Known algorithms for Edge Clique Cover are probably optimal

Marek Cygan[1], Marcin Pilipczuk[1], Michał Pilipczuk[2]

[1] Institute of Informatics, University of Warsaw, Poland

[2] Department of Informatics, University of Bergen, Norway

SODA 2013, January 7th

Problem definition

Edge Clique Cover

- **Input:** Graph G, integer k

Problem definition

Edge Clique Cover

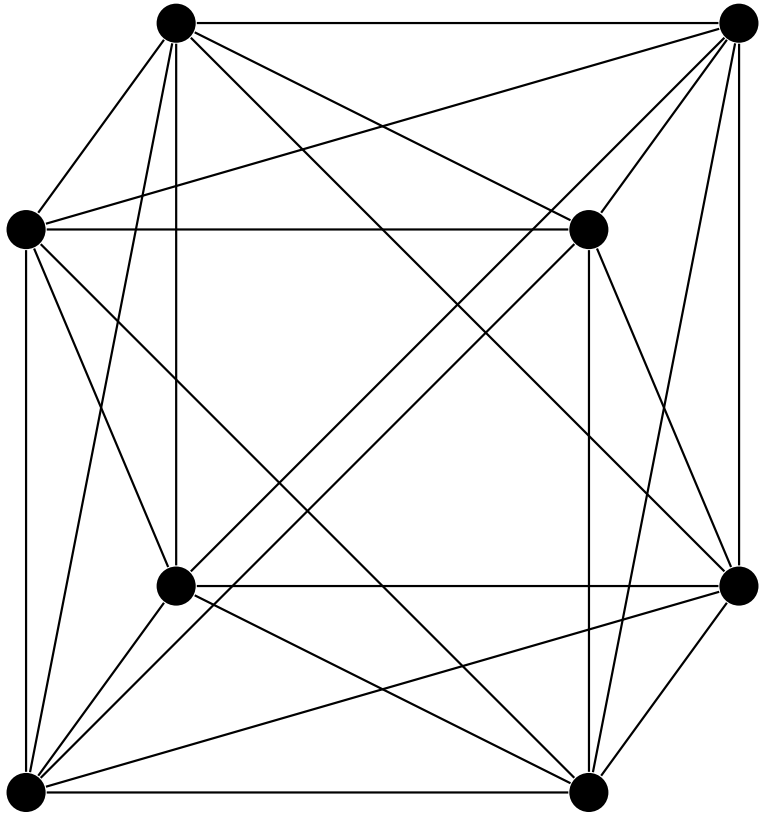
- **Input:** Graph G, integer k
- **Parameter:** k

Problem definition

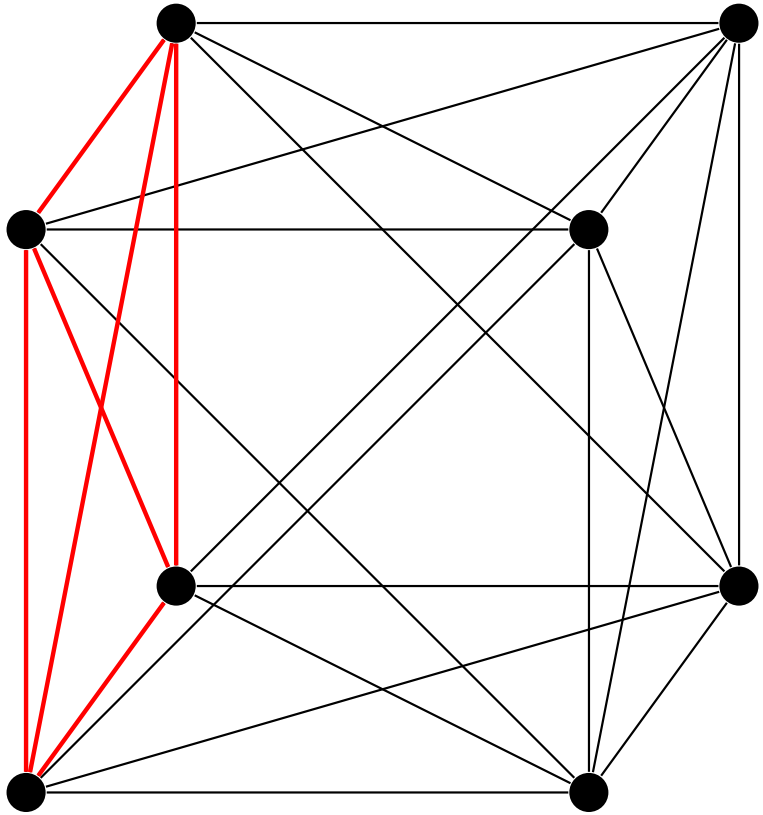
Edge Clique Cover

- **Input:** Graph G, integer k
- **Parameter:** k
- **Question:** Does there exists a collection of k cliques $C_1, C_2, \ldots, C_k$ in G, such that $E(G) = \bigcup_{i=1}^{k} E(C_i)$?

