An $O^*((2 + \epsilon)^k)$ Time Algorithm for Cograph Deletion Using Unavoidable Subgraphs in Large Prime Graphs

Manuel Lafond **Université de Sherbrooke**

Francis Sarrazin Université de Sherbrooke

For a set of graphs $\mathcal{H}$, a graph G is $\mathcal{H}$-free if it contains no **induced** subgraph isomorphic to a graph in $\mathcal{H}$.

The $\mathcal{H}$-free-deletion problem

Input: a graph G.

Parameter: k

Question: can G be made $\mathcal{H}$-free by deleting at most k edges?

$\mathcal{H}$-free-completion: same, but must add edges

$\mathcal{H}$-free-editing: same, but can add/remove edges

Review article

A survey of parameterized algorithms and the complexity of edge modification ☆

Christophe Crespelle [a], Pål Grønås Drange [b,*], Fedor V. Fomin [b], Petr Golovach [b]

Table 1
Kernelization complexity of edge modification problems into hereditary graph classes characterized by a finite number of forbidden induced subgraphs. NOKER means that the problem does not have a polynomial kernel unless NP $\subseteq$ coNP/poly. OPEN means that the complexity is open, while "—" means that the problem is probably open but most likely nobody looked at this question. P means the problem is solvable in polynomial time. A dagger next to the name of the class marks self-complementary classes, for which any result for one of the completion problem or deletion problem automatically gives the same result for the other problem.

Graph class	Polynomial kernel		
	COMPLETION	DELETION	EDITING
line	OPEN	OPEN	OPEN
s-plex cluster	—	—	s^2k [36]
chain ($\{K_3, 2K_2, C_5\}$)	as deletion	k^2 [37,38]	k^2 [38]
starforest ($\{K_3, C_4, P_4\}$)	P	$4k$ [39]	as deletion
threshold† ($\{2K_2, C_4, P_4\}$)	k^2 [38]	k^2 [38]	k^2 [38]
split† ($\{2K_2, C_4, C_5\}$)	k [39], $5k^{1.5}$ [40]	k [39], $5k^{1.5}$ [40]	P [9]
clique + IS ($\{P_3, 2K_2\}$)	P	$k/\log k$ [39]	$2k$ [folkl.]
trivially perfect ($\{C_4, P_4\}$)	k^2 [39,40]	k^3 [41]	k^3 [41]
$\{claw, diamond\}$	OPEN	$k^{\mathcal{O}(1)}$ [42]	OPEN
pseudosplit† ($\{2K_2, C_4\}$)	$5k^{1.5}$ [40]	$5k^{1.5}$ [40]	P [9,43]
cluster ($\{P_3\}$)	P	$2k$ [40]	$2k$ [44,45]
$\{K_3\}$	P	$6k$ [46]	as deletion
cograph† ($\{P_4\}$)	k^3 [28]	k^3 [28]	k^3 [28]
$\{paw\}$	k [47]	k^3 [48]	k^6 [48]
$\{diamond\}$	P	k^3 [49,50]	k^8 [50]
$\{claw\}$	OPEN	OPEN	OPEN
$\{K_4\}$	P	k^3 [51]	as deletion
$\{P_\ell\}$, $\ell > 4$	NOKER [52]	NOKER [29]	NOKER [52]
$\{C_\ell\}$, $\ell > 3$	NOKER [52]	NOKER [52]	NOKER [52]

Cograph modification problems

A *cograph* is a $\{P_4\}$-free graph.

Cograph-deletion = $\{P_4\}$-free deletion

Cograph modification problems

A *cograph* is a $\{P_4\}$-free graph.

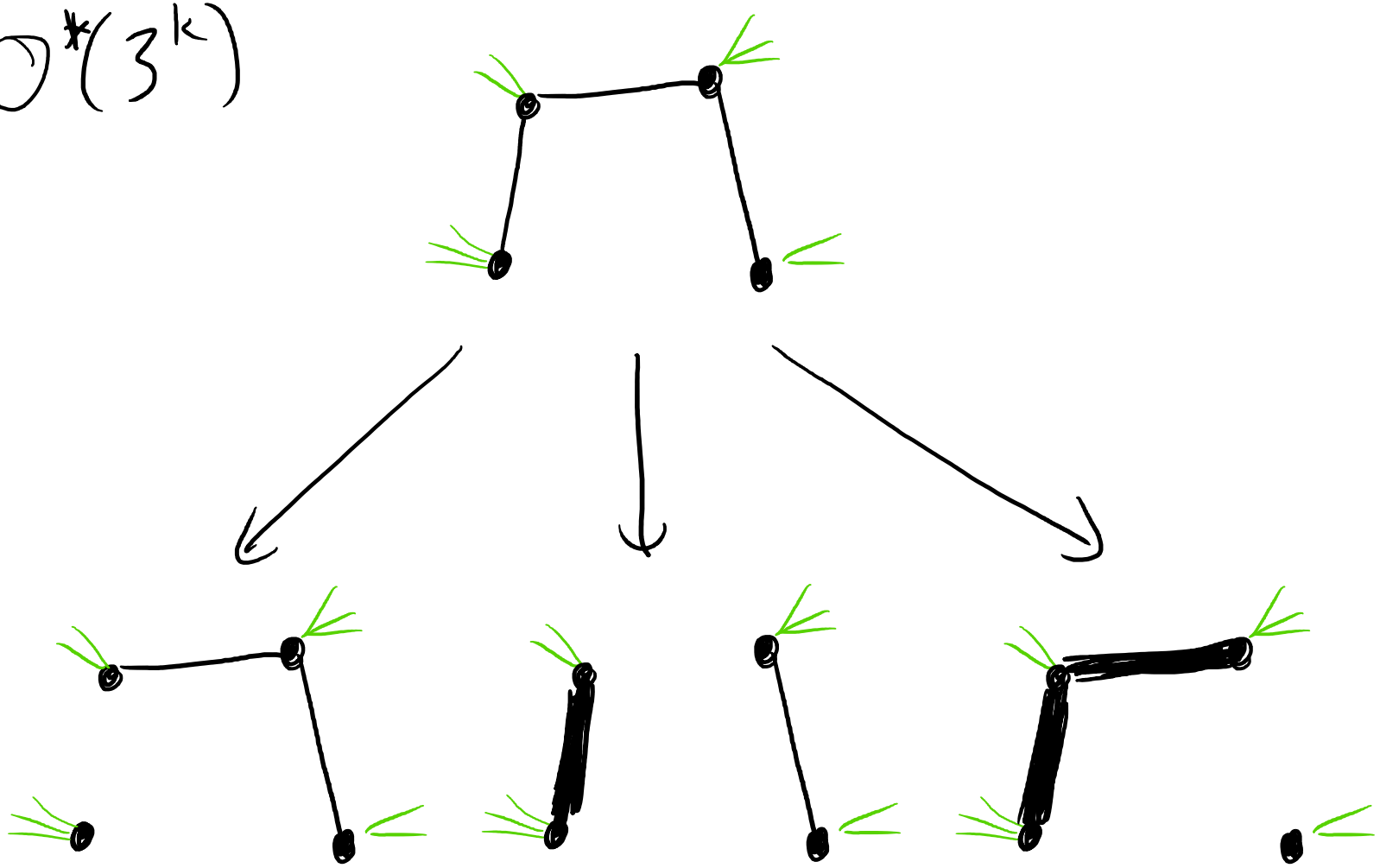
Cograph-deletion = $\{P_4\}$-free deletion

Simple textbook $O^*(3^k)$ time algorithm.

