A Branch-and-Cut Algorithm for Constrained Graph Clustering

Behrouz Babaki[1], Dries Van Daele[1], Bram Weytjens[2], Tias Guns[1,3]

[1]Department of Computer Science, KU Leuven, Belgium
[2]Department of of Microbial and Molecular Systems, KU Leuven, Belgium
[3]Department of Business Technology and Operations, VUB, Belgium

Structure of the presentation

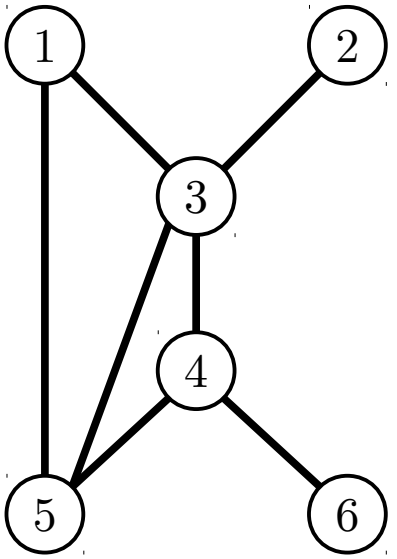
- Introduction
- A novel clustering problem
- An *incomplete* MIP formulation
- Enforcing connectedness: enumerative method
- Enforcing connectedness: branch-and-cut
- Computing the Pareto front
- Experiments and results
- Conclusion

Introduction

- Clustering is a common task in data science
- New problems in new domains: existing clustering algorithms are not sufficient
- An increasingly popular approach: model the clustering problem as a CP or MIP model and solve it by a solver
- Instead of developing an algorithm for each new problem, write an optimization model

A novel clustering problem

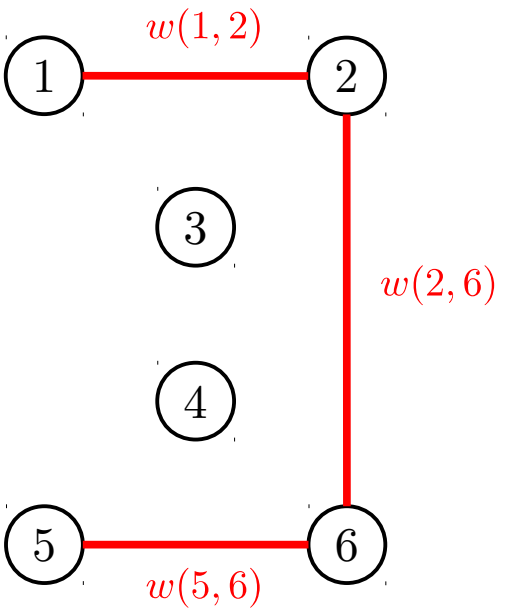
- Genes and gene products have interactions, represented by an *interaction graph*
- Mission: Find genes and interactions that are relevant for breast cancer
- Genes that interact with each other (a *pathway*): a connected subgraph of interaction graph
- We also care about the size of clusters (preferably not too small)



interaction graph

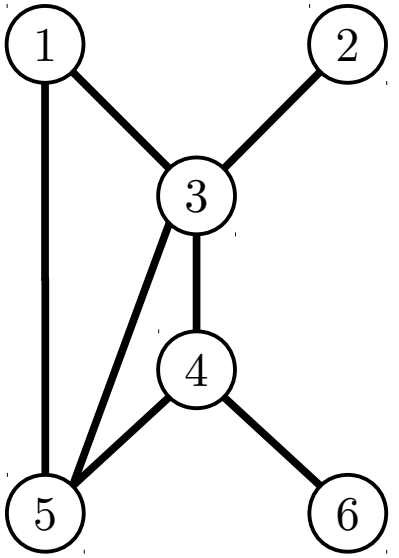
A novel clustering problem

- Mission: Find genes and interactions that are relevant for breast cancer
- Check observed *mutations* to detect relevance
- We have already included genes that are mutated
- How to know which mutations work together? *Cancer tends to avoid unnecessary effort!*
- Mutations that are observed together do not belong to the same pathway
- Penalties for co-membership of such genes represented as edge weights

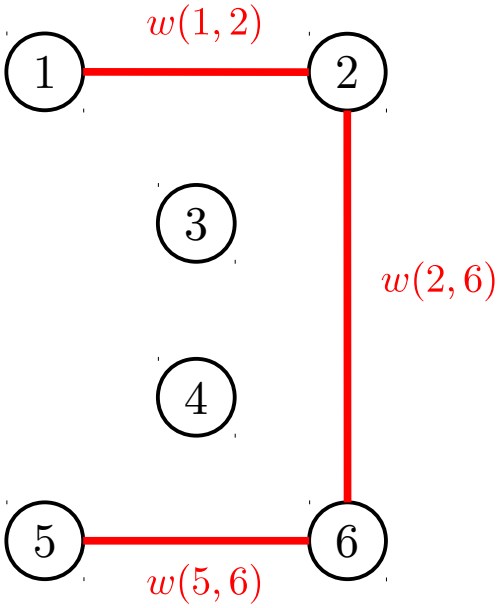


co-membership graph

A novel clustering problem

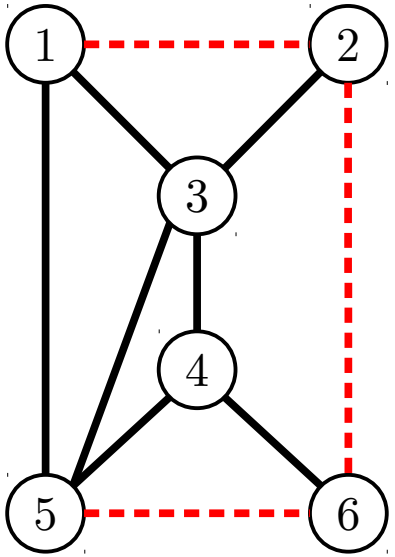


- Two graphs over the same set of nodes. Cluster the nodes
- Each cluster should be connected w.r.t the *interaction graph*
- If two node in a cluster are connected in the *co-membership graph*, a penalty is induced
- We prefer not to have too small clusters (We include the size of the smallest cluster in the objective function)
- **Optimization problem:** Minimize the penalties and maximize the size of the smallest cluster, subject to the *clustering* and *connectedness constraint*

