Searching and generating sparse (sub)graphs

Bence Deák

Eötvös Loránd University
Supervisor: Péter Madarasi

Project work III.
8th January, 2026

Let $k, \ell \in \mathbb{N}$, $\ell < 2k$.

Let $k, \ell \in \mathbb{N}$, $\ell < 2k$.

Definition

A graph $G = (V, E)$ is (k, ℓ)-**sparse** if each vertex set $X \subseteq V$ induces at most $\max\{0, k|X| - \ell\}$ edges.

The notion of (k, ℓ)-sparsity

Let $k, \ell \in \mathbb{N}$, $\ell < 2k$.

Definition

A graph $G = (V, E)$ is (k, ℓ)-**sparse** if each vertex set $X \subseteq V$ induces at most $\max\{0, k|X| - \ell\}$ edges.

Definition

A (k, ℓ)-sparse graph $G = (V, E)$ is (k, ℓ)-**tight** if $|E| = k|V| - \ell$.

The notion of (k, ℓ)-sparsity

Let $k, \ell \in \mathbb{N}$, $\ell < 2k$.

Definition

A graph $G = (V, E)$ is (k, ℓ)-**sparse** if each vertex set $X \subseteq V$ induces at most $\max\{0, k|X| - \ell\}$ edges.

Definition

A (k, ℓ)-sparse graph $G = (V, E)$ is (k, ℓ)-**tight** if $|E| = k|V| - \ell$.

Example (Nash-Williams, 1961)

G can be partitioned into k forests $\Leftrightarrow$ G is (k, k)-sparse.

Let $k, \ell \in \mathbb{N}$, $\ell < 2k$.

Definition

A graph $G = (V, E)$ is (k, ℓ)-**sparse** if each vertex set $X \subseteq V$ induces at most $\max\{0, k|X| - \ell\}$ edges.

Definition

A (k, ℓ)-sparse graph $G = (V, E)$ is (k, ℓ)-**tight** if $|E| = k|V| - \ell$.

Example (Nash-Williams, 1961)

G can be partitioned into k forests $\Leftrightarrow$ G is (k, k)-sparse.

Example (Laman, 1970)

G is minimally rigid in the plane $\Leftrightarrow$ G is $(2, 3)$-tight.

Problem (maximum-weight (k, ℓ)-sparse subgraph)

Input: *A graph* $G = (V, E)$ *with a weight function* $w : E \to \mathbb{R}_+$.

Output: $F \subseteq E$ *such that* $H = (V, F)$ *is sparse and* $w(F) = \sum_{e \in F} w(e)$ *is maximal.*

The considered problems

Problem (maximum-weight (k, ℓ)-sparse subgraph)

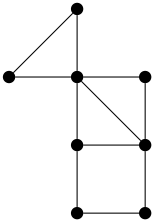
Input: *A graph* $G = (V, E)$ *with a weight function* $w : E \to \mathbb{R}_+$.

Output: $F \subseteq E$ *such that* $H = (V, F)$ *is sparse and* $w(F) = \sum_{e \in F} w(e)$ *is maximal.*

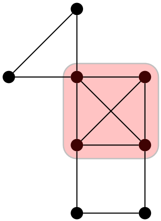
Problem (recognizing (k, ℓ)-sparse graphs)

Input: *A graph* $G = (V, E)$.

Output: `true` *if* G *is sparse, otherwise* `false`.



A $(2, 3)$-sparse graph



A graph that is not $(2, 3)$-sparse

Optimization (past work)

Optimization (past work)

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).

Optimization (past work)

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).
- The improvement turned out to be wrong (Mihálykó, 2022; Madarasi, Matúz, 2023).

Optimization (past work)

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).
- The improvement turned out to be wrong (Mihálykó, 2022; Madarasi, Matúz, 2023).

Question

Is there a quadratic-time algorithm for the problem?

Optimization (past work)

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).
- The improvement turned out to be wrong (Mihálykó, 2022; Madarasi, Matúz, 2023).

Question

Is there a quadratic-time algorithm for the problem?

Answer: yes!

- Maintain the (k, ℓ)-components efficiently $\Rightarrow$ reject edges in $O(1)$.

Optimization (past work)

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).
- The improvement turned out to be wrong (Mihálykó, 2022; Madarasi, Matúz, 2023).

Question

Is there a quadratic-time algorithm for the problem?

Answer: yes!

- Maintain the (k, ℓ)-components efficiently $\Rightarrow$ reject edges in $O(1)$.
- Analysis: components can intersect, but "not too much" $\Rightarrow O(n^2 + m)$ can be achieved.