$O^*\left(2.562^k\right)$ [Nastos & Gao, 2012]

$O^*\left(2.303^k\right)$ [Tsur, 2020]

$O^*(3^k)$



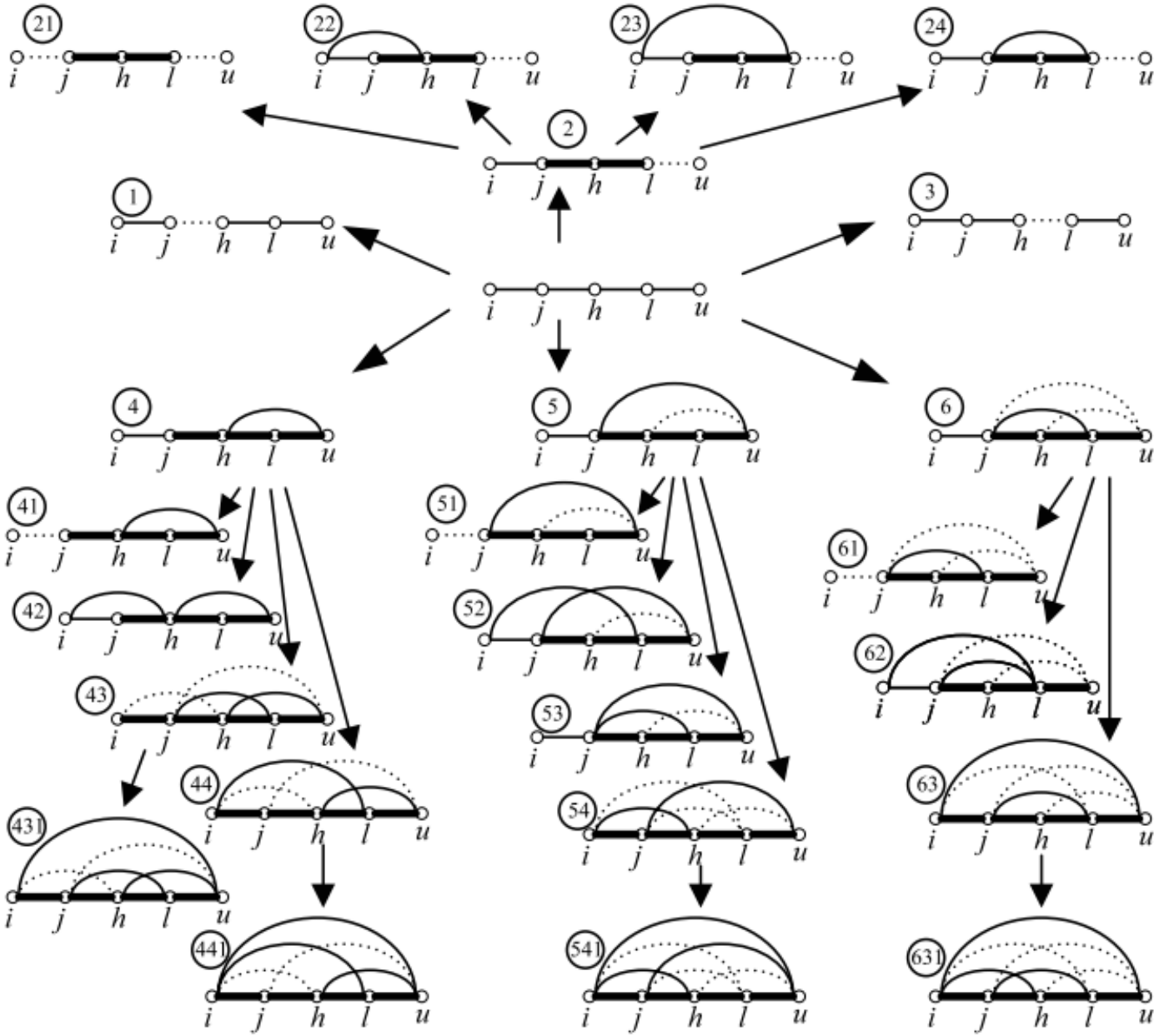


Fig. 5. Branching on a P_5. All bold lines and dashed lines are annotated as permanent and forbidden, respectively. Moreover, the dashed lines on the P_5 denote deleted edges, while the bold lines not on the P_5 are added edges.

Cograph modification problems

A *cograph* is a $\{P_4\}$-free graph.

Cograph-deletion = $\{P_4\}$-free deletion

Simple textbook $O^*(3^k)$ time algorithm.

$O^*\left(2.562^k\right)$ [Nastos & Gao, 2012]

$O^*\left(2.303^k\right)$ [Tsur, 2020]

[Tsur, 2020] needs a python script to generate and analyze possible cases.

In this work

Theorem

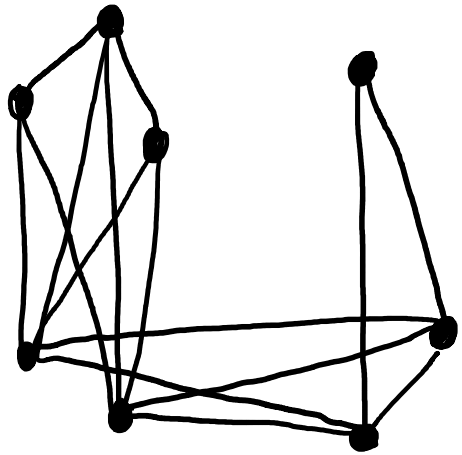
For any $\epsilon > 0$, the Cograph-deletion problem can be solved in time $O^*((2+\epsilon)^k)$.

Approach based on modular decompositions.

Hidden polynomial factors have the form $n^{f(\epsilon)}$ for some (existing) function f.

Modules, modular decompositions, and prime graphs

Module of G: a subset of vertices M such that every vertex of M has the same set of neighbors outside of M.



Prime graph: only modules are individual vertices or $V(G)$.

Modular decomposition: partition $M^* = \{M_1, \dots, M_p\}$ into maximal strong modules. (see paper for definitions)

G/M^*: graph obtained by contracting each M_i. (quotient)

Is either (1) disconnected; (2) its complement is disconnected; (3) or it is prime.

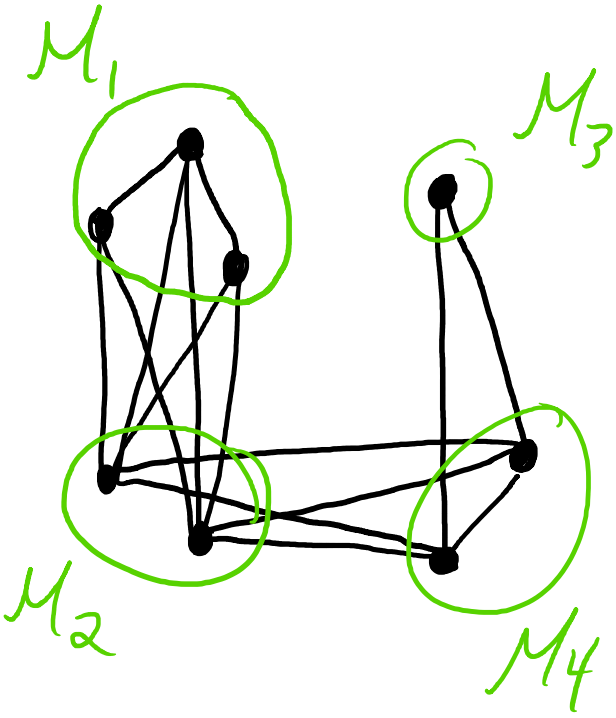
Module of G: a subset of vertices M such that every vertex of M has the same set of neighbors outside of M.

Prime graph: only modules are individual vertices or $V(G)$.

Modular decomposition: partition $M^* = \{M_1, \dots, M_p\}$ into maximal strong modules. (see paper for definitions)

G/M^*: graph obtained by contracting each M_i. (quotient)

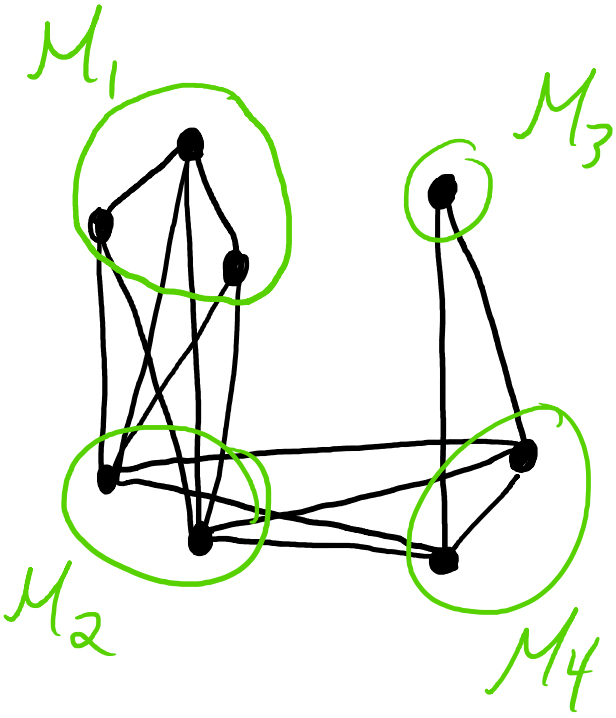
Is either (1) disconnected; (2) its complement is disconnected; (3) or it is prime.



Module of G: a subset of vertices M such that every vertex of M has the same set of neighbors outside of M.

Prime graph: only modules are individual vertices or $V(G)$.

Modular decomposition: partition $M^* = \{M_1, \dots, M_p\}$ into maximal strong modules. (see paper for definitions)

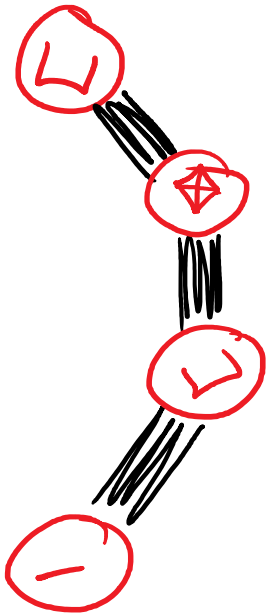


G/M^*: graph obtained by contracting each M_i. (quotient)

Is either (1) disconnected; (2) its complement is disconnected; (3) or it is prime.

