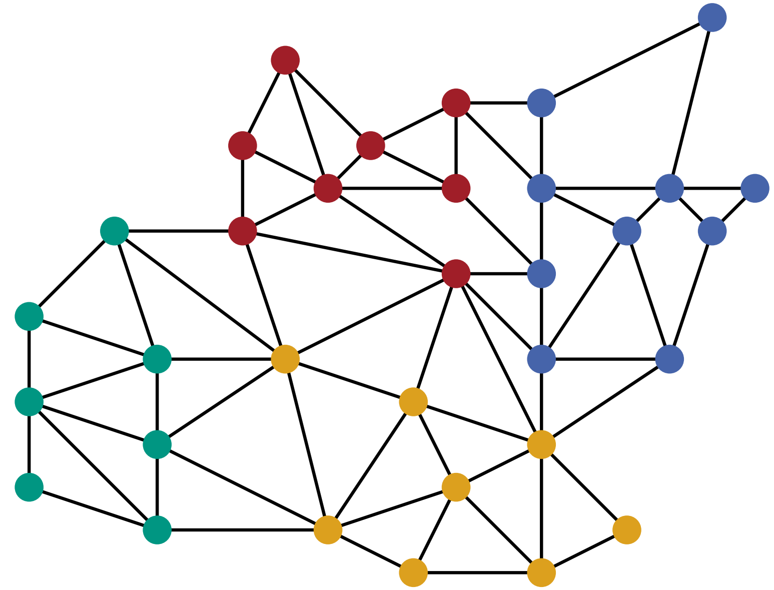


Graphs and Hypergraphs

Karlsruhe Institute of Technology

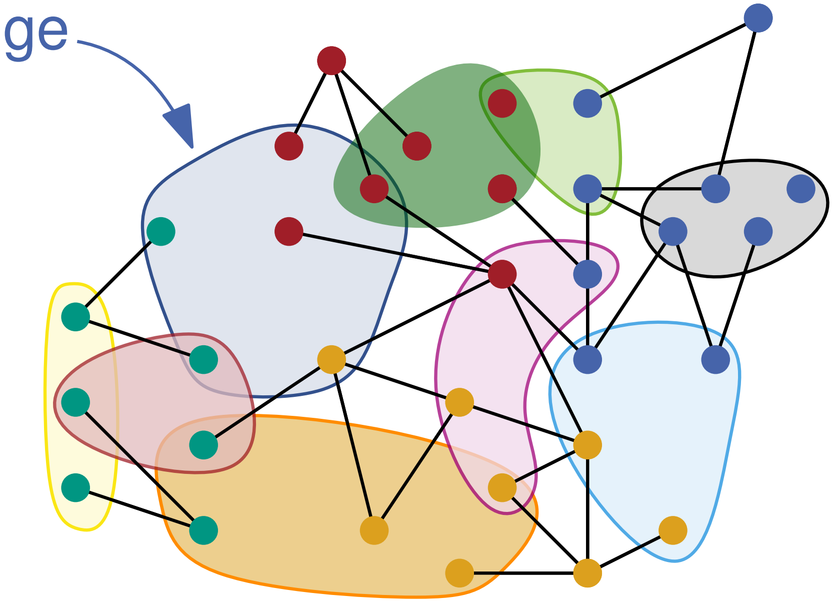
Graph $G = (V, E)$

vertices → V, edges → E



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

hyperedge



Hypergraph $H = (V, E)$

- 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 (HGP)

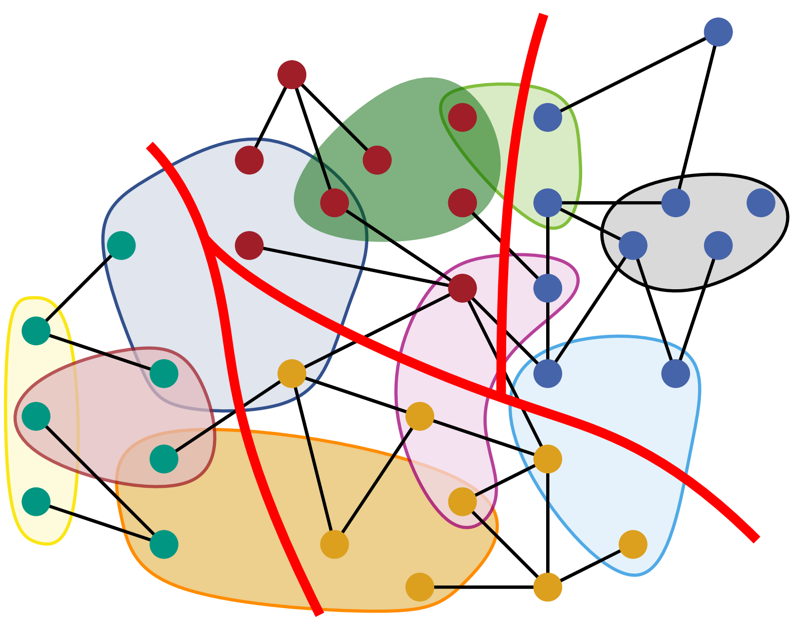
Karlsruhe Institute of Technology

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:

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

- Objective function on hyperedges is minimized



Institute of Theoretical Informatics
Algorithmics Group

ε-Balanced Hypergraph Partitioning (HGP)

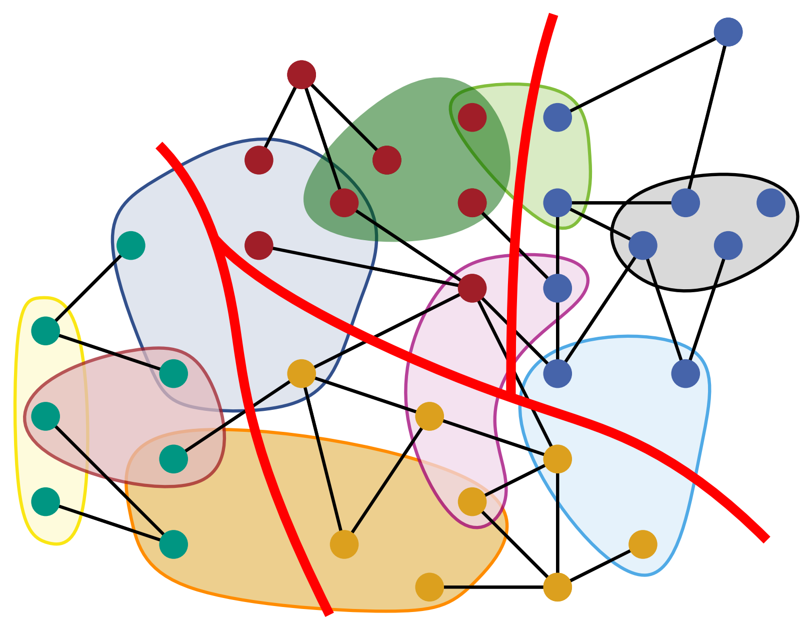
Partition hypergraph $H = (V, E, c : V \to \mathbb{R}_{>0}, \omega : E \to \mathbb{R}_{>0})$ into **k** disjoint blocks $\Pi = \{V_1, \ldots, V_k\}$ such that

- 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 (HGP)

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:

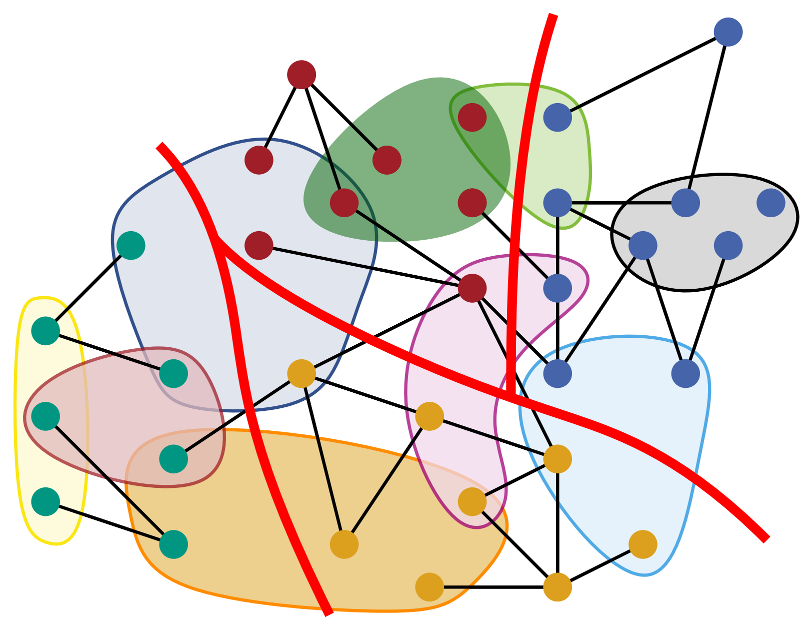
imbalance parameter

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

- Objective function on hyperedges is minimized

Common HGP Objectives:

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



Institute of Theoretical Informatics
Algorithmics Group

ε-Balanced Hypergraph Partitioning (HGP)

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

- Blocks V_i are roughly equal-sized:

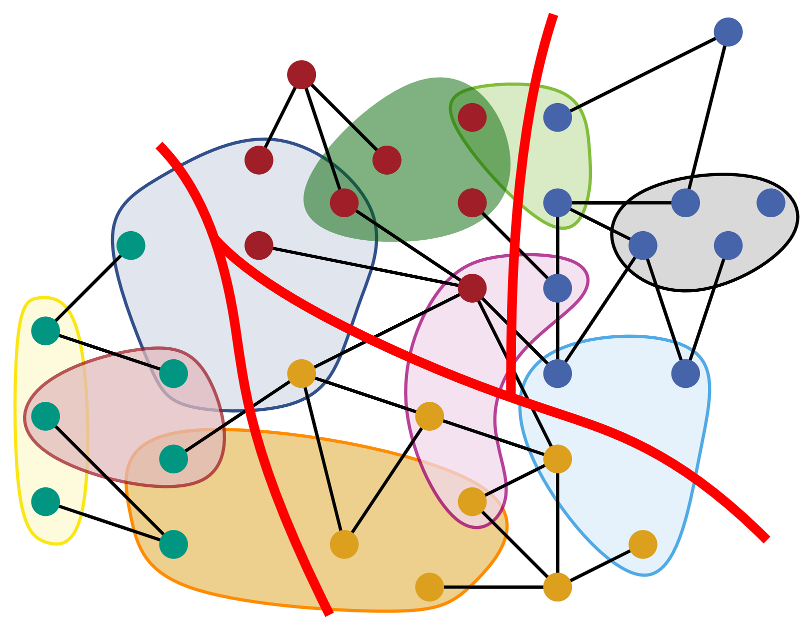
imbalance parameter

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

- Objective function on hyperedges is minimized

Common HGP Objectives:

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



ε-Balanced Hypergraph Partitioning (HGP)

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

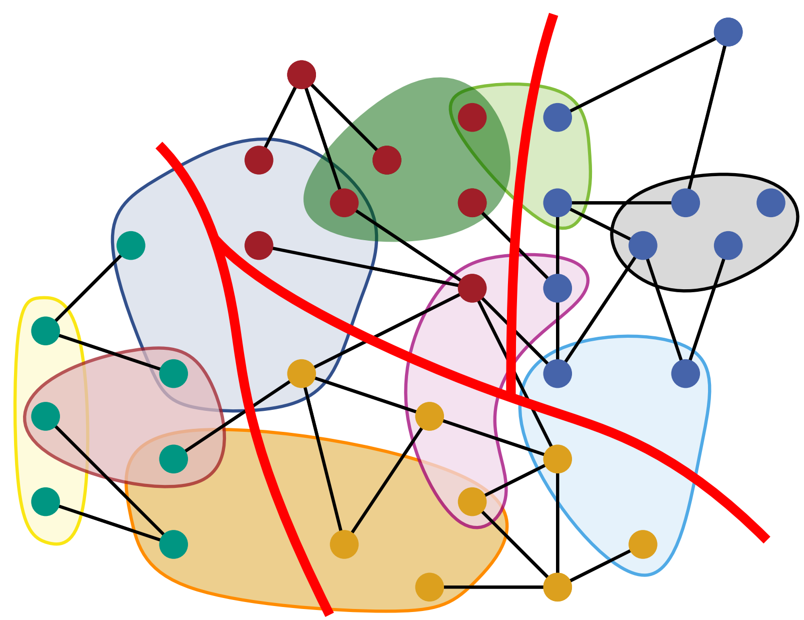
$$c(V_i) \leq (1 + \varepsilon) \left\lceil \frac{c(V)}{k} \right\rceil$$

- Objective function on hyperedges is minimized

Common HGP Objectives:

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

 # blocks connected by *e*



Institute of Theoretical Informatics
Algorithmics Group

ε-Balanced Hypergraph Partitioning (HGP)

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

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

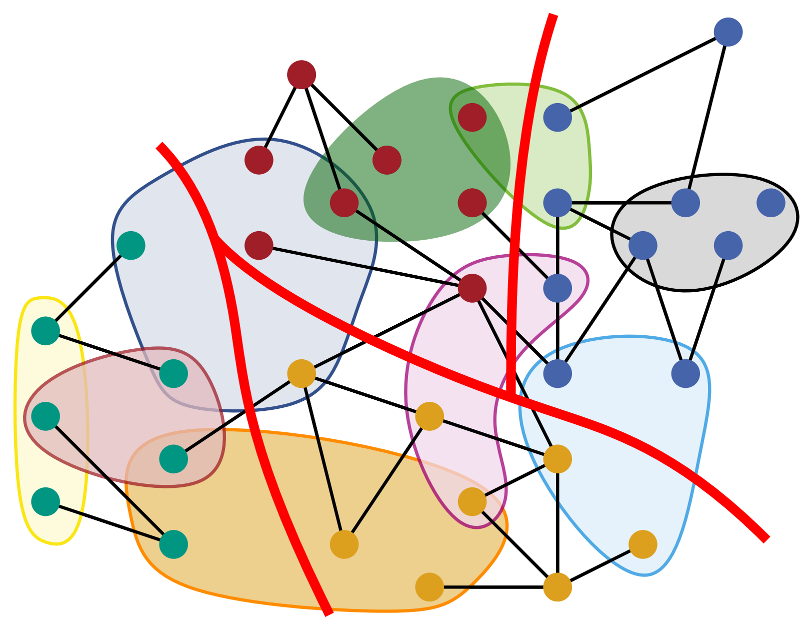
- Objective function on hyperedges is minimized

Common HGP Objectives:

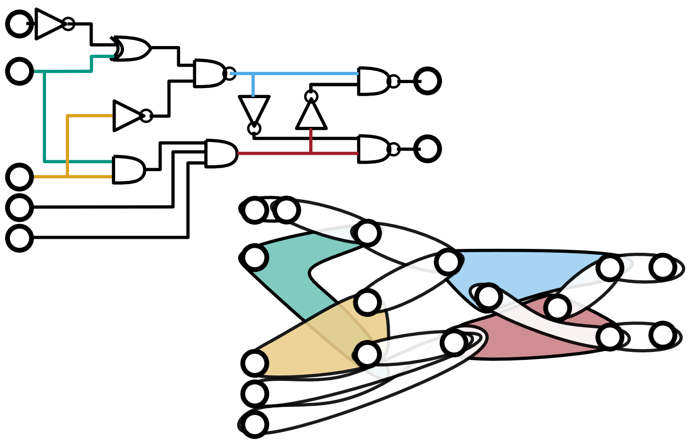
- Cut-Net: $\sum_{e \in \text{Cut}} \omega(e)$
- Connectivity: $\sum_{e \in \text{cut}} (\lambda - 1)\, \omega(e)$

 # blocks connected by *e*

$\Rightarrow$ Both revert to edge-cut for graphs



Applications



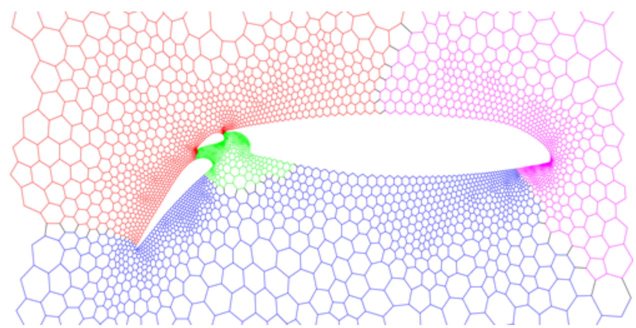
VLSI Design

Warehouse Optimization

[Martin Grandjean, via Wikimedia Commons]