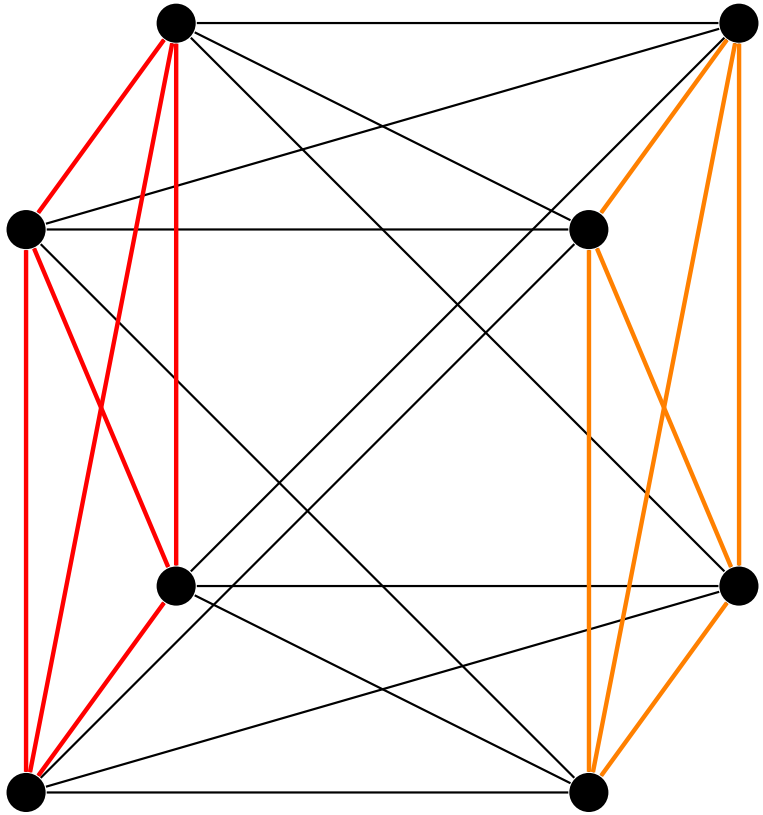
Example



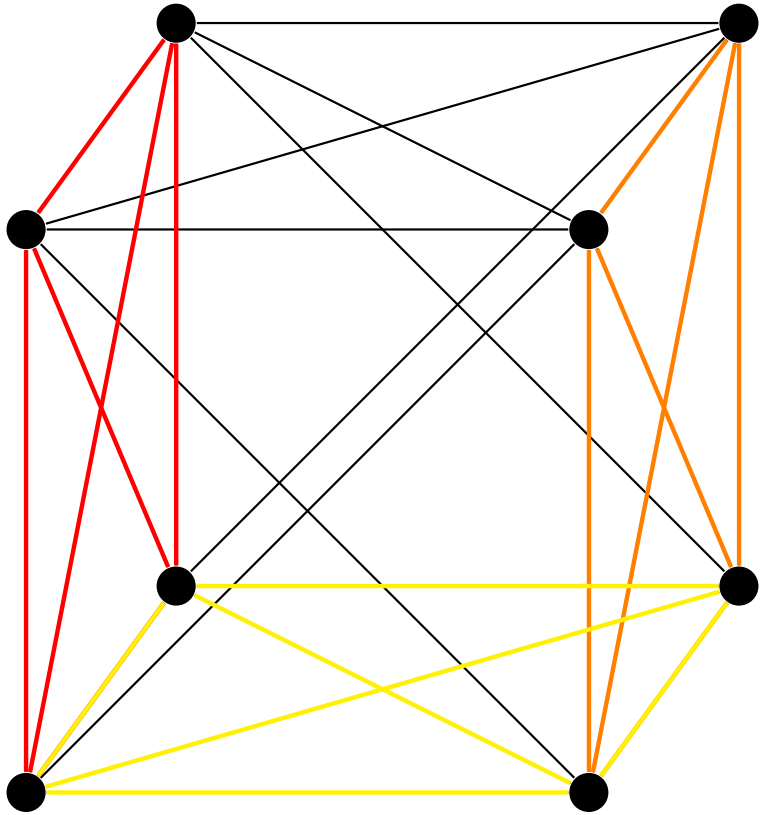
Example



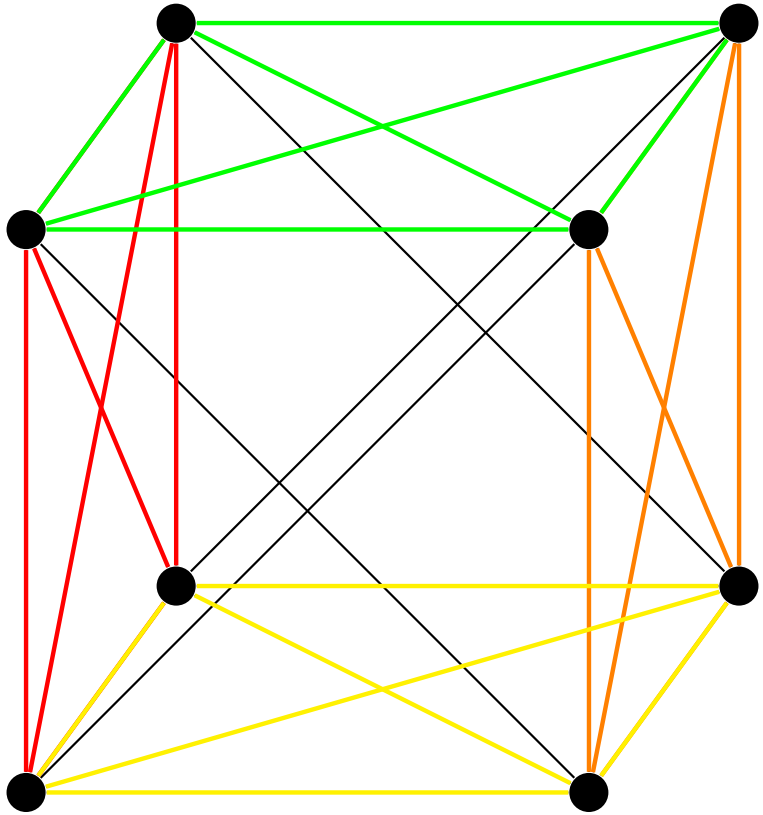
Example



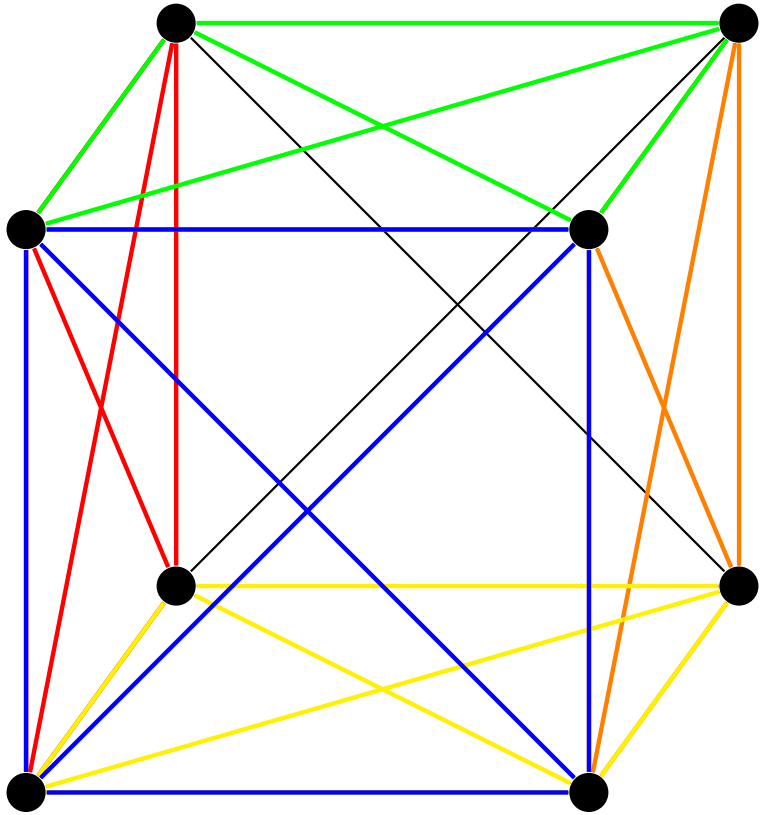
Example



