Karlsruher Institut für Technologie

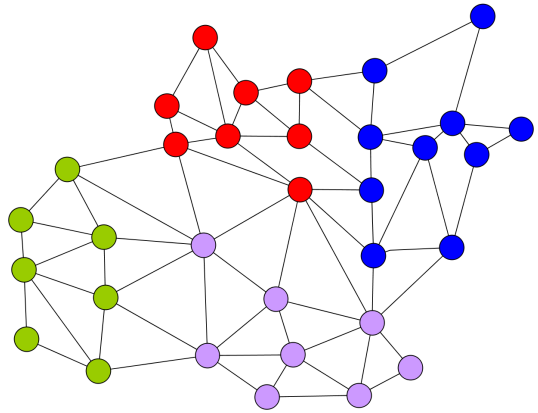
Engineering Multilevel Graph Partitioning Algorithms

Peter Sanders, Christian Schulz

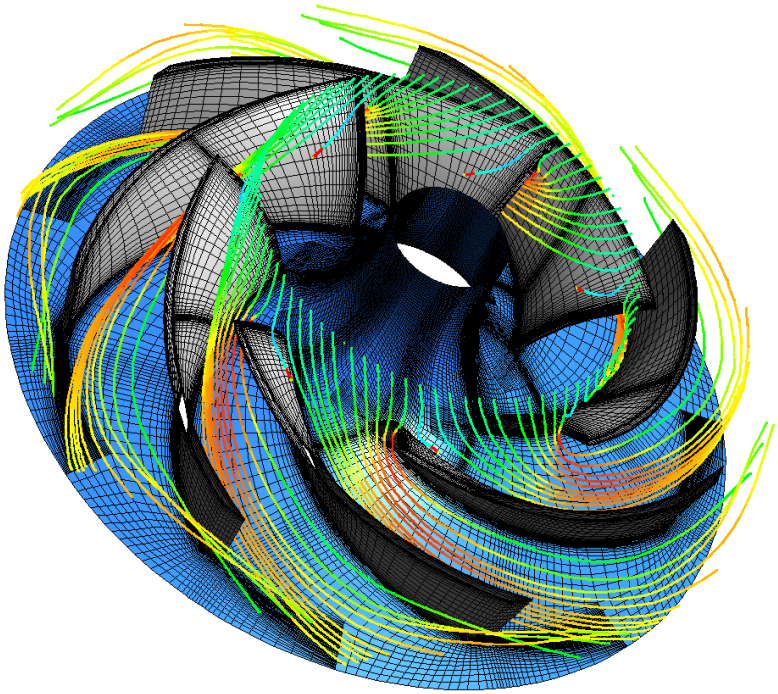
Institute for Theoretical Computer Science, Algorithmics II

Overview

- Introduction
- Multilevel Algorithms
- Local Improvement using Flows
- More Localized Search
- Global Search
- Experimental Evaluation
- Future Work

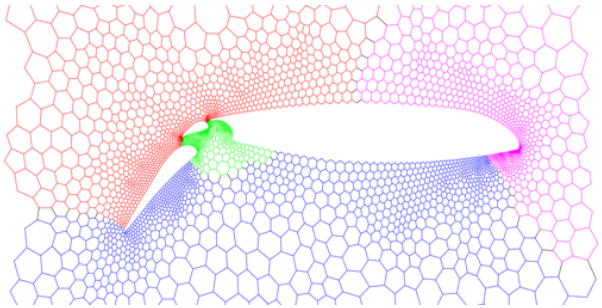


Simulation - FEM



- Simulation space is discretized into a mesh
- Solution of partitial differential equations are approximated by linear equations
- Number of vertices can become quite large $\rightarrow$ time and memory
- Parallel processing required

The Common Parallel Approach



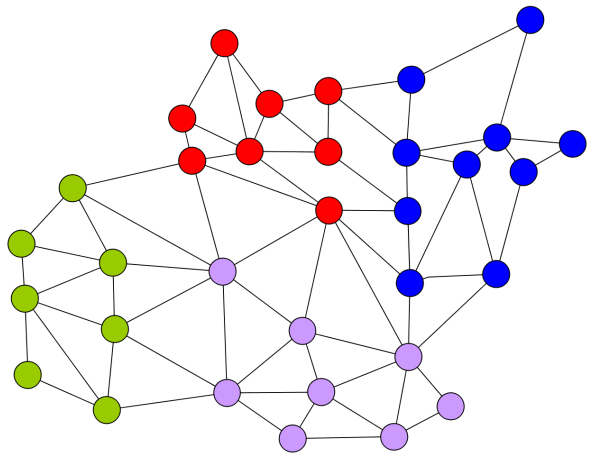
- Mesh partitioned via dual graph
  1. Each volume (data, calculation) is represented by a vertex
  2. Interdependencies are represented by edges
- All PE's get same amount of work
- Communication is expensive

Graph Partitioning Problem:
Partition a graph into (almost) equally sized blocks, such that the number of edges connecting vertices from different blocks is minimal.

ϵ-Balanced Graph Partitioning

Partition graph $G = (V, E, c : V \to \mathbf{R}_{>0}, \omega : E \to \mathbf{R}_{>0})$
into k disjoint blocks s.t.

