Fine-Grained Meta-Theorems for Vertex Integrity

Michael Lampis
LAMSADE

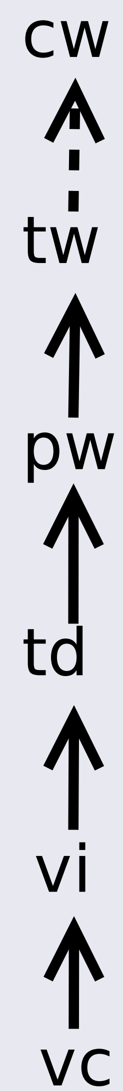
Valia Mitsou
IRIF

Oct 14th 2021

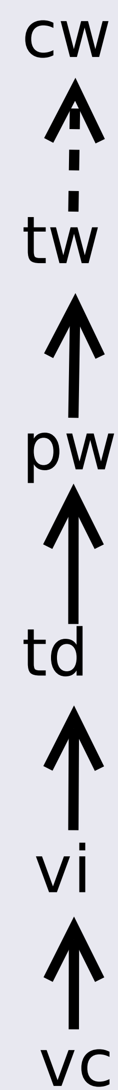
Vertex Integrity

A Map of Parameters

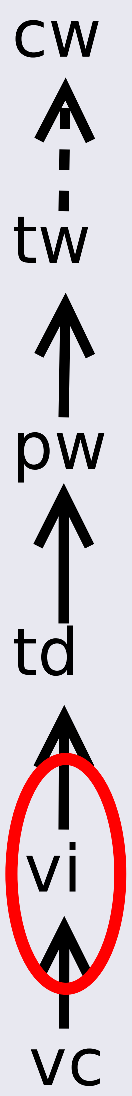
- Graph Structure Parameters:
  - k measures how “easy” a graph is
- Many ways to measure this.
- Algorithmically important:
  - A problem can be FPT (solvable in $f(k)n^{O(1)}$) or not.
  - The function $f(k)$ may be different.



- **Price of Generality**
  - Sometimes two parameters have a clear inclusion relation.
  - Algorithmically, this means one is more general, the other “easier”.
  - Want to understand algorithmic cost of generality.

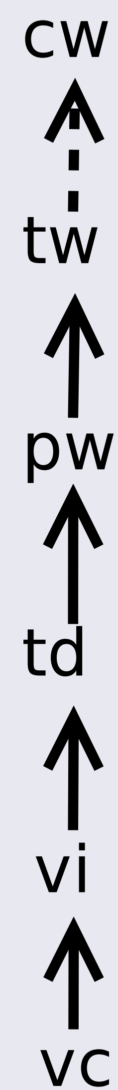


- **This talk**: Vertex Integrity
  - Want to understand relations between:
    1. Tree-depth
    2. Vertex Integrity
    3. Vertex Cover
  - How does complexity increase as we climb up?

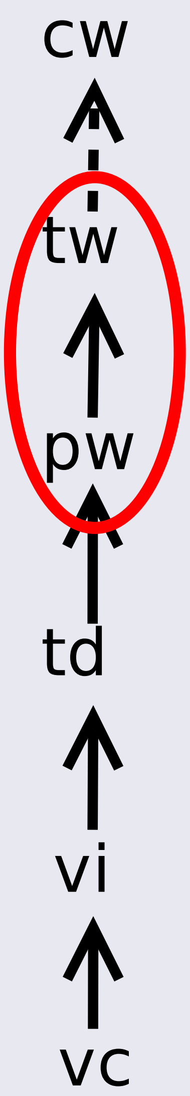


Parameters Review

- Graph Structure Parameters:
  - Clique-width
  - Treewidth
  - Pathwidth
  - Tree-depth
  - Vertex Integrity
  - Vertex Cover
- Arrows indicate Generalization
  - If G has pathwidth k, it has treewidth $\leq k$.
  - (Relation trickier for clique-width/treewidth).
- Algorithms propagate **down**.
- Hardness propagates **up**.



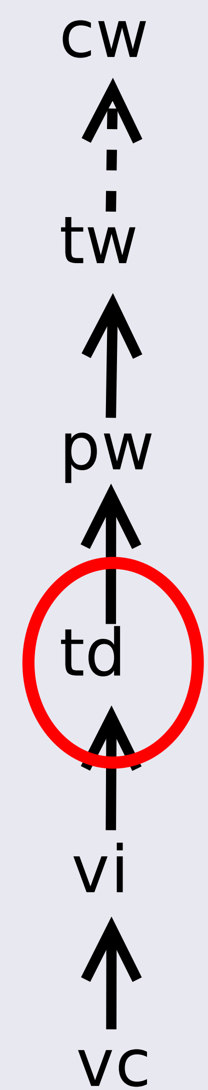
- Treewidth measures “tree-likeness”
- Complicated definition through **tree decompositions**.
- Pathwidth: restriction where decomposition is a path.
- Trees have treewidth 1 (but pathwidth up to $\log n$).
- Caterpillars have pathwidth 1.
- HUGE number of problems FPT by tw.
- BUT in some cases too general. . .



- Tree-depth

$$\mathrm{td}(G) = \min_{S \subseteq V(G)} \left\{ |S| + \max_{S' \in \mathrm{cc}(G-S)} \mathrm{td}(S') \right\}$$

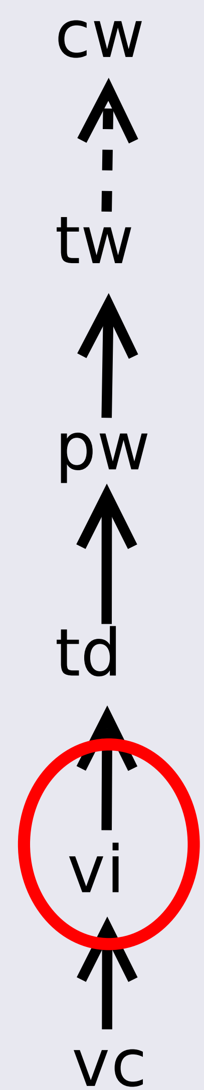
- Select small separator S so that all components have small **tree-depth**
- (Base case: K_1 has tree-depth 1)



- Vertex Integrity

$$\mathrm{vi}(G) = \min_{S \subseteq V(G)} \left\{ |S| + \max_{S' \in \mathrm{cc}(G-S)} |S'| \right\}$$

- Select small separator S so that all components have small **size**

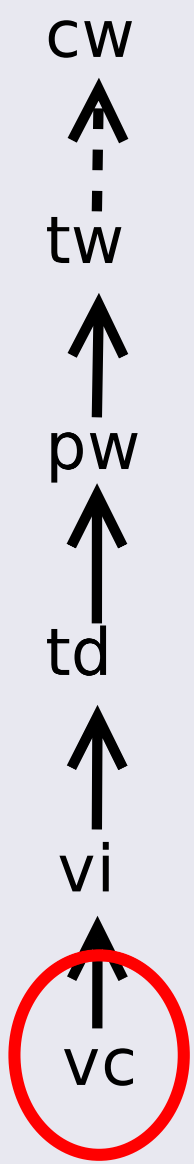


Parameters Review

- Vertex Cover

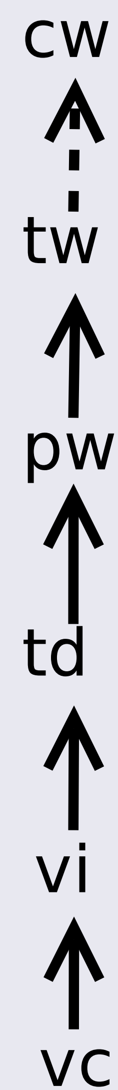
$$\mathrm{vc}(G) = \min_{S \subseteq V(G) \wedge G - S \text{ stable}} \{|S|\}$$

- Select small separator S so that all components **are singletons**.



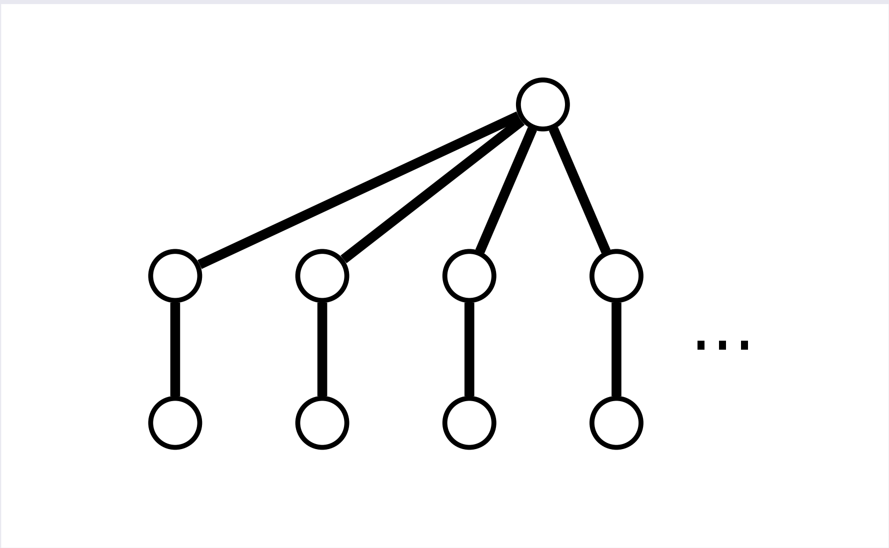
A Closer Look

- Will focus on tree-depth, vertex integrity, vertex cover
- Measure “complexity” as size of a small separator such that:
  - Each component is recursively defined as simple (tree-depth).
  - Each component is small, therefore simple (vertex integrity).
  - Each component is one vertex, therefore simple.

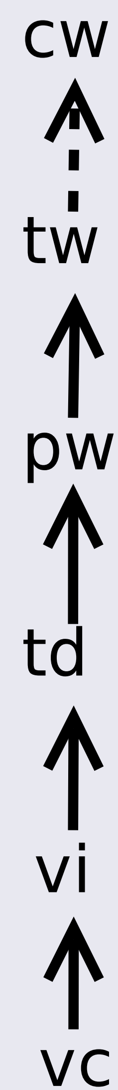


A Closer Look

- Inclusions are strict!

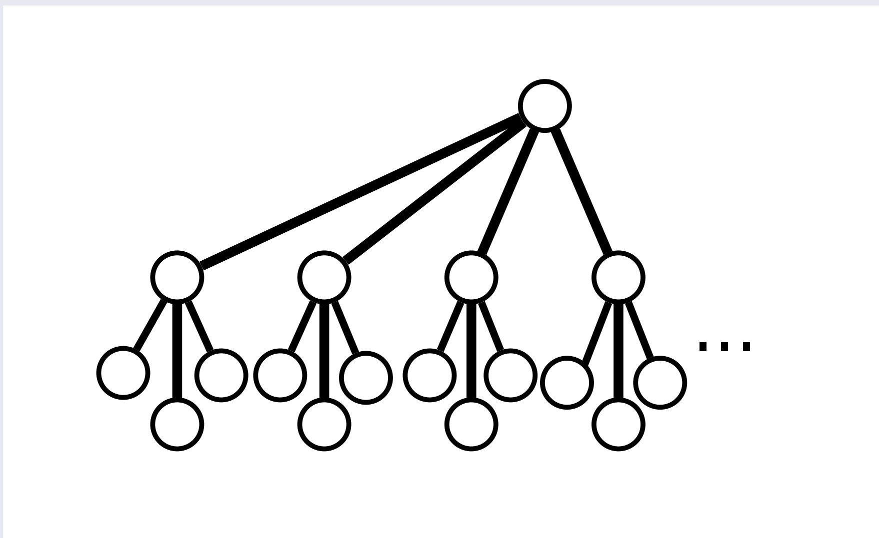


- Small vertex integrity, large vertex cover

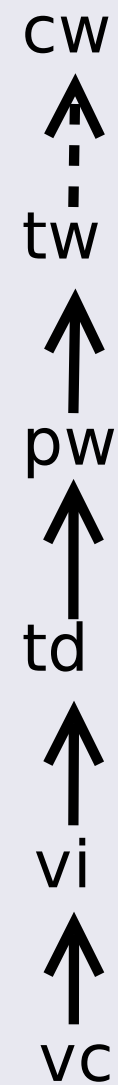


A Closer Look

- Inclusions are strict!

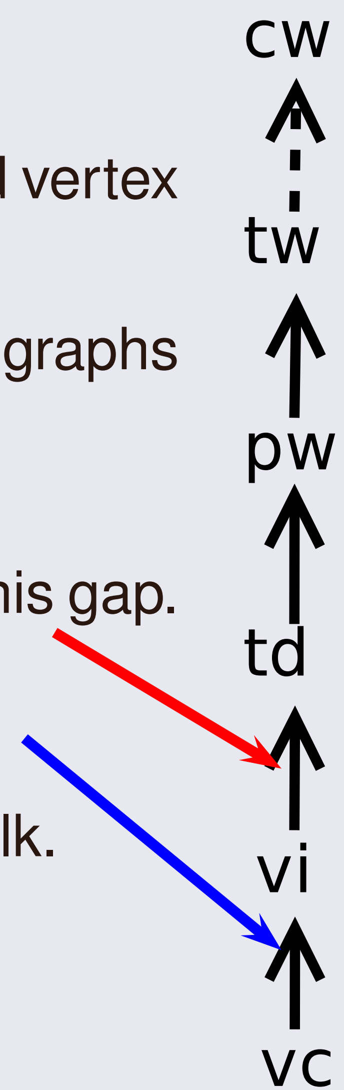


- Large vertex integrity, small tree-depth



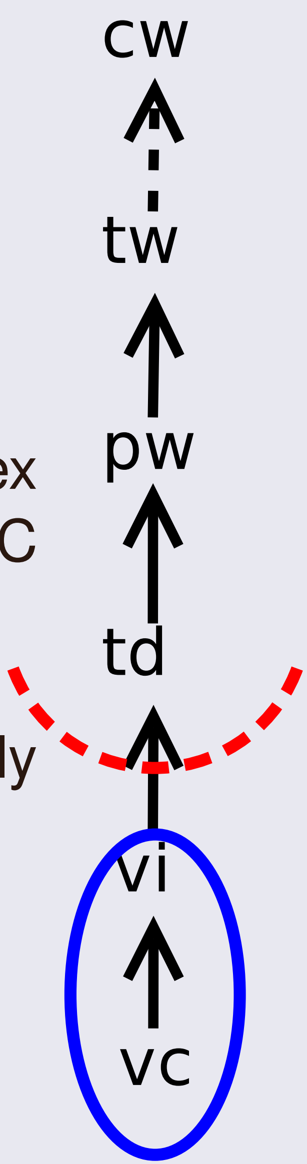
A Closer Look

- Generality: gap is **huge** between tree-depth and vertex integrity
  - If we fix k there are only polynomially many graphs of order n with vc, vi at most k
  - But exponentially many graphs with $\mathrm{td} \leq k$.
- **Intuitively** problems should become harder in this gap.
- **Intuitively** this gap should not be so important.
  - This is (more or less) the message of this talk.



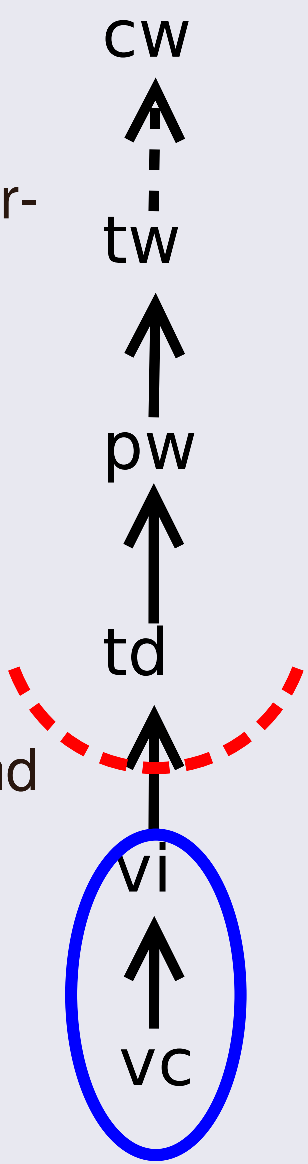
How to measure algorithmic cost?

- Look at many individual problems
    - For $\mathrm{vc} \to \mathrm{vi} \to \mathrm{td}$ cf.
      "Exploring the Gap Between Treedepth and Vertex Cover Through Vertex Integrity", Gima et al. CIAC 2021
    - Main message (approximately):
      "Problems hard for td but easy for vc are usually easy for vi"



Fine-Grained Meta-Theorems for Vertex Integrity

- Consider **categories** of problems expressible in a certain logic
  - $\rightarrow$ **Meta-Theorems**
- Measure complexity using ETH
  - $\rightarrow$ **Fine-Grained**
- Main message:
  Vertex Integrity is **a little** harder than vertex cover and **a lot** easier than tree-depth.



