Finding cuts of bounded degree: complexity, FPT and exact algorithms, and kernelization

Guilherme C. M. Gomes

Universidade Federal de Minas Gerais

Ignasi Sau

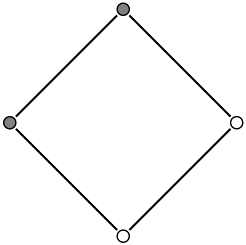
CNRS/LIRMM

Matching Cut

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.

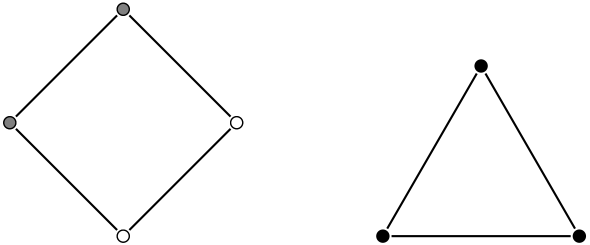
Matching Cut

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.



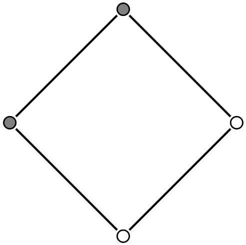
Matching Cut

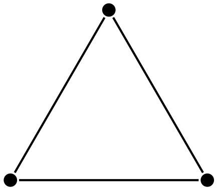
A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.

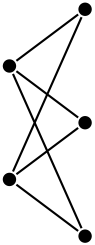


Matching Cut

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.

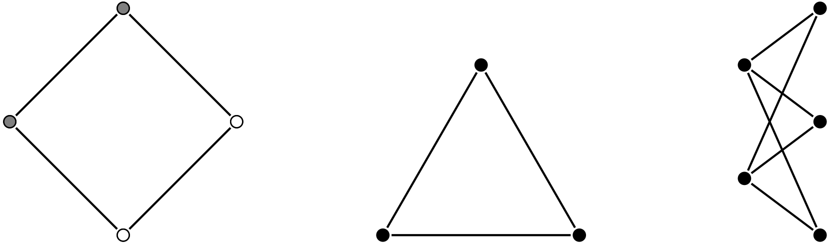






Matching Cut

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.



Ron L Graham. "On primitive graphs and optimal vertex assignments". In: *Annals of the New York academy of sciences* 175.1 (1970), pp. 170–186

Which graphs admit a matching cut?

Matching Cut and graph classes

Vasek Chvátal. "Recognizing decomposable graphs". In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53

MATCHING CUT and graph classes

Vasek Chvátal. "Recognizing decomposable graphs". In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53

Recognizing these graphs is NP-Hard even if $\Delta(G) = 4$...

Matching Cut and graph classes

Vasek Chvátal. “Recognizing decomposable graphs”. In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53

Recognizing these graphs is NP-Hard even if $\Delta(G) = 4$...

Paul S. Bonsma. “The complexity of the matching-cut problem for planar graphs and other graph classes”. In: *Journal of Graph Theory* 62.2 (2009), pp. 109–126

...G is planar...

Matching Cut and graph classes

Vasek Chvátal. "Recognizing decomposable graphs". In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53

Recognizing these graphs is NP-Hard even if $\Delta(G) = 4$...

Paul S. Bonsma. "The complexity of the matching-cut problem for planar graphs and other graph classes". In: *Journal of Graph Theory* 62.2 (2009), pp. 109–126

...G is planar...

Van Bang Le and Bert Randerath. "On stable cutsets in line graphs". In: *Theoretical Computer Science* 301.1-3 (2003), pp. 463–475

...or G is bipartite.

Algorithmic results for MATCHING CUT

Dániel Marx, Barry O'Sullivan, and Igor Razgon. "Treewidth Reduction for Constrained Separation and Bipartization Problems". In: *Proc. of the 27th International Symposium on Theoretical Aspects of Computer Science, (STACS)*. vol. 5. LIPIcs. 2010, pp. 561–572

FPT parameterized by the number of edges crossing the cut.