- total node weight of each block $\leq \frac{1+\epsilon}{k}$ total node weight
- total weight of cut edges as small as possible

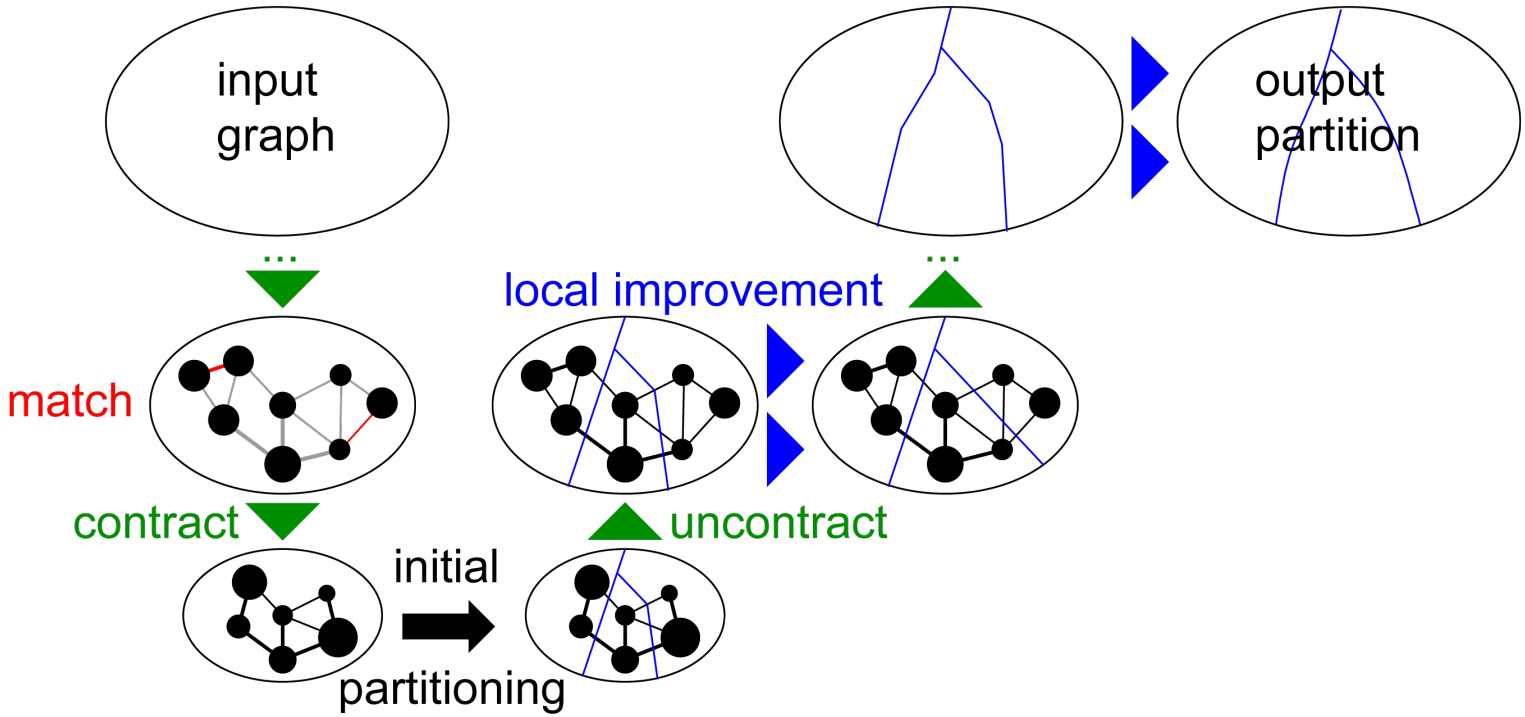


Applications:
finite element simulations, VLSI design, route planning, ...

More Applications

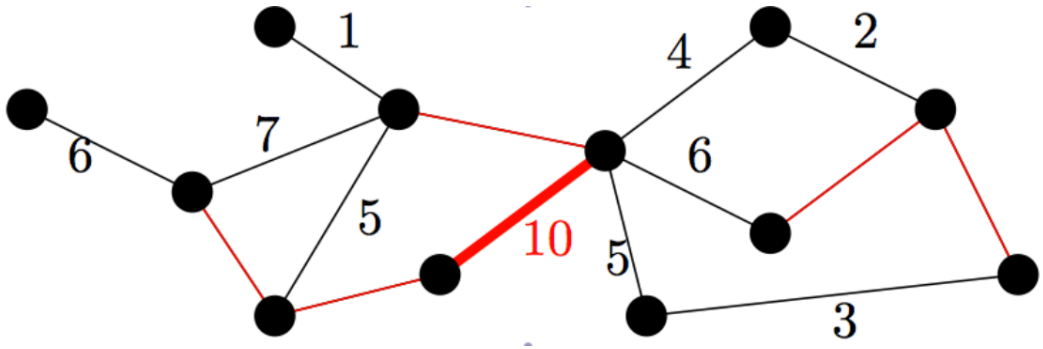
- Large Linear Equation Systems
  1. Direct Methods (Fill In Reduction)
  2. Iterative Methods (Parallel)
- where do they come from?
- how do the methods work?