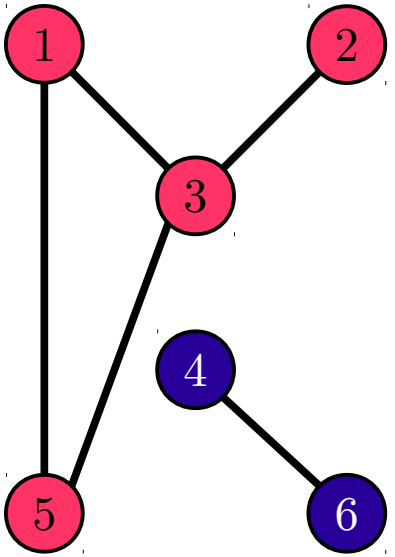


MIP formulation

x_{ic} whether instance i is in cluster c

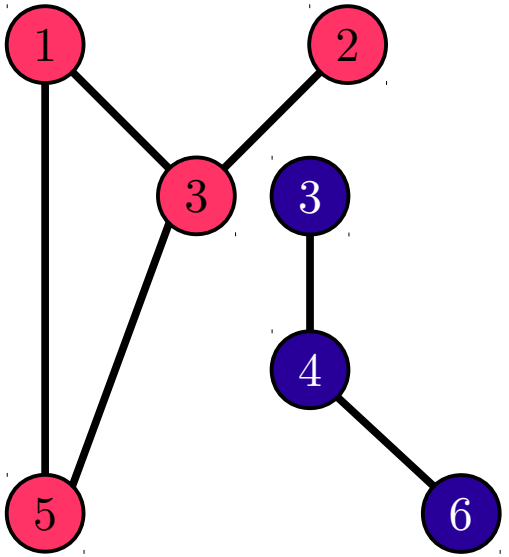


$$x_{31} = 1$$
$$x_{32} = 0$$



$$\sum_c x_{ic} = 1$$

$$x_{31} = 1$$
$$x_{32} = 1$$



$$\sum_c x_{ic} \geq 1$$

MIP formulation

Minimize the penalties

y_{ijc} whether instances i and j are in cluster c

$$\text{min.} \sum_{c} \sum_{(i,j) \in E} \underbrace{w(i,j)}.y_{ijc}$$

penalty for i and j being in the same cluster

MIP formulation

Minimize the penalties

y_{ijc} whether instances i and j are in cluster c

$$\min. \sum_{c} \sum_{(i,j) \in E} \underbrace{w(i,j)}.y_{ijc}$$

penalty for i and j being in the same cluster

$$(y_{ijc} = 1) \iff x_{ic} \wedge x_{jc}$$

$$y_{ijc} \geq x_{ic} + x_{jc} - 1$$

MIP formulation

Maximize the size of smallest cluster

S size of the smallest cluster

$$\max. S - \gamma \sum_{c} \sum_{(i,j) \in E} w(i,j).y_{ijc}$$

MIP formulation

Maximize the size of smallest cluster

S size of the smallest cluster

$$\max. S - \gamma \sum_{c} \sum_{(i,j) \in E} w(i,j).y_{ijc}$$

$$S \leq \sum_{i} x_{ic} \quad \forall c$$

MIP formulation

$$\max. S - \gamma \sum_{c} \sum_{(i,j) \in E} w(i,j).y_{ijc}$$

$$\sum_{c} x_{ic} = 1 \qquad \forall i$$

$$y_{ijc} \geq x_{ic} + x_{jc} - 1 \qquad \forall c, \forall (i,j) \in E$$

$$S \leq \sum_{i} x_{ic} \qquad \forall c$$

x_{ic}, y_{ijc} binary

S integer

MIP formulation

$$\max. S - \gamma \sum_{c} \sum_{(i,j) \in E} w(i,j).y_{ijc}$$

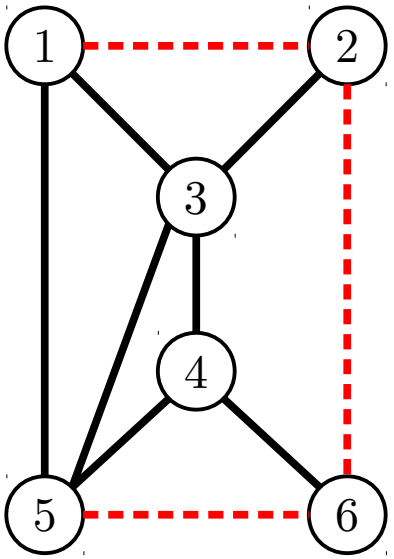
$$\sum_{c} x_{ic} = 1 \qquad \forall i$$

