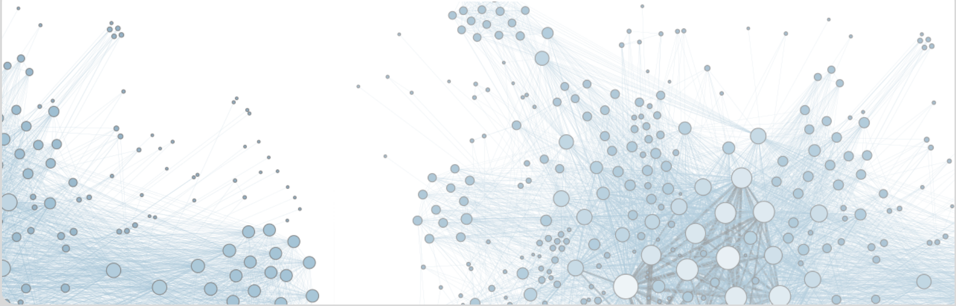
Practical Kernelization Techniques for the Maximum Cut Problem

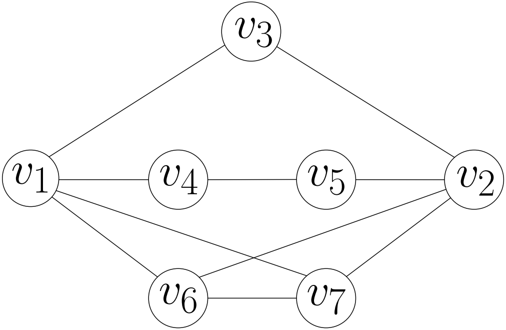
Damir Ferizovic, **Demian Hespe**, Sebastian Lamm,
Matthias Mnich, Christian Schulz, Darren Strash | February 18, 2019

DEPARTMENT OF INFORMATICS: INSTITUTE OF THEORETICAL INFORMATICS



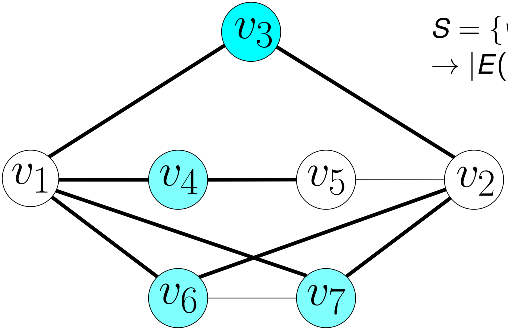
Max-Cut: Definition and Example

- Given $G = (V, E)$, find $S \subseteq V$ such that $|E(S, V \setminus S)|$ is maximum
- Notation: $mc(G) := \max_{S \subseteq V} |E(S, V \setminus S)|$



Max-Cut: Definition and Example

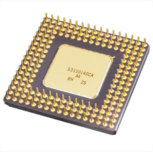
- Given $G = (V, E)$, find $S \subseteq V$ such that $|E(S, V \setminus S)|$ is maximum
- Notation: $mc(G) := \max_{S \subseteq V} |E(S, V \setminus S)|$



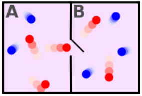
$S = \{v_3, v_4, v_6, v_7\}$
$\rightarrow |E(S, V \setminus S)| = 8$

Max-Cut: Importance of Studying it

- Member of Karp's 21 **NP-complete** problems
- Used in...



