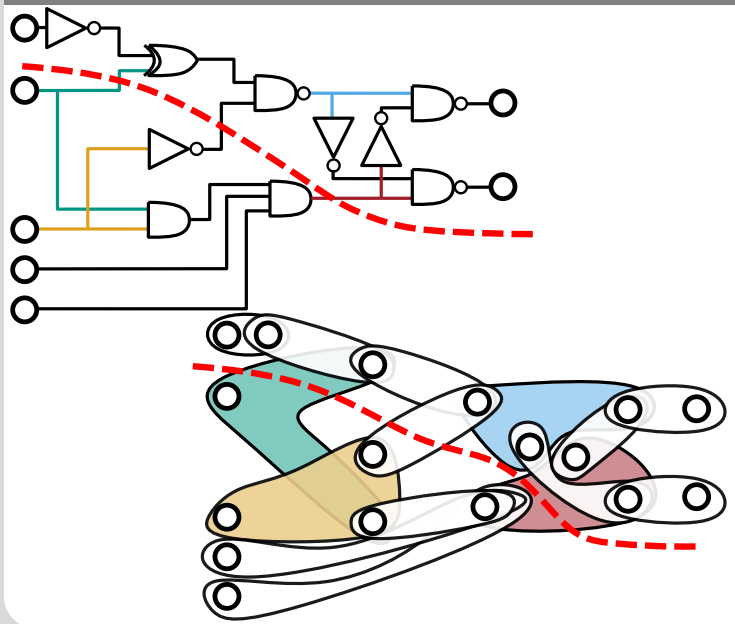
Karlsruher Institut für Technologie

Hochqualitative Hypergraphpartitionierung

Promotionsvortrag · 11. Dezember 2019

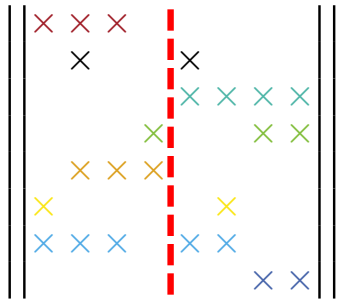
Sebastian Schlag

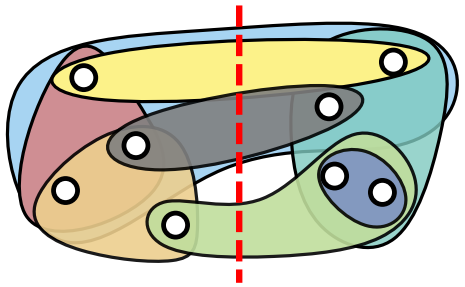
Institut für Theoretische Informatik · Algorithmik



KaHyPar

KARLSRUHE HYPERGRAPH PARTITIONING

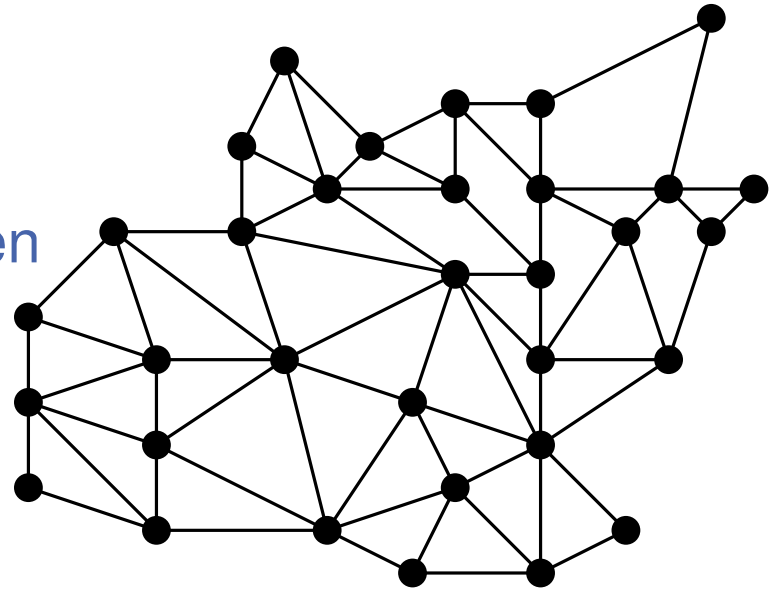




Graphen und Hypergraphen

Graph $G = (V, E)$

Knoten, Kanten

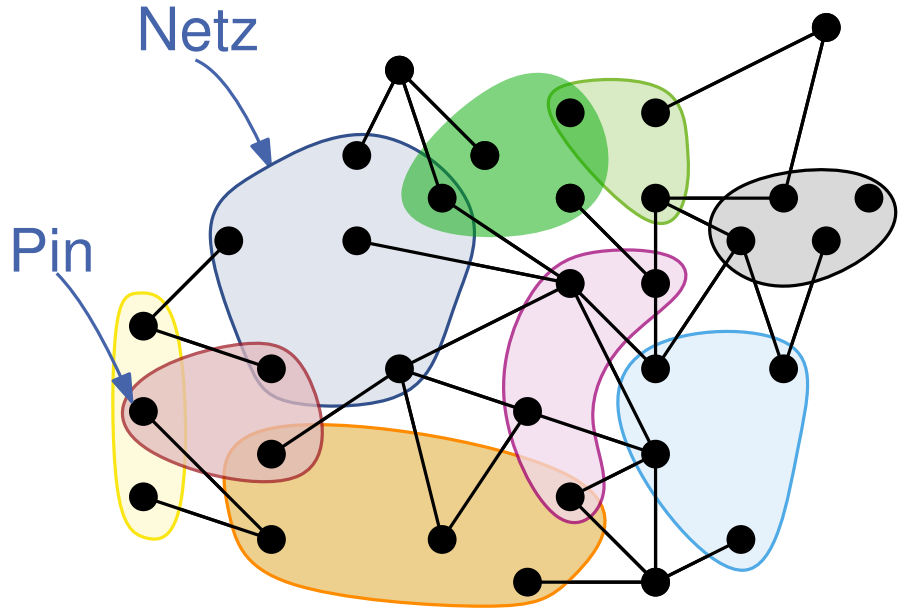


- Modelliert Beziehungen zwischen Objekten
- Zweierbeziehungen

Hypergraph $H = (V, E, c, \omega)$

- Generalisierung

  $\Rightarrow$ Hyperkanten verbinden ≥ 2 Knoten
- Komplexere Beziehungen
- Knotengewichte $c : V \to \mathbb{R}_{>0}$
- Kantengewichte $\omega : E \to \mathbb{R}_{>0}$



Institut für Theoretische Informatik
Lehrstuhl Algorithmik

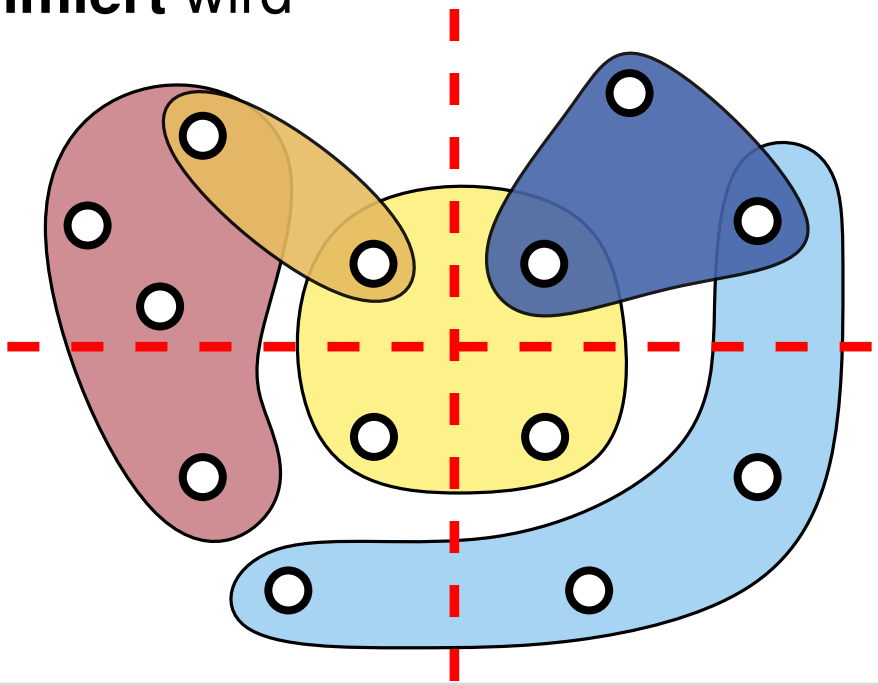
ε-balancierte Hypergraphpartitionierung (HGP)

Partitioniere Hypergraphen $H = (V, E, c, \omega)$ so in **k** Blöcke $\Pi = \{V_1, \ldots, V_k\}$, dass:

- Blöcke V_i **ungefähr gleich schwer** sind:

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

- **Zielfunktion** auf den Hyperkanten **minimiert** wird



ε-balancierte Hypergraphpartitionierung (HGP)

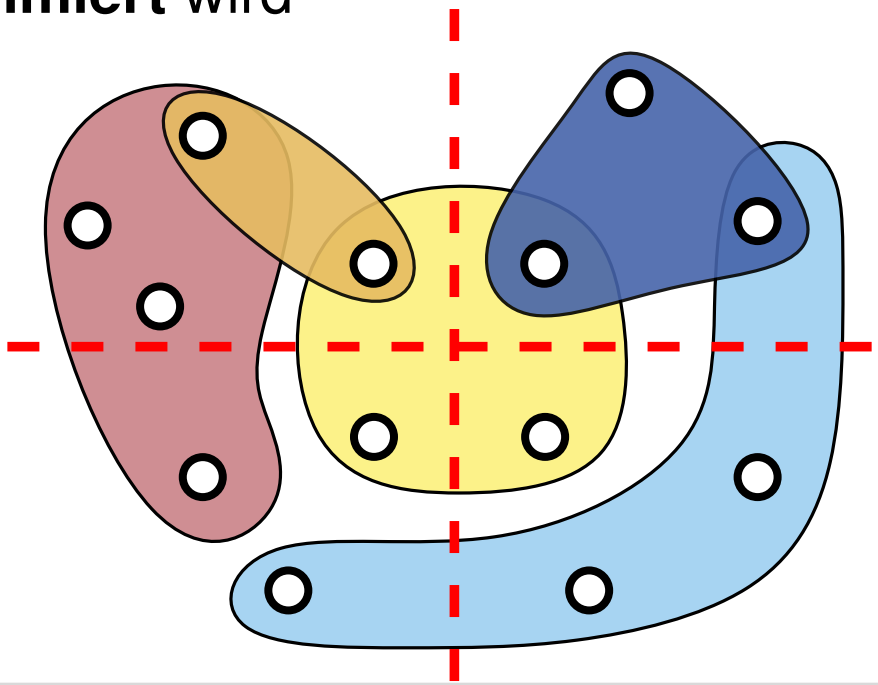
Partitioniere Hypergraphen $H = (V, E, c, \omega)$ so in **k** Blöcke $\Pi = \{V_1, \ldots, V_k\}$, dass:

- Blöcke V_i **ungefähr gleich schwer** sind:

Balancierungsparameter

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

- **Zielfunktion** auf den Hyperkanten **minimiert** wird



ε-balancierte Hypergraphpartitionierung (HGP)

Partitioniere Hypergraphen $H = (V, E, c, \omega)$ so in **k** Blöcke $\Pi = \{V_1, \ldots, V_k\}$, dass:

- Blöcke V_i **ungefähr gleich schwer** sind:

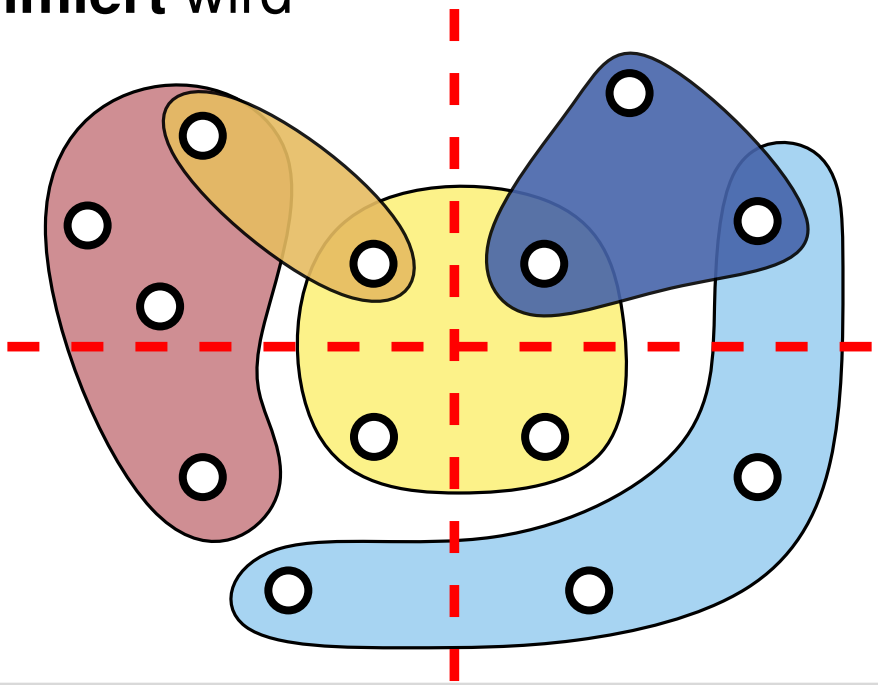
Balancierungsparameter

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

- **Zielfunktion** auf den Hyperkanten **minimiert** wird

Gängige Zielfunktionen:

- Kantenschnitt: $\sum_{e \in \text{Schnitt}} \omega(e)$



ε-balancierte Hypergraphpartitionierung (HGP)

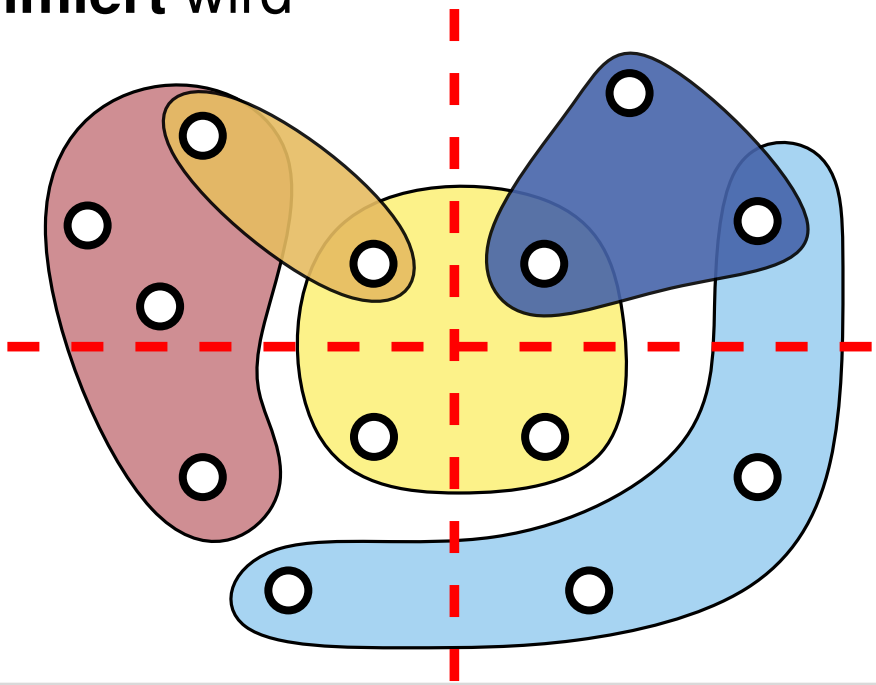
Partitioniere Hypergraphen $H = (V, E, c, \omega)$ so in **k** Blöcke $\Pi = \{V_1, \ldots, V_k\}$, dass:

- Blöcke V_i **ungefähr gleich schwer** sind:

Balancierungsparameter

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

- **Zielfunktion** auf den Hyperkanten **minimiert** wird



Gängige Zielfunktionen:

- Kantenschnitt: $\sum_{e \in \text{Schnitt}} \omega(e)$
- Konnektivität: $\sum_{e \in \text{Schnitt}} (\lambda - 1)\, \omega(e)$

ε-balancierte Hypergraphpartitionierung (HGP)

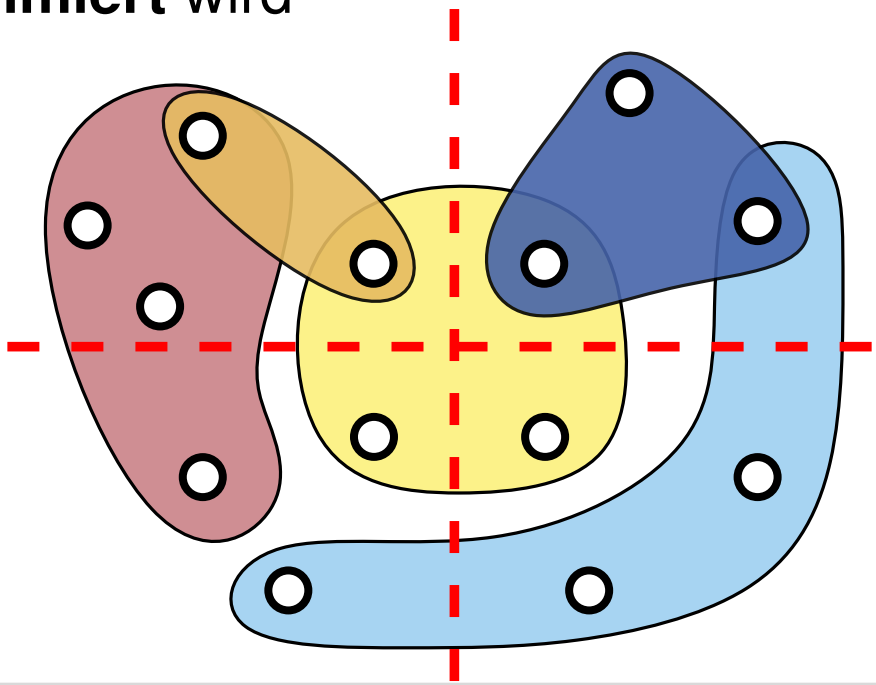
Partitioniere Hypergraphen $H = (V, E, c, \omega)$ so in **k** Blöcke $\Pi = \{V_1, \ldots, V_k\}$, dass:

- Blöcke V_i **ungefähr gleich schwer** sind:

Balancierungsparameter

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

- **Zielfunktion** auf den Hyperkanten **minimiert** wird



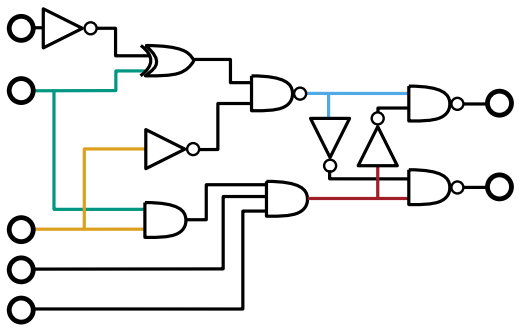
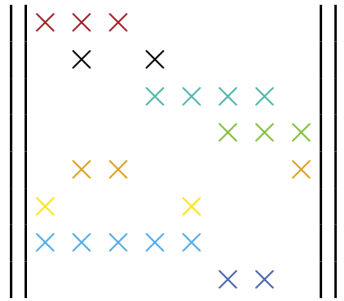
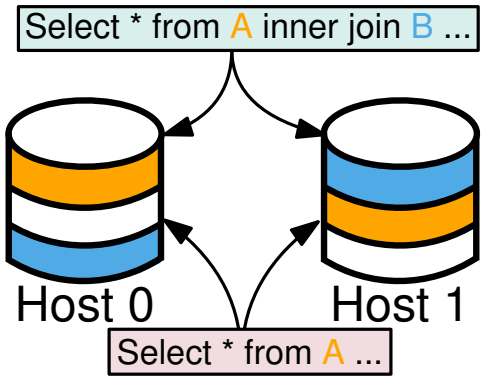
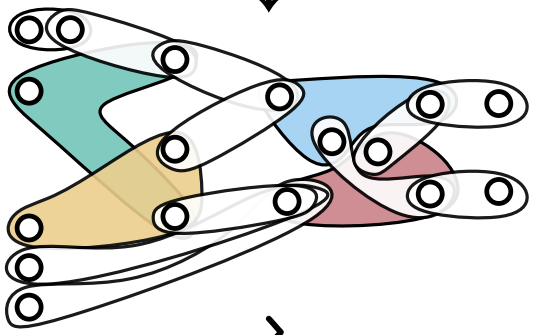
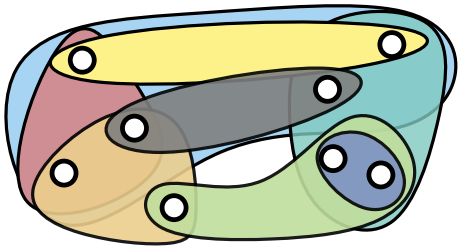
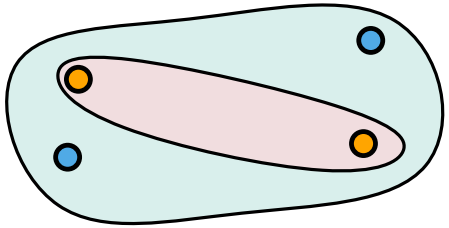
Gängige Zielfunktionen:

- Kantenschnitt: $\sum_{e \in \text{Schnitt}} \omega(e)$
- Konnektivität: $\sum_{e \in \text{Schnitt}} (\lambda - 1)\, \omega(e)$

durch e verbundene **Blöcke**

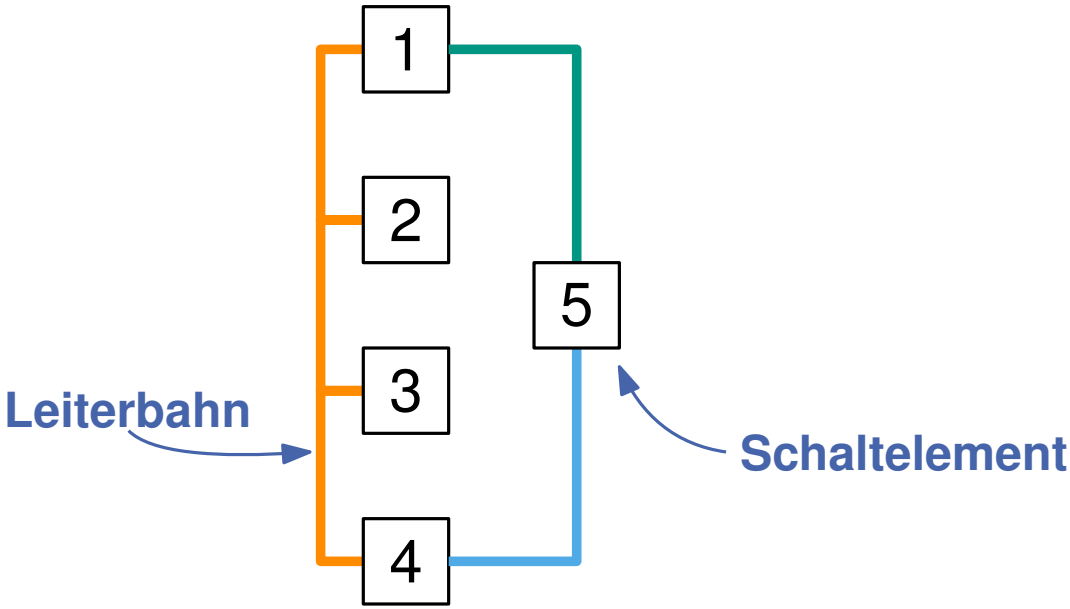
Bekannte Anwendungsbeispiele

Karlsruher Institut für Technologie

	VLSI Design	Wiss. Rechnen	Verteilte DBs
Anwendung			
Modell			
Ziel	Platzierung vereinfachen	Kommunikation minimieren	Verteilte Anfragen minimieren

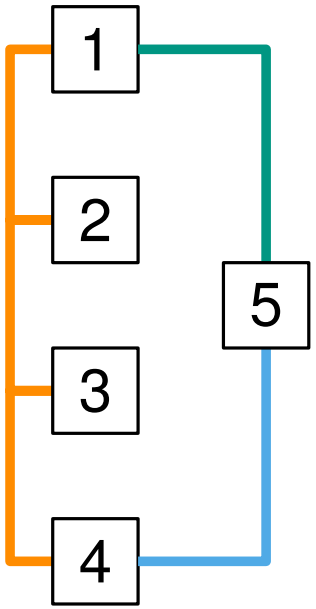
Motivation: Grenzen der Graphpartitionierung

Elektrische Schaltkreise [SK72]



Motivation: Grenzen der Graphpartitionierung

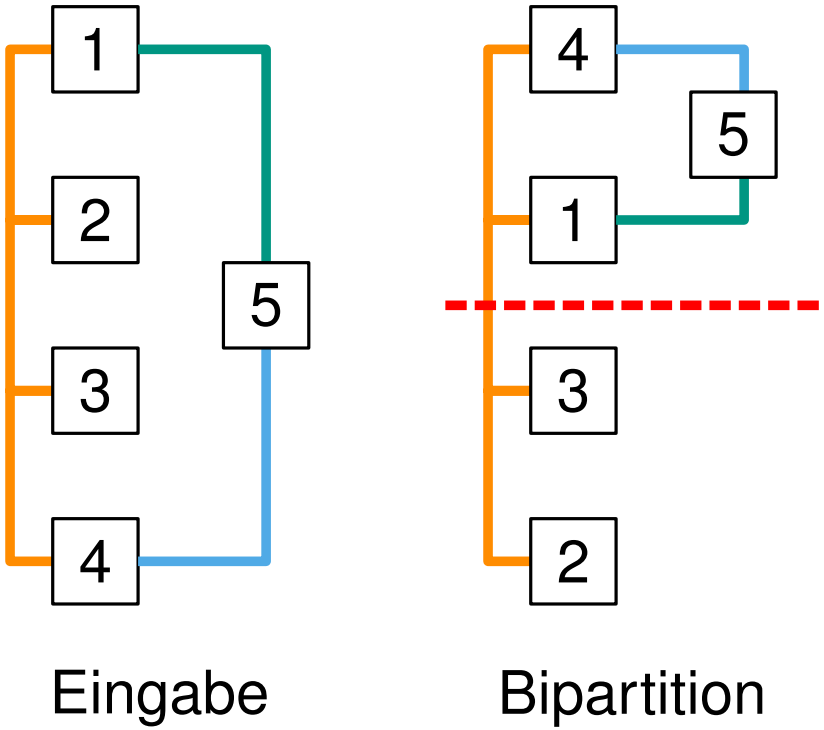
Elektrische Schaltkreise [SK72]



