Hidden Bilevel Structures in Graph Disconnection Problems

Fabio Furini

SAPIENZA UNIVERSITÀ DI ROMA

Outline

References

[1] Denis Cornaz, F.F., Mathieu Lacroix, Enrico Malaguti, A. Ridha Mahjoub, and Sébastien Martin.
The Vertex k-Cut Problem.
Discrete Optimization, 31:8–28, 2019.

[2] F.F., Ivana Ljubić, Enrico Malaguti and Paolo Paronuzzi
On integer and bilevel formulations for the k-vertex cut problem.
Mathematical Programming Computation, 12(2):133–164, 2020.

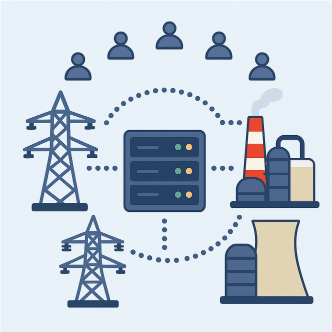
[3] F.F., Ivana Ljubić, Enrico Malaguti and Paolo Paronuzzi
Casting Light on the Hidden Bilevel Combinatorial Structure of the Capacitated Vertex Separator Problem.
Operations Research, 70(4):2399-2420, 2022.

Problem definition and classical ILP models

Graph disconnection problems

Graph disconnection optimization problems belong to the broader family of **Critical Node Detection Problems**, which arise in several real-world applications:

- **Network Resilience**
  - Identify critical nodes whose failure would fragment the network.
- **Infrastructure Protection**
  - Prevent cascading failures in power grids or transportation systems.
- **Containment Strategies**
  - Block the spread of disease or misinformation by fragmenting social networks.
- **Security and Surveillance**
  - Strategically disconnect regions in adversarial scenarios.



The k-vertex cut problem

- Given a graph $G = (V, E)$, a subgraph is a graph $G' = (V', E')$ such that $V' \subseteq V$ (subset of vertices) and $E' \subseteq E$ (subset of edges), where every edge $e \in E'$ has both endpoints in V'. A connected component is a connected subgraph.
- A vertex cut is a set of vertices whose removal disconnects the graph into several connected components. If the number of resulting connected components is at least k, the vertex cut is called a k-vertex cut.

Definition

Given a graph $G = (V, E)$, a positive weight w_v for each vertex $v \in V$, and an integer $k \geq 2$, the k-vertex cut problem (k-VCP) asks to find a k-vertex cut of minimum total weight.

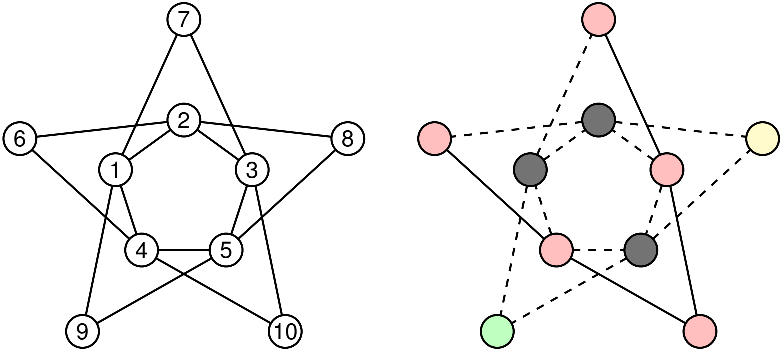
- By reduction from the vertex k-multiclique problem[1] on the complement graph, the k-VCP is NP-hard for any fixed $k \geq 3$.
- For $k = 2$, the problem is solvable in polynomial time [2].

[1] A vertex k-multiclique is a subset of vertices that can be partitioned into k non-empty subsets such that every pair of vertices belonging to different subsets is adjacent.

[2] Since it is equivalent to calculating the vertex-connectivity of the graph.

Example of k-vertex cuts (1/2)

- Let's consider the following graph with 10 vertices and 15 edges, all having unit weight, and let $k = 3$.



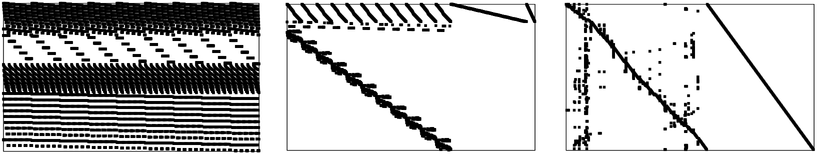
- An optimal 3-vertex cut (shown on the right) is represented by the black subset of vertices $\{1, 2, 5\}$.

Another important application

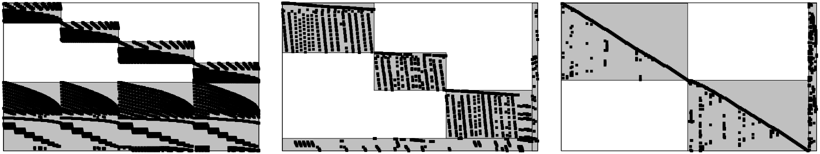
- Beyond network analysis, the k-vertex cut problem also arises in **matrix decomposition** for solving systems of equations/constraints via **parallel computing**.
- Given a system of equations with coefficient matrix A, we define its intersection graph:
  - One vertex per column/variable,
  - An edge between two vertices if and only if there exists a row in which both variables have a nonzero coefficient.
- To solve the system in parallel, the equations are partitioned into k subsystems.
  - These subsystems are solved separately,
  - Their solutions must then be merged consistently.
- **Goal:** Minimize the number of variables shared across subsystems.
  - This problem is the k-vertex cut problem on the intersection graph.