Meta-Theorems

Meta-Theorems Reminder

- Statements of the form:
  "Every problem in family $\mathcal{F}$ is *tractable*"
  - Family $\mathcal{F}$: often "expressible in FO/MSO or other logic"
  - Tractable: often "FPT parameterized by some parameter"

Meta-Theorems Reminder

- Statements of the form:
  "Every problem in family $\mathcal{F}$ is *tractable*"
  - Family $\mathcal{F}$: often "expressible in FO/MSO or other logic"
  - Tractable: often "FPT parameterized by some parameter"

  Courcelle's famous meta-theorem:

All problems expressible in MSO logic are FPT parameterized by treewidth.

Meta-Theorems Reminder

- Statements of the form:
  "Every problem in family $\mathcal{F}$ is *tractable*"
  - Family $\mathcal{F}$: often "expressible in FO/MSO or other logic"
  - Tractable: often "FPT parameterized by some parameter"

  Courcelle's famous meta-theorem:

All problems expressible in MSO logic are FPT parameterized by treewidth.

- Notice that since this applies to treewidth, it applies to pathwidth, tree-depth, vertex integrity, vertex cover!

FO logic:

- Two relations: $=$ and $\sim$ (equality, adjacency)
- (Quantified) Variables $x_1, x_2, \ldots$ represent vertices
- Standard boolean connectives $(\vee, \wedge, \neg, \rightarrow)$

Standard Example: 2-Dominating set

$$\exists x_1 \exists x_2 \forall x_3 \, (x_1 = x_3 \vee x_2 = x_3 \vee x_1 \sim x_3 \vee x_2 \sim x_3)$$

FO logic:

- Two relations: $=$ and $\sim$ (equality, adjacency)
- (Quantified) Variables $x_1, x_2, \ldots$ represent vertices
- Standard boolean connectives $(\vee, \wedge, \neg, \rightarrow)$

MSO logic: FO logic plus the following

- $\in$ relation
- (Quantified) **Set** Variables $X_1, X_2, \ldots$ represent sets of vertices

Standard Examples: 3-Coloring, Connectivity

$$
\begin{aligned}
\exists X_1 \exists X_2 \exists X_3 \quad \Big(\forall x_1 \quad & (x_1 \in X_1 \vee x_1 \in X_2 \vee x_1 \in X_3) \wedge \\
\forall x_2 \quad & (x_1 \sim x_2 \rightarrow (\neg(x_1 \in X_1 \wedge x_2 \in X_1)) \wedge \\
& (\neg(x_1 \in X_2 \wedge x_2 \in X_2)) \wedge \\
& (\neg(x_1 \in X_3 \wedge x_2 \in X_3))) \Big)
\end{aligned}
$$

FO and MSO logic reminder

FO logic:

- Two relations: $=$ and $\sim$ (equality, adjacency)
- (Quantified) Variables $x_1, x_2, \ldots$ represent vertices
- Standard boolean connectives $(\vee, \wedge, \neg, \rightarrow)$

MSO logic: FO logic plus the following

- $\in$ relation
- (Quantified) **Set** Variables $X_1, X_2, \ldots$ represent sets of vertices

Standard Examples: 3-Coloring, Connectivity

$$\begin{aligned} \forall X_1 \qquad & ((\exists x_1 \exists x_2 \;\; x_1 \in X_1 \wedge x_2 \notin X_1) \rightarrow \\ & \exists x_3 \exists x_4 \; (x_3 \in X_1 \wedge x_4 \notin X_1 \wedge x_3 \sim x_4)) \end{aligned}$$

FO logic:

- Two relations: $=$ and $\sim$ (equality, adjacency)
- (Quantified) Variables $x_1, x_2, \ldots$ represent vertices
- Standard boolean connectives $(\vee, \wedge, \neg, \rightarrow)$

MSO logic: FO logic plus the following

- $\in$ relation
- (Quantified) **Set** Variables $X_1, X_2, \ldots$ represent sets of vertices

Standard Examples: 3-Coloring, Connectivity
Brute-force Complexity:

- FO: n^q
- MSO: 2^{nq}

Note: MSO=MSO$_1$. No edge set quantifiers in this talk.

A Closer Look

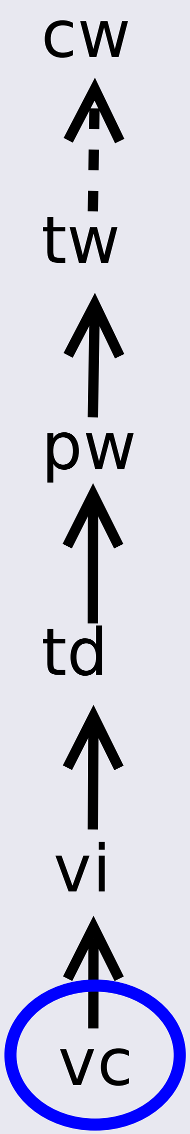
- Courcelle: If G has treewidth tw, we can check if it satisfies an MSO property ϕ in time
$$f(\mathrm{tw}, \phi) \cdot |G|$$
- Problem: f is approximately $2^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathrm{tw}}}}}}}}$, where the height of the tower is upper-bounded by the number of **quantifier alternations** in ϕ.

A Closer Look

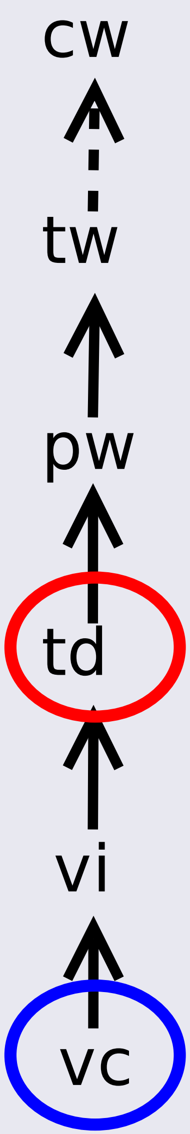
- Courcelle: If G has treewidth tw, we can check if it satisfies an MSO property ϕ in time
$$f(\mathrm{tw}, \phi) \cdot |G|$$
- Problem: f is approximately $2^{2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathrm{tw}}}}}}}}$, where the height of the tower is upper-bounded by the number of **quantifier alternations** in ϕ.
- **Serious Problem**: This tower of exponentials cannot be avoided[1] even for **FO logic on trees**!
  - "The complexity of first-order and monadic second-order logic revisited", Frick and Grohe, APAL 2004.
- **Question**: Does f become nicer if we go lower in our parameter map?

[1] Assuming P$\neq$NP

- Vertex Cover
  - MSO with q quantifiers can be decided in $2^{2^{O(\mathrm{vc}+q)}}$
  - FO with q quantifiers can be decided in $2^{O(\mathrm{vc}\cdot q)}q^{O(q)}$
  - These are **optimal under ETH**.
    - There exists fixed MSO formula which cannot be decided in $2^{2^{o(\mathrm{vc})}}$.
- "Algorithmic Meta-Theorems for Restrictions of Treewidth", L. Algorithmica 2012.

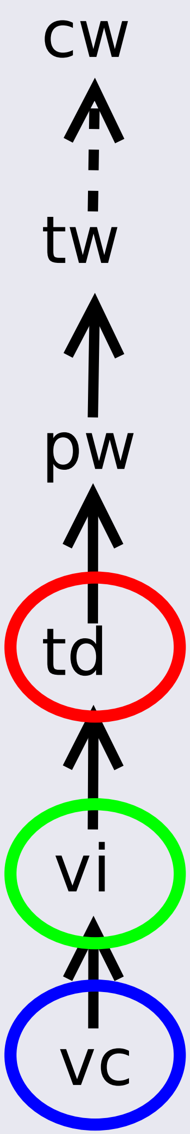


- Tree-depth
  - MSO/FO with q quantifiers can be decided by an algorithm running in time $2^{2^{\cdot^{\cdot^{\cdot^{2^{\mathrm{td}+q}}}}}}$
  - ...where height of tower is at most td (even for large q)
  - This is **optimal under ETH**.
- "Kernelizing MSO Properties of Trees of Fixed Height, and Some Consequences", Gajarsky and Hlineny, LMCS 2015.
- "Model-Checking Lower Bounds for Simple Graphs", L. LMCS 2014.



This talk

- Vertex Integrity
  - FO can be done in: $2^{O(\mathrm{vi}^2 q)} q^{O(q)}$
  - MSO can be done in: $2^{2^{O(\mathrm{vi}^2 + \mathrm{vi} \cdot q)}}$
  - Both of these results are optimal under the ETH.
- Comparison:
  - For vc we have similar complexity, without the square.
    MSO in $2^{2^{O(\mathrm{vc}+q)}}$, FO in $2^{O(\mathrm{vc} \cdot q)}$.
  - For td we have tower of exps.
- Conclusion:
  - Complexity of vi much closer to vc, slightly worse.

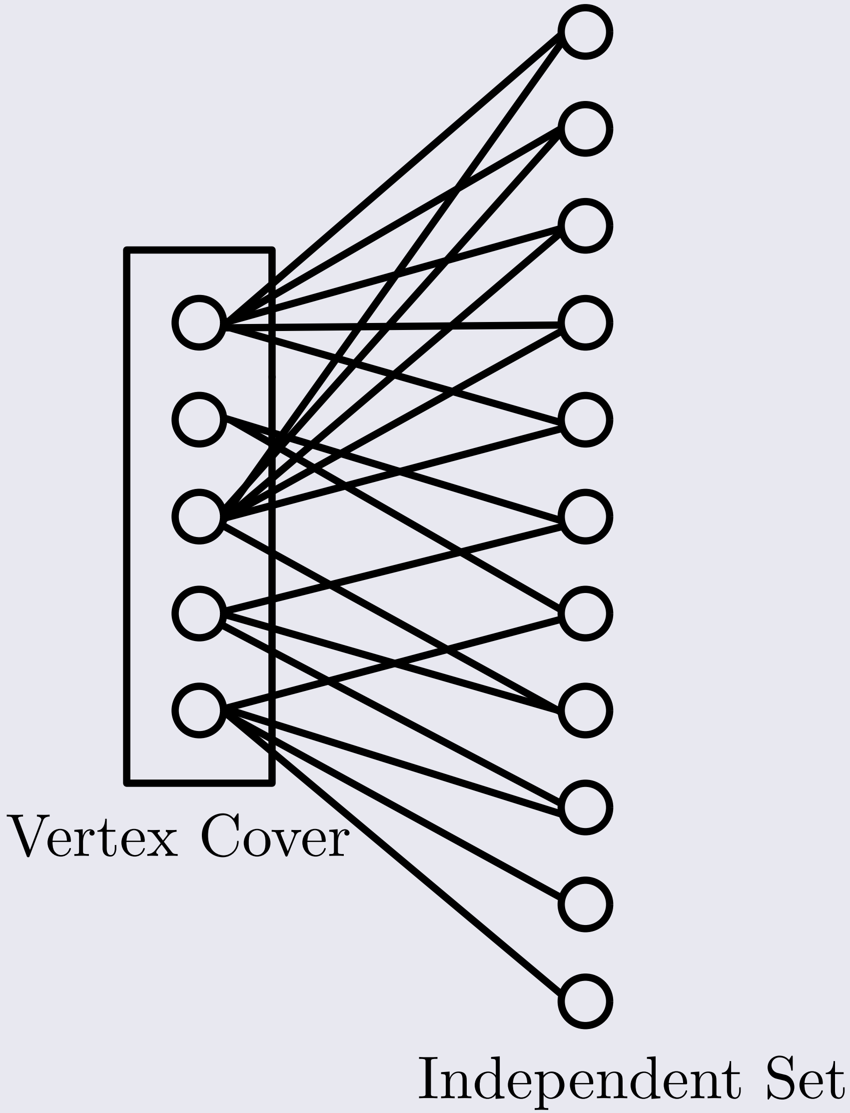


Meta-Theorems for Vertex Integrity

High-level Idea

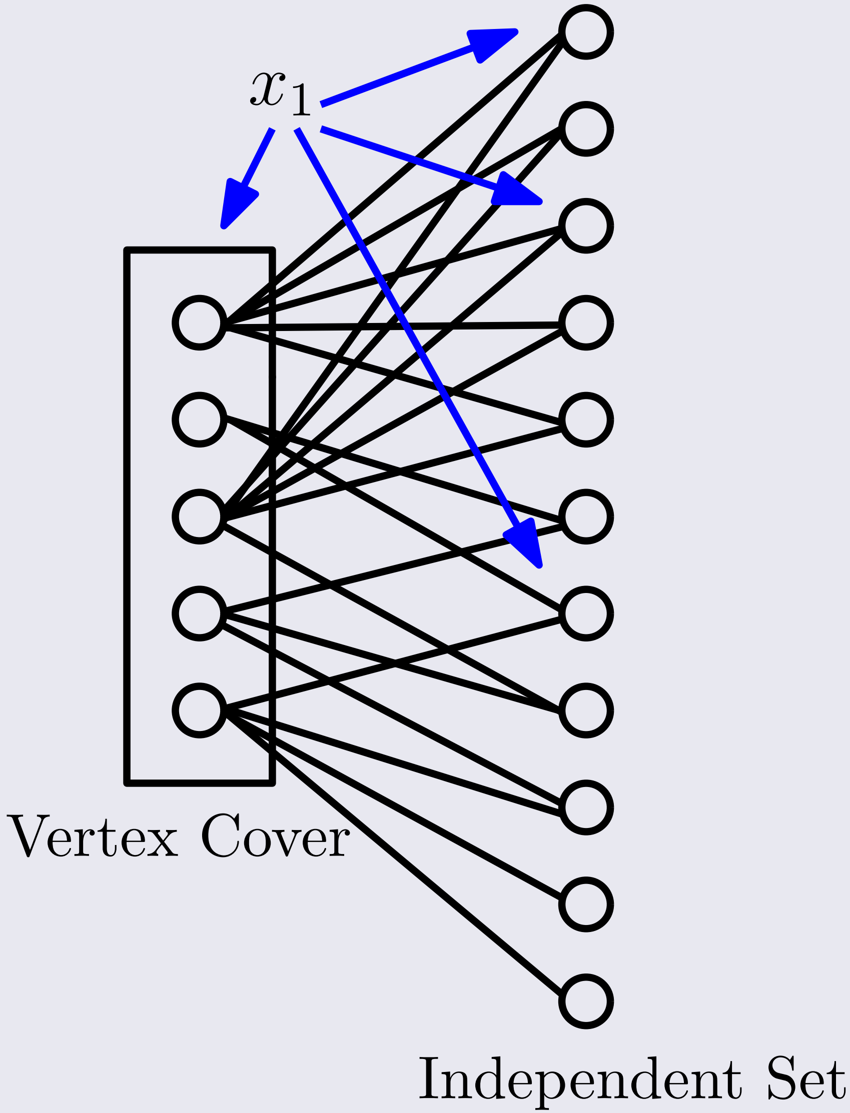
- Algorithm idea similar to meta-theorems for **vertex cover** and **tree-depth**.
- Kernelization argument.
    - If graph too large, we can delete something without affecting whether given property is satisfied.
- Brute-force.
    - Once previous argument does not apply, size of graph can be bounded by function of parameter and q.
    - Run trivial algorithm on this kernel.
- Main Kernelization Trick:
    - If we have many copies of the same thing, we can delete some.
    - (cf. What is the counting power of FO and MSO logic?)

Vertex Cover Meta-Theorem – Reminder



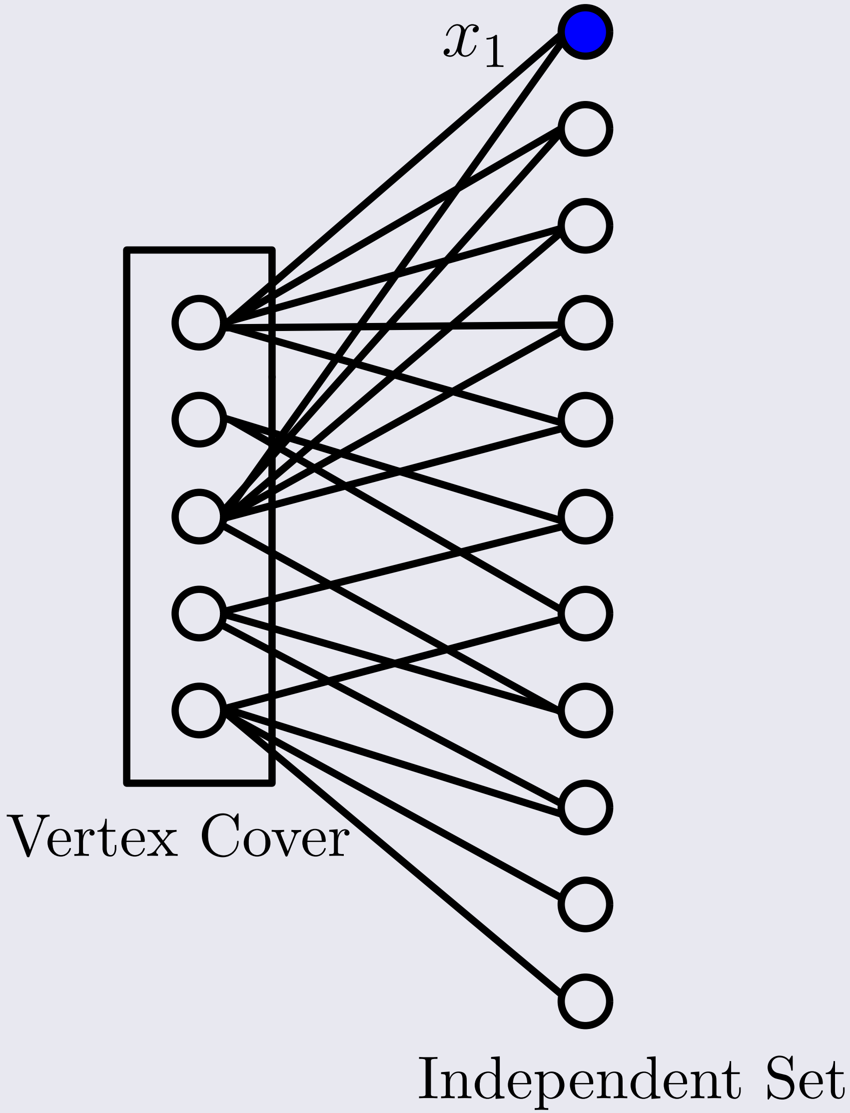
- Given a graph with vertex cover $\mathrm{vc} = 5$
- we want to check an FO property ϕ with $q = 3$ variables.