Complex Networks

Route Planning

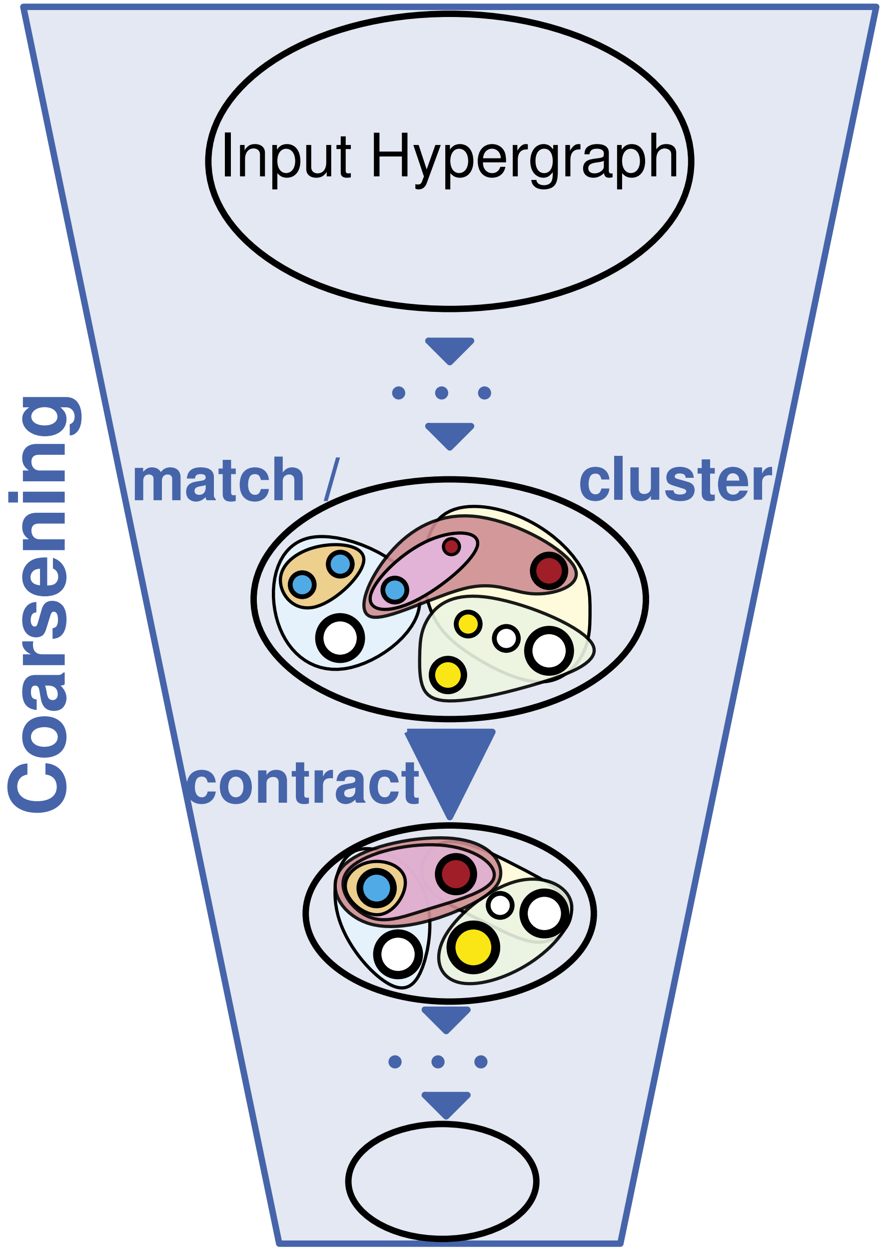


$$\mathbb{R}^{n \times n} \ni Ax = b \in \mathbb{R}^n$$

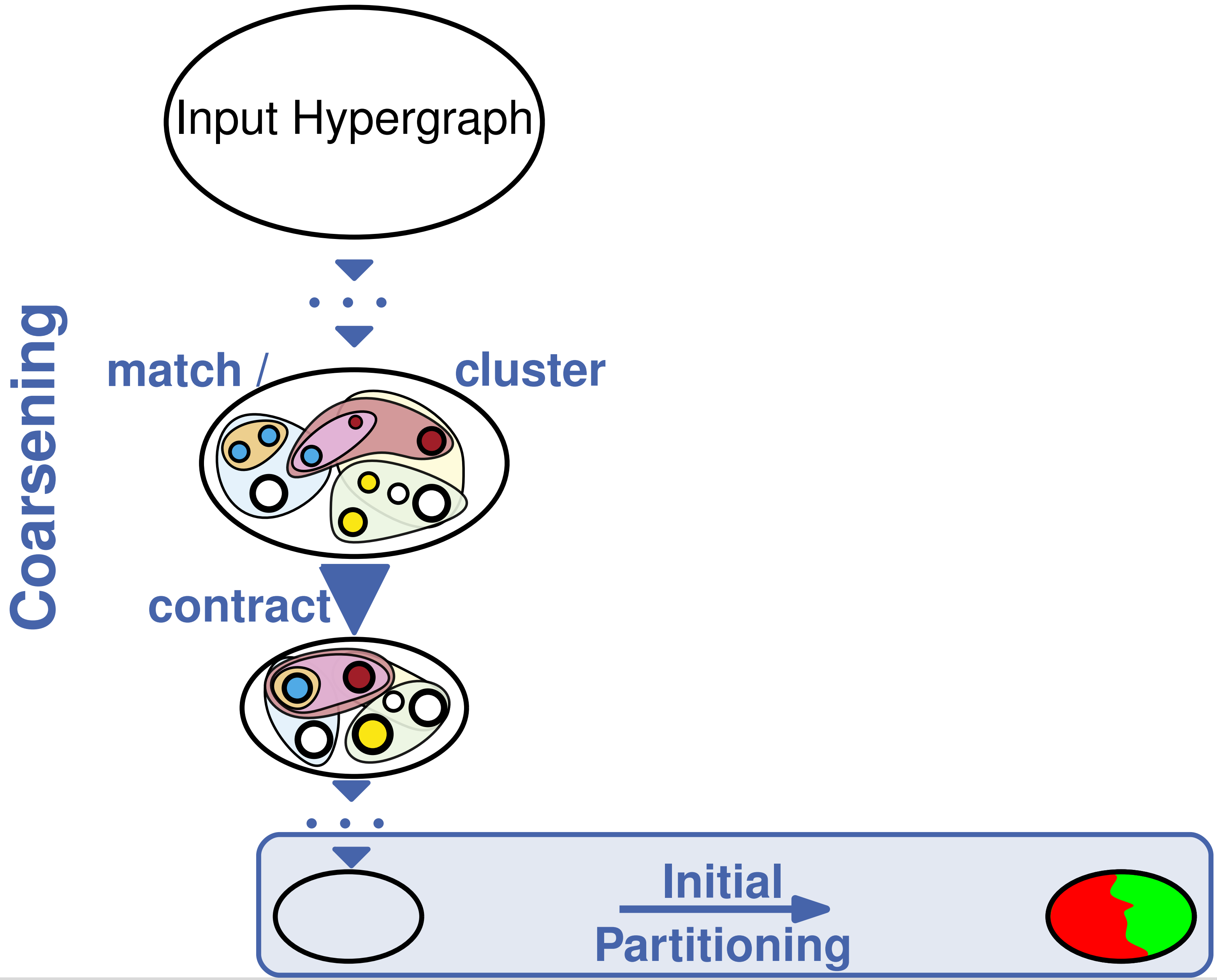
Scientific Computing

High-Quality Hypergraph Partitioning

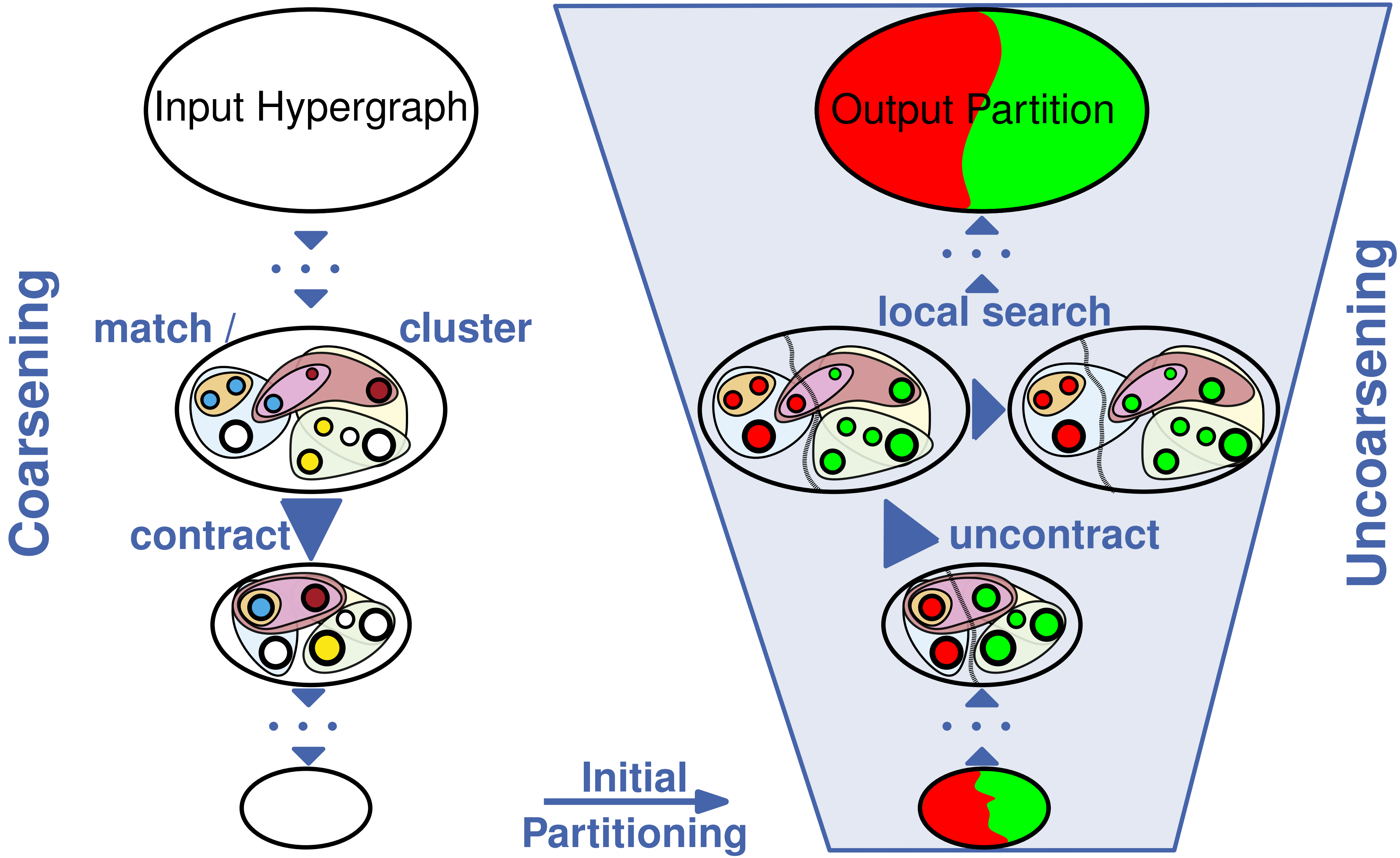
Successful Heuristic: Multilevel Paradigm



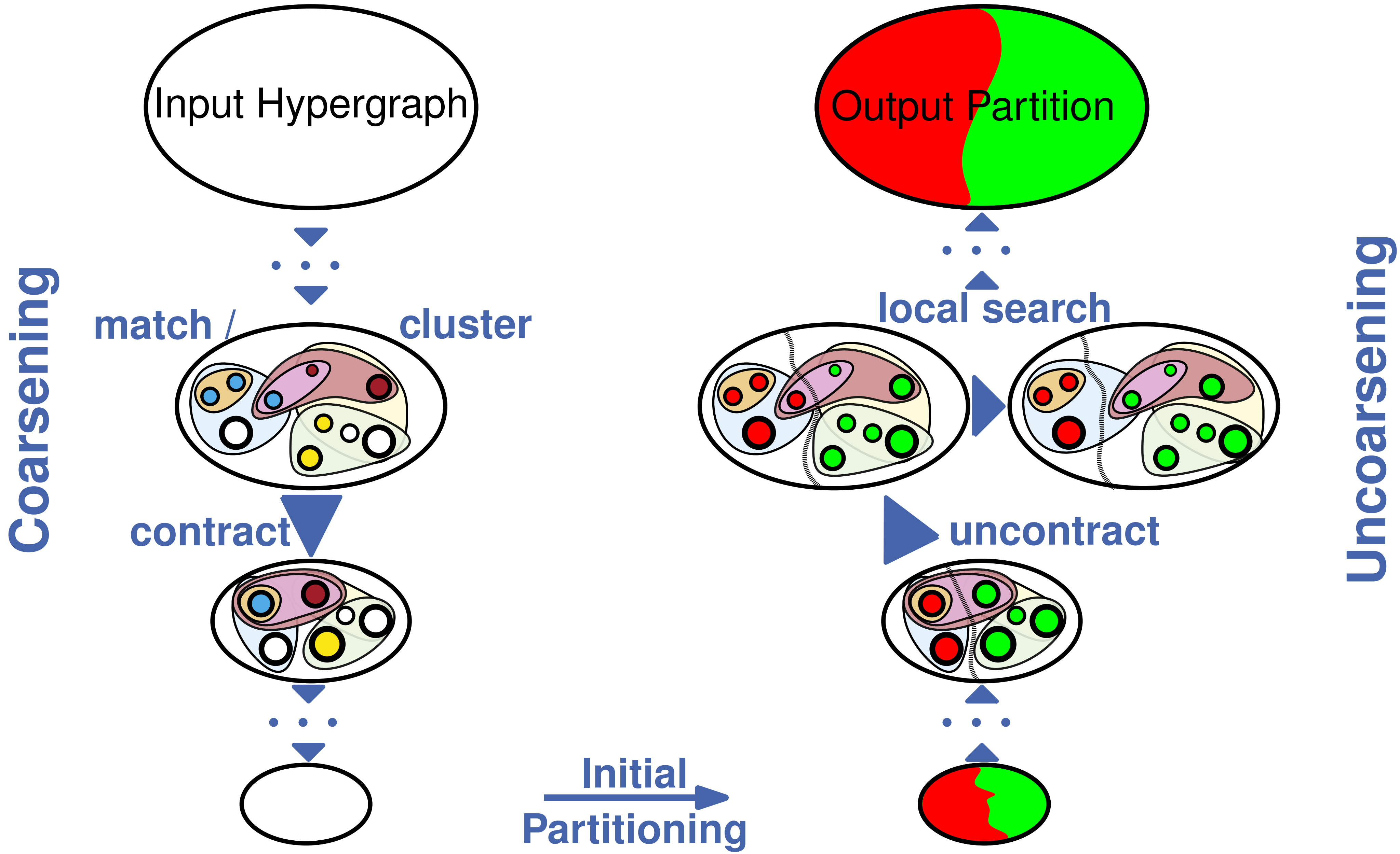
Successful Heuristic: Multilevel Paradigm



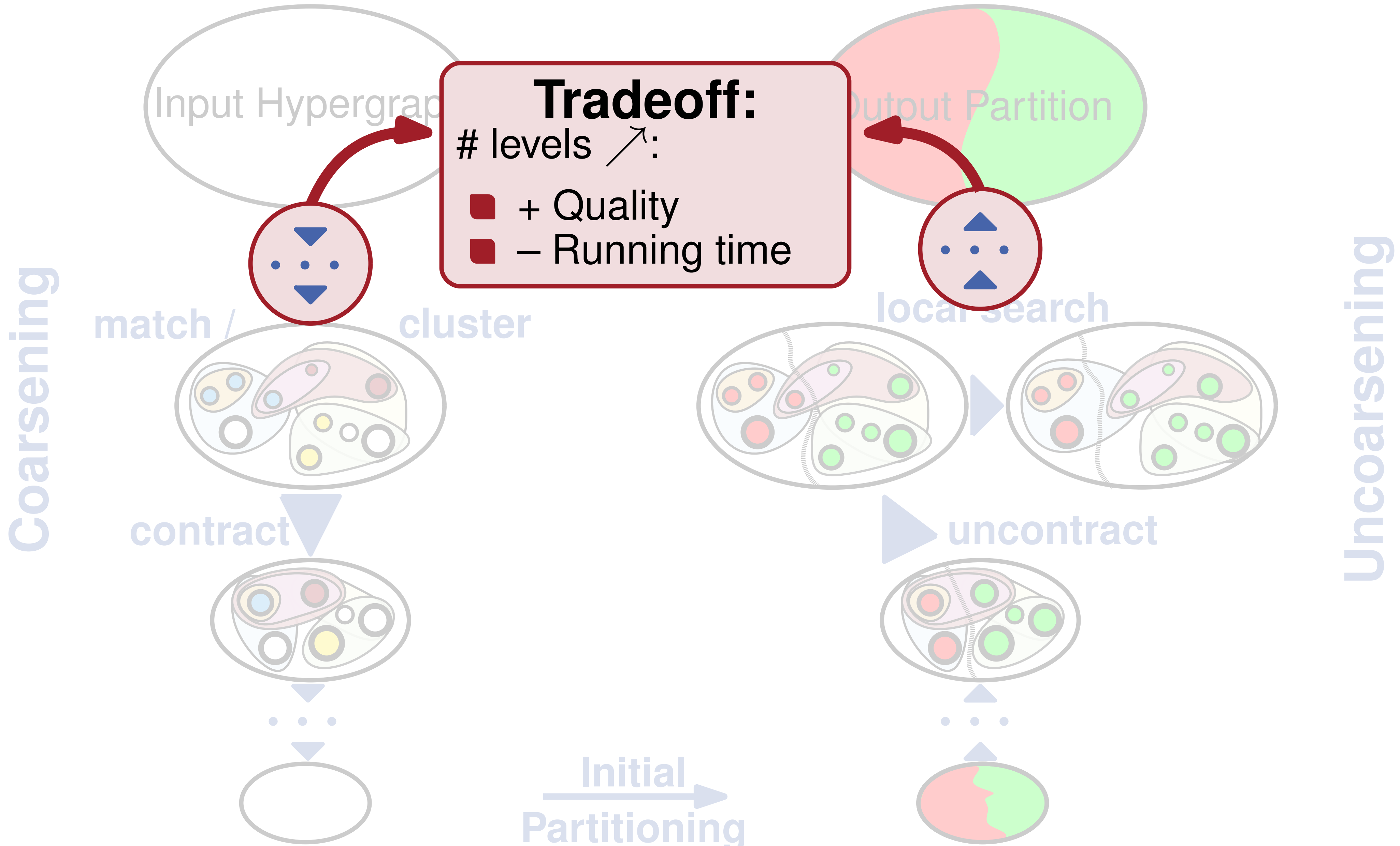
Successful Heuristic: Multilevel Paradigm