Eingabe

Motivation: Grenzen der Graphpartitionierung

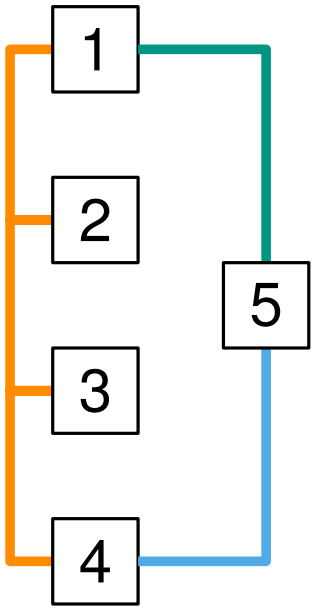
Elektrische Schaltkreise [SK72]



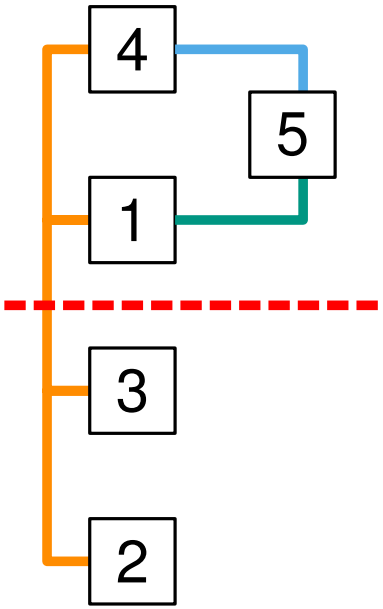
Eingabe Bipartition

Motivation: Grenzen der Graphpartitionierung

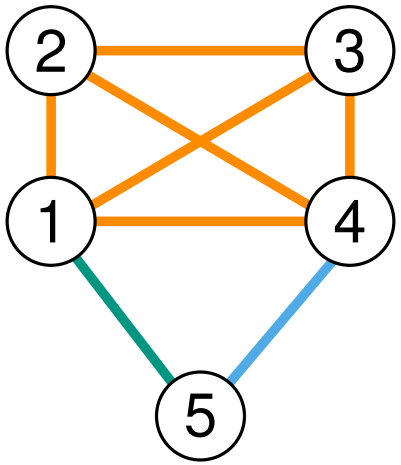
Elektrische Schaltkreise [SK72]



Eingabe



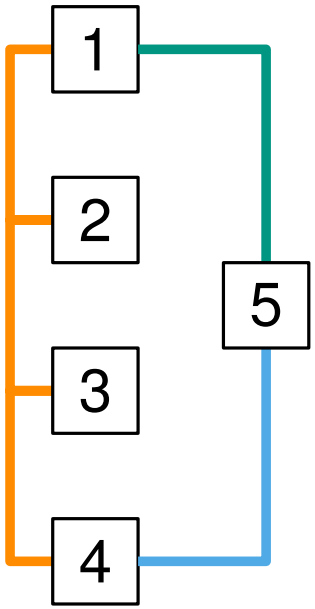
Bipartition



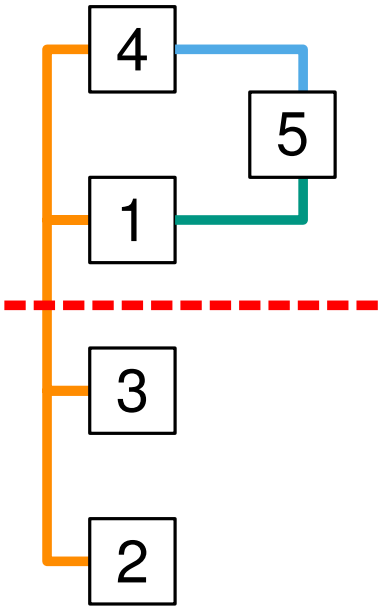
Graph

Motivation: Grenzen der Graphpartitionierung

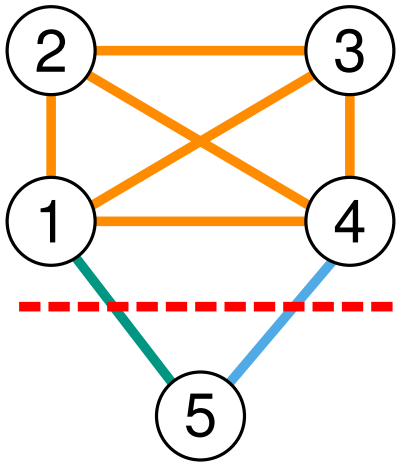
Elektrische Schaltkreise [SK72]



Eingabe



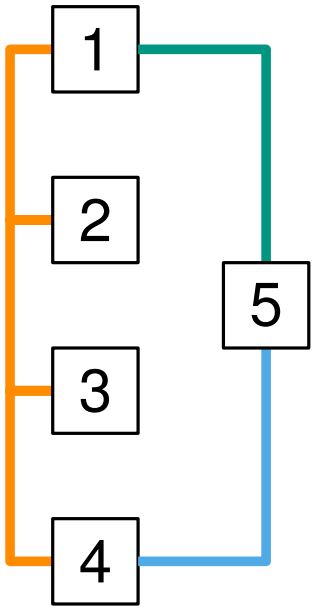
Bipartition



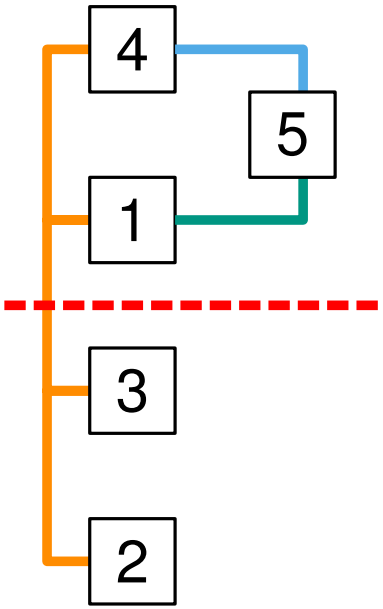
Graphpartition

Motivation: Grenzen der Graphpartitionierung

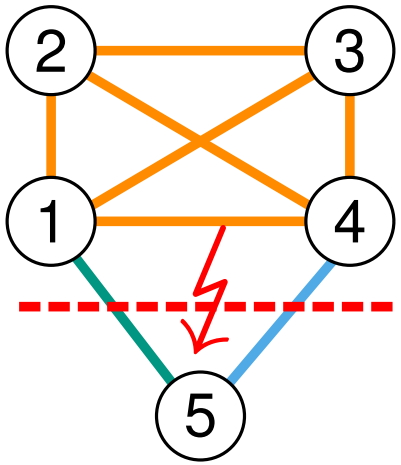
Elektrische Schaltkreise [SK72]



Eingabe



Bipartition

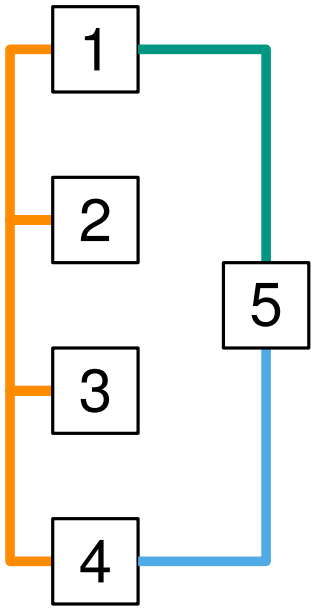


Graphpartition

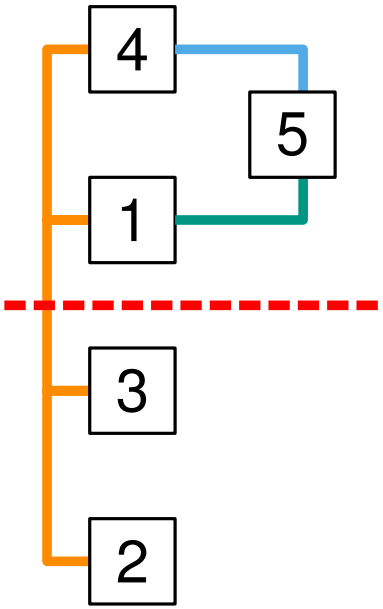
Problem: Beschränktheit auf **Zweierbeziehungen**!

Motivation: Grenzen der Graphpartitionierung

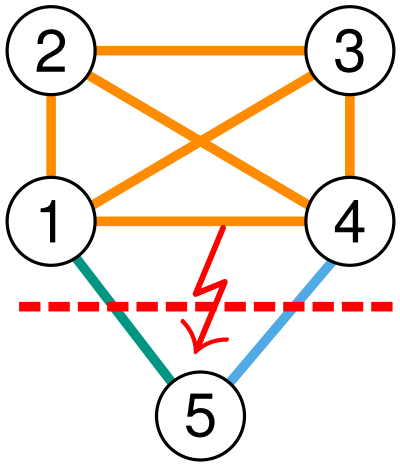
Elektrische Schaltkreise [SK72]



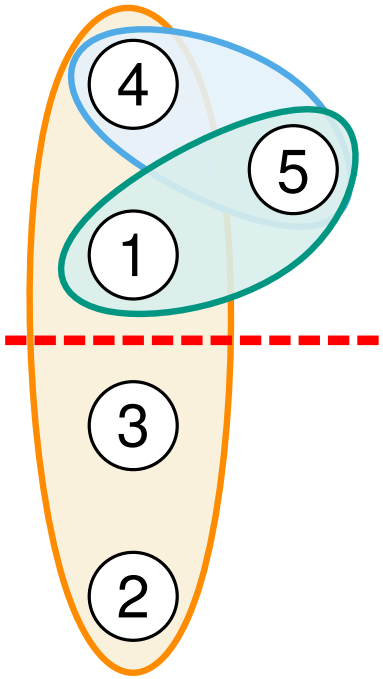
Eingabe



Bipartition



Graphpartition



Hypergraphpartition

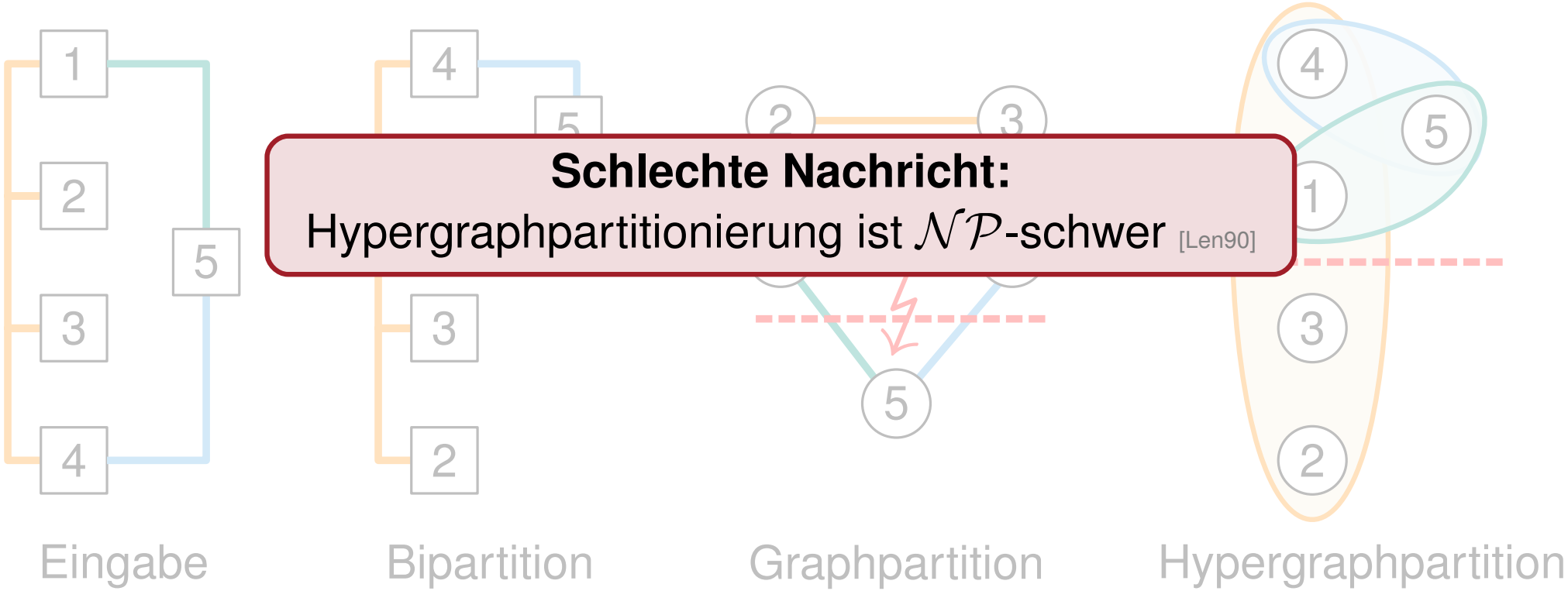
Problem: Beschränktheit auf **Zweierbeziehungen**!

Lösung: **Hypergraphen**!

Motivation: Grenzen der Graphpartitionierung

Karlsruher Institut für Technologie

Elektrische Schaltkreise [SK72]



Schlechte Nachricht:
Hypergraphpartitionierung ist $\mathcal{NP}$-schwer [Len90]

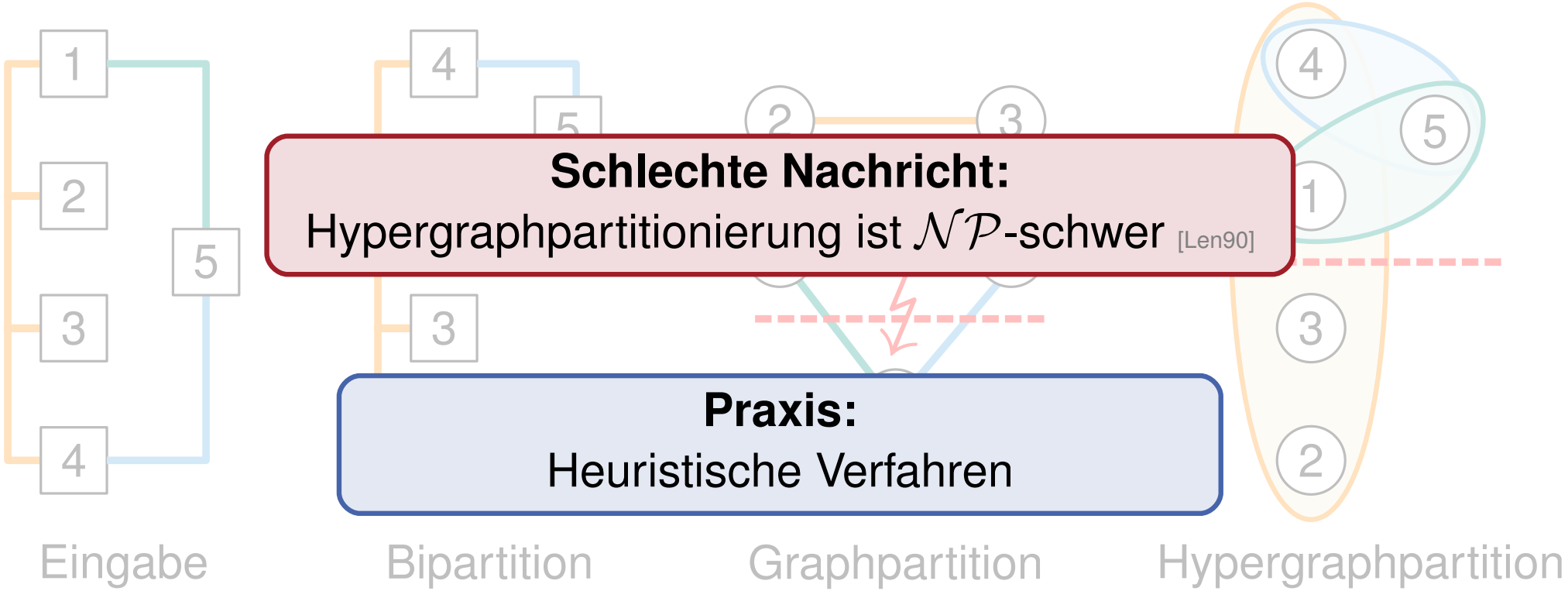
Problem: Beschränktheit auf **Zweierbeziehungen**!

Lösung: **Hypergraphen**!

Motivation: Grenzen der Graphpartitionierung

Karlsruher Institut für Technologie

Elektrische Schaltkreise [SK72]



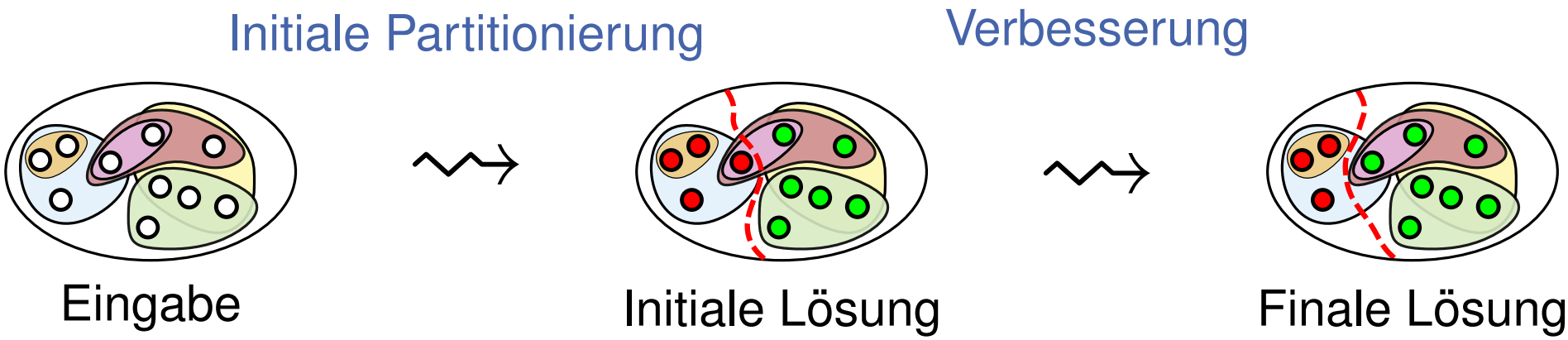
Schlechte Nachricht:
Hypergraphpartitionierung ist $\mathcal{NP}$-schwer [Len90]

Praxis:
Heuristische Verfahren

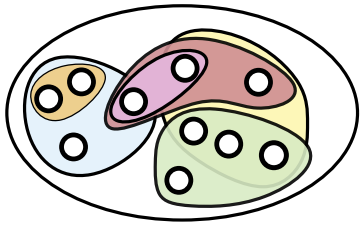
Problem: Beschränktheit auf **Zweierbeziehungen**!

Lösung: **Hypergraphen**!