Circuit design



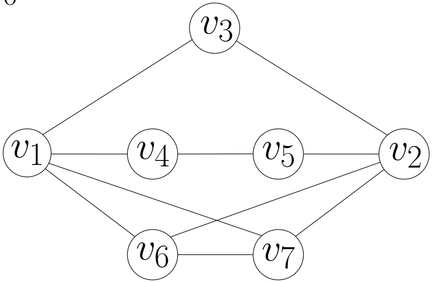
Statistical physics

Social networks

Kernelization: Definition and Example

- Kernelization: Compress graph while preserving optimality

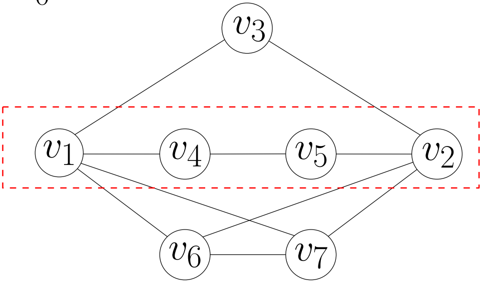
$G_0 = G:$



Kernelization: Definition and Example

- Kernelization: Compress graph while preserving optimality

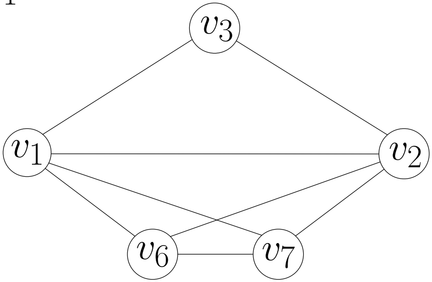
$G_0 = G$:



Kernelization: Definition and Example

- Kernelization: Compress graph while preserving optimality

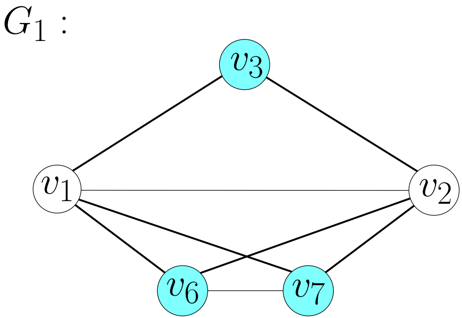
G_1 :



$$mc(G_0) = mc(G_1) + 2$$

Kernelization: Definition and Example

- Kernelization: Compress graph while preserving optimality

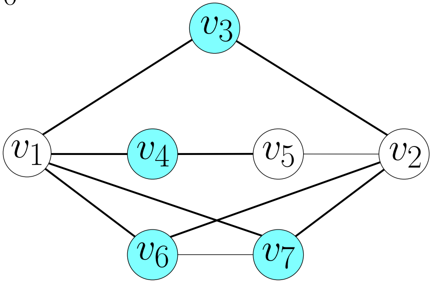


$mc(G_0) = mc(G_1) + 2$
$mc(G_1) = 6$

Kernelization: Definition and Example

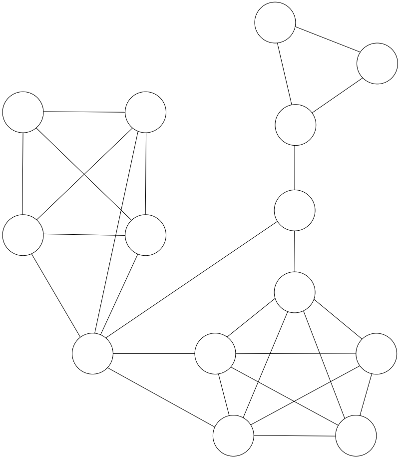
- Kernelization: Compress graph while preserving optimality

$G_0 = G$:

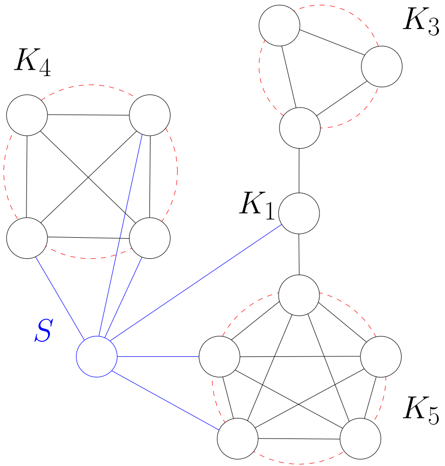


$mc(G_0) = 6 + 2 = 8$

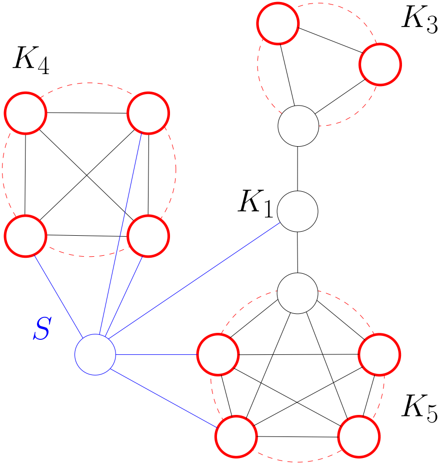
Theory: Kernelization Rule 8 in [EM18]