Algorithmic results for MATCHING CUT

Dániel Marx, Barry O'Sullivan, and Igor Razgon. "Treewidth Reduction for Constrained Separation and Bipartization Problems". In: *Proc. of the 27th International Symposium on Theoretical Aspects of Computer Science, (STACS)*. vol. 5. LIPIcs. 2010, pp. 561–572

FPT parameterized by the number of edges crossing the cut.

Dieter Kratsch and Van Bang Le. "Algorithms solving the matching cut problem". In: *Theoretical Computer Science* 609 (2016), pp. 328–335

FPT parameterized by vertex cover; $\mathcal{O}^*\left(2^{n/2}\right)$ exact exponential algorithm.

Algorithmic results for Matching Cut

N. R. Aravind, Subrahmanyam Kalyanasundaram, and Anjeneya Swami Kare. “On Structural Parameterizations of the Matching Cut Problem”. In: *Proc. of the 11th International Conference on Combinatorial Optimization and Applications (COCOA)*. vol. 10628. LNCS. 2017, pp. 475–482

FPT parameterized by treewidth, neighborhood diversity, or twin cover.

Algorithmic results for MATCHING CUT

Christian Komusiewicz, Dieter Kratsch, and Van Bang Le. "Matching Cut: Kernelization, Single-Exponential Time FPT, and Exact Exponential Algorithms". In: *Proc. of the 13th International Symposium on Parameterized and Exact Computation (IPEC)*. vol. 115. LIPIcs. 2018, 19:1–19:13

- FPT parameterized by distance to cluster, distance to co-cluster;
- Quadratic kernel for distance to cluster, linear kernel for distance to clique;
- No polynomial kernel for treewidth + number of crossing edges + maximum degree (unless NP $\subseteq$ coNP/poly).
- Exact exponential running in $\mathcal{O}^*(1.38^n)$.

Algorithmic results for MATCHING CUT

Christian Komusiewicz, Dieter Kratsch, and Van Bang Le. "Matching Cut: Kernelization, Single-Exponential Time FPT, and Exact Exponential Algorithms". In: *Proc. of the 13th International Symposium on Parameterized and Exact Computation (IPEC)*. vol. 115. LIPIcs. 2018, 19:1–19:13

- FPT parameterized by distance to cluster, distance to co-cluster;
- Quadratic kernel for distance to cluster, linear kernel for distance to clique;
- No polynomial kernel for treewidth + number of crossing edges + maximum degree (unless NP $\subseteq$ coNP/poly).
- Exact exponential running in $\mathcal{O}^*(1.38^n)$.

During their presentation, asked for results on the generalization of MATCHING CUT we discuss here.

Cuts and degree constraints

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.

Cuts and degree constraints

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.

Option 1

Look for a **bipartition** such that each vertex has at most $\boldsymbol{d}$ neighbors in the other part.

Cuts and degree constraints

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.

Option 1

Look for a **bipartition** such that each vertex has at most $\boldsymbol{d}$ neighbors in the other part.

Option 2

Look for a **$\boldsymbol{p}$-partition** such that each vertex has at most **one** neighbor outside its part.

Cuts and degree constraints

A matching cut is a bipartition of $V(G)$ such that each vertex has at most one neighbor in the other part.

Option 1

Look for a **bipartition** such that each vertex has at most $\boldsymbol{d}$ neighbors in the other part.

Option 2

Look for a **$\boldsymbol{p}$-partition** such that each vertex has at most **one** neighbor outside its part.

Option 3

Look for a **$\boldsymbol{p}$-partition** such that each pair of parts forms a matching cut.

Cuts and degree constraints

Option 1

Look for a **bipartition** such that each vertex has at most $\boldsymbol{d}$ neighbors in the other part.

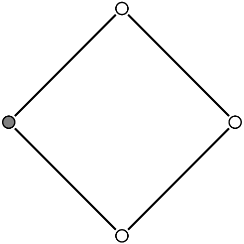
Some definitions

A bipartition (A, B) of $V(G)$ is a d-cut if and only if each vertex has at most d neighbors across the cut.

Some definitions

A bipartition (A, B) of $V(G)$ is a d-cut if and only if each vertex has at most d neighbors across the cut.