Klassifikation: Ein-Level-Algorithmen

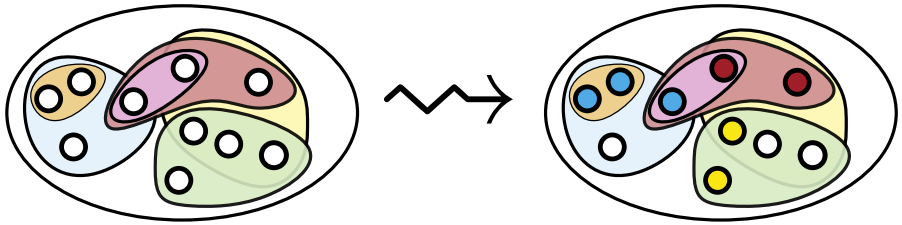


Klassifikation: Zwei-Level-Algorithmen

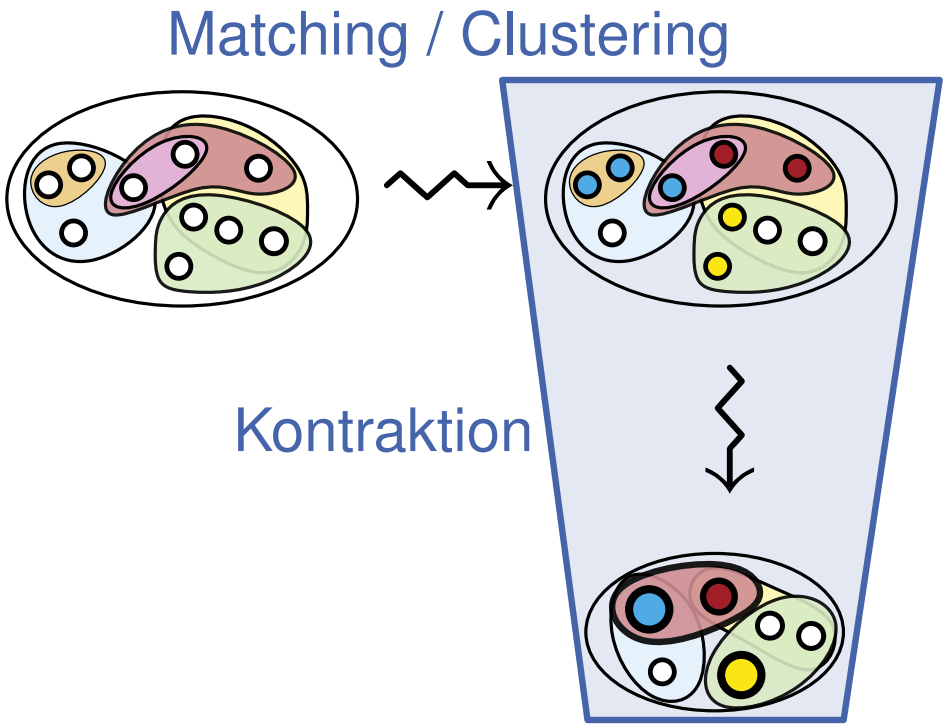


Klassifikation: Zwei-Level-Algorithmen

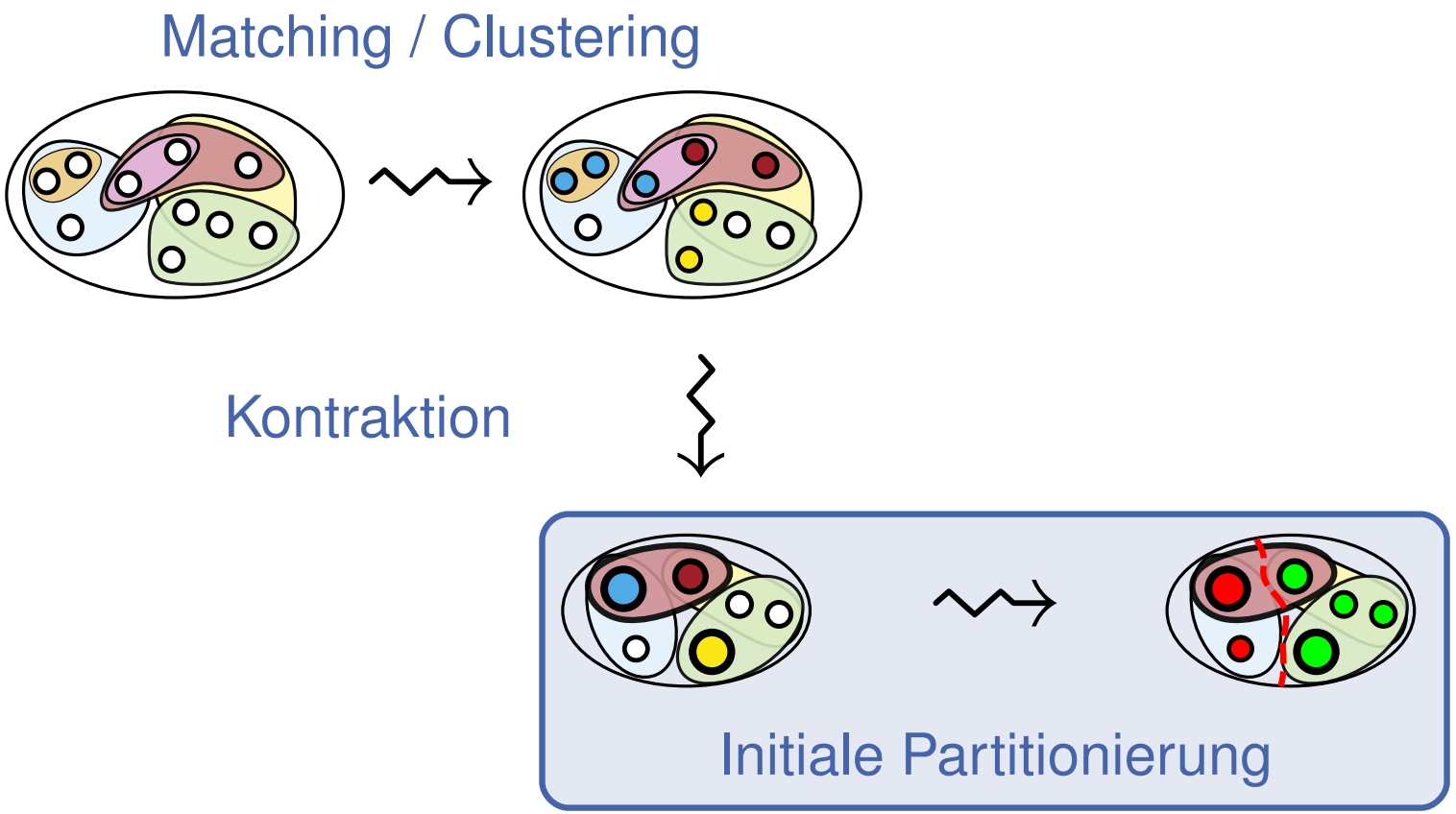
Matching / Clustering



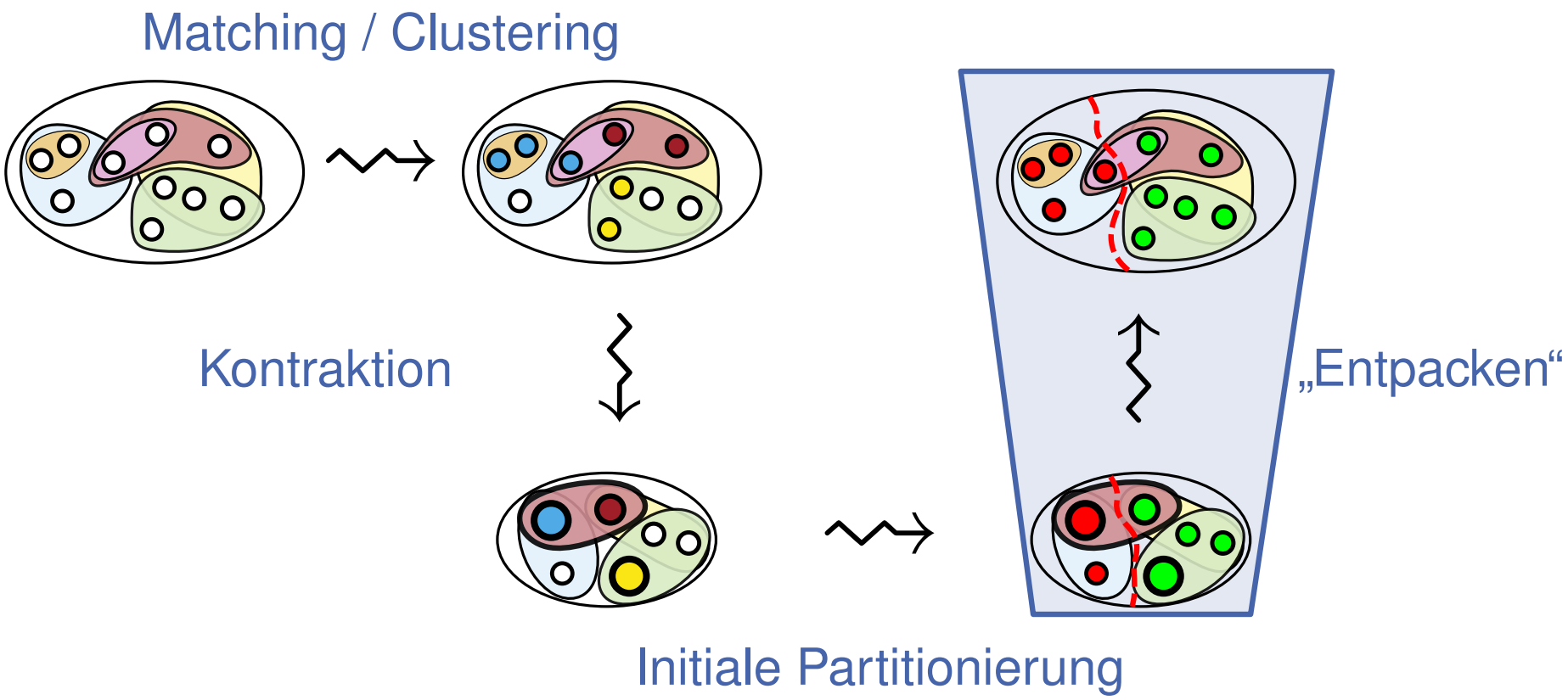
Klassifikation: Zwei-Level-Algorithmen



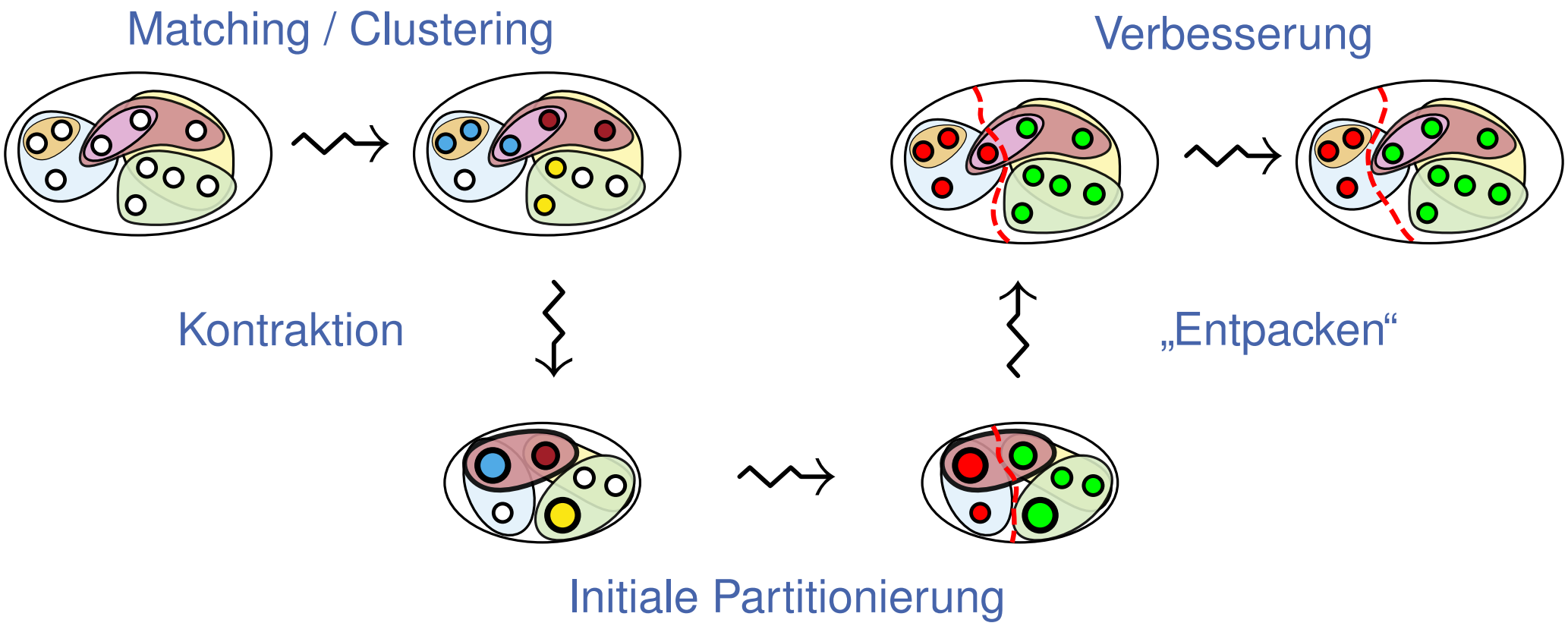
Klassifikation: Zwei-Level-Algorithmen



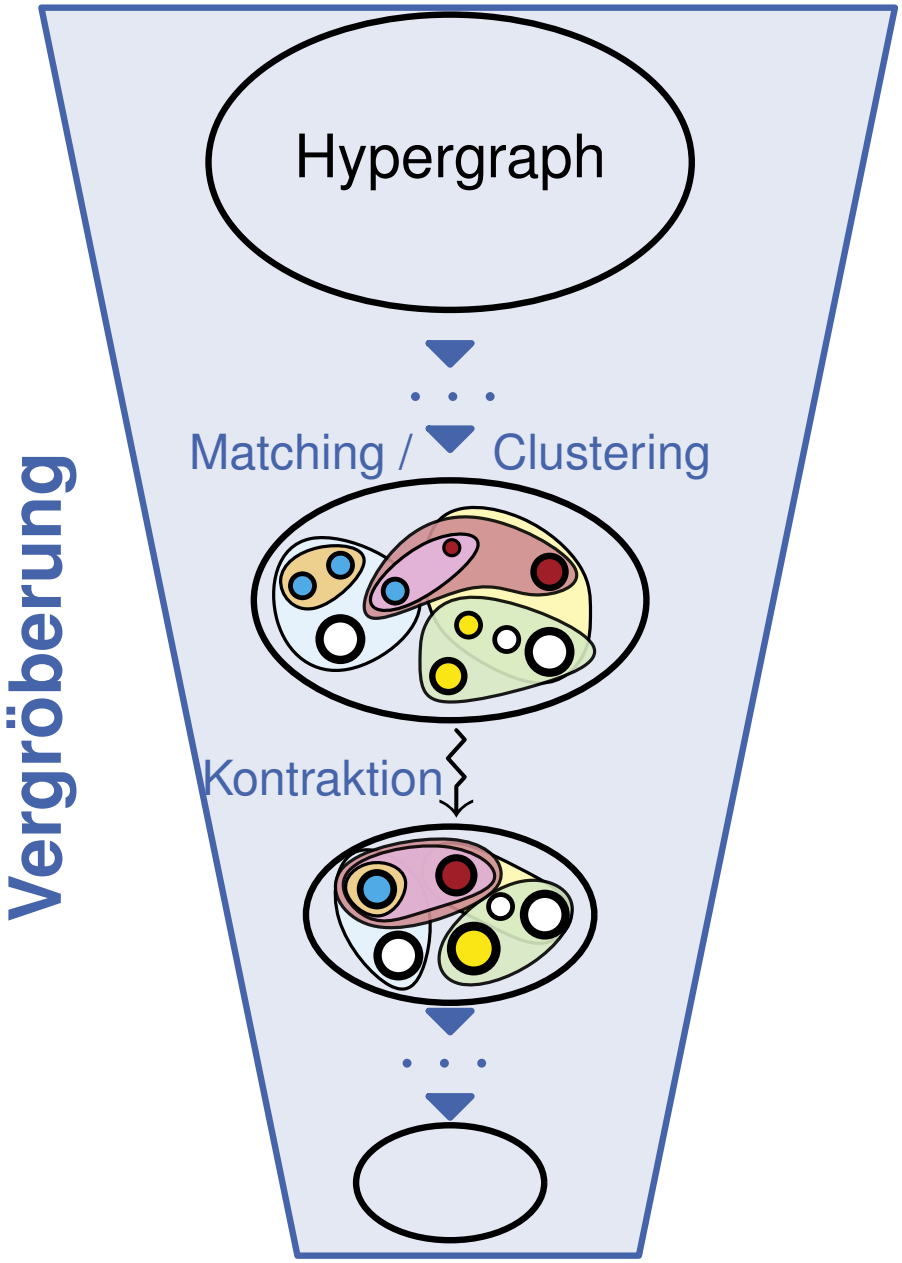
Klassifikation: Zwei-Level-Algorithmen



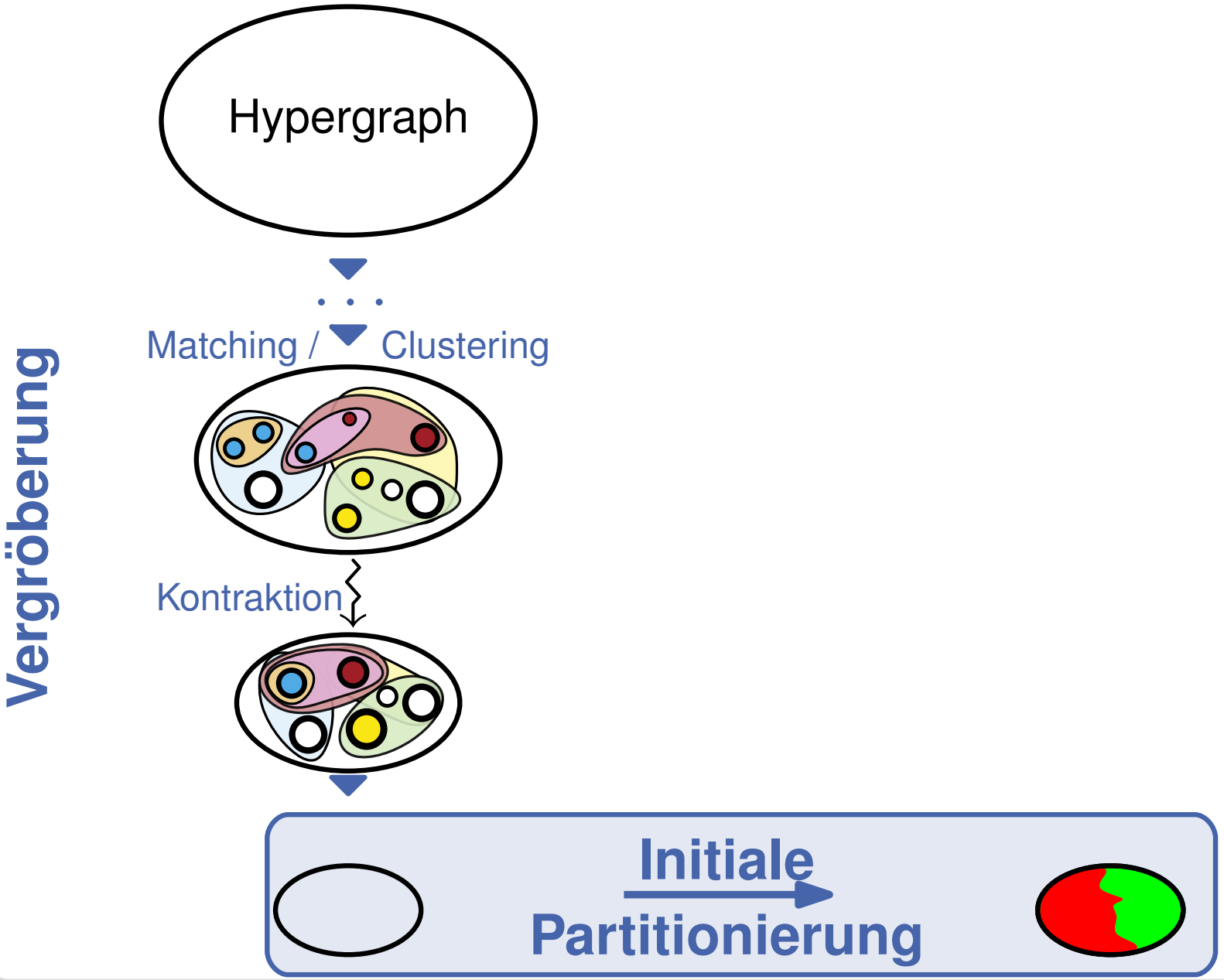
Klassifikation: Zwei-Level-Algorithmen



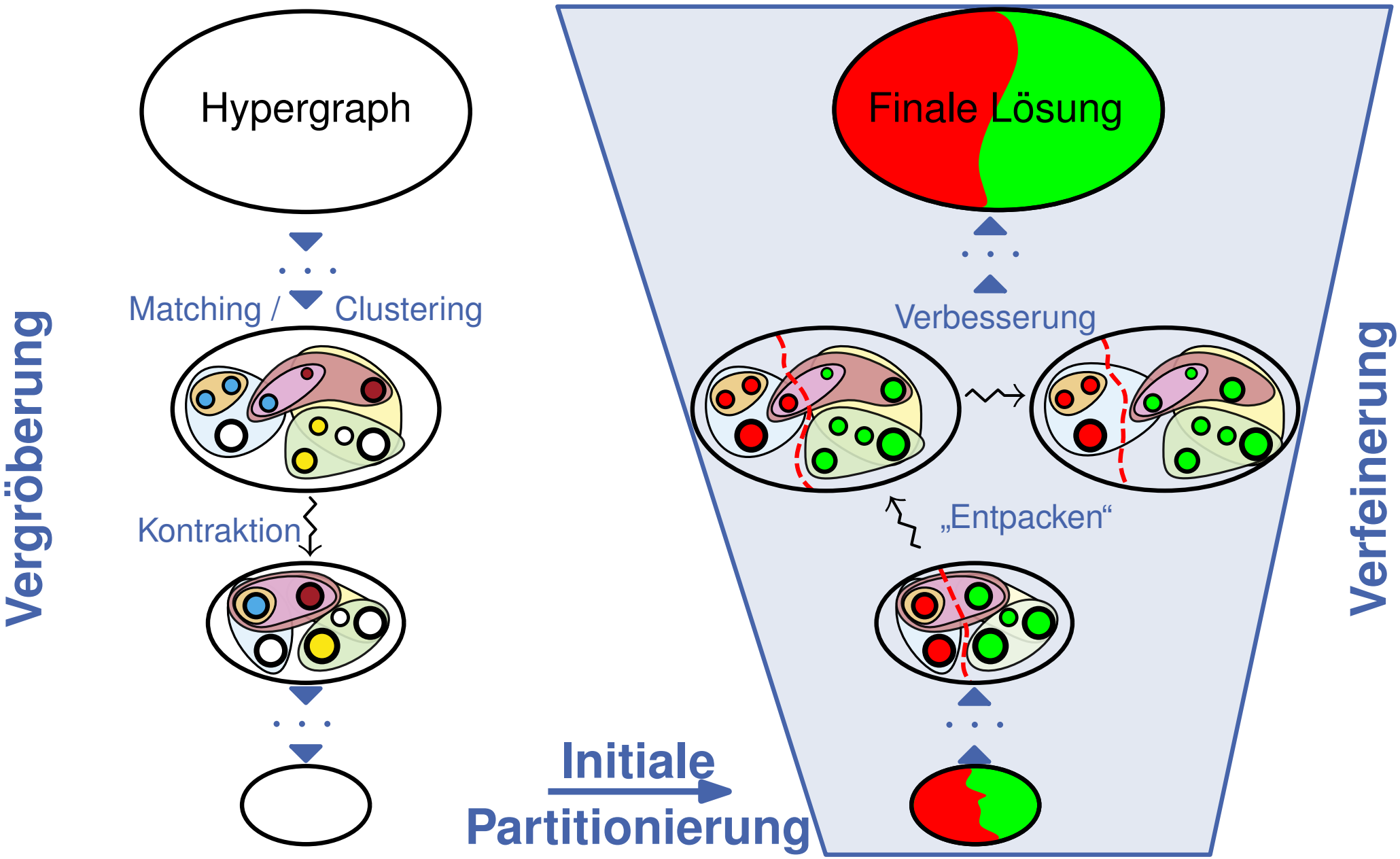
Klassifikation: Multi-Level-Algorithmen



Klassifikation: Multi-Level-Algorithmen



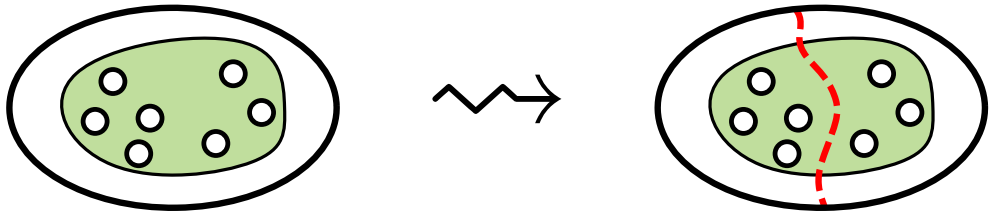
Klassifikation: Multi-Level-Algorithmen



Allgemeine Klassifikation

$k = 2$:

- Bipartitionierung:



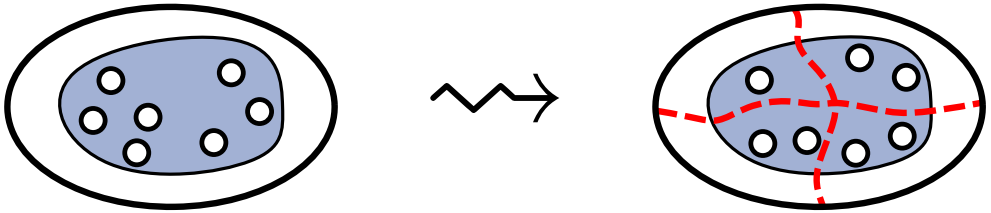
Allgemeine Klassifikation

$k = 2$:

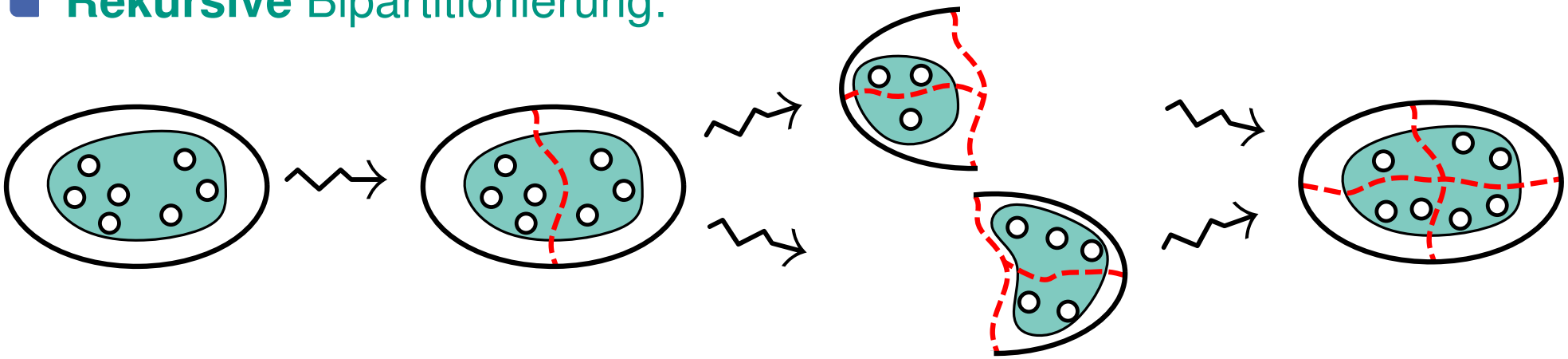
- Bipartitionierung:

$k \geq 2$:

- **Direkte** k-Wege-Partitionierung:



- **Rekursive** Bipartitionierung:



Historische Entwicklung: Ein-Level-Algorithmen

Karlsruher Institut für Technologie

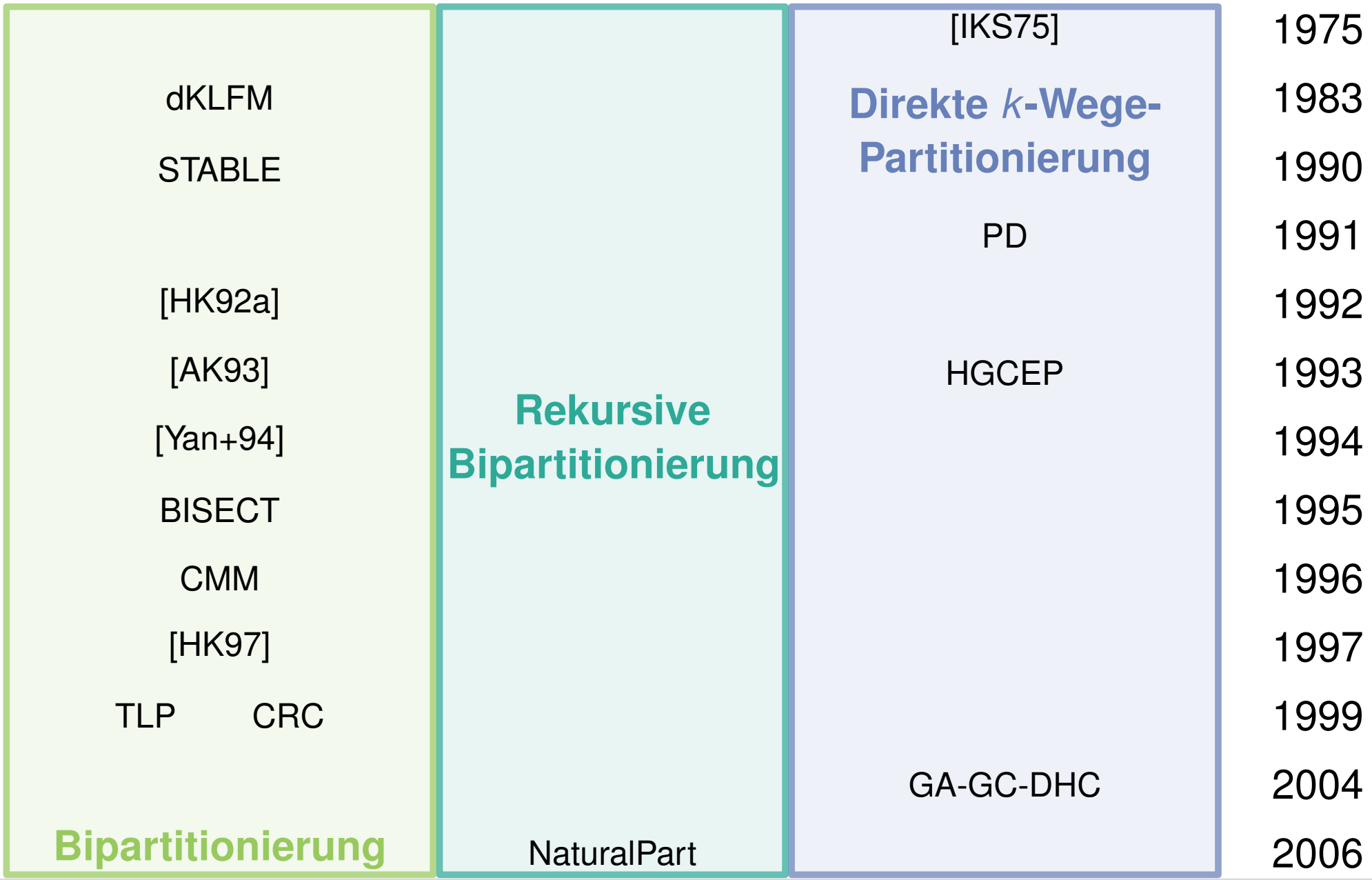
Bipartitionierung	Direkte k-Wege-Partitionierung	
KL		1972
FM		1982
[KGV83]		1983
LA_ℓ-FM		1984
	k-LA_ℓ-FM	1986
AlgI BIPART RCut	EV	1989
	MCPG	1990
WHB EIG1 EIG-IG		1991
IG-Match	NETPART	1992
	SA/EIG-TS [PP93] KP KC/AGG	1993
DLA GRCA Paraboli FBB MELO	P(L/F)M SFC Window KDualPartFM	1994
PANZA LIFO-LA_ℓ-FM	GFM	1995
LSMC PROP CLIP/CDIP		1996
LSR ASFM PROP-Rex		1997
DEEP/VAR-PROP MMP	K-PM/LR	1998
VRW Shrink-PROP	IDP [Are99]	1999
$CLIP_2$/$LIFO_2$ [CY00]	MDC-RS NGSP	2000
WalkPart		2003
LFM		2004
	Hyper-PuLP	2016
	SHP	2017
	HYPE	2018