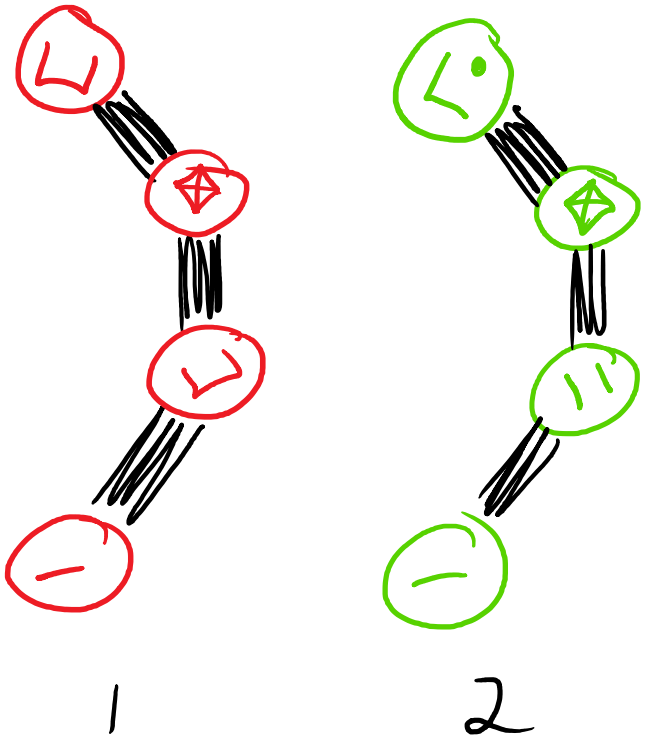
A generic $\mathcal{H}$-free modification approach

1. Compute $M^* = \{M_1, \dots, M_p\}$.
2. Solve each $G[M_i]$ recursively.
3. Solve G/M^*.
4. Apply the edge modifications of G corresponding to those on G/M^*.



A generic $\mathcal{H}$-free modification approach

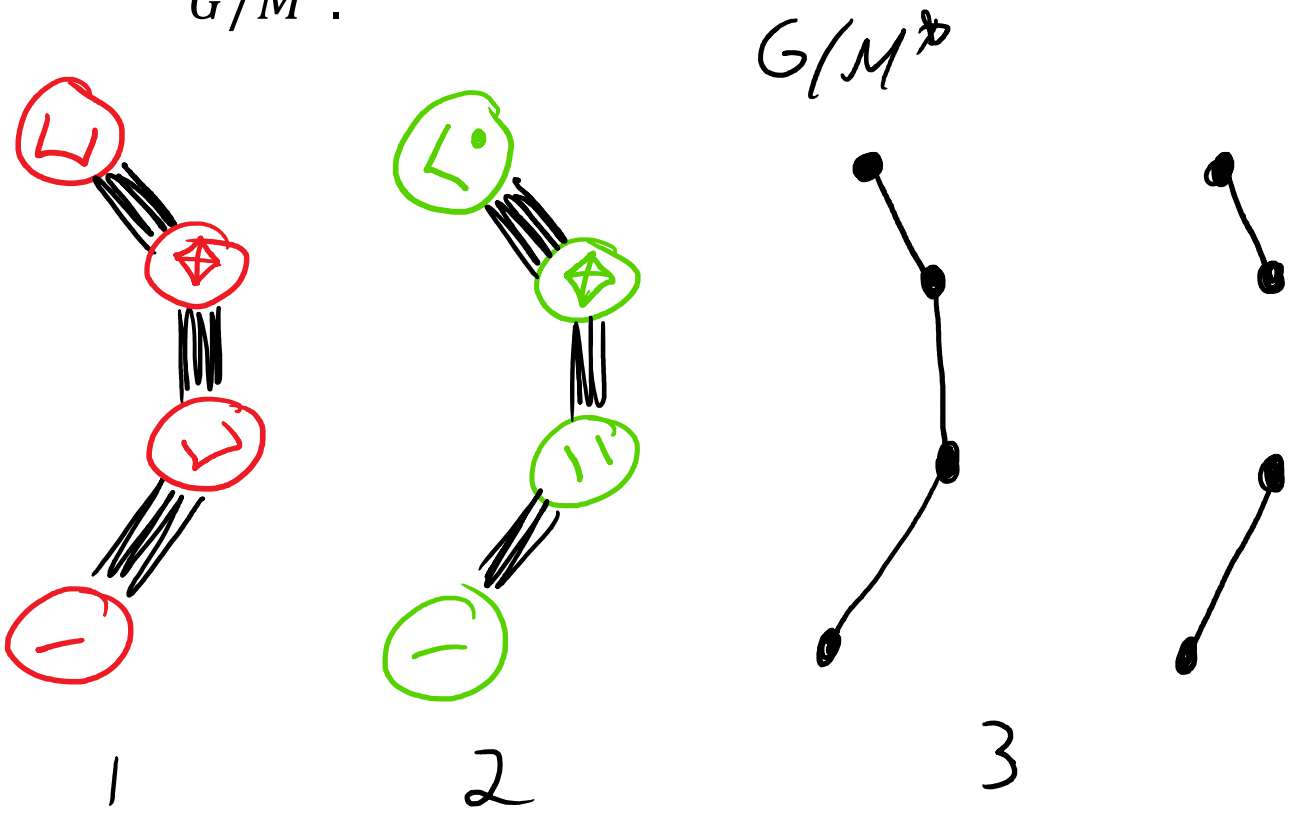
1. Compute $M^* = \{M_1, \dots, M_p\}$.
2. Solve each $G[M_i]$ recursively.
3. Solve G/M^*.
4. Apply the edge modifications of G corresponding to those on G/M^*.



1 2

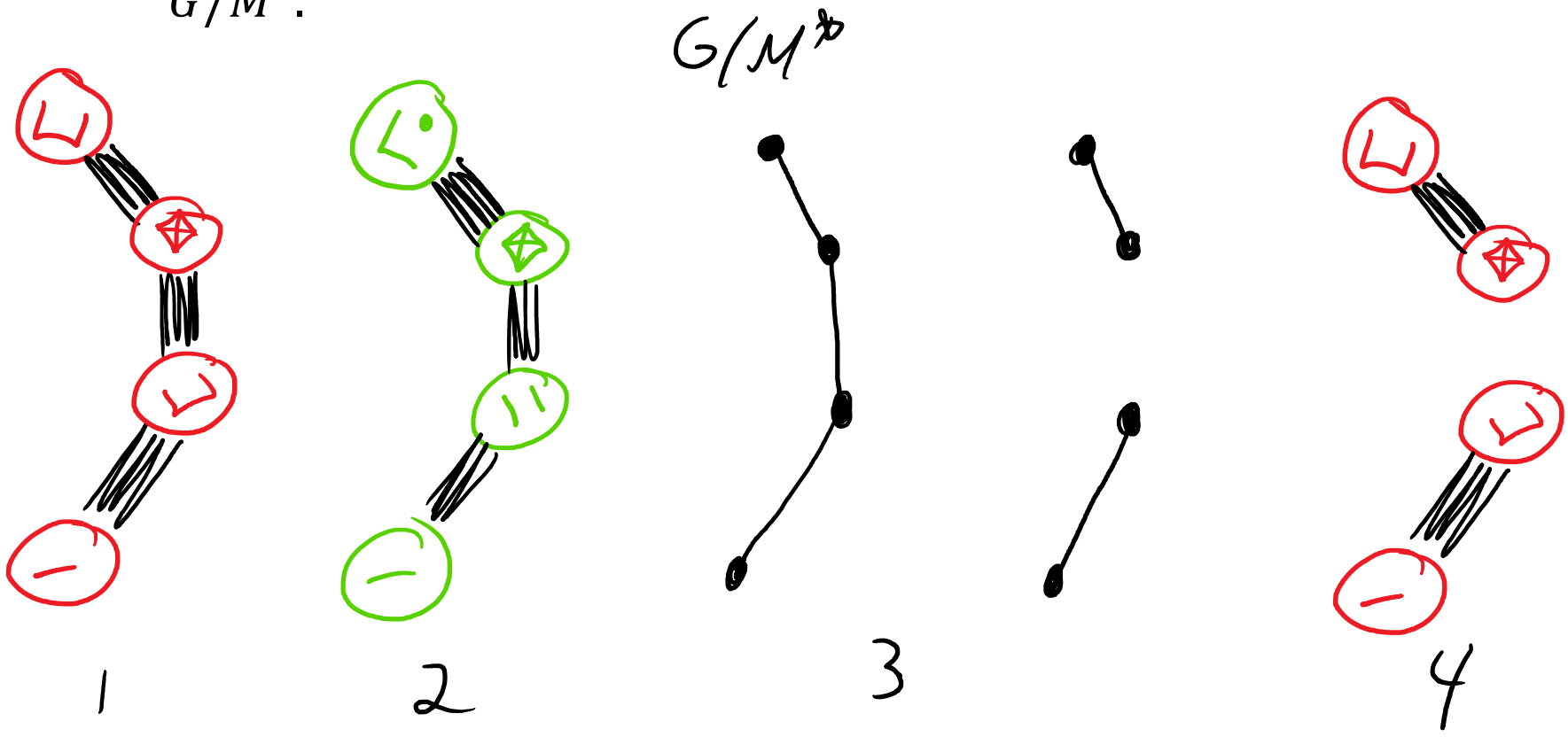
A generic $\mathcal{H}$-free modification approach

1. Compute $M^* = \{M_1, \dots, M_p\}$.
2. Solve each $G[M_i]$ recursively.
3. Solve G/M^*.
4. Apply the edge modifications of G corresponding to those on G/M^*.



