Linear Kernels on Sparse Graph Classes

Somnath Sikdar

Theoretical Computer Science,
RWTH Aachen University, Germany.

Outline

Tackling NP-Hardness

Many interesting optimization problems are NP-hard.

Traditional techniques

- Heuristics (no performance guarantees)
- Polynomial-time approximation algorithms (approximate solutions)

New weapons (more than 20 years old)

- Moderately exponential algorithms (exact solutions, effective for small inputs)
- Parameterized complexity (exact solutions, effective for small parameters)

Parameterized Problems

Two components

- input
- parameter (fixed by the algorithm designer)

Example parameterizations

- solution size: Does graph G have a vertex cover of size k, parameter k?
- structural measure: Does graph G have a dominating set of size k, parameter treewidth of G?
- excess solution size: Given a CNF formula with m clauses, is there an assignment that satisfies $m/2+k$ clauses, parameter k?

Fixed-Parameter Tractability

Running times are **measured wrt both x and k**.

Definition

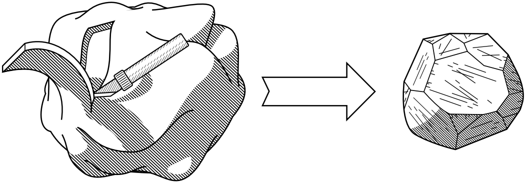
A parameterized problem is **fixed-parameter tractable** if there is an algorithm with running time $O(f(k) \cdot |x|^c)$, where f is a function of k alone and c is a constant.

Example running times

- Vertex Cover (parameter: solution size): $O(1.2738^k \cdot k \cdot |V(G)|)$ [Chen, Kanj, Xia, 2010.]
- Dominating Set (parameter: treewidth): $O(3^{\text{tw}} \cdot |V(G)|^{O(1)})$ [van Rooij, Bodlaender, Rossmanith, 2009.]
- Max Sat (parameter: excess above $m/2$): $O(\phi^{6k} k + |F|)$, $\phi =$ golden ratio [Mahajan and Raman, 1999.]

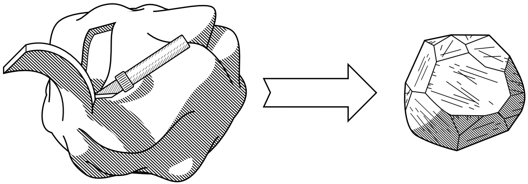
A closely related concept: **kernelization algorithm**.

Kernelization



A kernelization algorithm strips away easy parts of the input and exposes the core (the kernel).

Kernelization



A kernelization algorithm strips away easy parts of the input and exposes the core (the kernel).

Definition

A **kernelization algorithm** transforms an instance (x, k) into an equivalent instance (x', k') in time polynomial in $|x| + k$ s.t.

- $|x'|, k' \leq f(k)$, for some function f.

The function f is called the **size of the kernel**.

Example Kernelization

Vertex Cover: Does G have a vertex cover of size at most k? (Parameter: k).

Observation: Any vertex of degree at least $k+1$ must be in a solution.

Kernelization Algorithm

- Delete all vertices of degree at least $k+1$ from the graph. If the number of such vertices is $> k$, report no-instance.

The remaining graph has at most $O(k^2)$ vertices.

Kernelization and Fixed-Parameter Tractability

Folklore

A problem is fixed-parameter tractable (FPT) iff it has a kernelization algorithm.

The kernel size obtained from a fixed-parameter algorithm is **usually exponential or worse**.

Goal

To obtain polynomial (or even better, linear) kernels.

Basic Technique

- devise **reduction rules** that preserve **equivalence** of instances;
- when reduction rules cannot be applied anymore, show that the resulting instance has small size.

Outline

Why Sparse Graph Classes?

Many hard problems are fixed-parameter tractable on sparse graphs.

- Dominating Set on bounded-genus graphs.
- Independent Set on planar graphs.
- MSO-definable problems on bounded-treewidth graphs.

Meta-results showed that a large class of problems admit linear kernels on certain sparse classes.

No polynomial kernels on general graphs for many problems

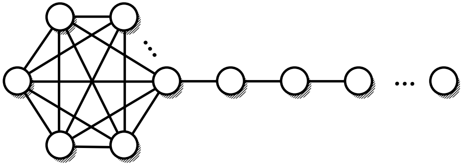
- In particular: "connectivity" problems: Longest Path, Disjoint Paths, Connected Vertex Cover, Steiner Tree, . . .