Vertex Cover Meta-Theorem – Reminder



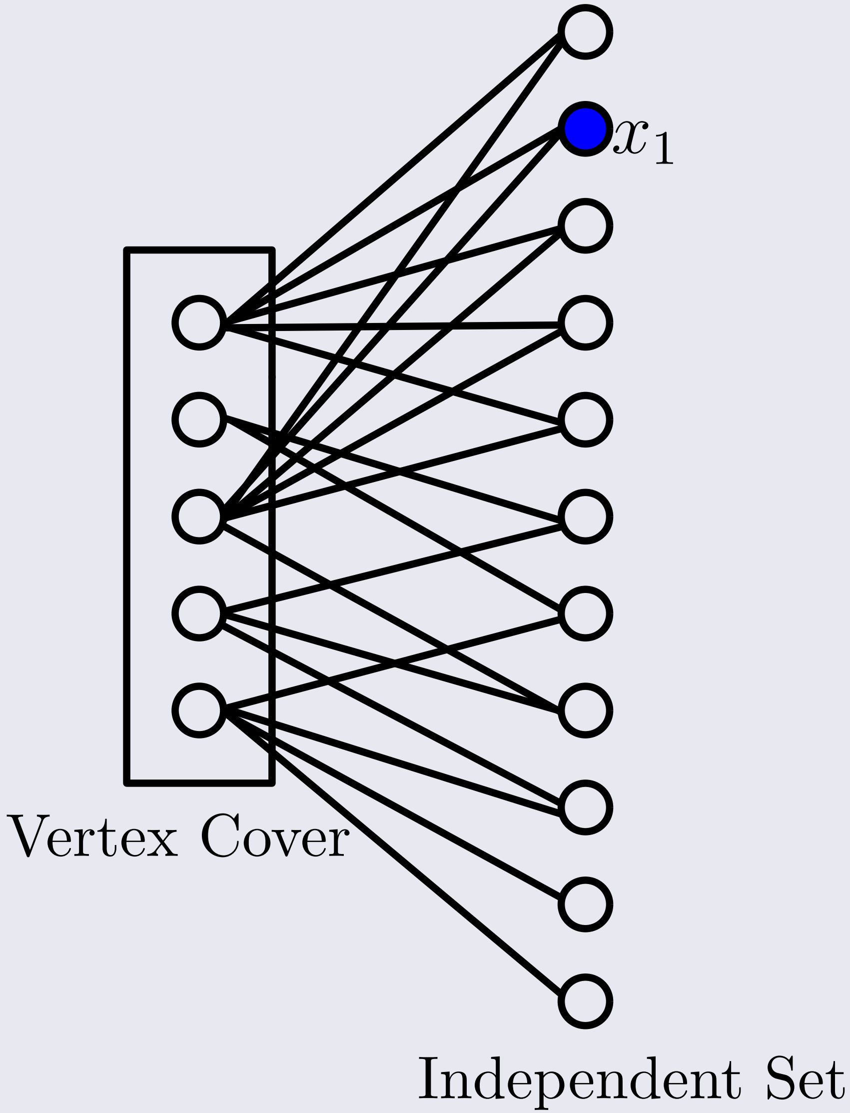
- Sentence has form $\exists x_1 \psi(x_1)$
- We must “place” x_1 somewhere in the graph
- If we try all cases we get n^q running time.

Vertex Cover Meta-Theorem – Reminder



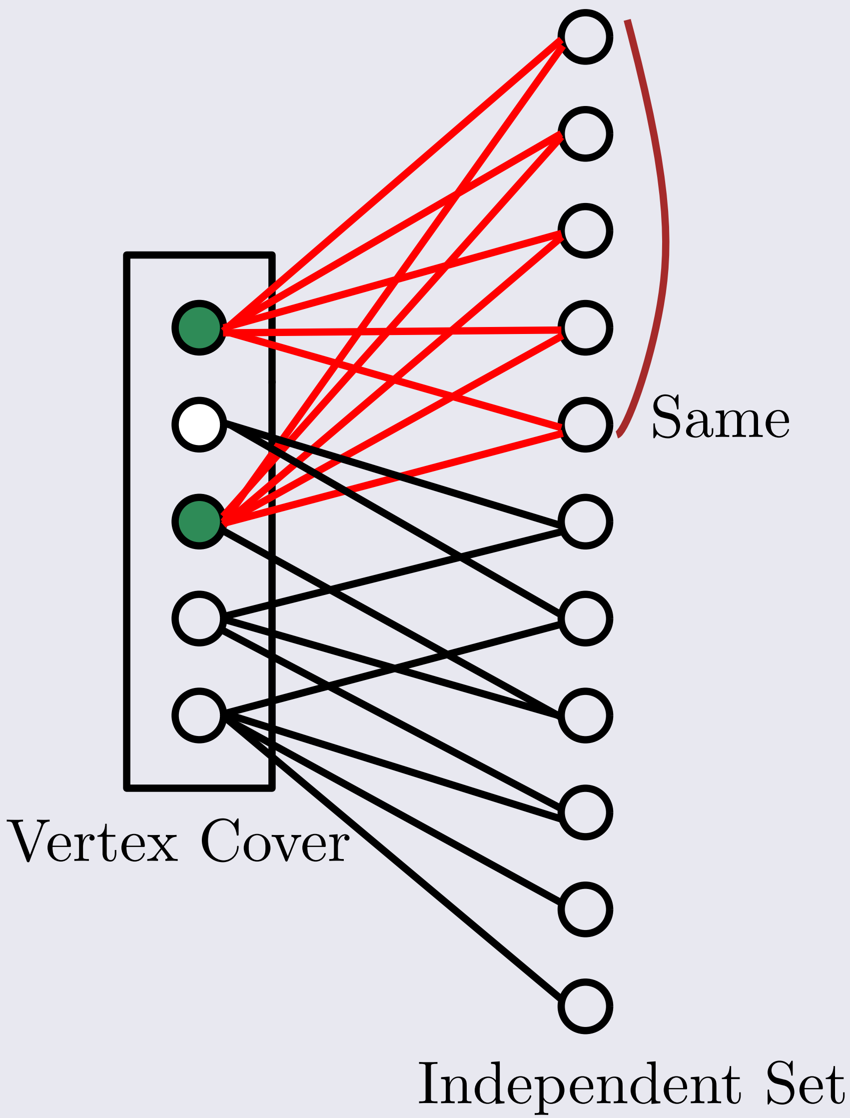
- Sentence has form $\exists x_1 \psi(x_1)$
- We must "place" x_1 somewhere in the graph
- If we try all cases we get n^q running time.

Vertex Cover Meta-Theorem – Reminder



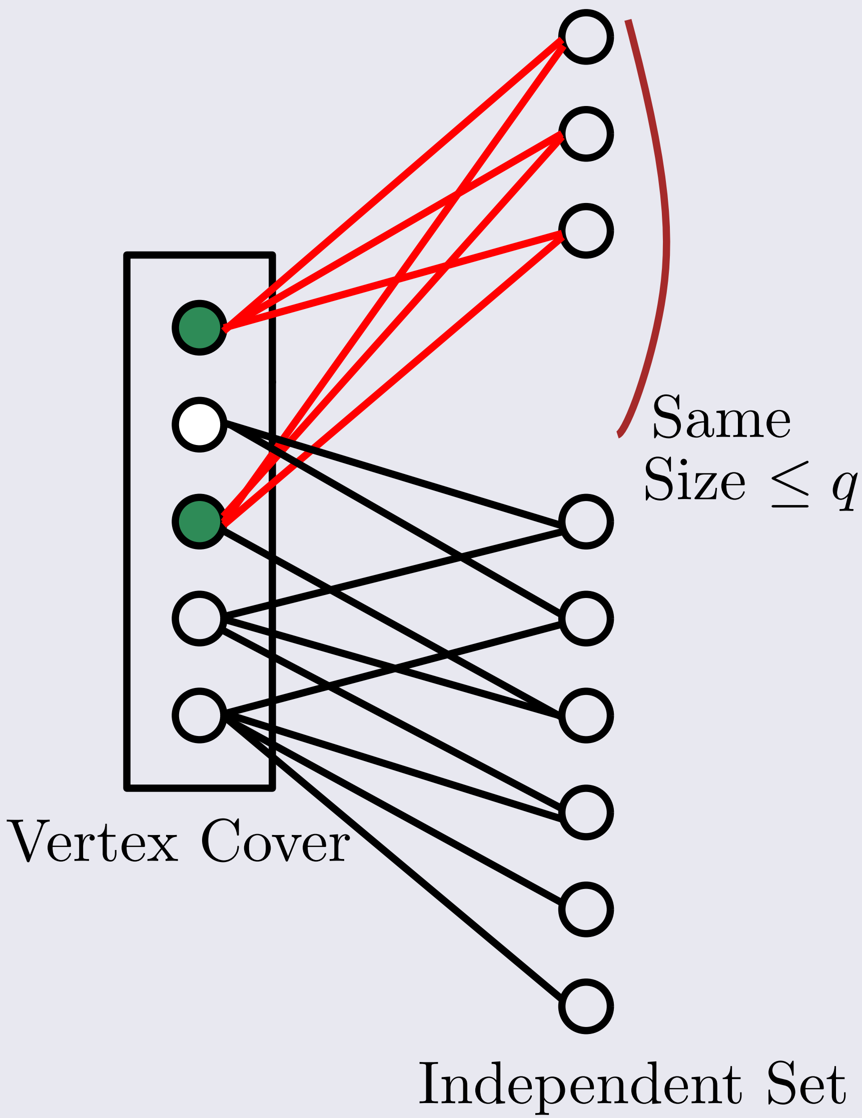
- Sentence has form $\exists x_1 \psi(x_1)$
- We must “place” x_1 somewhere in the graph
- If we try all cases we get n^q running time.

Vertex Cover Meta-Theorem – Reminder



- We observe that some vertices of the independent set have the same neighbors.
- These vertices should be equivalent.

Vertex Cover Meta-Theorem – Reminder



- We observe that some vertices of the independent set have the same neighbors.
- These vertices should be equivalent.
- Key idea: if a group has $> q$ vertices, we can simply remove one!

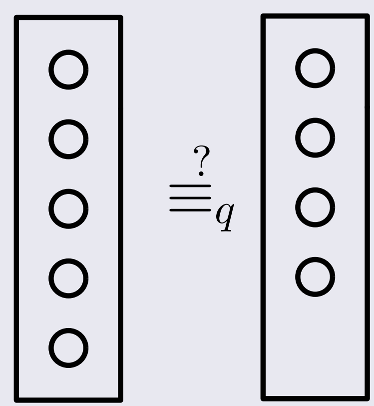
Vertex Cover and FO logic

Summary of previous argument:

- Partition graph into $2^{\mathrm{vc}} + \mathrm{vc}$ sets of equivalent vertices.
- If a set has $> q$ vertices, delete one, repeat.
- If not, $|V(G)| \leq q2^{O(\mathrm{vc})}$.
- Trivial algorithm now runs in $2^{O(\mathrm{vc}\cdot q)}q^q$.

Key idea:

$\boxed{\text{FO logic with } q \text{ quantifiers can distinguish sets of size at most } q.}$



We need at least 5 quantifiers to construct a formula that is true on exactly one of these graphs.

Vertex Cover and FO logic

Summary of previous argument:

- Partition graph into $2^{\mathrm{vc}} + \mathrm{vc}$ sets of equivalent vertices.
- If a set has $> q$ vertices, delete one, repeat.
- If not, $|V(G)| \leq q2^{O(\mathrm{vc})}$.
- Trivial algorithm now runs in $2^{O(\mathrm{vc}\cdot q)}q^q$.

Key idea:

$\boxed{\text{FO logic with } q \text{ quantifiers can distinguish sets of size at most } q.}$

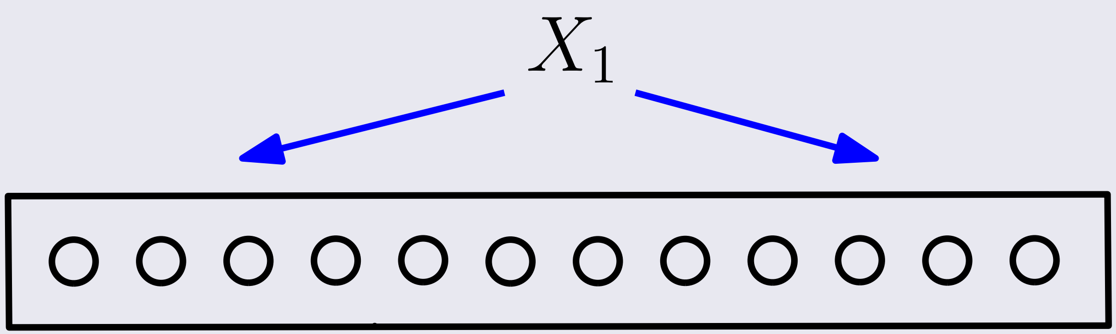
What about MSO?

Key idea:

$\boxed{\text{MSO logic with } q \text{ quantifiers can distinguish sets of size at most } 2^q.}$

Proof by induction:

- Want to prove, if set has size $> 2^q$, can delete one vertex.
- Suppose OK for up to $q-1$ quantifiers.
- Want to check if $\exists X_1 \psi(X_1)$, where ψ has $q-1$ quantifiers.



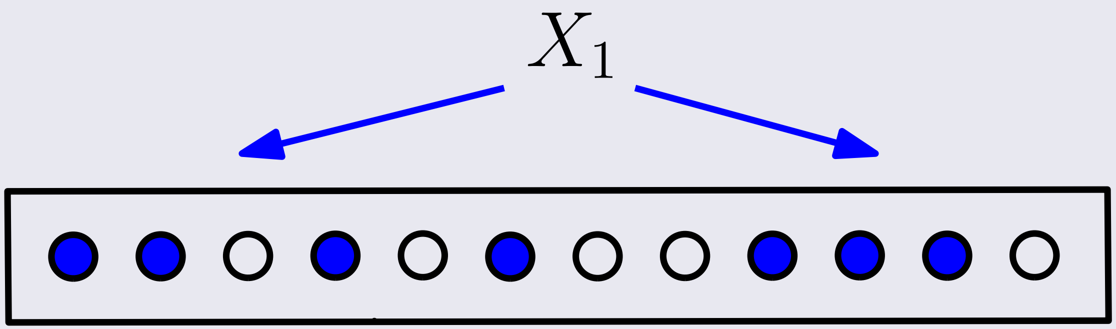
MSO and Vertex Cover

Key idea:

MSO logic with q quantifiers can distinguish sets of size at most 2^q.

Proof by induction:

- Want to prove, if set has size $> 2^q$, can delete one vertex.
- Suppose OK for up to $q-1$ quantifiers.
- Want to check if $\exists X_1 \psi(X_1)$, where ψ has $q-1$ quantifiers.



- For any choice of X_1 a set of 2^{q-1} identical vertices remains.
- Apply inductive hypothesis.

