



Matching Cut

A Parameterized Approach

Project Report By
Hemanth Kumar J
B181004CS

Reference Paper :

Komusiewicz, Christian & Kratsch, Dieter & Le, Van Bang. (2019). Matching Cut: Kernelization, Single-Exponential Time FPT, and Exact Exponential Algorithms. Discrete Applied Mathematics. 283. 10.1016/j.dam.2019.12.010.

[Matching Cut: Kernelization, Single-Exponential Time FPT, and Exact Exponential Algorithms](#)



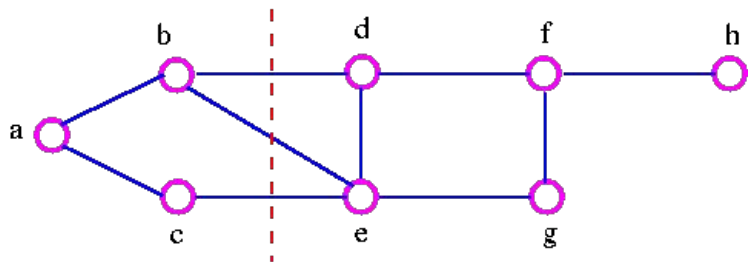
Problem Statement

A cut is a vertex partition (A, B) where $A \neq \emptyset$, $B \neq \emptyset$, $A \cap B = \emptyset$ and $A \cup B = V$ for a graph $G(V, E)$.

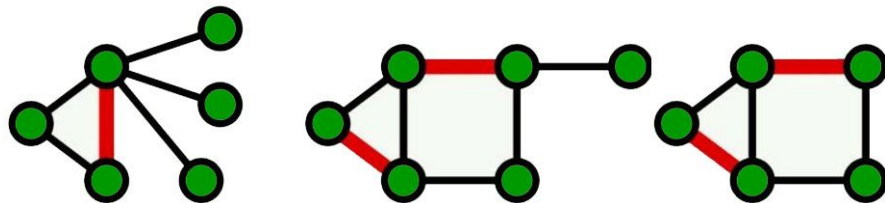
An edge cut is the set of all the edges with one endpoint in A and another in B , denoted by $E(A, B)$.

A matching is an edge set $M \subseteq E$ such that no two edges $e_i, e_j \in M$ shares any endpoint, A matching cut is an edge cut which is also a matching.

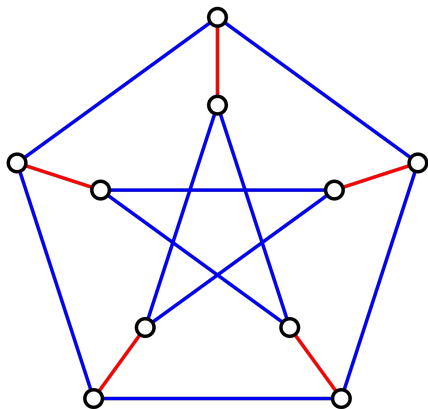
The MATCHING CUT problem is to decide if a given undirected graph G has a matching cut or not.



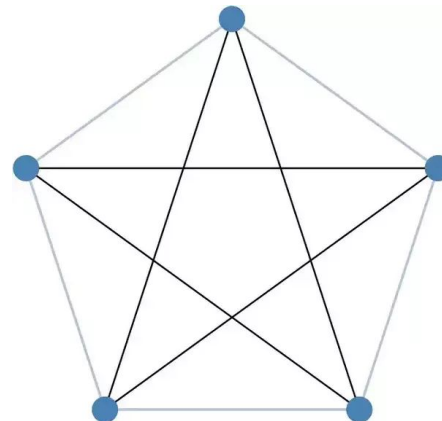
Example of Edge cut



Example of Matching



Example of YES instance of Matching cut



Example of NO instance of Matching cut



Classical Complexity

The matching cut problem is proved to be NP Complete in specific graphs such as planar graphs of maximum degree 4, showing that matching cut is NP Hard.

Furthermore, we can verify a matching cut in polynomial time. Therefore Matching Cut problem is NP-Complete



Optimization version

The optimization version of the matching cut problem is

Given an undirected graph $G(V, E)$, is there a Matching Cut of size at most k



Parameterized version

The FPT version of the matching cut problem is

Given an undirected graph $G (V, E)$, is there a Matching Cut Parameterized by **Distance to cluster** $dc(G)$

Cluster graph

A Cluster graph is a disjoint set of clusters (cliques).