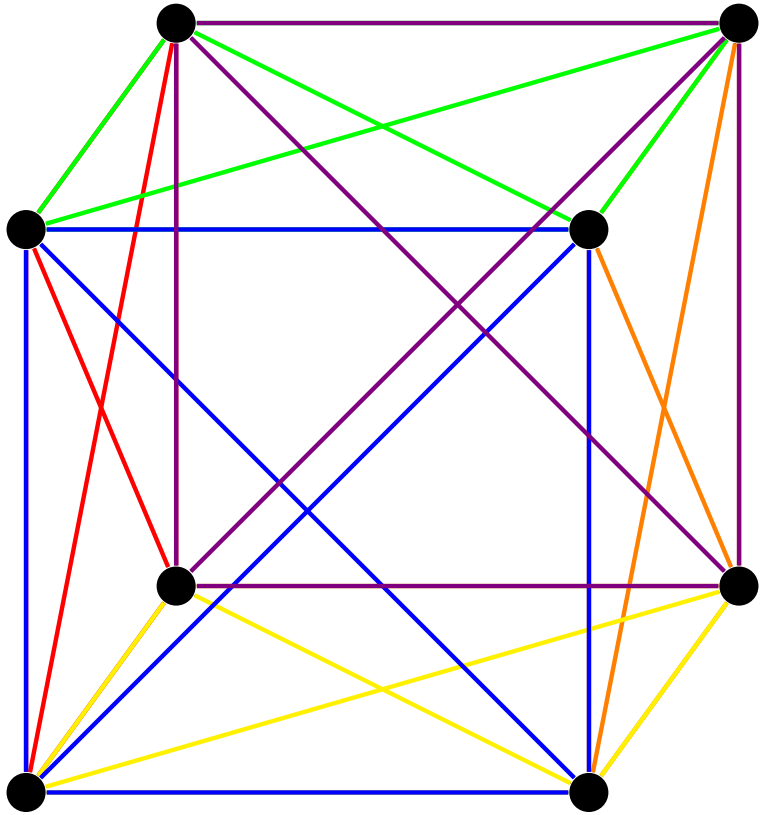
Example



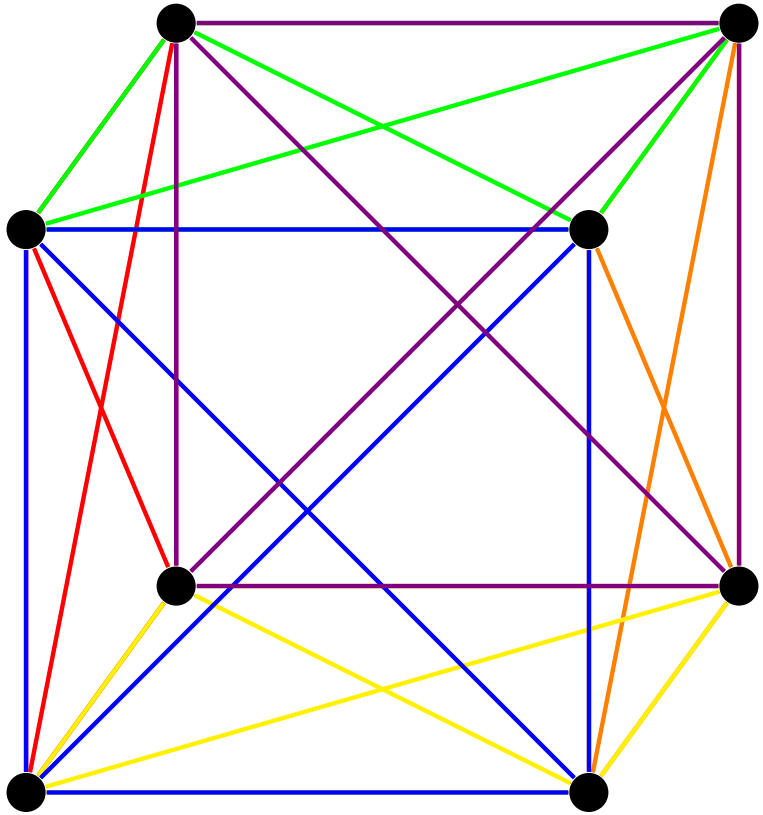
Example



Example

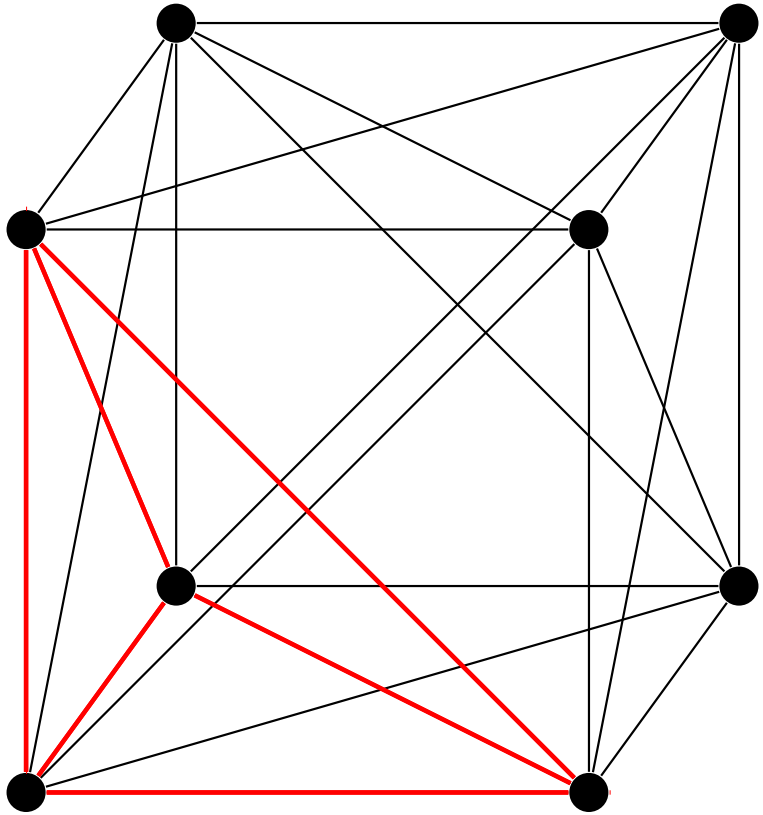


Example

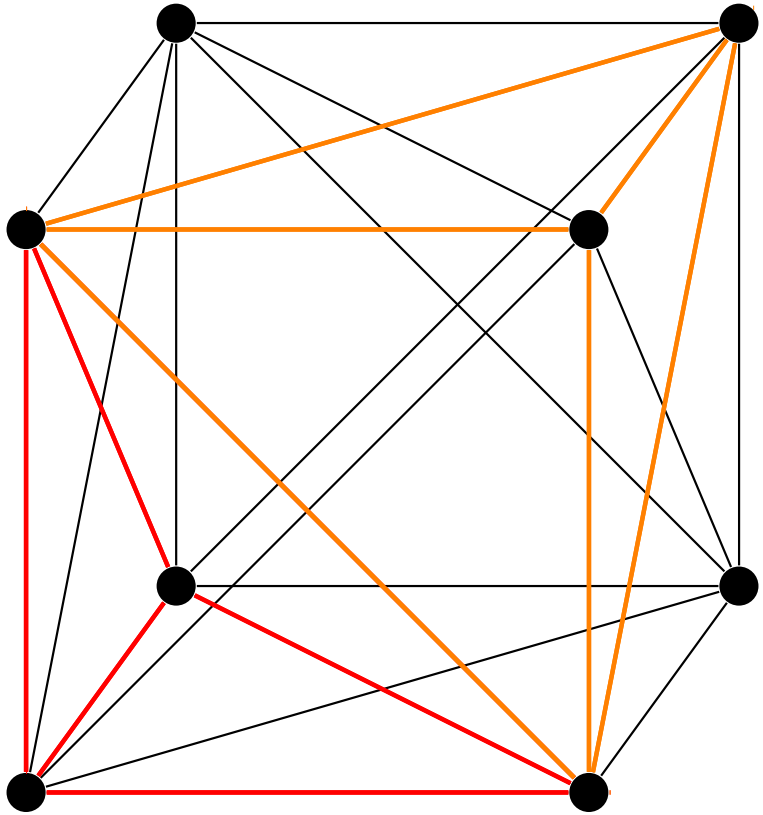


6 cliques: optimal?