Historische Entwicklung: Ein-Level-Algorithmen

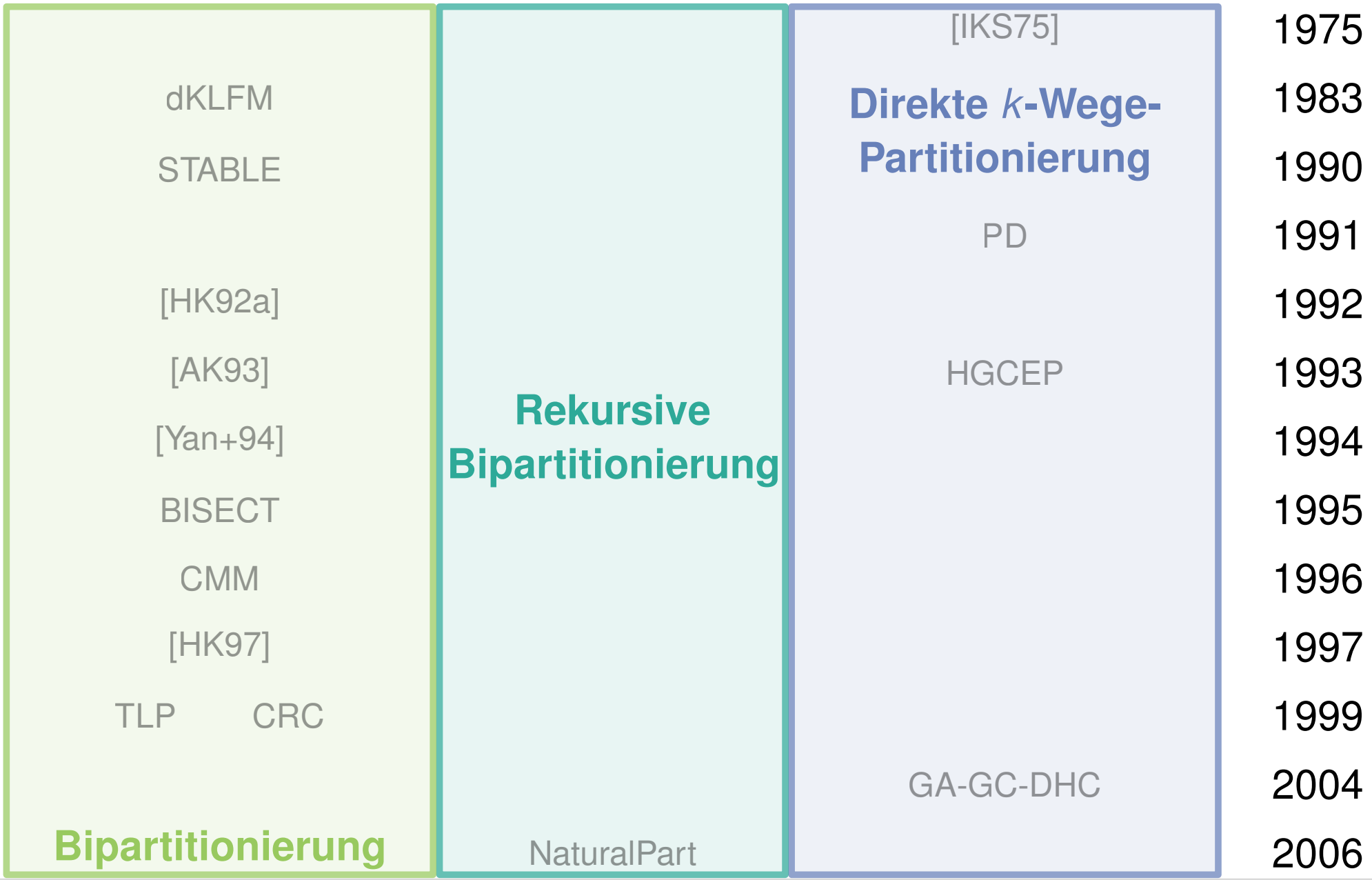
Karlsruher Institut für Technologie

Bipartitionierung	Direkte k-Wege-Partitionierung	
KL		1972
FM		1982
[KGV83]		1983
LA$_\ell$-FM		1984
	k-LA$_\ell$-FM	1986
AlgI BIPART RCut	EV	1989
	MCPG	1990
WHB EIG1 EIG-IG		1991
IG-Match	NETPART	1992
	SA/EIG-TS [PP93] KP KC/AGG	1993
DLA GRCA Paraboli FBB MELO	P(L/F)M SFC Window KDualPartFM	1994
PANZA LIFO-LA$_\ell$-FM	GFM	1995
LSMC PROP CLIP/CDIP		1996
LSR ASFM PROP-Rex		1997
DEEP/VAR-PROP MMP	K-PM/LR	1998
VRW Shrink-PROP	IDP [Are99]	1999
CLIP$_2$/LIFO$_2$ [CY00]	MDC-RS NGSP	2000
WalkPart		2003
LFM		2004
	Hyper-PuLP	2016
	SHP	2017
	HYPE	2018

Historische Entwicklung: Zwei-Level-Algorithmen



Historische Entwicklung: Zwei-Level-Algorithmen

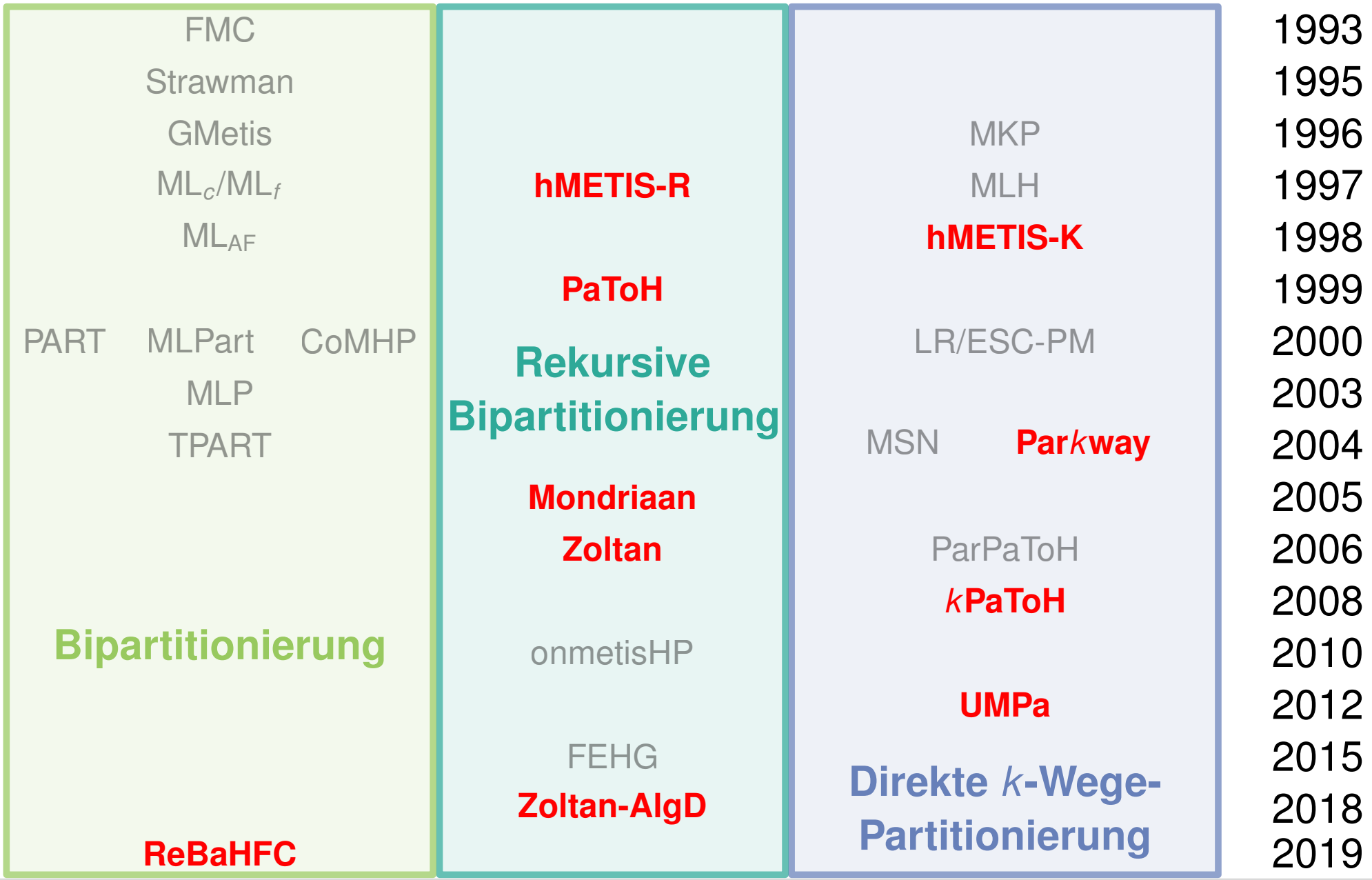


Historische Entwicklung: Multi-Level-Algorithmen

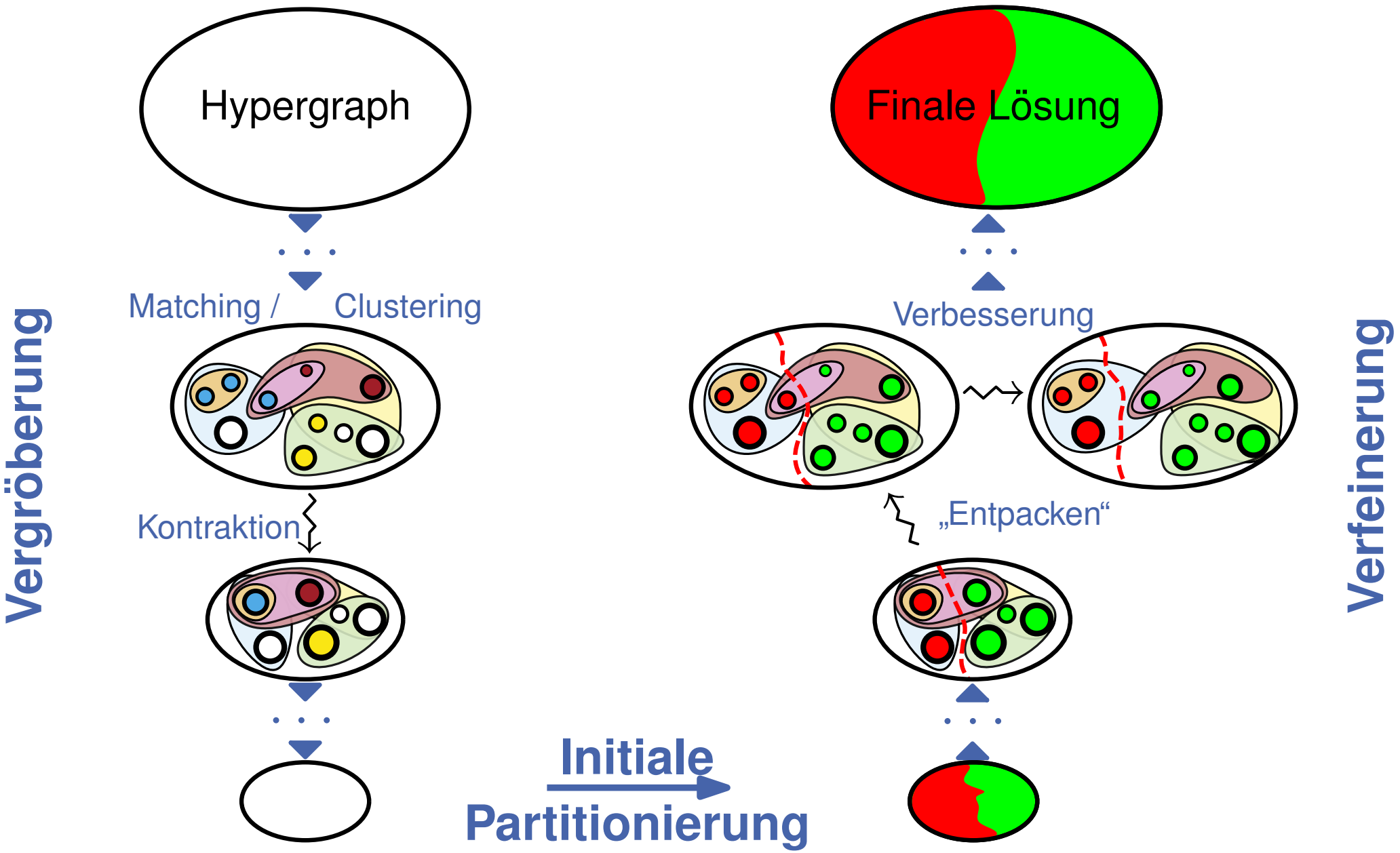
Karlsruher Institut für Technologie

Bipartitionierung	Rekursive Bipartitionierung	Direkte k-Wege-Partitionierung	
FMC			1993
Strawman			1995
GMetis		MKP	1996
ML_c/ML_f	hMETIS-R	MLH	1997
ML_{AF}		hMETIS-K	1998
	PaToH		1999
PART MLPart CoMHP		LR/ESC-PM	2000
MLP			2003
TPART		MSN Par*k*way	2004
	Mondriaan		2005
	Zoltan	ParPaToH	2006
		*k*PaToH	2008
	onmetisHP		2010
		UMPa	2012
	FEHG		2015
	Zoltan-AlgD		2018
ReBaHFC			2019

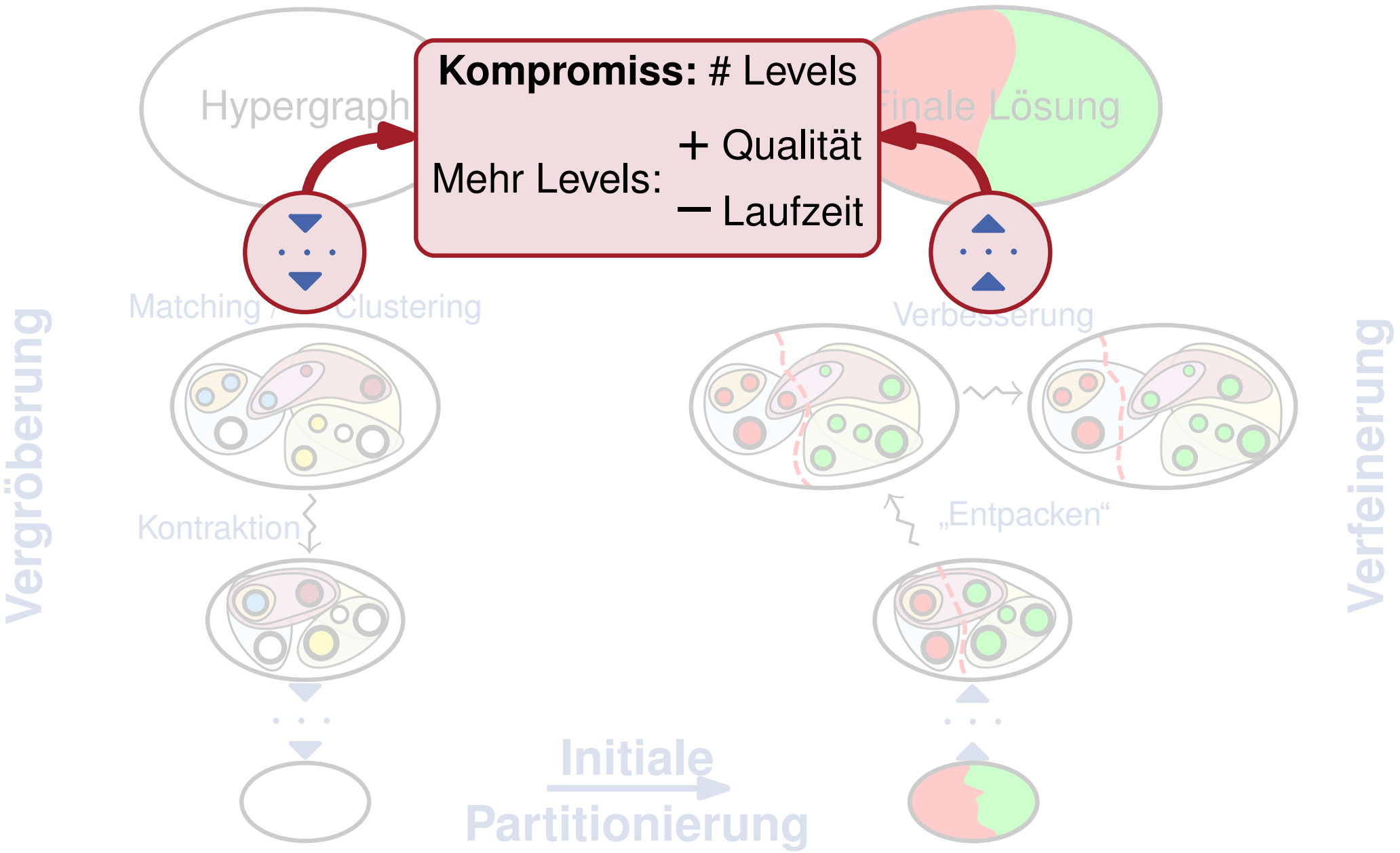
Historische Entwicklung: Multi-Level-Algorithmen



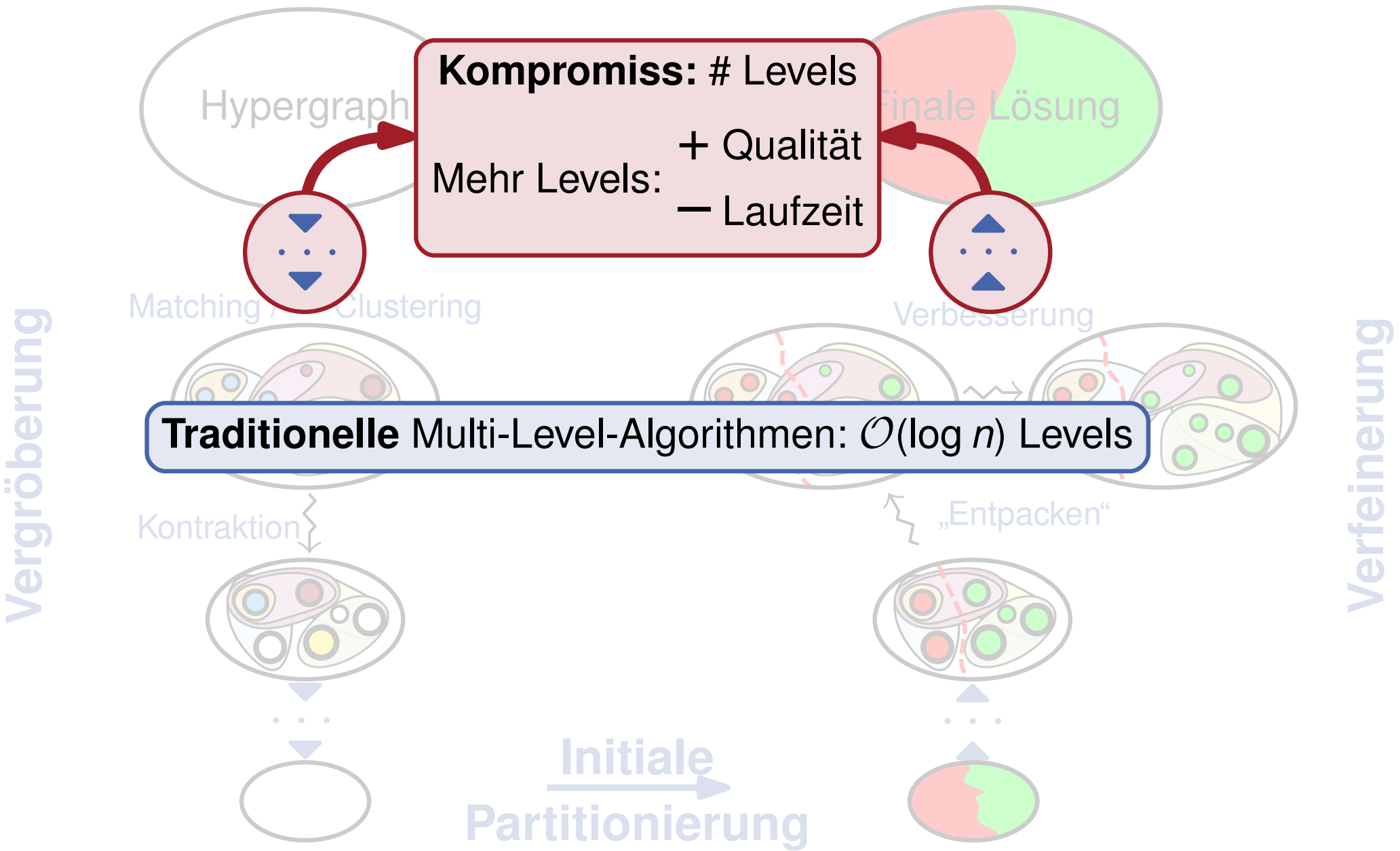
Warum noch ein Multi-Level-Algorithmus?



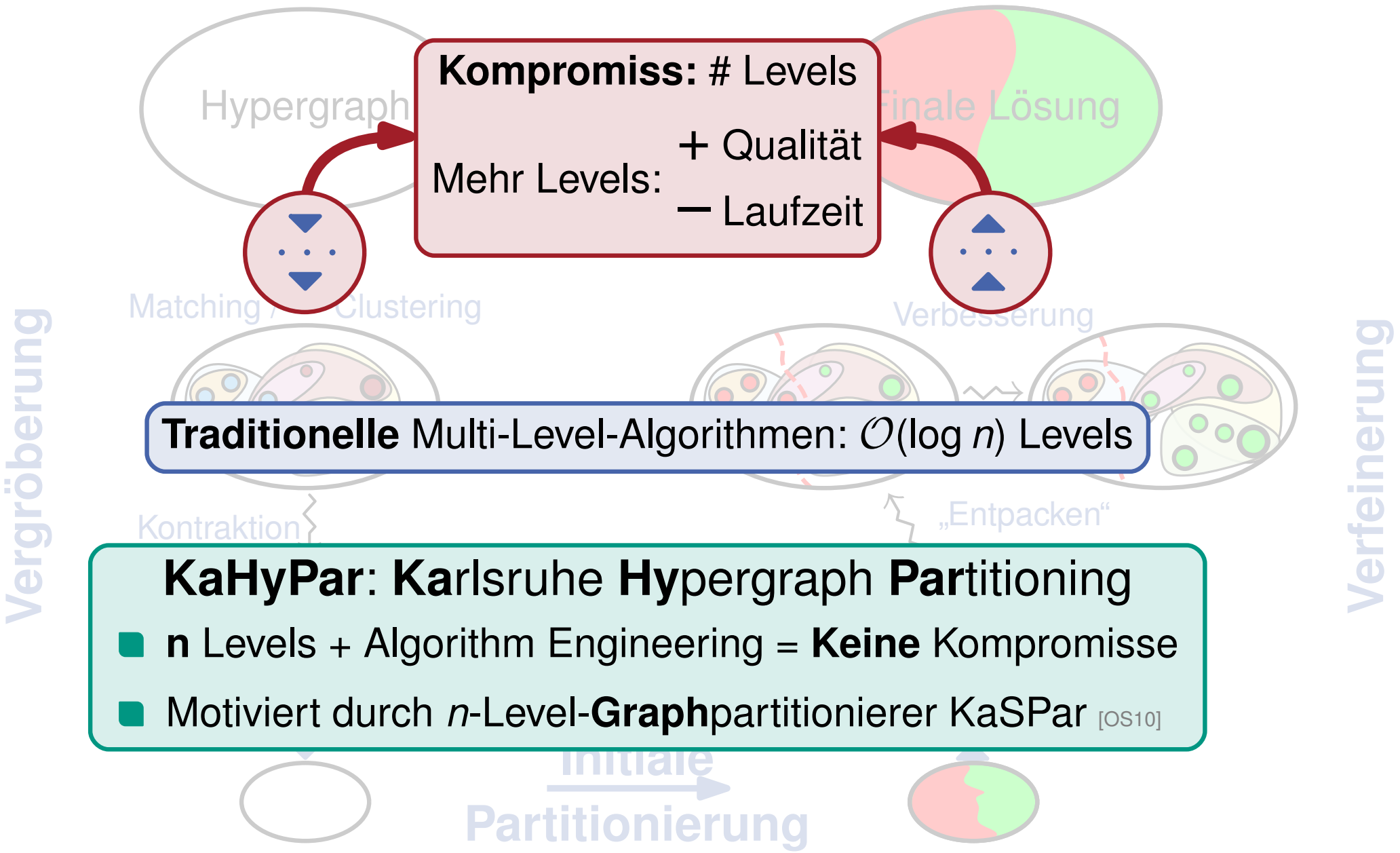
Warum noch ein Multi-Level-Algorithmus?



Warum noch ein Multi-Level-Algorithmus?



Warum noch ein Multi-Level-Algorithmus?