Optimization (past work)

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).
- The improvement turned out to be wrong (Mihálykó, 2022; Madarasi, Matúz, 2023).

Question

Is there a quadratic-time algorithm for the problem?

Answer: yes!

- Maintain the (k, ℓ)-components efficiently $\Rightarrow$ reject edges in $O(1)$.
- Analysis: components can intersect, but "not too much" $\Rightarrow$ $O(n^2 + m)$ can be achieved.

Applications:

- Minimum-cost rigid spanning subgraph problem.

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).
- The improvement turned out to be wrong (Mihálykó, 2022; Madarasi, Matúz, 2023).

Question

Is there a quadratic-time algorithm for the problem?

Answer: yes!

- Maintain the (k, ℓ)-components efficiently $\Rightarrow$ reject edges in $O(1)$.
- Analysis: components can intersect, but "not too much" $\Rightarrow O(n^2 + m)$ can be achieved.

Applications:

- Minimum-cost rigid spanning subgraph problem.
- 2-approximation for the minimum-cost redundantly rigid (or globally rigid) spanning subgraph problem (Jordán, Mihálykó, 2019).

Naive algorithm: $O(nm)$ via augmenting paths (Berg, Jordán, 2003; Lee, Streinu, 2008).

- A paper claimed to improve this to $O(n^2 + m)$ (Lee, Streinu, Theran, 2005).
- The improvement turned out to be wrong (Mihálykó, 2022; Madarasi, Matúz, 2023).

Question

Is there a quadratic-time algorithm for the problem?

Answer: yes!

- Maintain the (k, ℓ)-components efficiently $\Rightarrow$ reject edges in $O(1)$.
- Analysis: components can intersect, but "not too much" $\Rightarrow$ $O(n^2 + m)$ can be achieved.

Applications:

- Minimum-cost rigid spanning subgraph problem.
- 2-approximation for the minimum-cost redundantly rigid (or globally rigid) spanning subgraph problem (Jordán, Mihálykó, 2019).

Now all of these can be solved in quadratic time.

Recognition – Intro

Note

When $0 \leq \ell < 2k$, the recognition algorithm is a matroid independence oracle.

Note

When $0 \le \ell < 2k$, the recognition algorithm is a matroid independence oracle.

Range	Case	Old bounds	
$0 \le \ell \le k$	general case	$O(n\sqrt{n \log n})$	(Gabow, Westermann, 1988)
	case $\ell = k$	$O(n^{1+o(1)})$	(Arkhipov, Kolmogorov, 2024)
$k < \ell < 2k$	general case	$O(n^2)$	(Berg, Jordán, 2003; Lee, Streinu, 2008)
	Laman case	$O(n^{1+o(1)})$	(Daescu, Kurdia, 2009)
$2k \le \ell < 3k$	general case	$O(n^3)$	(Madarasi, Matúz, 2023)
	case $\ell = 2k$	$O(n^2)$	(Madarasi, Matúz, 2023)

(In the case $2k \le \ell < 3k$, we require the sparsity bound only for sets with at least 3 vertices.)

Recognition – Intro

Note

When $0 \leq \ell < 2k$, the recognition algorithm is a matroid independence oracle.

Range	Case	Old bounds	
$0 \leq \ell \leq k$	general case	$O(n\sqrt{n \log n})$	(Gabow, Westermann, 1988)
	case $\ell = k$	$O(n^{1+o(1)})$	(Arkhipov, Kolmogorov, 2024)
$k < \ell < 2k$	general case	$O(n^2)$	(Berg, Jordán, 2003; Lee, Streinu, 2008)
	Laman case	$O(n^{1+o(1)})$	(Daescu, Kurdia, 2009)
$2k \leq \ell < 3k$	general case	$O(n^3)$	(Madarasi, Matúz, 2023)
	case $\ell = 2k$	$O(n^2)$	(Madarasi, Matúz, 2023)

(In the case $2k \leq \ell < 3k$, we require the sparsity bound only for sets with at least 3 vertices.)

Note

When $0 \leq \ell < 2k$, the recognition algorithm is a matroid independence oracle.