Key idea:

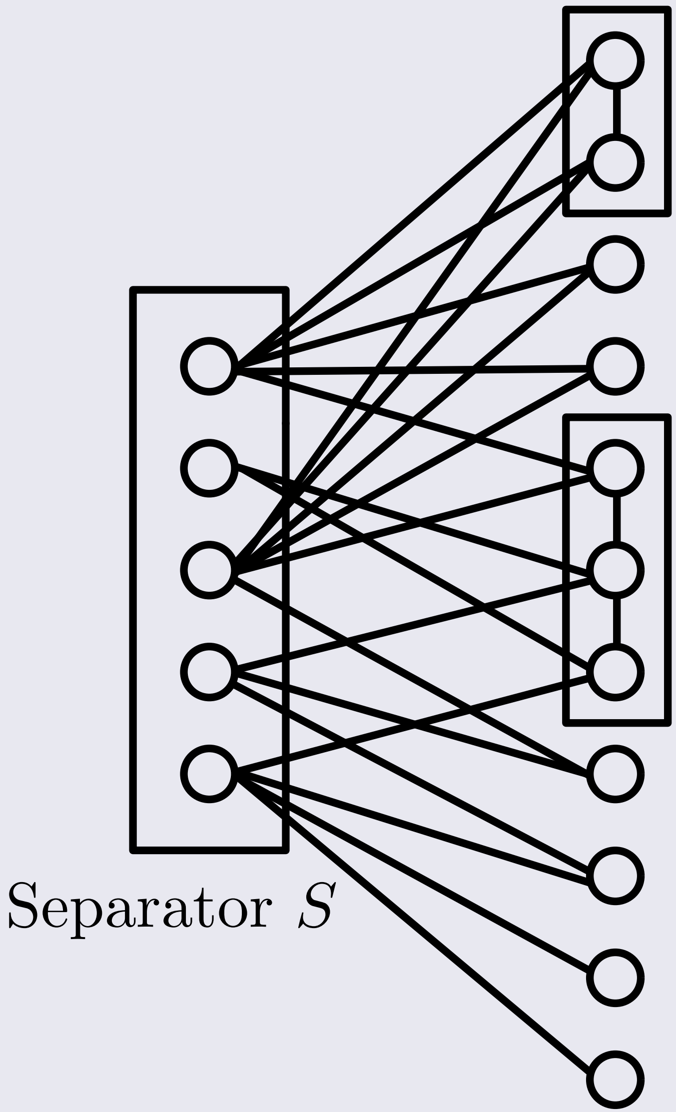
$\boxed{\text{MSO logic with } q \text{ quantifiers can distinguish sets of size at most } 2^q.}$

- Graph has 2^{vc} sets of equivalent vertices.
- While one has size $> 2^q$, delete a vertex.
- Otherwise, $|V(G)| \leq 2^{\mathrm{vc}+q}$.
- Brute force:

$$2^{nq} \leq 2^{2^{\mathrm{vc}+q}q} = 2^{2^{O(\mathrm{vc}+q)}}$$

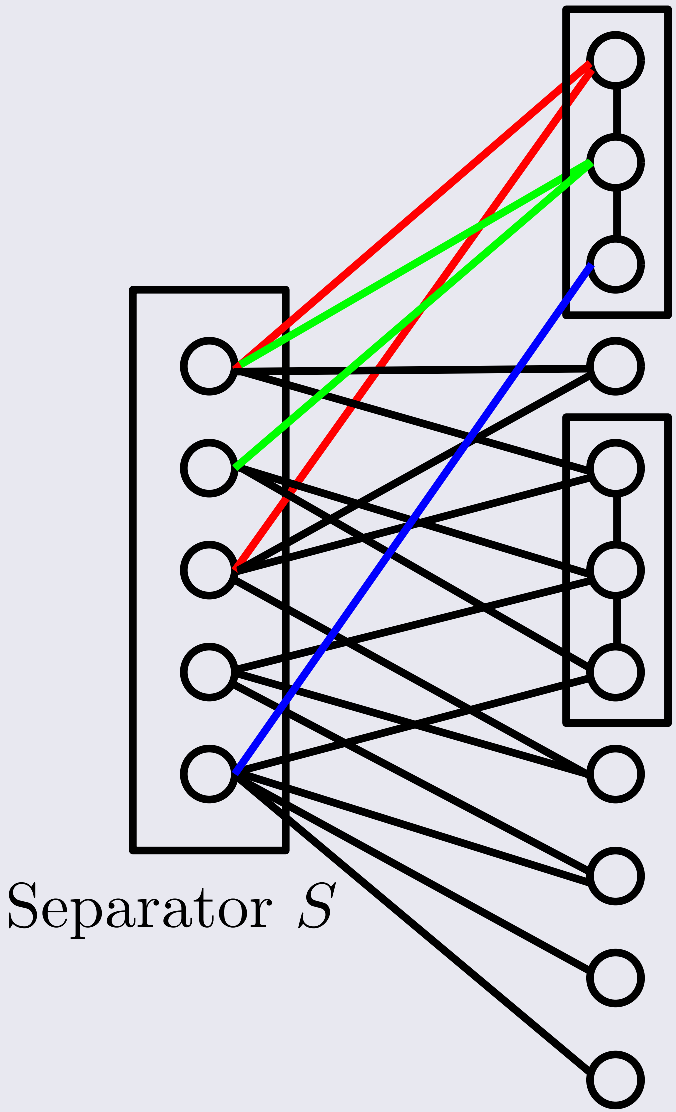
Vertex Integrity

What is different now?



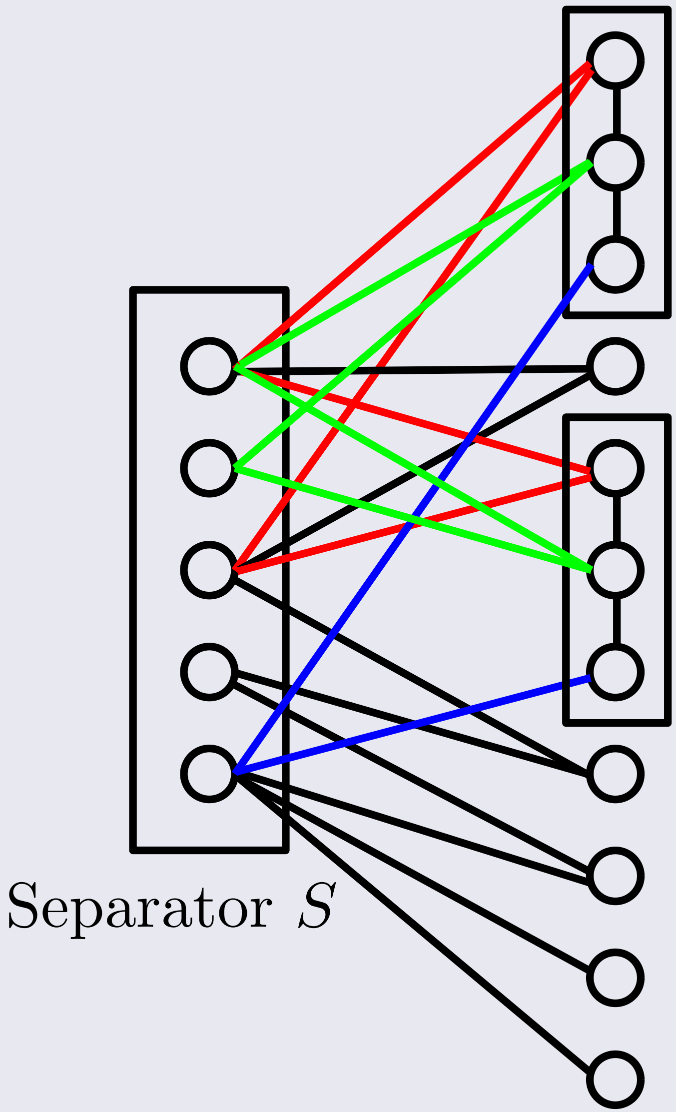
- Main idea: some components of $G - S$ are the same.
  - The same internally.
  - The same with respect to S.
- More precisely:
  - Two components C_1, C_2 of $G - S$ are "the same" if there exists an automorphism of G that maps C_1 to C_2.

What is different now?



- Main idea: some components of $G - S$ are the same.
  - The same internally.
  - The same with respect to S.
- More precisely:
  - Two components C_1, C_2 of $G - S$ are "the same" if there exists an automorphism of G that maps C_1 to C_2.

What is different now?



- Main idea: some components of $G - S$ are the same.
  - The same internally.
  - The same with respect to S.
- More precisely:
  - Two components C_1, C_2 of $G - S$ are "the same" if there exists an automorphism of G that maps C_1 to C_2.

Vertex Integrity

- Previously:
  - We defined "equivalence" for vertices.
  - We showed that if we have many equivalent vertices, we can delete one.
  - We counted how many equivalence types there are.
- Now:
  - We defined "equivalence" for components of $G - S$.

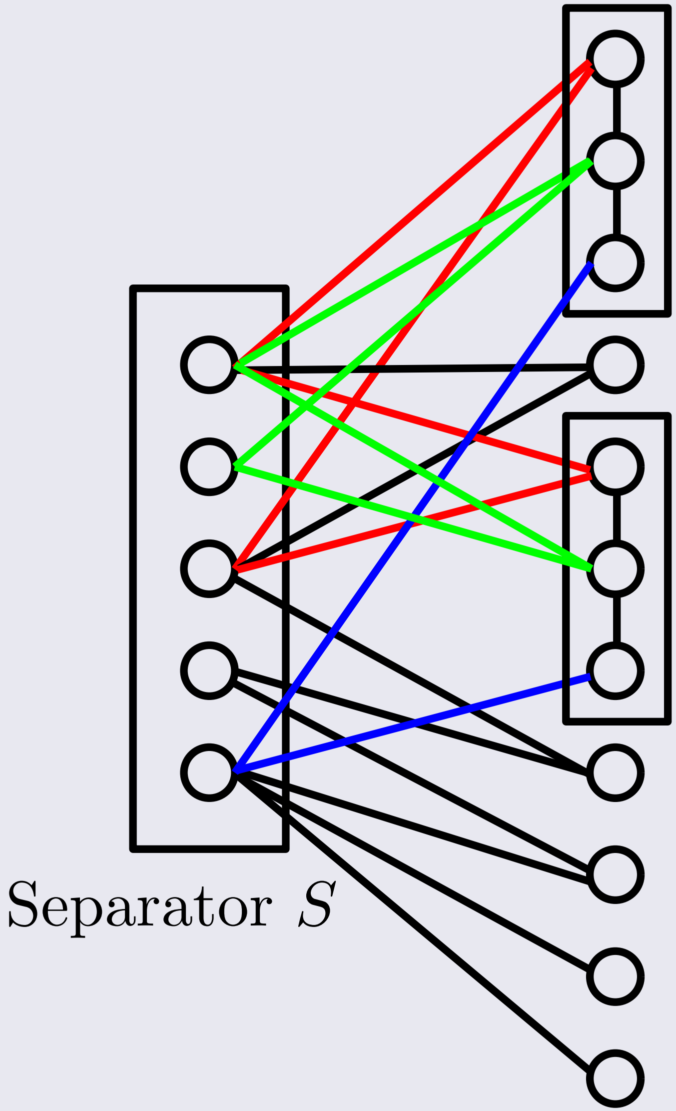
Vertex Integrity

- Previously:
  - We defined "equivalence" for vertices.
  - We showed that if we have many equivalent vertices, we can delete one.
  - We counted how many equivalence types there are.
- Now:
  - We defined "equivalence" for components of $G - S$.

What do we need now?

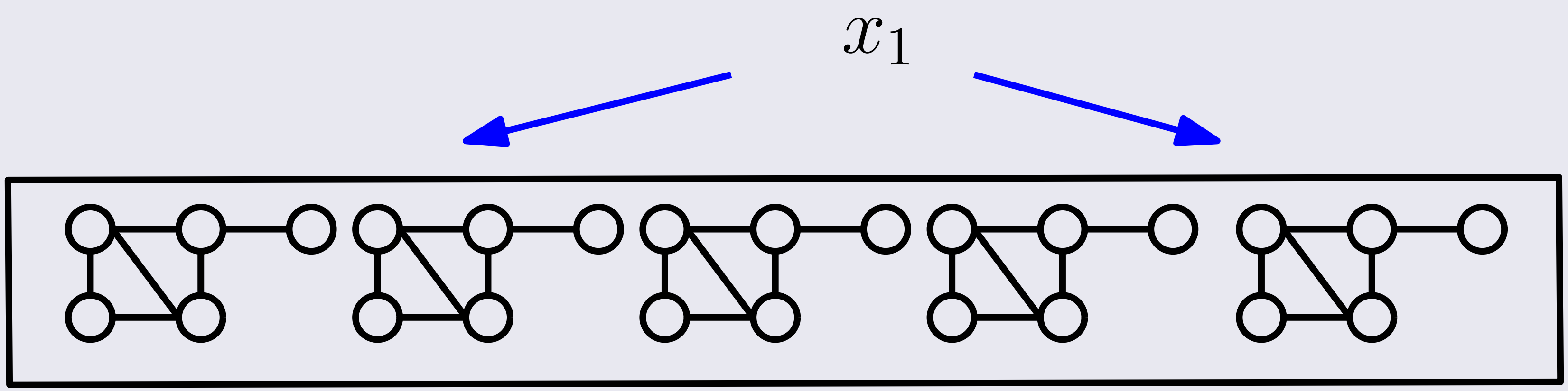
- Understand counting power of FO/MSO for collections of identical components.
- Count number of possible component types.

How many types of components?



- Equivalent components of $G - S$ are
  - The same internally.
  - The same with respect to S.
- How many choices?
- Recall, components of $G - S$ have size $\leq \mathrm{vi}$
  - At most 2^{vi^2} different internal structures.
  - At most 2^{vi^2} different connections to S.
- All in all, $2^{O(\mathrm{vi}^2)}$ possible types.

How many identical components can we distinguish with q FO quantifiers?



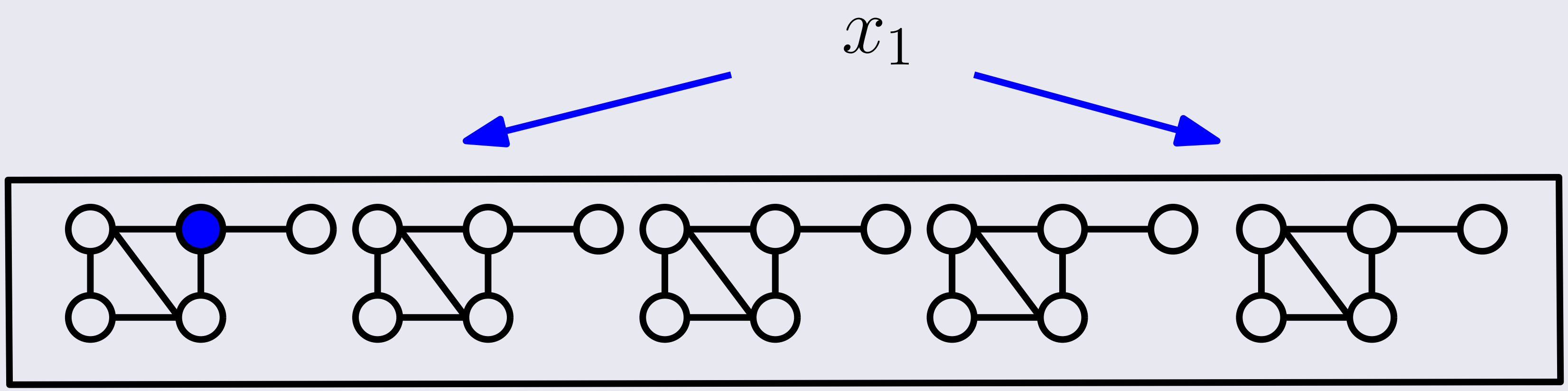
Claim: if we have $> q$ components, we can delete one.

Induction:

- Suppose true for $q - 1$ quantifiers.
- We have a formula $\exists x_1 \psi(x_1)$, where ψ has $q - 1$ quantifiers.
- Mapping it to any component is the same.
- We have $> q - 1$ identical components left.
- By induction, we can delete one.

Counting Power – FO

How many identical components can we distinguish with q FO quantifiers?



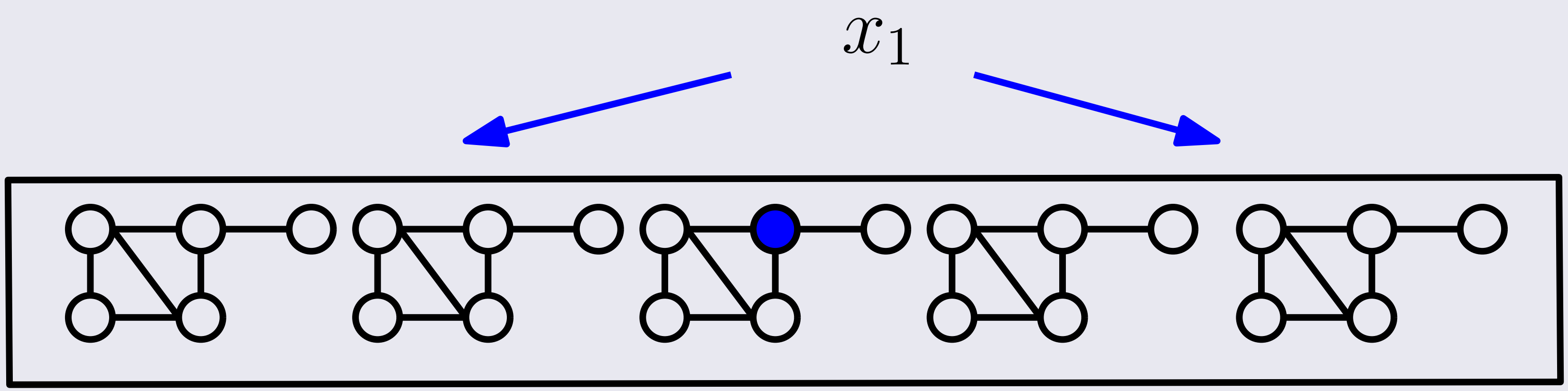
Claim: if we have $> q$ components, we can delete one.