KaHyPar: **Ka**rlsruhe **Hy**pergraph **Par**titioning

- **n** Levels + Algorithm Engineering = **Keine** Kompromisse
- Motiviert durch *n*-Level-**Graph**partitionierer KaSPar [OS10]

Ein kleiner Vorgeschmack: Lösungsqualität

Karlsruher Institut für Technologie

KaHyPar
[SHH+16; AHS+17; HS17; HSS18; ASS18; HSS19]

Zoltan-AlgD
[SS18; SSC19]

hMETIS-R
[Kar+97; Kar+99]

hMETIS-K
[KK99; KK00]

HYPE
[May+18]

ReBaHFC
[GHW19]

PaToH
[ÇA99]

UMPa
[Çat+12; Çat+15]

SHP
[Kab+17]

Mondriaan
[VB05]

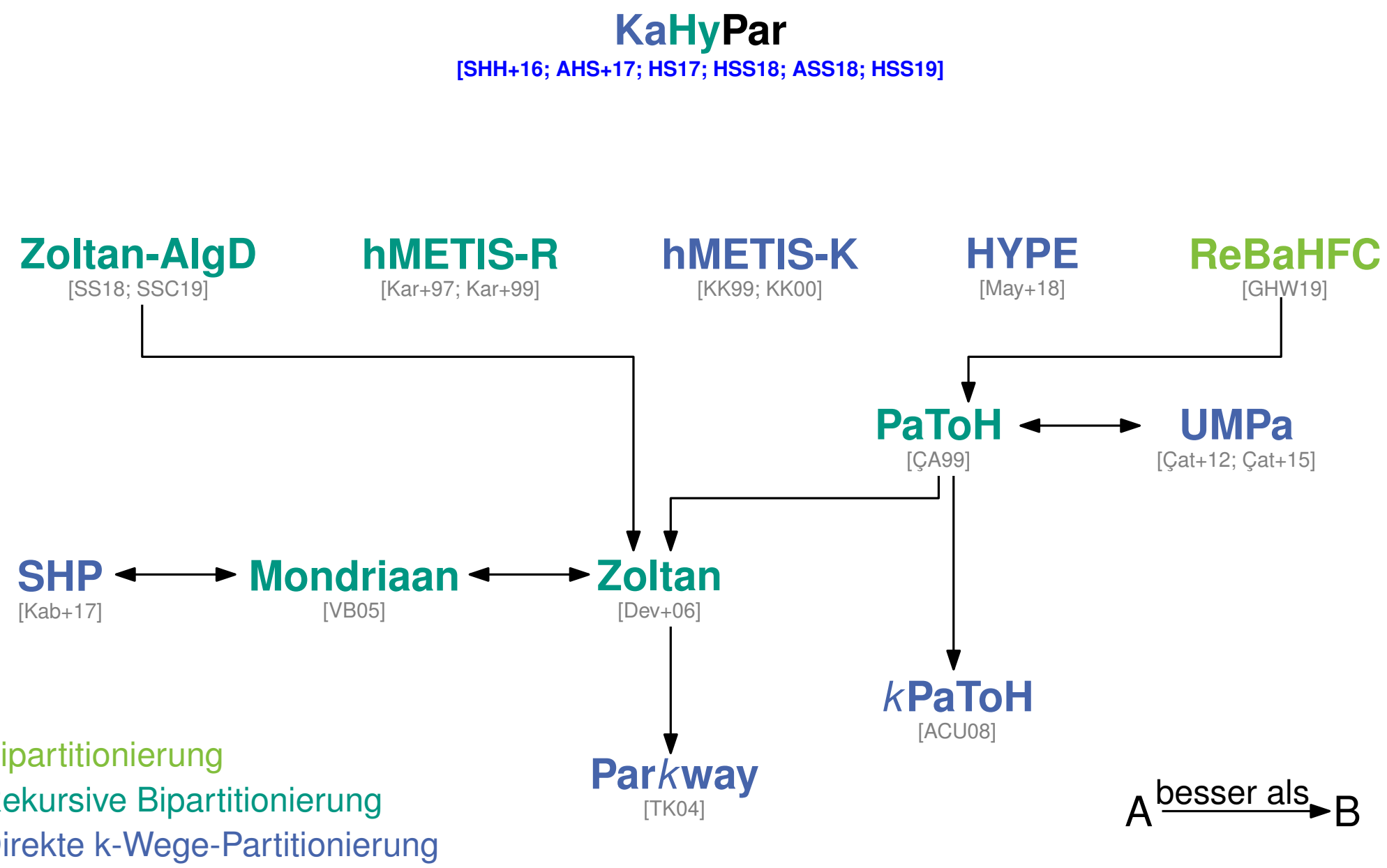
Zoltan
[Dev+06]

***k*PaToH**
[ACU08]

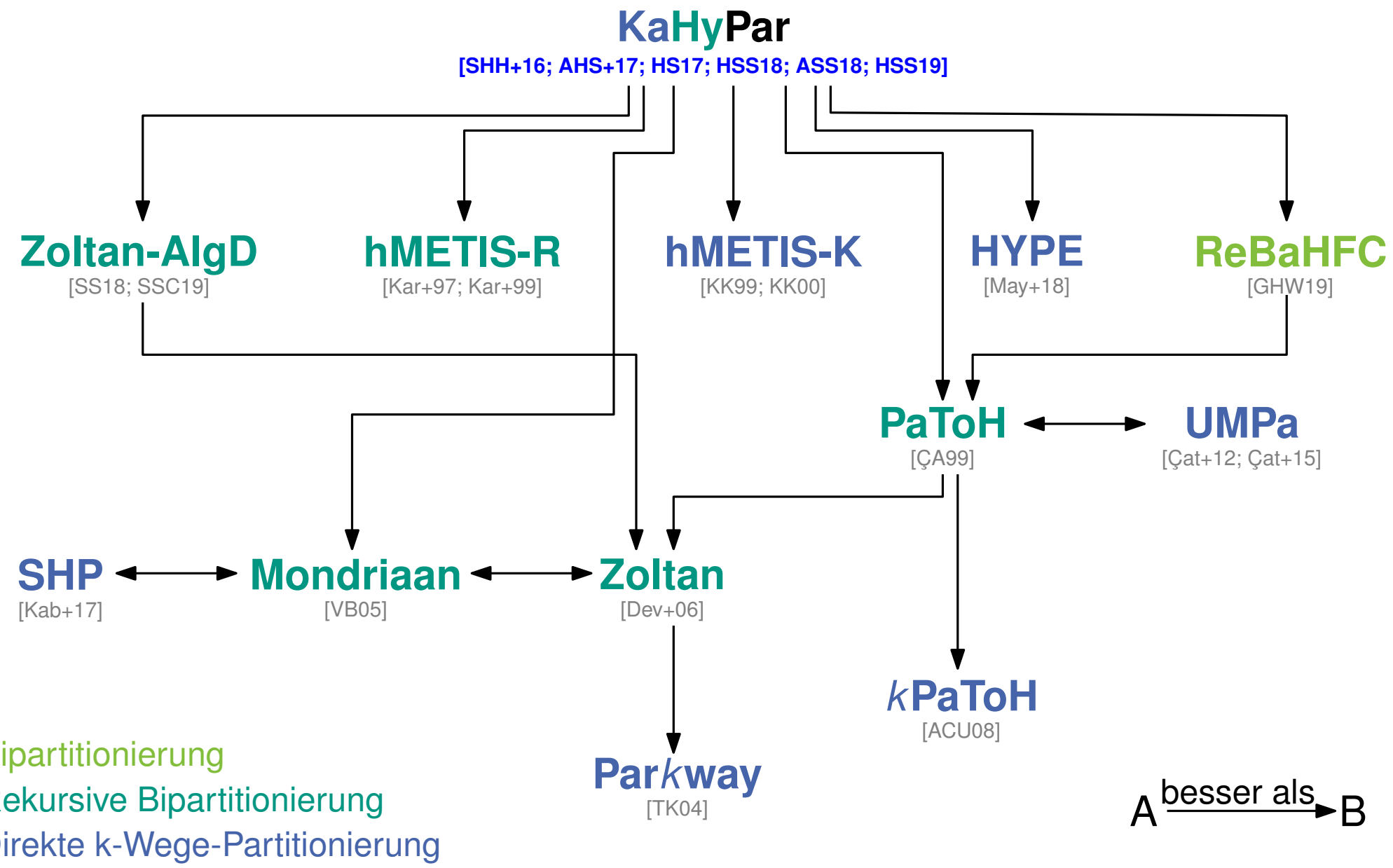
Par*k*way
[TK04]

Bipartitionierung
Rekursive Bipartitionierung
Direkte k-Wege-Partitionierung

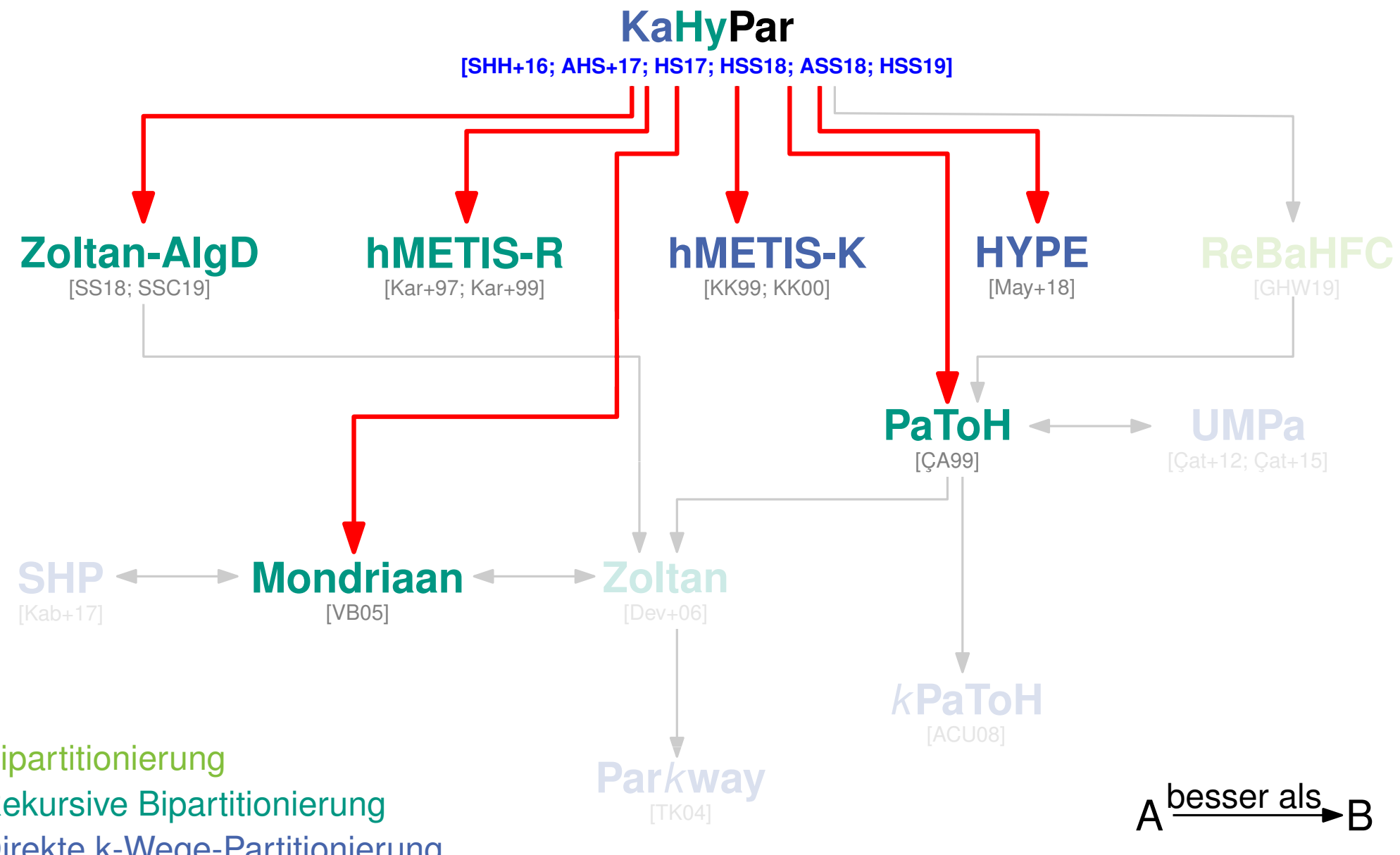
Ein kleiner Vorgeschmack: Lösungsqualität



Ein kleiner Vorgeschmack: Lösungsqualität

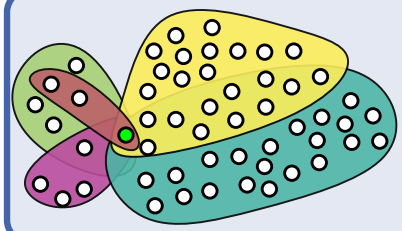


Ein kleiner Vorgeschmack: Lösungsqualität



Übersicht: Algorithmische Kernbestandteile

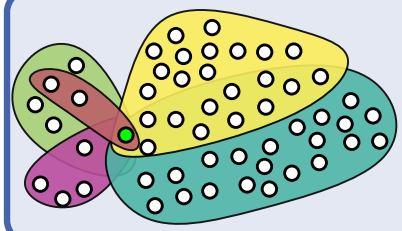
Übersicht: Algorithmische Kernbestandteile



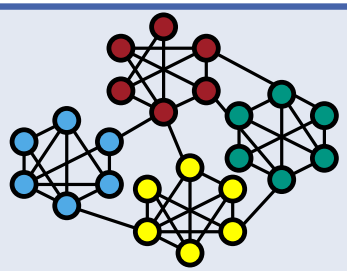
Pin-Ausdünnung

[AHS+17]

Übersicht: Algorithmische Kernbestandteile



Pin-Ausdünnung
[AHS+17]



Strukturerhaltende Vergröberung
[HS17]

Verbesserung

„Entpacken“

Vergröberung

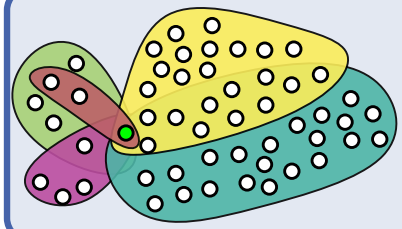
Kontraktion

Initiale Partitionierung

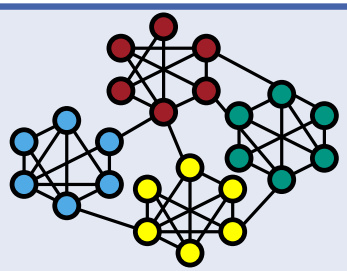
Verfeinerung

Institut für Theoretische Informatik
Lehrstuhl Algorithmik

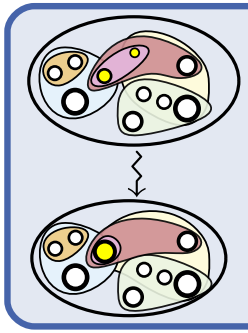
Übersicht: Algorithmische Kernbestandteile



Pin-Ausdünnung
[AHS+17]



Strukturerhaltende Vergröberung
[HS17]



Schnelle *n*-Level-Vergröberung
[SHH+16; AHS+17]

Verbesserung

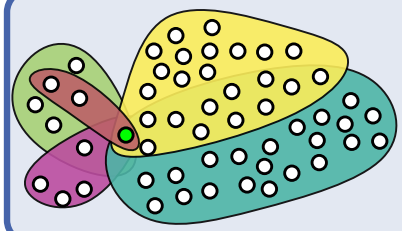
„Entpacken“

Initiale Partitionierung

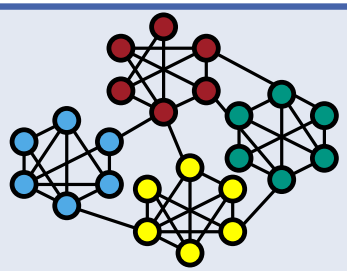
Vergröberung

Verfeinerung

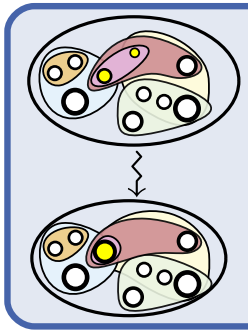
Übersicht: Algorithmische Kernbestandteile



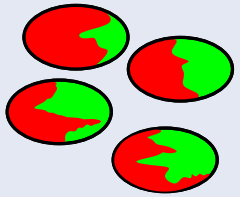
Pin-Ausdünnung
[AHS+17]



Strukturerhaltende Vergröberung
[HS17]



Schnelle *n*-Level-Vergröberung
[SHH+16; AHS+17]



Portfolio-basierte initiale Partitionierung
[SHH+16]

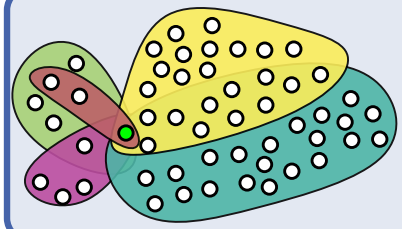
Verbesserung

„Entpacken“

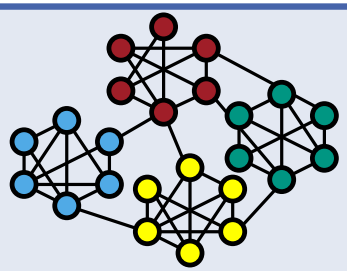
Vergröberung

Verfeinerung

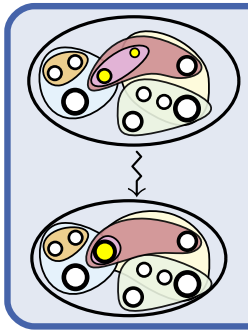
Übersicht: Algorithmische Kernbestandteile



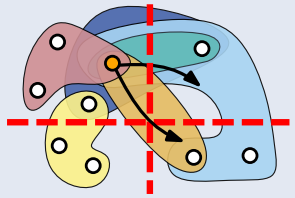
Pin-Ausdünnung
[AHS+17]



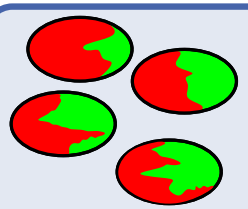
Strukturerhaltende Vergröberung
[HS17]



Schnelle *n*-Level-Vergröberung
[SHH+16; AHS+17]



„Engineerte“ 2-Wege & **erste** effiziente *k*-Wege lokale Suche
[SHH+16; AHS+17]



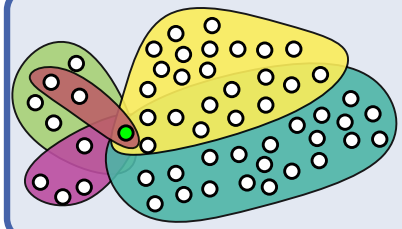
Portfolio-basierte initiale Partitionierung
[SHH+16]

Verbesserung

Vergröberung

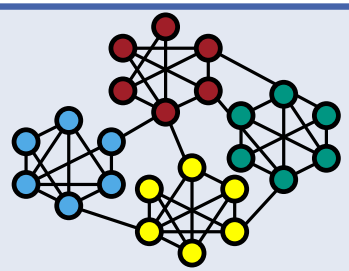
Verfeinerung

Übersicht: Algorithmische Kernbestandteile

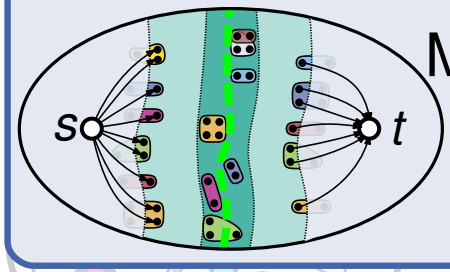


Pin-Ausdünnung
[AHS+17]

Finale Lösung

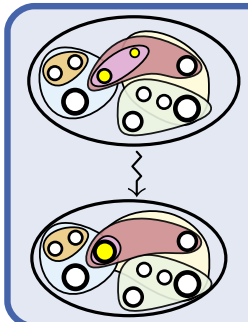


Strukturerhaltende Vergröberung
[HS17]

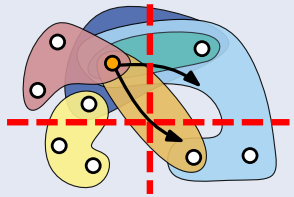


Max-Flow Min-Cut Verbesserung
[HSS18; HSS19]

Vergröberung

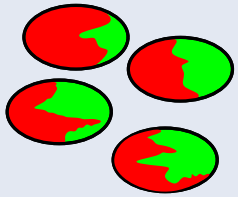


Schnelle *n*-Level-Vergröberung
[SHH+16; AHS+17]



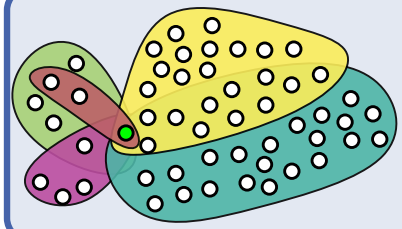
„Engineerte“ 2-Wege & **erste** effiziente *k*-Wege lokale Suche
[SHH+16; AHS+17]

Verfeinerung



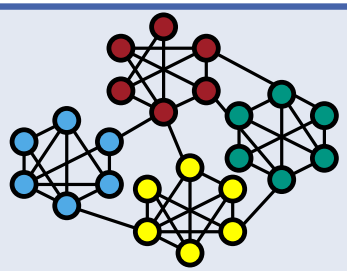
Portfolio-basierte initiale Partitionierung
[SHH+16]

Übersicht: Algorithmische Kernbestandteile

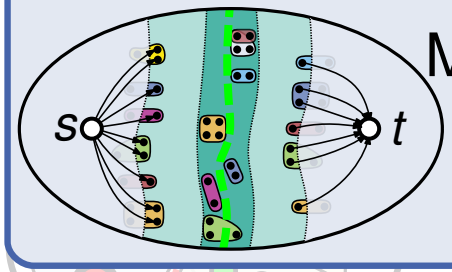


Pin-Ausdünnung
[AHS+17]

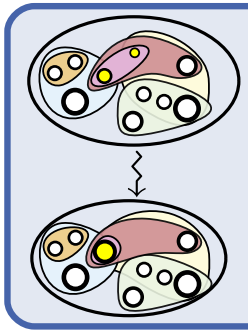
Evolutionärer n-Level-Algorithmus
[ASS18]



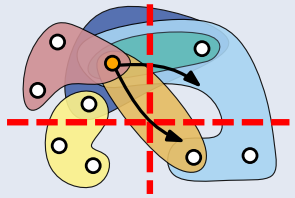
Strukturerhaltende Vergröberung
[HS17]



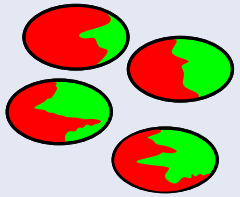
Max-Flow Min-Cut Verbesserung
[HSS18; HSS19]



Schnelle *n*-Level-Vergröberung
[SHH+16; AHS+17]



„Engineerte“ 2-Wege & **erste** effiziente *k*-Wege lokale Suche
[SHH+16; AHS+17]



Portfolio-basierte initiale Partitionierung
[SHH+16]

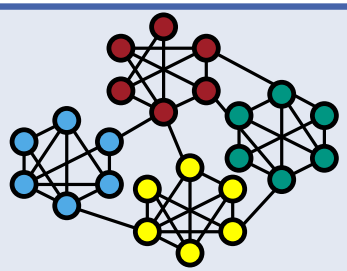
Vergröberung

Verfeinerung

Übersicht: Algorithmische Kernbestandteile

Pin-Ausdünnung
[AHS+17]

Evolutionärer n-Level-Algorithmus
[ASS18]



Strukturerhaltende Vergröberung
[HS17]

Max-Flow Min-Cut Verbesserung
[HSS18; HSS19]

Vergröberung

Verfeinerung

Schnelle *n*-Level-Vergröberung
[SHH+16; AHS+17]

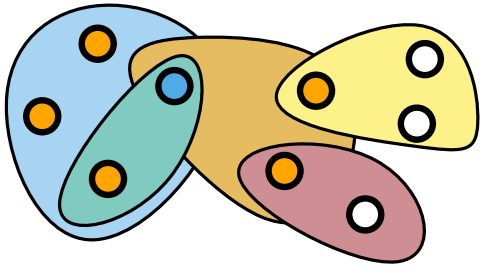
„Engineerte“ 2-Wege & **erste** effiziente *k*-Wege lokale Suche
[SHH+16; AHS+17]

Portfolio-basierte initiale Partitionierung
[SHH+16]

Vergröberung von Hypergraphen

Vergröberungsalgorithmus (Skizze):

Für alle *Knoten* v **:**
  cluster[v] := argmax `Bewertung` $r(v,u)$
  Nachbar u
`kontrahiere` cluster



Gängige Strategie:
Vermeide globale Entscheidungen $\rightsquigarrow$ **lokale** Greedy-Algorithmen

Haupt-Designziele: [KK99]

1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen

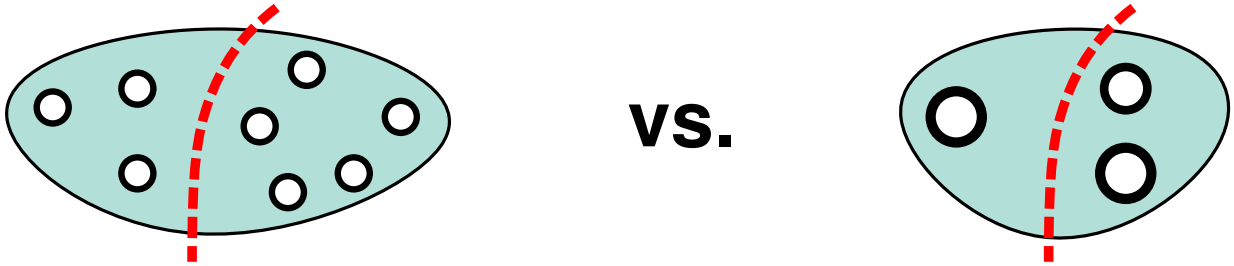
2: Anzahl Netze **reduzieren** $\rightsquigarrow$ einfachere initiale Partitionierung

3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

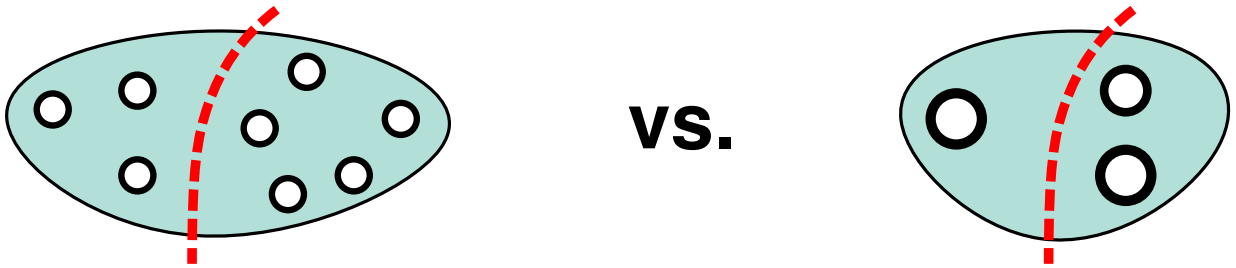
1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen



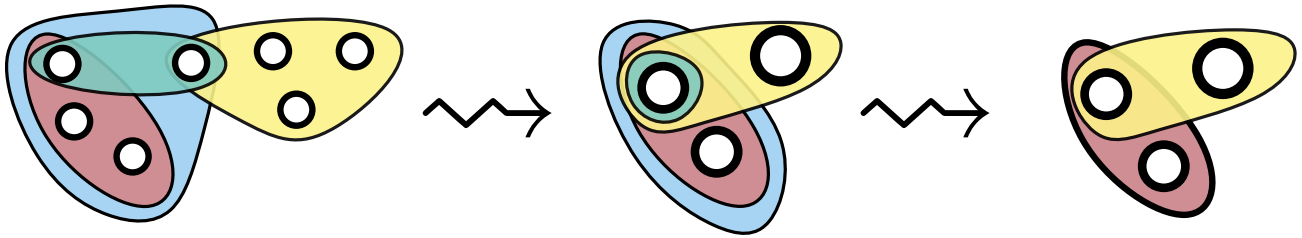
Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen



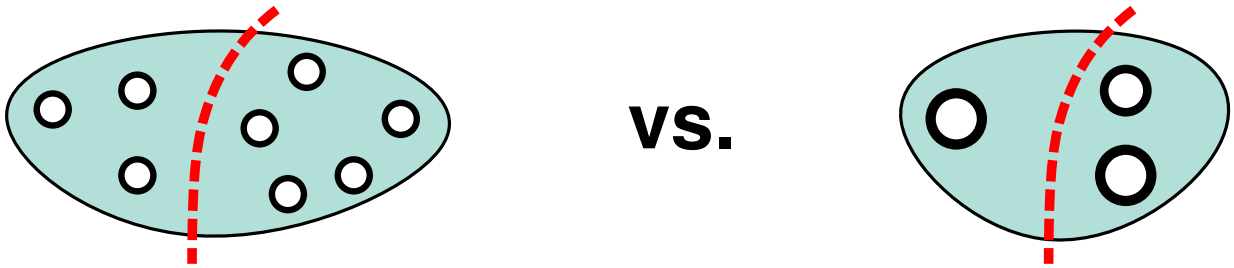
2: **Anzahl** Netze reduzieren $\rightsquigarrow$ einfachere initiale Partitionierung



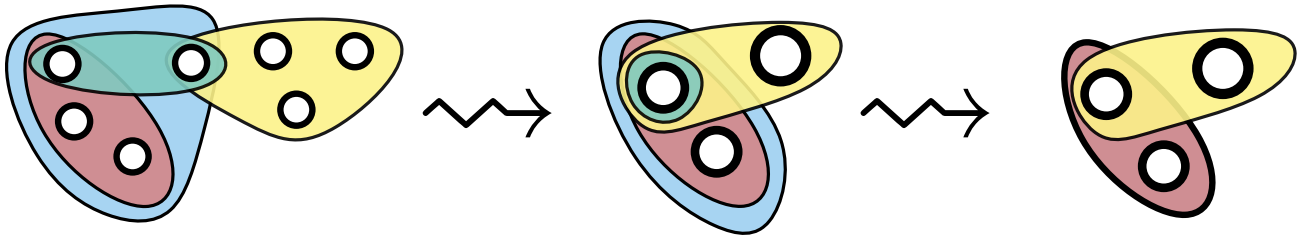
Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen



2: **Anzahl** Netze reduzieren $\rightsquigarrow$ einfachere initiale Partitionierung



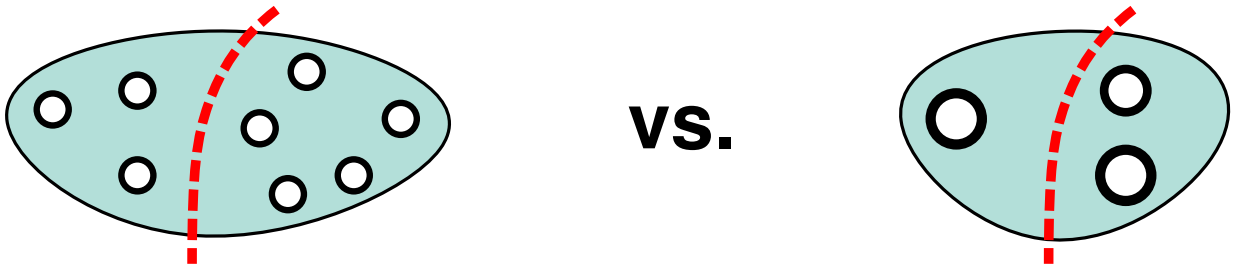
$\Longrightarrow$ Hypergraph-spezifische **Bewertungsfunktionen:**

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

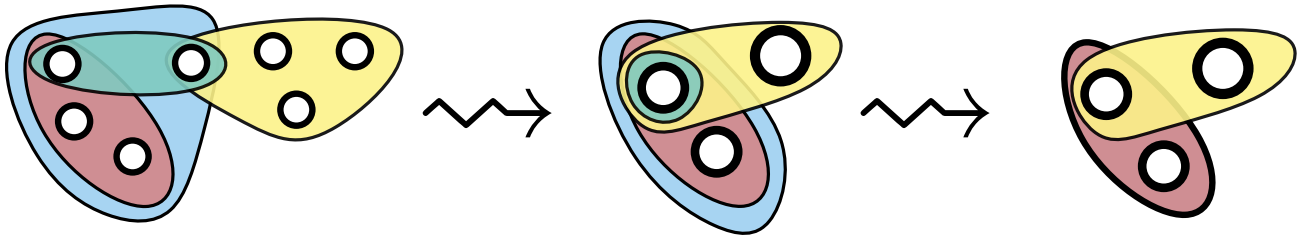
Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen



2: **Anzahl** Netze reduzieren $\rightsquigarrow$ einfachere initiale Partitionierung



$\Longrightarrow$ Hypergraph-spezifische **Bewertungsfunktionen:**

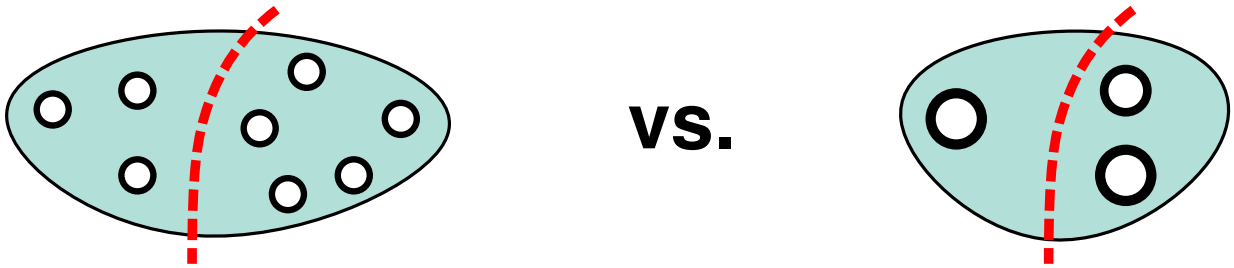
$$r(v,u) := \sum_{\substack{\text{Netz } e: \\ \{v,u\} \in e}} \frac{\omega(e)}{|e|-1}$$

Viele ...