What Kind of Sparseness?

Requesting a linear number of edges not particularly useful.

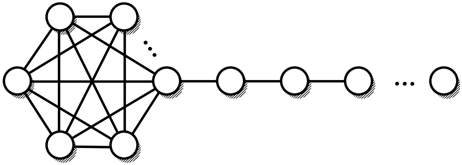
What Kind of Sparseness?

Requesting a linear number of edges not particularly useful.



What Kind of Sparseness?

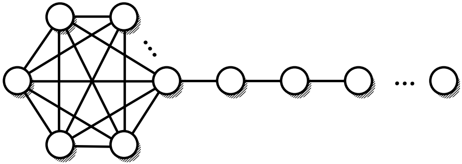
Requesting a linear number of edges not particularly useful.



We need graph classes that are uniformly sparse.

What Kind of Sparseness?

Requesting a linear number of edges not particularly useful.



We need graph classes that are uniformly sparse.

Definition

A graph class $\mathcal{C}$ is **d-degenerate** if for every $G \in \mathcal{C}$, every subgraph of G contains a vertex of degree $\leqslant d$.

Degenerate Graphs

Definition

A graph class $\mathcal{C}$ is **d-degenerate** if for every $G \in \mathcal{C}$, every subgraph of G contains a vertex of degree $\leqslant d$.

Degenerate Graphs

Definition

A graph class $\mathcal{C}$ is **d-degenerate** if for every $G \in \mathcal{C}$, every subgraph of G contains a vertex of degree $\leqslant d$.

Equivalent characterizations

- Vertices can be ordered s.t. every vertex has at most d neighbours to its right.
- Edges can be oriented s.t. every vertex has out-degree at most d.

Degenerate Graphs

Definition

A graph class $\mathcal{C}$ is d-**degenerate** if for every $G \in \mathcal{C}$, every subgraph of G contains a vertex of degree $\leqslant d$.

Equivalent characterizations

- Vertices can be ordered s.t. every vertex has at most d neighbours to its right.
- Edges can be oriented s.t. every vertex has out-degree at most d.

Useful properties

- $|E(G)| \leqslant d \cdot |V(G)|$, therefore average degree $\leqslant 2d$.
- $\chi(G) \leq d+1$ and $\omega(G) \leq d+1$.
- At most $2^d \cdot |V(G)|$ cliques.
- Hereditary.

Problems with Degeneracy

Degeneracy is a good start, but not strong enough for general results:

> *Any graph can be made degenerate by subdividing its edges a lot.*

Problems with Degeneracy

Degeneracy is a good start, but not strong enough for general results:

> *Any graph can be made degenerate by subdividing its edges a lot.*

Problems such as Feedback Vertex Set, Treewidth are invariant under edge subdivisions.

Problems with Degeneracy

Degeneracy is a good start, but not strong enough for general results:

> *Any graph can be made degenerate by subdividing its edges a lot.*

Problems such as Feedback Vertex Set, Treewidth are invariant under edge subdivisions.

Degeneracy does not seem to provide a good handle to solve problems.

Problems with Degeneracy

Degeneracy is a good start, but not strong enough for general results:

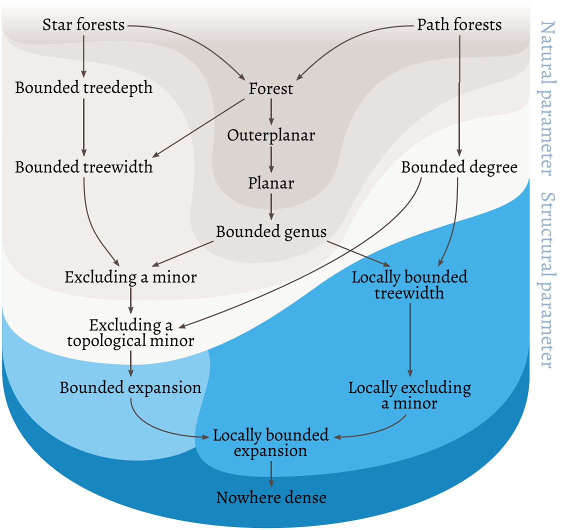
> *Any graph can be made degenerate by subdividing its edges a lot.*

Problems such as Feedback Vertex Set, Treewidth are invariant under edge subdivisions.