Induction:

- Suppose true for $q-1$ quantifiers.
- We have a formula $\exists x_1 \psi(x_1)$, where ψ has $q-1$ quantifiers.
- Mapping it to any component is the same.
- We have $> q-1$ identical components left.
- By induction, we can delete one.

Counting Power – FO

How many identical components can we distinguish with q FO quantifiers?



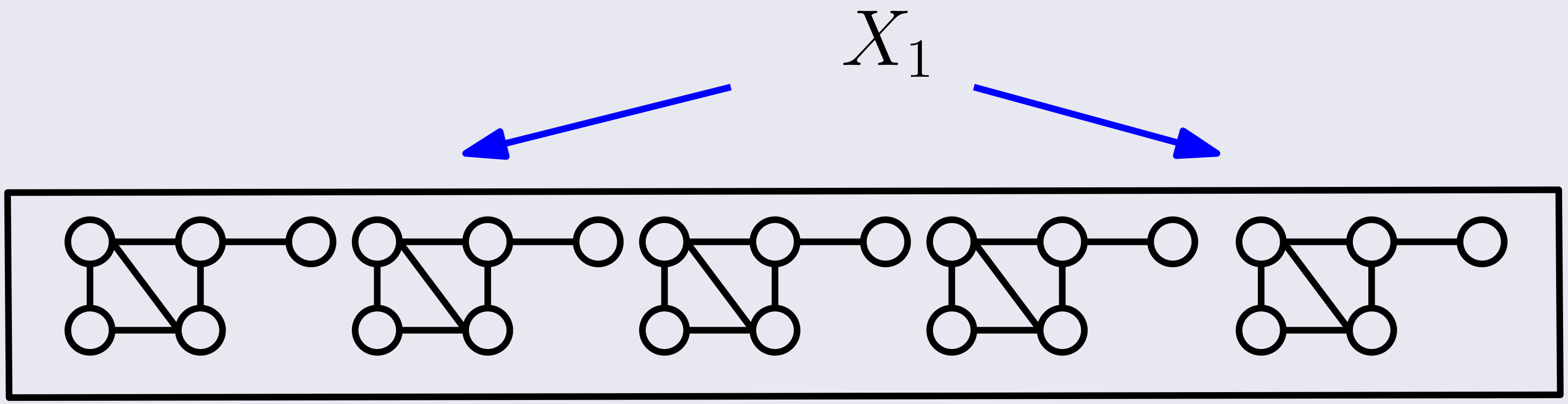
Claim: if we have $> q$ components, we can delete one.

Induction:

- Suppose true for $q-1$ quantifiers.
- We have a formula $\exists x_1 \psi(x_1)$, where ψ has $q-1$ quantifiers.
- Mapping it to any component is the same.
- We have $> q-1$ identical components left.
- By induction, we can delete one.

Counting Power – MSO

How many components can we distinguish with q MSO quantifiers?



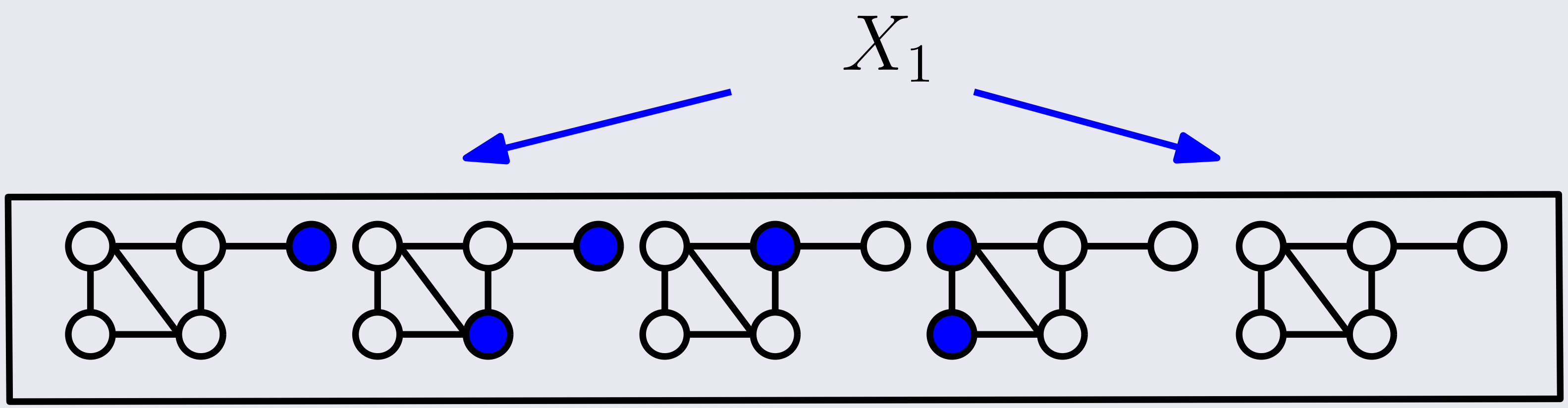
Claim: if we have $>$?? components, we can delete one.

Problem:

- When we select a set X_1 this may distinguish many components.
- Intuitively: if X_1 interacts with two previously identical components in different ways, these components are not identical any more!
- What to do?

Counting Power – MSO

How many components can we distinguish with q MSO quantifiers?

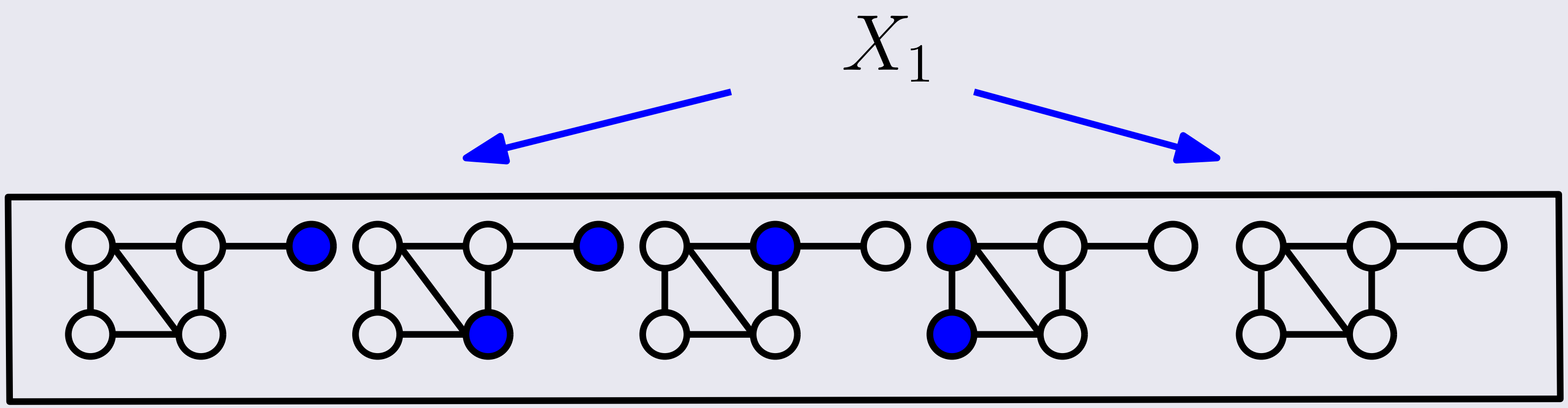


Claim: if we have $>$?? components, we can delete one.

Problem:

- When we select a set X_1 this may distinguish many components.
- Intuitively: if X_1 interacts with two previously identical components in different ways, these components are not identical any more!
- What to do?

How many components can we distinguish with q MSO quantifiers?



Claim: if we have $> 2^{\mathrm{vi} \cdot q}$ components, we can delete one.

Solution:

- Our components have size $\leq \mathrm{vi}$.
- There are at most 2^{vi} intersections of X_1 with each component.
- If we have $> 2^{\mathrm{vi} \cdot q}$ identical components initially...
- ...by PHP one intersection type appears $> 2^{\mathrm{vi} \cdot q}/2^{\mathrm{vi}} = 2^{\mathrm{vi}(q-1)}$ times.
- These components are identical, use inductive hypothesis!

Putting things together

- There are at most 2^{vi^2} types of components.
- Maximum number of same components in reduced graph is
  - q for FO logic.
  - $2^{\mathrm{vi} \cdot q}$ for MSO logic.

Putting things together

- There are at most 2^{vi^2} types of components.
- Maximum number of same components in reduced graph is
  - q for FO logic.
  - $2^{\mathrm{vi}\cdot q}$ for MSO logic.
- For FO logic
  - Reduced graph has size $q2^{\mathrm{vi}^2}$.
  - Trivial algorithm runs in $2^{q\cdot\mathrm{vi}^2}q^q$.

Putting things together

- There are at most 2^{vi^2} types of components.
- Maximum number of same components in reduced graph is
  - q for FO logic.
  - $2^{\mathrm{vi}\cdot q}$ for MSO logic.
- For FO logic
  - Reduced graph has size $q2^{\mathrm{vi}^2}$.
  - Trivial algorithm runs in $2^{q\cdot\mathrm{vi}^2}q^q$.
- For MSO logic
  - Reduced graph has size $2^{\mathrm{vi}^2+\mathrm{vi}\cdot q}$.
  - Trivial algorithm runs in $2^{2^{vi^2+\mathrm{vi}\cdot q}}$.
- Are these meta-theorems optimal?

Fine-Grained Lower Bounds

Fine-Grained Lower Bounds

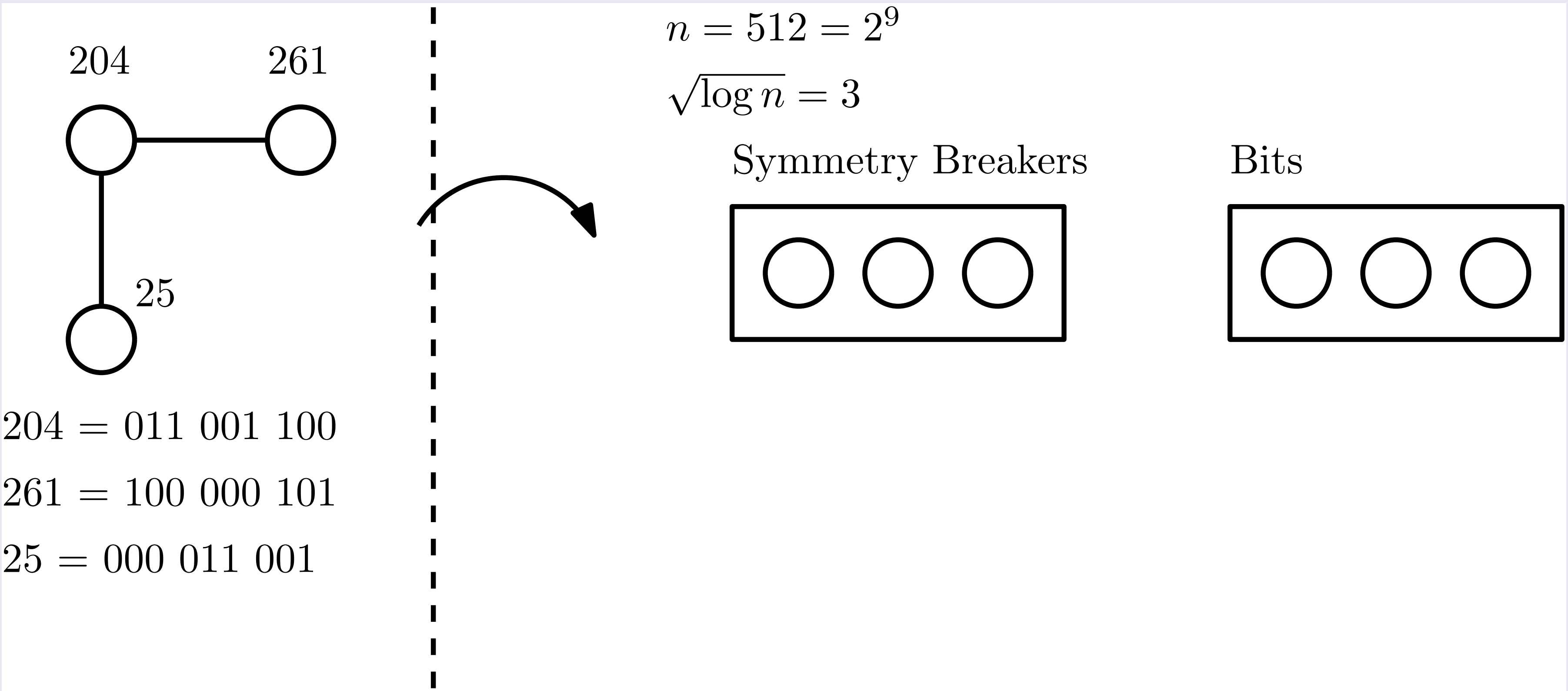
High-Level Idea

- We claim that we need time at least
  - $2^{\mathrm{vi}^2 \cdot q}$ for FO
  - $2^{2^{\mathrm{vi}^2}}$ for MSO

Strategy:

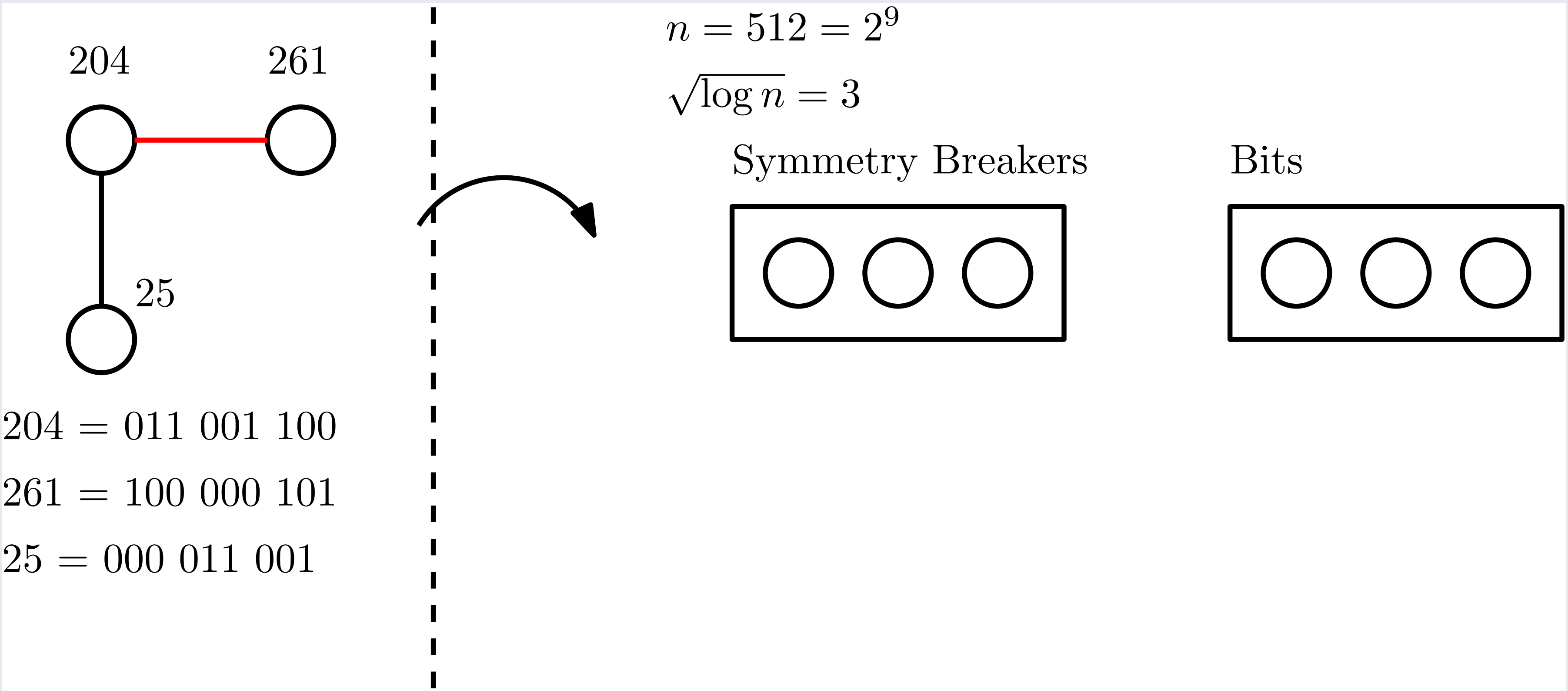
- Take an arbitrary n-vertex graph G
- Encode it into a graph H with the following properties:
  - $\mathrm{vi}(H) = \sqrt{\log n}$
  - Whether $uv \in E(G)$ can be tested with a simple FO formula on H
- Translate questions about G into questions about H.
  - G has k-clique? $\rightarrow$ FO on H with $q = k$
  - G is 3-colorable? $\rightarrow$ MSO on H with $q = O(1)$