Multi-Level Graph Partitioning



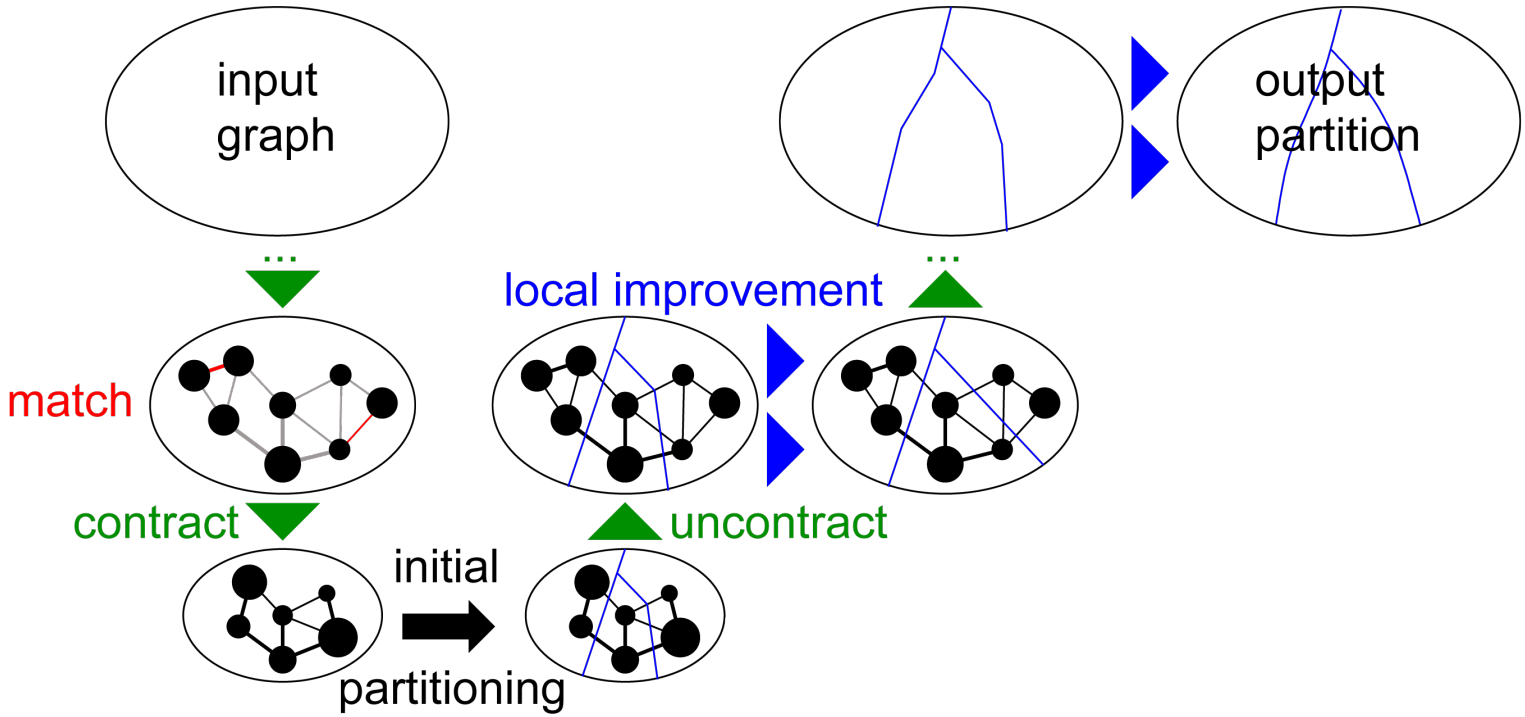
Successful in existing systems:
DiBaP, Chaco, Jostle, Metis, Scotch,. . . , **KaPPa**, **KaSPar**

GPA - Global Paths Alghoritm



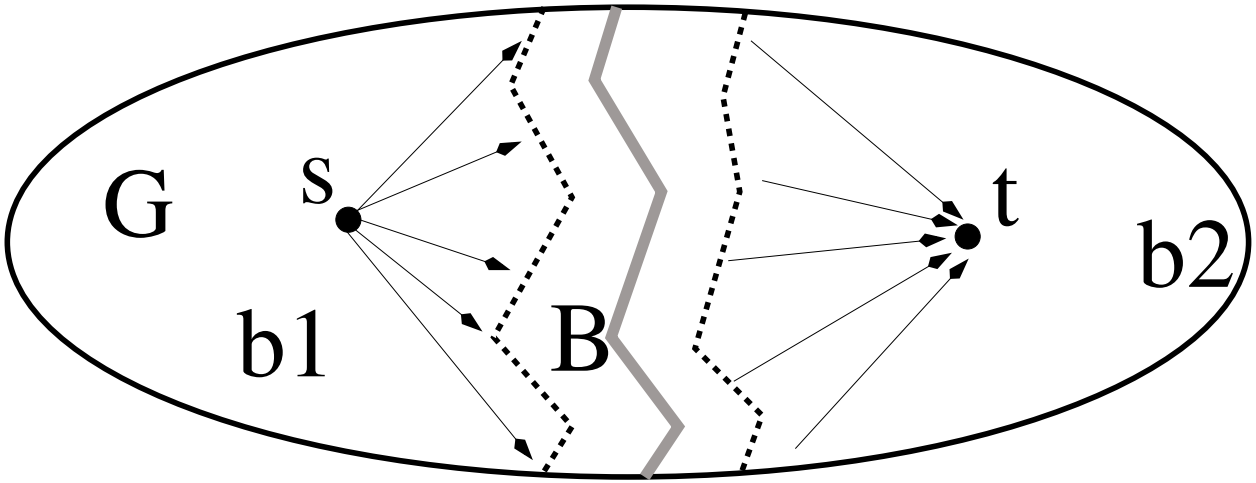
- Sort edges according to their weight
- Grow a set of paths and even-length cycles
- Find optimum matching for every path and cycle
- Running time $O(m + \text{sort}(m))$
- Approximation $\frac{1}{2}$OPT
- outperforms HEM, SHEM

