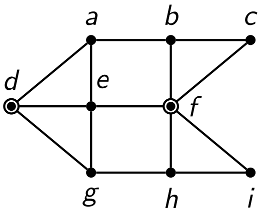
Parameterized algorithms for locating-dominating sets

Márcia Cappelle

Universidade Federal de Goiás - UFG

joint work with Guilherme Gomes, and Vinicius F. dos Santos
(Universidade Federal de Minas Gerais - UFMG)

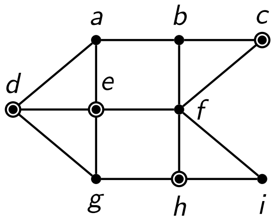
Definitions



Let $S \subseteq V(G)$. S is a dominating set if for all $v \in V(G)$,

$$|N[v] \cap S| \geqslant 1.$$

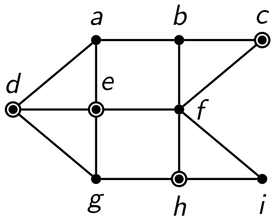
Definitions



Let $S \subseteq V(G)$. S is a locating-dominating set if it is a dominating set and

$$\forall\ u, v \in V(G) \setminus S,\ N(u) \cap S \neq N(v) \cap S.$$

Definitions



Let $S \subseteq V(G)$. S is a locating-dominating set if it is a dominating set and

$$\forall\ u, v \in V(G) \setminus S,\ N(u) \cap S \neq N(v) \cap S.$$

- Introduced by Slater (1983);
- Every graph G has a locating-dominating set, namely V(G);
- $LD(G)$ is the minimum cardinality of such a set.

Applications

- Fault diagnosis in multiprocessors - [Karpovsky et al., (1998)];
- Locating detection in hostile environments - [Ungrangsi, (2003)];
- Environmental monitoring - [Berger-Wolf et al., (2005)];
- Analysis of secondary RNA structures - [Haynes et al., (2006)];

Locating-Dominating Set decision problem

LOCATING-DOMINATING SET - (LD-SET) **Instance:** A graph G and an integer d. **Question:** Does G admit a locating-dominating set of size at most d?

The pair (G, d) is an instance of our problem.

Algorithmic aspects of locating dominating sets

- It is algorithmically hard to determine locating dominating sets of minimum size [Charon, Hudry and Lobstein, (2003)], even for
  - interval graphs and permutation graphs [Foucaud et al., (2015)],
  - bipartite graphs [Charon, Hudry and Lobstein, (2003)], and
  - graphs that are both planar and line graphs of subcubic bipartite graphs [Foucaud et al., (2013)].

Algorithmic aspects of locating dominating sets

- It is algorithmically hard to determine locating dominating sets of minimum size [Charon, Hudry and Lobstein, (2003)], even for
  - interval graphs and permutation graphs [Foucaud et al., (2015)],
  - bipartite graphs [Charon, Hudry and Lobstein, (2003)], and
  - graphs that are both planar and line graphs of subcubic bipartite graphs [Foucaud et al., (2013)].
- It is hard to approximate within factor α for any $\alpha = o(\log n)$ [Berger-Wolf et al. (2006)].
- These inapproximability results were extended to bipartite graphs, split graphs and co-bipartite graphs [Foucaud et al., (2005)].

Algorithmic aspects of locating dominating sets

- There are linear time algorithms that determine a locating dominating set of minimum order for trees [Auger (2010)].
- The optimization problem of determining a locating dominating set of minimum order is expressible in the $LinEMSOL(\tau_1)$ logic [Courcelle et al., (2000)].
  - Therefore, locating dominating sets of minimum order can be determined in polynomial time for the graphs with bounded clique-width (cf. Courcelle's theorem).

Basic definitions on parameterized complexity

A parameterized problem is a pair (x, k), where x is the input of the problem and the parameter k describes some property of x. k is called the parameter. Very useful when $k << |x|$.

Basic definitions on parameterized complexity

A parameterized problem is a pair (x, k), where x is the input of the problem and the parameter k describes some property of x. k is called the parameter. Very useful when $k << |x|$.