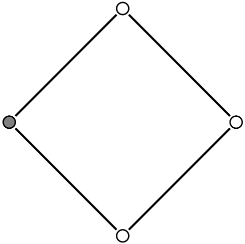
$d = 2$

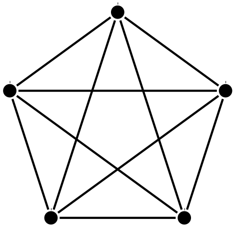


Some definitions

A bipartition (A, B) of $V(G)$ is a d-cut if and only if each vertex has at most d neighbors across the cut.

$d = 2$

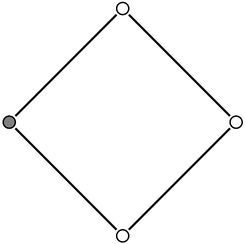


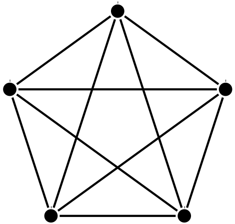


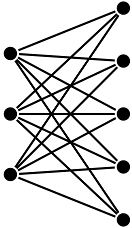
Some definitions

A bipartition (A, B) of $V(G)$ is a d-cut if and only if each vertex has at most d neighbors across the cut.

$d = 2$



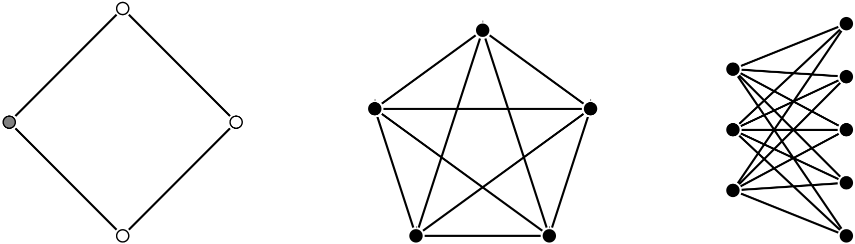




Some definitions

A bipartition (A, B) of $V(G)$ is a d-cut if and only if each vertex has at most d neighbors across the cut.

$d = 2$

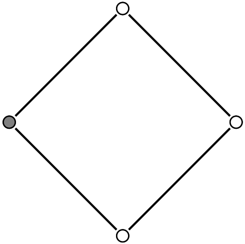


A set $S \subseteq V(G)$ is *monochromatic* if, in every d-cut (A, B), it holds that $S \subseteq A$ or $S \subseteq B$.

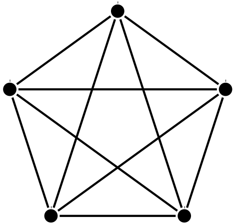
Some definitions

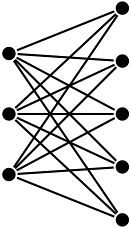
A bipartition (A, B) of $V(G)$ is a d-cut if and only if each vertex has at most d neighbors across the cut.

$d = 2$



$K_{2d+1} = K_5$



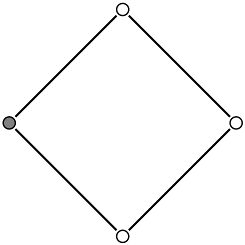


A set $S \subseteq V(G)$ is *monochromatic* if, in every d-cut (A, B), it holds that $S \subseteq A$ or $S \subseteq B$.

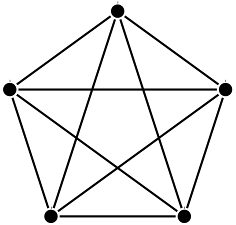
Some definitions

A bipartition (A, B) of $V(G)$ is a d-cut if and only if each vertex has at most d neighbors across the cut.

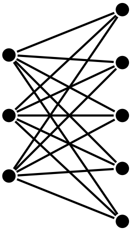
$d = 2$



$K_{2d+1} = K_5$



$K_{d+1,2d+1} = K_{3,5}$



A set $S \subseteq V(G)$ is *monochromatic* if, in every d-cut (A, B), it holds that $S \subseteq A$ or $S \subseteq B$.

Regular graphs are hard

Theorem

d-Cut on $(2d+2)$-regular graphs is NP-Hard.

Regular graphs are hard

Theorem

d-Cut on $(2d+2)$-regular graphs is NP-Hard.

3-Uniform Hypergraph Bicoloring

Instance: A hypergraph $\mathcal{H}$ with three vertices in each hyperedge.
Question: Can we 2-color $V(\mathcal{H})$ such that no hyperedge is monochromatic?