Multi-Level Graph Partitioning
Initial Partitioning



Initial Partitioner:
Metis, Scotch

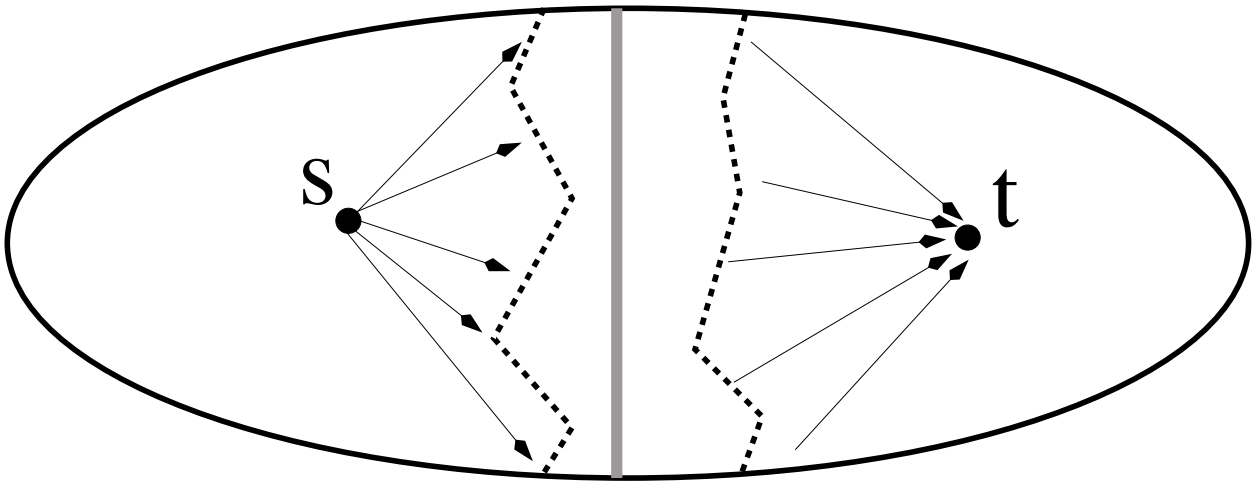
Flows as Local Improvement
Construction



- corridor B, such that each (s,t)-min cut is ϵ-balanced cut in G
- e.g. 2 times BFS (left, right)
- stop the BFS, if size would exceed $(1+\epsilon)\frac{n}{2} - \omega(b2)$
- $\Rightarrow \omega(b2_{\text{new}}) \leq \omega(b2) + (1+\epsilon)\frac{n}{2} - \omega(b2)$