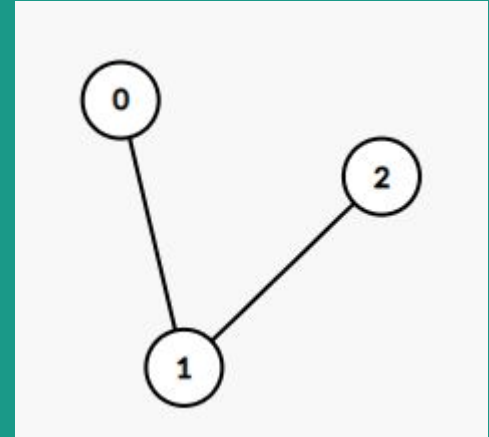
Distance to cluster, $dc(G)$ is the minimum number of vertices that can be removed from a graph G such that the resultant graph is a cluster graph



Approximation Algorithm For Cluster Vertex Deletion

Given a graph $G (V, E)$, if there is a P3 sub graph in G , delete all the 3 vertices of it.

Results in a 3-factor approximation algorithm for Cluster Vertex Deletion



P3 Subgraph

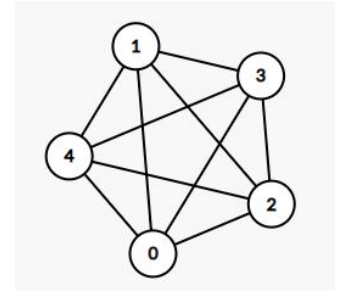
Monochromatic property

A subset $U \subseteq V$ is monochromatic if for every matching cut, $U \subseteq A$ or $U \subseteq B$.

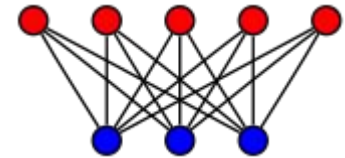
Monochromatic property is hereditary.

A complete subgraph K_n is monochromatic if $n = 1$ or $n \geq 3$

A complete bipartite subgraph $K_{n,m}$ is monochromatic if $n \geq 3$ and $m \geq 2$



A K_5 complete subgraph



A Complete bipartite subgraph with $n = 3$ and $m = 5$



Kernelization over $dc(G)$

The parameters that can be used size of the matching cut, tree width, and maximum degree of G do not accommodate a polynomial kernel: <https://arxiv.org/abs/1011.4224> while when parameterized by $dc(G)$ we get a polynomial kernel

We start with the vertex set $U = \{U_1, U_2, \dots, U_n\}$ from Cluster Vertex Deletion.

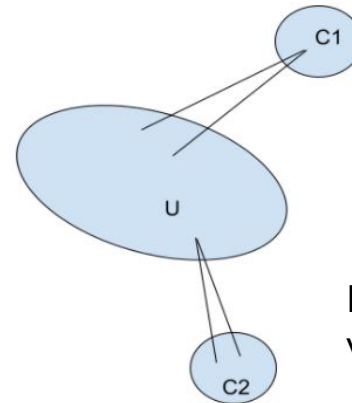
Merging 2 sets, U_i and U_j , is considered safe if they are monochromatic

Reduction Rule 1


If $V \setminus U$ contains a degree 1 vertex or a cluster C such that $(V \setminus C, C)$ is a matching cut, it is a YES instance

The resultant graph will have clusters such that

- At least 1 vertex in C will have 2 neighbours in U or
- At least 1 vertex in U will have 2 neighbours in C



Resultant graph
visualisation



For each U_i , we define $N^2(U_i)$ the set of vertices $v \in V \setminus U$ which follows at least one of the 3 properties:

- v has 2 neighbours in U_i ,
- v is in a cluster of size at least 3 in $G - U$ that contains a vertex that has 2 neighbours in U_i ,
- v is in a cluster C in $G - U$ and some vertex in U_i has 2 neighbours in C .

Proposition 1

$U_i \cup N^2(U_i)$ is monochromatic



A vertex is ambiguous if it has neighbours in U_i and U_j where $i \neq j$.