Detecting matrix decomposition

- Constraint matrices of MIPlib instances:



(a) original 10teams instance **(b)** original fiber instance **(c)** original timtab1 instance



(d) 10teams, detected structure **(e)** fiber, detected structure **(f)** timtab1, detected structure

Connected components, subsets of vertices and k-vertex cuts

Observation

A subset of vertices $V_0 \subseteq V$ is a k-vertex cut, if and only if the remaining vertices $V \setminus V_0$ can be partitioned into k non-empty pairwise disconnected [3] subsets of vertices, denoted

$$V_1, V_2, \dots, V_k$$

- Accordingly, there is a one-to-one correspondence between feasible solutions of the k-vertex cut problem and k-vertex-disjoint subsets of vertices that are pairwise disconnected.
- A generic subset of vertices may induce multiple connected components.

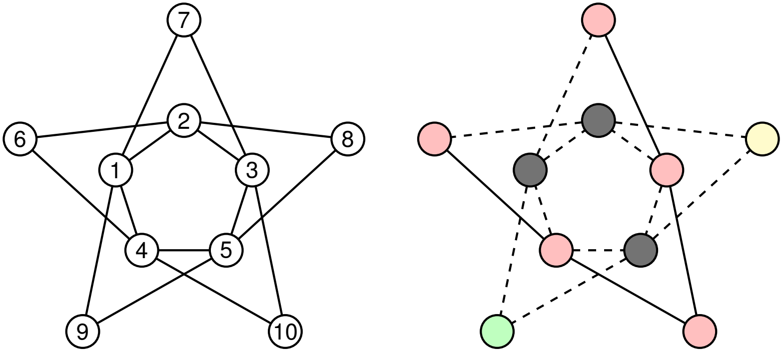
Observation

A graph G admits a k-vertex cut if and only if $\alpha(G) \geq k$, where $\alpha(G)$ is the stability number of the graph.

[3] i.e., there is no edge between any two subsets

Example of k-vertex cuts (2/2)

- Let's consider the following graph with 10 vertices and 15 egdes, all having unit weight, and let $k = 3$.



- In this case the number of subsets coincides with the number of connected components. We have:

$$V_0 = \{1, 2, 5\} \quad \text{and} \quad V_1 = \{3, 4, 6, 7, 10\}, V_2 = \{9\}, V_3 = \{8\}$$

A first ILP model (1/2)

Using the binary variables:

$$y_{vi} = \begin{cases} 1 & \text{if vertex } v \text{ is in subset } i, \\ 0 & \text{otherwise,} \end{cases} \qquad v \in V, i \in \underbrace{\{1, 2, \dots, k\}}_{=K},$$

the compact ILP model for k-vertex cut problem (called *COMP*) is:

$$\sum_{v \in V} w_v - \max \sum_{i \in K} \sum_{v \in V} w_v \, y_{vi} \tag{0.1}$$

$$\sum_{i \in K} y_{vi} \leq 1, \qquad v \in V, \tag{0.2}$$

$$y_{ui} + \sum_{j \in K \setminus \{i\}} y_{vj} \leq 1, \qquad i, j \in K, \{u, v\} \in E, \tag{0.3}$$

$$\sum_{v \in V} y_{vi} \geq 1, \qquad i \in K, \tag{0.4}$$

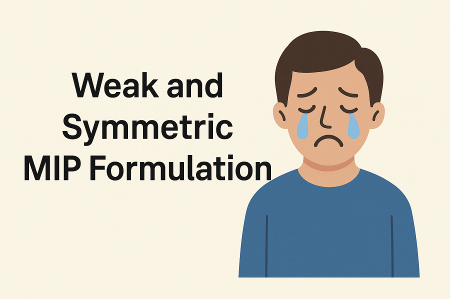
$$y_{vi} \in \{0, 1\}, \qquad i \in K, v \in V. \tag{0.5}$$

A first ILP model (2/2)

- The *COMP* model has two principal drawbacks:
  - It suffers from a weak LP relaxation: an optimal LP solution with objective value **zero** can be obtained by setting

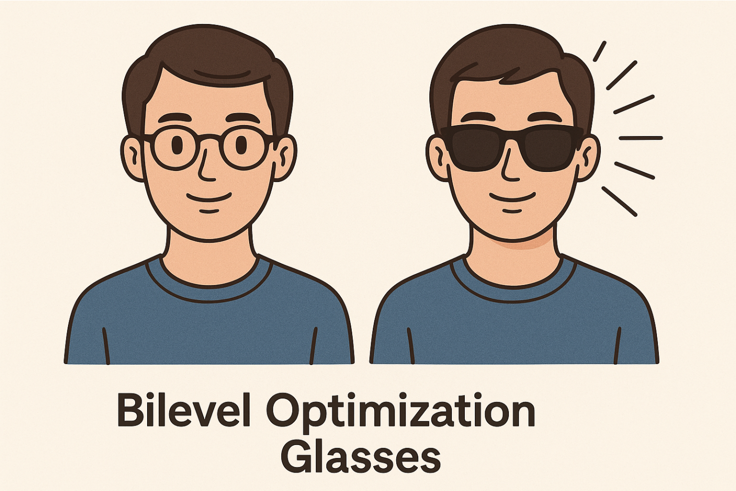
$$y_{vi} = \frac{1}{k}, \quad v \in V,\ i \in K.$$

  - It also exhibits symmetries: permutations of the k subsets yield equivalent LP and ILP solutions.