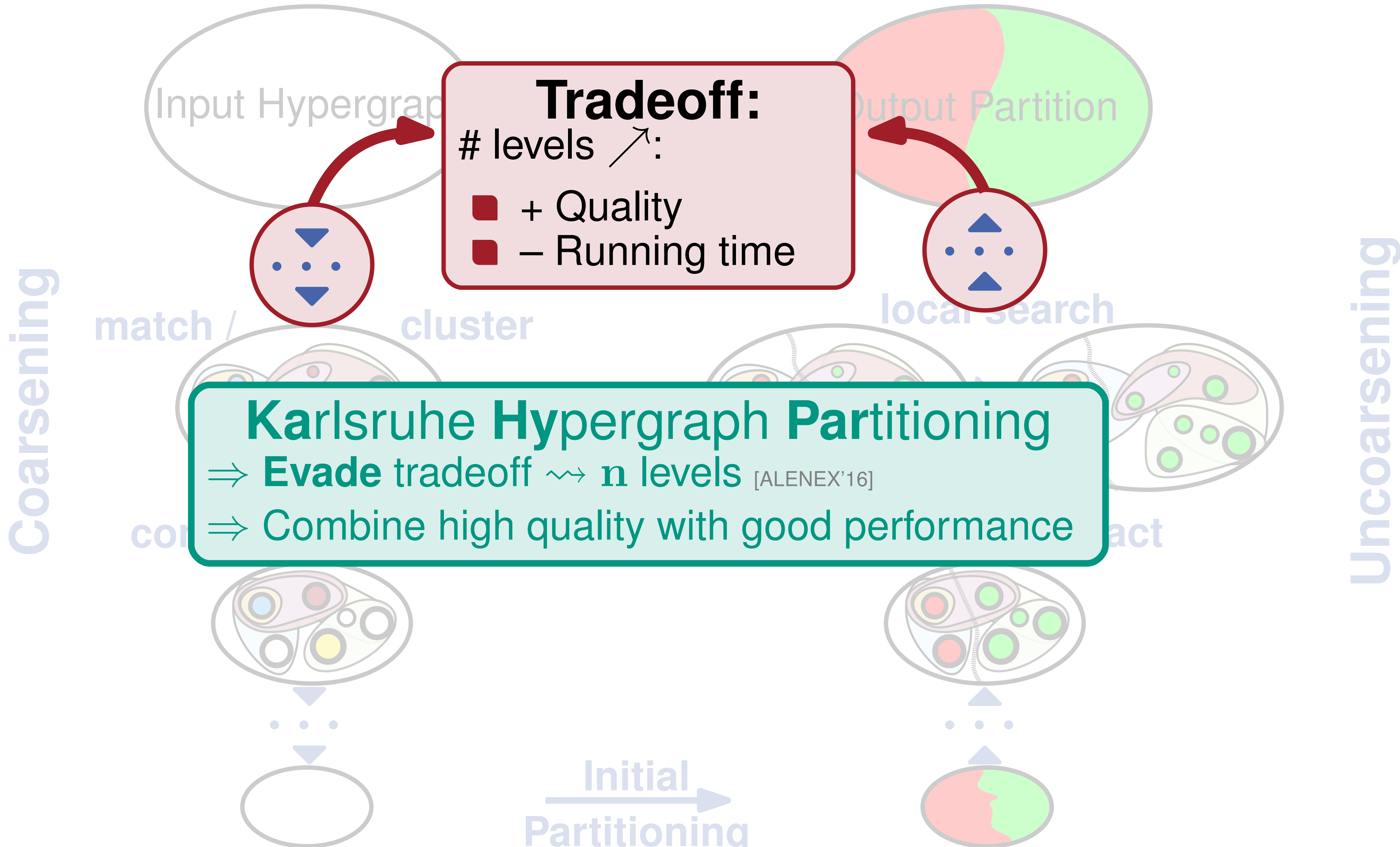
Why Yet Another Multilevel Algorithm?



Why Yet Another Multilevel Algorithm?



Why Yet Another Multilevel Algorithm?



Tradeoff:

\# levels $\nearrow$:

- + Quality
- – Running time

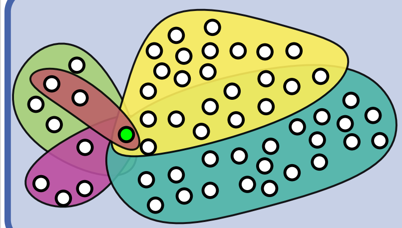
Karlsruhe **Hy**pergraph **Par**titioning

$\Rightarrow$ **Evade** tradeoff $\rightsquigarrow$ $\mathbf{n}$ levels [ALENEX'16]

$\Rightarrow$ Combine high quality with good performance

KaHyPar: Novel Algorithmic Ingredients

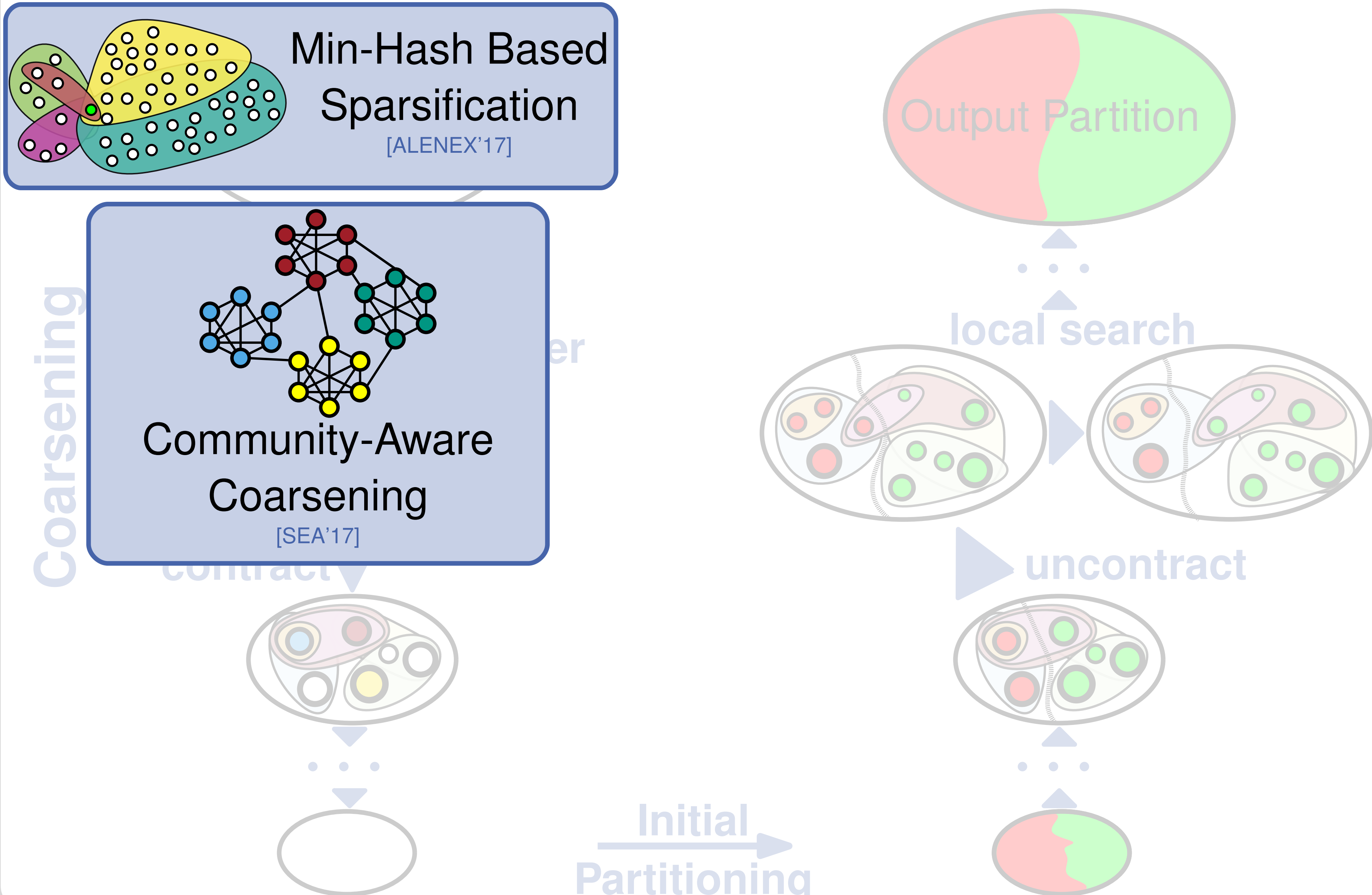
KaHyPar: Novel Algorithmic Ingredients



Min-Hash Based Sparsification

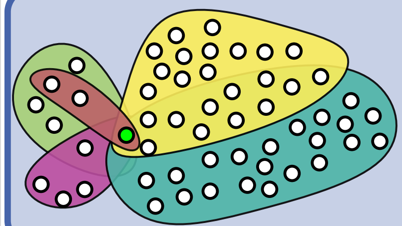
[ALENEX'17]

KaHyPar: Novel Algorithmic Ingredients



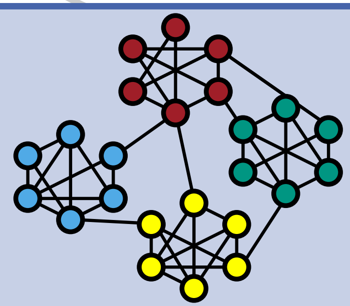
Institute of Theoretical Informatics
Algorithmics Group

KaHyPar: Novel Algorithmic Ingredients



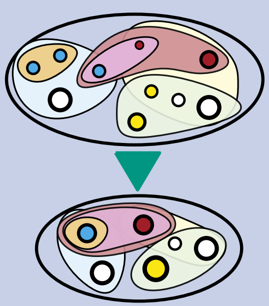
Min-Hash Based Sparsification

[ALENEX'17]



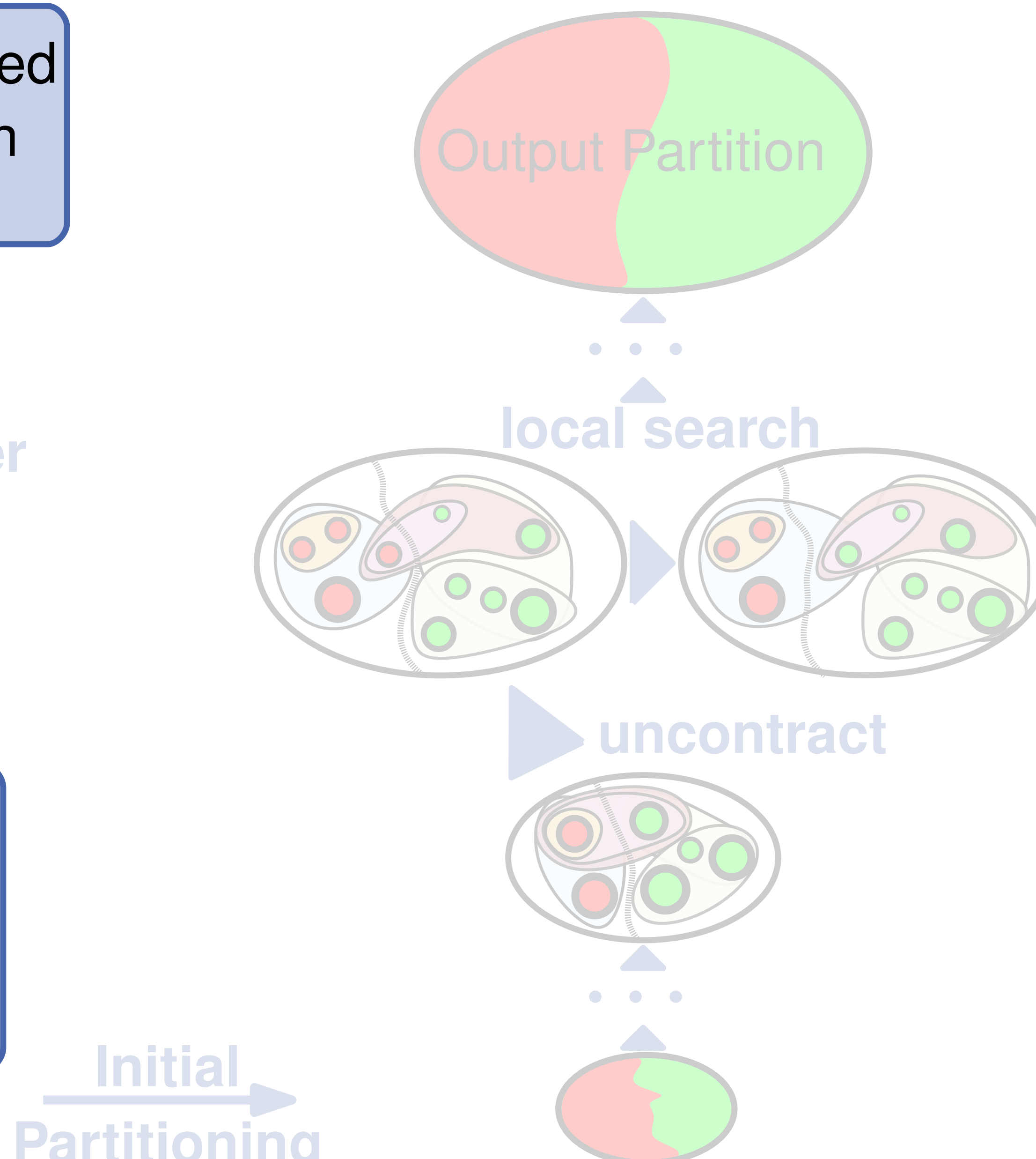
Community-Aware Coarsening

[SEA'17]

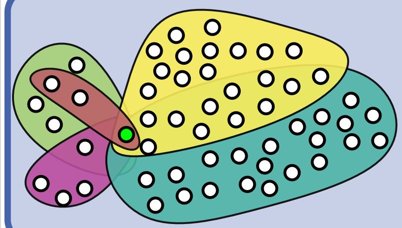


Fast *n*-Level Coarsening

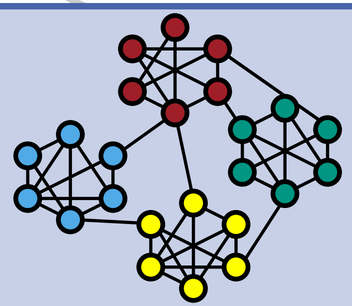
[ALENEX'16, ALENEX'17]



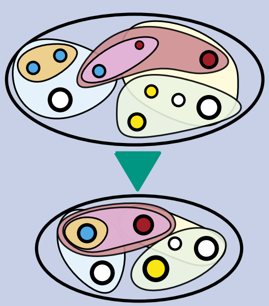
KaHyPar: Novel Algorithmic Ingredients



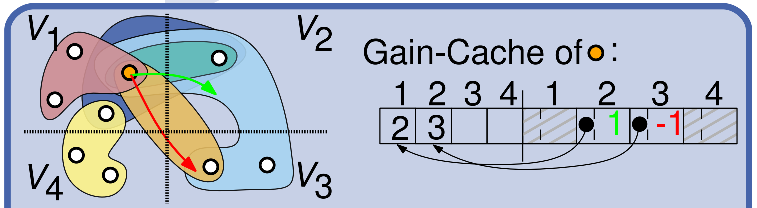
Min-Hash Based Sparsification
[ALENEX'17]



Community-Aware Coarsening
[SEA'17]



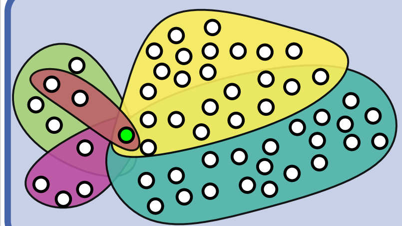
Fast *n*-Level Coarsening
[ALENEX'16, ALENEX'17]



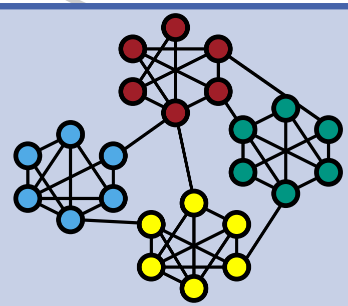
Engineered *k*-way FM
[ALENEX'17]

Institute of Theoretical Informatics
Algorithmics Group

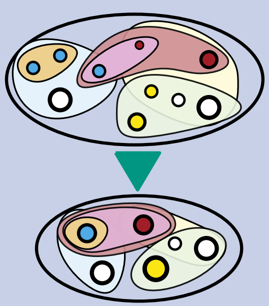
KaHyPar: Novel Algorithmic Ingredients



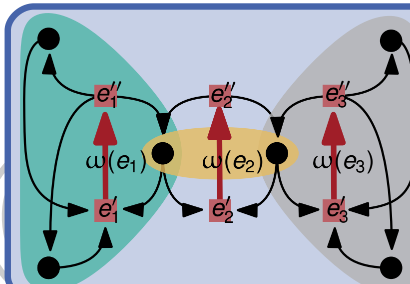
Min-Hash Based Sparsification
[ALENEX'17]



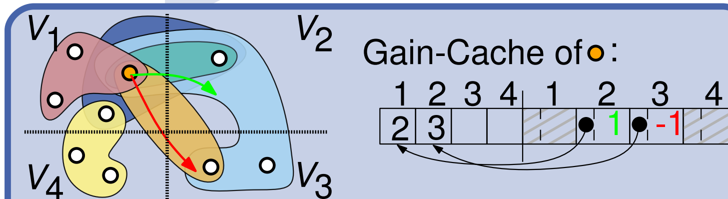
Community-Aware Coarsening
[SEA'17]