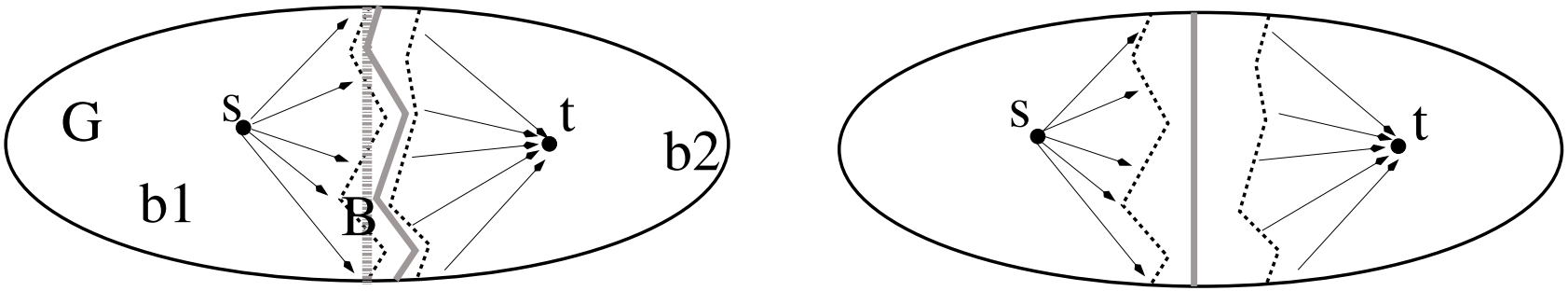
Flows as Local Improvement

- cut are optimal in corridor B
- ϵ-balanced partitioning NP-hard



Flows as Local Improvement
Adaptive Search

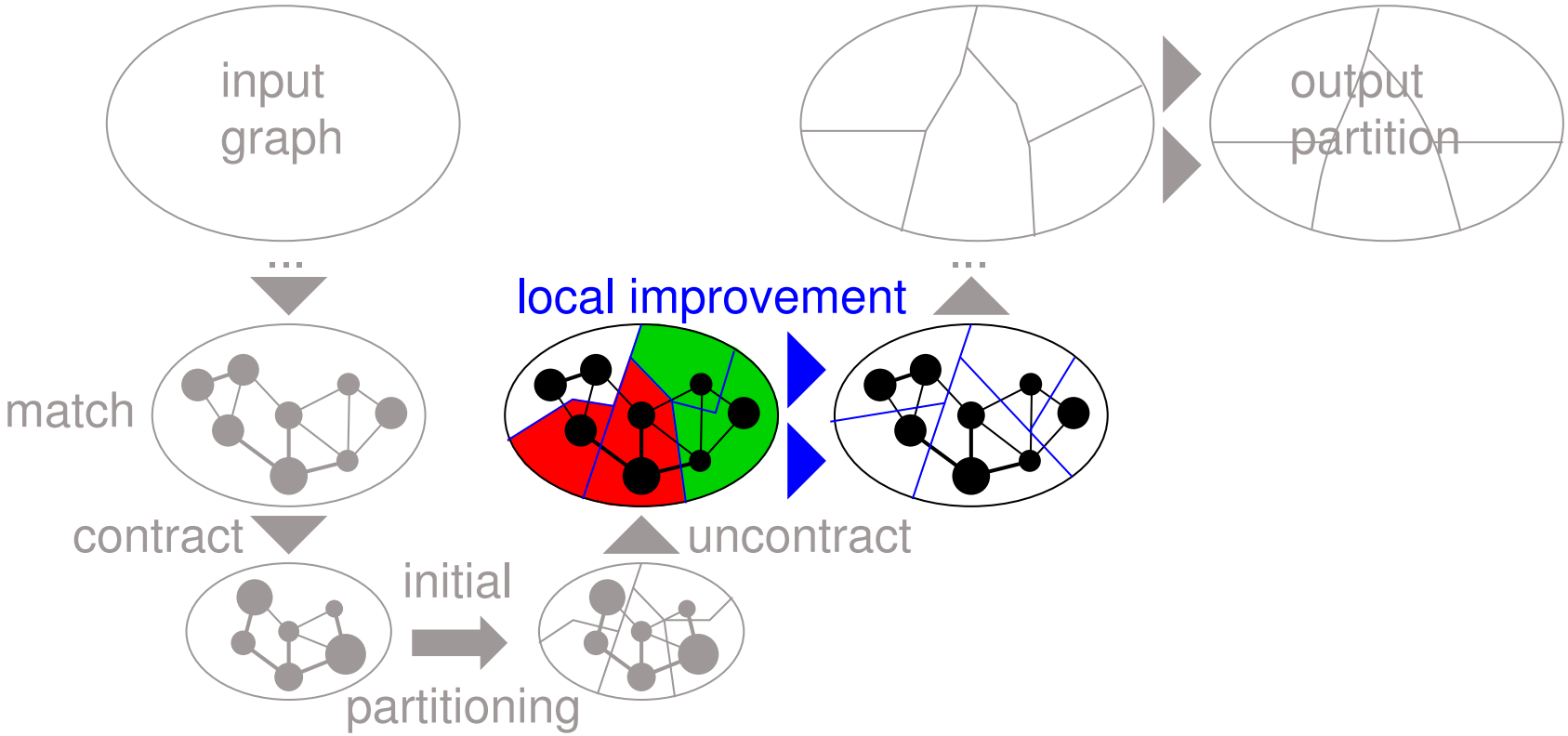
- search in larger areas for feasable cuts
- adaptively control the size of corridor B
- use heuristic Most Balanced Minimum Cuts (NP-hard)
- We need: SCC's, DAGs, Closed Vertex Sets, Topological Sorting



- the maximal upper bound factor is called α'

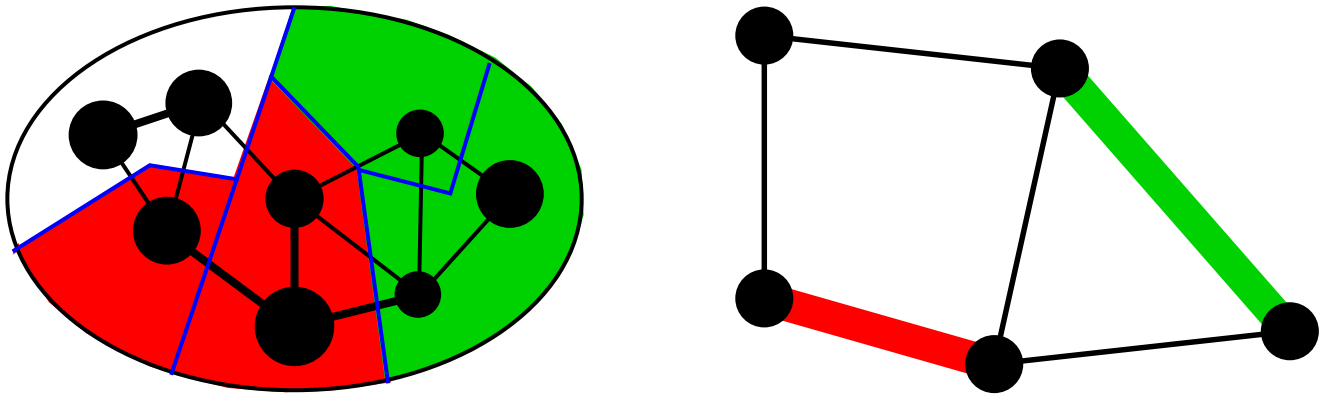
Local Improvement for k-partitions
Using Flows?