Bilevel Optimization and new ILP Models

A Bilevel Optimization point of view (1/5)



A Bilevel Optimization point of view (2/5)

- A **Bilevel Optimization** is a hierarchical decision-making framework involving two levels:
  - The **leader** (upper level) makes a decision, which defines the feasible region and objective of the **follower**
  - The **follower** (lower level) who then solves an optimization problem in response.
- We consider the special case where:
  - The **leader** (upper level) anticipates the optimal response of the **follower** (lower level), and selects a strategy that induces this optimal reaction
  - The **leader** (upper level) then solve the overall problem by accounting for the **follower**'s best reply (lower level).

A Bilevel Optimization point of view (3/5)

We now present a **bilevel optimization Perspective** for the k-vertex cut problem, which enables a valid ILP formulation in the natural space of the variables associated to the vertices.

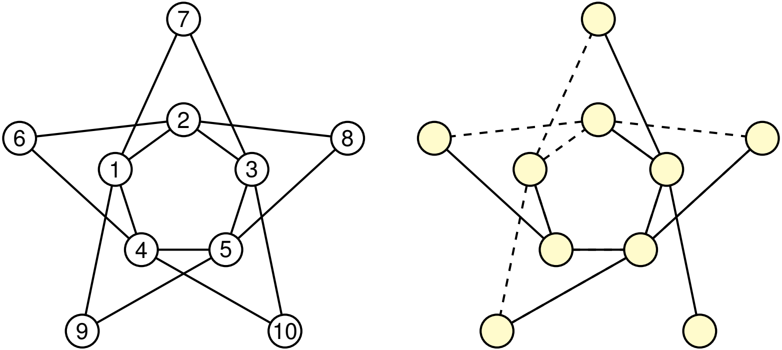
Bilevel Optimization Interpretation:

- **Leader (Upper Level)**:
  - Chooses a set of vertices to delete from the graph (called a strategy).
- **Follower (Lower Level)**:
  - Computes a maximum-size acyclic subgraph (i.e., a forest) in the remaining graph.
  - The solution is feasible for the leader if and only if the subgraph has at least k connected components.

The leader seeks a minimum-weight set of deleted vertices such that the follower's optimal response satisfies the component components requirement.

Related properties of graphs (1/4)

- A graph $G = (V, E)$ is **connected** if and only if it contains a spanning tree, i.e., a spanning[4] acyclic subgraph of G with $|V| - 1$ edges.



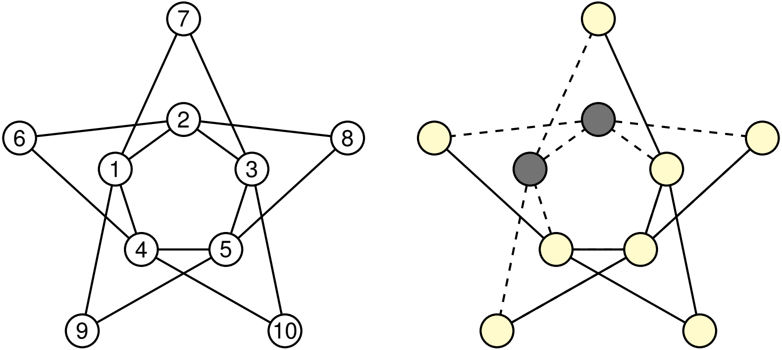
- The spanning tree has $|V| - 1 = 10 - 1 = 9$. This property admits the following generalization

4 including all vertices of the original graph

Related properties of graphs (2/4)

Observation

A graph $G = (V, E)$ has at least k connected components if and only if every *acyclic subgraph* *of G contains at most $|V| - k$ edges.*

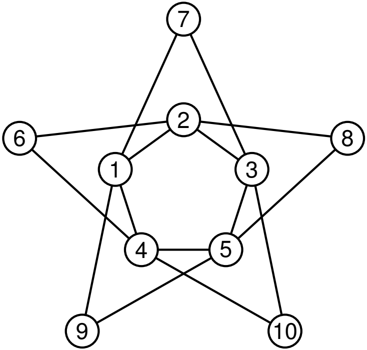


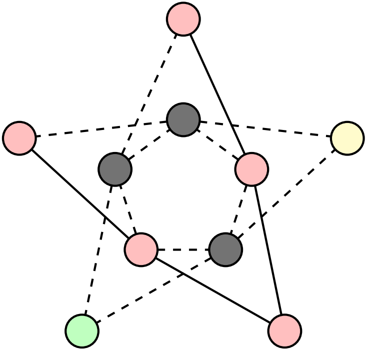
- Let $k = 3$, the graph on the right has $|V| = 8$ and $|E| = 8 > |V| - 3 = 8 - 3 = 5$. It does not contain 3 connected components.