Fast n-Level Coarsening
[ALENEX'16, ALENEX'17]

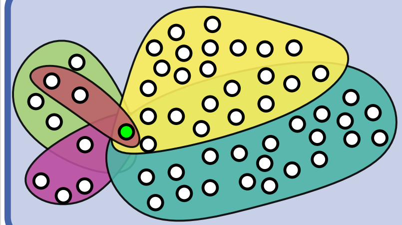


Max-Flow Min-Cut Refinement
[SEA'18, JEA'19]

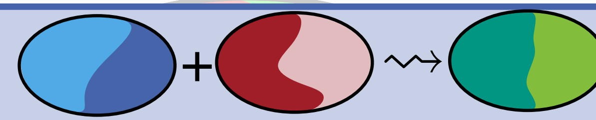


Engineered k-way FM
[ALENEX'17]

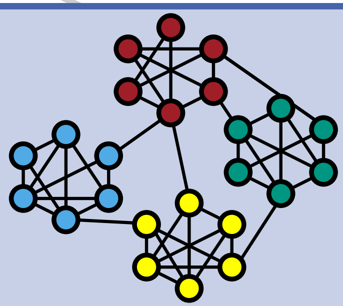
KaHyPar: Novel Algorithmic Ingredients



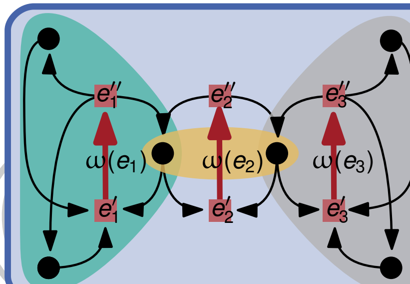
Min-Hash Based Sparsification
[ALENEX'17]



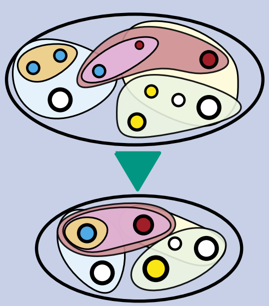
Memetic Multilevel Algorithm
[GECCO'18]



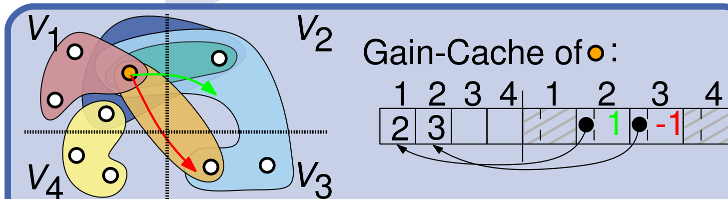
Community-Aware Coarsening
[SEA'17]



Max-Flow Min-Cut Refinement
[SEA'18, JEA'19]



Fast *n*-Level Coarsening
[ALENEX'16, ALENEX'17]



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

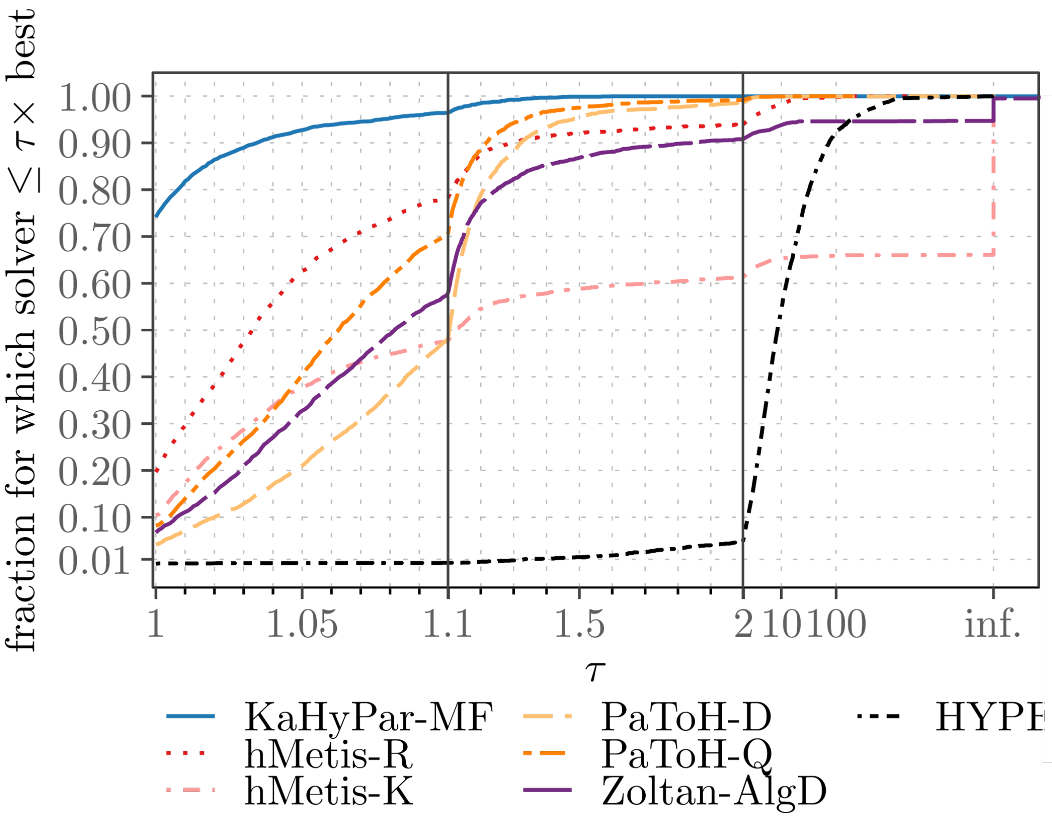
Institute of Theoretical Informatics
Algorithmics Group

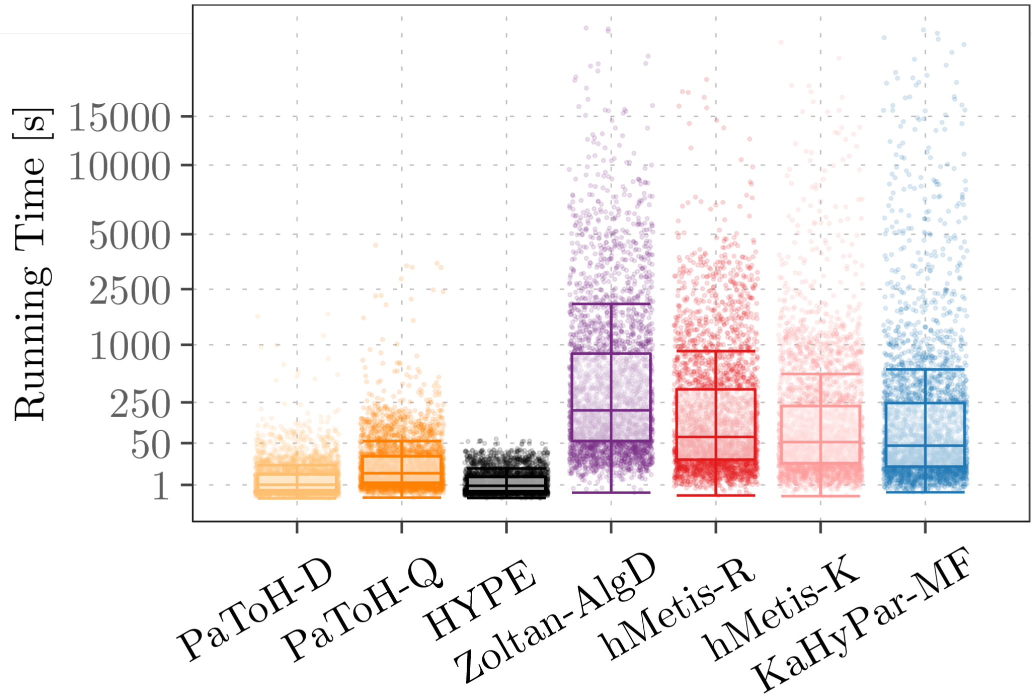
Experiments – Benchmark Setup

- System: 1 core of 2 Intel Xeon E5-2670 @ 2.6 Ghz, 64 GB RAM
- # Hypergraphs: [publicly available]
  - SuiteSparse Matrix Collection 184
  - SAT Competition 2014 (3 representations) 92·3
  - ISPD98 & DAC2012 VLSI Circuits 28
- $k \in \{2, 4, 8, 16, 32, 64, 128\}$ with imbalance: $\varepsilon = 3\%$
- Comparing KaHyPar with:
  - hMetis-R & hMetis-K
  - PaToH-Default & PaToH-Quality
  - HYPE
  - Zoltan-AlgD

Institute of Theoretical Informatics
Algorithmics Group

Experiments: Connectivity Optimization





⇒ Similar results for cut-net optimization

Parallel Shared-Memory Graph Partitioning

Parallel GP: Coarsening [EuroPar'18]

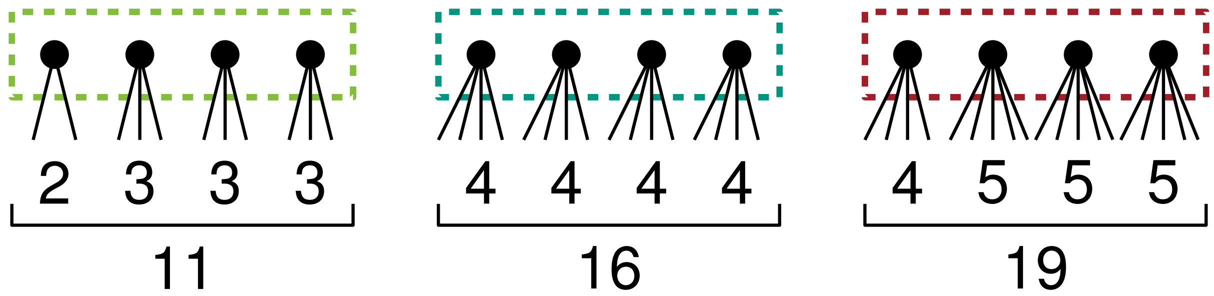
Algorithm: Parallel label propagation [SM'16] with improved load balancing

Institute of Theoretical Informatics
Algorithmics Group

Parallel GP: Coarsening [EuroPar'18]

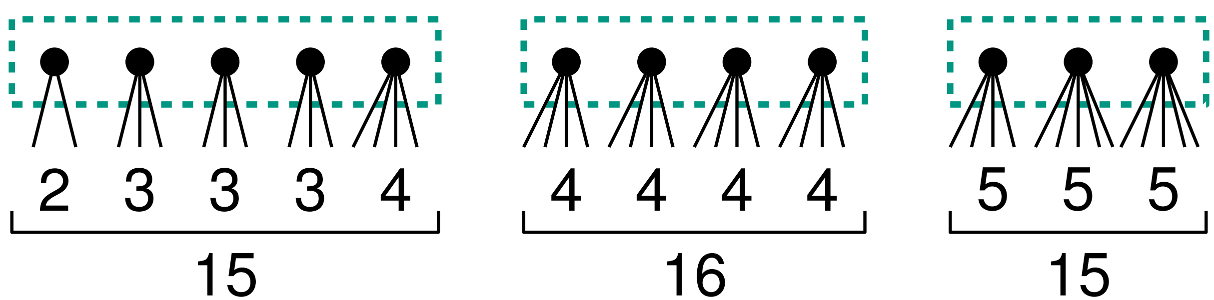
Algorithm: Parallel label propagation [SM'16] with improved load balancing

Problem: Vertex-based distribution ⇒ bad load-balance



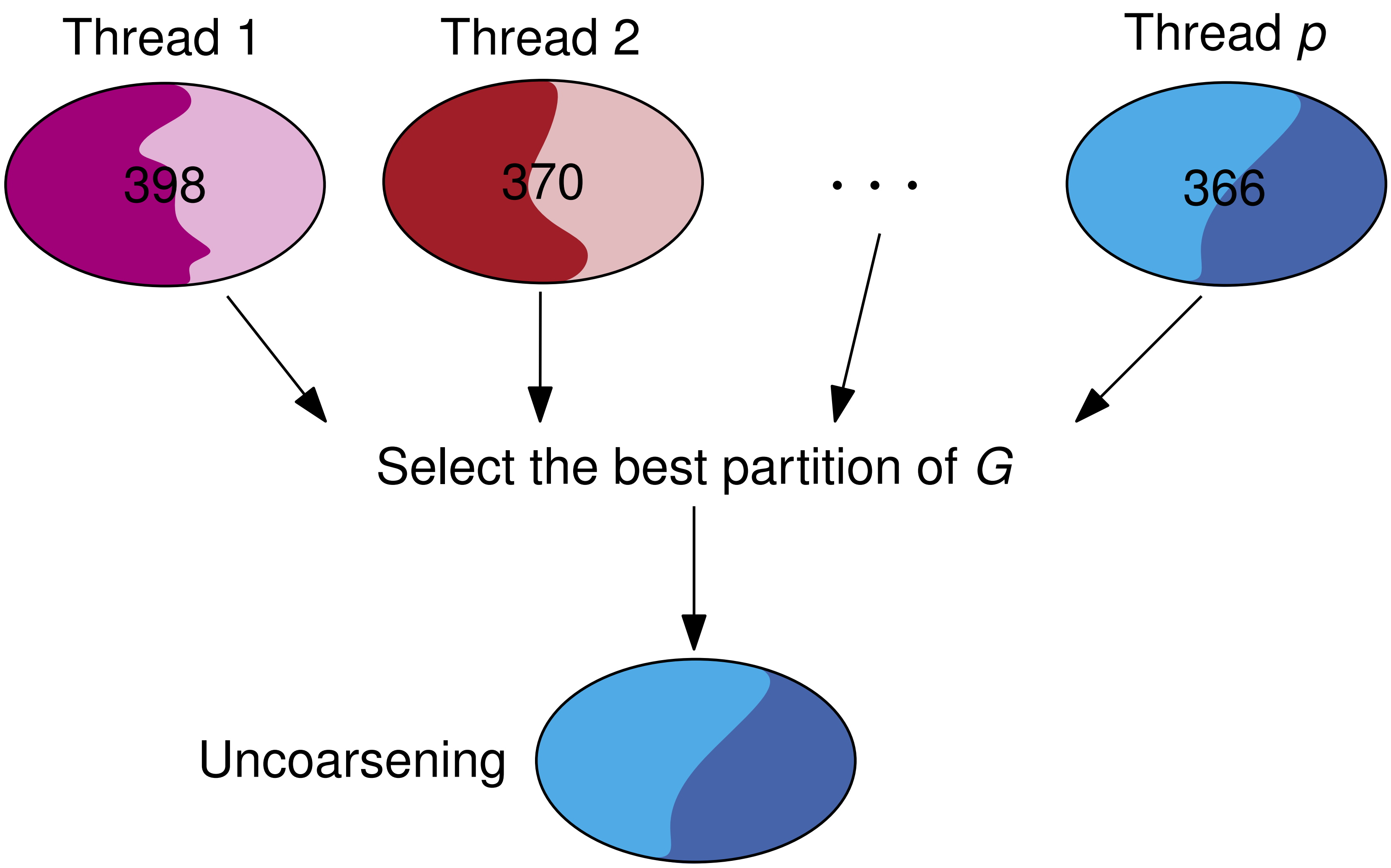
Solution: Edge-based distribution

- Packets P: $\sqrt{|E|} \leq \sum_{v \in P} d(v) \leq \sqrt{|E|} + \Delta$, $\Delta = \max_{v \in V} d(v)$



Parallel GP: Initial Partitioning [EuroPar'18]

Parallel Initial Partitioning using KaHIP [SEA'14]

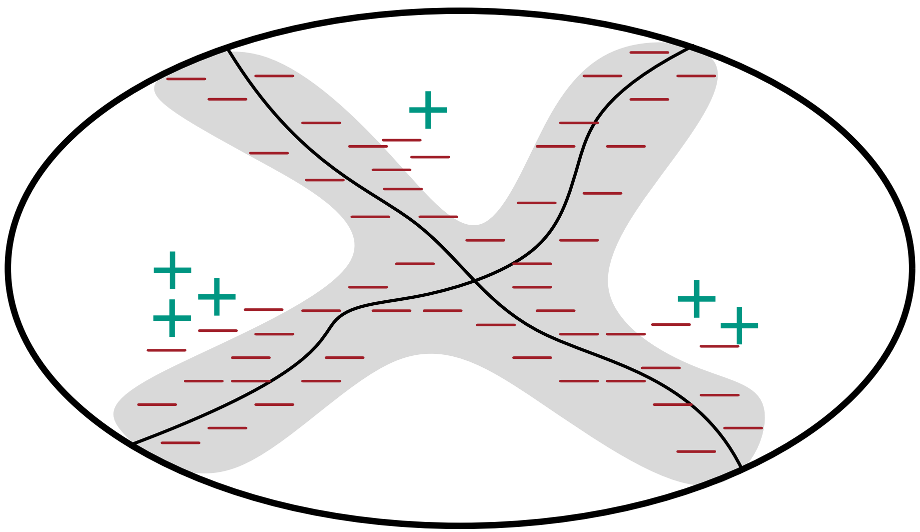


Parallel GP: Refinement [EuroPar'18]