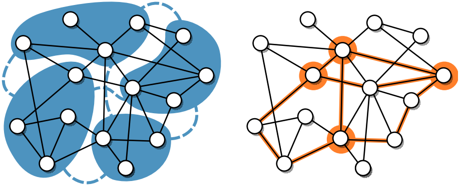
Degeneracy does not seem to provide a good handle to solve problems.

We need structurally sparse classes.

Hierarchy of Sparse Graphs



Minors



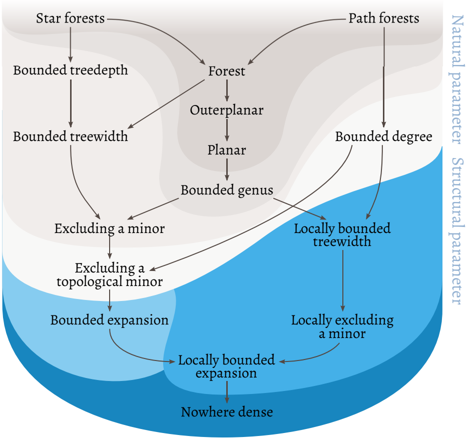
- Minor: take subgraph, contract vertex sets inducing connected subgraphs (branch sets).
- Topological minor: take subgraph, contract vertex-disjoint two-paths between nail vertices.
- Sparse $\Rightarrow$ excludes a fixed graph as a (topological) minor.

Overview of meta-results

Linear kernels in structurally sparse classes

- Framework for planar graphs [Guo and Niedermeier: *Linear problem kernels for NP-hard problems on planar graphs*.]
- Meta-result for graphs of bounded genus [Bodlaender, Fomin, Lokshtanov, Penninkx, Saurabh and Thilikos: *(Meta) Kernelization*.]
- Meta-result for graphs excluding a fixed graph as a minor [Fomin, Lokshtanov, Saurabh and Thilikos: *Bidimensionality and kernels*.]
- Meta-results for graphs excluding a fixed graph as a topological minor [Kim, Langer, Paul, Reidl, Rossmanith, Sau, and S.: *Linear kernels and single-exponential algorithms via protrusion decompositions*.]

Trade-off: sparseness vs. problem requirements



Outline

Main Theorem

Theorem

Let H be a fixed graph. A parameterized graph problem that has

1. *finite integer index, and*
2. *is treewidth-bounding,*

both on the class of H-topological-minor-free graphs admits a linear kernel on this graph class.

The Undefined Terms

A **parameterized graph problem** Π is a set of pairs (G, k), where G is a graph and k a non-negative integer.

Π **treewidth-bounding**: for some constants c, t, yes-instances (G, k) have a vertex subset $X \subseteq V(G)$ s.t.

$|X| \leq c \cdot k$ and $\mathbf{tw}\,(G - X) \leq t$.

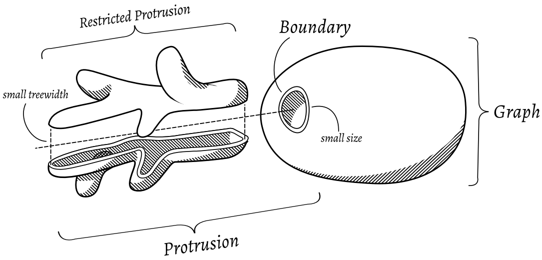
Finite integer index allows us to apply a generic and powerful reduction rule: **protrusion reduction rule**.

Protrusions

Finite integer index allows us to apply the **protrusion reduction rule**.

- allows us to replace a piece of the graph (satisfying certain properties) by a canonical structure.

Protrusions

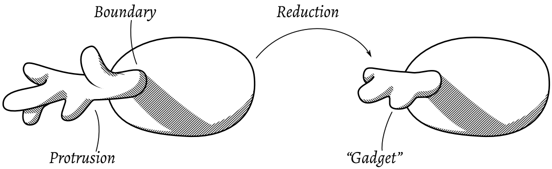


Definition

$W \subseteq V(G)$ is a **t-protrusion** if

1. **(small boundary)** $|N(W) \setminus W| \leq t$,
2. **(small treewidth)** $\mathbf{tw}\,(G[W]) \leq t$.

Protrusion Reduction Rule



Replace a large protrusion by a smaller canonical structure.