Related properties of graphs (3/4)

Observation

A graph $G = (V, E)$ has at least k connected components if and only if every acyclic subgraph of G contains at most $|V| - k$ edges.





- Let $k = 3$, the graph on the right has $|V| = 7$ and $|E| = 4 \leq |V| - 3 = 7 - 3 = 4$. It contains 3 connected components.

Related properties of graphs (4/4)

Observation

A graph $G = (V, E)$ has at least k connected components if and only if every acyclic subgraph of G contains at most $|V| - k$ edges.

- This yields a Bilevel Optimization interpretation of the ***k*-vertex cut problem**:
  - The **leader** chooses a subset of vertices $V_0 \subseteq V$ to delete.
  - The **follower** builds a maximum acyclic subgraph (i.e., a forest) on the remaining graph $G[V \setminus V_0]$.
  - The leader's solution is feasible if the resulting subgraph has at most $|V| - |V_0| - k$ edges.

Observation

A subset of $V_0 \subseteq V$ is a k-vertex cut, if and only if the maximum number of edges of every acyclic subgraph of the remaining graph $G[V \setminus V_0]$ is at most $|V| - |V_0| - k$.

A Bilevel Optimization point of view (4/5)

The BILEVEL ILP model for the k-vertex cut problem (called *BILP*) is:

$$\min \left\{ \sum_{v \in V} w_v \, x_v : \quad \Phi(x) \leq |V| - \sum_{v \in V} x_v - k, \;\; x_v \in \{0,1\}, \; v \in V \right\}. \quad (0.1)$$

Using the binary variables:

$$y_{uv} = \begin{cases} 1 & \text{if edge } \{u,v\} \text{ is in the acyclic subgraph,} \\ 0 & \text{otherwise,} \end{cases} \qquad \{u,v\} \in E,$$

and, given a leader strategy $\tilde{\boldsymbol{x}} \in \{0,1\}^{|V|}$, the follower's subproblem is:

$$\Phi(\tilde{\boldsymbol{x}}) = \max \sum_{uv \in E} y_{uv} \quad (0.2)$$

$$\sum_{\substack{\{u,v\} \in E: \\ u,v \in S}} y_{uv} \leq |S| - 1, \qquad S \subseteq V, |S| \geq 3, \quad (0.3)$$

$$y_{uv} \leq 1 - \tilde{x}_u, \;\; y_{uv} \leq 1 - \tilde{x}_v, \qquad \{u,v\} \in E, \quad (0.4)$$

$$y_{uv} \in \{0,1\}, \qquad \{u,v\} \in E. \quad (0.5)$$

A Bilevel Optimization point of view (5/5)

- The follower's subproblem is **reformulated** so that its feasible region is independent of the leader.

Observation

The follower subproblem can be equivalently restated as

$$\Phi(\tilde{\boldsymbol{x}}) = \max \sum_{\{u,v\} \in E} y_{uv} \left(1 - \tilde{x}_u - \tilde{x}_v\right) \tag{0.6}$$

$$\sum_{\substack{\{u,v\} \in E: \\ u,v \in S}} y_{uv} \leq |S| - 1, \quad S \subseteq V, |S| \geq 3 \tag{0.7}$$

$$y_{uv} \in \{0,1\}, \qquad \{u,v\} \in E. \tag{0.8}$$

- A **single-level reformulation** is derived with an exponential number of constraints, each associated with an extreme point of the follower's polytope.

A second ILP model (1/2)

- Let $\mathcal{AS}(G)$ denote the set of all acyclic subgraphs of G corresponding to extreme points of follower polytope.
- The non-linear constraints of the BILP model can be then replaced by the following exponential family of linear constraints:

$$\sum_{\{u,v\}\in E(\mathcal{G})} \big(1 - x_u - x_v\big) \leq |V| - \sum_{v\in V} x_v - k, \quad \mathcal{G} \in \mathcal{AS}(G), \qquad (0.9)$$

  where $E(\mathcal{G})$ is the set of edges of the acyclic subgraph $\mathcal{G}$.
- Since every vertex $v \in V(\mathcal{G})$ is counted $\deg_{\mathcal{G}}(v)$ many times in the above constraints , they can also be restated as:

$$\sum_{v\in V} \big(\deg_{\mathcal{G}}(v) - 1\big) x_v \geq k + |E(\mathcal{G})| - |V|, \quad \mathcal{G} \in \mathcal{AS}(G). \qquad (0.10)$$

A second ILP model (1/2)

- Let $\mathcal{SAS}(G)$ denote the set of all spanning acyclic subgraphs of G corresponding to extreme points of follower polytope.
- Recalling that:

$$x_v = \begin{cases} 1 & \text{if vertex } v \text{ is in the } k\text{-vertex cut,} \\ 0 & \text{otherwise,} \end{cases} \qquad v \in V,$$

Proposition

The following natural ILP model is a valid formulation (called (NAT)) for the k-vertex cut problem:

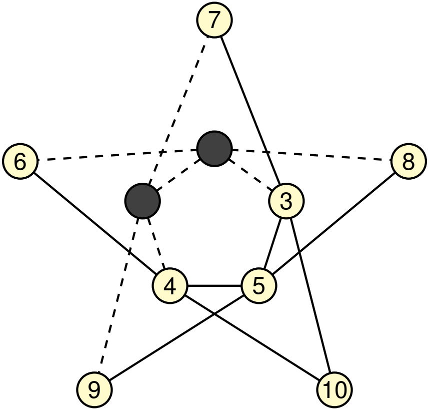
$$\min \sum_{v \in V} w_v \, x_v \tag{0.11}$$

$$\sum_{v \in V} \left(\deg_{\mathcal{G}}(v) - 1 \right) x_v \geq \underbrace{k + |E(\mathcal{G})| - |V|}_{constant\ term}, \qquad \mathcal{G} \in \mathcal{SAS}(G), \tag{0.12}$$

$$x_v \in \{0, 1\}, \qquad v \in V. \tag{0.13}$$

Example of the Subgraph Constraints (1/2)

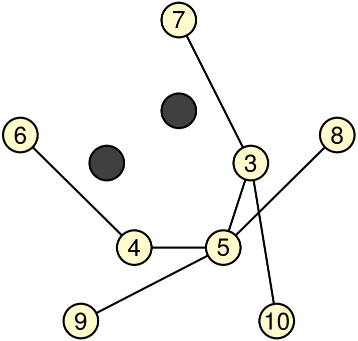
Infeasible solution for $k = 3$, the set black vertices $\{1, 2\}$ represent a leader strategy and the remaining vertices form one connected component:



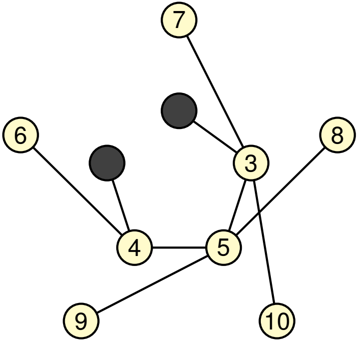
Example of the Subgraph Constraints (2/2)

- For a given (spanning) acyclic subgraph $\mathcal{G}$, we have the (spanning) acyclic subgraph constrain:

$$\sum_{v \in V} \left(\deg_{\mathcal{G}}(v) - 1 \right) x_v \geq k + |E(\mathcal{G})| - |V|$$



$-x_1 - x_2 + 2x_3 + x_4 + 3x_5 \geq 0$



$3x_3 + 2x_4 + 3x_5 \geq 2$

Separation of the Subgraph Constraints

Let x^* be the current solution. We define edge-weights as

$$w^*_{uv} = 1 - x^*_u - x^*_v, \quad uv \in E$$

and search for the maximum-weighted cycle-free subgraph in G. Let W^* denote the weight of the obtained subgraph; if $W^* > |V| - k - \sum_{v \in V} x^*_v$, we have detected a violated inequality.

The separation procedure can be performed in polynomial time:

- adaptation of Kruskal's algorithm for minimum-spanning trees (fractional points), or
- BFS (integer points) on the graph from which $x_v = 1$ vertices are removed. Extended to spanning subgraphs (dominating cuts).

Observation

Separation of the Subgraph Constraints can be performed in polynomial time.

A third ILP model (1/4)

- The key idea:
  - Use a representative vertex for each subset of vertices.
  - Ensure that the representative vertices are **pairwise disconnected**.
  - Connected components that are disconnected from any representative can be assigned to any subset.
- This is modeled using:
  - Two binary variables for each vertex:
    - Is the vertex a representative vertex?
    - Is the vertex in the k-vertex cut?
  - An exponential number of **Path Constraints**, ensuring no path exists between any two representatives.

Paths of the graph

Let P denotes a simple path in G, $V(P)$ are the vertices connected by P, and let Π_{uv} be the set of all simple paths between vertices u and v.

A third ILP model (2/4)

Using the binary variables:

$$z_v = \begin{cases} 1 & \text{if vertex } v \text{ is the representative of a subset,} \\ 0 & \text{otherwise,} \end{cases} \qquad v \in V,$$

$$x_v = \begin{cases} 1 & \text{if vertex } v \text{ is in the } k\text{-vertex cut,} \\ 0 & \text{otherwise,} \end{cases} \qquad v \in V,$$

the representative ILP model (called *REP*) for k-vertex cut problem is:

$$\min \sum_{v \in V} w_v \, x_v \tag{0.14}$$

$$\sum_{v \in V} z_v = k, \qquad v \in V, \tag{0.15}$$

$$z_u + z_v \leq 1, \qquad \{u, v\} \in E, \tag{0.16}$$

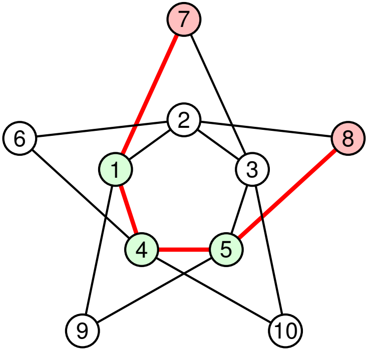
$$\sum_{w \in V(P) \setminus \{u,v\}} x_w \geq z_u + z_v - 1, \qquad u, v \in V, P \in \Pi_{uv}, \tag{0.17}$$

$$x_v, z_v \in \{0, 1\}, \qquad v \in V. \tag{0.18}$$

Example of the Path Constraints (1/2)