Theory: Kernelization Rule 8 in [EM18]

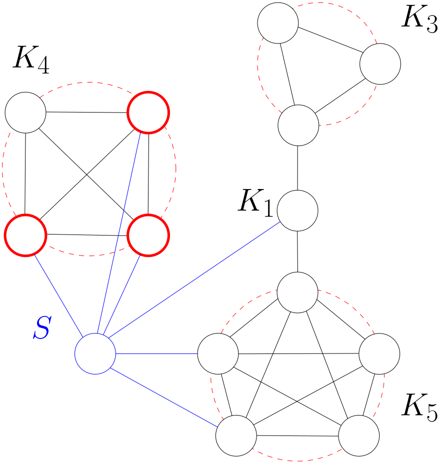


Theory: Kernelization Rule 8 in [EM18]



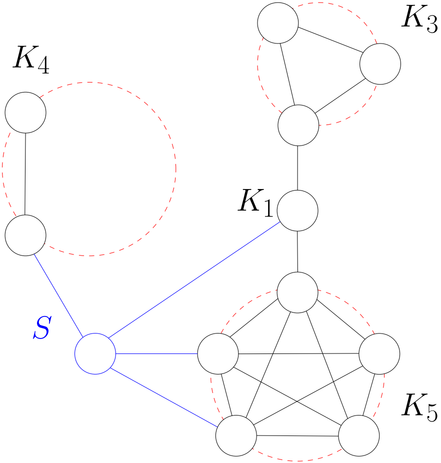
1. $N_G(x) \cap S = N_G(X) \cap S$
2. $|X| > \frac{|K|+|N_G(X) \cap S|}{2} \geq 1$

Theory: Kernelization Rule 8 in [EM18]



1. $N_G(x) \cap S = N_G(X) \cap S$ ✓
2. $|X| > \frac{|K| + |N_G(X) \cap S|}{2} \geq 1$ ✓

Theory: Kernelization Rule 8 in [EM18]



1. $N_G(x) \cap S = N_G(X) \cap S$
2. $|X| > \frac{|K|+|N_G(X) \cap S|}{2} \geq 1$

Theory $\rightarrow$ Practice

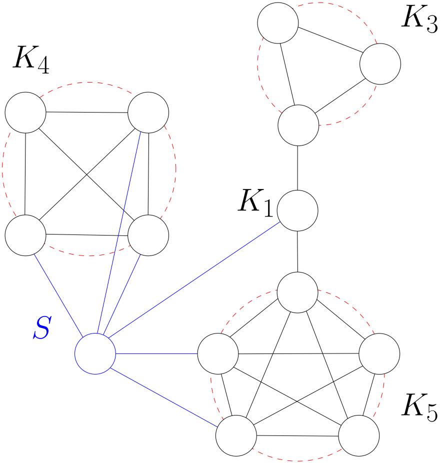
Karlsruhe Institute of Technology

- Weak-points in practice:
  - Reliance on clique-forest
  - Parameter k large in practice
    - **Kernel size $O(k)$ too large**
    - $O(k \cdot |E(G)|)$ time too slow

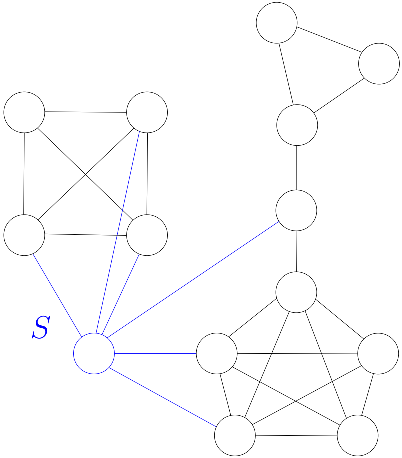
Our Contributions

- Implemented rules from [EM18]
- **Generalized existing kernelization rules**
  - Rules not dependent on a subgraph anymore
- **New kernelization rules**
- **Efficient implementation**
- Benchmark over a variety of instances