Range	Case	Old bounds	
$0 \leq \ell \leq k$	general case	$O(n\sqrt{n \log n})$	(Gabow, Westermann, 1988)
	case $\ell = k$	$O(n^{1+o(1)})$	(Arkhipov, Kolmogorov, 2024)
$k < \ell < 2k$	general case	$O(n^2)$	(Berg, Jordán, 2003; Lee, Streinu, 2008)
	Laman case	$O(n^{1+o(1)})$	(Daescu, Kurdia, 2009)
$2k \leq \ell < 3k$	general case	$O(n^3)$	(Madarasi, Matúz, 2023)
	case $\ell = 2k$	$O(n^2)$	(Madarasi, Matúz, 2023)

(In the case $2k \leq \ell < 3k$, we require the sparsity bound only for sets with at least 3 vertices.)

An open question from rigidity theory:

Question

Is there a subquadratic algorithm for recognizing $(2,3)$-sparse graphs?

Recognition – A useful reduction

Recognition – A useful reduction

Suppose $tk \leq \ell \leq (t+1)k$, and let $U_0 \subseteq V$ be a stable set with t vertices.

Subproblem: is there a violating set strictly containing U_0?

Recognition – A useful reduction

Suppose $tk \leq \ell \leq (t+1)k$, and let $U_0 \subseteq V$ be a stable set with t vertices.

Subproblem: is there a violating set strictly containing U_0?

Statement

The above subproblem can be reduced to rooted $(\ell - tk)$-arc-connectivity.

Suppose $tk \leq \ell \leq (t+1)k$, and let $U_0 \subseteq V$ be a stable set with t vertices.

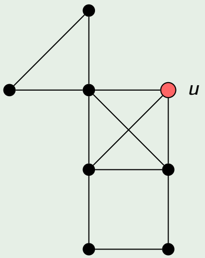
Subproblem: is there a violating set strictly containing U_0?

Statement

The above subproblem can be reduced to rooted $(\ell - tk)$-arc-connectivity.

Example

Let $(k, \ell) = (2, 3)$, $U_0 = \{u\}$.



Suppose $tk \leq \ell \leq (t+1)k$, and let $U_0 \subseteq V$ be a stable set with t vertices.

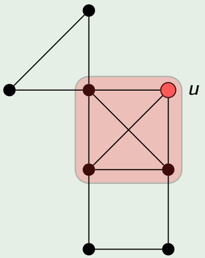
Subproblem: is there a violating set strictly containing U_0?

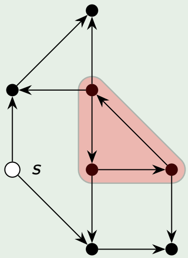
Statement

The above subproblem can be reduced to rooted $(\ell - tk)$-arc-connectivity.

Example

Let $(k, \ell) = (2, 3)$, $U_0 = \{u\}$. Consider the graph G and digraph D below. Here, $X \ni u$ violates the sparsity of $G \Leftrightarrow X - u$ violates rooted 1-arc-connectivity in D.





Recognition – The range $k < \ell < 2k$

Recognition – The range $k < \ell < 2k$

Nash-Williams theorem $\Longrightarrow$ a (k, ℓ)-sparse graph has a forest-decomposition $F_1, \ldots, F_k$.

Nash-Williams theorem $\Longrightarrow$ a (k, ℓ)-sparse graph has a forest-decomposition $F_1, \dots, F_k$.

Definition

A forest F **saturates** the vertex set X if $F[X]$ is connected.

Nash-Williams theorem $\Longrightarrow$ a (k, ℓ)-sparse graph has a forest-decomposition $F_1, \ldots, F_k$.

Definition

A forest F **saturates** the vertex set X if $F[X]$ is connected.

Observation

A violating set X is saturated by one of $F_1, \ldots, F_{\ell-k}$.

F_i saturates X if and only if one of its components does.

Nash-Williams theorem $\Longrightarrow$ a (k, ℓ)-sparse graph has a forest-decomposition $F_1, \ldots, F_k$.

Definition

A forest F **saturates** the vertex set X if $F[X]$ is connected.

Observation

A violating set X is saturated by one of $F_1, \ldots, F_{\ell-k}$.

F_i saturates X if and only if one of its components does.

Corollary

It suffices to detect if there is a violating set saturated by a given spanning tree F.

Recognition – The range $k < \ell < 2k$

We can quickly decide if there is a violating set containing some vertex c (use our reduction, $U_0 := \{c\}$).

We can quickly decide if there is a violating set containing some vertex c (use our reduction, $U_0 := \{c\}$).

Idea: Let c be the centroid of $F \Rightarrow$ centroid decomposition.