A generic $\mathcal{H}$-free modification approach

1. Compute $M^* = \{M_1, \dots, M_p\}$.
2. Solve each $G[M_i]$ recursively.
3. Solve G/M^*.
4. Apply the edge modifications of G corresponding to those on G/M^*.



A generic $\mathcal{H}$-free modification approach

1. Compute $M^* = \{M_1, \dots, M_p\}$.
2. Solve each $G[M_i]$ recursively.
3. Solve G/M^*.
4. Apply the edge modifications of G corresponding to those on G/M^*.

Core problem: solving on G/M^* when it is a prime graph.

A generic $\mathcal{H}$-free modification approach

1. Compute $M^* = \{M_1, \dots, M_p\}$.
2. Solve each $G[M_i]$ recursively.
3. Solve G/M^*.
4. Apply the edge modifications of G corresponding to those on G/M^*.

Core problem: solving on G/M^* when it is a prime graph.

When can we leverage the structure of prime graphs?

- Sometimes feasible when large enough.

Algorithm 1 FPT $\mathcal{H}$-FREE EDITING(σ) on a vertex-weighted graph G.

function $f(G = (V, E, \omega),\ k)$

  if $k \geq 0$ *and* G *is* $\mathcal{H}$*-free* **then return** 0;

  if $k \leq 0$ **then return** INF;

  if $|V| < C$ **then return solve**(G, k);

  Let $e = 0$ and let $\mathcal{M}^*$ be the modular decomposition of G;

  if $|V(G/\mathcal{M}^*)| \geq C$ *and* $G/\mathcal{M}^*$ *is not* $\mathcal{H}$*-free* **then**

$$e = \min_{\mathcal{E} \in \textbf{branch}(G/\mathcal{M}^*)} f(G\Delta ext(G, \mathcal{M}^*, \mathcal{E}), k - cost(ext(G, \mathcal{M}^*, \mathcal{E}))) + cost(ext(G, \mathcal{M}^*, \mathcal{E}));$$

  else $e =$ **solve**$(G/\mathcal{M}^*, k) + \sum_{M \in \mathcal{M}^*} f(G[M], k)$;

  if $e \leq k$ **then return** e;

  return INF ;

Theorem

Algorithm 1 is correct if and only if every minimal graph in $\mathcal{H}$ is prime.

(minimal: does not contain an induced copy of another graph in $\mathcal{H}$)

Theorem

Algorithm 1 is correct if and only if every minimal graph in $\mathcal{H}$ is prime.

(minimal: does not contain an induced copy of another graph in $\mathcal{H}$)

- Graphs of size 1 or 2 in $\mathcal{H}$ are trivial to handle, let us ignore them.
- No graph of size 3 is prime.
- Smallest possible $\mathcal{H}$ that this works on: $\mathcal{H} = \{P_4\}$

Theorem

Algorithm 1 is correct if and only if every minimal graph in $\mathcal{H}$ is prime.

(minimal: does not contain an induced copy of another graph in $\mathcal{H}$)

- Graphs of size 1 or 2 in $\mathcal{H}$ are trivial to handle, let us ignore them.
- No graph of size 3 is prime.
- Smallest possible $\mathcal{H}$ that this works on: $\mathcal{H} = \{P_4\}$

Note: "only if" needs to show that:

If $\mathcal{H}$ has some minimal non-prime graph, then there are infinitely many non-$\mathcal{H}$-free graphs but such that each $G[M_i]$ and G/M^* are $\mathcal{H}$-free.

Unavoidable induced subgraphs in prime graphs

UNAVOIDABLE INDUCED SUBGRAPHS IN LARGE GRAPHS WITH NO HOMOGENEOUS SETS

MARIA CHUDNOVSKY, RINGI KIM, SANG-IL OUM, AND PAUL SEYMOUR

▶ **Theorem 8** ([5]). *For every integer $c \geq 3$, there exists a constant C such that every prime graph with at least C vertices contains one of the following graphs or their complements as an induced subgraph.*

1. *The 1-subdivision of $K_{1,c}$ (denoted by $K_{1,c}^{(1)}$) (Figures 1 and 2).*
2. *The line graph of $K_{2,c}$, where $K_{2,c}$ is a complete bipartite graph with two vertices on one side and c on the other (Figures 3 and 4).*
3. *The thin spider with c legs, which has an independent set $\{v_1, \ldots, v_c\}$, a clique $\{w_1, \ldots, w_c\}$, and edges $v_i w_i$ for $i \in [c]$ (Figures 5 and 6). The complement of a thin spider is called a thick spider.*
4. *The half-graph of height c, denoted H_c, which has two independent sets $\{v_1, \ldots, v_c\}, \{w_1, \ldots, w_c\}$ and edge $v_i w_i$ iff $i \leq j$ (Figures 7 and 8).*
5. *The graph $H'_{c,I}$, obtained from H_c by making the w_i's a clique and adding a vertex u adjacent to all the v_i's (Figure 9). This graph is self-complementary.*
6. *The graph H^*_c, obtained from $H'_{c,I}$ by making u only adjacent to v_1 (Figures 10 and 11).*
7. *A chain with $c+1$ vertices that induces a prime graph.*

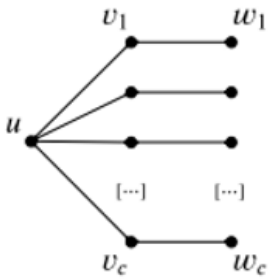


Figure 1 $K_{1,c}$

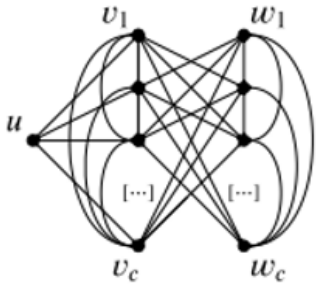


Figure 2 $\overline{K_{1,c}}$ (note, the vertex labels are not in correspondence with Figure 1 since the v_i, w_i's were swapped)

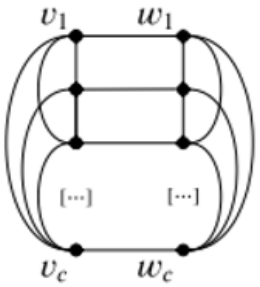


Figure 3 $L(K_{2,c})$

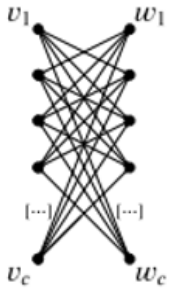


Figure 4 $\overline{L(K_{2,c})}$

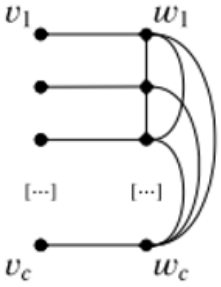


Figure 5 *Thin Spider*

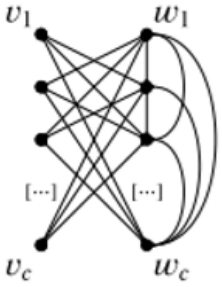


Figure 6 *Thick Spider*

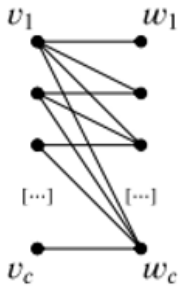


Figure 7 H_c

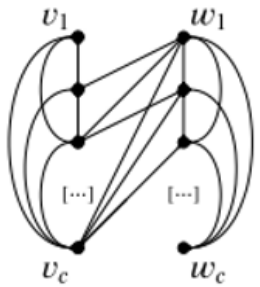


Figure 8 $\overline{H_c}$

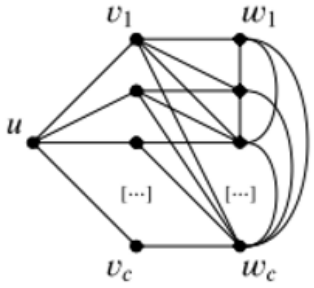


Figure 9 $H'_{c,I}$

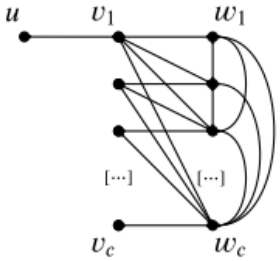


Figure 10 H^*

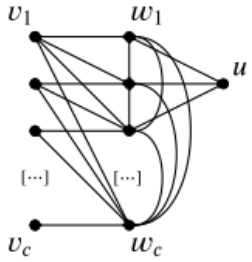


Figure 11 $\overline{H^*}$. Note, to ease future analysis the labels from H^* do not correspond. The w_c vertex from $\overline{H}^*$ is u from H^*, u is v_1 from H^*, and $w_1, \ldots, w_{c_1}$ is $v_2, \ldots, v_c$.