A parameterized problem is fixed-parameter tractable (FPT) if it can be solved in $f(k)|x|^{\mathcal{O}(1)}$ time where f is an arbitrary function depending only on k.

- Example: Vertex Cover parameterized by the size of the solution.

Basic definitions on parameterized complexity

A parameterized problem is a pair (x, k), where x is the input of the problem and the parameter k describes some property of x. k is called the parameter. Very useful when $k << |x|$.

A parameterized problem is fixed-parameter tractable (FPT) if it can be solved in $f(k)|x|^{\mathcal{O}(1)}$ time where f is an arbitrary function depending only on k.

- Example: Vertex Cover parameterized by the size of the solution.

A problem is W[1]-hard if there is no $f(k)|x|^{\mathcal{O}(1)}$ time (unless FPT $=$ W[1]-hard).

- Example: Clique parameterized by the size of the solution.

Basic definitions on parameterized complexity

A problem is paraNP-hard if it is NP-hard even if k is a constant.

- Example: Vertex Coloring parameterized by the number of colors.

Basic definitions on parameterized complexity

A problem is paraNP-hard if it is NP-hard even if k is a constant.

- Example: Vertex Coloring parameterized by the number of colors.

A kernelization algorithm (kernel) transforms an instance (x, k) in polynomial time into an equivalent instance (x', k') such that $|x'|, k' \leq f(k)$ for some computable function f, which is the size of the kernel.

Parameterized complexity

- The hardness results imply that the problem is paraNP-hard when parameterized by
  - diameter,
  - distance to interval,
  - distance to split,
  - maximum degree,
  - and genus;

Parameterized complexity

- The hardness results imply that the problem is paraNP-hard when parameterized by
  - diameter,
  - distance to interval,
  - distance to split,
  - maximum degree,
  - and genus;
- Parameterizing by the size of the solution would be trivially FPT since $d \in \Omega(\log n)$.
- The linear-time algorithm for graphs of bounded cliquewidth implies that the problem is FPT under several parameterizations such as
  - cliquewidth,
  - distance to cograph,
  - treewidth,
  - feedback vertex set,
  - max leaf number,
  - and vertex cover.

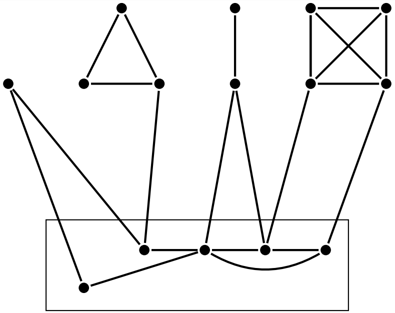
Our results

- We show an exponential kernel for the distance to cluster.
- We present a linear kernel for the max leaf number.
- We prove that unless NP $\subseteq$ coNP/poly, no polynomial kernel exists when parameterizing by
  1. vertex cover and maximum size of the solution,
  2. distance to clique and maximum size of the solution.

An exponential kernel for distance to cluster

Distance to cluster

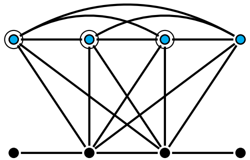
- A graph G is a cluster if each of its connected components is a clique (cluster).
- The distance to cluster of a graph G is the size of the smallest set $U \subseteq V(G)$ such that $G - U$ is a cluster graph.



An exponential kernel for distance to cluster

Reduction Rule 1

If $u, v, w \in V(G) \setminus U$ is a set of mutual twins of G, remove w from G and set $d \leftarrow d - 1$.



- The vertices u and v are called twins if $N_G[u] = N_G[v]$.
- With the application of the reduction rule, each vertex from $\mathcal{Q}$ has at most one twin.
  - The size of $Q \in \mathcal{Q}$ is at most $2^{|U|}$.
  - This can be done in $\mathcal{O}(n + m)$ time.

