Engineering Exact Quasi-Threshold Editing

Lars Gottesbüren, Michael Hamann, Philipp Schoch, Ben Strasser, Dorothea Wagner and Sven Zühlsdorf | June 2020
Quasi-Threshold Graphs

- Trivially perfect graphs
Quasi-Threshold Graphs

- Trivially perfect graphs
- No $P_4$ or $C_4$ as node-induced subgraph

Dense? Sparse? – Both!

Certifying recognition in linear time. [Chu08, BHSW15]
Quasi-Threshold Graphs

- Trivially perfect graphs
- No $P_4$ or $C_4$ as node-induced subgraph

Dense? Sparse? – Both!
Quasi-Threshold Graphs

- Trivially perfect graphs
- No $P_4$ or $C_4$ as node-induced subgraph

Dense? Sparse? – Both!

Certifying recognition in linear time. [Chu08, BHSW15]
Why Quasi-Threshold Graphs?

- Interesting graph class with natural definition
Why Quasi-Threshold Graphs?

- Interesting graph class with natural definition
- Components of quasi-threshold graphs are communities \[\text{[NG13]}\]
Why Quasi-Threshold Graphs?

- Interesting graph class with natural definition
- Components of quasi-threshold graphs are communities [NG13]
- Real world graphs are not quasi-threshold graphs
  \(\rightsquigarrow\) Find quasi-threshold graph with small edge edit distance
Why Quasi-Threshold Graphs?

- Interesting graph class with natural definition
- Components of quasi-threshold graphs are communities \[NG13]\n- Real world graphs are not quasi-threshold graphs
  \implies Find quasi-threshold graph with small edge edit distance
Why Quasi-Threshold Graphs?

- Interesting graph class with natural definition
- Components of quasi-threshold graphs are communities \([\text{NG13}]\)
- Real world graphs are not quasi-threshold graphs
  \(\leadsto\) Find quasi-threshold graph with small edge edit distance
Why Quasi-Threshold Graphs?

- Interesting graph class with natural definition
- Components of quasi-threshold graphs are communities [NG13]
- Real world graphs are not quasi-threshold graphs
  \[\Rightarrow\] Find quasi-threshold graph with small edge edit distance
Quasi-Threshold Editing

Quasi-Threshold Editing Problem

Given a graph $G$ find a quasi-threshold graph with minimum edge editing (insertion + deletion) distance to $G$. 
Quasi-Threshold Editing

Quasi-Threshold Editing Problem

Given a graph $G$ find a quasi-threshold graph with minimum edge editing (insertion + deletion) distance to $G$.

- Is NP-hard
- Is FPT $O(6^k (|V| + |E|))$
- Polynomial kernel exists ($O(k^7)$ vertices)
- Heuristics exist

[NG13] [Cai96] [DP17] [NG13, BHSW15]
# Quasi-Threshold Editing

## Quasi-Threshold Editing Problem
Given a graph $G$ find a quasi-threshold graph with minimum edge editing (insertion + deletion) distance to $G$.

- Is NP-hard \[\text{[NG13]}\]
- Is FPT $O(6^k (|V| + |E|))$ \[\text{[Cai96]}\]
- Polynomial kernel exists ($O(k^7)$ vertices) \[\text{[DP17]}\]
- Heuristics exist \[\text{[NG13, BHSW15]}\]

**Our contribution:**
- Exact algorithms – evaluation of heuristics and exact solutions
- Improved branch-and-bound FPT algorithm and ILP
- For forbidden subgraphs $\mathcal{F}$
- Experimental evaluation for $\{P_4, C_4\}$
Example: $P_3$-free, $k = 3$
Example: $P_3$-free, $k = 3$

\[ \text{−}\{A, B\} \quad \text{−}\{B, C\} \quad +\{A, C\} \]

![Diagram showing a tree with nodes A, B, C, D, E, F, G and operations on edges.](image)

If not: need to search the full tree.

If nothing found at level $k$: impossible with $k$ edits.

Time $O(3^k \cdot \text{poly}(n))$.

Best known: $O(1.62^k + m + n)$ [B¨oc12]
Example: $P_3$-free, $k = 3$
Example: $P_3$-free, $k = 3$
Example: $P_3$-free, $k = 3$

$$G$$

\[-\{A, B\}, \{-B, C\}, +\{A, C\}\]
Example: $P_3$-free, $k = 3$

\[
\begin{align*}
&\{A, B\} \quad \{B, C\} \quad \{A, C\} \\
\end{align*}
\]
Example: $P_3$-free, $k = 3$

\[
-\{A, B\} \quad -\{B, C\} \quad +\{A, C\}
\]

\[
\text{Found solution.}
\]

If not: need to search the full tree.

