A Graph Modification Approach for Finding Core–Periphery Structures in Protein Interaction Networks

Sharon Bruckner\textsuperscript{1}  Falk Hüffner\textsuperscript{2}  Christian Komusiewicz\textsuperscript{2}

\textsuperscript{1}Institut für Mathematik, Freie Universität Berlin
\textsuperscript{2}Institut für Softwaretechnik und Theoretische Informatik, TU Berlin

30 September 2014
Protein Complex Identification

**Task:** Given a protein interaction network, identify its protein complexes and functional modules

**Common assumptions:**
- Complexes and functional modules are dense subnetworks
- Functional modules have no or only small overlap

⇝ Formulation as graph clustering problem

**Cluster Editing**

**Input:** An undirected graph \( G = (V, E) \).

**Task:** Find a minimum-size set of edge deletions and edge insertions that converts the graph into a **cluster graph**, that is, a graph where each connected component is a clique.
Denseness of Complexes and Functional Units

**Problem:** Functional units are not necessarily dense

Nucleosome remodeling deacetylase (NuRD) complex of *M. musculus* and its interactions with transcription factors

⇝

Core–periphery model of protein complexes

[Gavin et al., Nature ’06]
Core–Periphery Model

**Aim:** Uncover global core–periphery structure of given PPI network with dense cores and sparse peripheries.

**Formalization:**

Split graph = can be partitioned into clique and independent set

Split cluster graph = every connected component is a split graph

⇝

**Split Cluster Editing**

**Input:** An undirected graph $G = (V, E)$.

**Task:** Find a minimum-size set of edge deletions and edge insertions that converts the graph into a split cluster graph.
Shared Peripheries

So far:
- Complexes and functional modules are dense subnetworks have core–periphery structure
- Functional modules have no or only small overlap

Now: allow overlap but only in peripheries

Monopolar graph = can be partitioned into cluster graph and independent set

Monopolar Editing
Input: An undirected graph $G = (V, E)$.
Task: Find a minimum-size set of edge deletions and edge insertions that converts the graph into a monopolar graph.
Problem Complexity—Split Cluster Editing

**Theorem:** (Foldes & Hammer ’71) A graph is a split graph iff it does not contain an induced subgraph that is a $2K_2$, $C_4$, or $C_5$.

\[ \text{bowtie} \quad \text{necktie} \]

**Main Results:**
A graph is a split cluster graph iff it does not contain an induced subgraph that is a $C_4$, $C_5$, $P_5$, necktie, or bowtie.

**Split Cluster Editing** is APX-hard and NP-hard even on graphs with maximum degree 11.

**Split Cluster Editing** can be solved in $O(10^k \cdot m)$ time, where $k$ is the number of necessary edge modifications.
Observation: Monopolar graphs have infinitely many forbidden subgraphs (smallest and only with 5 vertices is the wheel $W_4$).  

Known: Vertex-partitioning into fixed additive induced-hereditary properties is NP-hard [Farrugia, Electron. J. Combin. ’04].

$\Rightarrow$ Deciding whether a graph is monopolar is NP-hard.
ILP formulations

- Forbidden subgraph-based
- Partition variables
- Column generation
Forbidden subgraph-based ILP formulation for SCE

**First try:** use forbidden subgraph characterization

Binary variable $e_{uv} = 1$ if $\{u, v\}$ is in the solution graph
Define $\bar{e}_{uv} := 1 - e_{uv}$

minimize $\sum_{\{u,v\} \in E} \bar{e}_{uv} + \sum_{\{u,v\} \notin E} e_{uv}$

subject to

$\forall$ forbidden subgraph $F : \sum_{\{u,v\} \in F} \bar{e}_{uv} + \sum_{\{u,v\} \notin F} e_{uv} \geq 1$

$O(n^5)$ constraints $\rightsquigarrow$ use row generation (lazy constraints)
Idea: Fix the assignment to core and periphery before destroying the forbidden subgraphs