- For a given pair of vertices $u, v \in V$ and a path $P \in \Pi_{uv}$, we have the path inequality:

$$\sum_{w \in V(P) \setminus \{u,v\}} x_w \geq z_u + z_v - 1$$



$u = 7$ and $v = 8$

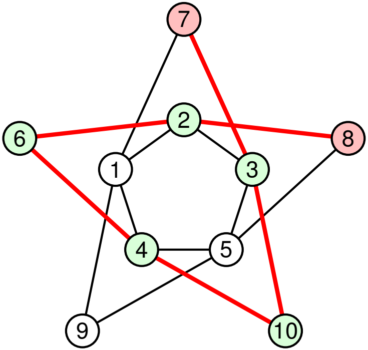
$$P \rightsquigarrow \underbrace{7 \to 1 \to 4 \to 5 \to 8}_{V(P)}$$

$$x_1 + x_4 + x_5 \geq z_7 + z_8 - 1$$

Example of the Path Constraints (2/2)

- For a given pair of vertices $u, v \in V$ and a path $P \in \Pi_{uv}$, we have the path inequality:

$$\sum_{w \in V(P) \setminus \{u,v\}} x_w \geq z_u + z_v - 1$$



$$u = 7 \quad \text{and} \quad v = 8$$

$$P \rightsquigarrow \underbrace{7 \rightarrow 3 \rightarrow 10 \rightarrow 4 \rightarrow 6 \rightarrow 2 \rightarrow 8}_{=V(P)}$$

$$x_3 + x_{10} + x_4 + x_6 + x_2 \geq z_7 + z_8 - 1$$

Separation of the Path Constraints

Given a solution $x^*, z^* \in [0,1]^V$, the separation problem asks for finding a pair of vertices u, v such that there is a path $P^* \in \Pi_{uv}$ with

$$z_u + z_v > \sum_{w \in V(P^*) \setminus \{u,v\}} x_w - 1.$$

We can search for such a path in polynomial time by solving a shortest path problem (for each pair of not adjacent vertices) on graph G, where we define the length of each edge $(u, v) \in E$ as:

$$l_{uv} = \frac{x_u^* + x_v^*}{2}$$

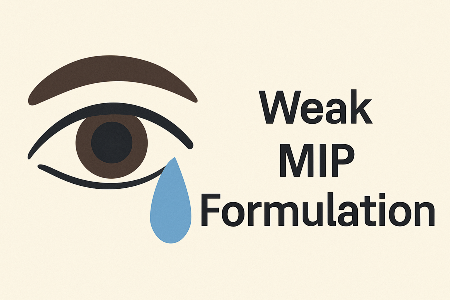
Observation

Separation of the Path Constraints can be performed in polynomial time.

A third ILP model (3/4)

- The *REP* model has one principal drawbacks:
  - It suffers from a weak LP relaxation: if $k \leq n/2$, an optimal LP solution with objective value **zero** can be obtained by setting:

$$x_v = z_v = 0, \quad v \in V.$$



A third ILP model (4/4)

- Valid inequalities in polynomial number:

$$x_u + z_u \leq 1, \qquad u \in V,$$

$$z_u + \sum_{v \in N(u)} z_v \leq 1 + (\deg(u) - 1)x_u \qquad u \in V.$$

- Strengthened Path Constraints:

$$\sum_{w \in V(P) \setminus \{u,v\}} x_w \geq z_u + z_v + \sum_{w \in V(P) \setminus \{u,v\}} z_w - 1, \quad u, v \in V, P \in \Pi_{uv}.$$

each time a path in Π_{uv} includes a representative vertex, an additional vertex of the path must be in the vertex-cut.

- Clique-Path Constraints... Each z on the RHS is replaced by a clique...

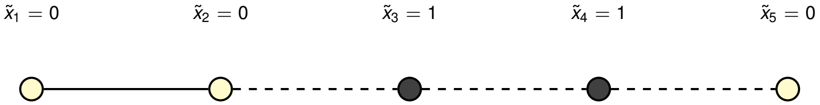
Additional results

Proposition

If $k \leq n/2$, the bound for the k-vertex cut problem provided by the optimal solution value of the LP relaxation of NAT model strictly dominates the corresponding bound provided by the REP model.

Proposition

Path Constraints derived from spanning trees only are not sufficient to ensure a valid formulation for the k-vertex cut problem.



Let $k = 3$. There is a single spanning tree in G, and the associated cut, which is $x_2 + x_3 + x_4 \geq 2$, does not cut off the infeasible point (one two connected components).

Computational results

Experimental Settings and Benchmark Instances (1/3)

- We want to assess the computational performance of the branch-and-cut algorithms to solve:
  1. the *NAT* model
  2. the *REP* model
  3. the *HYB* model (both models together)

  by comparison with:
  1. the *COMP* model
  2. the state-of-the-art branch-and-price algorithm proposed in the literature.