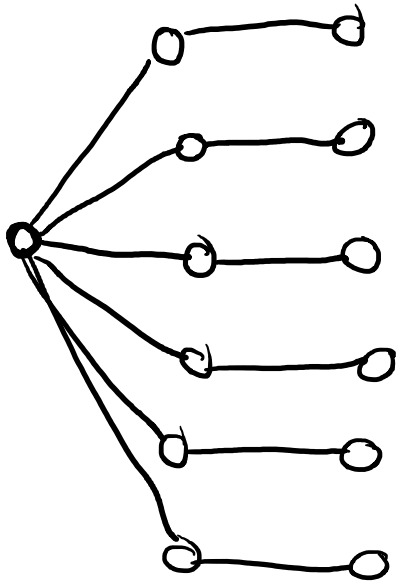
▶ **Theorem 8** ([5]). *For every integer $c \geq 3$, there exists a constant C such that every prime graph with at least C vertices contains one of the following graphs or their complements as an induced subgraph.*

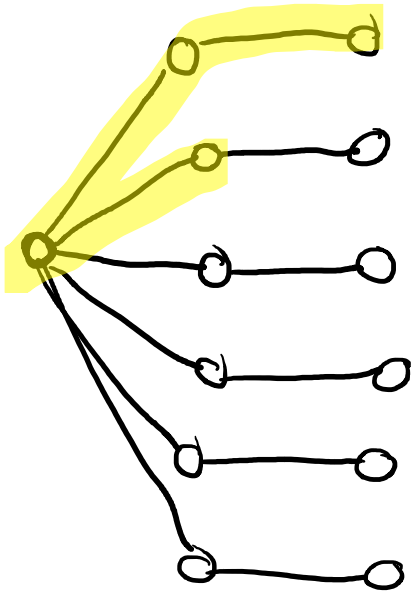
1. *The 1-subdivision of $K_{1,c}$ (denoted by $K^{(1)}_{1,c}$) (Figures 1 and 2).*
2. *The line graph of $K_{2,c}$, where $K_{2,c}$ is a complete bipartite graph with two vertices on one side and c on the other (Figures 3 and 4).*
3. *The thin spider with c legs, which has an independent set $\{v_1, \ldots, v_c\}$, a clique $\{w_1, \ldots, w_c\}$, and edges $v_i w_i$ for $i \in [c]$ (Figures 5 and 6). The complement of a thin spider is called a thick spider.*
4. *The half-graph of height c, denoted H_c, which has two independent sets $\{v_1, \ldots, v_c\}, \{w_1, \ldots, w_c\}$ and edge $v_i w_i$ iff $i \leq j$ (Figures 7 and 8).*
5. *The graph $H'_{c,I}$, obtained from H_c by making the w_i's a clique and adding a vertex u adjacent to all the v_i's (Figure 9). This graph is self-complementary.*
6. *The graph H^*_c, obtained from $H'_{c,I}$ by making u only adjacent to v_1 (Figures 10 and 11).*
7. *A chain with $c+1$ vertices that induces a prime graph.*

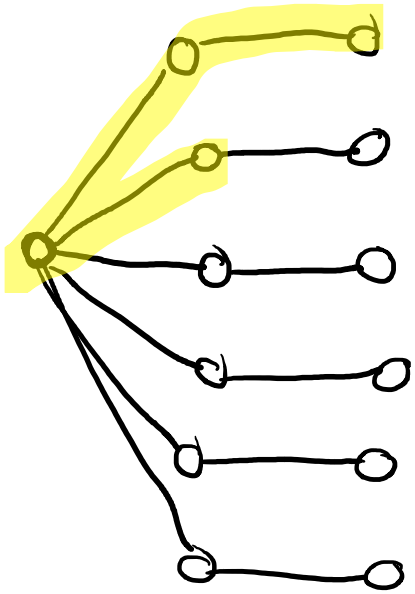
The cograph-deletion algorithm

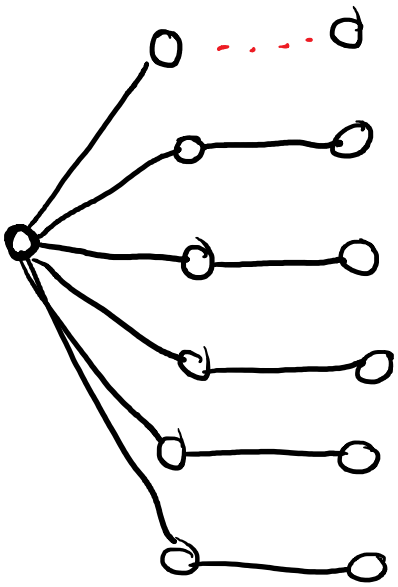
To solve prime graph G/M^* in time $O^*((2+\epsilon)^k)$:

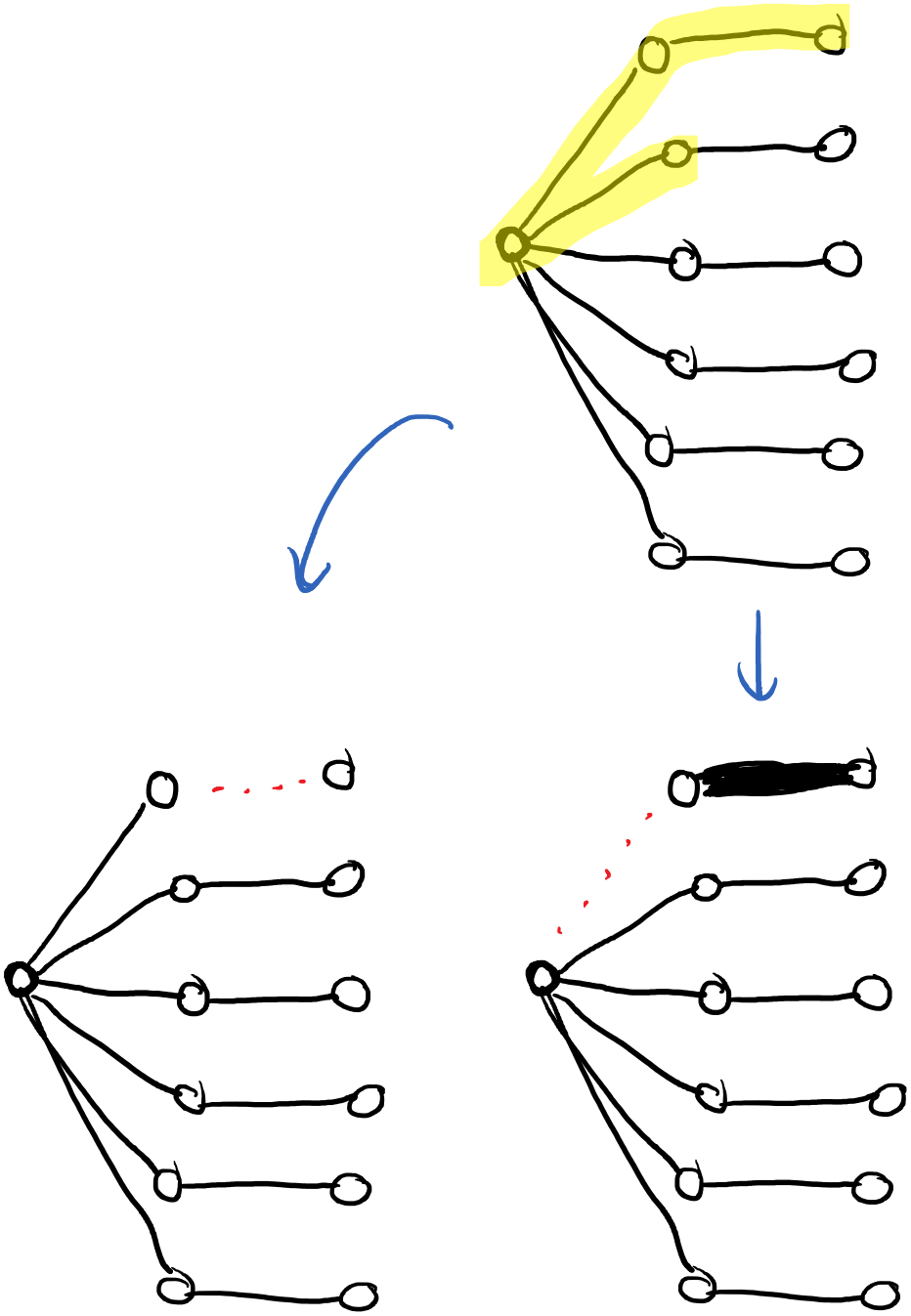
- Given the desired ϵ, choose a suitable c. Then let C be the large enough constant w.r.t. c.
- If $|V(G/M^*)| \leq C$, solve it with brute-force.
- Otherwise, find one of the unavoidable induced subgraphs. (time $O(n^c)$) (exponent is small c, not capital C)
- Do some branching.
  - Each possible subgraph can achieve branching factor $(2+\epsilon)$.
  - Chains are a pain.