Lemma: Let \( G = (V, E) \) be a graph and \( C \cup I = V \) a partition of the vertices. Then \( G \) is a split cluster graph with **core vertices** \( C \) and **independent set vertices** \( I \) iff it does not contain an edge with both endpoints in \( I \), nor an induced \( P_3 \) with both endpoints in \( C \).
Partition variable ILP formulation for SCE

Binary variable $e_{uv} = 1$ if $\{u, v\}$ is in the solution graph. Define $\bar{e}_{uv} := 1 - e_{uv}$

Binary variable $c_u = 1$ if $u$ is a core vertex. Define $\bar{c}_u := 1 - c_u$.

$$\text{minimize} \quad \sum_{\{u,v\}\in E} \bar{e}_{uv} + \sum_{\{u,v\}\notin E} e_{uv}$$

subject to
\begin{align*}
\forall u, v : c_u + c_v + \bar{e}_{uv} &\geq 1 \\
\forall u \neq v, v \neq w > u : \bar{e}_{uv} + \bar{e}_{vw} + e_{uw} + \bar{c}_u + \bar{c}_w &\geq 1
\end{align*}

$O(n^3)$ constraints $\rightsquigarrow$ still use row generation (lazy constraints)
Idea (again): Fix the assignment to core and periphery before destroying the forbidden subgraphs

Lemma: Let $G = (V, E)$ be a graph and $C \cup I = V$ a partition of the vertices. Then $G$ is a split cluster graph with core vertices $C$ and independent set vertices $I$ iff it does not contain an edge with both endpoints in $I$, nor an induced $P_3$ consisting only of vertices in $C$. 
Binary variable $e_{uv} = 1$ if $\{u, v\}$ is in the solution graph.
Define $\bar{e}_{uv} := 1 - e_{uv}$

Binary variable $c_u = 1$ if $u$ is a core vertex.
Define $\bar{c}_u := 1 - c_u$.

\[
\text{minimize} \quad \sum_{\{u,v\} \in E} \bar{e}_{uv} + \sum_{\{u,v\} \notin E} e_{uv}
\]
subject to
\[
\forall u, v : c_u + c_v + \bar{e}_{uv} \geq 1
\]
\[
\forall u \neq v, v \neq w > u : \bar{e}_{uv} + \bar{e}_{vw} + e_{uw} + \bar{c}_u + \bar{c}_v + \bar{c}_w \geq 1
\]

$O(n^3)$ constraints $\leadsto$ still use row generation (lazy constraints)
Column generation for Split Cluster Editing

Binary variables $z_C = 1$ if cluster $C \in 2^V$ is part of the solution.

$$\text{maximize} \quad \sum_{C \in 2^V} c_C z_C,$$

$$\text{s.t.} \quad \sum_{C \in 2^V \mid u \in C} z_C = 1 \quad \forall u \in V,$$

where $c_C$ is the “value” of the cluster (number of edges of $G[C]$ minus the splittance of $G[C]$, that is, the number of edge insertions and deletions to make it a split graph).

**Problem:** Exponentially many variables.

**Idea:** Successively add only those variables (“columns”) that are “needed”, that is, their introduction improves the objective.
**Column Generation: Auxiliary problem**

**Lemma:** For the relaxation of the ILP, the objective function change from adding a cluster $C$ is

$$c_C - \sum_{u \in C} \lambda_u,$$

where $\lambda_u$ is the shadow price associated with the constraint of vertex $u$.

$\Rightarrow$ need to find a cluster that maximizes cluster value minus vertex weights.