Encoding graphs with simple graphs



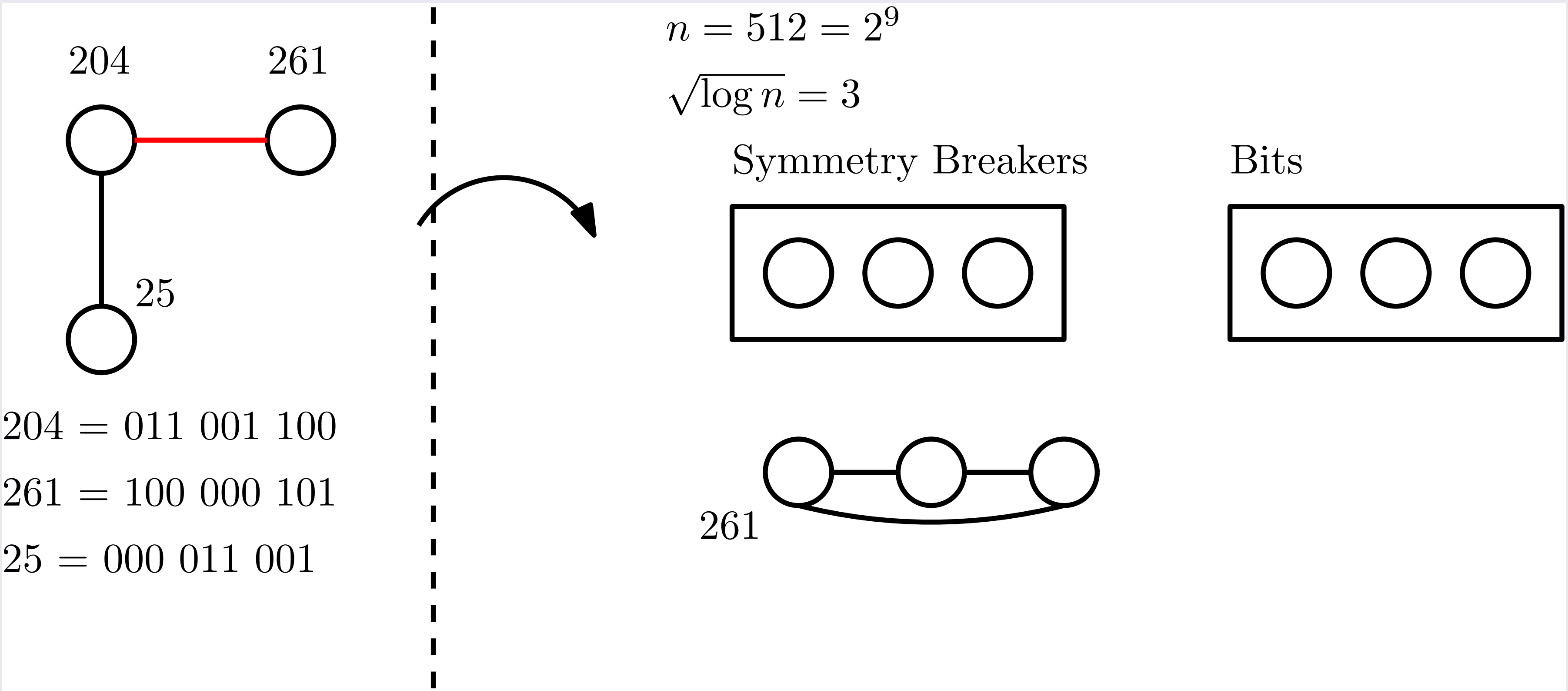
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



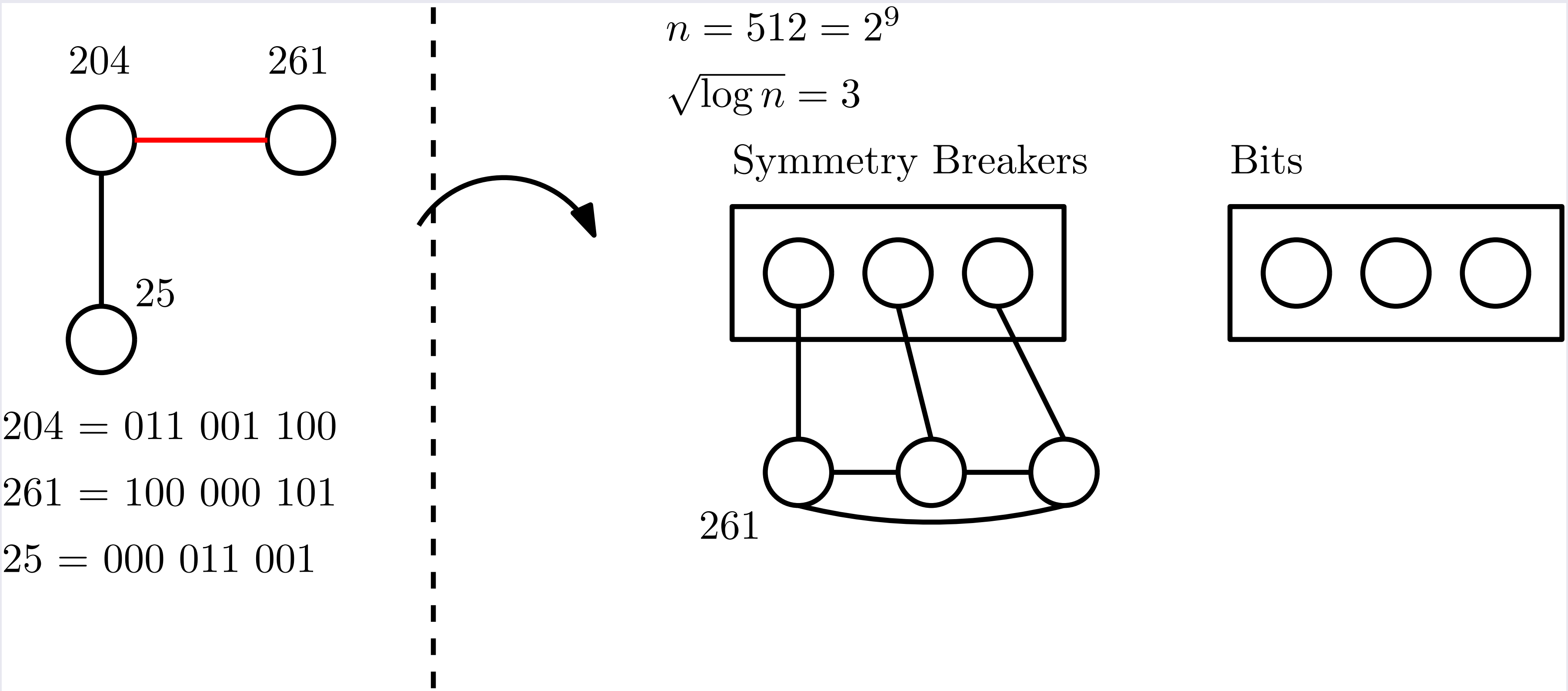
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



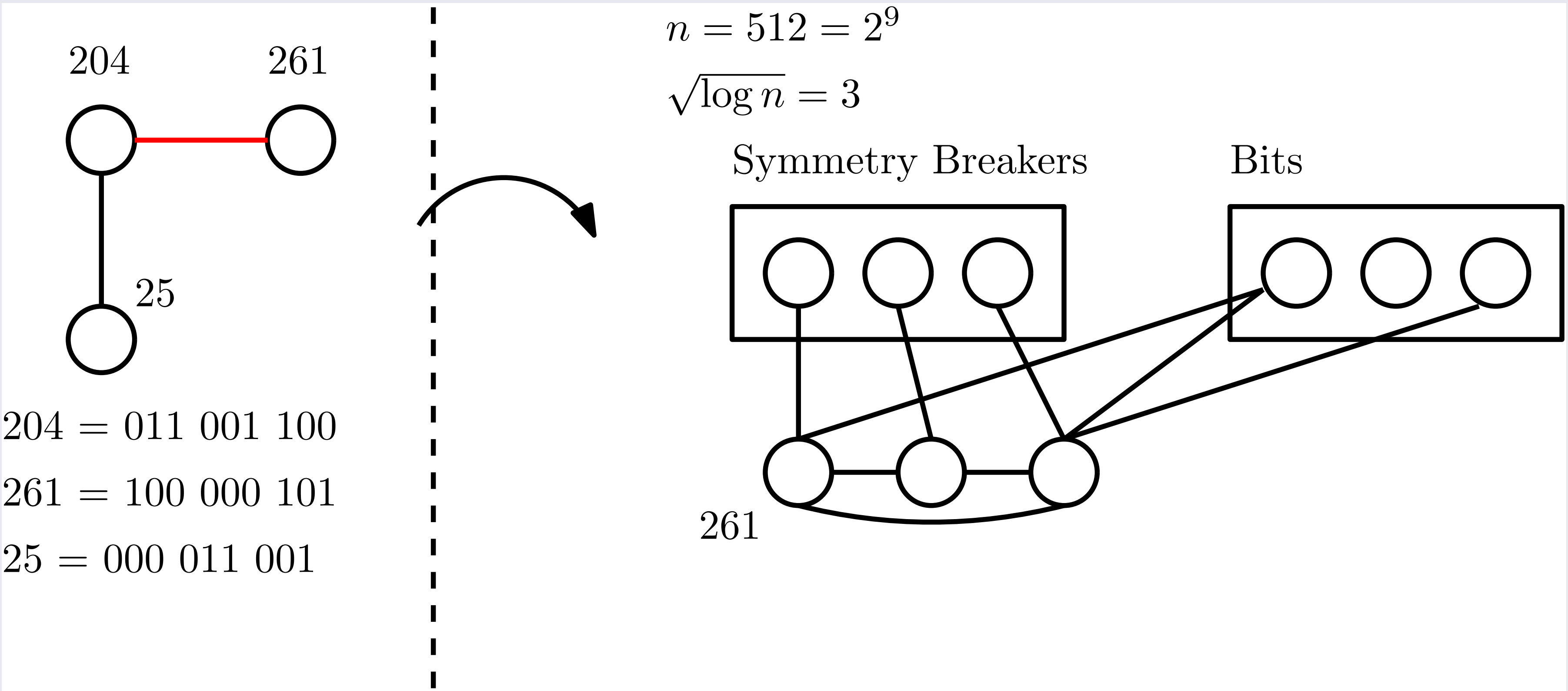
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



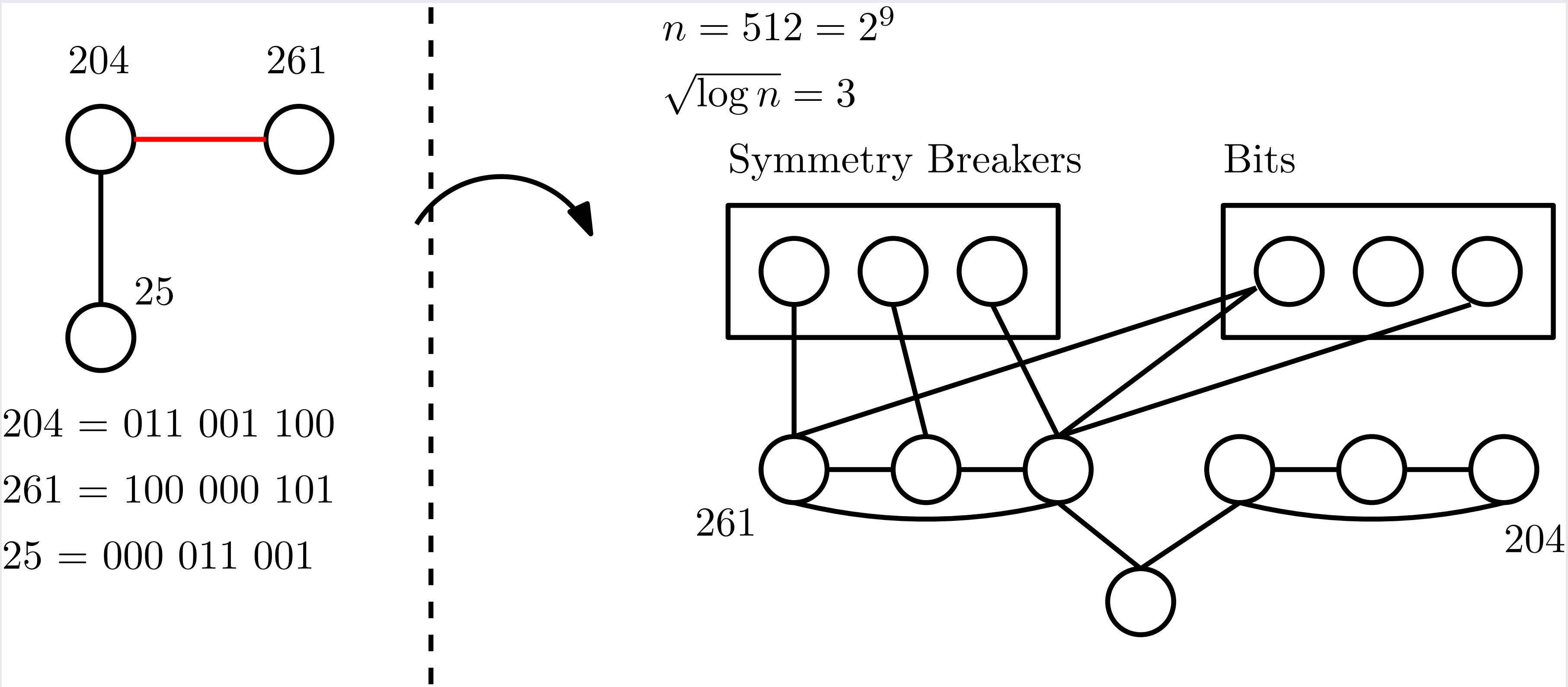
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



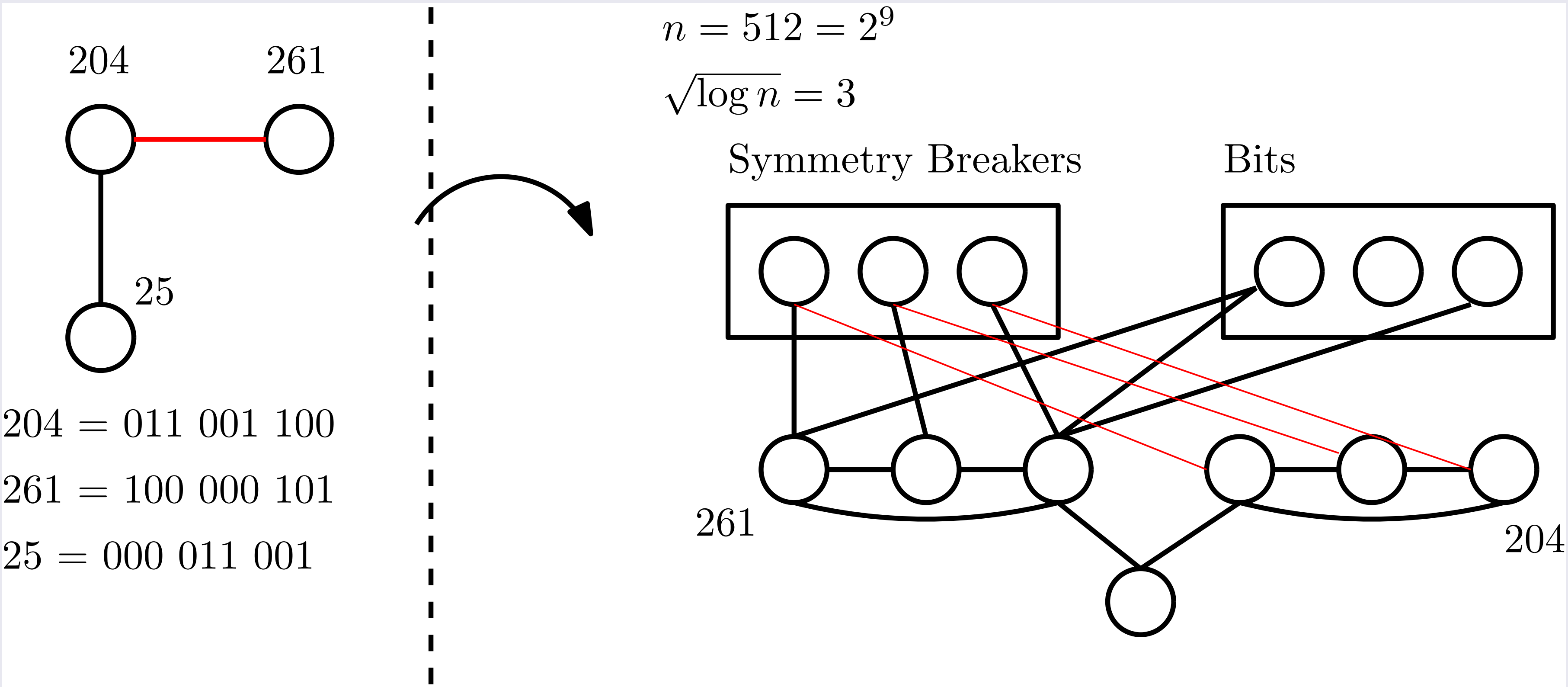
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