Regular graphs are hard

Theorem

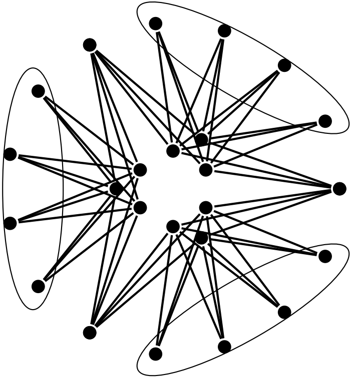
d-Cut on $(2d + 2)$-regular graphs is NP-Hard.

3-Uniform Hypergraph Bicoloring

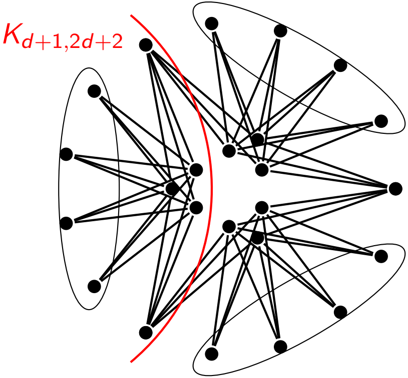
Instance: A hypergraph $\mathcal{H}$ with three vertices in each hyperedge.
Question: Can we 2-color $V(\mathcal{H})$ such that no hyperedge is monochromatic?

Heavily inspired on the reduction given by: Vasek Chvátal. "Recognizing decomposable graphs". In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53.

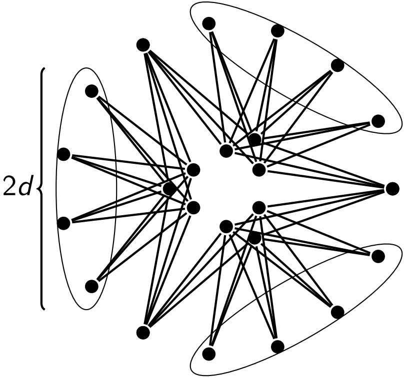
Monochromatic gadget: Spools



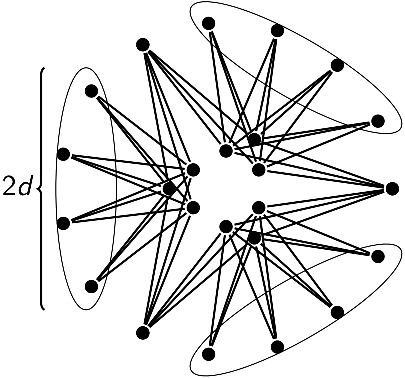
Monochromatic gadget: Spools



Monochromatic gadget: Spools

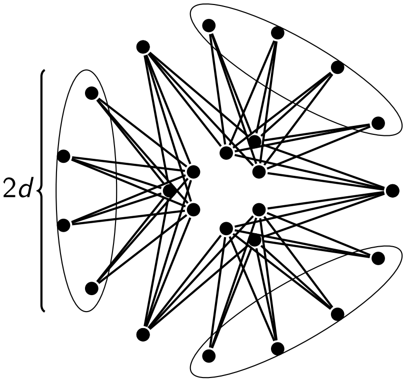


Monochromatic gadget: Spools



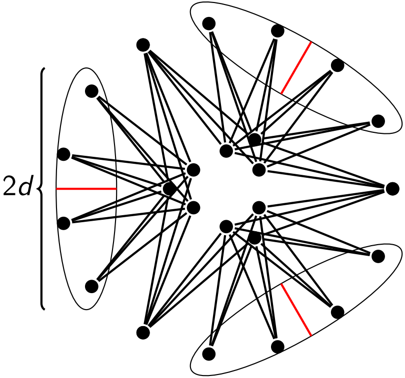
- Create one spool for each vertex v and one for each color. **Goal**: if there is no bicoloring of the hypergraph, the whole graph is monochromatic.

Monochromatic gadget: Spools



- Create one spool for each vertex v and one for each color. **Goal**: if there is no bicoloring of the hypergraph, the whole graph is monochromatic.
- Assign an unique label to each set of circled vertices.