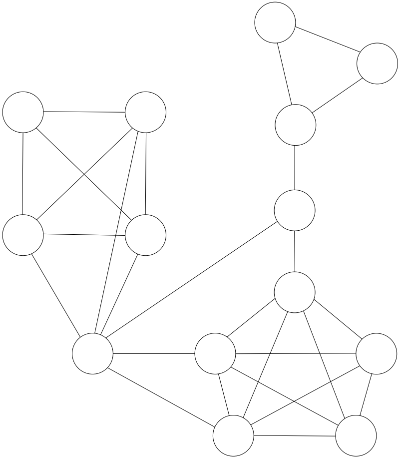
Rule Generalization: R8 – "Sharing Adjacencies"



Rule Generalization: R8 – "Sharing Adjacencies"

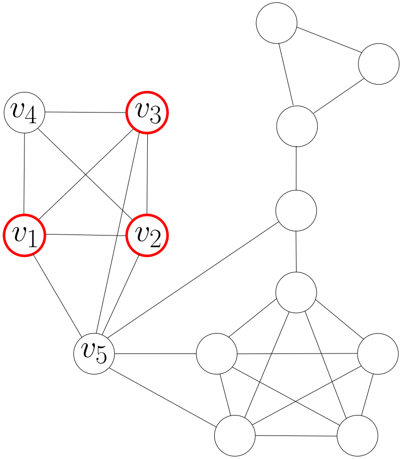


Rule Generalization: R8 – "Sharing Adjacencies"



1. $N_G(X) \cup X = N_G(x) \cup \{x\}$
2. $|X| > \max\{|N_G(X)|, 1\}$

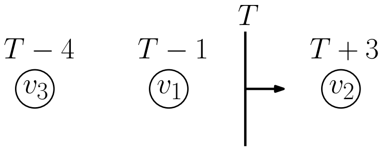
Rule Generalization: R8 – "Sharing Adjacencies"



1. $N_G(X) \cup X = N_G(x) \cup \{x\}$ ✓
2. $|X| > \max\{|N_G(X)|, 1\}$ ✓

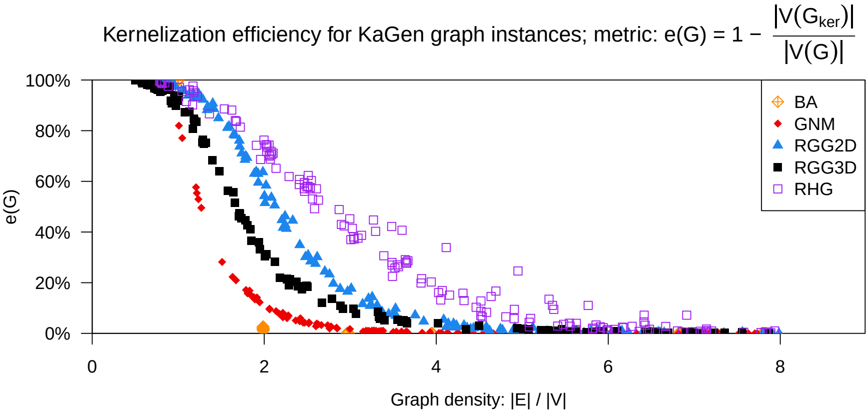
Techniques Used for Performance

- **Avoid time-intensive checks**
  - Vertex v internal in clique: $\forall w \in N_G(v) : Deg(v) \leq Deg(w)$
- **Speed up finding applications** of generalized rule 8 using Trie
- **Avoid checking the same** vertex twice
  - Keep timestamp T for each rule: All vertices $\leq T$ processed
  - Update vertex on change
  - $\rightarrow$ Heap