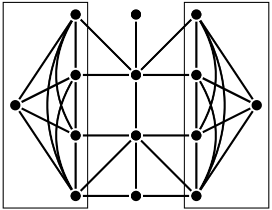
An exponential kernel for distance to cluster

Reduction Rule 2

Let $\mathcal{Q}$ be a pattern of r trivial cliques, each of size $s \geq 2$. If $r \geq 2s + |U| + 2$, remove a clique from $\mathcal{Q}$ and set $d \leftarrow d - 1$.

Reduction Rule 3

Let $\mathcal{Q}$ be a pattern of r non-trivial cliques, each of size $s \geq 2$. If $r \geq |U| + 2$, remove a clique from $\mathcal{Q}$ and set $d \leftarrow d - |\tau(\mathcal{Q})|$.



Two twin cliques.

An exponential kernel for distance to cluster

Theorem 4

When parameterized by the distance to cluster k, Locating-Dominating Set *admits a kernel with $\mathcal{O}\left(2^{8^k+3k}\right)$ vertices and can be computed in $\mathcal{O}(m + n \log n)$ time on graphs with n vertices and m edges.*

An exponential kernel for distance to cluster

Theorem 4

When parameterized by the distance to cluster k, LOCATING-DOMINATING SET *admits a kernel with $\mathcal{O}\left(2^{8^k+3k}\right)$ vertices and can be computed in $\mathcal{O}(m + n\log n)$ time on graphs with n vertices and m edges.*

We show how to obtain a reduced instance (G', d) from (G, d), such that

- each clique of G' has a bounded number of vertices (at most $2^{|U|+1}$);

An exponential kernel for distance to cluster

Theorem 4

When parameterized by the distance to cluster k, Locating-Dominating Set *admits a kernel with $\mathcal{O}\left(2^{8^k+3k}\right)$ vertices and can be computed in $\mathcal{O}(m + n\log n)$ time on graphs with n vertices and m edges.*

We show how to obtain a reduced instance (G', d) from (G, d), such that

- each clique of G' has a bounded number of vertices (at most $2^{|U|+1}$);
- G' has at most $2 \cdot 2^{2^{|U|}}$ patterns and each pattern has at most $\mathcal{O}(2^{|U|})$ cliques;

An exponential kernel for distance to cluster

Theorem 4

When parameterized by the distance to cluster k, LOCATING-DOMINATING SET *admits a kernel with $\mathcal{O}\left(2^{8^k+3k}\right)$ vertices and can be computed in $\mathcal{O}(m + n\log n)$ time on graphs with n vertices and m edges.*

We show how to obtain a reduced instance (G', d) from (G, d), such that

- each clique of G' has a bounded number of vertices (at most $2^{|U|+1}$);
- G' has at most $2 \cdot 2^{2^{|U|}}$ patterns and each pattern has at most $\mathcal{O}(2^{|U|})$ cliques;
- (G, d) and (G', d') are equivalent instances.

Since $|U| \leq 3k$, G' has at most $\mathcal{O}\left(2^{2^{|U|}+|U|}\right) = \mathcal{O}\left(2^{8^k+3k}\right)$ vertices.

An exponential kernel for distance to cluster

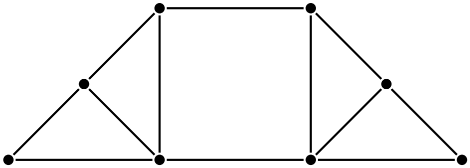
Corollary 5

When parameterized by the distance to clique k, LOCATING-DOMINATING SET *admits a kernel with $\mathcal{O}\left(16^k\right)$ vertices and can be computed in $\mathcal{O}(m+n)$ time on graphs with n vertices and m edges.*

A linear kernel for max leaf number

max leaf number

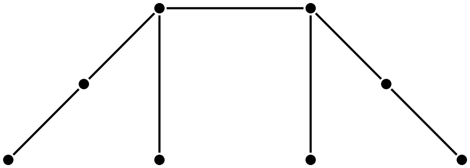
The max leaf number of a connected graph G is the maximum, over all spanning trees of G, of the number of leaves in the spanning tree.