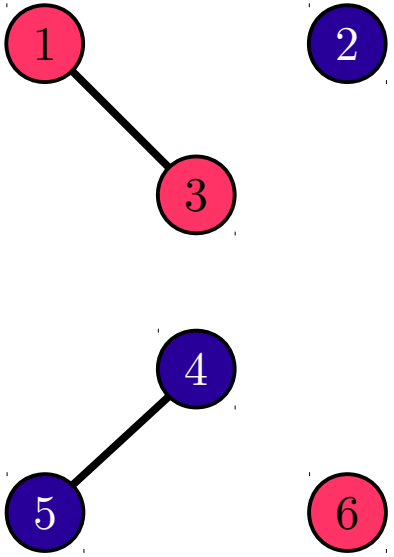
$$y_{ijc} \geq x_{ic} + x_{jc} - 1 \qquad \forall c, \forall (i,j) \in E$$

$$S \leq \sum_{i} x_{ic} \qquad \forall c$$

x_{ic}, y_{ijc} binary

S integer





MIP formulation

$$\max. S - \gamma \sum_{c} \sum_{(i,j) \in E} w(i,j).y_{ijc}$$

$$\sum_{c} x_{ic} = 1 \qquad \forall i$$

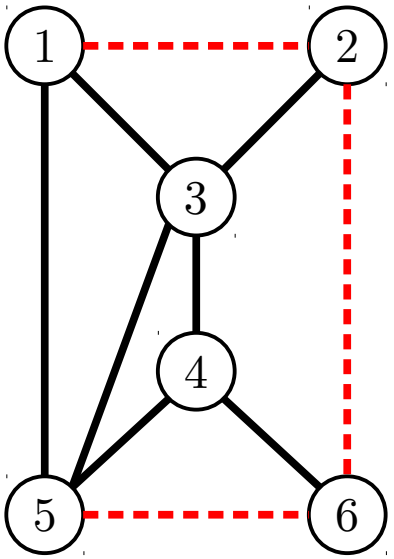
$$y_{ijc} \geq x_{ic} + x_{jc} - 1 \qquad \forall c, \forall (i,j) \in E$$

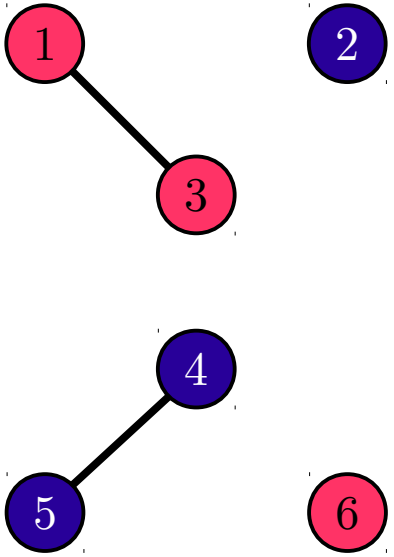
$$S \leq \sum_{i} x_{ic} \qquad \forall c$$

$$\mathsf{connected}(x_{1c}, \ldots, x_{nc}) \qquad \forall c$$

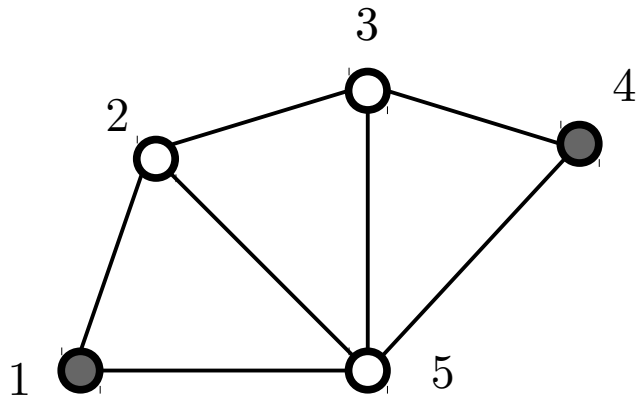
x_{ic}, y_{ijc} binary

S integer





Enforcing connectedness



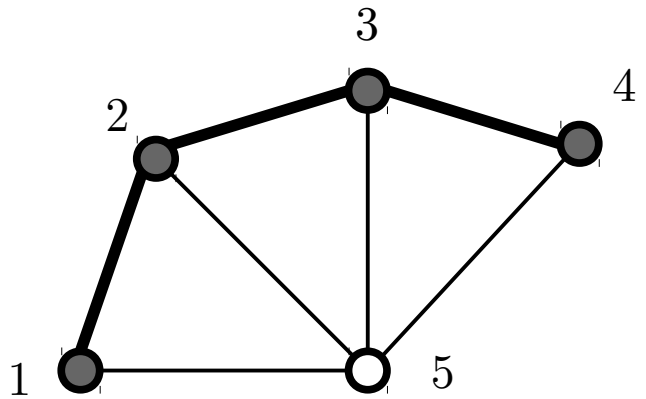
$$x_{1c} = 1$$
$$x_{2c} = 0$$
$$x_{3c} = 0$$
$$x_{4c} = 1$$
$$x_{5c} = 0$$

$$(x_1 = 1 \wedge x_4 = 1) \Rightarrow (x_2 = 1 \wedge x_3 = 1) \vee (x_5 = 1)$$

Two methods:

1. Enumerating all simple paths (`AllPaths`)
2. A branch-and-cut algorithm (`BnC`)

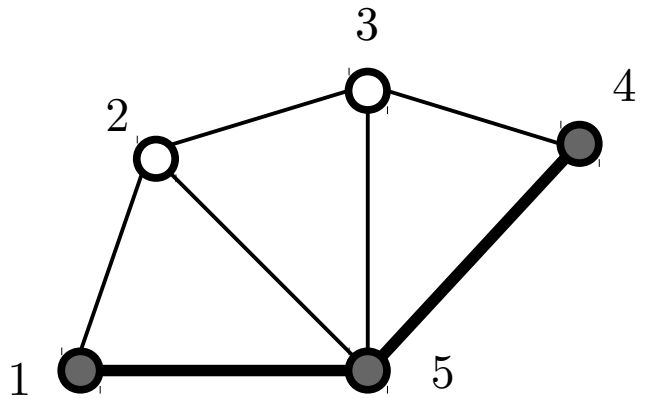
Enforcing connectedness (AllPaths)



z_1 whether first path is active

$$(z_1 = 1) \Leftrightarrow (x_{2c} = 1 \wedge x_{3c} = 1)$$

Enforcing connectedness (AllPaths)