Monochromatic gadget: Spools



- Create one spool for each vertex v and one for each color. **Goal**: if there is no bicoloring of the hypergraph, the whole graph is monochromatic.
- Assign an unique label to each set of circled vertices.
- Divide each of them in two equal sized sets. Add edges within and between these sets to encode the hyperedges and achieve regularity.

Why regularity is important

There is a very similar problem known as INTERNAL PARTITIONS, where we want a bipartition of $V(G)$ such that every vertex has at least half of its neighbors on its own part.

Why regularity is important

There is a very similar problem known as INTERNAL PARTITIONS, where we want a bipartition of $V(G)$ such that every vertex has at least half of its neighbors on its own part.

Amir Ban and Nati Linial. "Internal partitions of regular graphs". In: *Journal of Graph Theory* 83.1 (2016), pp. 5–18

Conjecture

For every r, there is a constant n_r such that every r-regular graph with at least n_r vertices has an internal partition (open since 2002). Known to hold for $r = 3, 4, 6$.

Why regularity is important

There is a very similar problem known as INTERNAL PARTITIONS, where we want a bipartition of $V(G)$ such that every vertex has at least half of its neighbors on its own part.

Amir Ban and Nati Linial. "Internal partitions of regular graphs". In: *Journal of Graph Theory* 83.1 (2016), pp. 5–18

Conjecture

For every r, there is a constant n_r such that every r-regular graph with at least n_r vertices has an internal partition (open since 2002). Known to hold for $r = 3, 4, 6$.

Improving upon our reduction would imply that the conjecture is false (or $\mathrm{P} = \mathrm{NP}$), while polynomial algorithms would likely yield a proof.

Some polynomial cases

Vasek Chvátal. “Recognizing decomposable graphs”. In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53

MATCHING CUT can be solved in polynomial time for graphs of maximum degree at most 3.

Some polynomial cases

Vasek Chvátal. "Recognizing decomposable graphs". In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53

MATCHING CUT can be solved in polynomial time for graphs of maximum degree at most 3.

Theorem

If $\Delta(G) \leq d + 2$, d-CUT can be solved in polynomial time.

Some polynomial cases

Vasek Chvátal. "Recognizing decomposable graphs". In: *Journal of Graph Theory* 8.1 (1984), pp. 51–53

MATCHING CUT can be solved in polynomial time for graphs of maximum degree at most 3.

Theorem

If $\Delta(G) \leq d + 2$, d-CUT can be solved in polynomial time.

Sketch of the proof: find a shortest cycle C. If there is some vertex with too many neighbors in C, we either find a shorter cycle, or we have that $d = 2$ and extend C to Q until $(Q, G - Q)$ is a d-cut or G is monochromatic.

Parameterized algorithms

Kernelization

Theorem

d-Cut does not admit a polynomial kernel when parameterized by treewidth, number of crossing edges and maximum degree, unless $\mathsf{NP} \subseteq \mathsf{coNP}/\mathsf{poly}$.

Kernelization

Theorem

d-$\textsc{Cut}$ *does not admit a polynomial kernel when parameterized by treewidth, number of crossing edges and maximum degree, unless* NP $\subseteq$ coNP/poly.

We show that d-$\textsc{Cut}$ (OR-)cross-composes onto itself.

Kernelization

Theorem

d-CUT does not admit a polynomial kernel when parameterized by treewidth, number of crossing edges and maximum degree, unless NP $\subseteq$ coNP/poly.

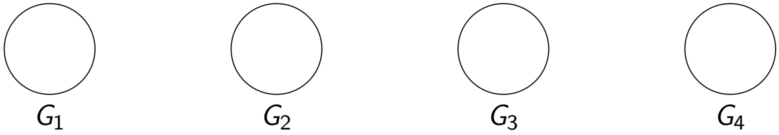
We show that d-CUT (OR-)cross-composes onto itself.

Kernelization

Theorem

d-CUT does not admit a polynomial kernel when parameterized by treewidth, number of crossing edges and maximum degree, unless NP $\subseteq$ coNP/poly.

We show that d-CUT (OR-)cross-composes onto itself.



Kernelization

Theorem

d-CUT does not admit a polynomial kernel when parameterized by treewidth, number of crossing edges and maximum degree, unless NP $\subseteq$ coNP/poly.