**Idea:** Use an ILP.
ILP tuning tricks

- Warm start with heuristic solution
- MIP emphasis: balance between proving optimality and finding better solutions
- Cutting planes for $P_5$: for all distinct $u, v, w, x, y \in V$:

\[ \bar{e}_{uv} + \bar{e}_{vw} + \bar{e}_{wx} + \bar{e}_{xy} + \frac{1}{2}e_{uw} + e_{vx} + \frac{1}{2}e_{wy} + \frac{1}{2}e_{xu} + \frac{1}{2}e_{yv} \geq 1. \]

(for monopolar, $W_4$)
Heuristics

- Forbidden subgraph-based
- Simulated annealing
Forbidden subgraph heuristic for Split Cluster Editing

Idea
Edit an edge that destroys many forbidden subgraphs.

Problems
- Slow
- Can get caught in loops
- Not very good results
Simulated Annealing heuristic for Split Cluster Editing

Simulated Annealing

- Start with a clustering where each vertex is a singleton.
- Randomly move a vertex to a cluster that contains one of its neighbors.
- Accept if this improves the objective $k$; otherwise, accept with small probability that decreases over time.

To evaluate the objective, we can use the following theorem:

**Theorem (Hammer & Simeone ’81)**

The minimum number of edits to make a graph a split graph can be found in linear time.
We didn’t find any useful data reduction rules. However, we have two rules that allow to fix the value of variables in the ILP:

**Rule 1**
If there is a degree-one vertex $u$ whose neighbor has degree larger than one, then label $u$ as periphery ($c_u = 0$).

**Rule 2**
If there is an edge $\{u, v\}$ between two vertices labeled as periphery, then this edge cannot be present in the solution ($e_{uv} = 0$).
Experimental Setup

**Data:** three yeast protein interaction subnetworks

Comparison with:
- Core–periphery enumeration algorithm [Luo et al., BMC Bioinformatics ’09]
- SCAN clustering algorithm [Xu et al., KDD ’07]
Experimental Results (I)

Objective value:

|                  | $n$ | $m$ | $k_{\text{SCE}}$ | $k_{\text{ME}}$ |
|------------------|-----|-----|-------------------|------------------|
| cell cycle       | 196 | 797 | 321               | 126              |
| transcription    | 215 | 786 | 273               | 106              |
| translation      | 188 | 2352| 308               | 240              |

Results for Simulated Annealing; confirmed as optimal by ILP in green.
Experimental Results (II)

**GO-term coherence & cluster number:**

|          | $K$ | $p$ | $k$ | $c_t$ | $c_c$ | $c_p$ |
|----------|-----|-----|-----|-------|-------|-------|
| **SCE**  | 13  | 112 | 273 | 0.54  | 0.54  | 0.57  |
| **ME**   | 26  | 78  | 106 | 0.55  | 0.61  | 0.54  |
| **SCAN** | 26  | 58  | —   | 0.53  | 0.51  | 0.47  |
| **Luo**  | 12  | 125 | —   | 0.40  | 0.52  | 0.38  |

$K =$ number of clusters  
$p =$ size of periphery  
$c_t =$ average cluster coherence  
$c_c =$ average core coherence  
$c_p =$ average periphery coherence
Experimental Results (III)

Overlap test with known protein complexes (CYC2008):

Hypothesis for perfect recovery:

- Core contains only complex proteins
- Complex is contained completely in cluster

| transcription | $D$ | core\% | comp\% |
|---------------|-----|--------|--------|
| SCE           | 7 / 11 | 89     | 100    |
| ME            | 11 / 11| 100    | 100    |
| SCAN          | 8 / 11 | 84     | 100    |
| Luo           | 6 / 11 | 87     | 100    |

$D$ : number of detected clusters
core\% : median percentage of core proteins in complex
comp\% : median percentage of complex proteins in cluster
Conclusion

Results:

- Two new concrete graph-theoretic models for uncovering global core–periphery structure of PPI networks
- Useful ILP formulations based on core/periphery-assignment
- Simulated Annealing heuristic performs well
- **Monopolar Editing** gives best biological results

Outlook:

- Algorithmic improvements
  (goal: good results for complete interactome)
- Incorporate interaction confidence scores
- Further combinatorial core–periphery models
- Find further approaches to exploit/evaluate predictions by **Monopolar Editing**