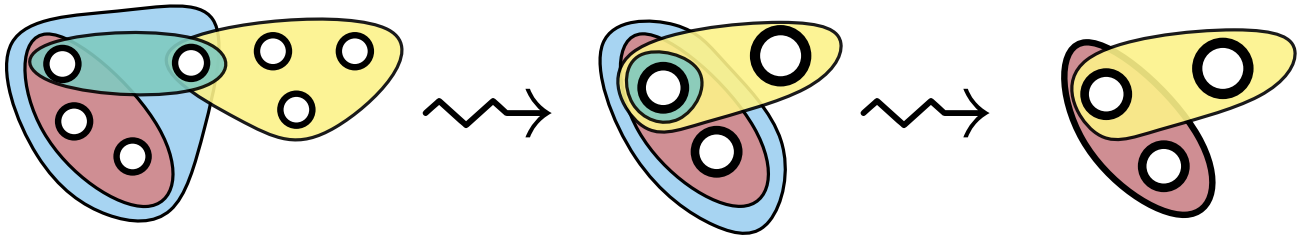
Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen



2: **Anzahl** Netze reduzieren $\rightsquigarrow$ einfachere initiale Partitionierung



$\Longrightarrow$ Hypergraph-spezifische **Bewertungsfunktionen:**

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

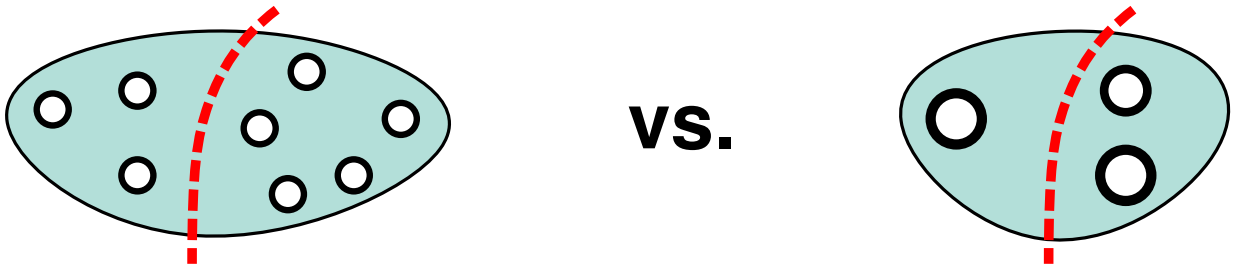
... schwere ...

Viele ...

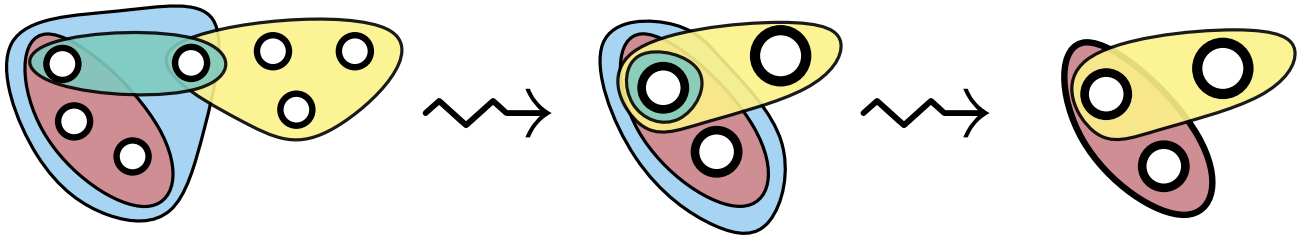
Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen



2: **Anzahl** Netze reduzieren $\rightsquigarrow$ einfachere initiale Partitionierung



$\Longrightarrow$ Hypergraph-spezifische **Bewertungsfunktionen:**

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

... schwere ...

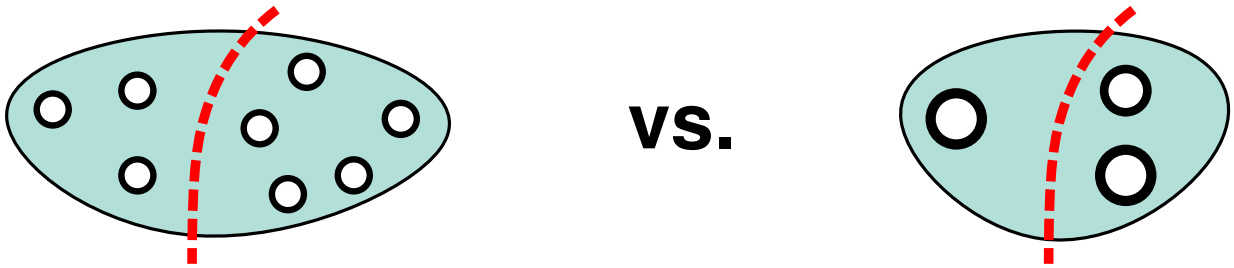
... kleine Hyperkanten

Viele ...

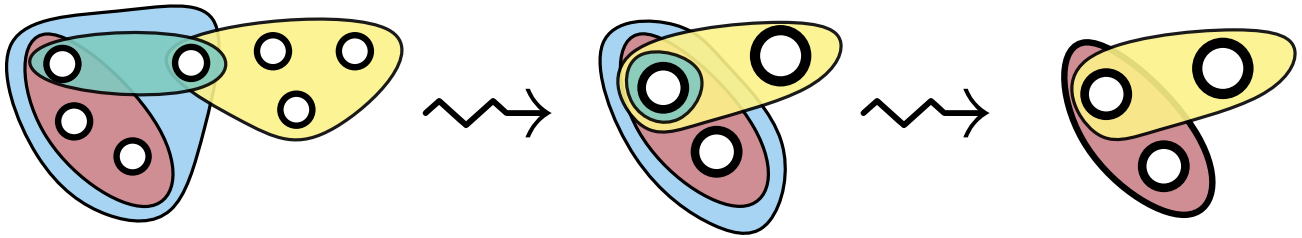
Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

1: Netze **verkleinern** $\rightsquigarrow$ einfachere lokale Suchen ✓



2: **Anzahl** Netze reduzieren $\rightsquigarrow$ einfachere initiale Partitionierung ✓



$\Longrightarrow$ Hypergraph-spezifische **Bewertungsfunktionen:**

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

... schwere ...

... kleine Hyperkanten

Viele ...

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

genug?

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

genug?

„Was soll schon **schief** gehen?“

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

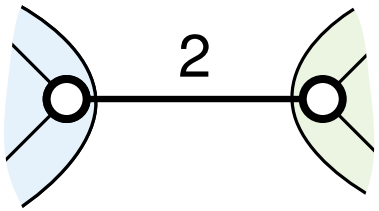
3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

genug?

„Was soll schon **schief** gehen?“



Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

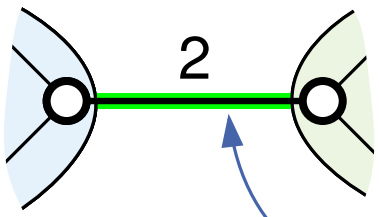
3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

genug?

„Was soll schon **schief** gehen?“



starke Verbindung

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

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

genug?

„Was soll schon **schief** gehen?“

Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

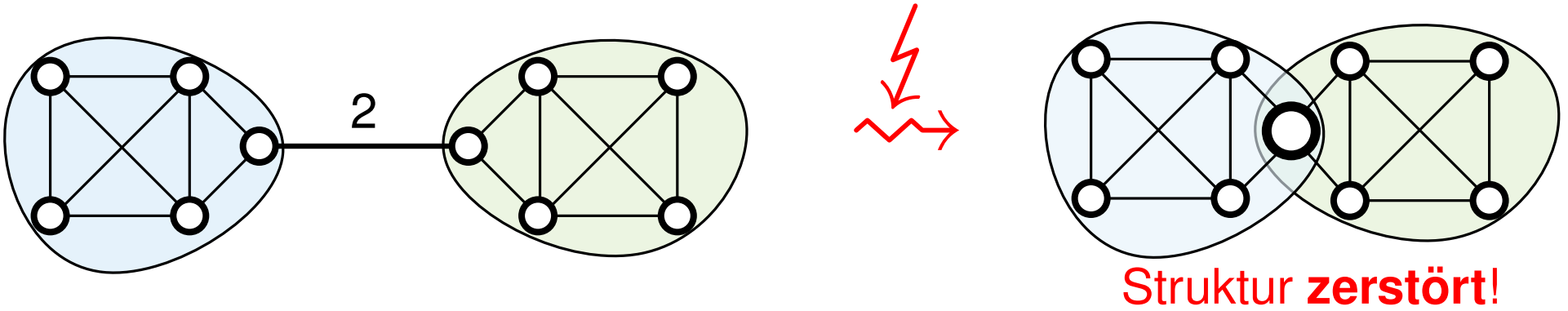
3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

genug?

„Was soll schon **schief** gehen?“



Vergröberung von Hypergraphen

Haupt-Designziele: [KK99]

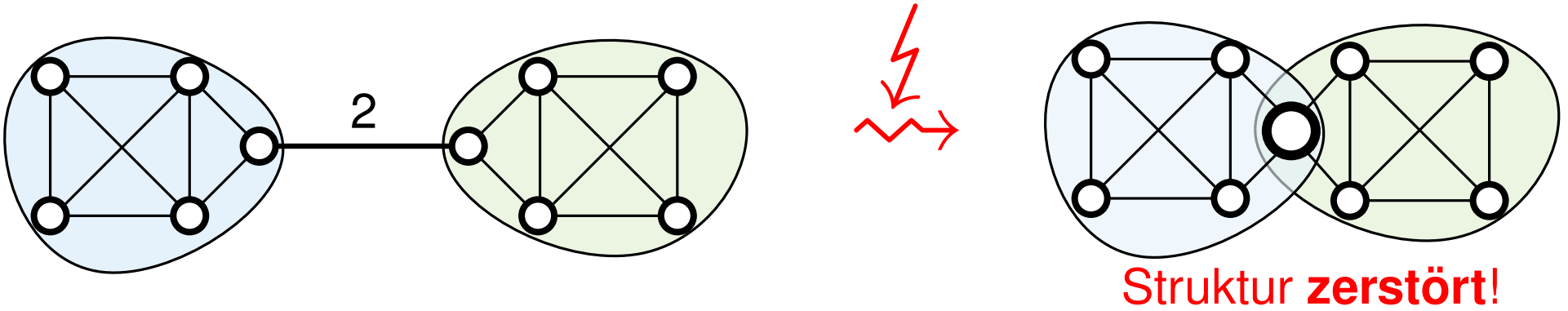
3: Struktur **erhalten** $\rightsquigarrow$ gute Lösungen auf gröbstem Level

$\Rightarrow$ Keine Beschränkung auf Matchings

$\Rightarrow$ Ungefähr balancierte Knotengewichte

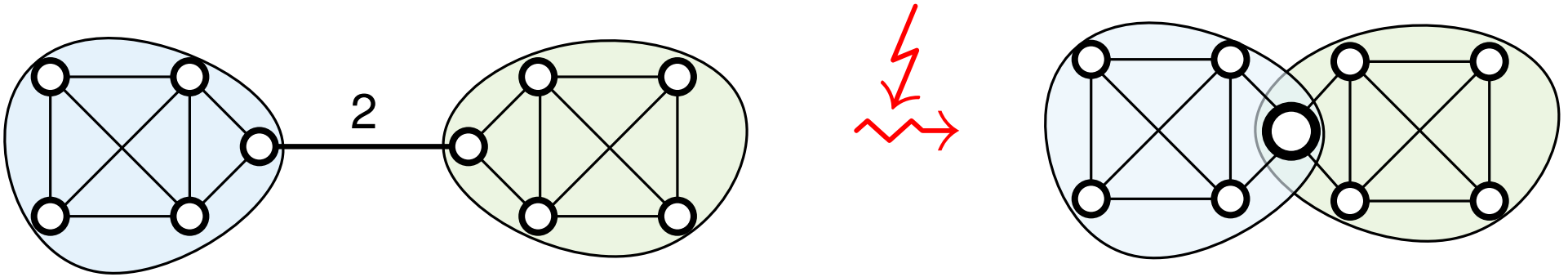
genug?

„Was soll schon **schief** gehen?“

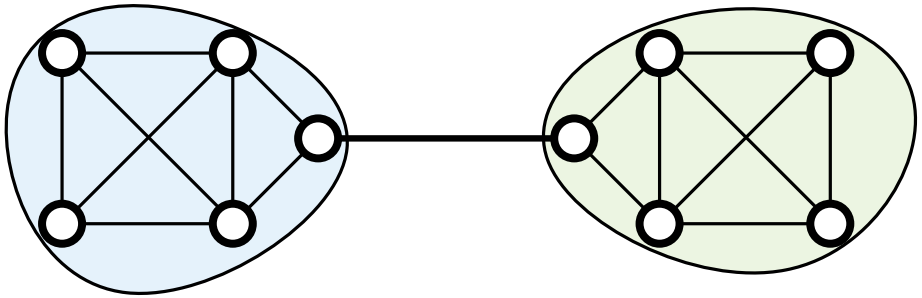


$\Rightarrow$ **Problem: Beschränkung** auf **lokale** Information!

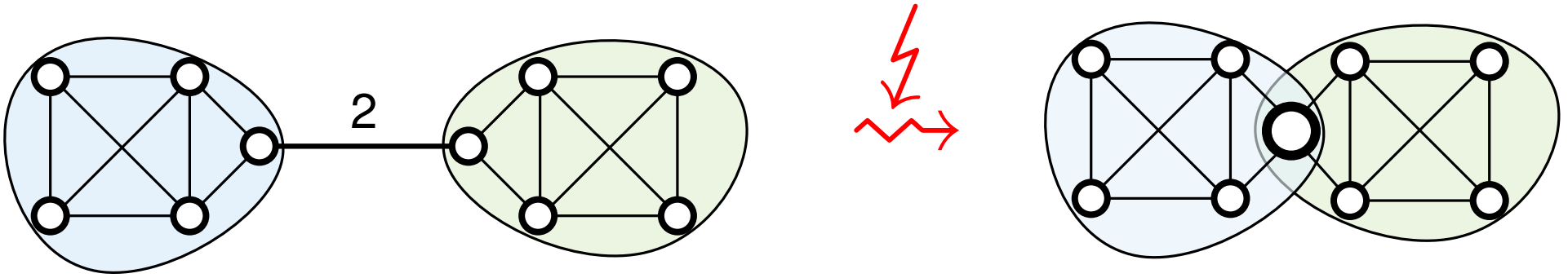
Strukturerhaltende Vergröberung [HS17]



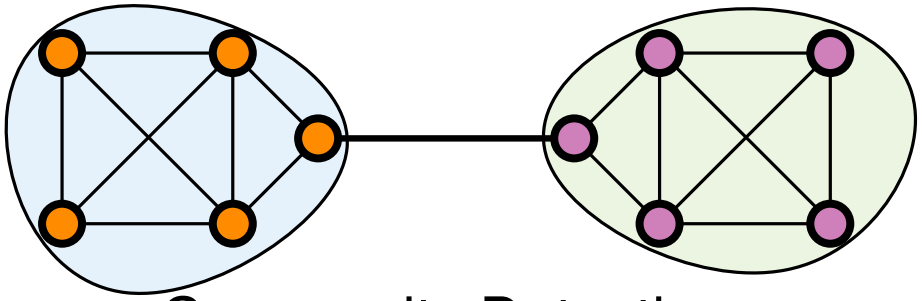
Lösung:



Strukturerhaltende Vergröberung [HS17]



Lösung:

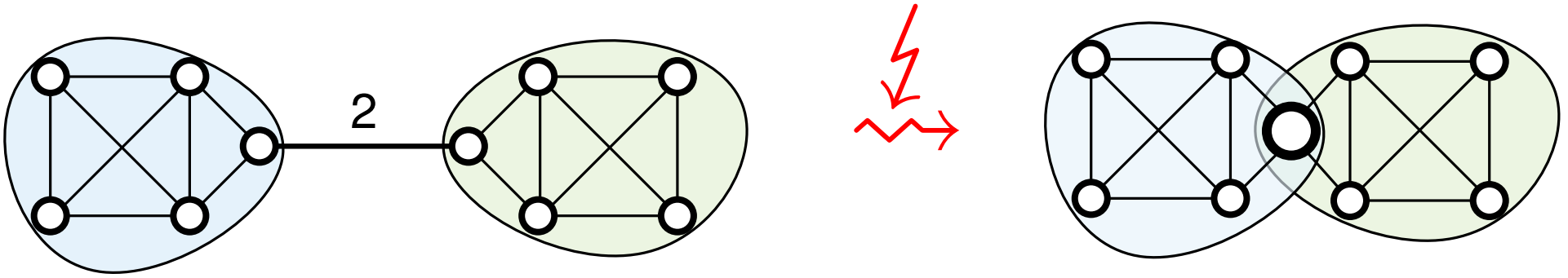


Community Detection

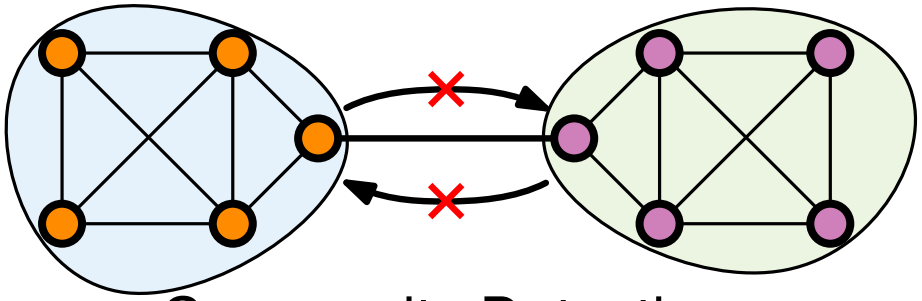
Framework:

- Vorverarbeitung: **Communitystruktur** bestimmen

Strukturerhaltende Vergröberung [HS17]



Lösung:

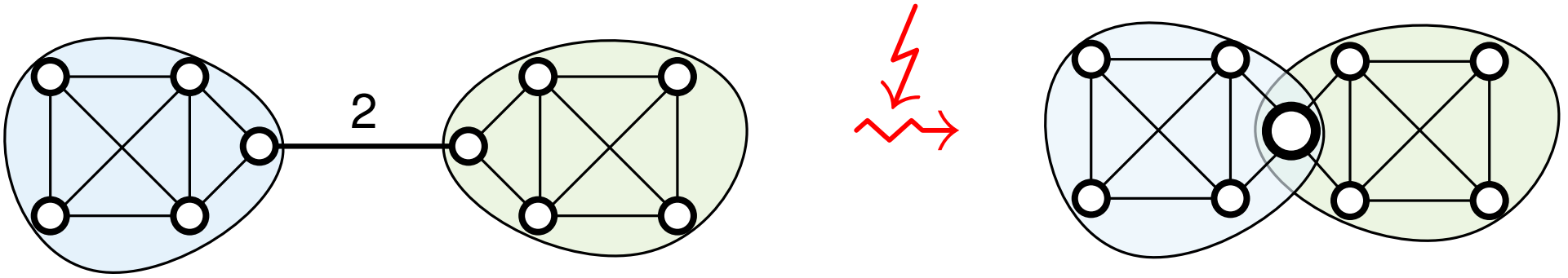


Community Detection

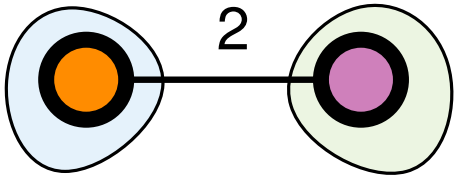
Framework:

- Vorverarbeitung: **Communitystruktur** bestimmen
- Verbot von **Inter-Community**-Kontraktionen

Strukturerhaltende Vergröberung [HS17]



Lösung:

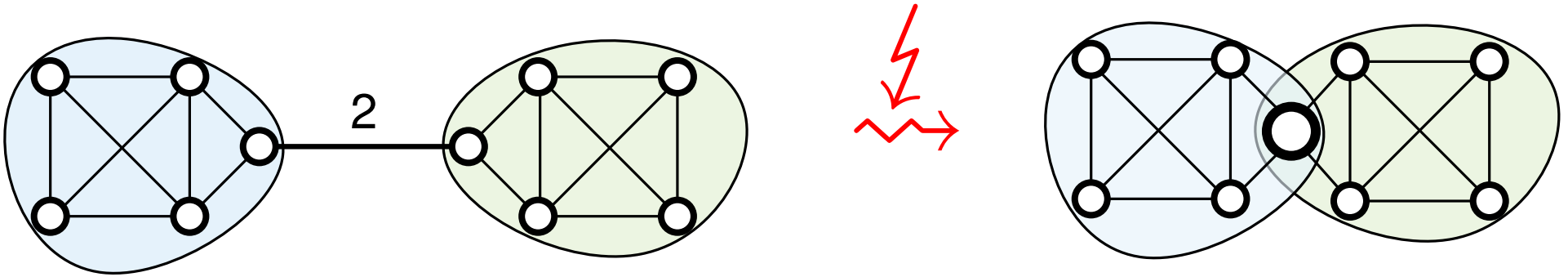


Community Detection

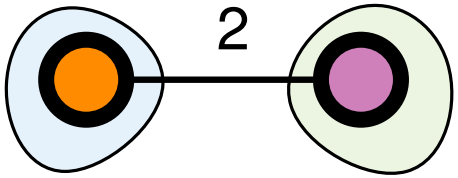
Framework:

- Vorverarbeitung: **Communitystruktur** bestimmen
- Verbot von **Inter-Community**-Kontraktionen

Strukturerhaltende Vergröberung [HS17]



Lösung:



Community Detection

Wie?