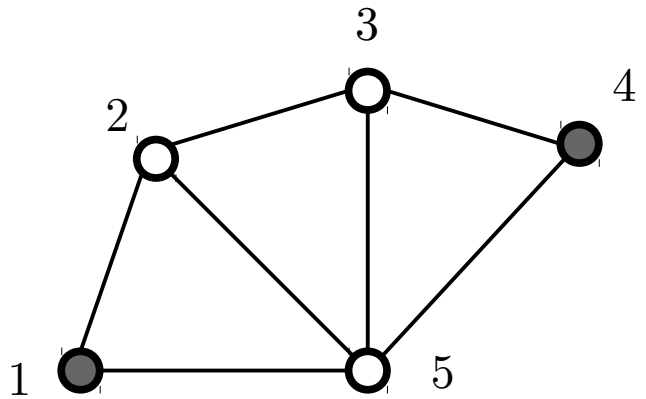
z_1 whether first path is active

z_2 whether second path is active

$$(z_1 = 1) \Leftrightarrow (x_{2c} = 1 \wedge x_{3c} = 1)$$

$$(z_2 = 1) \Leftrightarrow (x_{5c} = 1)$$

Enforcing connectedness (AllPaths)



z_1 whether first path is active

z_2 whether second path is active

z_3 ...

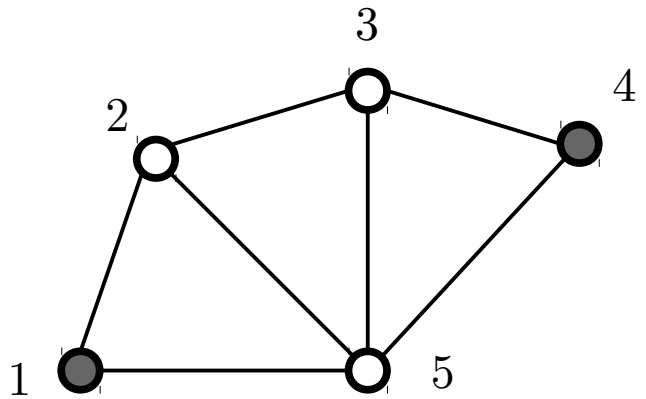
$(z_1 = 1) \Leftrightarrow (x_{2c} = 1 \wedge x_{3c} = 1)$

$(z_2 = 1) \Leftrightarrow (x_{5c} = 1)$

$(z_3 = 1) \Leftrightarrow \ldots$

$(x_{1c} = 1 \wedge x_{4c} = 1) \Rightarrow (z_1 = 1 \vee z_2 = 1 \vee \ldots)$

Enforcing connectedness (AllPaths)



z_1 whether first path is active

z_2 whether second path is active

z_3 ...

$$(z_1 = 1) \Leftrightarrow (x_{2c} = 1 \wedge x_{3c} = 1)$$

$$(z_2 = 1) \Leftrightarrow (x_{5c} = 1)$$

$$(z_3 = 1) \Leftrightarrow \ldots$$

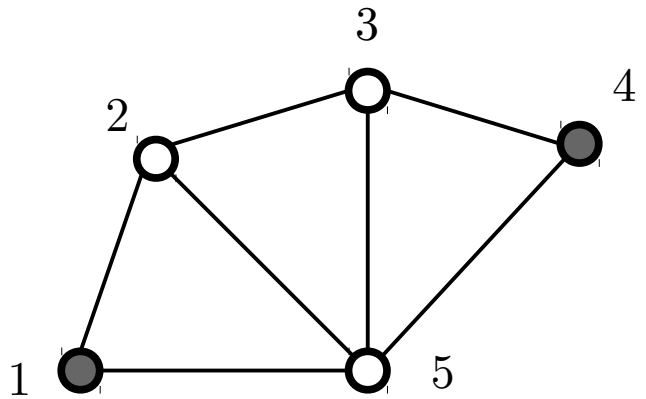
$$(x_{1c} = 1 \wedge x_{4c} = 1) \Rightarrow (z_1 = 1 \vee z_2 = 1 \vee \ldots)$$

conversion to linear constraints is simple

For all non-adjacent pairs of nodes and all clusters:

- Enumerate all simple paths
- Define an auxiliary variable for each path
- Ensure that the pair is present iff one of the paths is active

Enforcing connectedness (AllPaths)



P_{uv} simple paths between u and v

I_r intermediate nodes on path r

y_{rc} whether path r is active in cluster c

$$0 \leq \sum_{i \in I_r} x_{ic} - |I_r| y_{rc} \leq |I_r| - 1 \qquad \forall (u,v), \forall r, \forall c$$

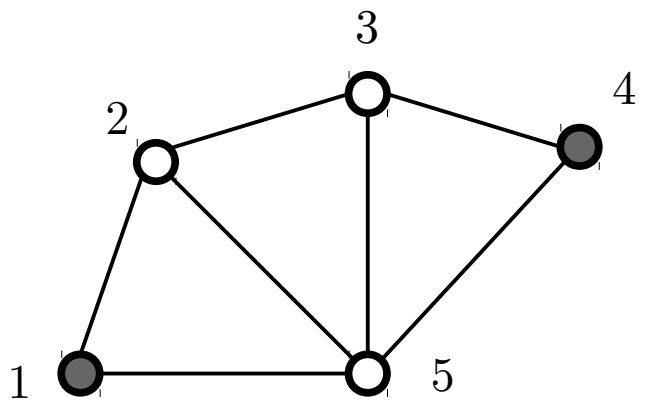
$$x_{uc} + x_{vc} - 1 \leq \sum_{r=1}^{|P_{uv}|} y_r c \qquad \forall (u,v), \forall c$$