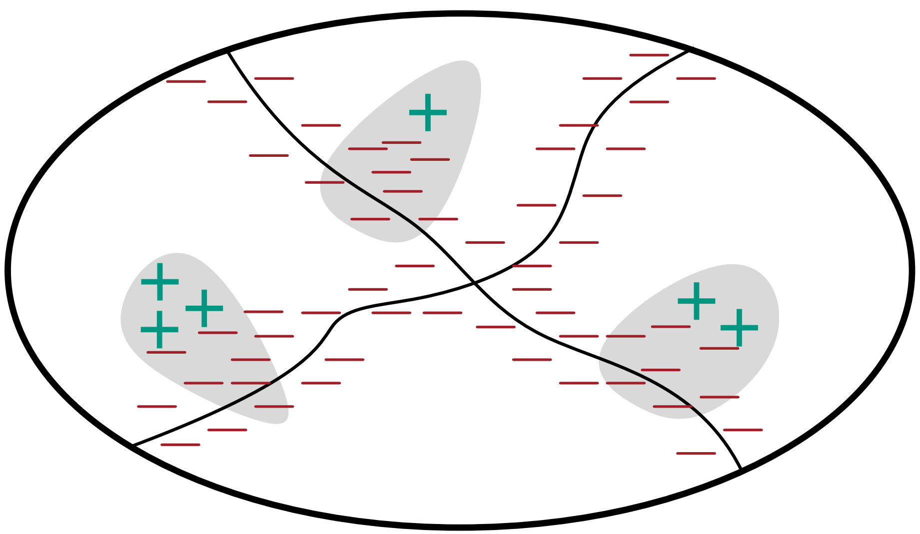
Algorithms:

- Parallel label propagation
- Parallel localized *k*-way local search
  - minimal coordination of searches
  - serialized execution of final moves

Traditional:

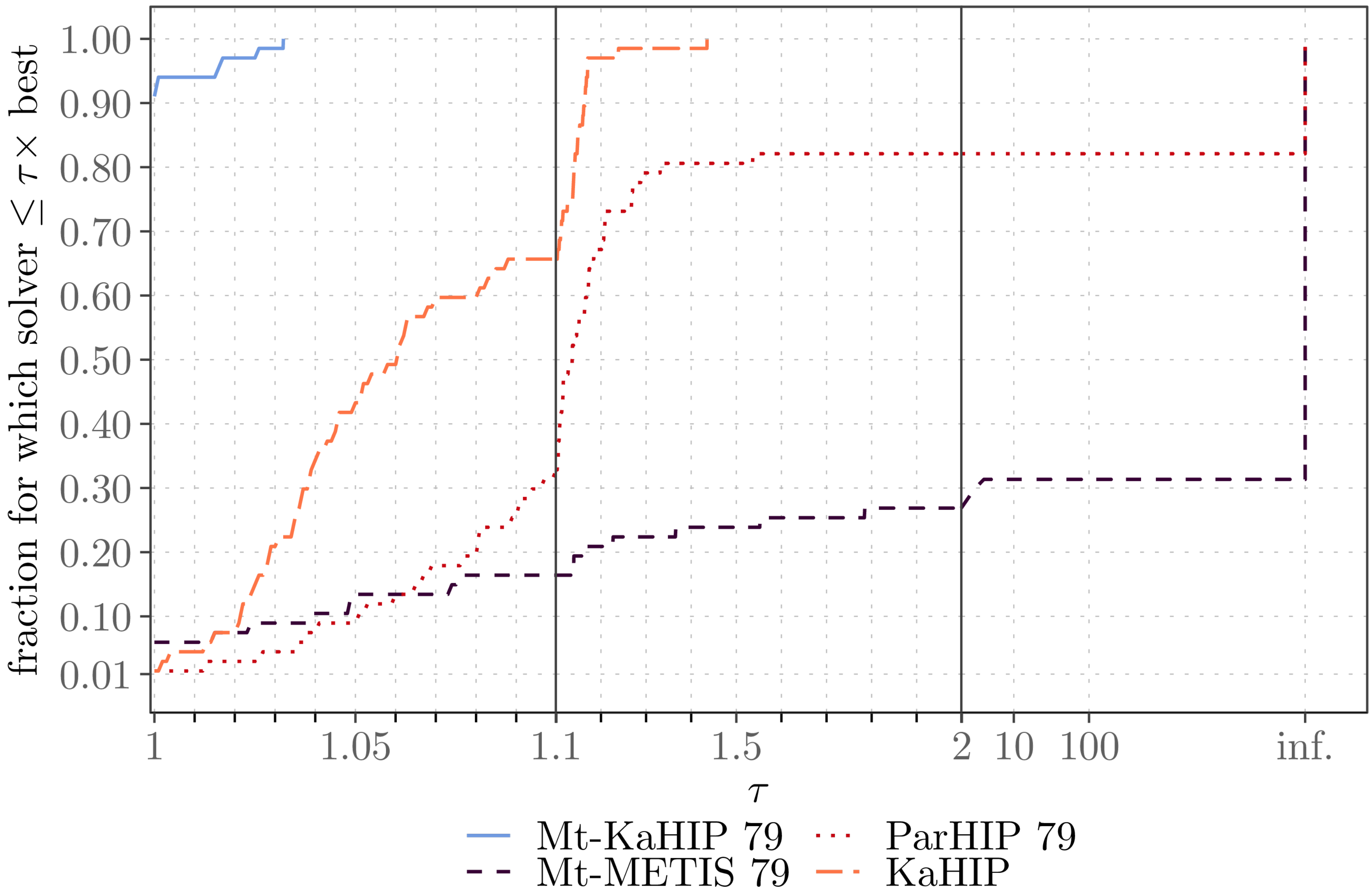


Localized:



Experiments: Solution Quality

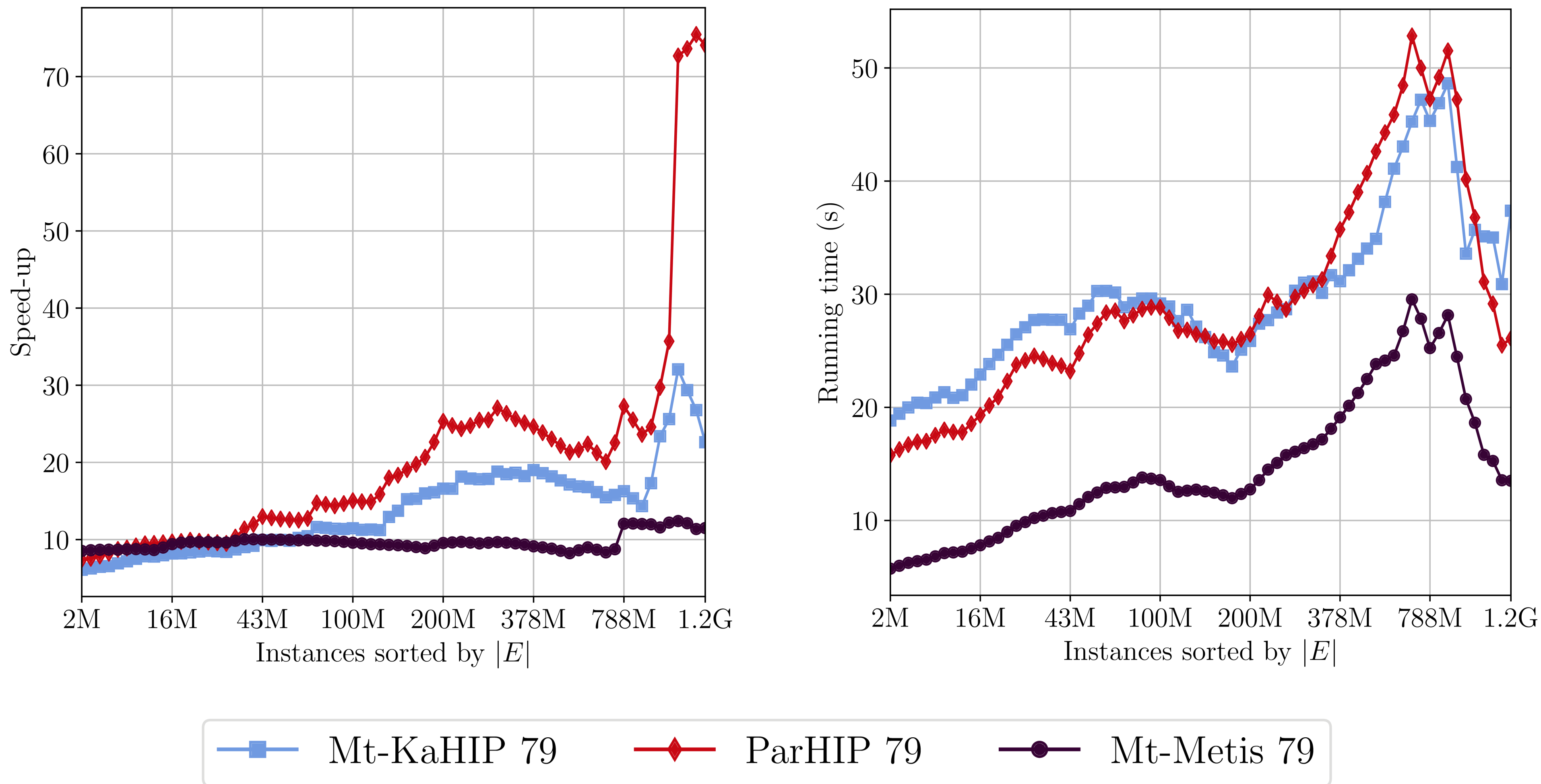
38 Graphs with $k = \{16, 64\}$



Institute of Theoretical Informatics
Algorithmics Group

Experiments: Speedup & Running Time

4 Socket Machine with 79 Threads



Cumulative: $(x, y) \rightarrow$ speedup/ running time for graphs with $|E| \geq x = y$

Institute of Theoretical Informatics
Algorithmics Group

Scalable Edge Partitioning

The Edge Partitioning Problem

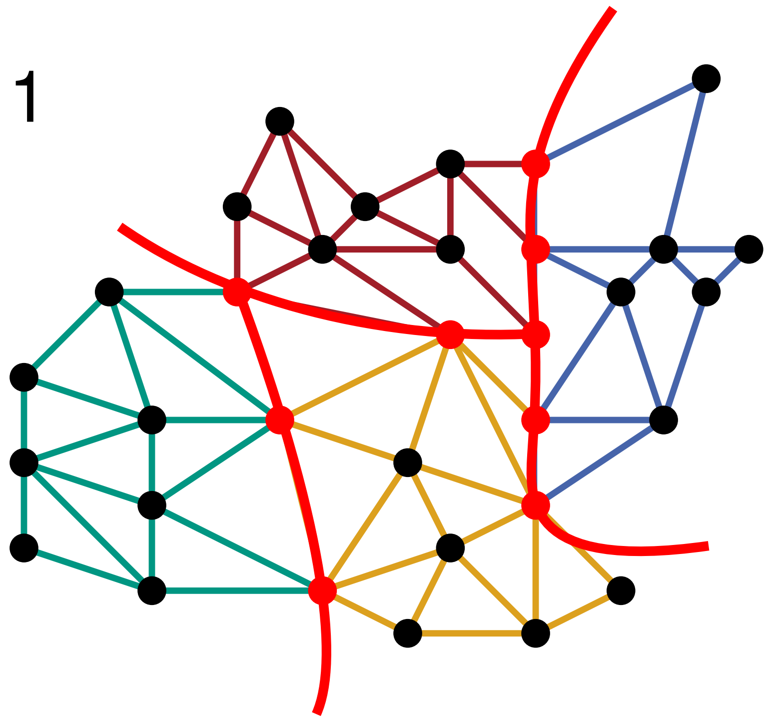
Partition edge set of graph $G = (V, E, c, \omega)$ into **k** disjoint blocks $\Pi = \{E_1, \dots, E_k\}$ such that

- Blocks E_i are roughly equal-sized:

$$\omega(E_i) \leq (1 + \varepsilon) \left\lceil \frac{\omega(E)}{k} \right\rceil$$

- minimize vertex cut:

$$\sum_{v \in V} |I(v)| - 1$$



The Edge Partitioning Problem

Partition edge set of graph $G = (V, E, c, \omega)$ into **k** disjoint blocks $\Pi = \{E_1, \ldots, E_k\}$ such that

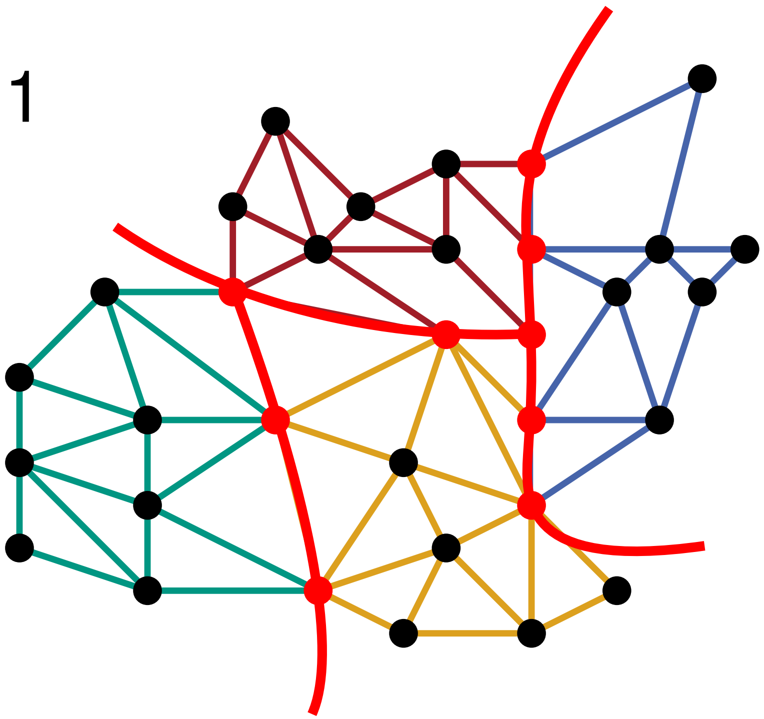
- Blocks E_i are roughly equal-sized:

$$\omega(E_i) \leq (1 + \varepsilon) \left\lceil \frac{\omega(E)}{k} \right\rceil$$

- minimize vertex cut:

$$\sum_{v \in V} |I(v)| - 1$$

 # blocks with edges incident to v



The Edge Partitioning Problem

Partition edge set of graph $G = (V, E, c, \omega)$ into **k** disjoint blocks $\Pi = \{E_1, \ldots, E_k\}$ such that

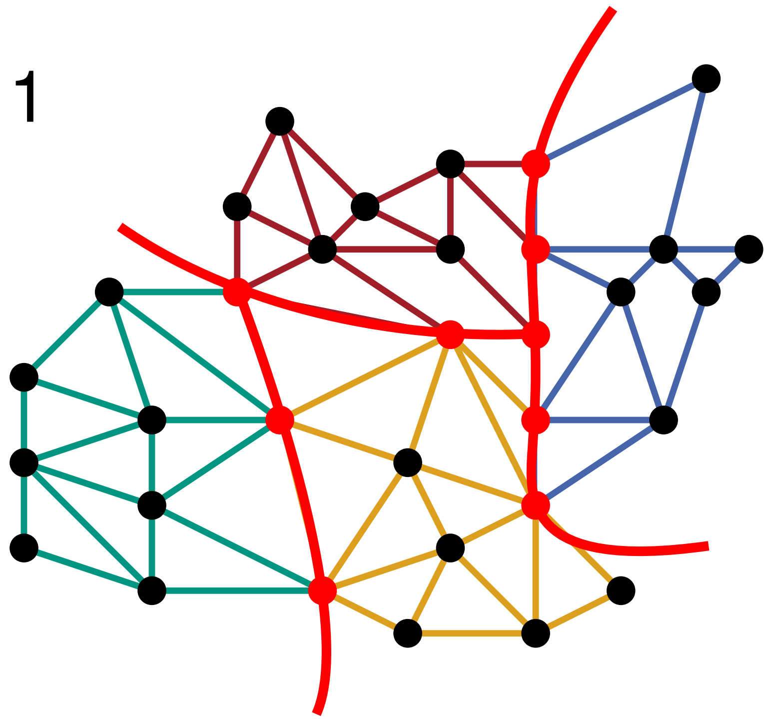
- Blocks E_i are roughly equal-sized:

$$\omega(E_i) \leq (1 + \varepsilon) \left\lceil \frac{\omega(E)}{k} \right\rceil$$

- minimize vertex cut:

$$\sum_{v \in V} |\mathrm{I}(v)| - 1$$

 # blocks with edges incident to v



Motivation [Gonzalez et al.'12]:

- edge-centric distributed computations
- combat shortcomings of TLAV approaches
- duplicate node-centric computations

The Edge Partitioning Problem

Partition edge set of graph $G = (V, E, c, \omega)$ into **k** disjoint blocks $\Pi = \{E_1, \ldots, E_k\}$ such that

- Blocks E_i are roughly equal-sized:

$$\omega(E_i) \leq (1 + \varepsilon) \left\lceil \frac{\omega(E)}{k} \right\rceil$$

- minimize vertex cut:

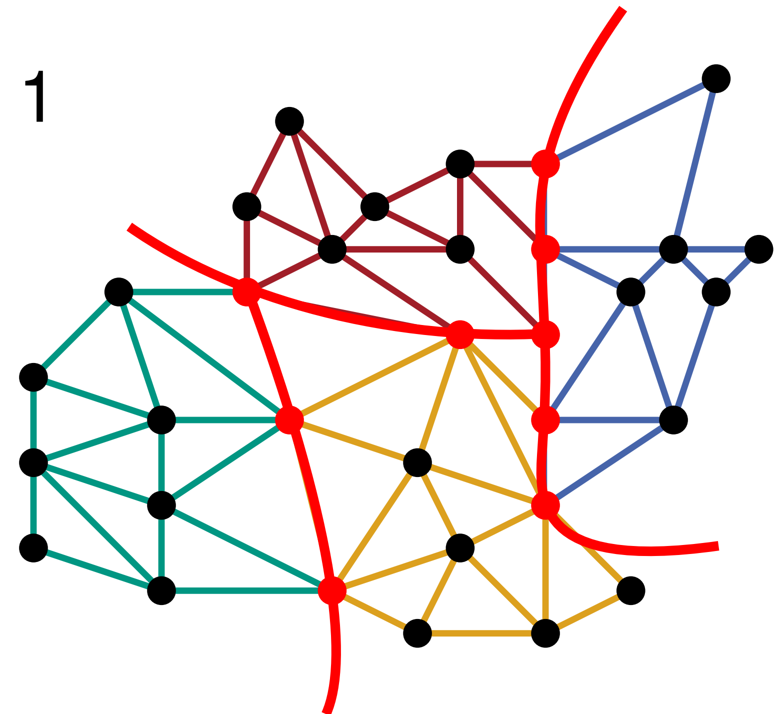
$$\sum_{v \in V} |I(v)| - 1$$

 # blocks with edges incident to v

Motivation [Gonzalez et al.'12]:

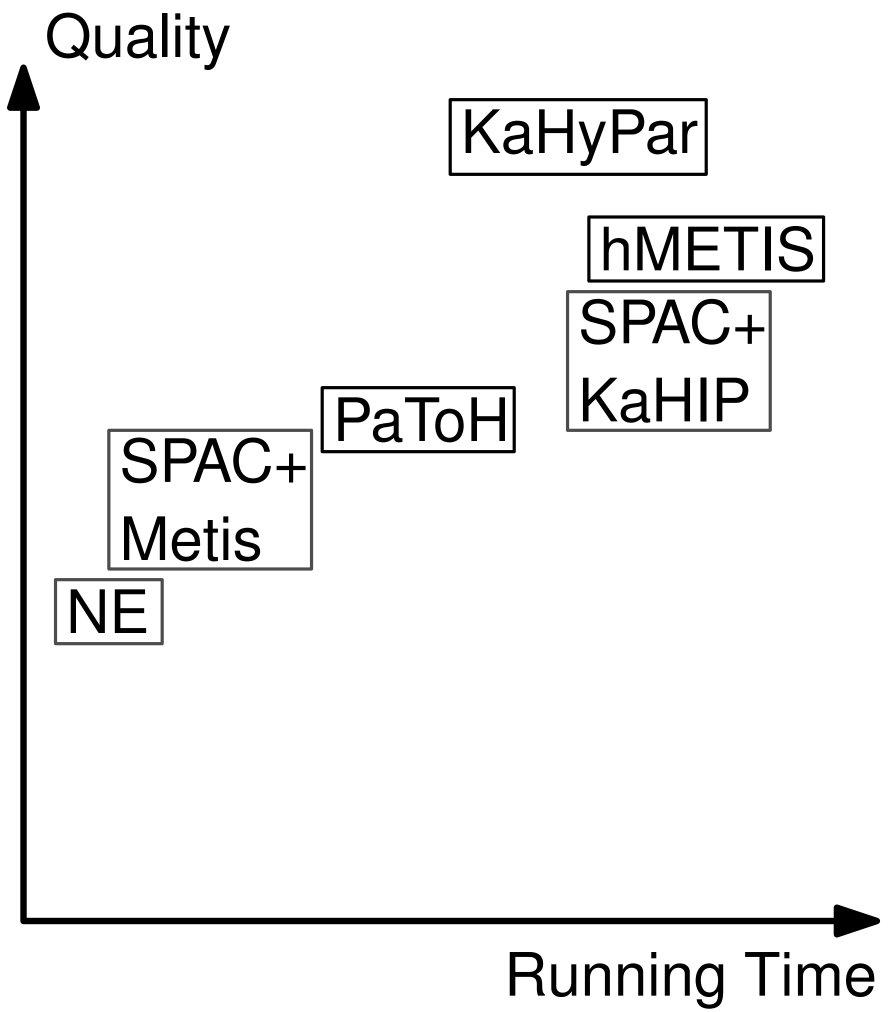
- edge-centric distributed computations
- combat shortcomings of TLAV approaches
- duplicate node-centric computations

Think-Like-A-Vertex



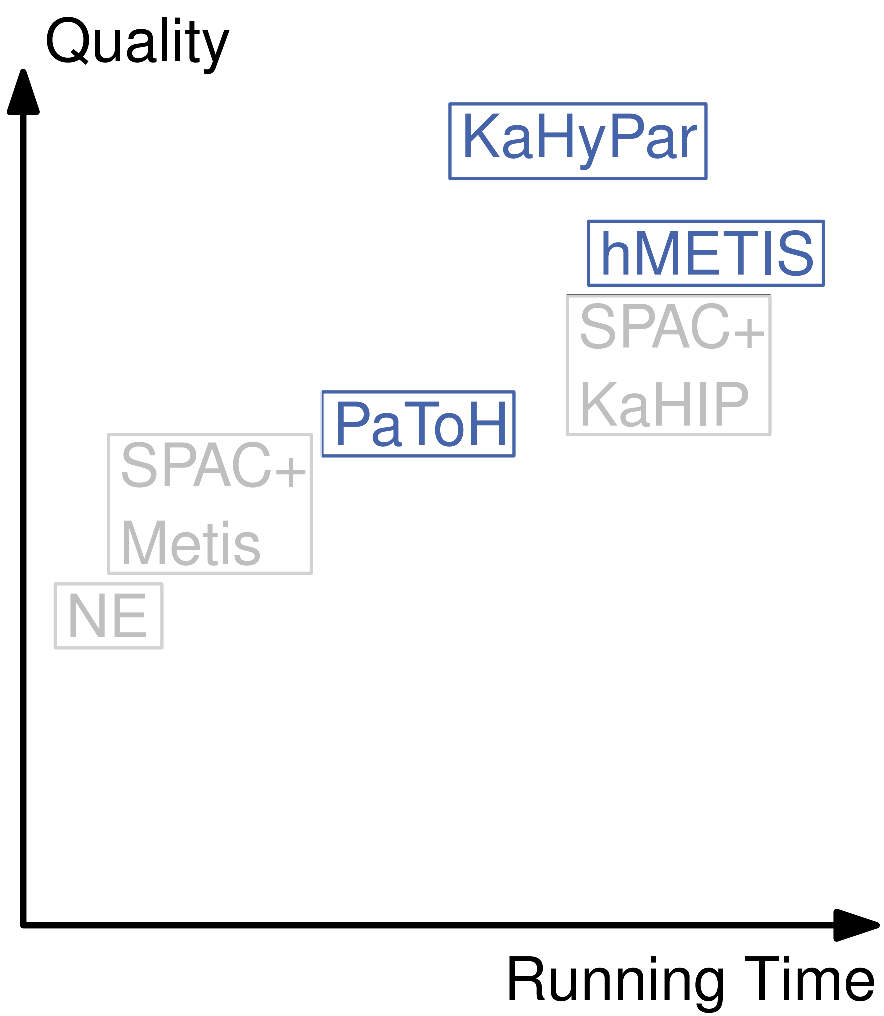
Edge Partitioning Algorithms

Sequential



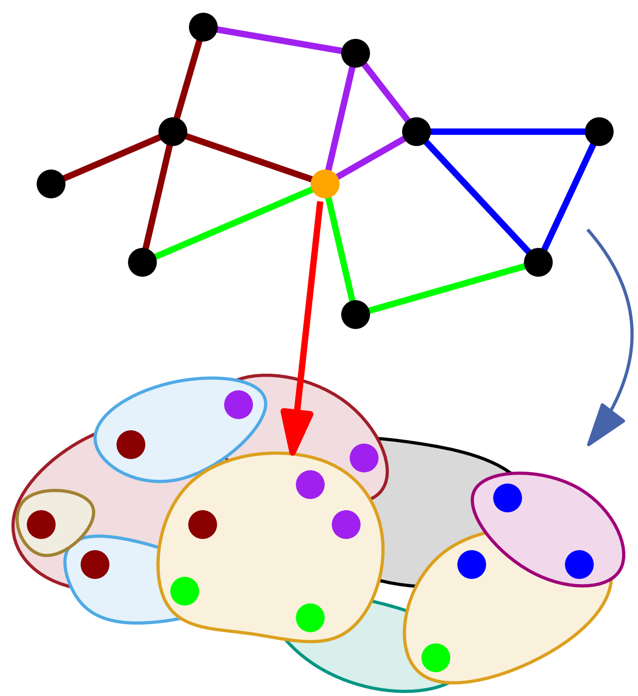
Edge Partitioning Algorithms

Sequential



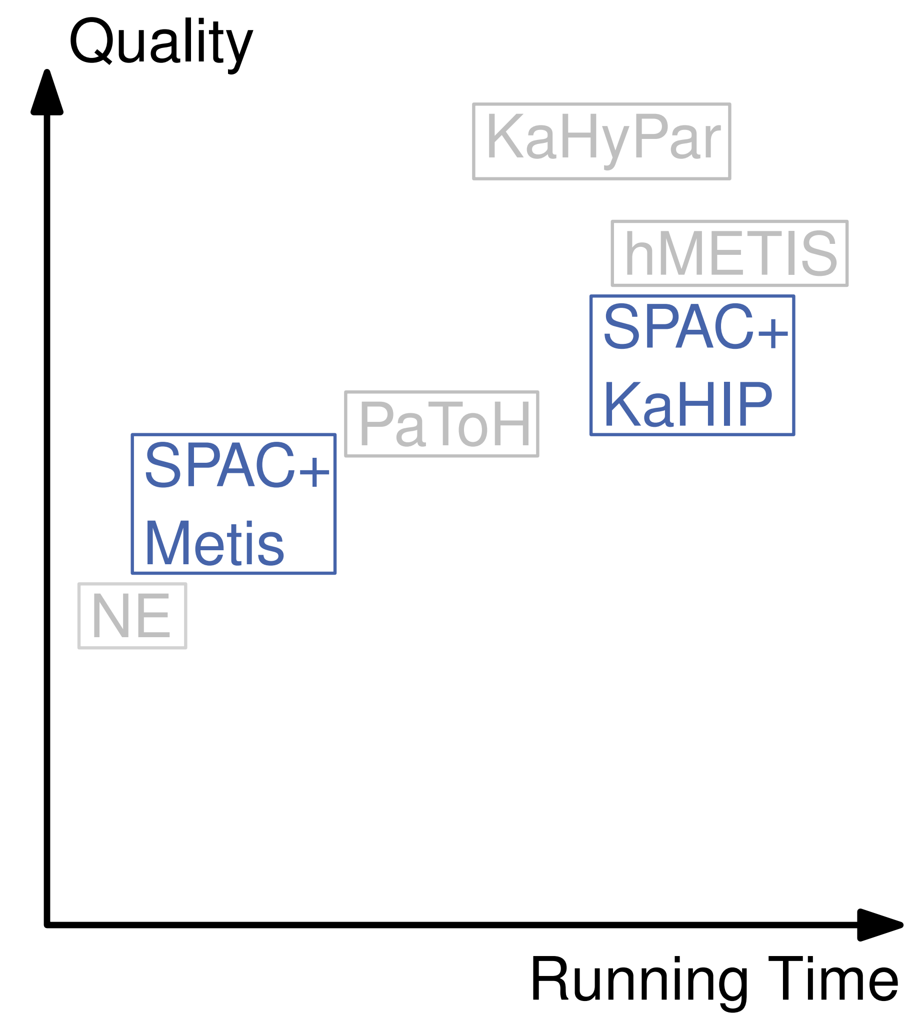
Hypergraph Model:

- Graph edge $\rightsquigarrow$ vertex
- Graph node $\rightsquigarrow$ hyperedge
- Optimize connectivity



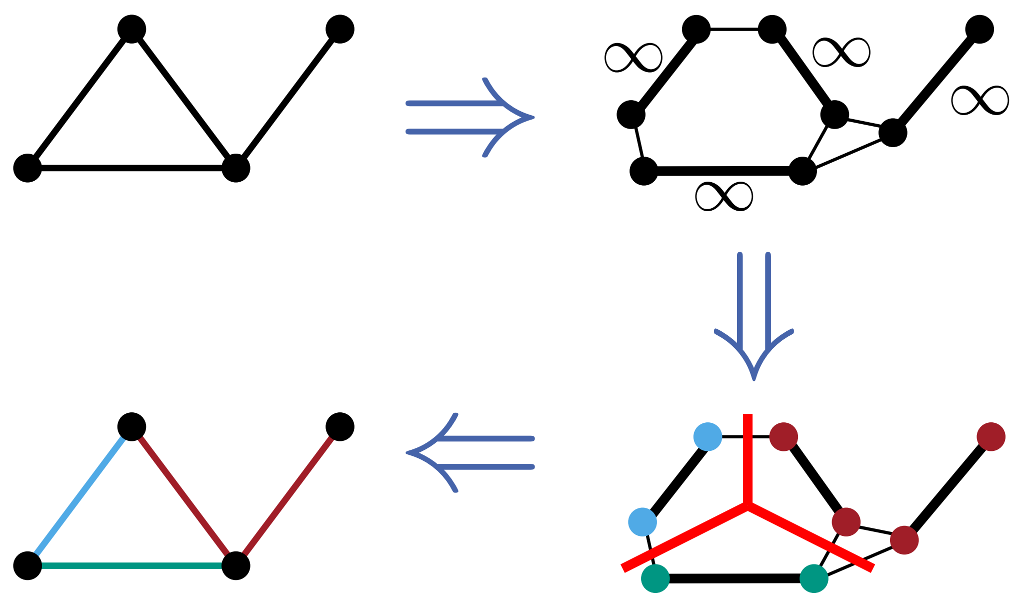
Edge Partitioning Algorithms

Sequential

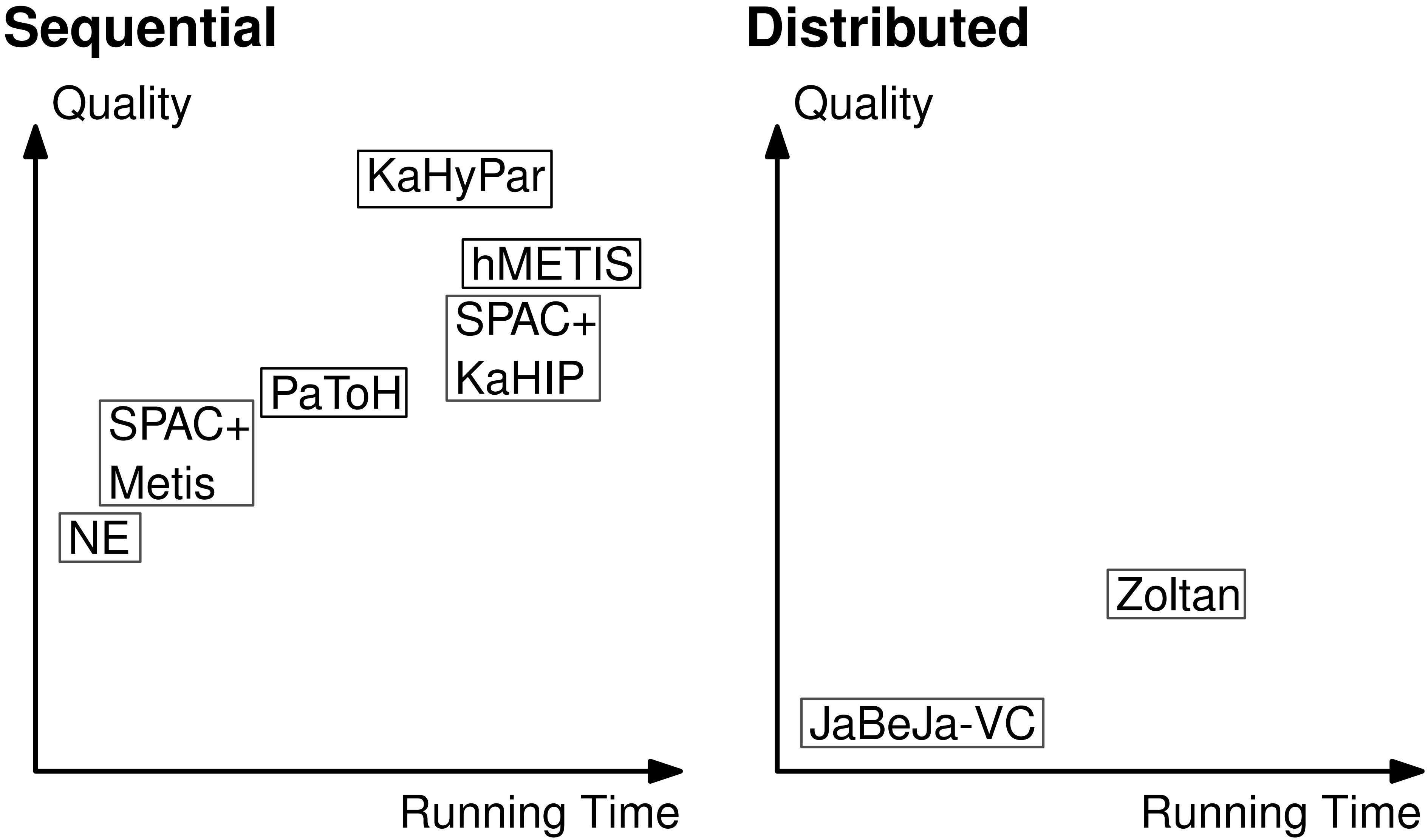


Split-And-Connect (SPAC) [Li et al.'17]:

- Build auxilary graph
- Use vertex partitioning algorithm



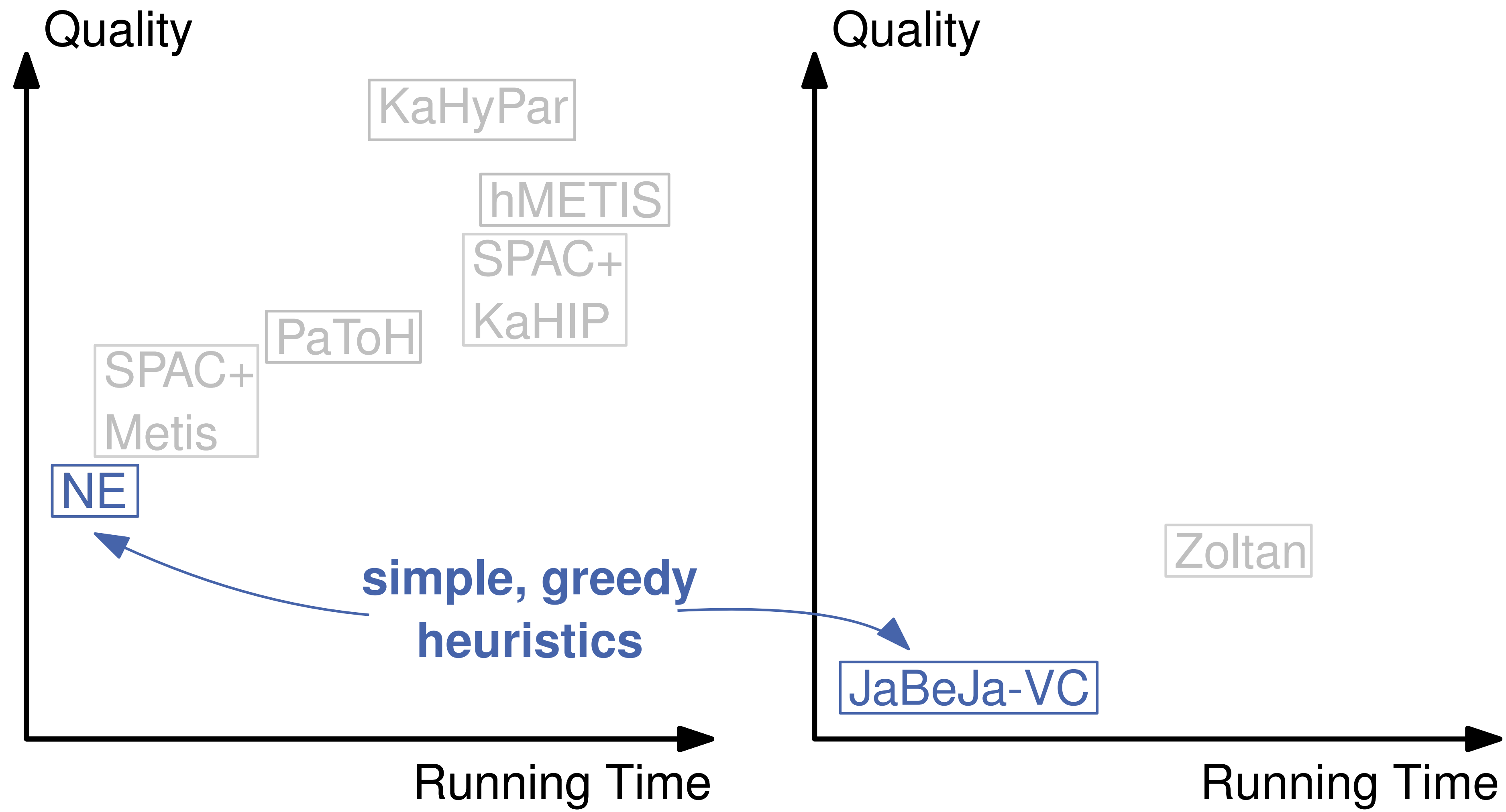
Edge Partitioning Algorithms



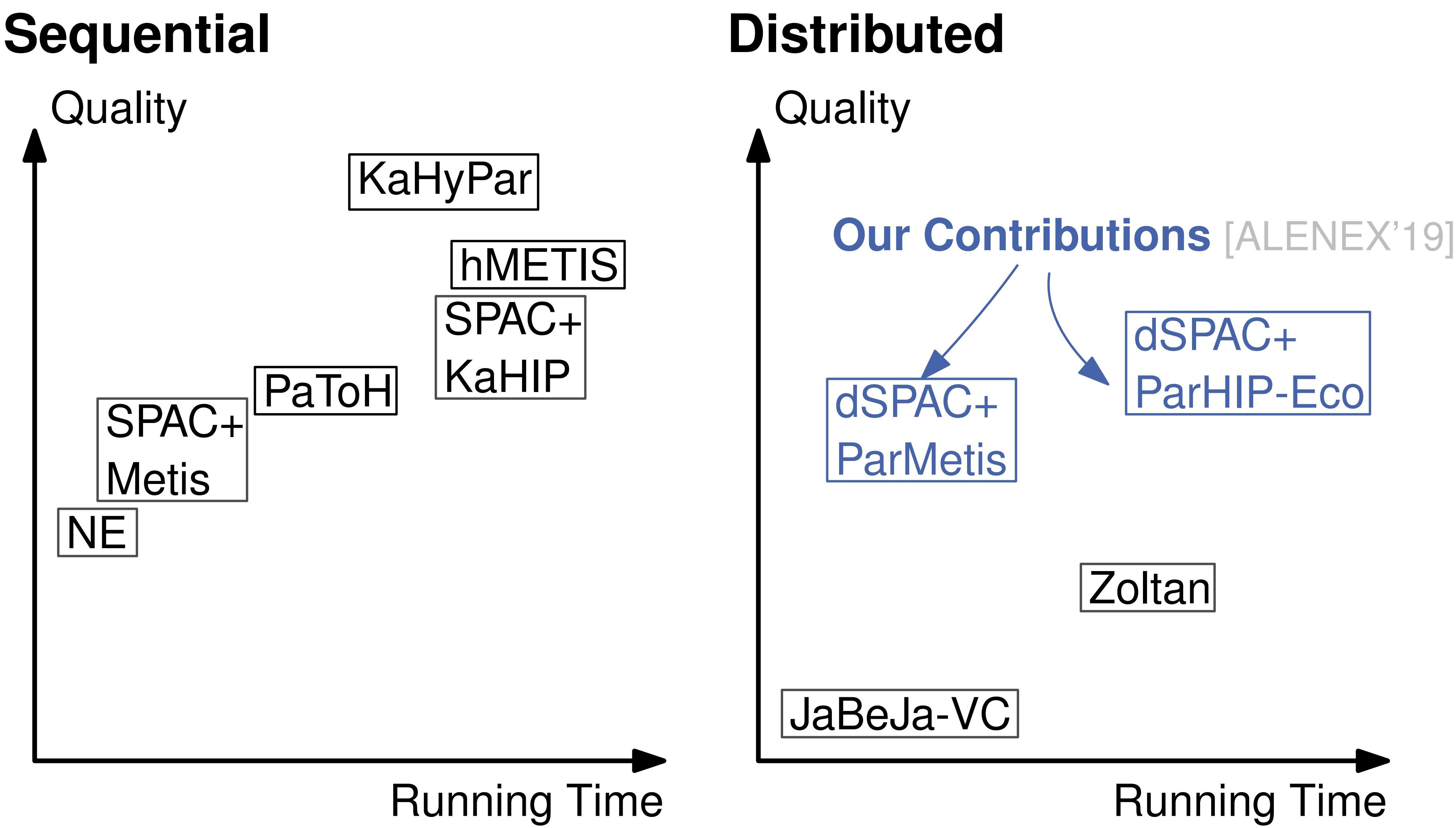
Edge Partitioning Algorithms

Sequential

Distributed



Edge Partitioning Algorithms



Experiments: Benchmark Setup

- Test suite: 70 graphs
  - Walshaw Graph Archive
  - Sparse Matrix-Vector Multiplication
  - Web & Social Graphs
  - Random Geometric Graphs
- $k \in \{2, 4, 8, 16, 32, 64, 128\}$
- Imbalance: $\epsilon = 3\%$
- Averages of 5 repetitions
- Sequential: 1 core
- Distributed: 32 * 20 cores

Competitors:

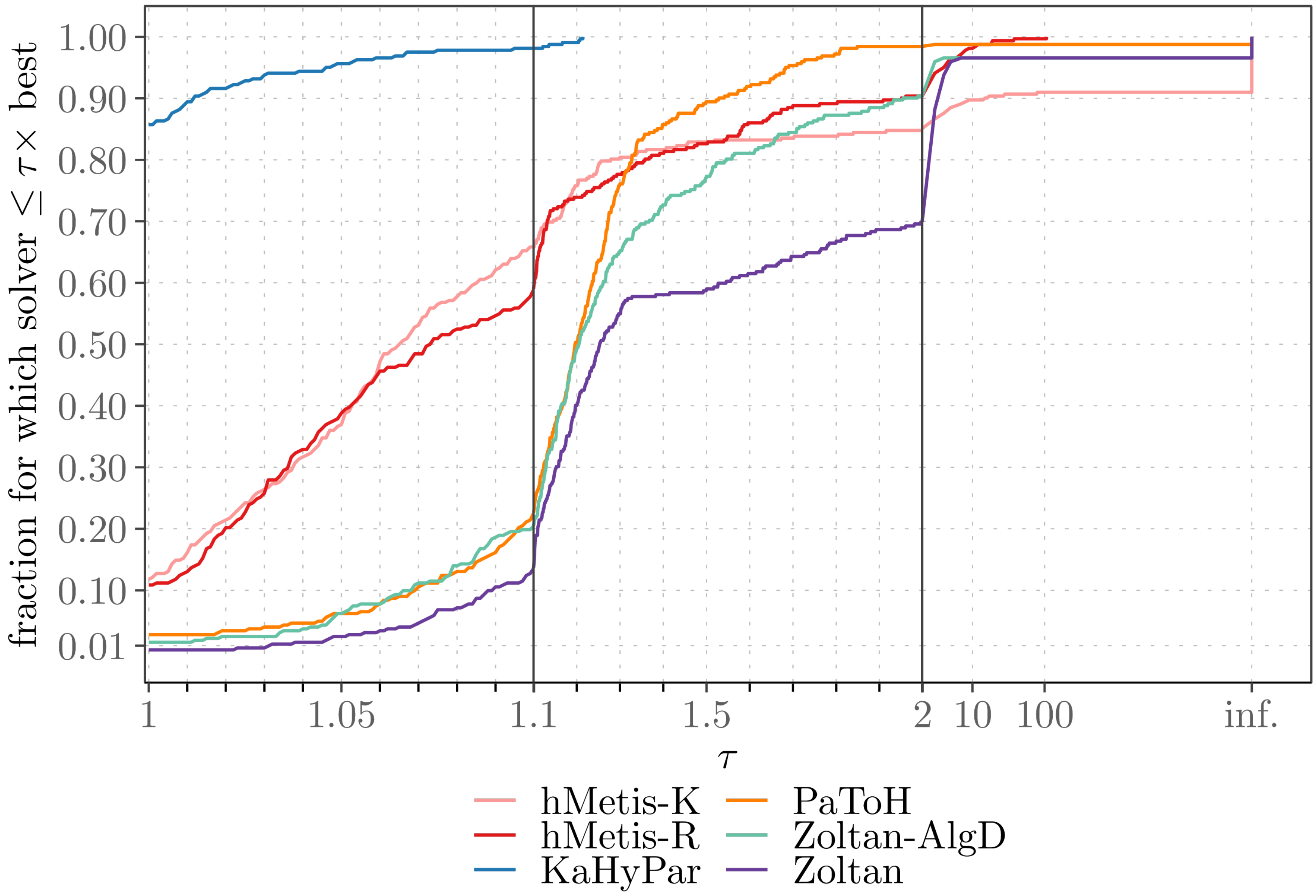
HGP's
- KaHyPar-MF
- PaToH
- Zoltan
- Zoltan-AlgD
- hMetis-{R, K}