If nothing found at level $k$: impossible with $k$ edits.

Time $O(3^k \cdot \text{poly}(n))$

Best known: $O(1.62^k + m + n)$ \[B¨oc12\]
Example: $P_3$-free, $k = 3$

$G$

$-\{A, B\} \quad -\{B, C\} \quad +\{A, C\}$

$\Rightarrow$ Found solution.

If not: need to search the full tree.

If nothing found at level $k$: impossible with $k$ edits.

Time $O(3^k \cdot \text{poly}(n))$

Best known: $O(1.62^k + m + n)$ [B¨oc12]
Example: $P_3$-free, $k = 3$
Example: $P_3$-free, $k = 3$
Example: $P_3$-free, $k = 3$
Example: $P_3$-free, $k = 3$

\[ \{A, B\} - \{B, C\} + \{A, C\} \]

\[ \{A, B\} - \{B, G\} + \{A, G\} \]

\[ \{D, C\} - \{C, E\} + \{D, E\} \]

\[ \{A, B\} - \{B, C\} + \{A, C\} \]

\[ \{A, B\} - \{B, G\} + \{A, G\} \]

\[ \{D, C\} - \{C, E\} + \{D, E\} \]
Example: $P_3$-free, $k = 3$

⇒ Found solution.
**Example:** \( P_3 \)-free, \( k = 3 \)

\[
\begin{align*}
\text{G} & \quad \{-\{A, B\}, -\{B, C\}, +\{A, C\}\} \\
\text{A} & \quad \{-\{A, B\}, -\{B, G\}, +\{A, G\}\} \\
\text{B} & \quad \{-\{D, C\}, -\{C, E\}, +\{D, E\}\} \\
\end{align*}
\]

⇒ Found solution. If not: need to search the full tree. If nothing found at level \( k \): impossible with \( k \) edits.
Example: $P_3$-free, $k = 3$

⇒ Found solution. If not: need to search the full tree. If nothing found at level $k$: impossible with $k$ edits.

- Time $O(3^k \cdot \text{poly}(n))$
- Best known: $O(1.62^k + m + n)$

[Böc12]
Branch-and-Bound Algorithm

- Increasing values of $k$ to find exact $k$
  - show impossibility with $k - 1$
  - show solution with $k$
- Blocking: avoid duplicate enumeration [Dam08]
- Bounding: limit explored branches.

$G - \{A, B\} - \{B, C\} + \{A, C\}$

$G - \{A, B\} - \{B, G\} + \{A, G\}$
Branch-and-Bound Algorithm

- Increasing values of $k$ to find exact $k$
  - show impossibility with $k - 1$
  - show solution with $k$
- Blocking: avoid duplicate enumeration [Dam08]
- Bounding: limit explored branches.

Our contribution:
- Good lower bounds
- Heuristic for selecting subgraphs to branch on
Lower Bound

- Implicit conflict graph representation
- Use two-improvements and plateau search
- Incremental updates, exploit blocked edges

[ARW12]
Lower Bound

Implicit conflict graph representation
- Use two-improvements and plateau search
- Incremental updates, exploit blocked edges

[ARW12]
Lower Bound

- Implicit conflict graph representation
- Use two-improvements and plateau search
- Incremental updates, exploit blocked edges

[ARW12]
Lower Bound

Implicit conflict graph representation
- Use two-improvements and plateau search
- Incremental updates, exploit blocked edges

[ARW12]
Lower Bound

- Implicit conflict graph representation
- Use two-improvements and plateau search
- Incremental updates, exploit blocked edges

[ARW12]
Lower Bound

- Implicit conflict graph representation
- Use two-improvements and plateau search
- Incremental updates, exploit blocked edges

Implicit conflict graph representation

\[ \{A, B, C\} - \{B, C\} - \{A, B\} + \{A, C\} \]

Institute of Theoretical Informatics
Group Algorithmics

Gottesbühren, Hamann, Schoch, Strasser, Wagner and Zühlisdorf – Eng. Exact Quasi-Threshold Editing
June 2020
Optimizing Branching Decisions

- Prune early: even before recursion
- First node pair = most likely edited node pair in all solutions
Optimizing Branching Decisions

- Prune early: even before recursion
- First node pair = most likely edited node pair in all solutions