Digression - The cutting plane method

Finding the most-violated constraint

- Given a solution x^* and constraints $a_i x \leq b_i$ find $\operatorname{argmin}_i a_i x^* - b_i$
- Exponential number of constraints: expensive to check
- Sometimes this problem can be solved efficiently by an external algorithm
- This problem is called the *separation* problem

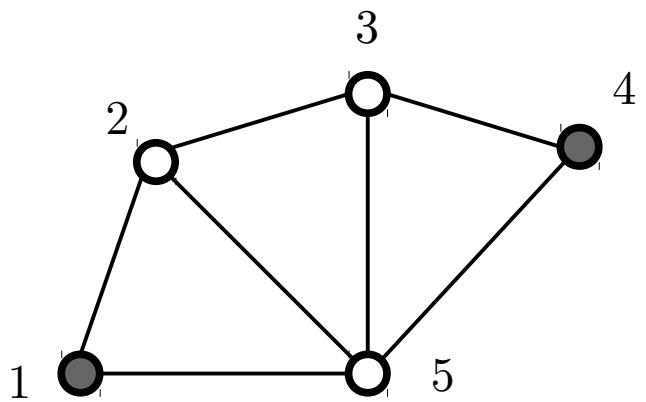
Digression – connectedness and node-cutsets



- Defined for pairs of nodes: a node-cutset of (1, 4)
- Every path between (1, 4) passes through this set

{2, 5}
{3, 5}
{2, 3, 5}

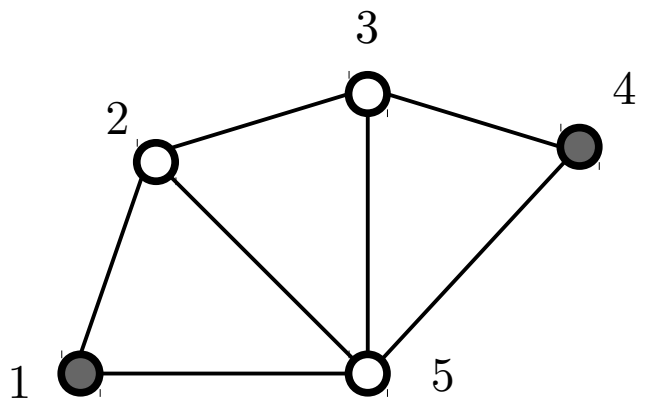
Digression – connectedness and node-cutsets



- Defined for pairs of nodes: a node-cutset of (1, 4)
- Every path between (1, 4) passes through this set
- **Minimal** node-cutset of (1, 4): not a node-cutset of (1, 4) after removing any member

{2, 5}
{3, 5}
~~{2, 3, 5}~~

Digression – connectedness and node-cutsets



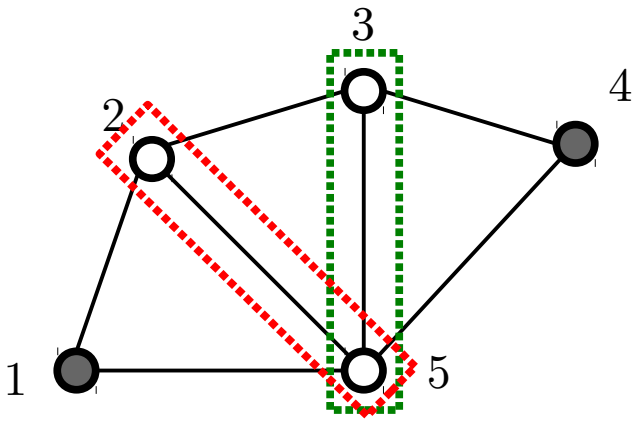
- Defined for pairs of nodes: a node-cutset of (1, 4)
- Every path between (1, 4) passes through this set
- **Minimal** node-cutset of (1, 4): not a node-cutset of (1, 4) after removing any member

{2, 5}
{3, 5}
~~{2, 3, 5}~~

Question: Is the subgraph induced by $U \subseteq V$ connected?

Theorem: In this subgraph, u and v are connected iff U contains at least one node from each minimal node-cutset of (u, v)

Digression – connectedness and node-cutsets



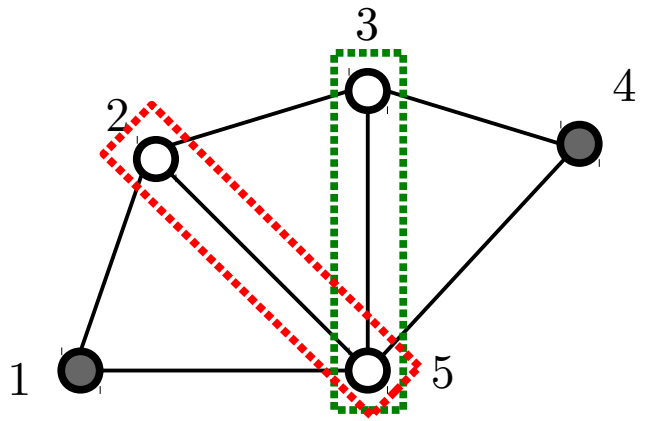
- Defined for pairs of nodes: a node-cutset of (1, 4)
- Every path between (1, 4) passes through this set
- **Minimal** node-cutset of (1, 4): not a node-cutset of (1, 4) after removing any member

{2, 5}
{3, 5}
~~{2, 3, 5}~~

Question: Is the subgraph induced by $U \subseteq V$ connected?