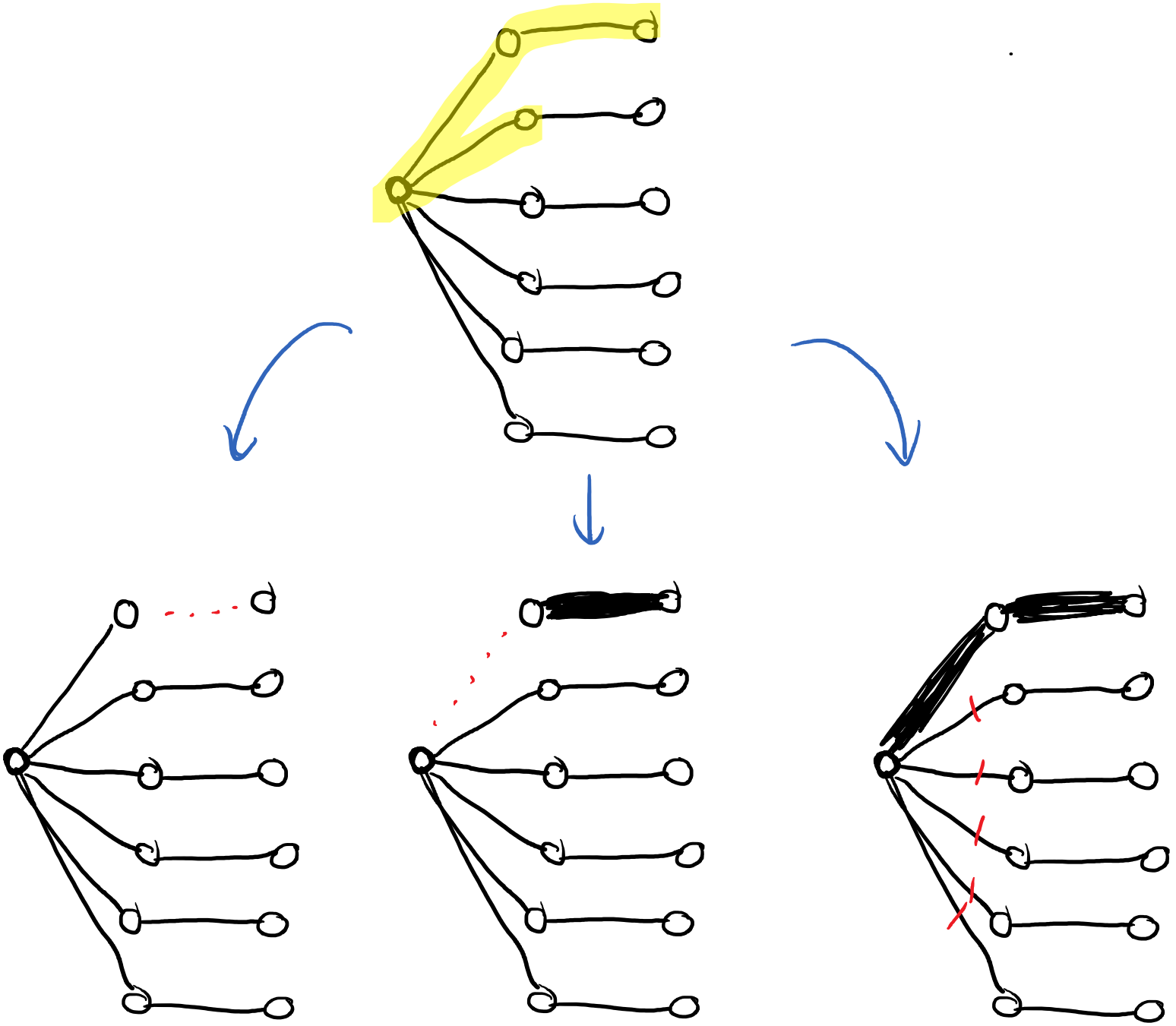


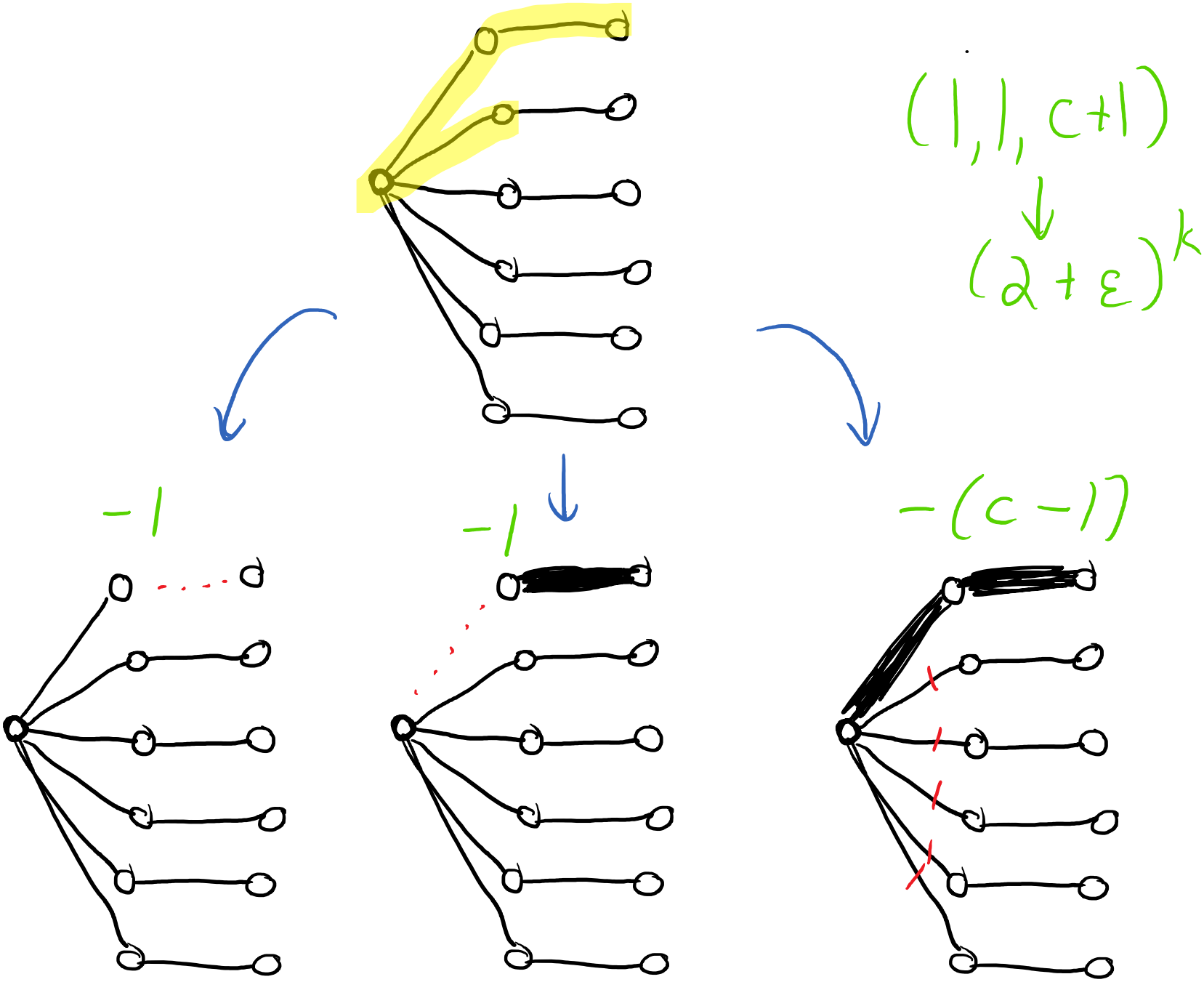


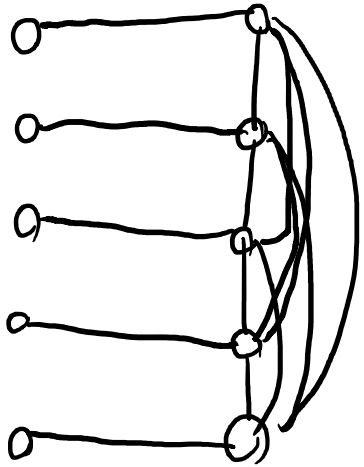


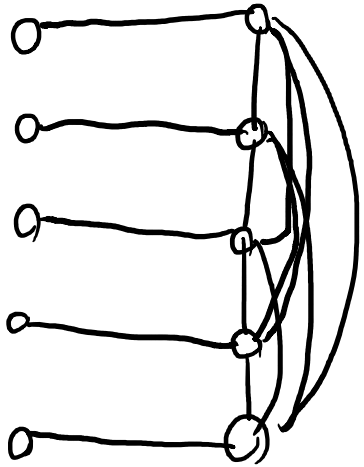


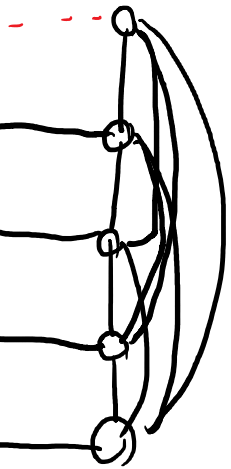


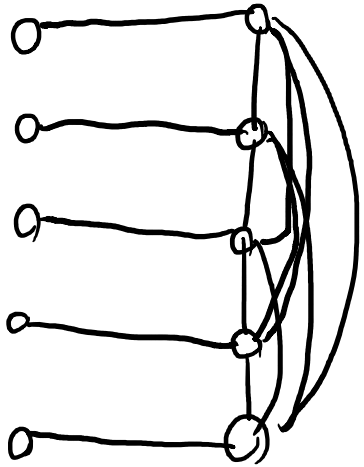


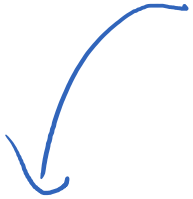

(1, 1, c+1)
(2 + ε)^k
-1
-1
-(c-1)