**Idea:** Prefer node pairs that are part of many forbidden subgraphs

![Graph Diagram](image-url)
Integer Linear Programming

- Binary variables $X_{uv} \forall u, v \in V_G$
- $X_{uv} = 1 \iff$ edge $\{u, v\}$ exists

$$\text{minimize} \quad \sum_{\{u, v\} \in E_G} (1 - x_{uv}) + \sum_{\{u, v\} \in \overline{E_G}} x_{uv}$$

subject to

$$\sum_{\{u, v\} \in E_H} (1 - x_{\pi(u)\pi(v)}) + \sum_{\{u, v\} \in \overline{E_H}} x_{\pi(u)\pi(v)} \geq 1$$

$\forall H \in \mathcal{F}, \forall \pi: V_H \hookrightarrow V_G$

- Row generation
- LP relaxation is upper bound for lower bound
Experiments

Implementation:
- \( \{P_4, C_4\}\)-free editing
- Implemented in C++, parallelization using work stealing
- Gurobi for ILP
- https://github.com/kit-algo/fpt-editing

Data:
- 716 of 3964 connected components of COG protein similarity data – remaining need < 20 edits
- 4 node id permutations

Setup:
- 2 · 8 core Intel Xeon E5-2670 (Sandy Bridge), 64 GB RAM
- 1000 seconds time limit
FPT Running Time

![Graph showing running time for FPT-G-F-All and FPT-G-MP-All algorithms.](image)
FPT Running Time

![Graph showing FPT running time with data points for different algorithms: FPT-G-F-All, FPT-G-MP-All, FPT-MD-F-All, and FPT-MD-MP-All.](image)

- **Total Time [s]**
  - 0
  - 500
  - 1000
  - 1500
  - 2000
  - 2500

- **Solved (Graphs × Node Id Permutations)**
  - 0
  - 500
  - 1000
  - 1500
  - 2000

- **Data Points**
  - FPT-G-F-All
  - FPT-G-MP-All
  - FPT-MD-F-All
  - FPT-MD-MP-All
FPT Running Time

![Graph showing FPT running time with different algorithms and permutations]

- **FPT-G-F-All**
- **FPT-G-MP-All**
- **FPT-MD-F-All**
- **FPT-MD-MP-All**
- **FPT-LP-MP-All**

Solved (Graphs × Node Id Permutations) vs Total Time [s]
ILP Variants

Total Time [s]

Solved (Graphs × Node Id Permutations)

ILP-B
ILP-S
ILP-S-R
ILP-S-R-C4

Gottesbüren, Hamann, Schoch, Strasser, Wagner and Zühlisdorf – Eng. Exact Quasi-Threshold Editing
June 2020
FPT vs. ILP

![Graph showing FPT-LS-MP and ILP-S-R-C4 performance comparison across total time and solved graphs.](image-url)
FPT vs. ILP

The diagram compares the performance of FPT and ILP algorithms in terms of solved graphs and node id permutations. The X-axis represents the total time in seconds, while the Y-axis shows the number of solved graphs and node id permutations. The algorithms compared are:

- FPT-LS-MP
- FPT-LS-MP-MT
- ILP-S-R-C4
- ILP-S-R-C4-MT

The graph illustrates how different algorithms perform under various time constraints, indicating which algorithms are more efficient for solving the problem at hand.
FPT vs. ILP
Comparison with QTM Heuristic

Not shown: Outlier at $k = 64$ where QTM needs 202 edits
Summary

- Carefully engineered branch and bound algorithm beats Gurobi

Future work:
- Adapt for edit costs
- Evaluate for other forbidden subgraphs
Diogo V. Andrade, Mauricio G. C. Resende, and Renato F. Werneck.
Fast local search for the maximum independent set problem.

Ulrik Brandes, Michael Hamann, Ben Strasser, and Dorothea Wagner.
Fast Quasi-Threshold Editing.

Sebastian Böcker.
A golden ratio parameterized algorithm for Cluster Editing.

Leizhen Cai.
Fixed-parameter tractability of graph modification problems for hereditary properties.

Frank Pok Man Chu.
A simple linear time certifying LBFS-based algorithm for recognizing trivially perfect graphs and their complements.

Peter Damaschke.
Fixed-parameter enumerability of cluster editing and related problems.

Pål Grønås Drange and Michał Pilipczuk.
A Polynomial Kernel for Trivially Perfect Editing.
James Nastos and Yong Gao.
Familial groups in social networks.
Number of Solutions

![Graph showing the number of solutions vs. k]