Finite integer index allows t-protrusions to be replaced by a member of a finite set $\mathcal{R}_t$.

- $|\mathcal{R}_t|$ depends on t, the problem (and the graph class).
- We assume that for each t, the set $\mathcal{R}_t$ is given.
- Non-uniform algorithms.

Properties of H-Topological-Minor-Free Graphs

Topological minor: select a subgraph and contract vertex-disjoint degree-two paths.

For a graph G excluding H as a topological minor,

- not interested in structure of H, but its size $r = |H|$.
- in particular: K_r not a topological minor of G.

Properties of H-Topological-Minor-Free Graphs

Topological minor: select a subgraph and contract vertex-disjoint degree-two paths.

For a graph G excluding H as a topological minor,

- not interested in structure of H, but its size $r = |H|$.
- in particular: K_r not a topological minor of G.

Important properties

1. $|E(G)| \leq \frac{1}{2}\beta r^2 |V(G)|$ (for some $\beta < 10$).
2. no. of cliques $\leq 2^{\tau r \log r}|V(G)|$ (for some $\tau < 4.51$).
3. Closed under taking topological minors.

Outline

Main Theorem and Kernelization Algorithm

Theorem

Let H be a fixed graph. A parameterized graph problem that has

1. *finite integer index, and*
2. *is treewidth-t bounding,*

both on the class of H-topological-minor-free graphs admits a linear kernel on this graph class.

Kernelization Algorithm

1. Replace all $(2t + r)$-protrusions by their representatives from $\mathcal{R}_{2t+r}$, where $r = |V(H)|$.

Time Taken $O(n^{O(t+r)})$.

On Treewidth Boundedness

Definition

A problem is **treewidth bounding** if for some constants c, t, yes-instances (G, k) have a vertex subset $X \subseteq V(G)$ s.t.

1. $|X| \leq c \cdot k$;
2. **tw** $(G - X) \leq t$.

On Treewidth Boundedness

Definition

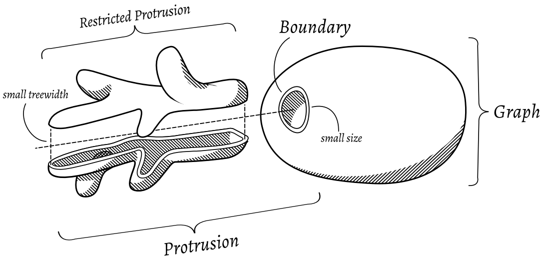
A problem is **treewidth bounding** if for some constants c, t, yes-instances (G, k) have a vertex subset $X \subseteq V(G)$ s.t.

1. $|X| \leq c \cdot k$;
2. $\mathbf{tw}\,(G - X) \leq t$.

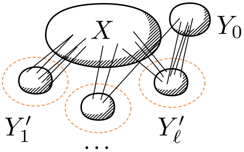
S usually is the solution set.

- Vertex Cover, Feedback Vertex Set in general graphs.
- Chordal Vertex Deletion in graphs with bounded clique-size.

Proof Sketch by a Picture



Using sparseness



- $|Y_0| = O(|X|)$ and each cluster Y'_i, $1 \leq i \leq \ell$, has constant size.
- Contract each component of Y'_i to an edge in $X \cup Y_0$, doing this for as many components as possible.
- The neighborhood in $X \cup Y_0$ of each cluster that remains is a clique.
- The total number of cliques is $O(|X| + |Y_0|)$ and hence at most these many components were contracted.
- $\sum_{i=1}^{l} |Y'_i| = O(|X \cup Y_0|)$.

The Proof: Main Components

Finite integer index

- Allows use of the protrusion reduction rule.