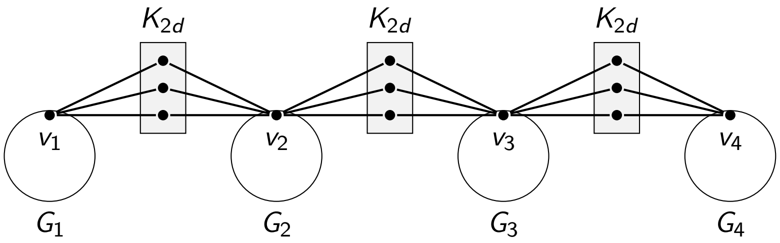
We show that d-CUT (OR-)cross-composes onto itself.

Kernelization

Theorem

d-Cut does not admit a polynomial kernel when parameterized by treewidth, number of crossing edges and maximum degree, unless NP $\subseteq$ coNP/poly.

We show that d-Cut (OR-)cross-composes onto itself.

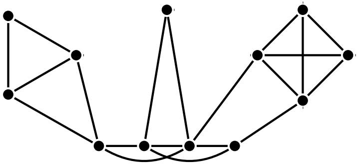


Distance to (co-)cluster

G is a cluster graph if each of its connected components is a clique (cluster).

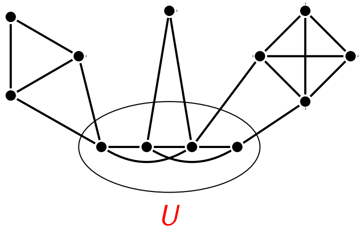
Distance to (co-)cluster

G is a cluster graph if each of its connected components is a clique (cluster).



Distance to (co-)cluster

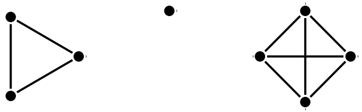
G is a cluster graph if each of its connected components is a clique (cluster).



The modulator $U \subset V(G)$ is a set such that $G - U$ is a cluster graph, with clusters $\{C_1, \ldots, C_\ell\}$.

Distance to (co-)cluster

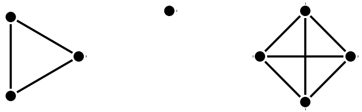
G is a cluster graph if each of its connected components is a clique (cluster).



The modulator $U \subset V(G)$ is a set such that $G - U$ is a cluster graph, with clusters $\{C_1, \ldots, C_\ell\}$.

Distance to (co-)cluster

G is a cluster graph if each of its connected components is a clique (cluster).



The modulator $U \subset V(G)$ is a set such that $G - U$ is a cluster graph, with clusters $\{C_1, \ldots, C_\ell\}$.

A co-cluster graph is the complement graph of a cluster graph.

The basics

Key idea

Partition U in monochromatic sets $\{U_1, \ldots, U_\ell\}$ and merge them until we get a kernel.

$N^{2d}(U_i)$

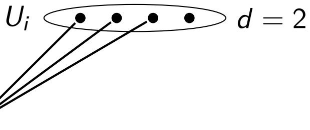
For each U_i, a vertex $v \in V(G) \setminus U$ is in $N^{2d}(U_i)$ if:

U_i  $d = 2$

$N^{2d}(U_i)$

For each U_i, a vertex $v \in V(G) \setminus U$ is in $N^{2d}(U_i)$ if:

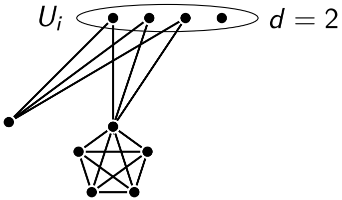
1. v has at least $d+1$ neighbors in U_i; or



$N^{2d}(U_i)$

For each U_i, a vertex $v \in V(G) \setminus U$ is in $N^{2d}(U_i)$ if:

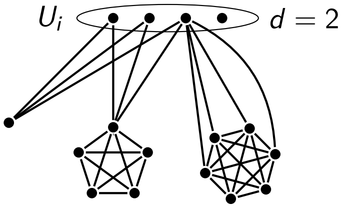
1. v has at least $d+1$ neighbors in U_i; or
2. v is in a cluster C of size at least $2d+1$ in $G-U$ such that there is some vertex of C with at least $d+1$ neighbors in U_i; or