A linear kernel for max leaf number

max leaf number

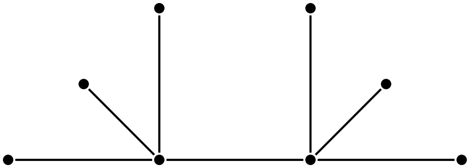
The max leaf number of a connected graph G is the maximum, over all spanning trees of G, of the number of leaves in the spanning tree.



A linear kernel for max leaf number

max leaf number

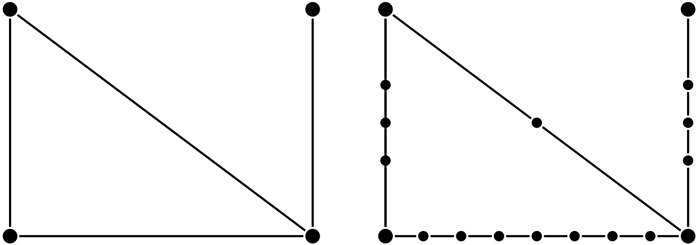
The max leaf number of a connected graph G is the maximum, over all spanning trees of G, of the number of leaves in the spanning tree.



A linear kernel for max leaf number

Some useful facts and observations:

- If G has max leaf number k, then G is the subdivision of a graph H on $4k$ vertices [Estivill-Castro et al., (2005)]; H is called the *host graph* of G.



- Since computing such a graph is an NP-hard problem ([Garey and Johnson, (1979)], we assume that H is part of the input of our kernelization algorithm.

A linear kernel for max leaf number

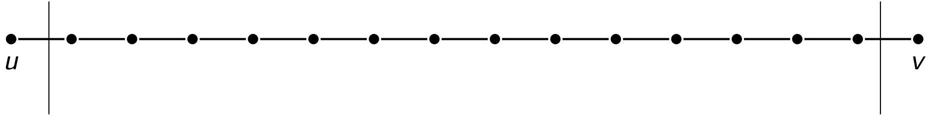
Theorem 6

When parameterized by the maximum leaf number, LOCATING-DOMINATING SET *admits a kernel with at most* $108k + \left\lfloor \frac{k}{2} \right\rfloor$ *vertices that can be computed in* $\mathcal{O}(k(n+m))$*, where* $n = |V(G)|$ *and* $m = |E(G)|$*.*

A linear kernel for max leaf number

Lemma 7

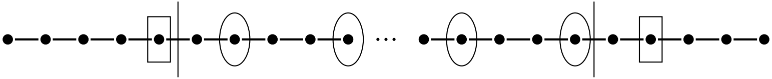
If G has max leaf number k and H is its host graph, then $|\mathcal{P}_2(G)| \leq 5k - 1 + \left\lfloor \frac{k}{2} \right\rfloor$.



- $P(u, v)$ be the path that replaced $uv \in E(H)$ in order to obtain G.
- $\mathcal{P}_2(G)$ is the set of paths formed by vertices of $G - H$.
- We upper bound $|\mathcal{P}_2(G)|$ by $5k - 1 + \left\lfloor \frac{k}{2} \right\rfloor$.
  - H has at most $4k$ vertices;
  - The spanning tree $T(G)$ of G has L leaves and $|L| = k$;
  - There is at most one path $P(u, v)$ between two leaves in L;

A linear kernel for max leaf number

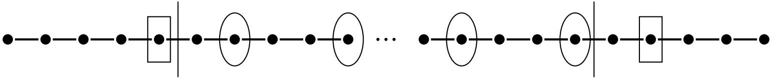
- The algorithm identifies long paths (at least 20 vertices) obtained through the subdivision of H and replaces them with shorter paths (at most 15 vertices).
- It analyses every possible configuration.
- In the reduced instance each path in $\mathcal{P}_2(G)$ has at most 19 vertices.
- The reduced instance has at most $4k + 19(5k + \lfloor \frac{k}{2} \rfloor - 1) = 108k + \lfloor \frac{k}{2} \rfloor - 1$ vertices.



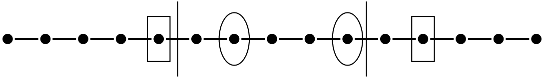
a) Original path. Case (i).