Theorem: In this subgraph, u and v are connected iff U contains at least one node from each minimal node-cutset of (u, v)

Enforcing connectedness (BnC)

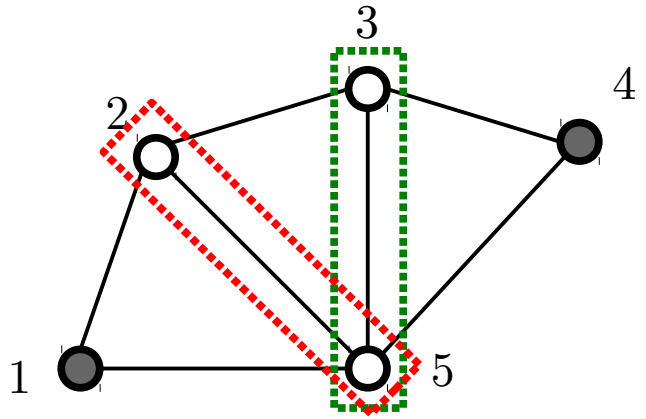


Theorem: In this subgraph, u and v are connected iff U contains at least one node from each minimal node-cutset of *(u, v)*

for every cluster c
for every non-adjacent pair *(u, v)*
for every minimal node-cutset S for *(u,v)*

$$(x_{uc} \wedge x_{vc}) \Rightarrow \vee_{w \in S}(x_{wc} = 1)$$

Enforcing connectedness (BnC)



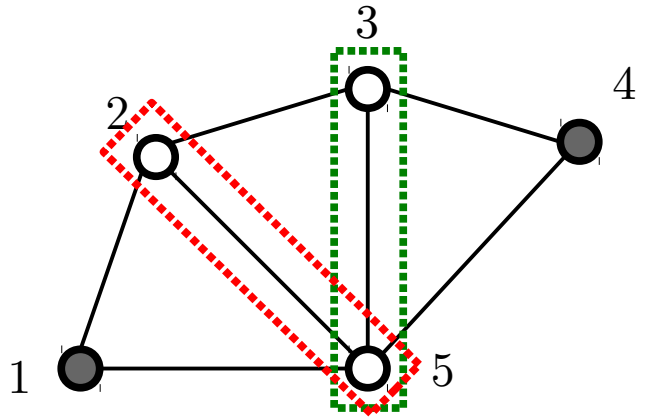
Theorem: In this subgraph, u and v are connected iff U contains at least one node from each minimal node-cutset of (u, v)

for every cluster c
for every non-adjacent pair (u, v)
for every minimal node-cutset S for (u,v)

$$(x_{uc} \wedge x_{vc}) \Rightarrow \vee_{w \in S}(x_{wc} = 1)$$

$$\sum_{w \in S} x_{wc} \geq x_{uc} + x_{vc} - 1$$

Enforcing connectedness (BnC)



Theorem: In this subgraph, u and v are connected iff U contains at least one node from each minimal node-cutset of (u, v)

for every cluster c
for every non-adjacent pair (u, v)
for every minimal node-cutset S for (u,v)

$$(x_{uc} \wedge x_{vc}) \Rightarrow \vee_{w \in S}(x_{wc} = 1)$$

$$\sum_{w \in S} x_{wc} \geq x_{uc} + x_{vc} - 1$$

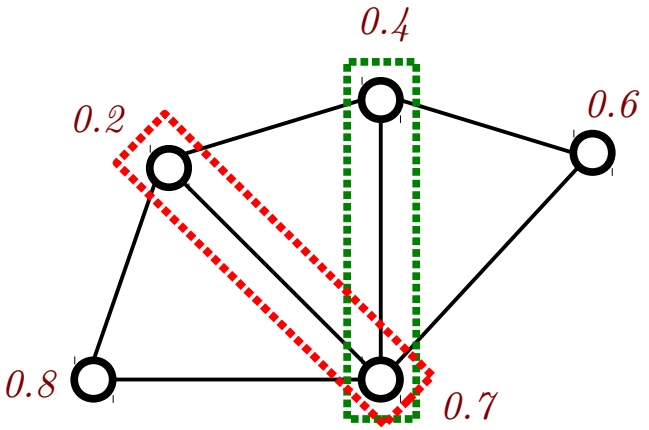
$$\operatorname*{argmin}_{S} \sum_{w \in S} x^*_{wc}$$

most-violated constraint for each (u, v)

Separation problem

- integer x^*: find an absent node-cutset
- continuous x^*: find the node-cutset with smallest sum of values

Enforcing connectedness (BnC)

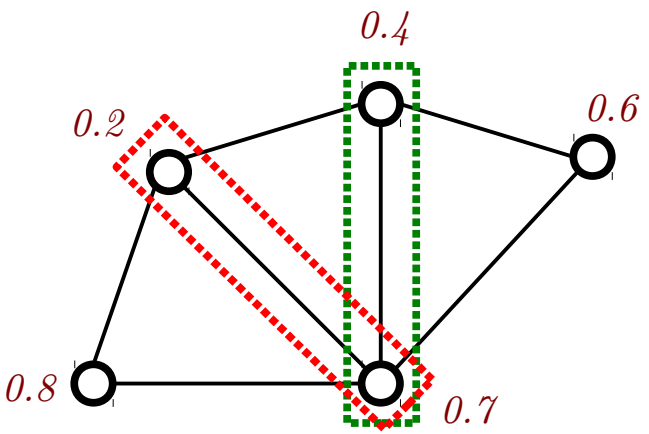


$$\underset{S}{\operatorname{argmin}} \sum_{w \in S} x^*_{wc}$$

most-violated constraint for each $(u,\ v)$

Separation problem: find the node-cutset with smallest sum of values

Enforcing connectedness (BnC)



$$\underset{S}{\operatorname{argmin}} \sum_{w \in S} x^*_{wc}$$

most-violated constraint for each (u, v)