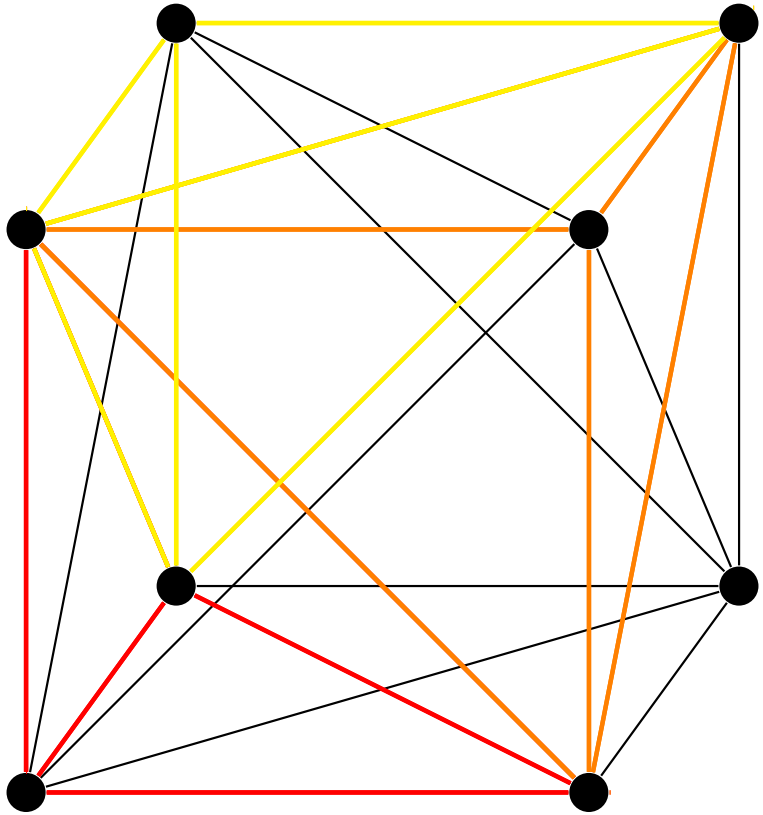
Example



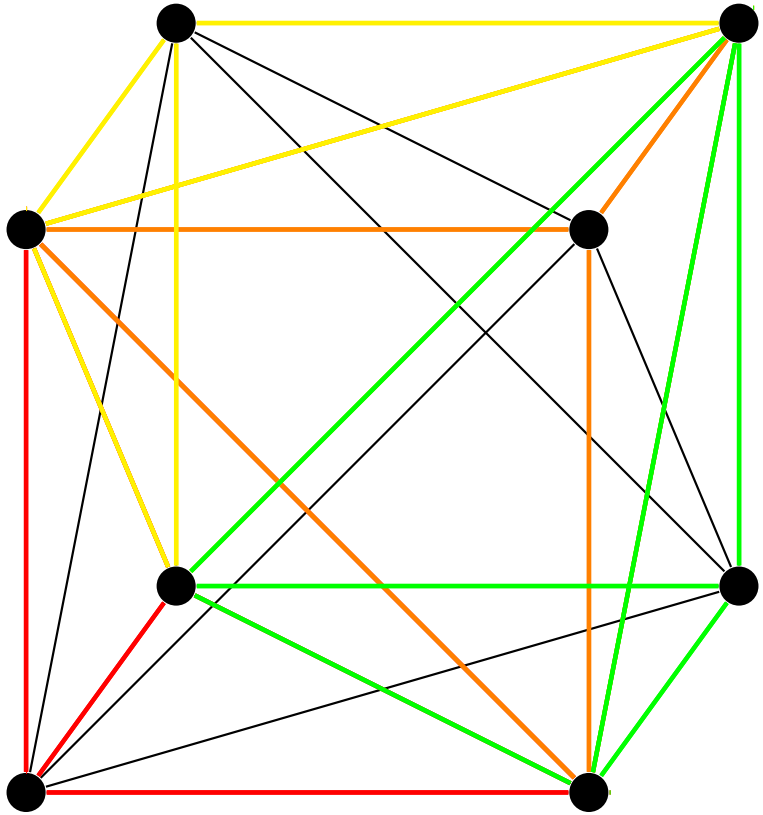
Example



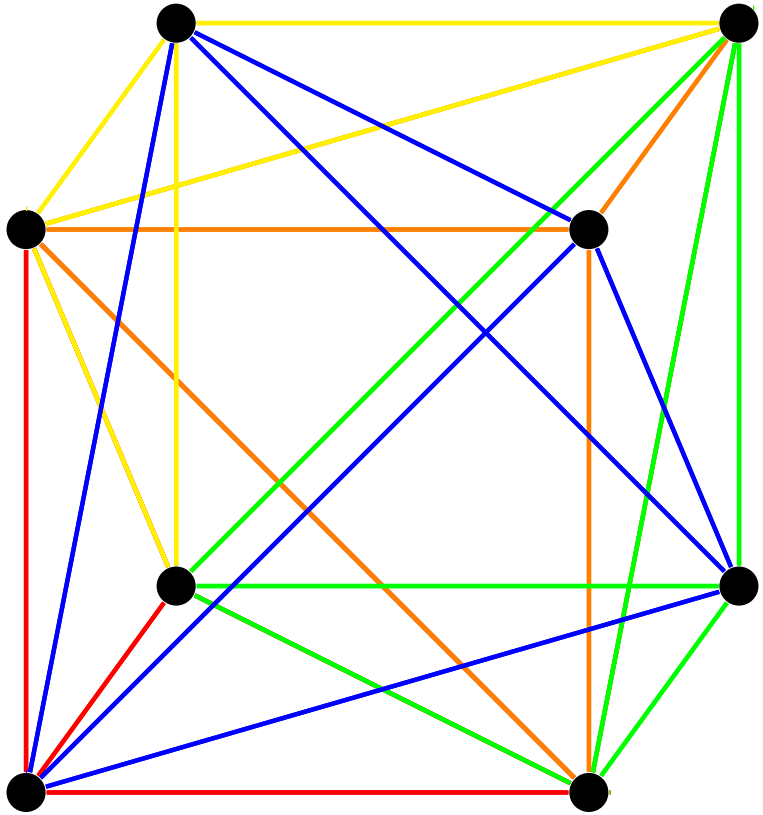
Example



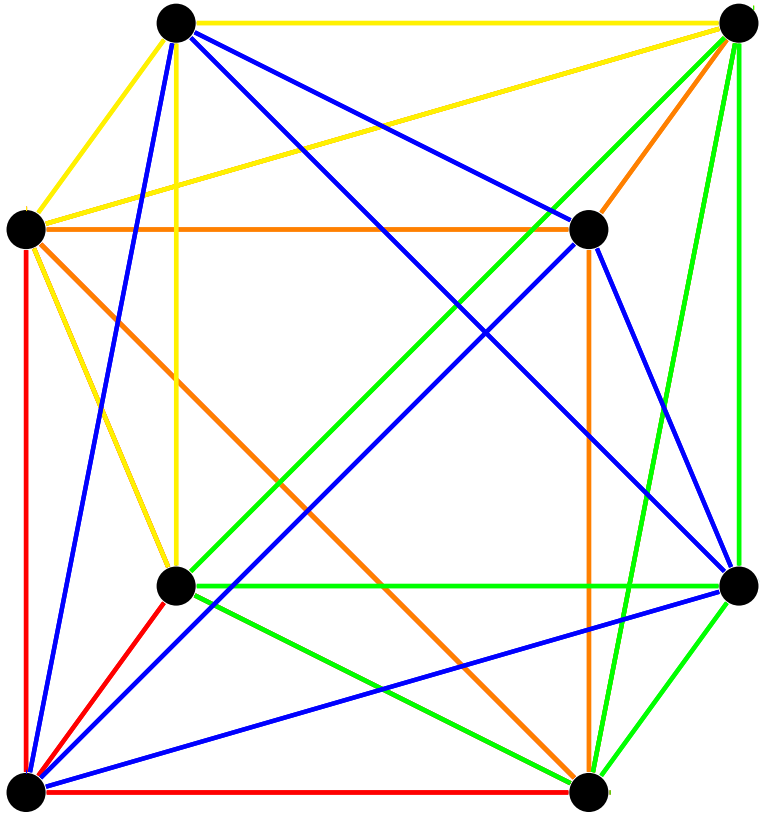
Example



Example



Example



Magical solution with 5 cliques

History of the problem

- Known under names: Covering by Cliques, Intersection Graph Basis, Keyword Conflict.