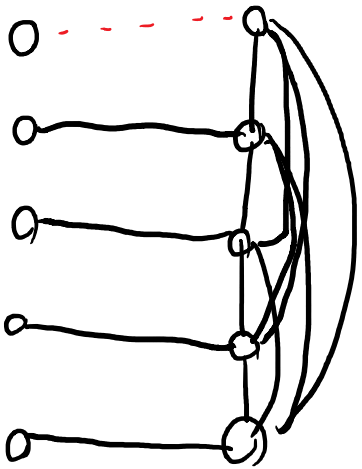


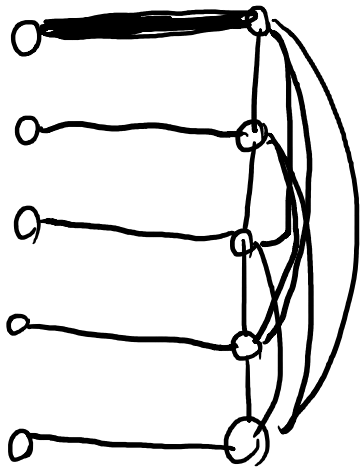


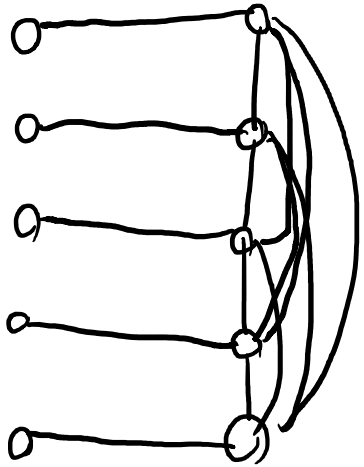


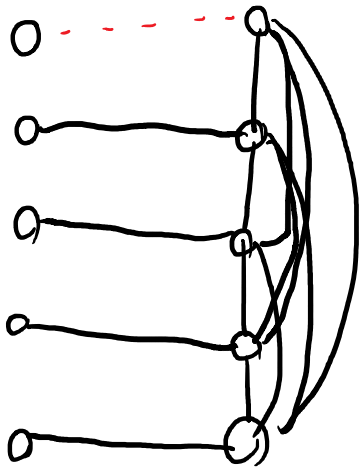


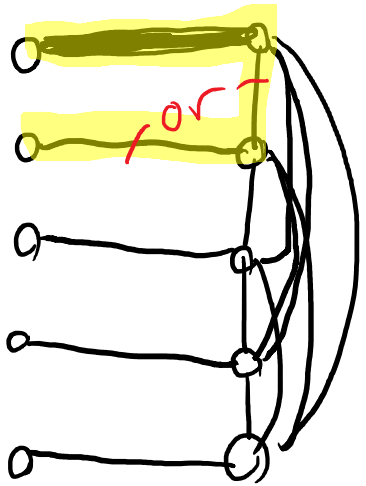


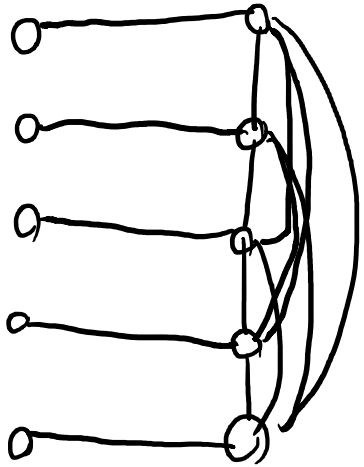


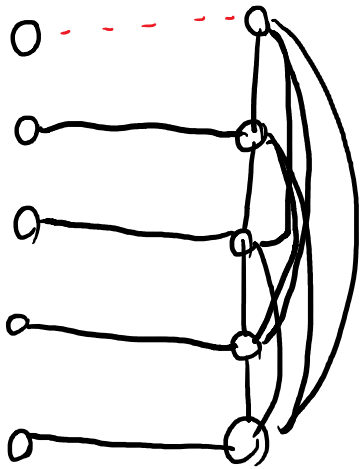


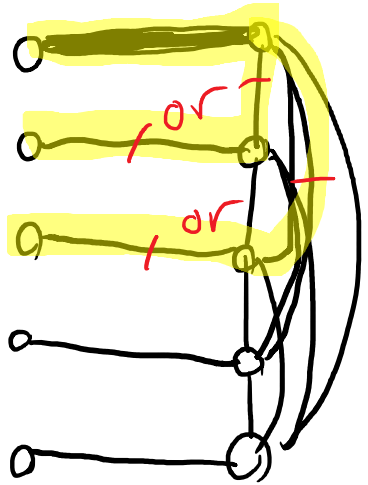


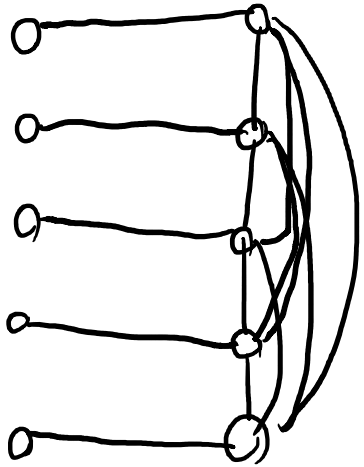


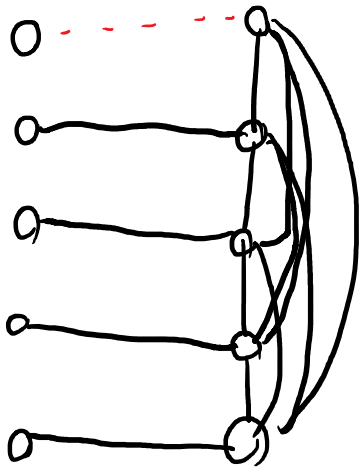

or

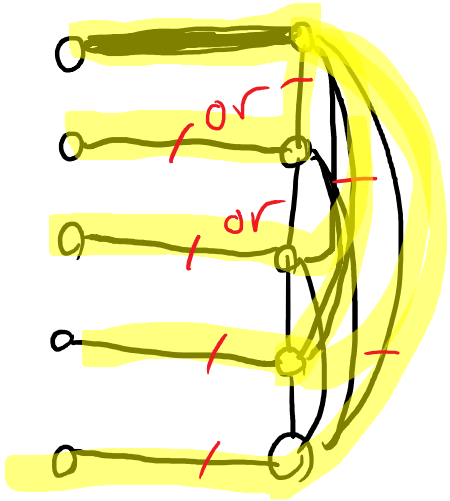


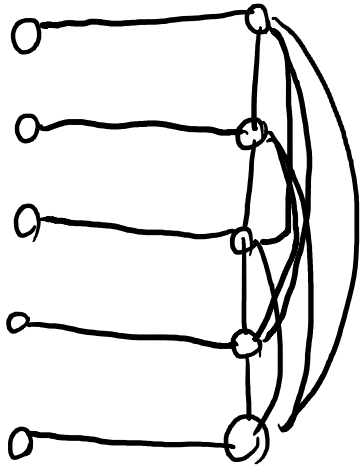



or
or



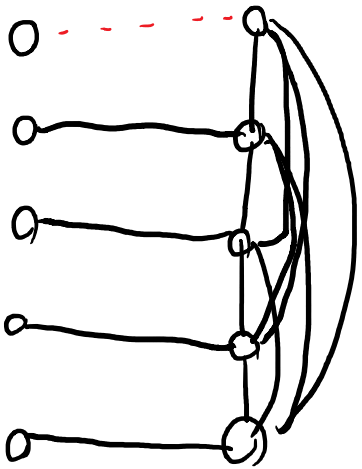


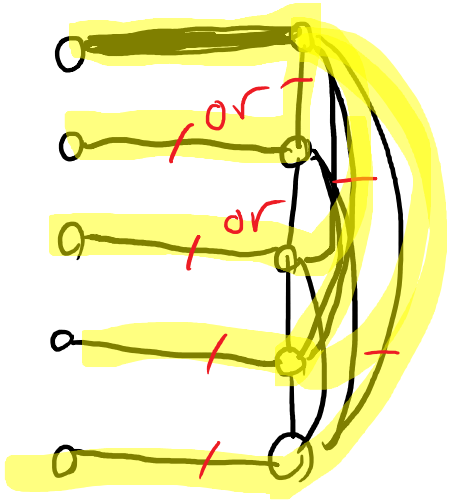

or
or

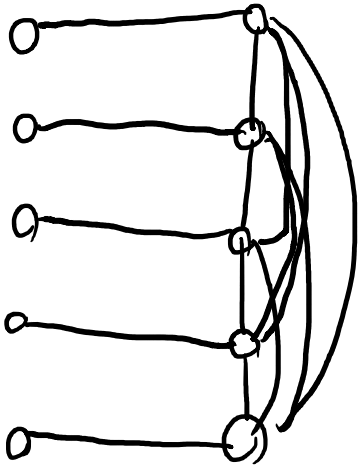


2^{c-1} combinations that each delete c-1 edges

-1

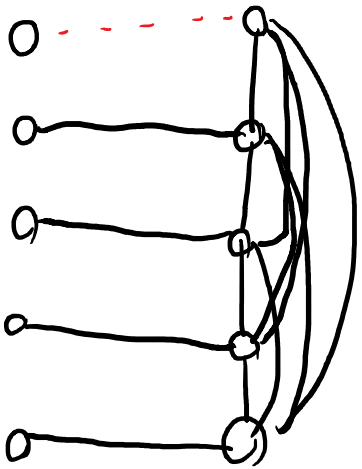


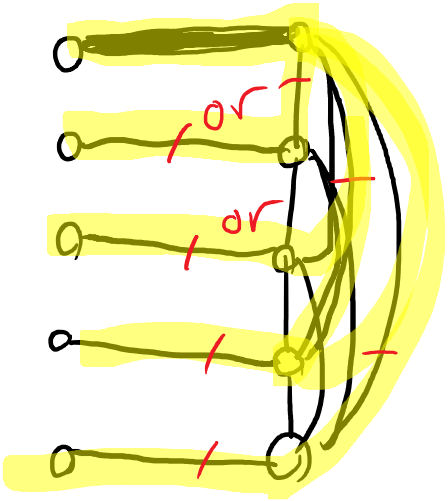




2^{c-1} combinations that each delete $c-1$ edges

-1





$(-1, -c+1, -c+1, \ldots, -c+1)$

$\downarrow$

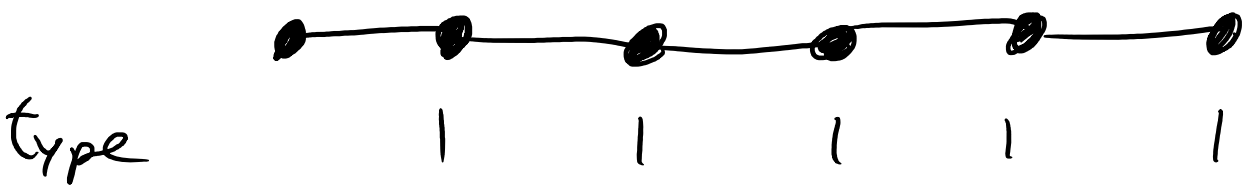
$(2+\varepsilon)^k$

▶ **Theorem 8** ([5]). *For every integer $c \geq 3$, there exists a constant C such that every prime graph with at least C vertices contains one of the following graphs or their complements as an induced subgraph.*

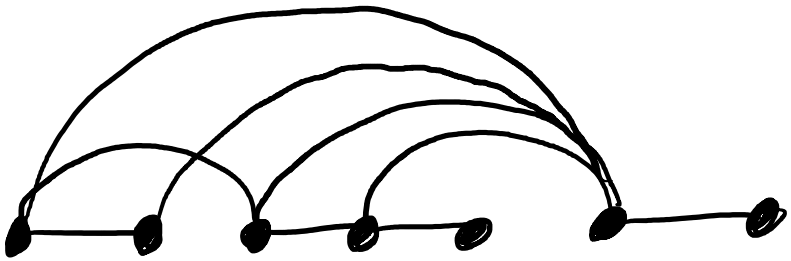
1. *The 1-subdivision of $K_{1,c}$ (denoted by $K_{1,c}^{(1)}$) (Figures 1 and 2).*
2. *The line graph of $K_{2,c}$, where $K_{2,c}$ is a complete bipartite graph with two vertices on one side and c on the other (Figures 3 and 4).*
3. *The thin spider with c legs, which has an independent set $\{v_1, \ldots, v_c\}$, a clique $\{w_1, \ldots, w_c\}$, and edges $v_i w_i$ for $i \in [c]$ (Figures 5 and 6). The complement of a thin spider is called a thick spider.*
4. *The half-graph of height c, denoted H_c, which has two independent sets $\{v_1, \ldots, v_c\}, \{w_1, \ldots, w_c\}$ and edge $v_i w_i$ iff $i \leq j$ (Figures 7 and 8).*