- The source code of our branch-and-cut algorithms can be downloaded at:

`https://github.com/paoloparonuzzi/k-Vertex-Cut-Problem/`

- A time limit of one hour is set for each tested instance and `CPLEX 12.7.1`

Experimental Settings and Benchmark Instances (2/2)

Branch-and-cut Algorithms Settings:

- *REP*: separates path constraints via shortest paths in graphs with positive edge weights.
- REP_{lp}: uses a heuristic to separate path constraints by exploring long paths in graphs with both positive and negative weights.
- *NAT*: includes base connectivity constraints, lifted when enforcing spanning properties.
- NAT_s: connectivity constraints are always made spanning for integer solutions and then lifted.

Cut Separation Settings:

- Absolute violation tolerance set to 0.5.
- Cuts are separated:
  - at all integer solutions;
  - every 100 nodes for REP/REP_{lp},
  - every node for NAT/NAT_s.

Experimental Settings and Benchmark Instances (3/3)

- We evaluate our methods on two benchmark sets of instances, with both unit and random vertex weights.
- **First Set:**
  - instances from DIMACS Vertex Coloring problems (up to 200 vertices) and Graph Partitioning problems (up to 300 vertices), all with $\alpha(G) \geq 5$.
- **Second Set:**
  - instances from the literature, based on the intersection graphs of coefficient matrices from linear systems.
- For each value of $k \in \{5, 10, 15, 20\}$, we exclude infeasible and trivially solved instances from the analysis.

Performance comparison between the MIP models

- Performance comparison for different configurations of the Representative, Natural and Hybrid Formulations on the first set of instances (Vertex Coloring and DIMACS).

	REP	REP_{lp}	*NAT*	NAT_s	*HYB*
Total Opt. (out of 166)	89	96	126	128	132
Total Avg Time	146.75	194.04	121.10	66.20	2.55
Total Avg Nodes	50656	23169	45	43	15
Total Avg LP Gap	73.19	51.69	18.11	18.07	18.11
Total Avg LP Time	0.04	34.98	0.25	0.24	0.41

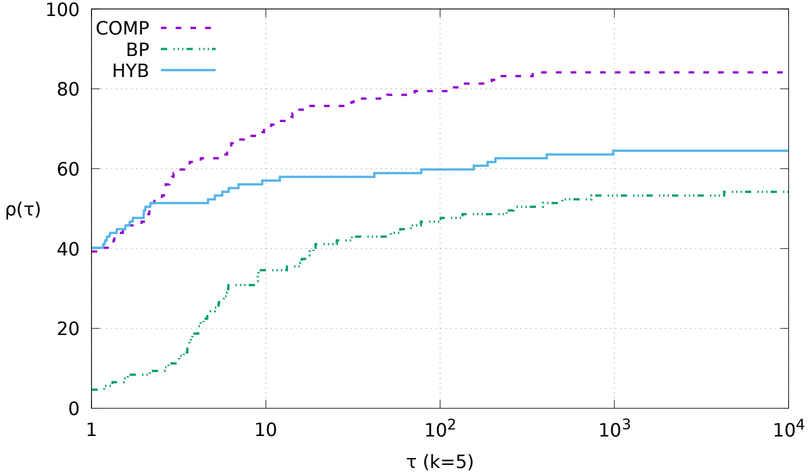
Vertex Coloring, DIMACS and Intersection graphs

k		*COMP*	*BP*	*HYB*
	Opt. (out of 107)	92	60	71
5	Avg Time	31.84	59.93	84.78
	Avg Nodes	10768	30	106
	Opt. (out of 80)	37	43	51
10	Avg Time	105.64	52.19	1.39
	Avg Nodes	67123	7	26
	Opt. (out of 65)	29	36	46
15	Avg Time	219.33	23.38	2.81
	Avg Nodes	41750	19	25
	Opt. (out of 52)	19	29	38
20	Avg Time	196.06	169.52	0.39
	Avg Nodes	58673	16	6
	Total Opt. (out of 304)	177	168	206
	Total Avg Time	98.66	61.78	43.66
	Total Avg Nodes	33040	22	64

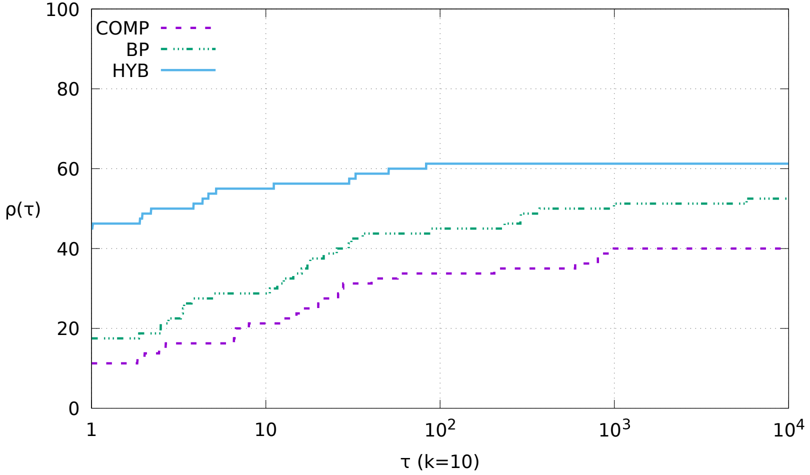
Vertex Coloring, DIMACS and Intersection graphs – weighted