on each pair of blocks



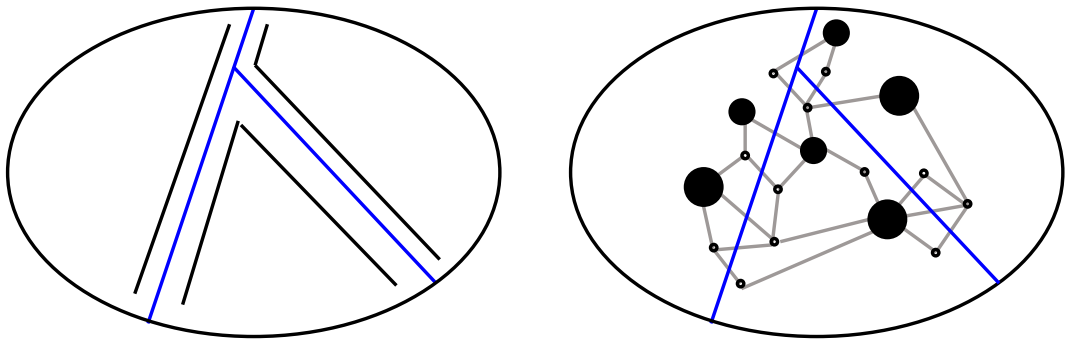
Local Improvement for k-partitions
Using Flows?

on each pair of blocks



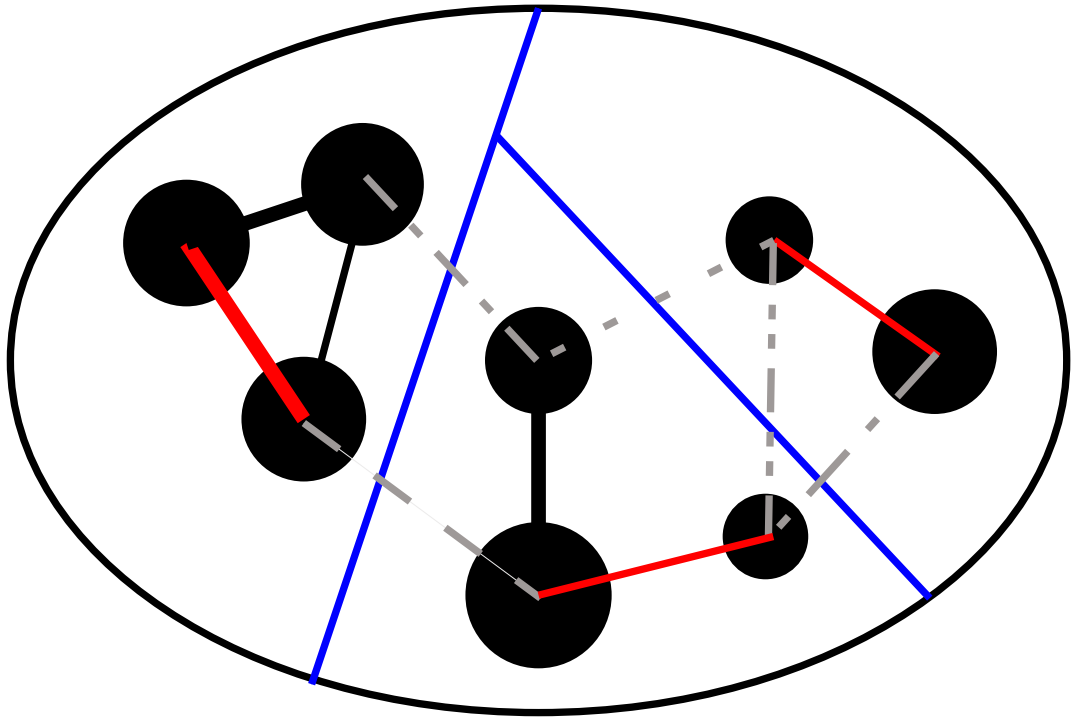
More Localized Local Search

- Idea: **KaPPa** and **KaSPar** $\Rightarrow$ more local searches are better
- Typical: k-way local search initialized with complete boundary
- each node moved at most once
- Localization:
  1. complete boundary $\Rightarrow$ maintained todo list T
  2. initialize search with single node $v \in_{\text{rnd}} T$
  3. iterate until $T = \emptyset$