- JaBeJa-VC
- NE

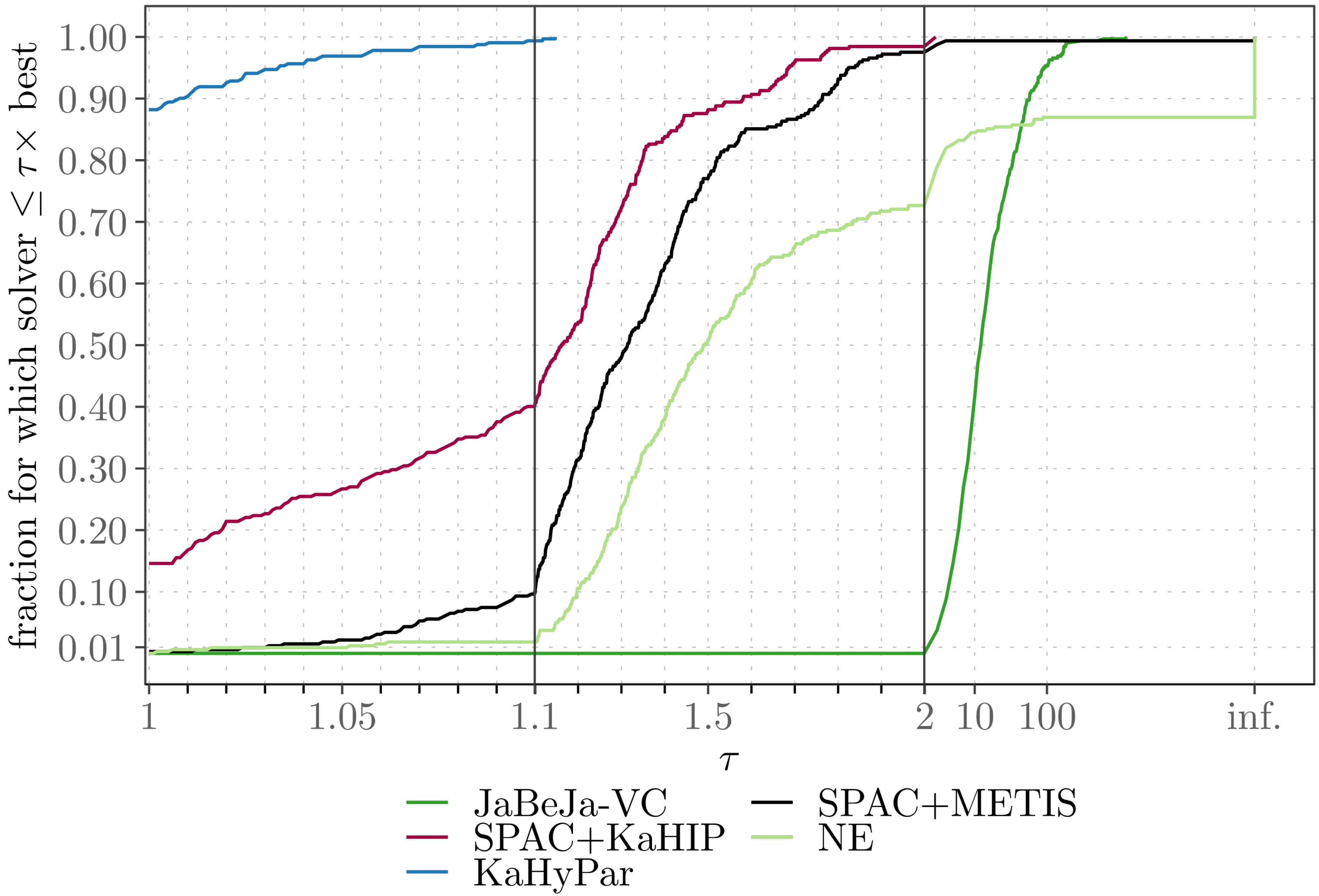
- SPAC + KaHIP
- SPAC + Metis

- dSPAC + ParHIP
- dSPAC + ParMetis

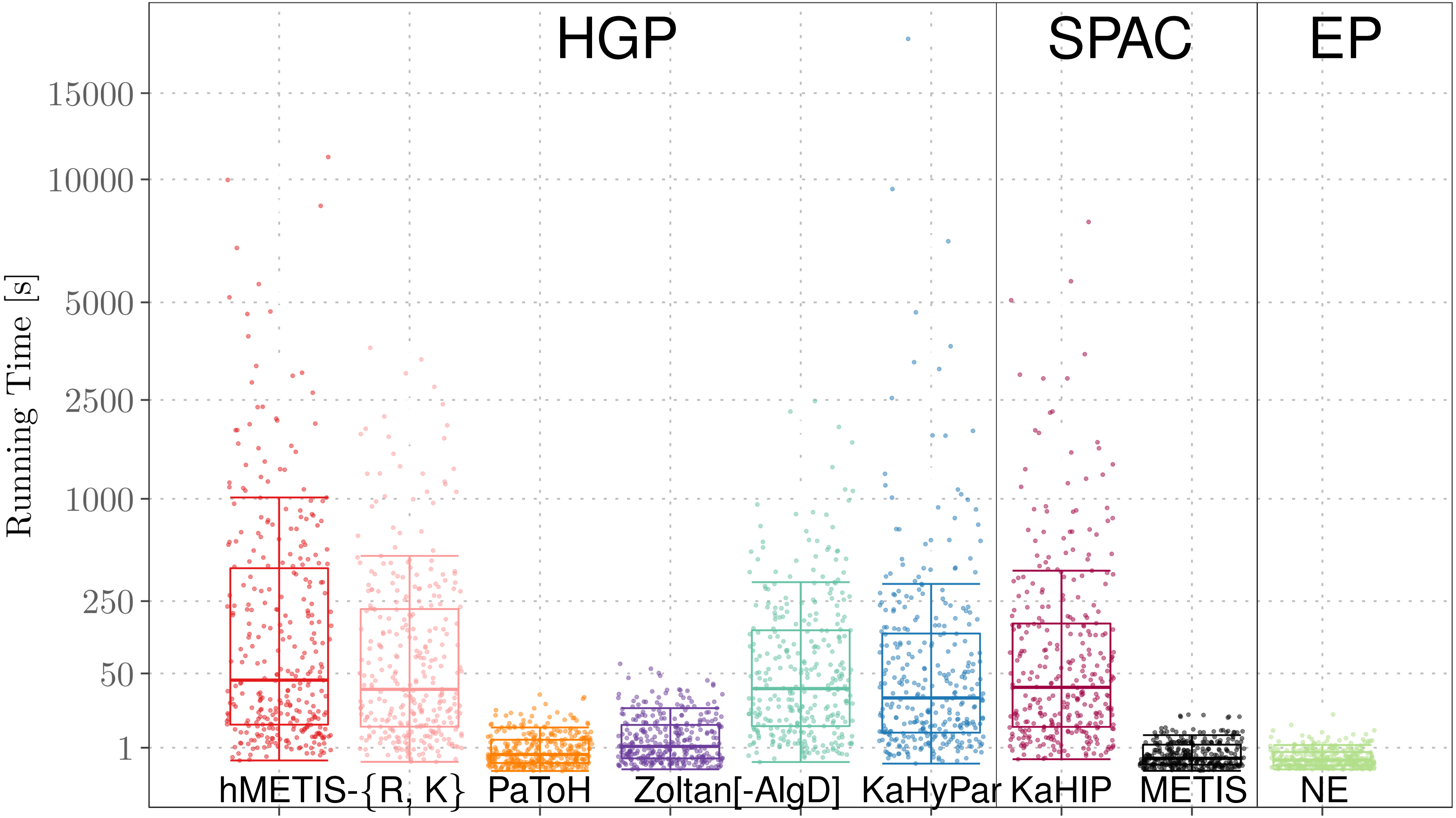
Experiments: Sequential HGP



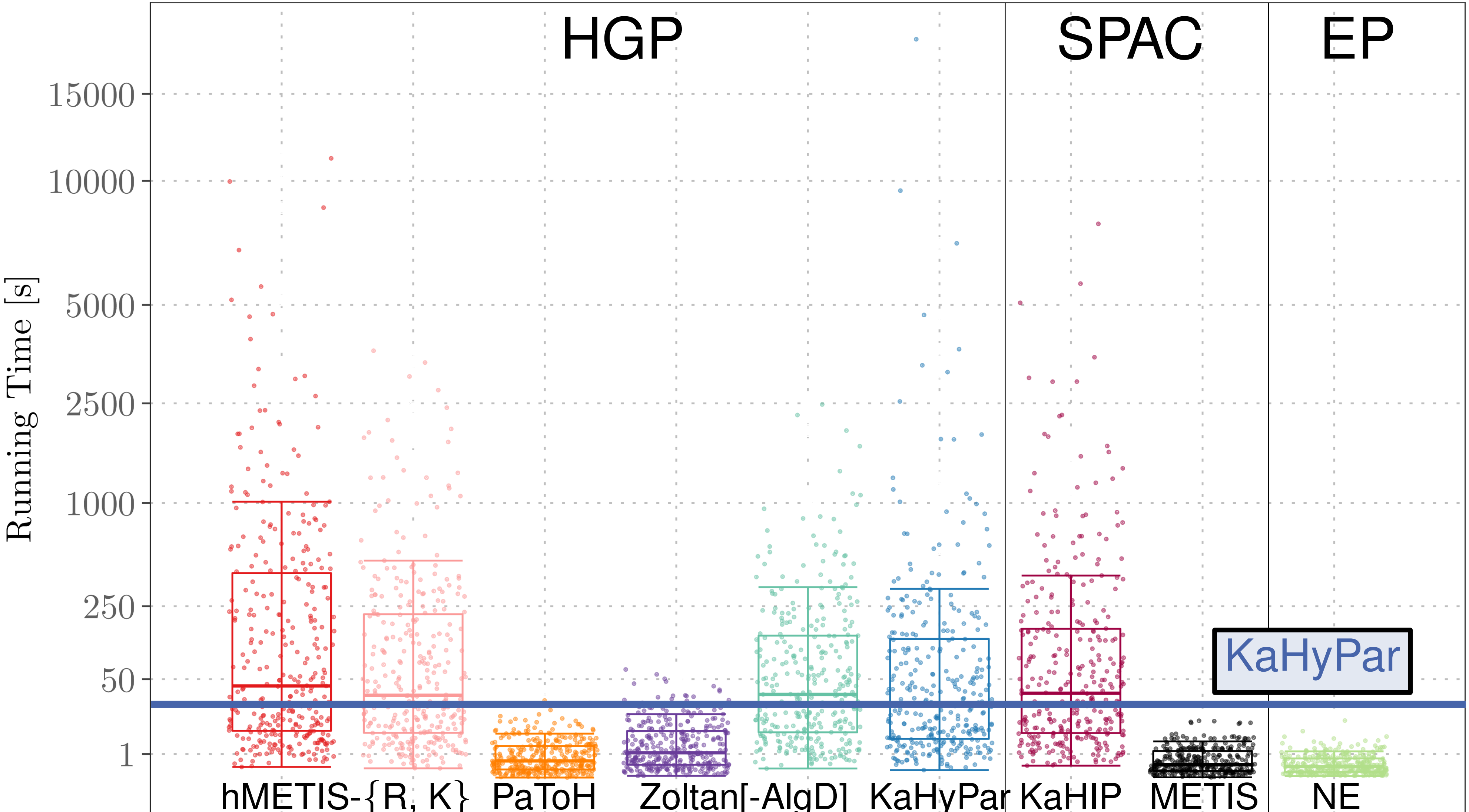
Experiments: Sequential HGP & SPAC+X



Experiments: Sequential Running Time

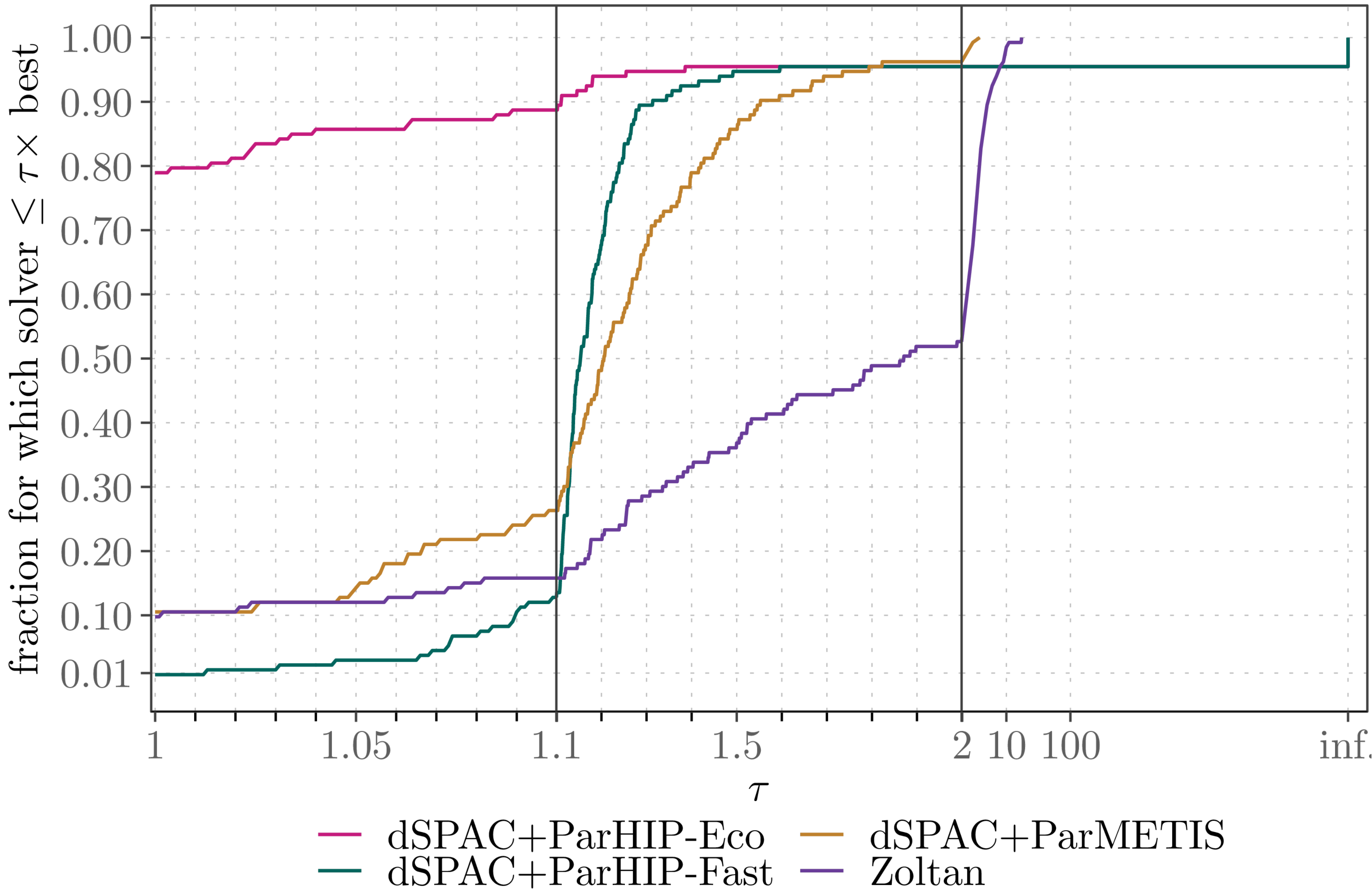


Experiments: Sequential Running Time

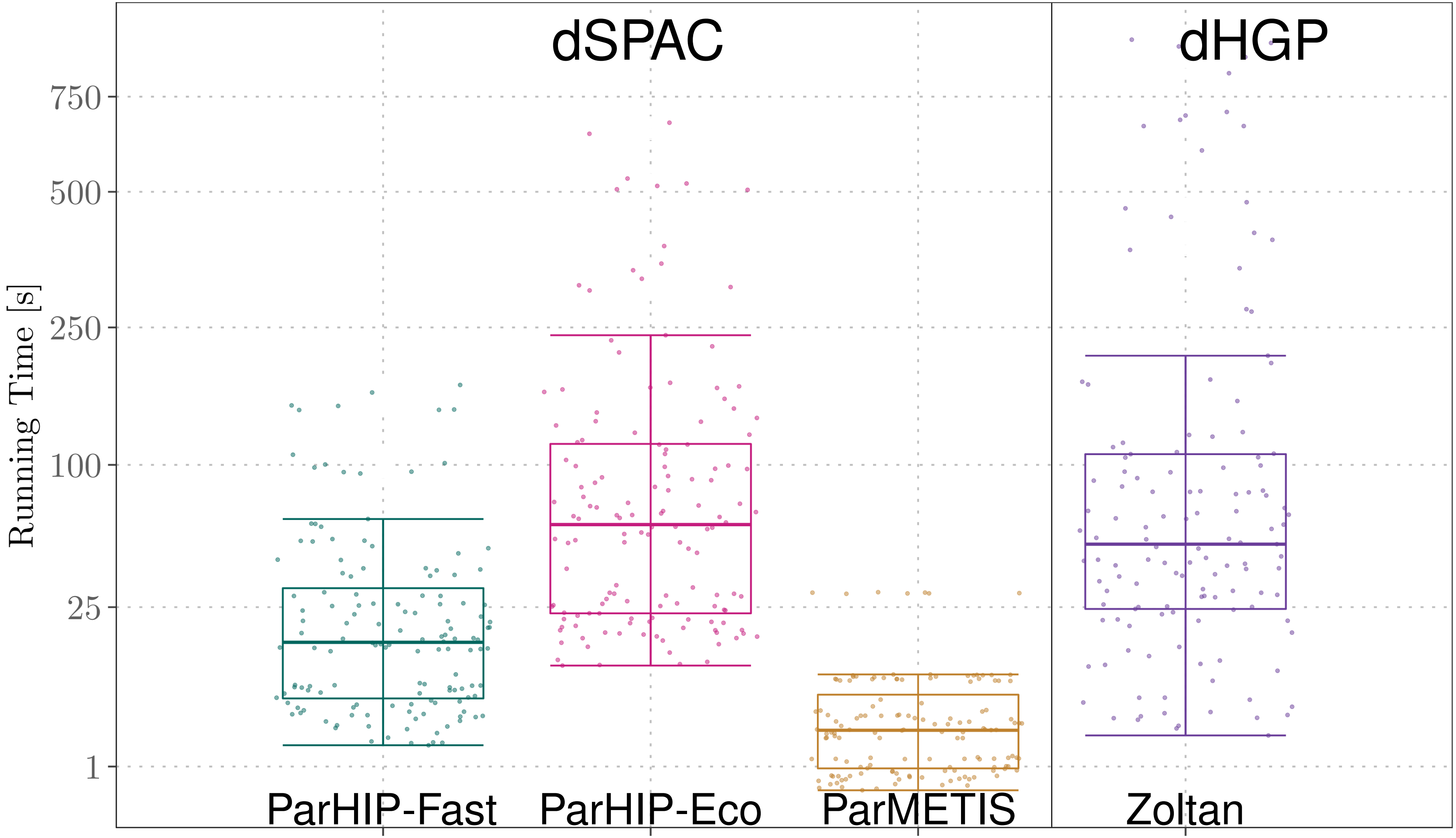


Institute of Theoretical Informatics
Algorithmics Group

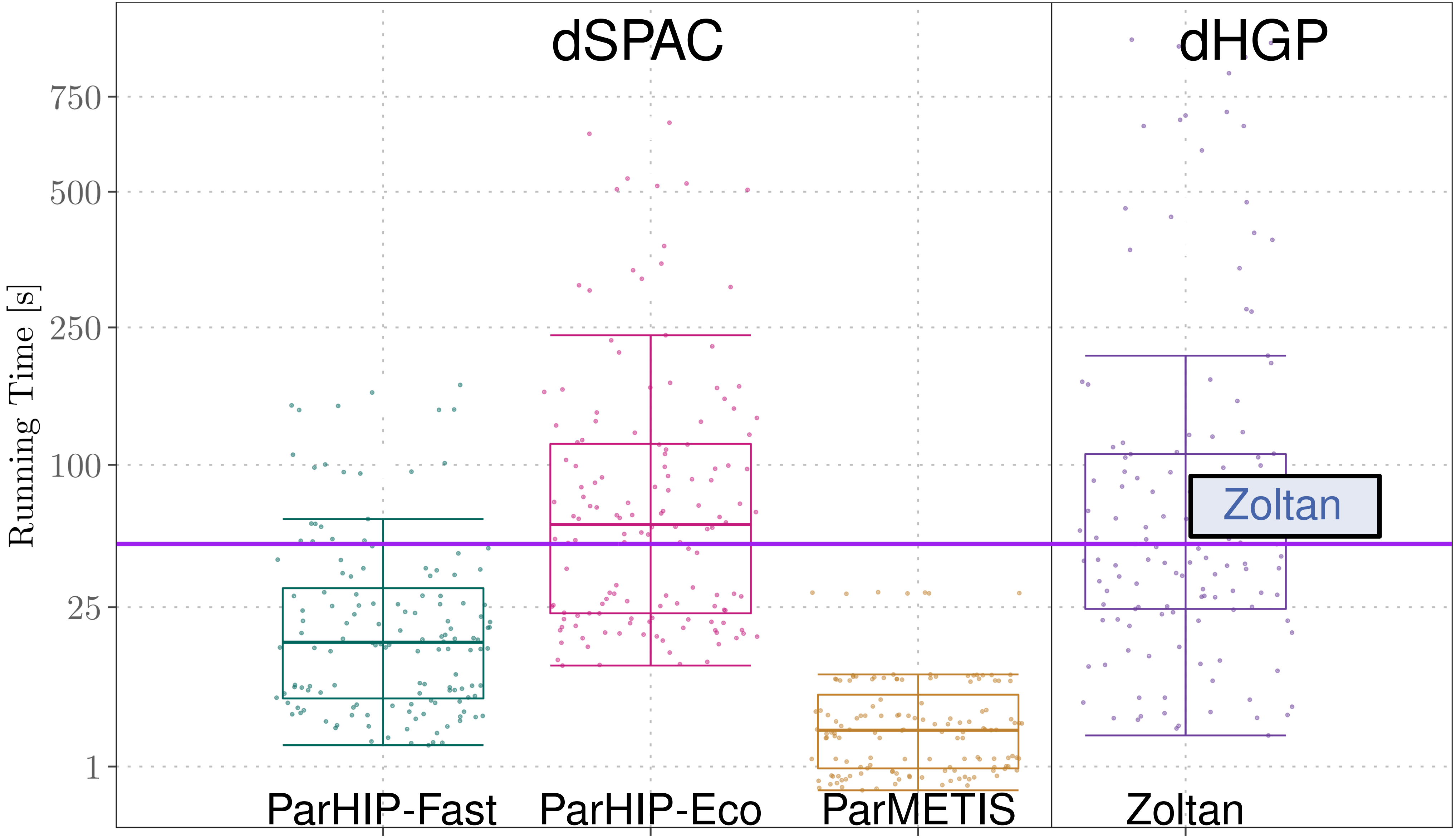
Experiments: Distributed HGP & dSPAC+X



Experiments: Distributed Running Time



Experiments: Distributed Running Time



Conclusion & Outlook

- **High-Quality Graph & Hypergraph Partitioning Frameworks:**
  - KaHIP– http://algo2.iti.kit.edu/kahip/
  - KaHyPar – http://www.kahypar.org
- **Future Work:**
  - Shared-Memory HGP
  - Distributed-Memory HGP
  - Shift focus towards fast (H)GP algorithms with reasonable quality
- **(Personal) Open Questions:**
  - What would benefit the CSC community?
  - What are ”difficult” instances?

Institute of Theoretical Informatics
Algorithmics Group