Separation problem: find the node-cutset with smallest sum of values

The min-cut algorithm

The cut-generation procedure

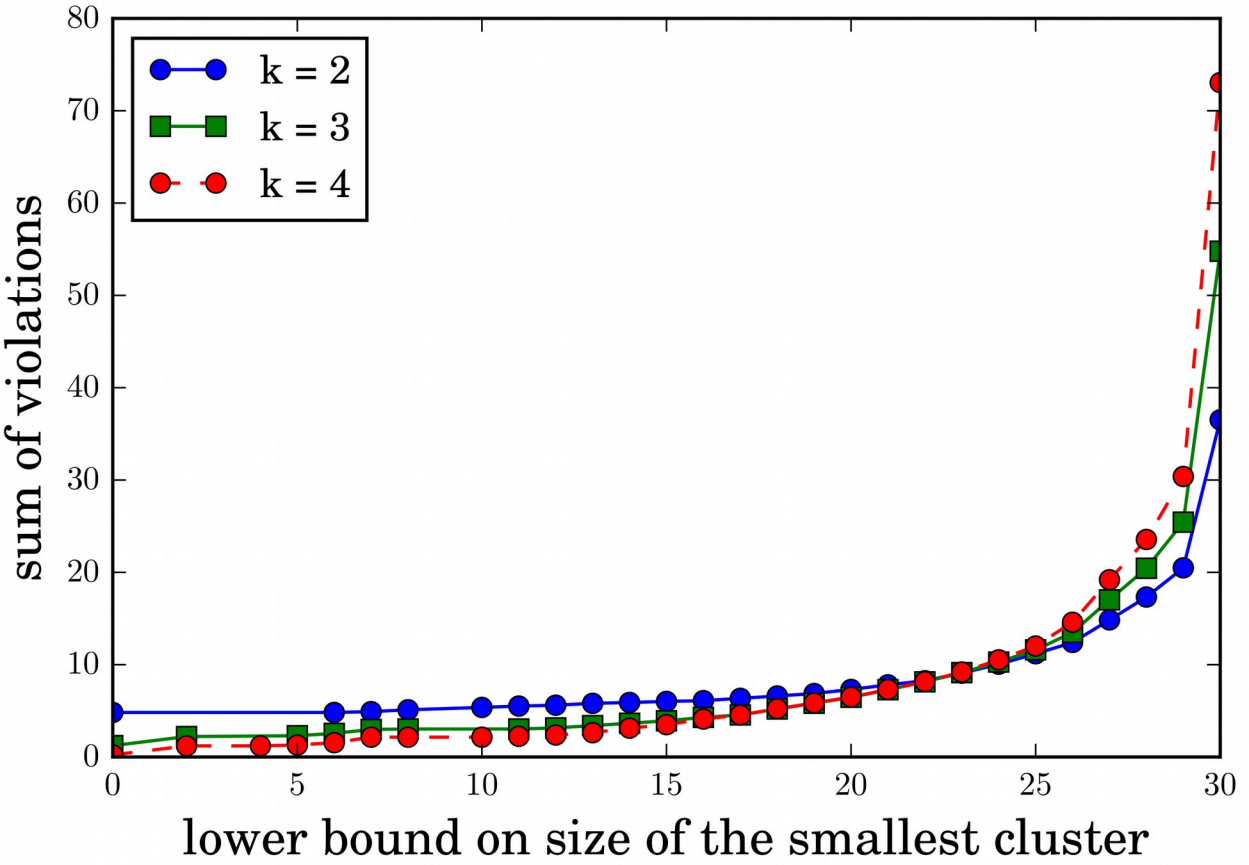
$\mathcal{C} \leftarrow \emptyset$ ▷ Set of constraints to add
for $c \in \{1, \ldots, k\}$ **do**
  for u, v such that $(u, v) \notin E_i$ and $x^*_{uc} + x^*_{vc} > 1$ **do**
    $S^* \leftarrow$ min-cut$(u, v, \mathbf{x}^*_c, G_i)$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{\sum_{w \in S^*} x_{wc} \geq x_{uc} + x_{vc} - 1\}$
  end for
end for
Add constraints $\mathcal{C}$ to the model

The Pareto front

Computing the set of Pareto optimal solutions

1: $\mathcal{P} \leftarrow \emptyset$ ▷ the set of Pareto optimal solutions
2: $m \leftarrow 0$ ▷ Minimum size of the previous solution
3: **repeat**
4: $solution \leftarrow$ MINIMIZEPENALTIES(V, G_i, G_c, k, m)
5: $\mathcal{P} \leftarrow \mathcal{P} \cup solution$
6: $m \leftarrow$ size of the smallest cluster in $solution$
7: **until** no $solution$ was found
8: **return** $\mathcal{P}$