History of the problem

- Known under names: Covering by Cliques, Intersection Graph Basis, Keyword Conflict.
- Edge Clique Cover is equivalent to checking whether a graph can be represented by an intersection model with a universe of at most k elements (Erdős, 1966).

2^k kernel (Gramm et al.)

- **Rule 1.** Delete isolated vertices.

2^k kernel (Gramm et al.)

- **Rule 1.** Delete isolated vertices.
- **Rule 2.** If there is an isolated K_2, delete it and decrease k by 1.

2^k kernel (Gramm et al.)

- **Rule 1.** Delete isolated vertices.
- **Rule 2.** If there is an isolated K_2, delete it and decrease k by 1.
- **Rule 3.** If there are true twins, that is, vertices v, w with $N[v] = N[w]$, then delete one of them.

2^k kernel (Gramm et al.)

- **Rule 1.** Delete isolated vertices.
- **Rule 2.** If there is an isolated K_2, delete it and decrease k by 1.
- **Rule 3.** If there are true twins, that is, vertices v, w with $N[v] = N[w]$, then delete one of them.
  - *Proof:* Solution after deletion can be extended by adding the deleted twin to all the cliques containing the undeleted one.

2^k kernel (Gramm et al.)

- **Rule 1.** Delete isolated vertices.
- **Rule 2.** If there is an isolated K_2, delete it and decrease k by 1.
- **Rule 3.** If there are true twins, that is, vertices v, w with $N[v] = N[w]$, then delete one of them.
  - *Proof:* Solution after deletion can be extended by adding the deleted twin to all the cliques containing the undeleted one.
- **Rule 4.** If there are more than 2^k vertices, answer NO.

2^k kernel (Gramm et al.)

- **Rule 1.** Delete isolated vertices.
- **Rule 2.** If there is an isolated K_2, delete it and decrease k by 1.
- **Rule 3.** If there are true twins, that is, vertices v, w with $N[v] = N[w]$, then delete one of them.
  - *Proof:* Solution after deletion can be extended by adding the deleted twin to all the cliques containing the undeleted one.
- **Rule 4.** If there are more than 2^k vertices, answer NO.
  - *Proof:* In this situation two vertices must be in the same set of cliques, which means that they must be true twins.

2^k kernel (Gramm et al.)

- **Rule 1.** Delete isolated vertices.
- **Rule 2.** If there is an isolated K_2, delete it and decrease k by 1.
- **Rule 3.** If there are true twins, that is, vertices v, w with $N[v] = N[w]$, then delete one of them.
  - *Proof:* Solution after deletion can be extended by adding the deleted twin to all the cliques containing the undeleted one.
- **Rule 4.** If there are more than 2^k vertices, answer NO.
  - *Proof:* In this situation two vertices must be in the same set of cliques, which means that they must be true twins.
- If no rule can be applied, we have an instance with at most 2^k vertices (kernel).

Classical results

- 2^k kernel.

Classical results

- 2^k kernel.
- DP on subsets of edges in the kernel: $\mathcal{O}^\star(2^{2^{\mathcal{O}(k)}})$ FPT algorithm.

Classical results

- 2^k kernel.
- DP on subsets of edges in the kernel: $\mathcal{O}^\star(2^{2^{\mathcal{O}(k)}})$ FPT algorithm.
- Nothing better was known.

Classical results

- 2^k kernel.
- DP on subsets of edges in the kernel: $\mathcal{O}^\star(2^{2^{\mathcal{O}(k)}})$ FPT algorithm.
- Nothing better was known.
- Embarrasing...

Recent results

- Cygan, Kratsch, Pilipczuk, P, Wahlström, *Edge Clique Cover and graph separation: new incompressibility results*, ICALP 2012

Recent results

- Cygan, Kratsch, Pilipczuk, P, Wahlström, *Edge Clique Cover and graph separation: new incompressibility results*, ICALP 2012
- **First**: simple AND-composition

Recent results

- Cygan, Kratsch, Pilipczuk, P, Wahlström, *Edge Clique Cover and graph separation: new incompressibility results*, ICALP 2012
- **First**: simple AND-composition
  - Now we know that it gives no-poly-kernel unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$ (by results of Andrew Drucker, FOCS 2012).

Recent results

- Cygan, Kratsch, Pilipczuk, P, Wahlström, *Edge Clique Cover and graph separation: new incompressibility results*, ICALP 2012
- **First**: simple AND-composition
  - Now we know that it gives no-poly-kernel unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$ (by results of Andrew Drucker, FOCS 2012).
- **Second:** more involved OR-composition

Recent results

- Cygan, Kratsch, Pilipczuk, P, Wahlström, *Edge Clique Cover and graph separation: new incompressibility results*, ICALP 2012
- **First**: simple AND-composition
  - Now we know that it gives no-poly-kernel unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$ (by results of Andrew Drucker, FOCS 2012).
- **Second:** more involved OR-composition
  - Was known to give no-poly-kernel unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$ (Fortnow and Santhanam, STOC 2008; Bodlaender et al., ICALP 2008).

Recent results

- Cygan, Kratsch, Pilipczuk, P, Wahlström, *Edge Clique Cover and graph separation: new incompressibility results*, ICALP 2012
- **First**: simple AND-composition
  - Now we know that it gives no-poly-kernel unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$ (by results of Andrew Drucker, FOCS 2012).
- **Second:** more involved OR-composition
  - Was known to give no-poly-kernel unless $\mathrm{NP} \subseteq \mathrm{coNP/poly}$ (Fortnow and Santhanam, STOC 2008; Bodlaender et al., ICALP 2008).
  - Nontrivial tweak of the AND-composition.

New results

- $\mathcal{O}^\star(2^{2^{o(k)}})$ time complexity lower bound under Exponential Time Hypothesis (ETH).

New results

- $\mathcal{O}^\star(2^{2^{o(k)}})$ time complexity lower bound under Exponential Time Hypothesis (ETH).
- Reduction in fact proves that there is no $2^{o(k)}$ kernel, unless ETH fails **and** $\mathrm{NP} \subseteq \mathrm{coNP/poly}$.

New results

- $\mathcal{O}^\star(2^{2^{o(k)}})$ time complexity lower bound under Exponential Time Hypothesis (ETH).
- Reduction in fact proves that there is no $2^{o(k)}$ kernel, unless ETH fails **and** $\mathrm{NP} \subseteq \mathrm{coNP/poly}$.
- First natural FPT problem for which doubly-exponential complexity is optimal.