k		*COMP*	*BP*	*HYB*
	Opt. (out of 107)	92	60	71
5	Avg Time	35.99	67.55	210.67
	Avg Nodes	11350	77	217
	Opt. (out of 80)	37	43	51
10	Avg Time	69.61	174.96	2.30
	Avg Nodes	22872	21	26
	Opt. (out of 65)	29	37	47
15	Avg Time	343.26	36.61	21.76
	Avg Nodes	109726	180	86
	Opt. (out of 52)	19	30	39
20	Avg Time	559.17	300.40	1.15
	Avg Nodes	180529	31	15
	Total Opt. (out of 304)	177	170	208
	Total Avg Time	151.21	112.23	106.13
	Total Avg Nodes	48594	77	127

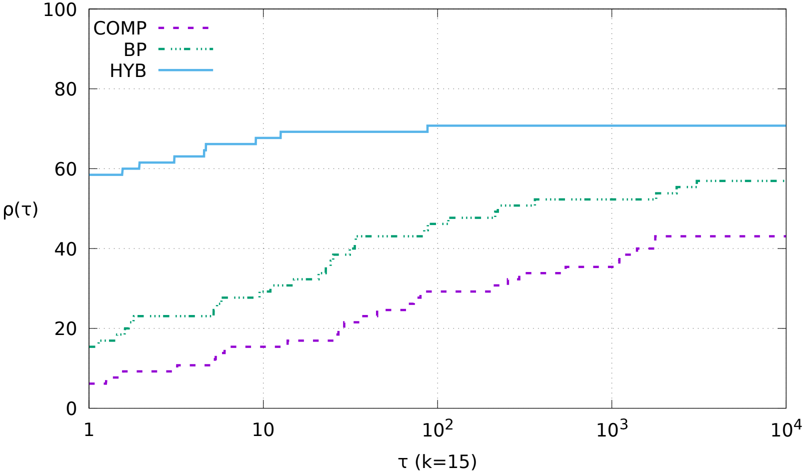
Performance profiles (1/4)



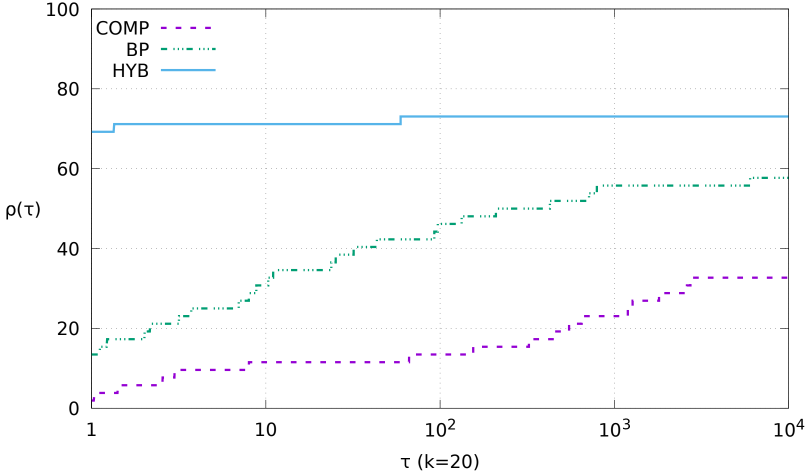
Performance profiles (2/4)



Performance profiles (3/4)



Performance profiles (4/4)



Conclusions

Conclusions (1/2) – Main Contributions

- Many graph disconnection problems naturally exhibit a bilevel structure.
- We reveal this structure in two key problems:
  - ***k*-Vertex Cut**
  - **Capacitated Vertex Separator**
- Both problems are modeled as Bilevel Optimization Problems:
  - Leader deletes nodes;
  - Follower optimizes connectivity/reactive behavior.
- We introduce new **bilevel integer programming formulations**, capturing this interaction.
- Our models are strengthened through:
  - Families of **valid inequalities**
  - **Polynomial-time separation procedures**

Conclusions (2/2) – Outlook and Perspectives

- Our computational results show that the bilevel approach:
  - Improves **solution quality** on benchmark instances
  - Achieves **faster convergence** compared to the state-of-the-art
- The bilevel modeling perspective offers a unified framework for:
  - Graph partitioning and separator problems
  - Network interdiction and security
  - Clustering, community detection, and more
- Future work includes:
  - Extending bilevel models to **edge deletion** and **dynamic settings**
  - Integrating these formulations into **other exact methods** (e.g., branch-and-cut-and-price algorithms)
  - Designing **approximation and heuristic** algorithms leveraging the bilevel structure

References

[1] Lalou, M., Tahraoui, M.A., Kheddouci, H.: The critical node detection problem in networks: A survey.
Computer Science Review **28**, 92 – 117 (2018)

[2] Borndörfer, R., Ferreira, C., Martin, A.: Decomposing matrices into blocks.
SIAM Journal on Optimization **9**(1), 236–269 (1998)

[3] Magnouche, Y., Mahjoub, A.R., Martin, S.: The multi-terminal vertex separator problem: Polyhedral analysis and branch-and-cut.
Discrete Applied Mathematics **290**, 86-111 (2021)