A linear kernel for max leaf number

- The algorithm identifies long paths (at least 20 vertices) obtained through the subdivision of H and replaces them with shorter paths (at most 15 vertices).
- It analyses every possible configuration.
- In the reduced instance each path in $\mathcal{P}_2(G)$ has at most 19 vertices.
- The reduced instance has at most $4k + 19(5k + \lfloor \frac{k}{2} \rfloor - 1) = 108k + \lfloor \frac{k}{2} \rfloor - 1$ vertices.



a) Original path. Case (i).



b) Replaced path.

Lower bounds for kernelization

- A natural question for any FPT problem is whether it admits polynomial-time kernelization to a problem kernel that in the worst case is bounded by a polynomial function of the parameter.

Lower bounds for kernelization

- A natural question for any FPT problem is whether it admits polynomial-time kernelization to a problem kernel that in the worst case is bounded by a polynomial function of the parameter.
- There are FPT problems that have only kernels of exponential size (under complexity-theoretic assumptions).

Lower bounds for kernelization

- We say that an NP-hard problem Γ OR-cross-composes [Bodlaender et al. (2011)] into a parameterized problem Π if, given t instances $Y = \{y_0, \ldots, y_{t-1}\}$ of Γ, we can construct an instance (x, k) of Π in time polynomial in $\sum_{y \in Y} |y|$ and:
  - (i) (x, k) is a YES instance if and only if at least one $y \in Y$ is also a YES instance, and
  - (ii) it holds that $k \leq \text{poly}(\max_{y \in Y} |y| \log t)$.

Lower bounds for kernelization

- We say that an NP-hard problem Γ OR-cross-composes [Bodlaender et al. (2011)] into a parameterized problem Π if, given t instances $Y = \{y_0, \ldots, y_{t-1}\}$ of Γ, we can construct an instance (x, k) of Π in time polynomial in $\sum_{y \in Y} |y|$ and:
  - (i) (x, k) is a YES instance if and only if at least one $y \in Y$ is also a YES instance, and
  - (ii) it holds that $k \leq \text{poly}(\max_{y \in Y} |y| \log t)$.
- It is required that the size of the output parameter is polynomially bounded in the size of the largest input instance.
- The output parameter may depend polynomially on the logarithm of the number of input instances, which often simplifies the constructions and proofs.

Lower bounds for kernelization

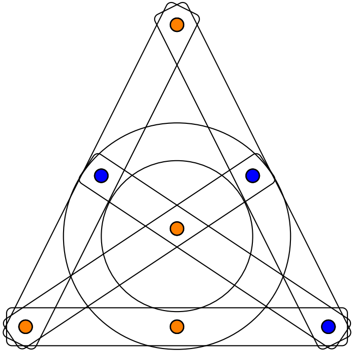
Theorem 8

LOCATING-DOMINATING SET *does not admit a polynomial kernel when parameterized by vertex cover and maximum size of the solution unless* $NP \subseteq coNP/poly$.

Lower bounds for kernelization

3-uniform hypergraph bicoloring

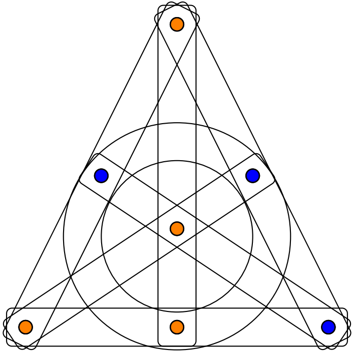
A hypergraph $H = (U, E)$ with vertices U and edges $E \subseteq 2^V$ is 3-uniform if every edge in E has exactly 3 vertices. A bicoloring of a hypergraph H is a coloring of its vertices by using two colors such that no edge of H is monochromatic.



Lower bounds for kernelization

3-uniform hypergraph bicoloring

A hypergraph $H = (U, E)$ with vertices U and edges $E \subseteq 2^V$ is 3-uniform if every edge in E has exactly 3 vertices. A bicoloring of a hypergraph H is a coloring of its vertices by using two colors such that no edge of H is monochromatic.