New results

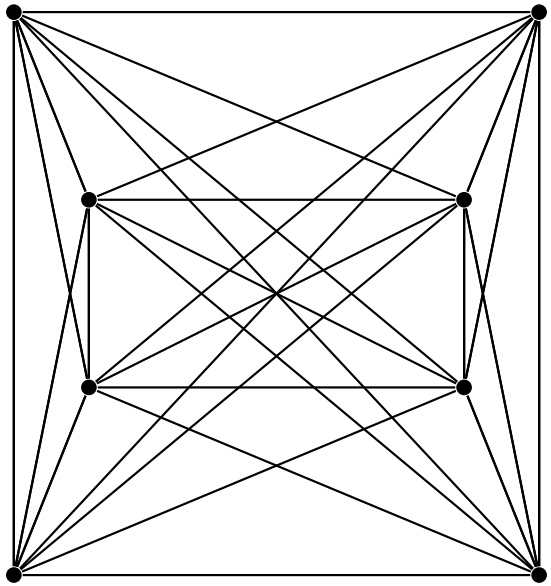
- $\mathcal{O}^\star(2^{2^{o(k)}})$ time complexity lower bound under Exponential Time Hypothesis (ETH).
- Reduction in fact proves that there is no $2^{o(k)}$ kernel, unless ETH fails **and** $\mathrm{NP} \subseteq \mathrm{coNP/poly}$.
- First natural FPT problem for which doubly-exponential complexity is optimal.
- First result excluding subexponential kernels, instead of just polynomial.

New results

- $\mathcal{O}^\star(2^{2^{o(k)}})$ time complexity lower bound under Exponential Time Hypothesis (ETH).
- Reduction in fact proves that there is no $2^{o(k)}$ kernel, unless ETH fails **and** $\mathrm{NP} \subseteq \mathrm{coNP/poly}$.
- First natural FPT problem for which doubly-exponential complexity is optimal.
- First result excluding subexponential kernels, instead of just polynomial.
- **Plan for now**: Sketch of the ETH lower bound.

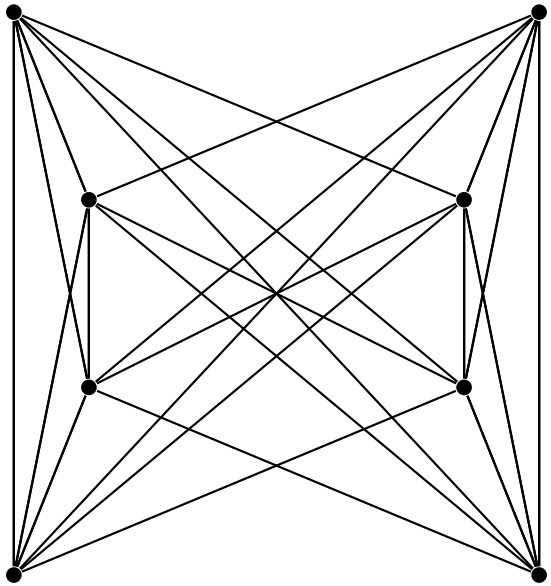
Hero of the day

Hero of the day



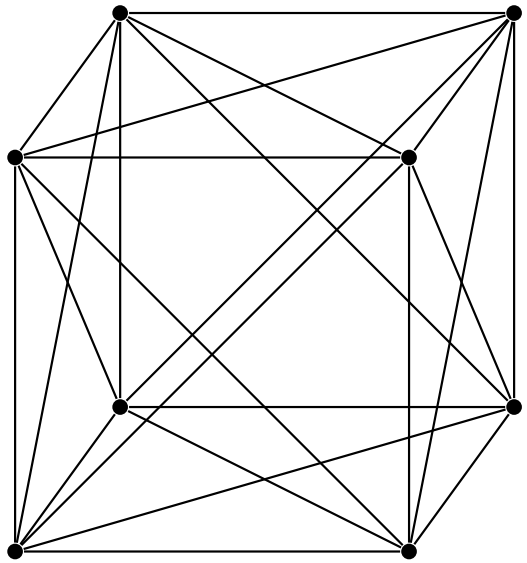
H_{2n}: A $2n$-clique

Hero of the day



H_{2n}: A $2n$-clique without a matching.

Hero of the day



H_{2n}: A $2n$-clique without a matching.

Covering $H_{2\ell+1}$

- No true twins: $\ell + 1$ lower bound.

Covering $H_{2\ell+1}$

- No true twins: $\ell + 1$ lower bound.
- $2\ell + 2$ upper bound:

Covering $H_{2\ell+1}$

- No true twins: $\ell + 1$ lower bound.
- $2\ell + 2$ upper bound:
  - Introduce two cliques: whole left part and whole right part.

Covering $H_{2\ell+1}$

- No true twins: $\ell + 1$ lower bound.
- $2\ell + 2$ upper bound:
  - Introduce two cliques: whole left part and whole right part.
  - For every binary position $i = 1, 2, \ldots, \ell$ introduce two cliques:

Covering $H_{2\ell+1}$

- No true twins: $\ell + 1$ lower bound.
- $2\ell + 2$ upper bound:
  - Introduce two cliques: whole left part and whole right part.
  - For every binary position $i = 1, 2, \ldots, \ell$ introduce two cliques:
    - one containing left vertices on indices with 0-s on the i-th position and right on indices with 1-s;

Covering $H_{2\ell+1}$

- No true twins: $\ell + 1$ lower bound.
- $2\ell + 2$ upper bound:
  - Introduce two cliques: whole left part and whole right part.
  - For every binary position $i = 1, 2, \ldots, \ell$ introduce two cliques:
    - one containing left vertices on indices with 0-s on the i-th position and right on indices with 1-s;
    - one containing left vertex on indices with 1-s on the i-th position and right on indices with 0-s.

Covering $H_{2\ell+1}$

- No true twins: $\ell + 1$ lower bound.
- $2\ell + 2$ upper bound:
  - Introduce two cliques: whole left part and whole right part.
  - For every binary position $i = 1, 2, \ldots, \ell$ introduce two cliques:
    - one containing left vertices on indices with 0-s on the i-th position and right on indices with 1-s;
    - one containing left vertex on indices with 1-s on the i-th position and right on indices with 0-s.
- For every two distinct indices, the cross between left and right vertices is covered by the cliques for the position, on which the indices differ.

Covering $H_{2\ell+1}$

- What is the optimal value for $H_{2\ell+1}$?
  Somewhere between $\ell+1$ and $2\ell+2$...

Covering $H_{2^{\ell+1}}$

- What is the optimal value for $H_{2^{\ell+1}}$?
  Somewhere between $\ell + 1$ and $2\ell + 2$...
- Not so obvious: H_{2^3} has a nontrivial solution with 5 cliques.

Covering $H_{2^{\ell+1}}$

- What is the optimal value for $H_{2^{\ell+1}}$?
  Somewhere between $\ell + 1$ and $2\ell + 2$...
- Not so obvious: H_{2^3} has a nontrivial solution with 5 cliques.
- At first we posted this as an open question.