Experiments on Random Graphs

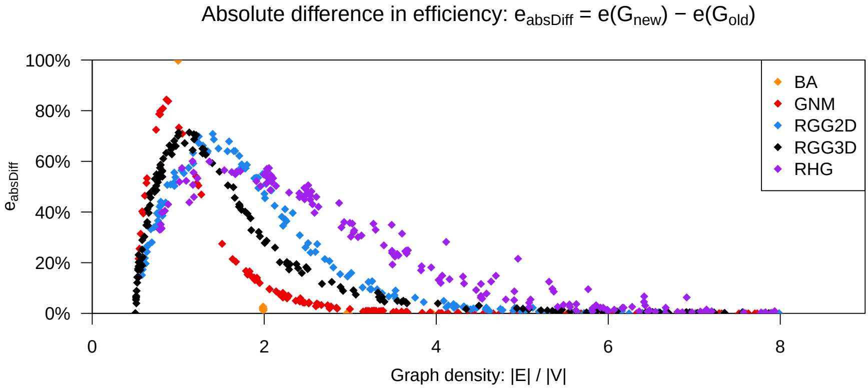
- Random graphs by KaGen, 150 per graph type. $|V| = 2048$
- **Total running time: 16 sec.** (68 min. with rules from [EM18]!)



Experiments on Random Graphs

Karlsruhe Institute of Technology

- Improvement over [EM18]. $|V| = 2048$
- Discrepancy between theory and practice



LocalSolver: Exact solutions

Name	$\lvert V(G)\rvert$	deg_{avg}	$e(G)$	$T_{\text{LS}}(G)$	$T_{\text{LS}}(G_{\text{ker}})$	
ca-CSphd	1882	0.92	0.99	24.07	0.32	[75.40]
ego-facebook	2888	1.03	1.00	20.09	0.09	[228.91]
ENZYMES_g295	123	1.13	0.86	1.22	0.33	[3.70]
road-euroroad	1174	1.21	0.79	-	-	-
bio-yeast	1458	1.34	0.81	-	-	-
rt-twitter-copen	761	1.35	0.85	-	834.71	[∞]
bio-diseasome	516	2.30	0.93	-	4.91	[∞]
ca-netscience	379	2.41	0.77	-	956.03	[∞]
soc-firm-hi-tech	33	2.76	0.36	4.67	1.61	[2.90]
imgseg_271031	900	1.14	0.99	12.33	0.22	[56.96]
imgseg_105019	3548	1.22	0.93	-	17.67	[∞]
imgseg_35058	1274	1.42	0.37	180.92	30.68	[5.90]
imgseg_374020	5735	1.52	0.82	1614.23	638.70	[2.53]
imgseg_106025	1565	1.68	0.68	25.97	-	[-∞]
g000302	317	1.50	0.21	0.63	0.54	[1.18]
g001918	777	1.59	0.12	1.72	1.42	[1.21]
g000981	110	1.71	0.28	10.73	4.73	[2.27]
g001207	84	1.77	0.19	1.23	0.14	[8.70]
g000292	212	1.80	0.03	0.39	0.43	[0.92]

Future Work

- Parallelism?
- Weighted kernelization?
- New kernelization rules?
- Hybrid approach: Use solver for kernelization?

Summary

- Previous work: Good in theory, not so good in practice
- **Sparse graphs highly reducible**
- **Fast implementation possible**
- **Significant benefits for Solvers**

References I

Francisco Barahona et al. “An application of combinatorial optimization to statistical physics and circuit layout design”. In: *Operations Research* 36.3 (1988), pp. 493–513.

Francisco Barahona. “On the computational complexity of Ising spin glass models”. In: *Journal of Physics A: Mathematical and General* 15.10 (1982), p. 3241.

Charles Chiang et al. “Fast and efficient bright-field AAPSM conflict detection and correction”. In: *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems* 26.1 (2007), pp. 115–126.

Michael Etscheid and Matthias Mnich. “Linear kernels and linear-time algorithms for finding large cuts”. In: *Algorithmica* 80.9 (2018), pp. 2574–2615.

References II

Frank Harary. "On the measurement of structural balance". In: *Behavioral Science* 4.4 (1959), pp. 316–323.

Frank Harary, Meng-Hiot Lim, and Donald C Wunsch. "Signed graphs for portfolio analysis in risk management". In: *IMA Journal of management mathematics* 13.3 (2002), pp. 201–210.