$N^{2d}(U_i)$

For each U_i, a vertex $v \in V(G) \setminus U$ is in $N^{2d}(U_i)$ if:

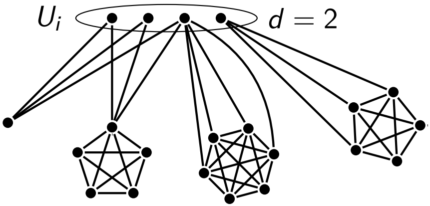
1. v has at least $d+1$ neighbors in U_i; or
2. v is in a cluster C of size at least $2d+1$ in $G-U$ such that there is some vertex of C with at least $d+1$ neighbors in U_i; or
3. v is in a cluster C and some vertex in U_i has $2d$ neighbors in C; or



$N^{2d}(U_i)$

For each U_i, a vertex $v \in V(G) \setminus U$ is in $N^{2d}(U_i)$ if:

1. v has at least $d+1$ neighbors in U_i; or
2. v is in a cluster C of size at least $2d+1$ in $G-U$ such that there is some vertex of C with at least $d+1$ neighbors in U_i; or
3. v is in a cluster C and some vertex in U_i has $2d$ neighbors in C; or
4. v is in a cluster C of size at least $2d+1$ and there is some vertex in U_i with $d+1$ neighbors in C.



The kernel

Theorem

*When parameterized by distance to cluster, d-*CUT *admits a polynomial kernel with* $\mathcal{O}\left(d^2\mathrm{dc}(G)^{2d+1}\right)$ *vertices that can be computed in* $\mathcal{O}\left(d^4\mathrm{dc}(G)^{2d+1}(n+m)\right)$ *time.*

Number of crossing edges

We can solve d-Cut in FPT time using the treewidth reduction technique.

Dániel Marx, Barry O'Sullivan, and Igor Razgon. "Treewidth Reduction for Constrained Separation and Bipartization Problems". In: *Proc. of the 27th International Symposium on Theoretical Aspects of Computer Science, (STACS)*. vol. 5. LIPIcs. 2010, pp. 561–572

Number of crossing edges

We can solve d-Cut in FPT time using the treewidth reduction technique.

Dániel Marx, Barry O'Sullivan, and Igor Razgon. "Treewidth Reduction for Constrained Separation and Bipartization Problems". In: *Proc. of the 27th International Symposium on Theoretical Aspects of Computer Science, (STACS)*. vol. 5. LIPIcs. 2010, pp. 561–572

We cast d-Cut as a separation problem on the line graph: for some pair $s, t \in V(L(G))$, we want a cutset with a maximum clique of size $\leq d$.

Number of crossing edges

We can solve d-Cut in FPT time using the treewidth reduction technique.

Dániel Marx, Barry O'Sullivan, and Igor Razgon. "Treewidth Reduction for Constrained Separation and Bipartization Problems". In: *Proc. of the 27th International Symposium on Theoretical Aspects of Computer Science, (STACS)*. vol. 5. LIPIcs. 2010, pp. 561–572

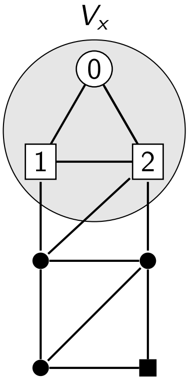
We cast d-Cut as a separation problem on the line graph: for some pair $s, t \in V(L(G))$, we want a cutset with a maximum clique of size $\leq d$.

Theorem

d-Cut is FPT paramterized by the number of crossing edges.

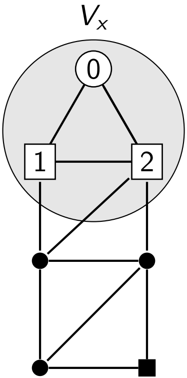
Treewidth

Dynamic programming on a nice tree decomposition.



Treewidth

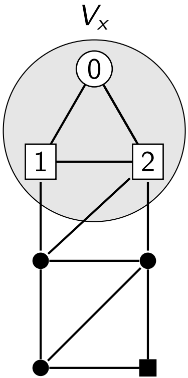
Dynamic programming on a nice tree decomposition.