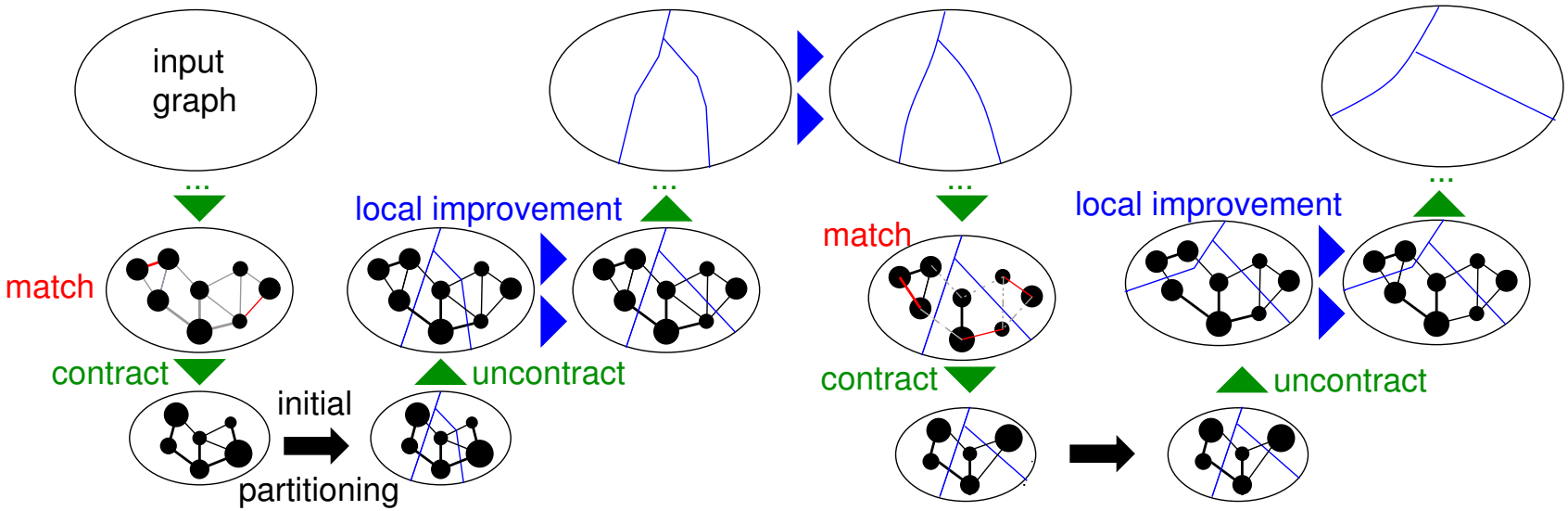


Global Search
Iterated Multilevel

match

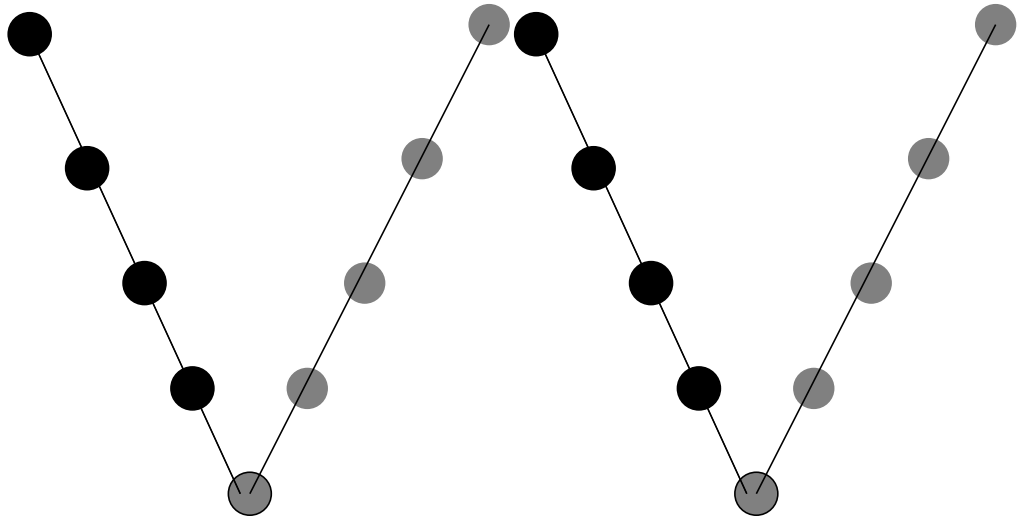


Global Search
Iterated Multilevel

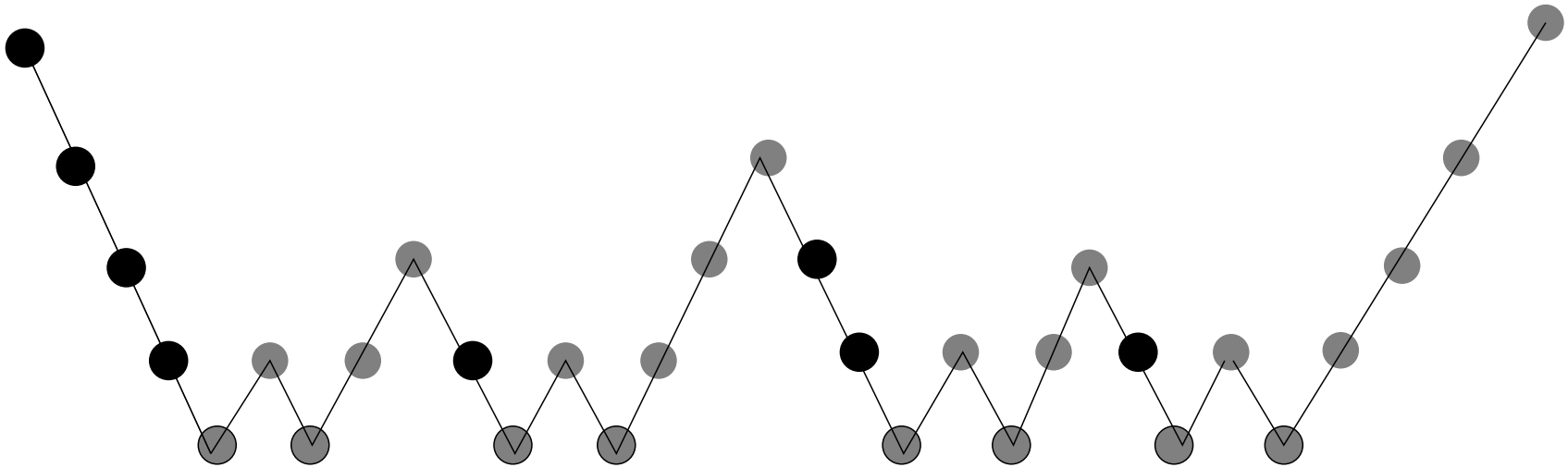


- no new initial partitioning needed
- random tiebreaking needed
- local improvement has to assure $\leq$ cuts