Covering $H_{2\ell+1}$

- What is the optimal value for $H_{2\ell+1}$?
  Somewhere between $\ell+1$ and $2\ell+2$...
- Not so obvious: H_{2^3} has a nontrivial solution with 5 cliques.
- At first we posted this as an open question.
- But it was already posted by Orlin in 1977 and resolved by Gregory and Pullman in 1982.

Covering $H_{2\ell+1}$

- What is the optimal value for $H_{2\ell+1}$?
  Somewhere between $\ell+1$ and $2\ell+2$...
- Not so obvious: H_{2^3} has a nontrivial solution with 5 cliques.
- At first we posted this as an open question.
- But it was already posted by Orlin in 1977 and resolved by Gregory and Pullman in 1982.
- Note that an $\mathcal{O}^\star(2^{\mathcal{O}(k)})$ FPT algorithm for ECC would need to resolve this question in polynomial time.

Covering $H_{2\ell+1}$

- What is the optimal value for $H_{2\ell+1}$?
  Somewhere between $\ell+1$ and $2\ell+2$...
- Not so obvious: H_{2^3} has a nontrivial solution with 5 cliques.
- At first we posted this as an open question.
- But it was already posted by Orlin in 1977 and resolved by Gregory and Pullman in 1982.
- Note that an $\mathcal{O}^\star(2^{\mathcal{O}(k)})$ FPT algorithm for ECC would need to resolve this question in polynomial time.
- **Idea:** Use $H_{2\ell+1}$ as the backbone of the reduction.

Starting the reduction

- Unless ETH fails, there is no $2^{o(n+m)}$ algorithm for 3-CNF-SAT.

Starting the reduction

- Unless ETH fails, there is no $2^{o(n+m)}$ algorithm for 3-CNF-SAT.
- We prove the following theorem:

Starting the reduction

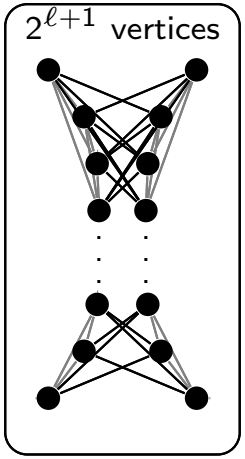
- Unless ETH fails, there is no $2^{o(n+m)}$ algorithm for 3-CNF-SAT.
- We prove the following theorem:

Reduction

There exists a polynomial-time reduction that, given a 3-CNF-SAT instance φ with n variables and m clauses, outputs an equivalent instance (G, k) of ECC with $|V(G)| = O(n+m)$ and $k = \theta(\log n)$.

A crude sketch of the reduction

$n = 2^\ell$

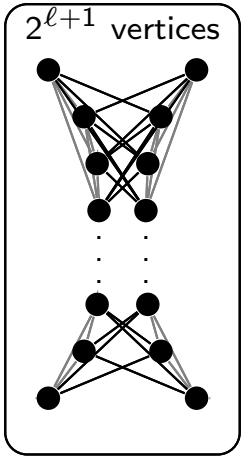


A crude sketch of the reduction

$n = 2^\ell$

---

magically covered

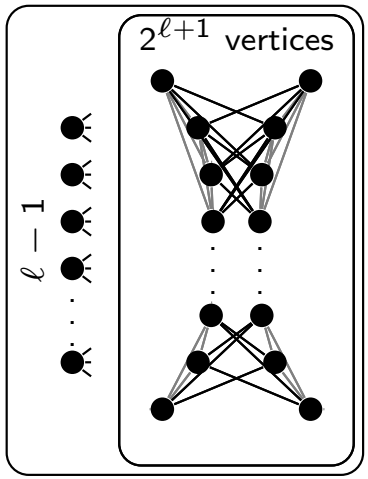


A crude sketch of the reduction

$n = 2^\ell$

---

magically covered

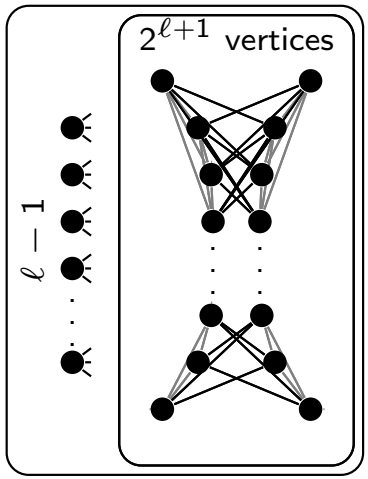


A crude sketch of the reduction

$n = 2^\ell$

---

magically covered



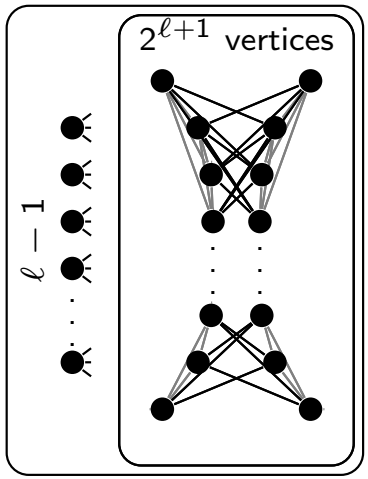
Twin cover:
every clique comes
with its complement

A crude sketch of the reduction

$n = 2^\ell$

---

magically covered



Twin cover:
every clique comes with its complement

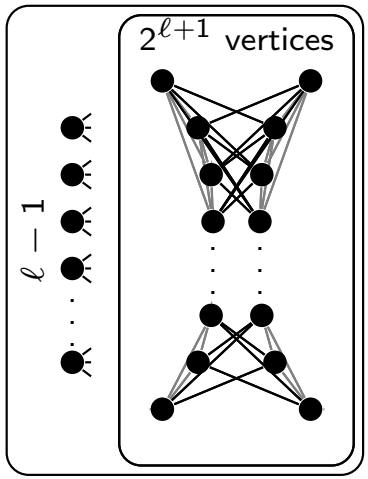
Fact:
$2(\ell + 1)$ is optimal for twin cover

A crude sketch of the reduction

$n = 2^\ell$

---

magically covered



Twin cover:

every clique comes with its complement

Fact:

$2(\ell + 1)$ is optimal for twin cover

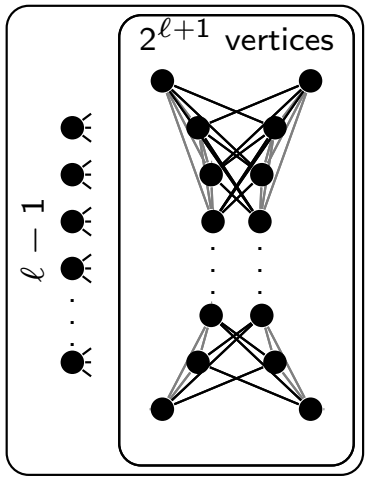
Moreover:

ℓ twins can be extended only by a twin

A crude sketch of the reduction

$n = 2^\ell$

magically covered



Clause checkers

Twin cover:
every clique comes with its complement

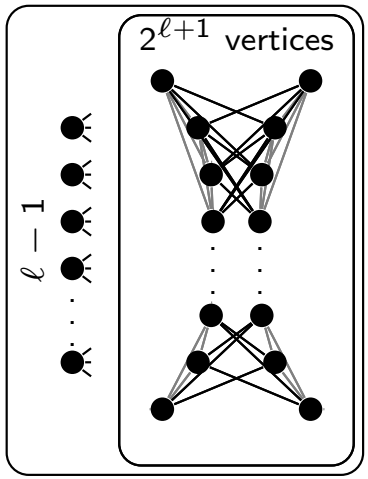
Fact:
$2(\ell + 1)$ is optimal for twin cover