Lower bounds for kernelization - Vertex Cover

We show that the NP-hard problem 3-Uniform Hypergraph Bicoloring OR-cross-composes into Locating-Dominating Set parameterized by vertex cover and size of the solution.

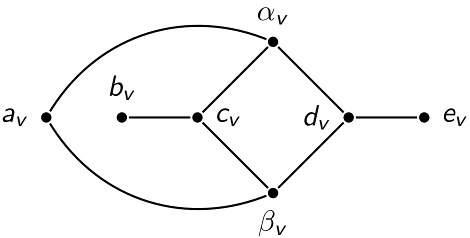
We consider

- $\mathcal{H} = \{H_0, \ldots, H_t\}$ the input instances to 3-Uniform Hypergraph Bicoloring, with $H_i = (U, E_i)$, all of which have n vertices, and
- (G, d) is the instance of Locating-Dominating Set we have built.

We prove

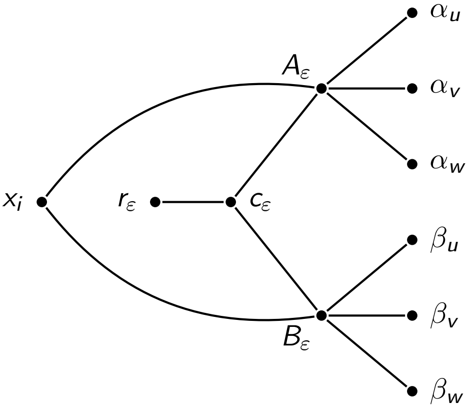
- (G, d) admits a solution if and only if at least one H_i admits a valid bicoloring.
- G has a vertex cover with $\mathcal{O}(n^3 + \log|\mathcal{H}|)$ vertices.

Lower bounds for kernelization



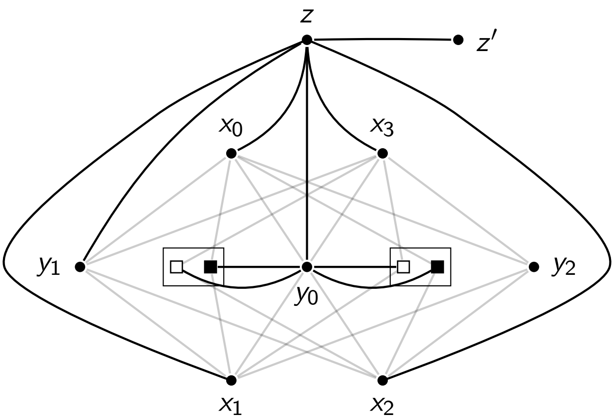
Binary choice gadget $G_v(\alpha, \beta)$.

Lower bounds for kernelization



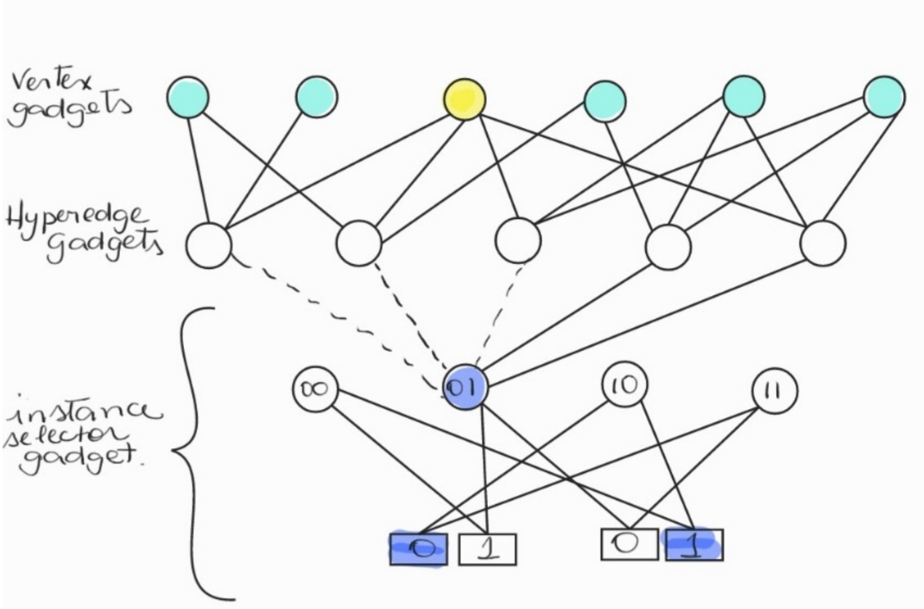
Hyperedge gadget for some $\varepsilon \in \mathcal{E}$ with $\varepsilon = \{u, v, w\}$.