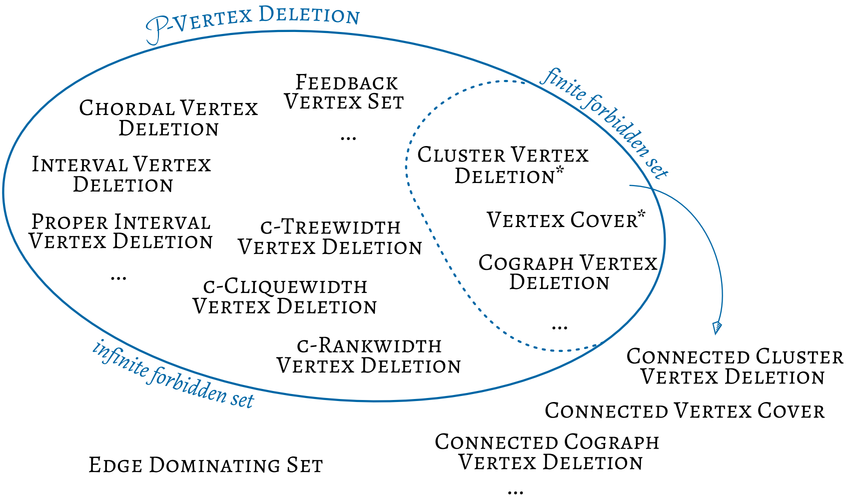
Treewidth modulation

- Allows a convenient decomposition into clusters each of which is a protrusion.

Sparsity and protrusion reduction

- Allows the total size of all clusters to be bounded.

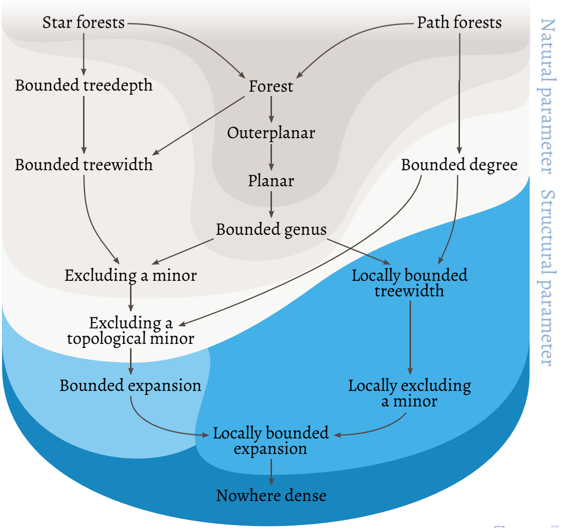
Examples



$\mathcal{P}$-Vertex Deletion
Feedback Vertex Set
...
Chordal Vertex Deletion
Interval Vertex Deletion
Proper Interval Vertex Deletion
...
c-Treewidth Vertex Deletion
c-Cliquewidth Vertex Deletion
c-Rankwidth Vertex Deletion
finite forbidden set
Cluster Vertex Deletion*
Vertex Cover*
Cograph Vertex Deletion
...
infinite forbidden set
Connected Cluster Vertex Deletion
Connected Vertex Cover
Connected Cograph Vertex Deletion
...
Edge Dominating Set

Outline

Overview of Sparse Graph Classes (Again!)



Beyond H-Minor-Free Graphs

Theorem

Problems that have **finite integer index** *on graphs of constant treedepth admit linear kernels on graphs of* **bounded expansion** *if parameterized by a* **modulator to constant treedepth**.

Beyond H-Minor-Free Graphs

Theorem

Problems that have **finite integer index** *on graphs of constant treedepth admit linear kernels on graphs of* **bounded expansion** *if parameterized by a* **modulator to constant treedepth**.

- Kernelization algorithm runs in linear time.
- Quadratic kernels on graphs of **locally bounded expansion**.
- Polynomial kernels on **nowhere dense graphs**.

Consequences

The problems...

Dominating Set, Connected Dominating Set, r-Dominating Set, Efficient Dominating Set, Connected Vertex Cover, (Connected) Vertex Cover, Hamiltonian Path/Cycle, 3-Colorability, Independent Set, Feedback Vertex Set, Edge Dominating Set, Induced Matching, Chordal Vertex Deletion, Interval Vertex Deletion, Odd Cycle Transversal, Induced d-Degree Subgraph, Min Leaf Spanning Tree, Max Full Degree Spanning Tree, Longest Path/Cycle, Exact s, t-Path, Exact Cycle, Treewidth, Pathwidth

...parameterized by a treedepth-modulator have ...

- ...linear kernels on graphs of bounded expansion.
- ...quadratic kernels on graphs of locally bounded expansion.
- ...polynomial kernels on nowhere-dense graphs.

Outline

Kernelization Landscape for Sparse Graph Classes

Our Interpretation

- Up until topo-minor-free graphs, treewidth boundedness seems to the main ingredient of the meta-kernel results.
- Beyond this, structural parameters are required (for a reason not discussed here).

Open Problems

- Which problems admit linear kernels beyond topo-minor-free graphs (natural parameter)?

Thank You!