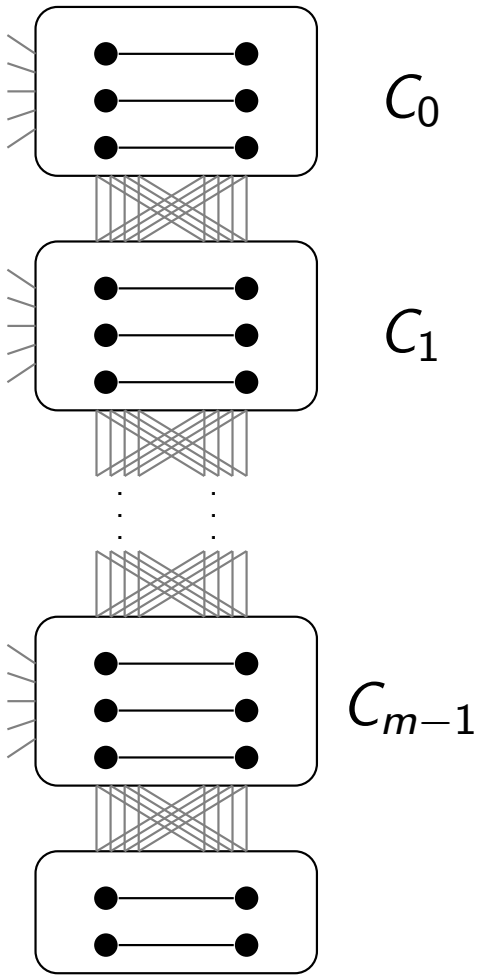
Moreover:
ℓ twins can be extended only by a twin

A crude sketch of the reduction

$n = 2^\ell$

magically covered





Twin cover:
every clique comes with its complement

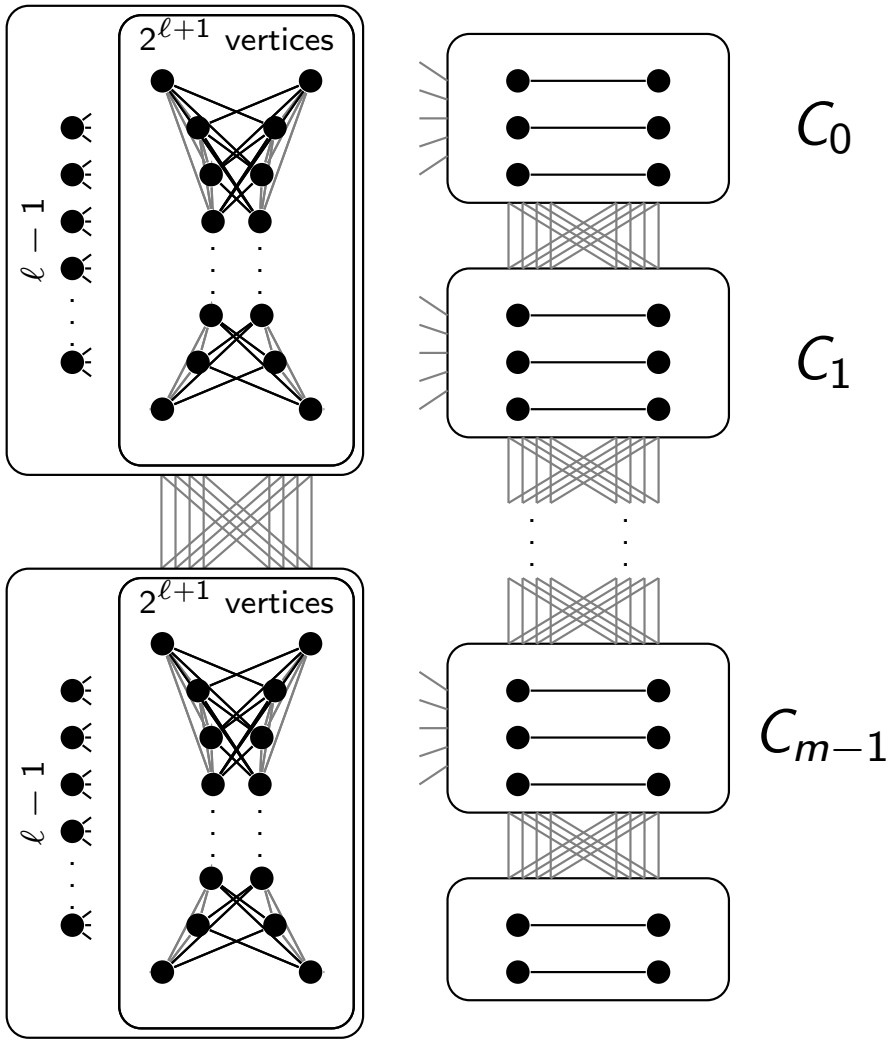
Fact:
$2(\ell + 1)$ is optimal for twin cover

Moreover:
ℓ twins can be extended only by a twin

A crude sketch of the reduction

$n = 2^\ell$

magically covered



Twin cover:
every clique comes with its complement

Fact:
$2(\ell + 1)$ is optimal for twin cover

Moreover:
ℓ twins can be extended only by a twin

Covering gray edges

- We need to gadget covering of the gray edges.

Covering gray edges

- We need to gadget covering of the gray edges.
- Consider a simplicial vertex.

Covering gray edges

- We need to gadget covering of the gray edges.
- Consider a simplicial vertex.
- One can greedily take its closed neighbourhood.

Covering gray edges

- We need to gadget covering of the gray edges.
- Consider a simplicial vertex.
- One can greedily take its closed neighbourhood.
- At cost of 1 in budget we may introduce a clique of gray edges.

Covering gray edges

- We need to gadget covering of the gray edges.
- Consider a simplicial vertex.
- One can greedily take its closed neighbourhood.
- At cost of 1 in budget we may introduce a clique of gray edges.
- In our reduction we need $46 + 36\lceil \log m \rceil + 24\ell = O(\log n)$ simplicial vertices.

Covering gray edges

- We need to gadget covering of the gray edges.
- Consider a simplicial vertex.
- One can greedily take its closed neighbourhood.
- At cost of 1 in budget we may introduce a clique of gray edges.
- In our reduction we need $46 + 36\lceil \log m \rceil + 24\ell = O(\log n)$ simplicial vertices.
  - Technical construction on binary encodings.

Conclusion

- We show that simple doubly-exponential FPT and exponential kernel is essentially optimal for ECC.

Conclusion

- We show that simple doubly-exponential FPT and exponential kernel is essentially optimal for ECC.
- A natural problem with very high lower bound.

Conclusion

- We show that simple doubly-exponential FPT and exponential kernel is essentially optimal for ECC.
- A natural problem with very high lower bound.
- **Reason:** instances with logarithmic parameter are already nontrivial.

Conclusion

- We show that simple doubly-exponential FPT and exponential kernel is essentially optimal for ECC.
- A natural problem with very high lower bound.
- **Reason:** instances with logarithmic parameter are already nontrivial.
- Do there exist more such problems?

Thank you for your attention

Questions?