Lower bounds for kernelization



Instance selector gadget for $h = \log_2 |\mathcal{H}|$. Black squares are vertices of the form 1_j, while white squares the 0_j vertices.

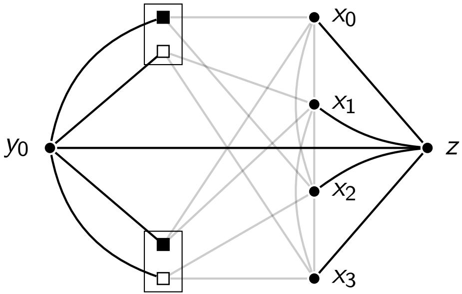
Lower bounds for kernelization



Lower bounds for kernelization

Theorem 9

LOCATING-DOMINATING SET *does not admit a polynomial kernel when parameterized by distance to clique and maximum size of the solution unless NP $\subseteq$ coNP/poly.*

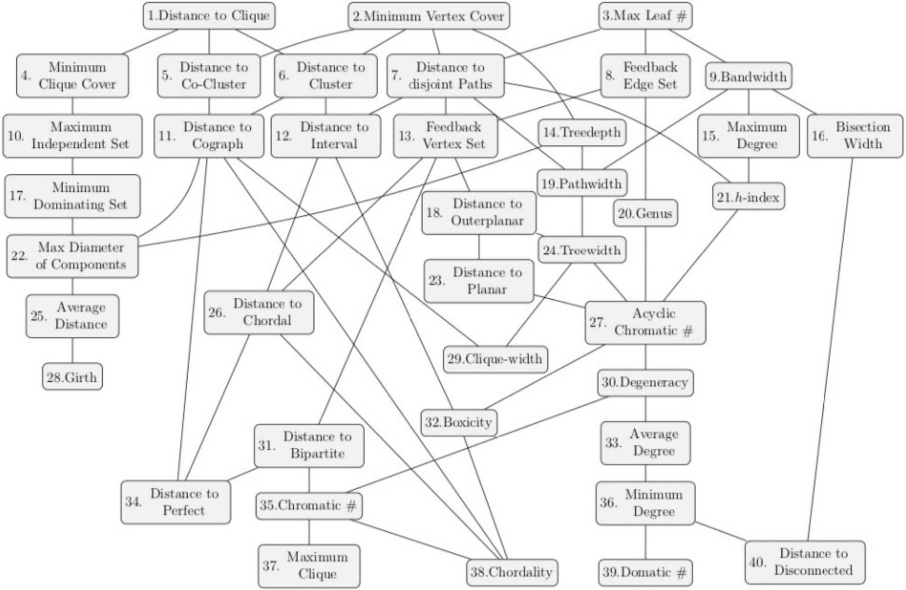


Instance selector gadget for $h = \log_2 |\mathcal{H}|$. Black squares are vertices of the form 1_j, while white squares the 0_j vertices.

Final considerations

- We show an exponential kernel for the distance to cluster,
- We prove that unless NP $\subseteq$ coNP/poly, no polynomial kernel exists when parameterizing by
  1. vertex cover and maximum size of the solution,
  2. distance to clique and maximum size of the solution.
- We present a linear kernel for the max leaf number.
- Some open problems:
  - parameterization by feedback edge set;
  - investigation of the Identifying Code (vertices *inside* the dominating set have a unique (closed) neighborhood in the solution).

Graph parameter Hierarchy



Thank you for your attention!