- $S \subseteq V_x$ represents which (squared) elements of V_x are assigned to A.

Treewidth

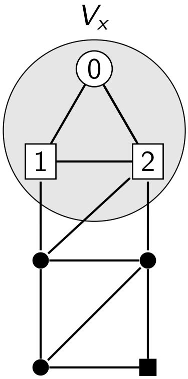
Dynamic programming on a nice tree decomposition.



- $S \subseteq V_x$ represents which (squared) elements of V_x are assigned to A.
- $\alpha \in \{0, \ldots, d\}^{|V_x|}$ stores how many neighbors each $v \in V_x$ has across the cut ***outside*** of V_x.

Treewidth

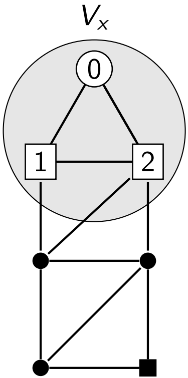
Dynamic programming on a nice tree decomposition.



- $S \subseteq V_x$ represents which (squared) elements of V_x are assigned to A.
- $\alpha \in \{0, \ldots, d\}^{|V_x|}$ stores how many neighbors each $v \in V_x$ has across the cut ***outside*** of V_x.
- $t = 1$ if A and B are non-empty.

Treewidth

Dynamic programming on a nice tree decomposition.



- $S \subseteq V_x$ represents which (squared) elements of V_x are assigned to A.
- $\alpha \in \{0, \ldots, d\}^{|V_x|}$ stores how many neighbors each $v \in V_x$ has across the cut ***outside*** of V_x.
- $t = 1$ if A and B are non-empty.
- $f_x(S, \alpha, t) = 1$ iff the subtree rooted at bag x has a partition that satisfies all of the above.

Treewidth

Theorem

For every $d \geq 1$, d-CUT can be solved in $\mathcal{O}^*\left(2^{\mathrm{tw}(G)}(d+1)^{2\mathrm{tw}(G)}\right)$.

Treewidth

Theorem

For every $d \geq 1$, d-Cut can be solved in $\mathcal{O}^*\left(2^{\mathrm{tw}(G)}(d+1)^{2\mathrm{tw}(G)}\right)$.

Corollary

Matching Cut *can be solved in $\mathcal{O}^*\left(8^{\mathrm{tw}(G)}\right)$. (Improvement upon $12^{\mathrm{tw}(G)}$).*

N. R. Aravind, Subrahmanyam Kalyanasundaram, and Anjeneya Swami Kare. "On Structural Parameterizations of the Matching Cut Problem". In: *Proc. of the 11th International Conference on Combinatorial Optimization and Applications (COCOA)*. vol. 10628. LNCS. 2017, pp. 475–482

Other results

Theorem

d-$\textsc{Cut}$ *can be solved in* $\mathcal{O}^*\left((d+1)^{\mathrm{dc}(G)}\right)$ *time.*

Other results

Theorem

d-CUT can be solved in $\mathcal{O}^*\left((d+1)^{\mathrm{dc}(G)}\right)$ time.

Theorem

d-CUT can be solved in $\mathcal{O}^*\left((d+1)^{\mathrm{d\overline{c}}(G)}\right)$ time.

Open Problems

Settle the complexity for graphs of maximum degree between $d+3$ and $2d+1$. We guess most of these should be solvable in polynomial time, but right now we have no idea how to tackle this problem.

Open Problems

Settle the complexity for graphs of maximum degree between $d+3$ and $2d+1$. We guess most of these should be solvable in polynomial time, but right now we have no idea how to tackle this problem.

Decide if the exponential dependency on d is necessary, or if we can restrict its influence to a polynomial factor.

Open Problems

Settle the complexity for graphs of maximum degree between $d + 3$ and $2d + 1$. We guess most of these should be solvable in polynomial time, but right now we have no idea how to tackle this problem.

Decide if the exponential dependency on d is necessary, or if we can restrict its influence to a polynomial factor.

Explore the other generalizations of MATCHING CUT, in particular Option 3, which seems considerably harder than the other two.

Option 3

Look for a ***p*-partition** such that each pair of parts forms a matching cut.

Thank you!