A cluster having at least 1 ambiguous vertex is ambiguous

A cluster is fixed if it is contained in $N^2(U_i)$ for some U_i

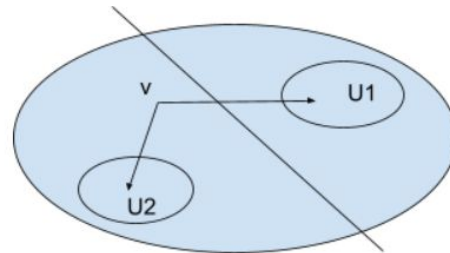
Proposition 2

After applying Rule 1, every non-edge cluster in G is ambiguous or fixed (can be both)



Reduction Rule 2

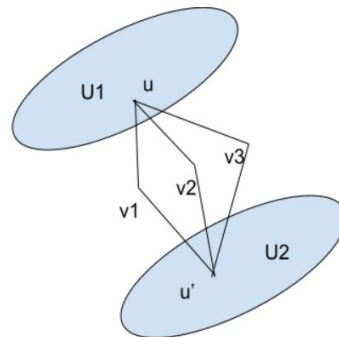
If there is a vertex v in $N^2(U_i)$ and $N^2(U_j)$ for $i \neq j$, merge U_i and U_j



Negative example of Rule 2

Reduction Rule 3

If there are 3 vertices v_1, v_2, v_3 in V that has 2 common neighbours $u \in U_i$ and $u' \in U_j$, $i \neq j$ then U_i and U_j are monochromatic, merge them



Rule 3 visualization



Reduction Rule 4

If 2 clusters are contained in the same $N^2(U_i)$, add edges between them.

After application of rules 1- 4, the number of fixed clusters is $O(|U|)$



Reduction Rule 5

If there is a cluster C with more than 3 vertices that contains a vertex v with no neighbours in U , remove v



Reduction Rule 6

If there is a cluster C with at least 3 vertices, and a monochromatic set U_i where $C \subseteq N^2(U_i)$, remove all edges between them and choose an arbitrary vertex $u \in U_i$ and 2 vertices $v_1, v_2 \in C$, add edges $\{u, v_1\}$ and $\{u, v_2\}$.

If $|U_i| = 2$, add an edge between $u' \in U_i \setminus \{u\}$ and $v_3 \in C \setminus \{v_1, v_2\}$.

If $|U_i| > 2$, make U_i a clique

After applying rules 1 to 6, G now is reduced to having

- $O(|U|^2)$ ambiguous vertices and
- $O(|U|^2)$ nonedge clusters each containing $O(|U|)$ vertices



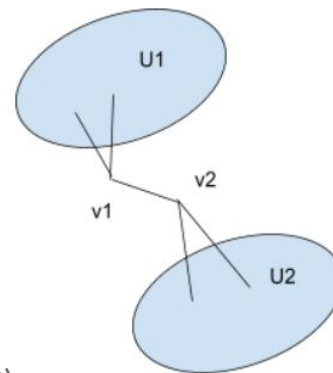
A single edge cluster is simple if for an edge cluster $\{u, v\}$, u has neighbours only in U_i and v has neighbours only in U_j

Reduction Rule 7

If there is a simple edge cluster $\{u, v\}$, remove u and v from G

Each edge cluster has at least 1 ambiguous vertex and therefore number of edge clusters is $O(dc(G)^2)$

After applying rules 1 to 7, we obtain a kernel with $O(dc(G)^3)$ vertices, $O(dc(G)^2)$ clusters, each containing $O(dc(G))$ vertices.



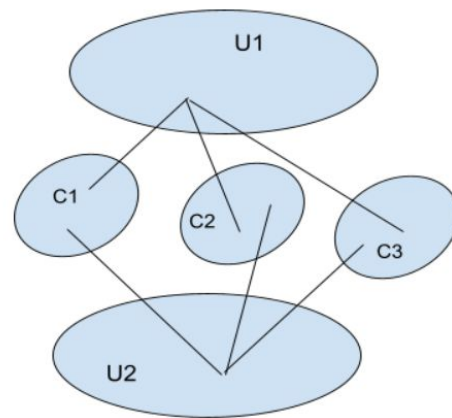
Rule 7 Visualisation



Reduction Rule 8

If there are 2 vertices $u \in U_i$ and $u' \in U_j$, $i \neq j$, and 3 non edge clusters C_1, C_2 and C_3 such that u and u' has at least 1 neighbour in each of these clusters, merge U_i and U_j .

After applying rules 1 to 8, G now is reduced to having $O(dc(G)^2)$ vertices



Rule 8 Visualization



Time Complexity

All the rules can be tested in $O(dc(G)^{2*}(n+m))$ time and applied in $O(n)$ time

Each rule can be applied at most $O(dc(G) * n)$ time and $N^2(U_i)$ is calculated after each rule, taking a running time of $O(dc(G) * (n + m))$

Total time complexity is $O(dc(G)^{3*} n * (n + m))$ or $O^*(dc(G)^3)$



Questions?



—

Thank You