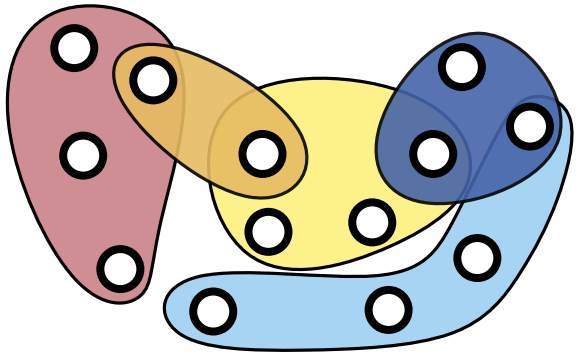
Framework:

- Vorverarbeitung: **Communitystruktur** bestimmen
- Verbot von **Inter-Community**-Kontraktionen

Graphrepräsentationen

Karlsruher Institut für Technologie

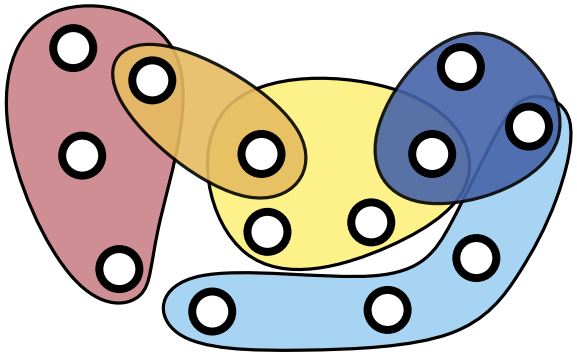
Hypergraph



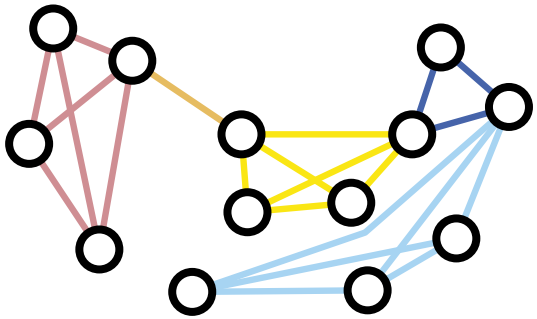
Graphrepräsentationen

Karlsruher Institut für Technologie

Hypergraph



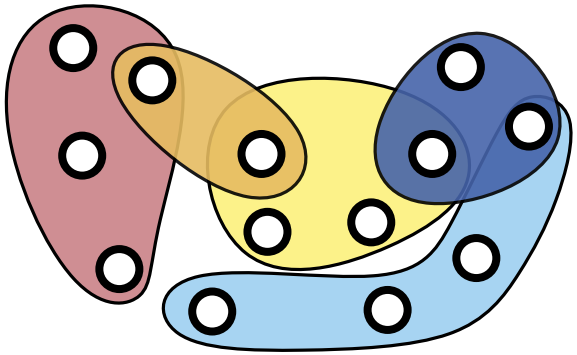
Clique-Graph



- $\binom{|e|}{2}$ Graphkanten pro Hyperkante e
- Große Hyperkanten werden überbewertet

Graphrepräsentationen

Hypergraph

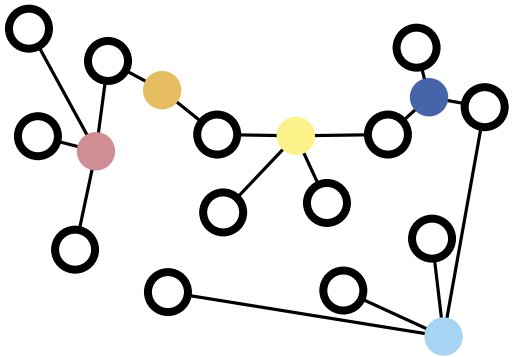


Clique-Graph

– $\binom{|e|}{2}$ Graphkanten pro Hyperkante e

– Große Hyperkanten werden überbewertet

Bipartiter Graph

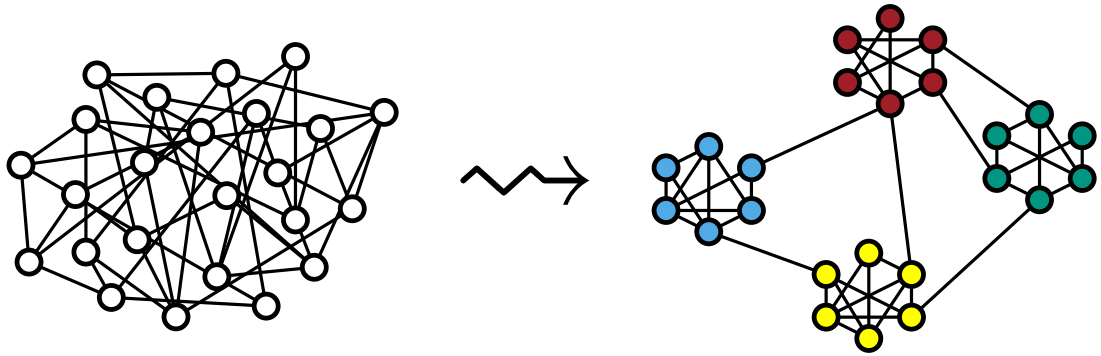


\+ Kompakte Repräsentation: $\mathcal{O}(|\text{Pins}|)$

– Netze werden Knoten & Teil des Clusterings

Analyse der Communitystruktur

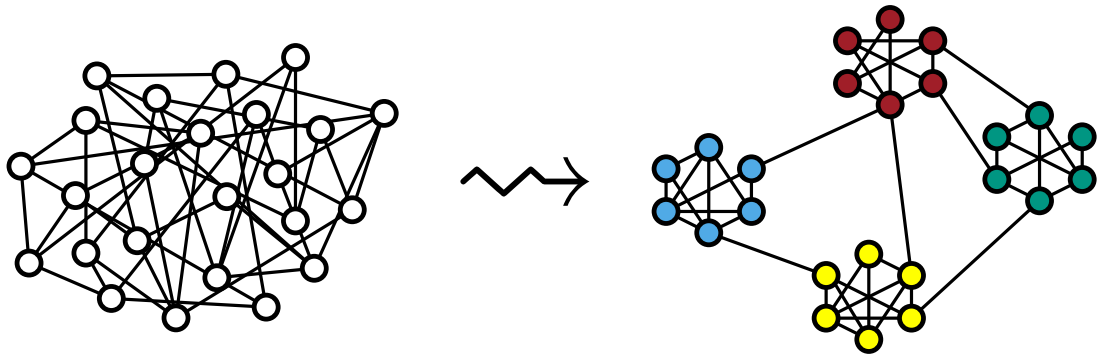
Ziel: Zerlege Graphen in **natürliche** Subgraphen $\mathcal{C}$



Community:
innen **dicht**,
außen **lose**
verbundener
Subgraph

Analyse der Communitystruktur

Ziel: Zerlege Graphen in **natürliche** Subgraphen $\mathcal{C}$



Community:
innen **dicht**,
außen **lose**
verbundener
Subgraph

(Eine) **Formalisierung**: [NG04]

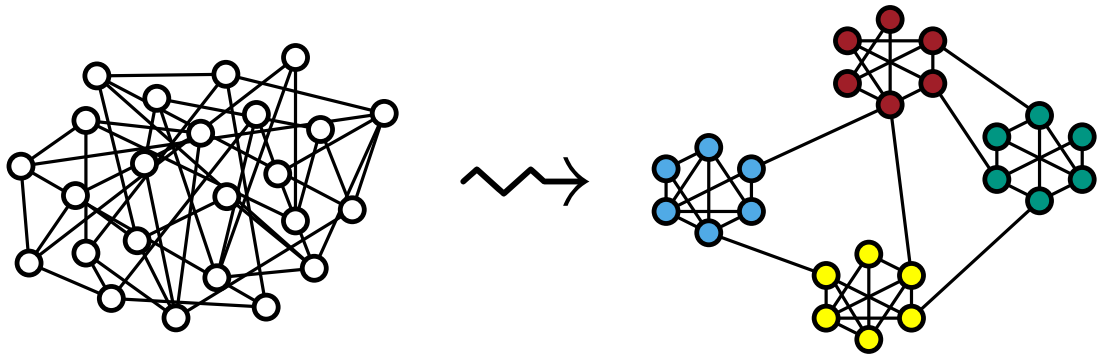
Modularity $\mathtt{mod}(G, \mathcal{C}) := \mathtt{cov}(G, \mathcal{C}) - \mathbb{E}[\mathtt{cov}(G, \mathcal{C})]$

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

Anteil „innerer“ Kanten

Analyse der Communitystruktur

Ziel: Zerlege Graphen in **natürliche** Subgraphen $\mathcal{C}$



Community:
innen **dicht**,
außen **lose**
verbundener
Subgraph

(Eine) **Formalisierung**: [NG04]

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

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

Anteil „innerer“ Kanten

Effiziente Heuristik: Louvain-Methode [Blo+08]

- Verschiebe Knoten wiederholt in benachbarte Communities
- Vergröbere Graphen & wiederhole Prozess

Auswirkungen auf die Lösungsqualität

Verbesserung gegenüber traditioneller Vergröberung:

164 Hypergraphen, $k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$

Verbesserung (%)	VLSI		Dünnb. Matrizen		SAT
	DAC	ISPD	All	Web/Social	Literal
Initiale Lösung [∅]	20,34	13,97	4,05	26,18	34,20
Finale ” [Min]	3,07	0,79	0,82	4,56	3,11
Finale ” [∅]	3,70	1,23	1,16	6,55	4,84
Finale ” [Max]	3,92	1,77	1,60	8,57	7,16

Auswirkungen auf die Lösungsqualität

Verbesserung gegenüber traditioneller Vergröberung:

164 Hypergraphen, $k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$

Verbesserung (%)	VLSI		Dünnb. Matrizen		SAT
	DAC	ISPD	All	Web/Social	Literal
Initiale Lösung [∅]	20,34	13,97	4,05	26,18	34,20
Finale ” [Min]	3,07	0,79	0,82	4,56	3,11
Finale ” [∅]	3,70	1,23	1,16	6,55	4,84
Finale ” [Max]	3,92	1,77	1,60	8,57	7,16

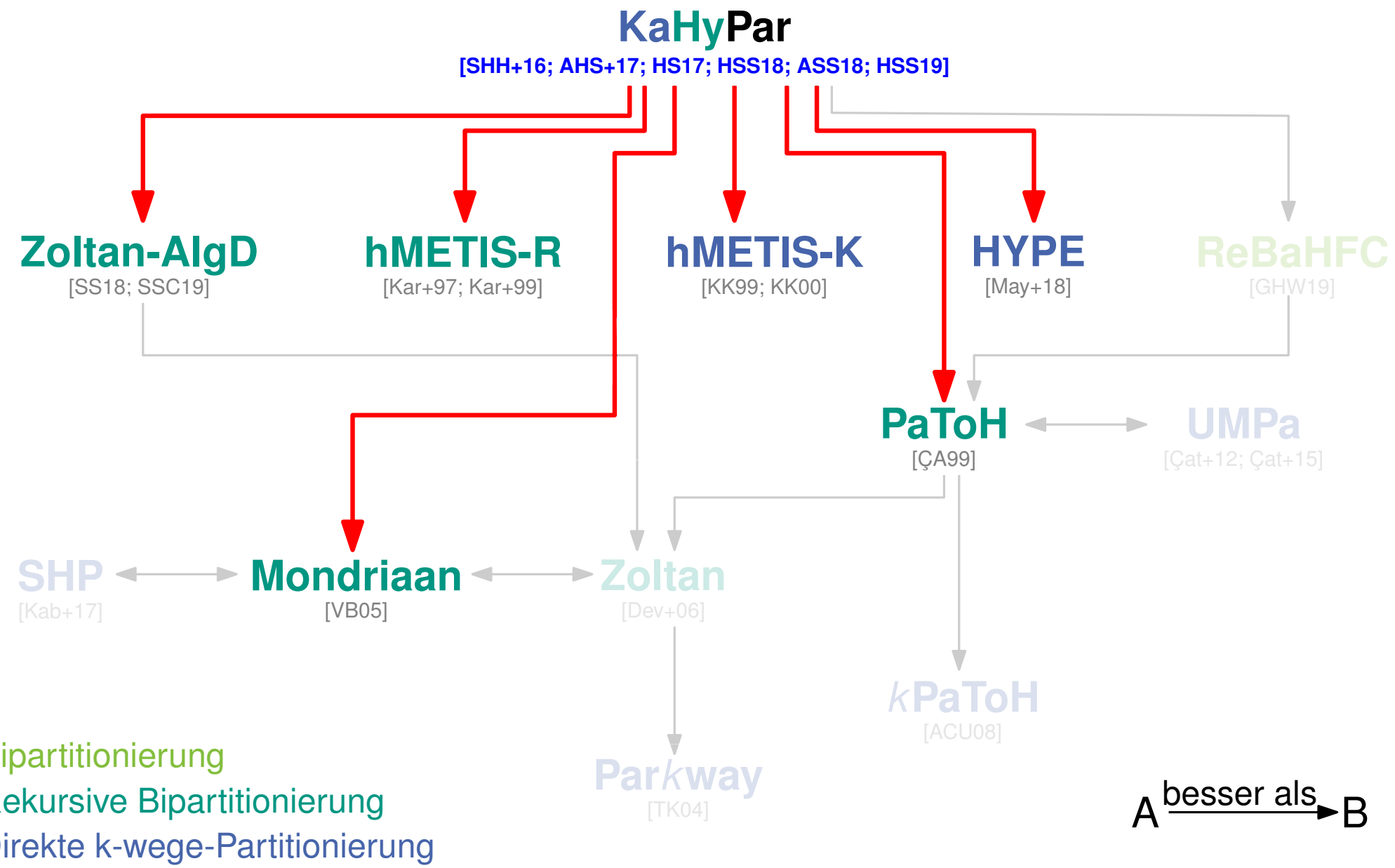
Auswirkungen auf die Lösungsqualität

Verbesserung gegenüber traditioneller Vergröberung:

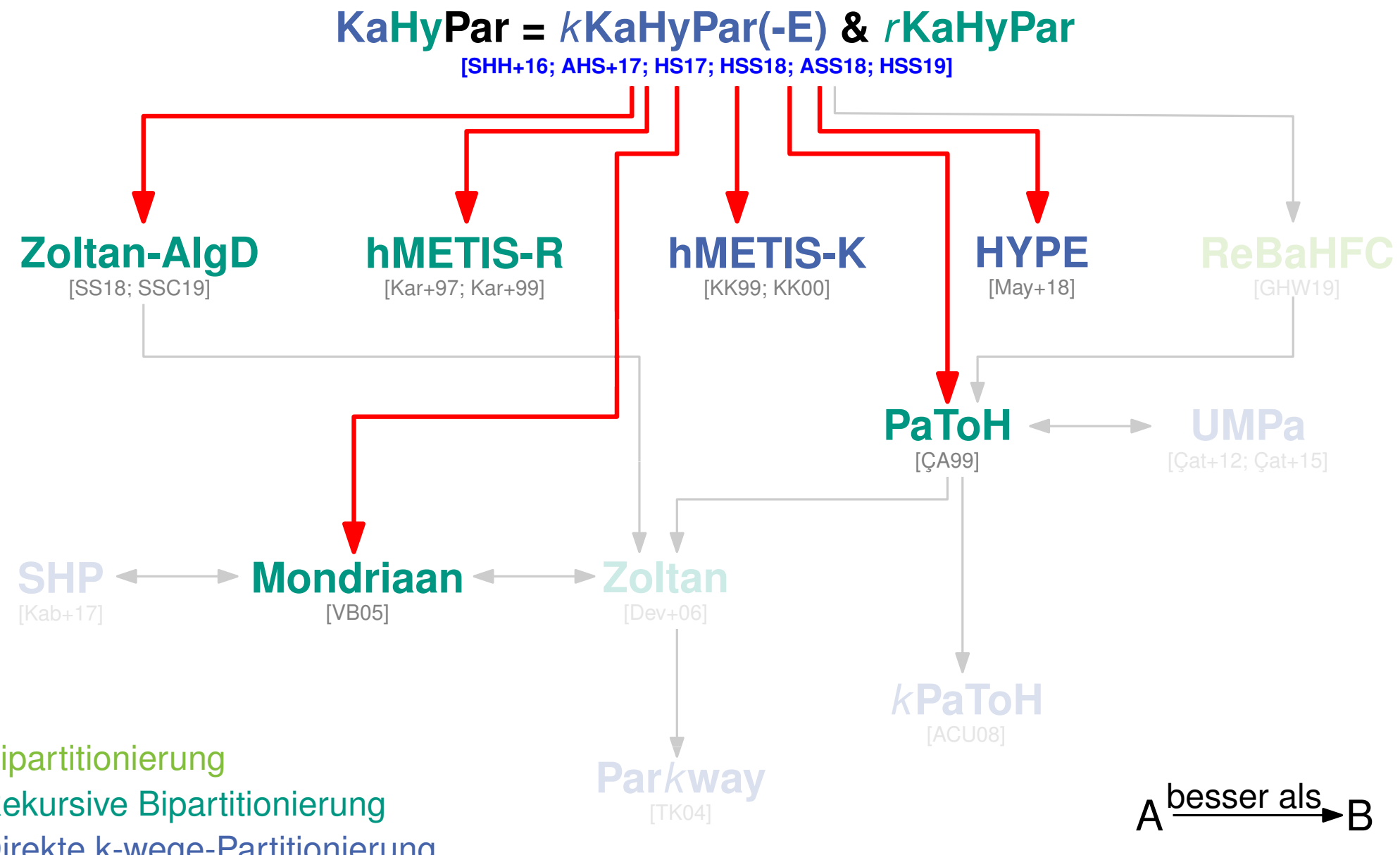
164 Hypergraphen, $k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$

Verbesserung (%)	VLSI		Dünnb. Matrizen		SAT
	DAC	ISPD	All	Web/Social	Literal
Initiale Lösung [∅]	20,34	13,97	4,05	26,18	34,20
Finale ” [Min]	3,07	0,79	0,82	4,56	3,11
Finale ” [∅]	3,70	1,23	1,16	6,55	4,84
Finale ” [Max]	3,92	1,77	1,60	8,57	7,16

Vergleich mit dem Stand der Technik?



Vergleich mit dem Stand der Technik?

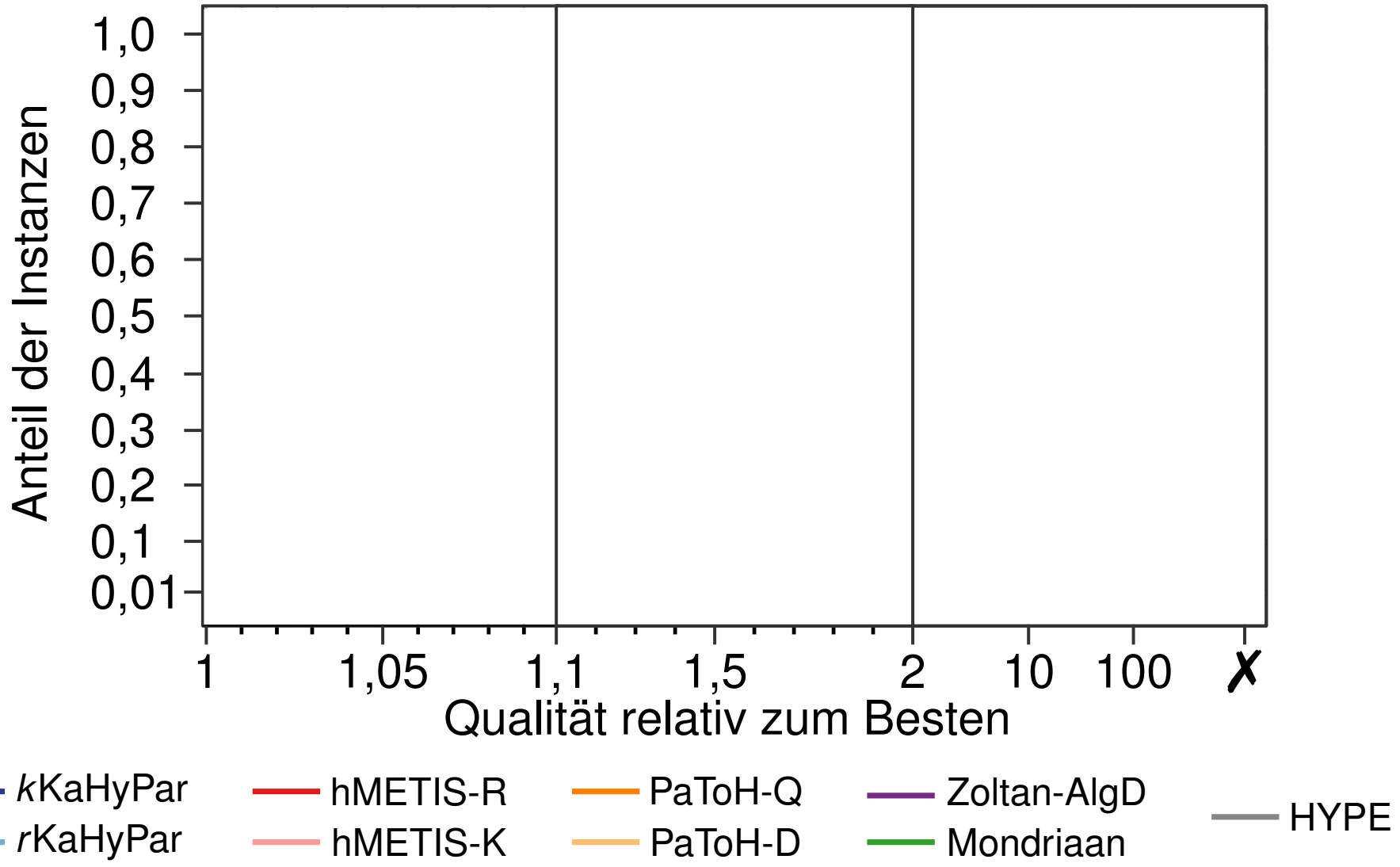


Konnektivitätsoptimierung: Lösungsqualität

Karlsruher Institut für Technologie

≈ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h



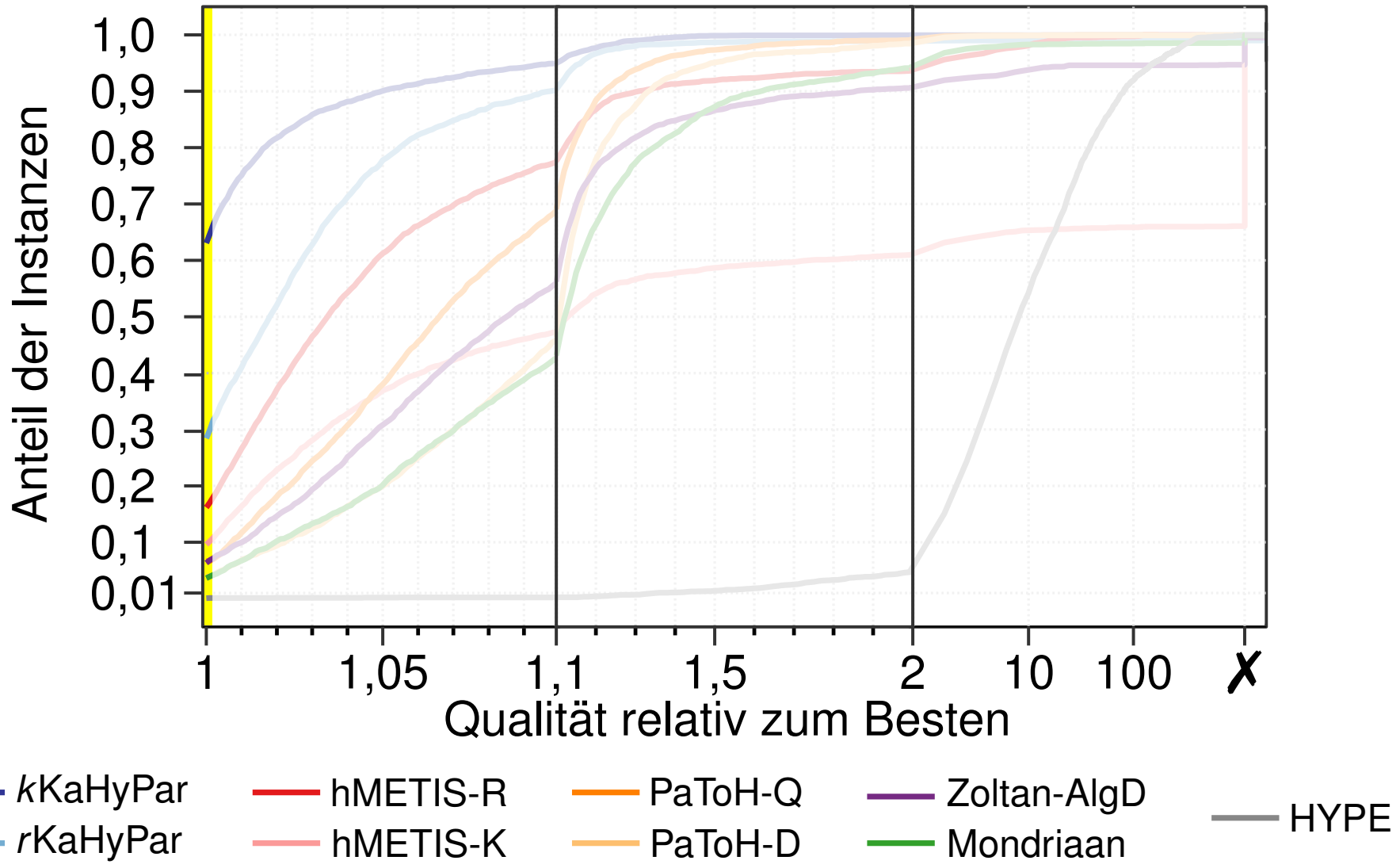
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Konnektivitätsoptimierung: Lösungsqualität

Karlsruher Institut für Technologie

≈ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h



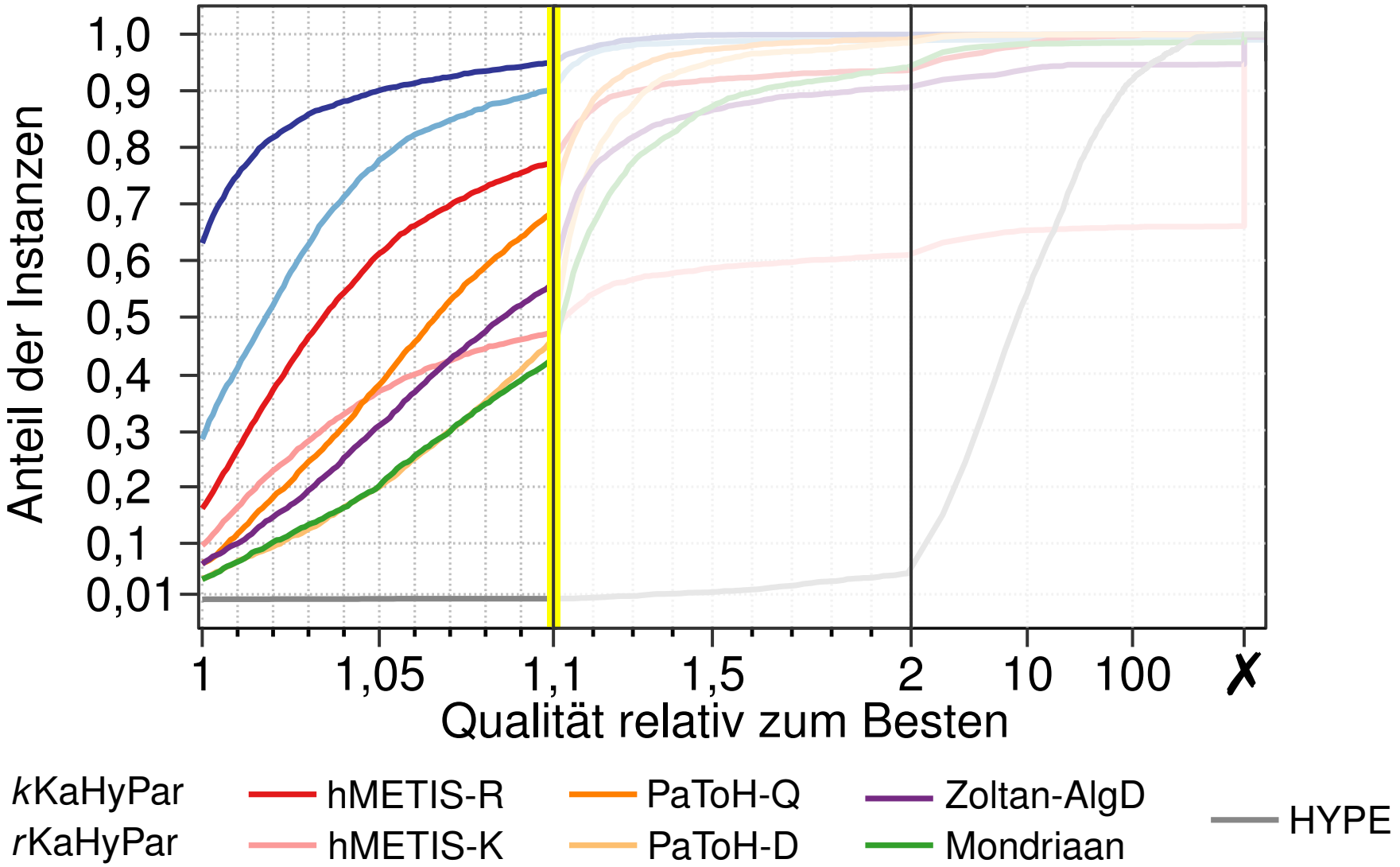
Institut für Theoretische Informatik
Lehrstuhl Algorithmik