5. *The graph $H'_{c,I}$, obtained from H_c by making the w_i's a clique and adding a vertex u adjacent to all the v_i's (Figure 9). This graph is self-complementary.*
6. *The graph H^*_c, obtained from $H'_{c,I}$ by making u only adjacent to v_1 (Figures 10 and 11).*
7. *A chain with $c+1$ vertices that induces a prime graph.*

▶ **Definition 7.** *A* chain *is a set of vertices* $\{v_1, v_2, \ldots, v_c\}$ *in which, for* $i \in \{2, \ldots, c\}$, *the vertex* v_i *is of one of two types:*
(type 0) v_i *is adjacent to every vertex in* $v_1, \ldots, v_{i-2}$ *and is not adjacent to* v_{i-1}*; or*
(type 1) v_i *is adjacent to* v_{i-1} *and not adjacent to any of* $v_1, \ldots, v_{i-2}$.

▶ **Definition 7.** *A* chain *is a set of vertices* $\{v_1, v_2, \ldots, v_c\}$ *in which, for* $i \in \{2, \ldots, c\}$*, the vertex* v_i *is of one of two types:*
(type 0) v_i *is adjacent to every vertex in* $v_1, \ldots, v_{i-2}$ *and is not adjacent to* v_{i-1}*; or*
(type 1) v_i *is adjacent to* v_{i-1} *and not adjacent to any of* $v_1, \ldots, v_{i-2}$*.*



▶ **Definition 7.** *A* chain *is a set of vertices* $\{v_1, v_2, \ldots, v_c\}$ *in which, for* $i \in \{2, \ldots, c\}$, *the vertex* v_i *is of one of two types:*
(type 0) v_i *is adjacent to every vertex in* $v_1, \ldots, v_{i-2}$ *and is not adjacent to* v_{i-1}*; or*
(type 1) v_i *is adjacent to* v_{i-1} *and not adjacent to any of* $v_1, \ldots, v_{i-2}$.



type 1 0 1 1 0 1

Open questions

1. Is there a SETH lower bound of $O^*(2^k)$ for cograph-editing? Current reductions cannot be used.
2. Can the approach be used to improve the complexity of ***Cograph-editing,*** or other H-free editing problems?
3. Does cograph-editing have a $O(k^2)$ kernel?
   - [Crespelle, Pellerin & Thomassé, 2022] got $O(k^2 \log k)$.