The Pareto front



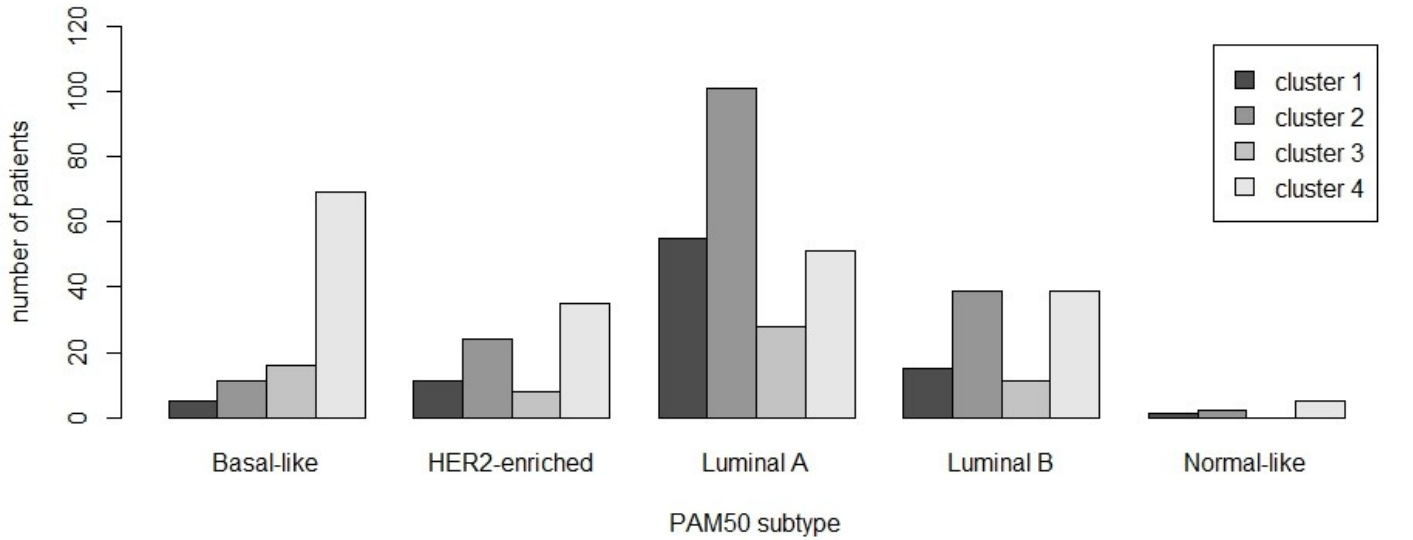
Experiments

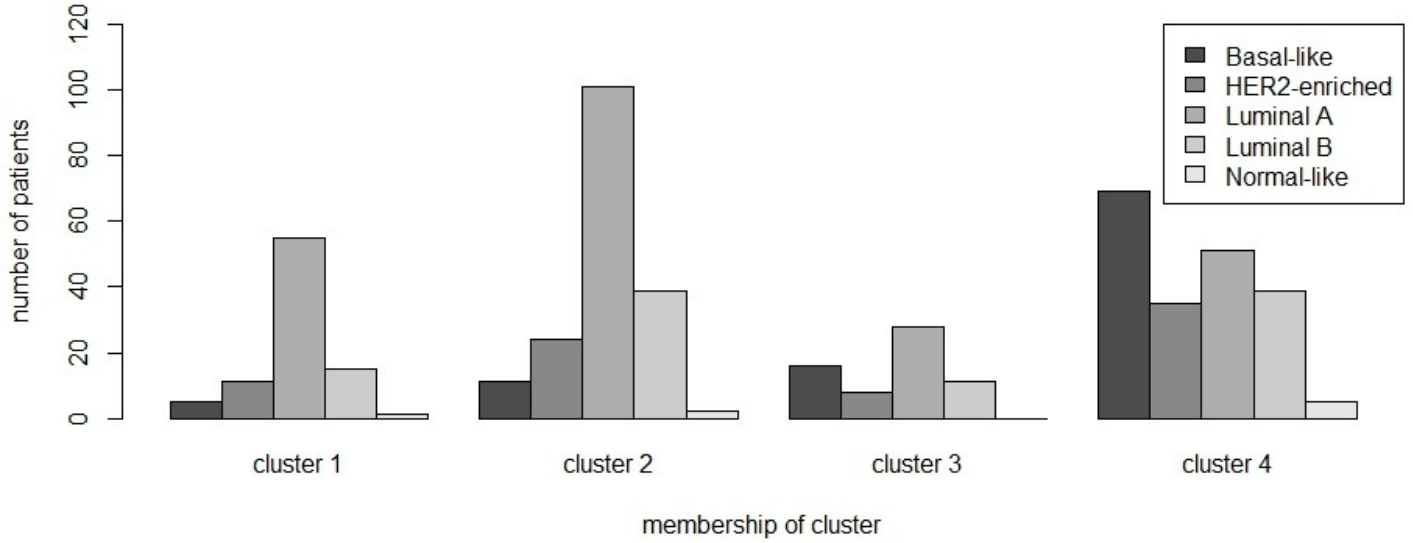
instance	*#paths*	*k*	*overlapping*		*non-overlapping*	
			AllPaths	*BnC*	*AllPaths*	*BnC*
200_1000	93	2	**0.830**	12.320	**0.319**	13.398
		3	**3.211**	16.255	**4.474**	15.007
		4	**14.846**	28.714	**15.578**	52.262
250_750	45	2	**0.166**	4.058	**0.025**	3.873
		3	**0.321**	6.433	**0.387**	3.965
		4	**0.737**	10.564	**0.698**	7.374
300_750	59	2	**0.333**	12.232	**0.209**	10.546
		3	**0.761**	20.365	**0.812**	16.924
		4	**1.728**	28.267	**3.598**	50.836
350_500	12	2	**0.004**	0.093	**0.003**	0.151
		3	**0.006**	0.129	**0.021**	0.257
		4	**0.007**	0.181	**0.039**	0.211
350_750	73	2	**0.503**	44.842	**0.591**	20.130
		3	**1.971**	113.311	**2.016**	58.417
		4	**7.892**	323.188	**9.841**	309.998
450_750	105	2	**1.560**	115.541	**1.334**	89.672
		3	**10.686**	361.855	**12.655**	320.706
		4	**90.516**	430.954	**75.395**	–

Experiments

instance	*#paths*	k	*overlapping*		*non-overlapping*	
			AllPaths	*BnC*	*AllPaths*	*BnC*
200_1250	1508	2	225.735	**43.740**	113.210	**38.849**
		3	582.541	**52.950**	298.390	**72.643**
		4	–	**188.287**	595.930	**205.454**
250_1000	1040	2	487.222	**78.243**	370.898	**69.481**
		3	587.700	**121.677**	–	**217.079**
		4	–	**206.952**	–	–

Biological validation





Conclusions

- By modeling the problem as a MIP, we can *quickly* develop a *flexible* solving method
- Different models suit different types of data (sparse/tree-like graphs vs. dense graphs)
- The real-life instance can only be solved using the branch-and-cut method
- Future work: characterizing instances w.r.t their behavior with the two methods
- Future work: extensive biological evaluation of the method within a data analysis pipeline