Konnektivitätsoptimierung: Lösungsqualität

Karlsruher Institut für Technologie

≈ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h

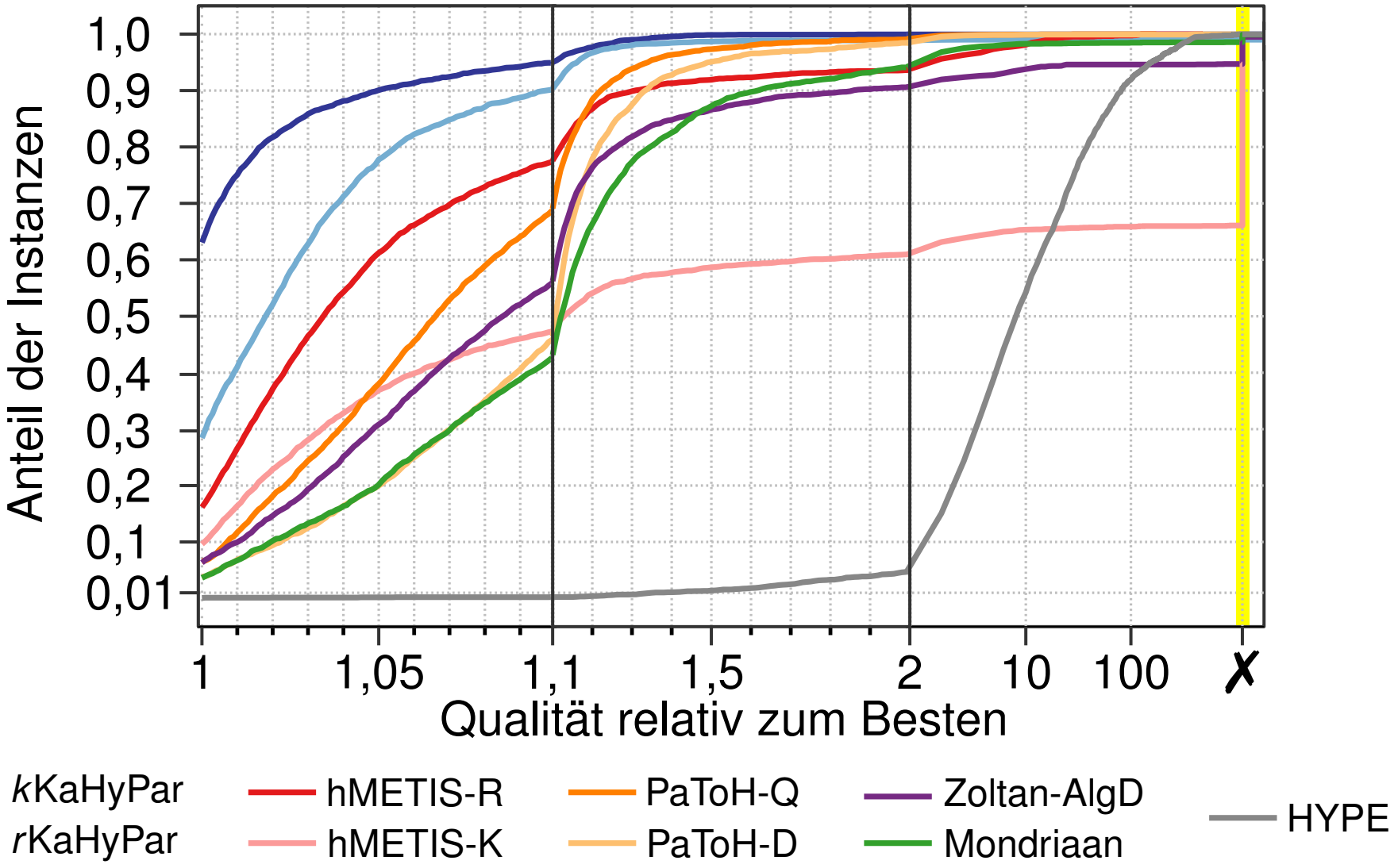


Konnektivitätsoptimierung: Lösungsqualität

Karlsruher Institut für Technologie

≈ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h

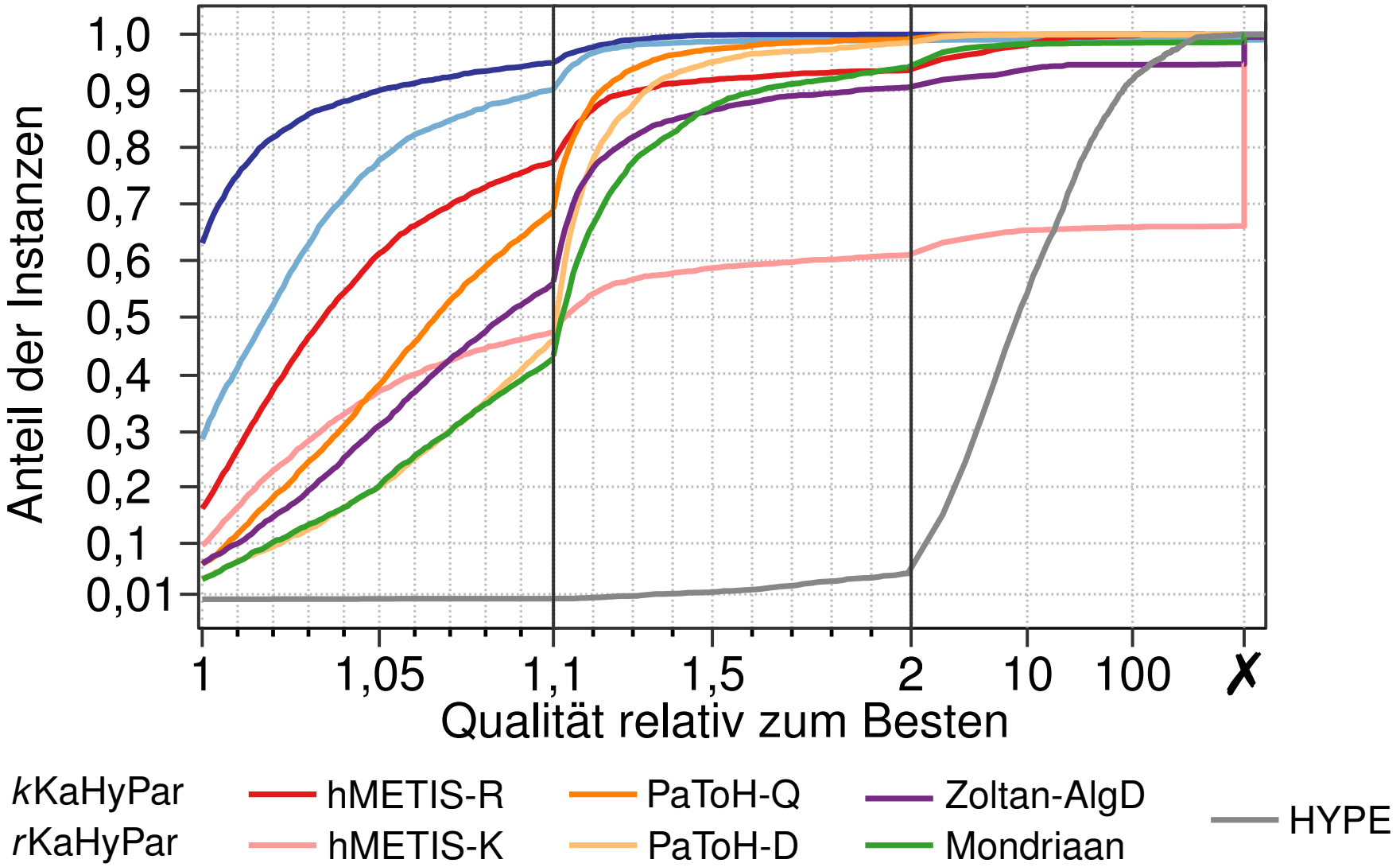


Konnektivitätsoptimierung: Lösungsqualität

Karlsruher Institut für Technologie

≈ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h

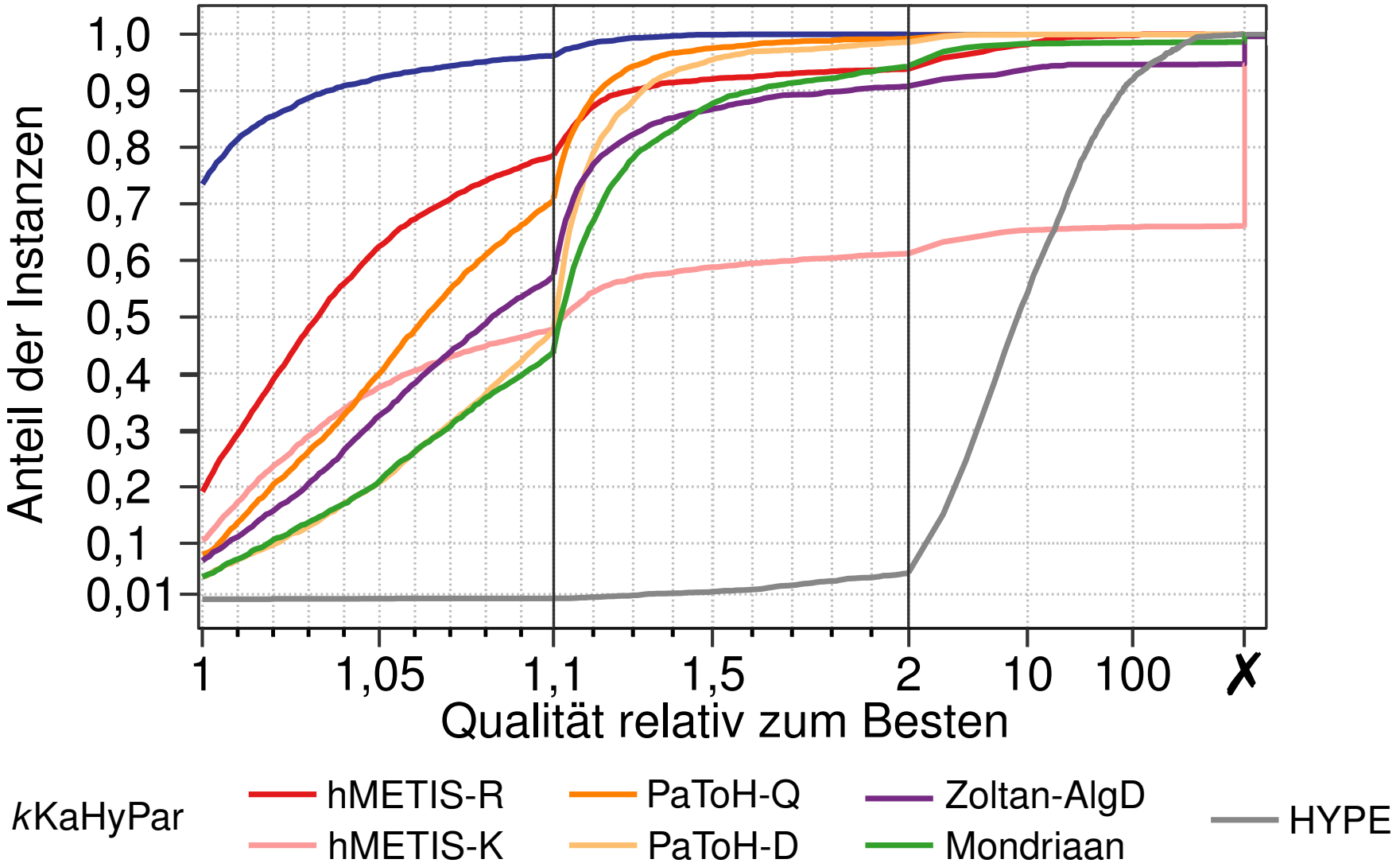


Konnektivitätsoptimierung: Lösungsqualität

Karlsruher Institut für Technologie

≈ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h

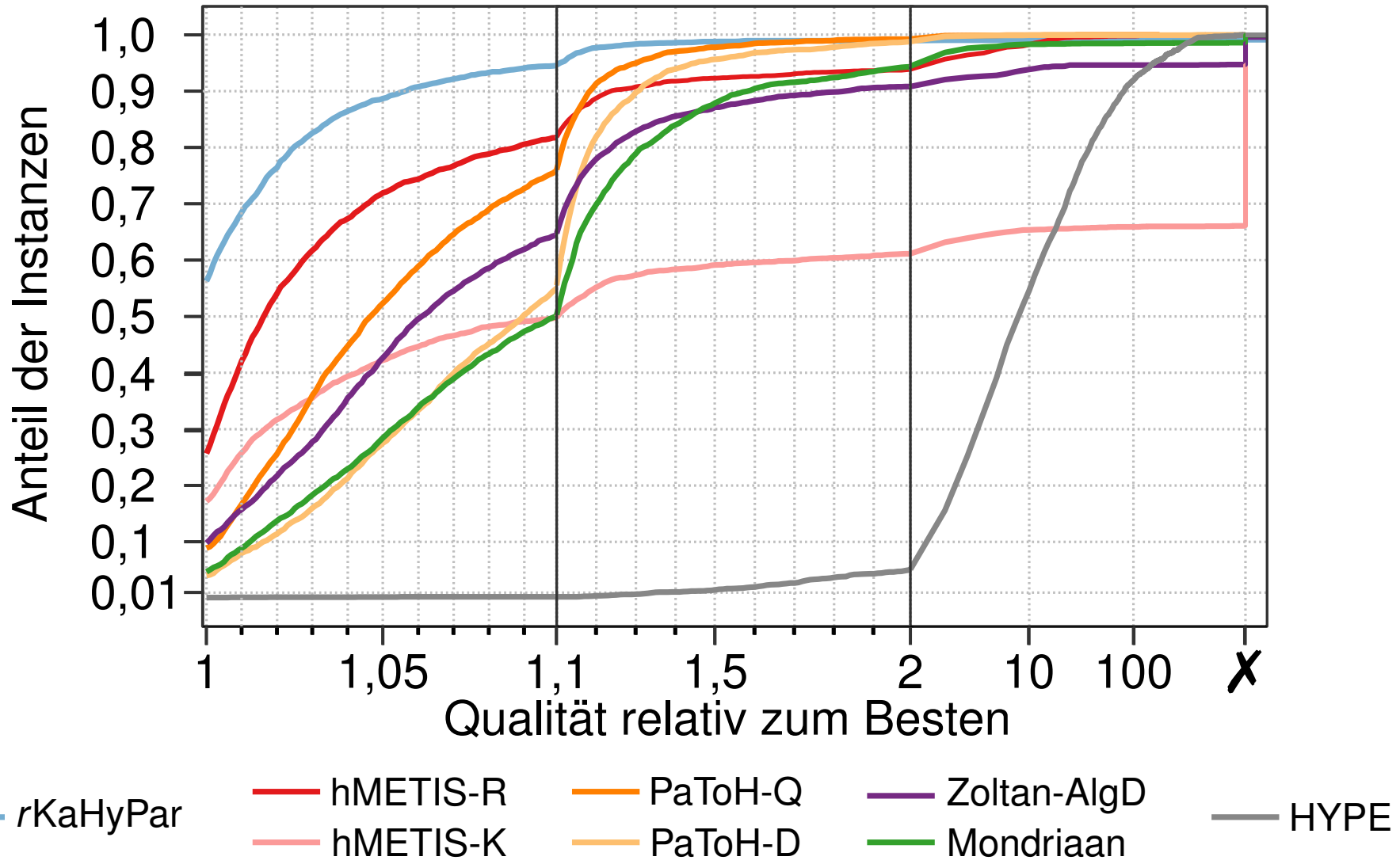


Konnektivitätsoptimierung: Lösungsqualität

Karlsruher Institut für Technologie

≈ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

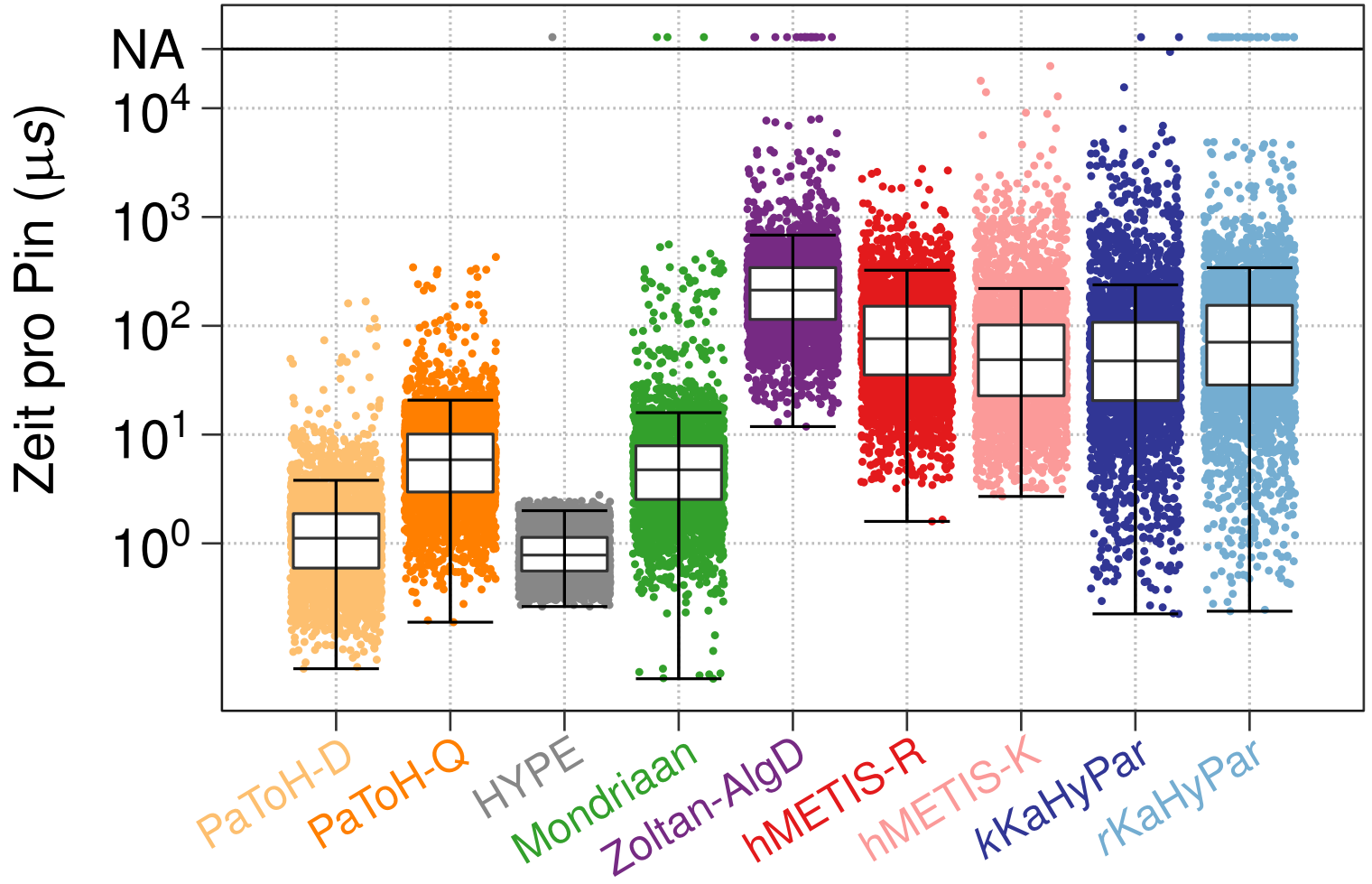
$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h



Konnektivitätsoptimierung: Laufzeit

$\approx$ 500 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

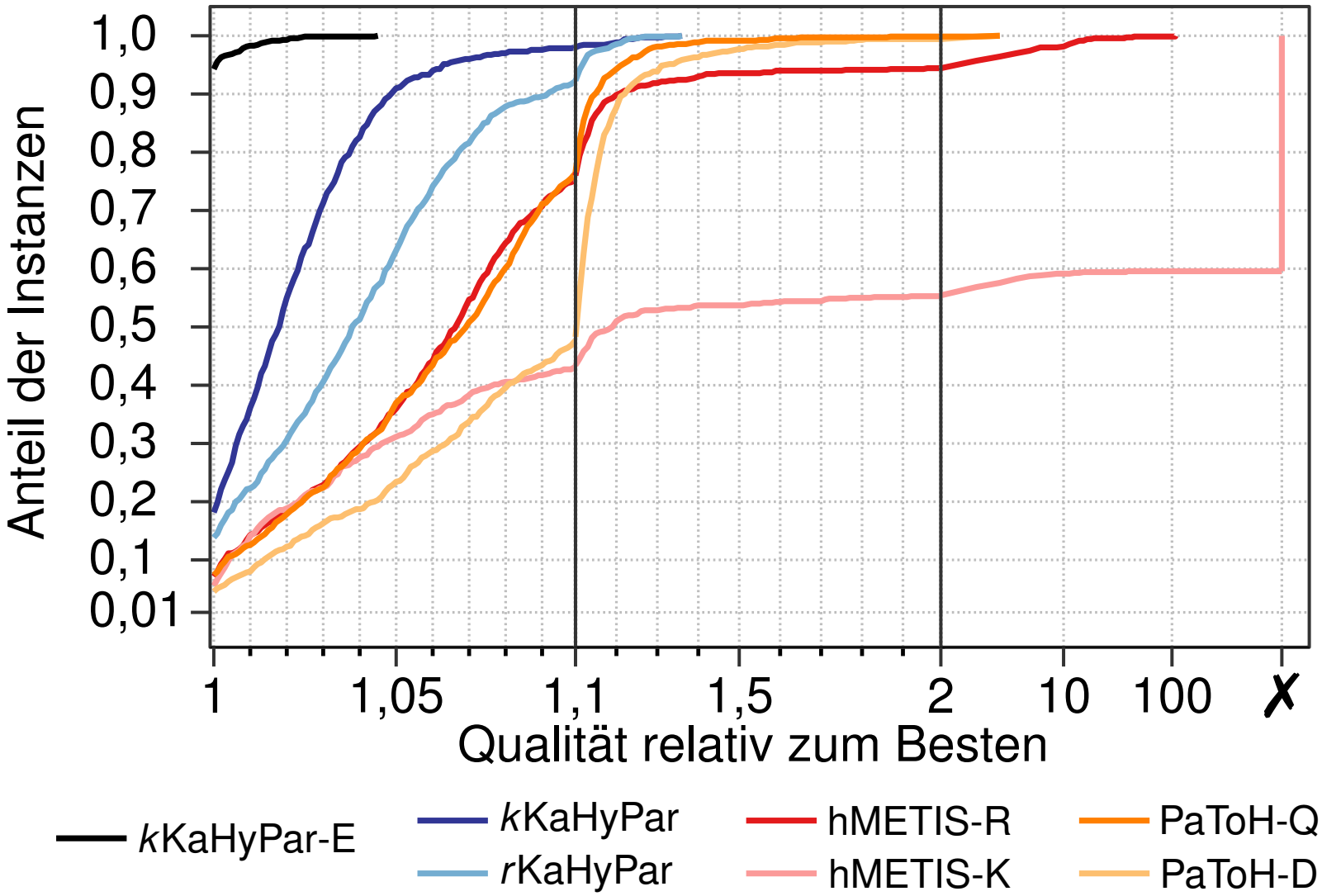
$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, Zeitlimit: 8 h



Konnektivitätsoptimierung: Evolutionärer Alg.

100 Hypergraphen (VLSI, SAT, Dünnb. Matrizen)

$k \in \{2, 4, 8, 16, 32, 64, 128\}$, $\varepsilon = 0{,}03$, **Zeit pro Instanz**: 8 h



Fazit

Zentraler Beitrag: Karlsruhe **Hy**pergraph **Par**titioning Framework

- Hochqualitative Partitionierung von **Hypergraphen** und **Graphen**
- Publikationen: ALENEX (’16, ’17), SEA (’17, ’18), GECCO’18, JEA’19
- Open source: `http://www.kahypar.org`
- In der Forschung eingesetzt [AH19; GLA19; Got19; Jal19b; PS19; Sch+19a; SSS19b; Gra19; Tom19]

Fazit

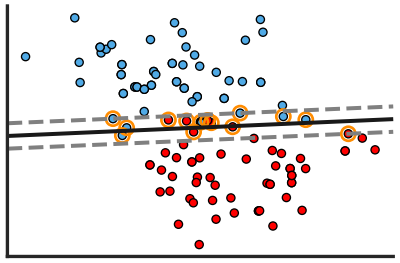
Zentraler Beitrag: Karlsruhe Hypergraph Partitioning Framework

- Hochqualitative Partitionierung von **Hypergraphen** und **Graphen**
- Publikationen: ALENEX (’16, ’17), SEA (’17, ’18), GECCO’18, JEA’19
- Open source: `http://www.kahypar.org`
- In der Forschung eingesetzt [AH19; GLA19; Got19; Jal19b; PS19; Sch+19a; SSS19b; Gra19; Tom19]

Weitere Publikationen:

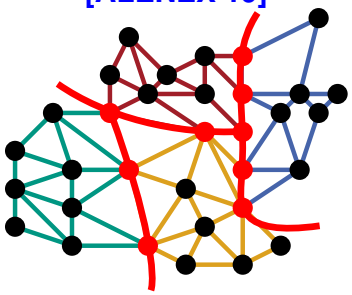
Schnellere SVMs

[ALENEX’19]



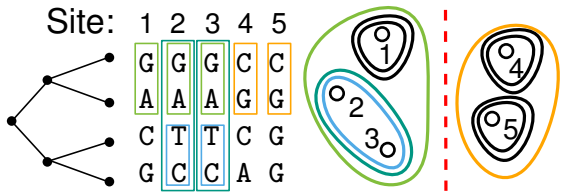
Skalierbare Kanten-Partitionierung

[ALENEX’19]



Lastbalancierung für phylogenetische Inferenz

[HiComb’19]



Thrill

[BigData’16]