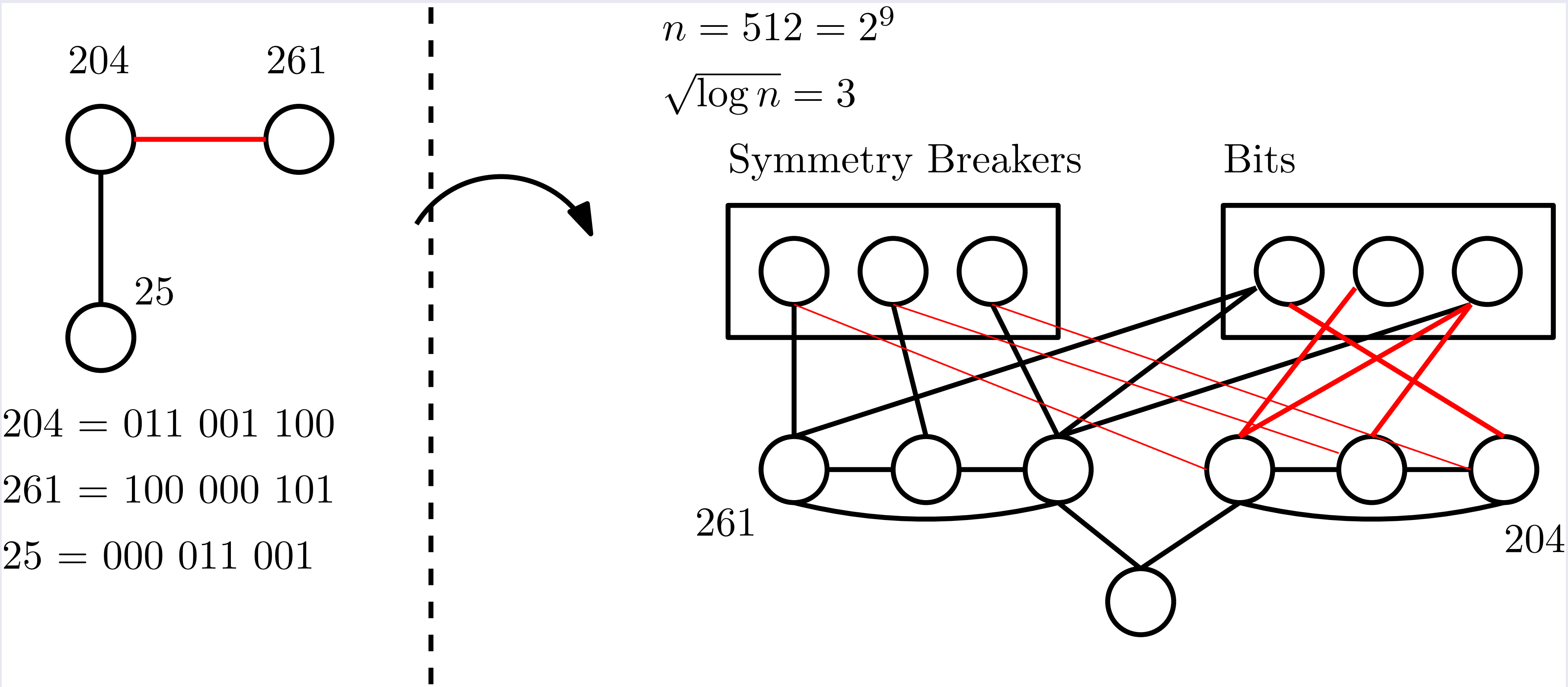
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



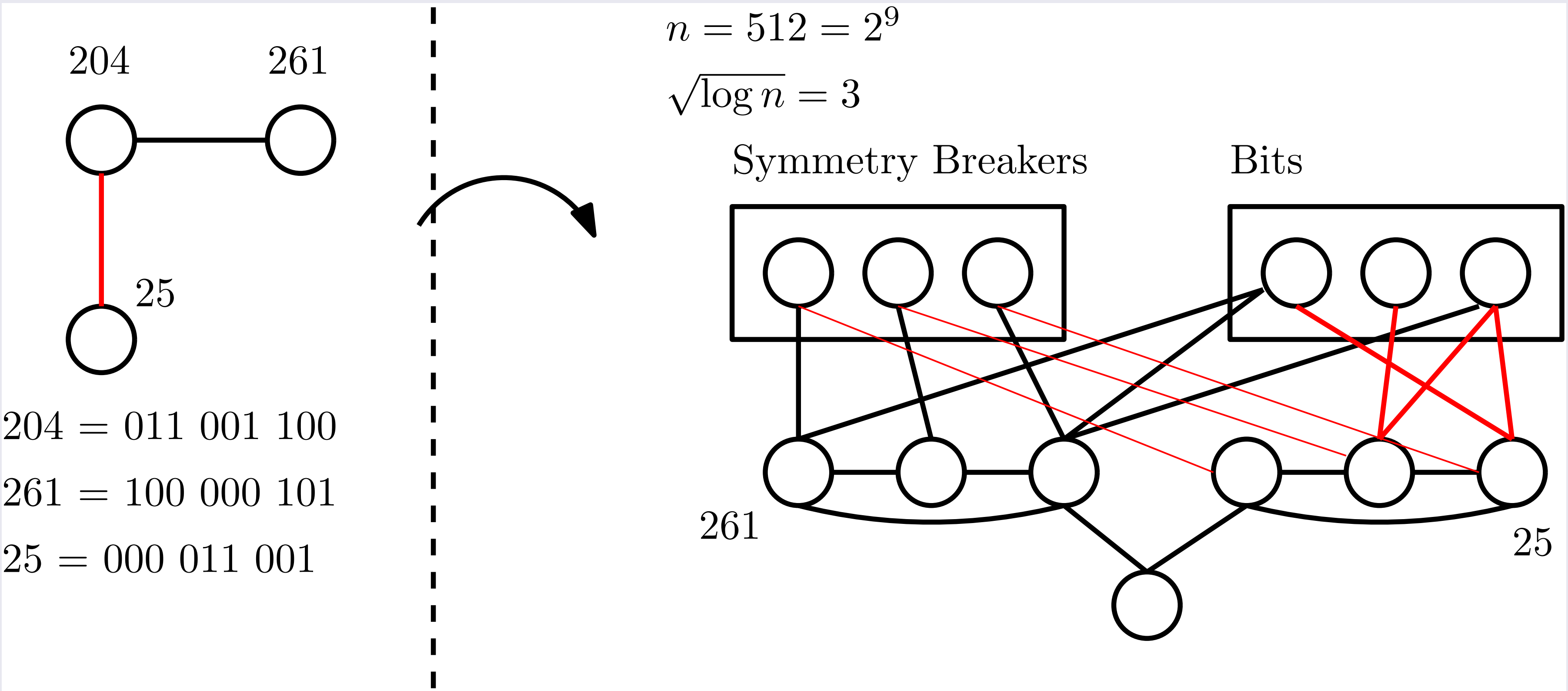
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



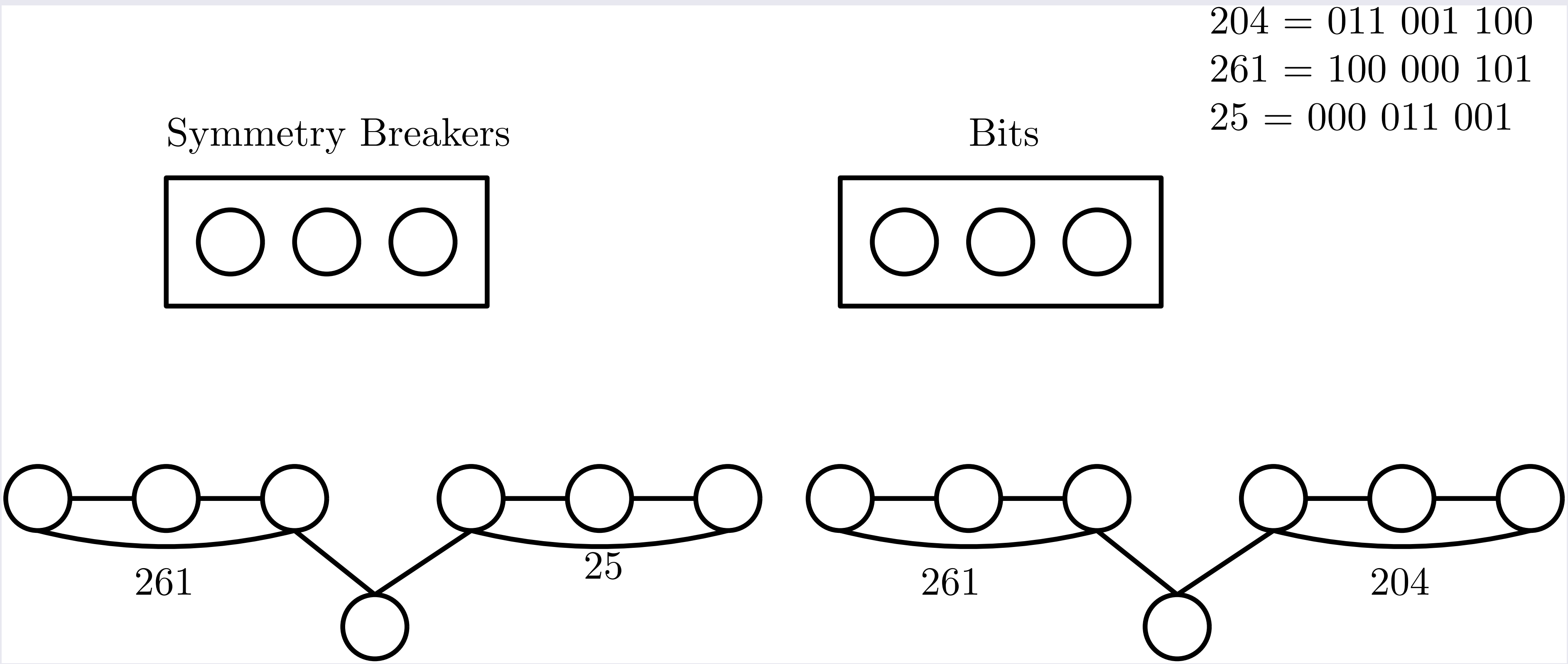
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs



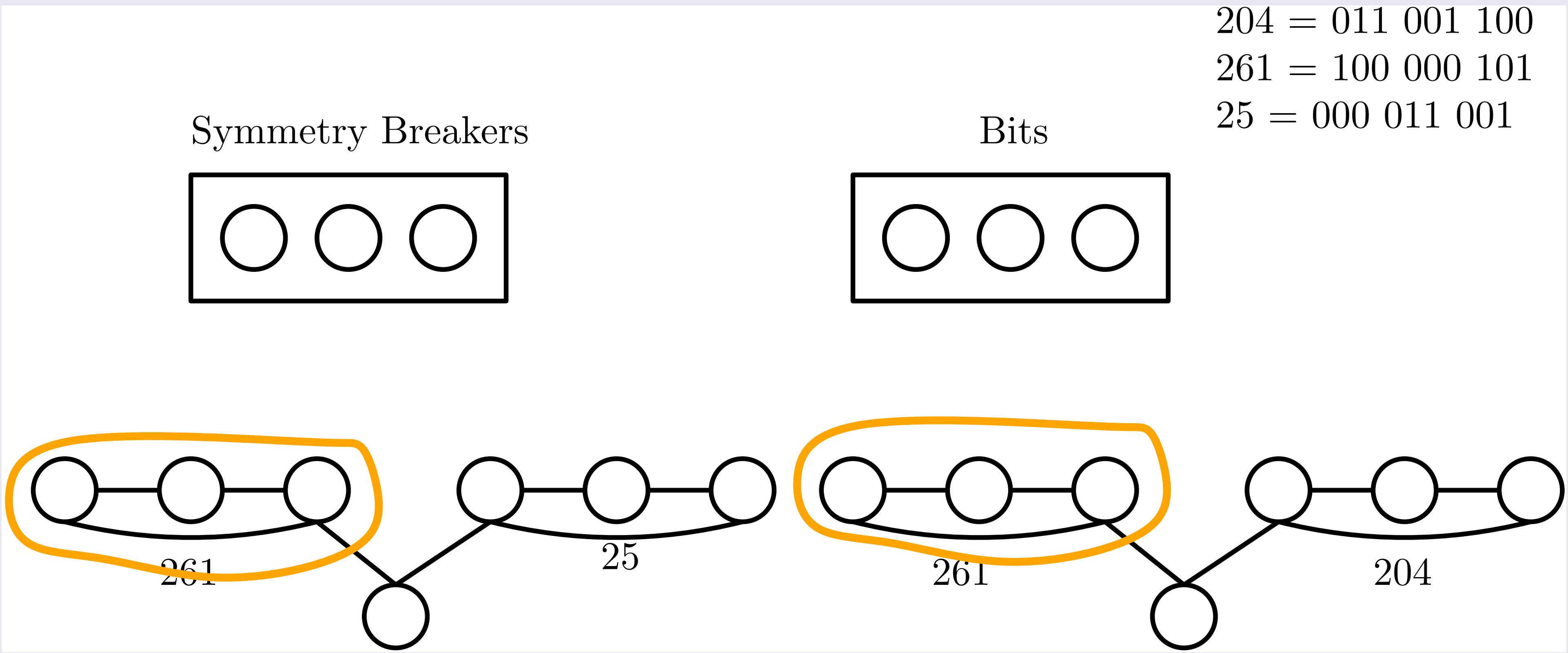
- Separator has $2\sqrt{\log n}$ vertices.
- Each edge of G is represented by a component of $H - S$ made up of two cliques of size $\sqrt{\log n}$.
- Connections from the cliques to S encode indices.

Encoding graphs with simple graphs (cont'd)



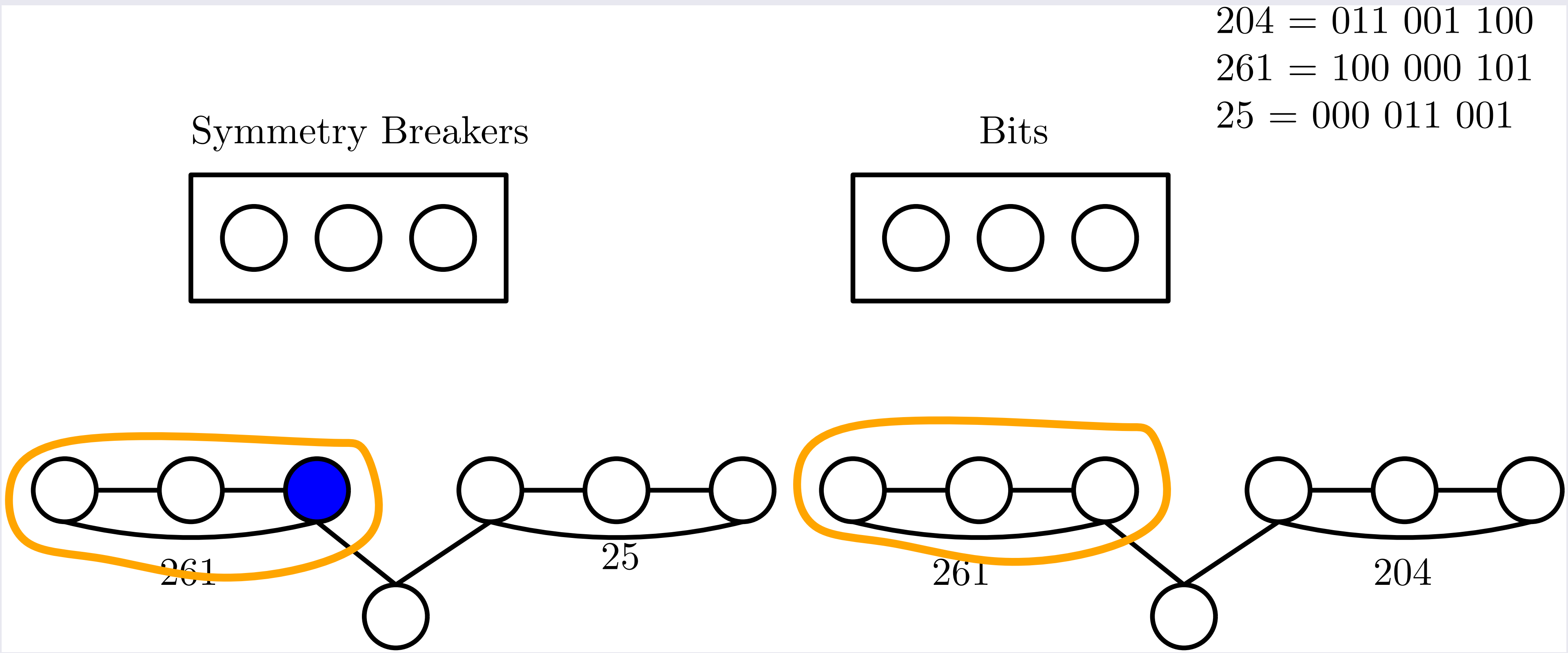
- Goal: a simple FO formula that states: these two edges have a common endpoint.
- Equivalently: these cliques of size $\sqrt{\log n}$ have isomorphic neighbors in S.