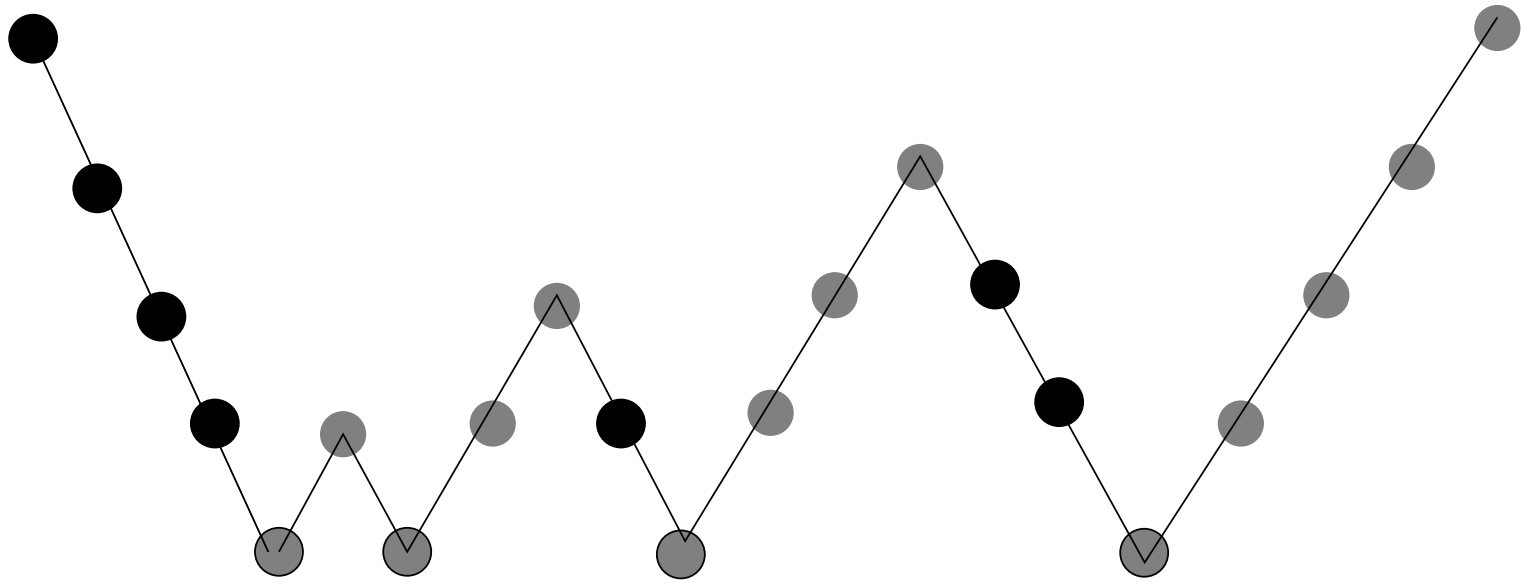
Global Search
V-Cycles



Global Search
W-Cycles



Global Search
F-Cycles



How Expensive Are They?

Theorem

If the time for coarsening and refinement is $T_{cr}(n) := bn$ and a constant shrink factor $a \in [1/2, 1)$ is given. Then:

$$T_{W,d}(n) \begin{cases} \lessapprox \frac{1-a^d}{1-2a^d} T_V(n) & \text{iff } 2a^d < 1 \\ \in \Theta(n \log n) & \text{iff } 2a^d = 1 \\ \in \Theta(n^{\frac{\log 2}{\log \frac{1}{a^d}}}) & \text{iff } 2a^d > 1 \end{cases} \tag{1}$$

$$T_{F,d}(n) \leq \frac{1}{1-a^d} T_V(n) \tag{2}$$

where T_V is the time for a single V-cycle and $T_{W,d}, T_{F,d}$ are the time for a W-cycle and F-cycle with level split parameter d.

Experiments
Today's Testset

Medium sized instances		
graph	n	m
rgg17	2^{17}	1 457 506
rgg18	2^{18}	3 094 566
Delaunay17	2^{17}	786 352
Delaunay18	2^{18}	1 572 792
bcsstk29	13 992	605 496
4elt	15 606	91 756
fesphere	16 386	98 304
cti	16 840	96 464
memplus	17 758	108 384
cs4	33 499	87 716
pwt	36 519	289 588
bcsstk32	44 609	1 970 092
body	45 087	327 468
t60k	60 005	178 880
wing	62 032	243 088
brack2	62 631	733 118
finan512	74 752	522 240
bel	463 514	1 183 764
nld	893 041	2 279 080
af_shell9	504 855	17 084 020

Experiments
Remove Components

Repetition	Avg.	Best.	t
KaFFPa Strong	2683	2617	8.93
-KWay	2682	2614	9.23
-MoreLocalizedSearch	2729	2656	5.55
-FCycle	2748	2668	3.27
-Flow	2934	2823	1.66

Effectiveness	Avg.	Best.
KaFFPa Strong	2636	2616
-KWay	2636	2618
-MoreLocalizedSearch	2668	2650
-FCycle	2669	2653
-Flow	2799	2775

Outlook

- Walshaw Benchmark: 317 improvements (of about 600 possible)
- 118 same cut as previous best
- TR: http://arxiv.org/abs/1012.0006
- Many more experiments

Future Work

1. better initial partitioning for large k
2. integration into meta-heuristics (even parallel)
3. exploit shared memory parallelism
4. vertex separators can be computed from edge separators
5. graph clustering

Thank you!

Contact: christian.schulz@kit.edu