Encoding graphs with simple graphs (cont'd)



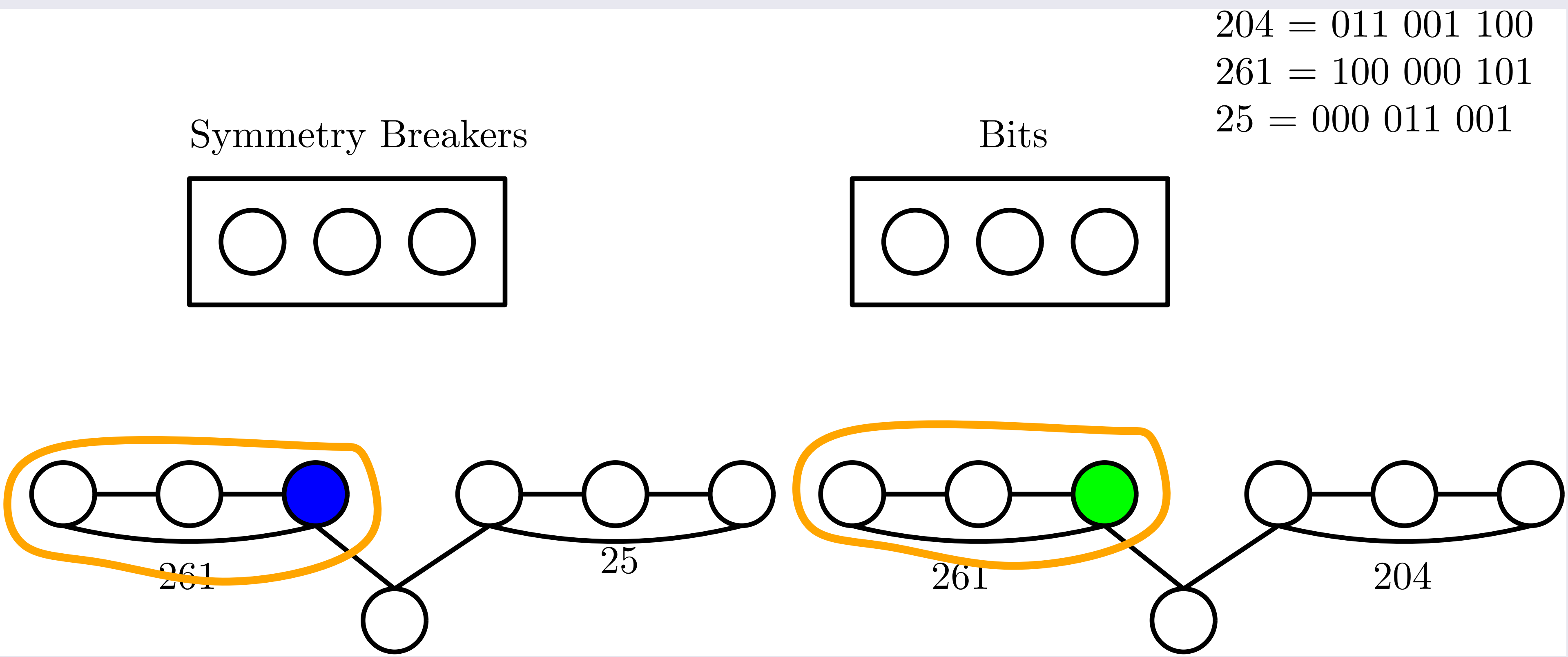
- Goal: a simple FO formula that states: these two edges have a common endpoint.
- Equivalently: these cliques of size $\sqrt{\log n}$ have isomorphic neighbors in S.

Encoding graphs with simple graphs (cont'd)

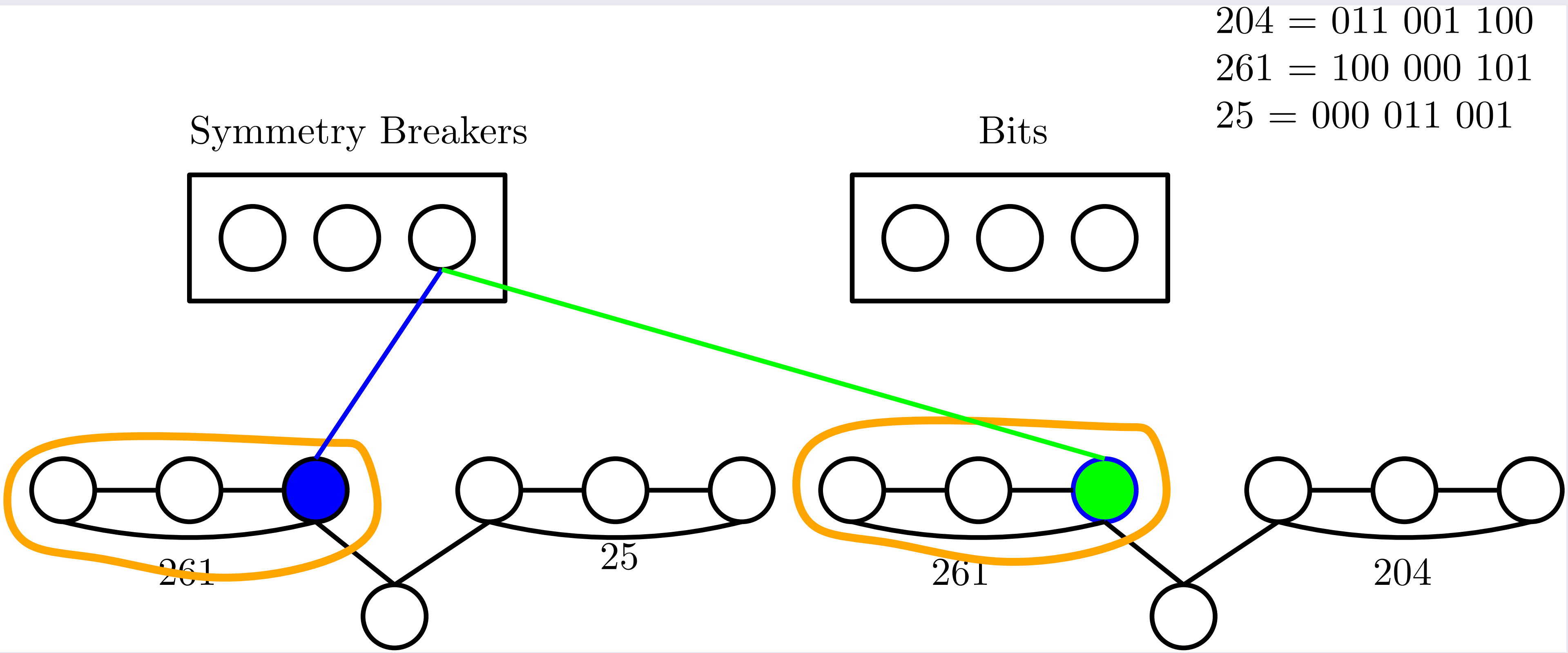


- Goal: a simple FO formula that states: these two edges have a common endpoint.
- Equivalently: these cliques of size $\sqrt{\log n}$ have isomorphic neighbors in S.

Encoding graphs with simple graphs (cont'd)

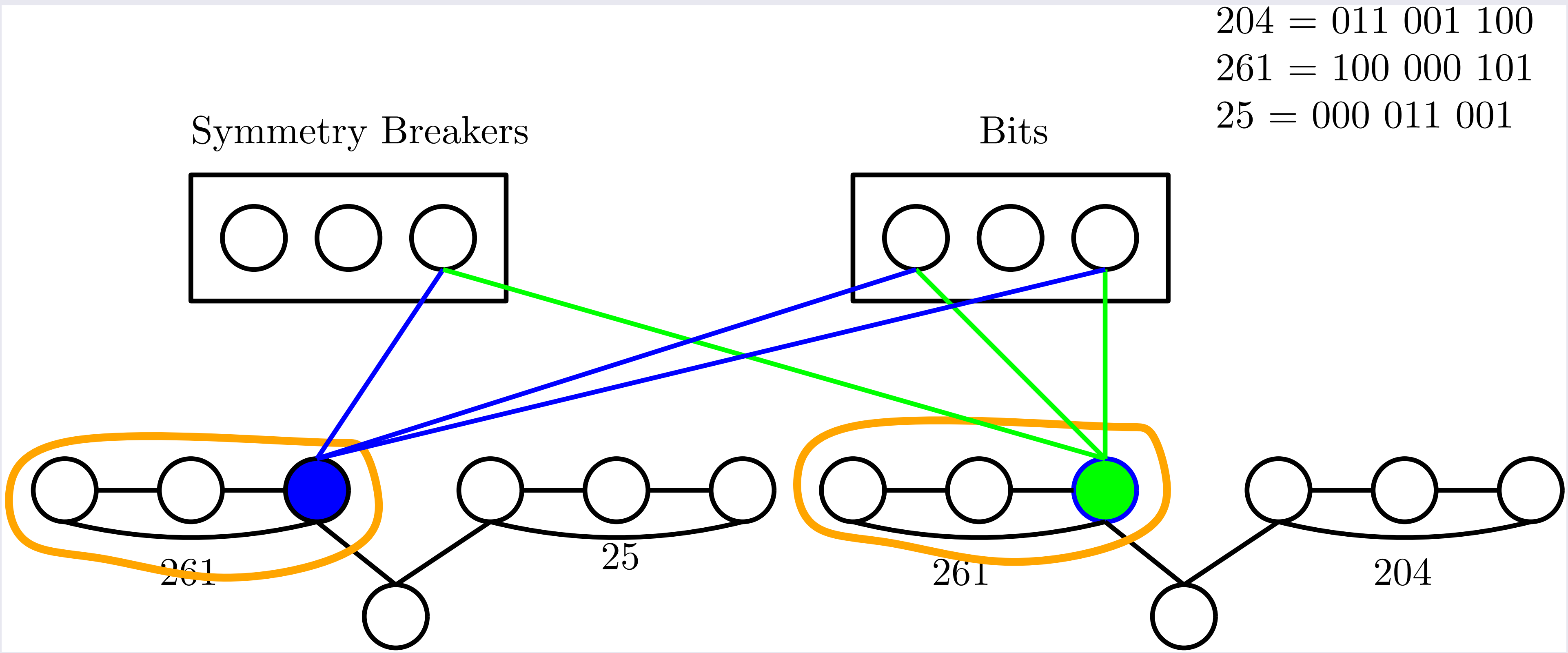


- Goal: a simple FO formula that states: these two edges have a common endpoint.
- Equivalently: these cliques of size $\sqrt{\log n}$ have isomorphic neighbors in S.



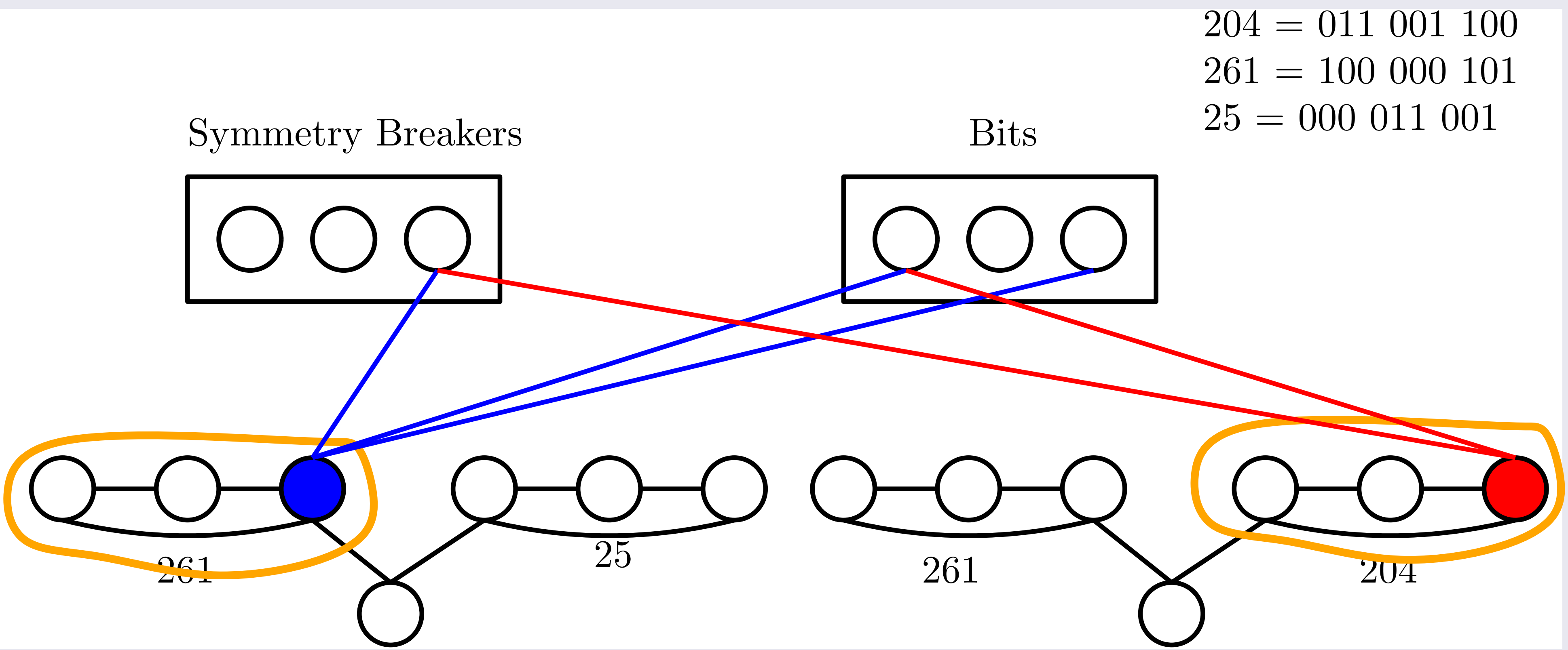
- Goal: a simple FO formula that states: these two edges have a common endpoint.
- Equivalently: these cliques of size $\sqrt{\log n}$ have isomorphic neighbors in S.

Encoding graphs with simple graphs (cont'd)



- Goal: a simple FO formula that states: these two edges have a common endpoint.
- Equivalently: these cliques of size $\sqrt{\log n}$ have isomorphic neighbors in S.

Encoding graphs with simple graphs (cont'd)



- Goal: a simple FO formula that states: these two edges have a common endpoint.
- Equivalently: these cliques of size $\sqrt{\log n}$ have isomorphic neighbors in S.

Putting Things Together

- Can translate G to H so that:
  - $\mathrm{vi}(H) = O(\sqrt{\log n})$
  - Can "read" G in H.
- Is G 3-colorable?
  - Do there exist three sets of vertices partitioning H that represent independent sets in G?
  - MSO-expressible with $q = O(1)$.
  - If $2^{2^{o(\mathrm{vi}^2)}}$ algorithm we have $2^{o(n)}$ algorithm for 3-COLORING!!
- Does G have k-Ind. Set?
  - Do there exist k vertices of H belonging to cliques that represent an independent set of G?
  - FO-expressible with $q = O(k)$.
  - If $2^{o(\mathrm{vi}^2 \cdot q)}$ algorithm we have $2^{o(\log n \cdot k)} = n^{o(k)}$ algorithm for k-CLIQUE!!

Conclusions – Open Problems

Conclusions

- Vertex Integrity “between” vertex cover and tree-depth.
- “(Double-)Exponential in the square” behavior is natural and optimal.

Questions:

- What about MSO_2?
- Other widths between vertex integrity and tree-depth?

Conclusions

- Vertex Integrity “between” vertex cover and tree-depth.
- “(Double-)Exponential in the square” behavior is natural and optimal.

Questions:

- What about MSO_2?
- Other widths between vertex integrity and tree-depth?

Thank you!