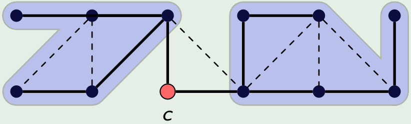
Recognition – The range $k < \ell < 2k$

We can quickly decide if there is a violating set containing some vertex c (use our reduction, $U_0 := \{c\}$).

Idea: Let c be the centroid of $F \Rightarrow$ centroid decomposition.

Example

Let $(k, \ell) = (2, 3)$. Two steps of the centroid decomposition:

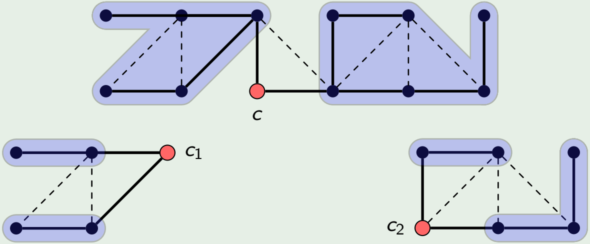


We can quickly decide if there is a violating set containing some vertex c (use our reduction, $U_0 := \{c\}$).

Idea: Let c be the centroid of $F \Rightarrow$ centroid decomposition.

Example

Let $(k, \ell) = (2, 3)$. Two steps of the centroid decomposition:



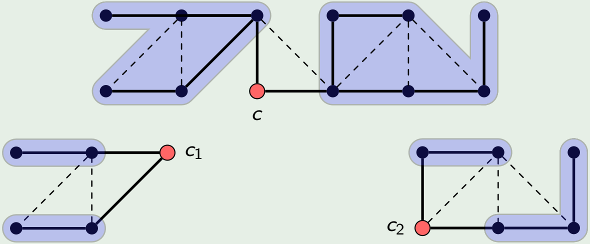
Recognition – The range $k < \ell < 2k$

We can quickly decide if there is a violating set containing some vertex c (use our reduction, $U_0 := \{c\}$).

Idea: Let c be the centroid of $F \Rightarrow$ centroid decomposition.

Example

Let $(k, \ell) = (2, 3)$. Two steps of the centroid decomposition:



Note

For the Laman case, we get the best previous running time, but our algorithm is simpler, much more general, and also provides a violating set.

The range $\ell \leq k$:

1. Find a k-indegree-bounded orientation.

The range $\ell \leq k$:

1. Find a k-indegree-bounded orientation.
2. Apply the reduction for $U_0 = \emptyset$.

The range $\ell \leq k$:

1. Find a k-indegree-bounded orientation.
2. Apply the reduction for $U_0 = \emptyset$.

The range $2k \leq \ell < 3k$:

1. Add the edges one by one.

The range $\ell \leq k$:

1. Find a k-indegree-bounded orientation.
2. Apply the reduction for $U_0 = \emptyset$.

The range $2k \leq \ell < 3k$:

1. Add the edges one by one.
2. Edge uv can be inserted $\Leftrightarrow$ there is no $X \supsetneq \{u, v\}$ violating $(k, \ell + 1)$-sparsity.
3. Apply the reduction for $U_0 = \{u, v\}$.

Summary

Range	Problem	Old bounds	New bounds
$0 \le \ell < k$	recognition	$O(n\sqrt{n \log n})$	$O(n^{1+o(1)})$
$k < \ell < 2k$	recognition	$O(n^2)$	$O(n^{1+o(1)})$
	optimization	$O(nm)$	$O(n^2 + m)$
$2k < \ell < 3k$	recognition	$O(n^3)$	$\begin{cases} O(n^2) & \text{if } \ell = 2k+1 \\ O(n^2 \log n) & \text{if } \ell > 2k+1 \end{cases}$

Summary

Range	Problem	Old bounds	New bounds
$0 \leq \ell < k$	recognition	$O(n\sqrt{n \log n})$	$O(n^{1+o(1)})^*$
$k < \ell < 2k$	recognition	$O(n^2)$	$O(n^{1+o(1)})^\dagger$
	optimization	$O(nm)$	$O(n^2 + m)$
$2k < \ell < 3k$	recognition	$O(n^3)$	$\begin{cases} O(n^2) & \text{if } \ell = 2k+1 \\ O(n^2 \log n) & \text{if } \ell > 2k+1 \end{cases}$

* Purely combinatorial version: $O(n\sqrt{n})$.

† Purely combinatorial version: $O(n\sqrt{n \log n})$.

Summary

Range	Problem	Old bounds	New bounds
$0 \le \ell < k$	recognition	$O(n\sqrt{n \log n})$	$O(n^{1+o(1)})^*$
$k < \ell < 2k$	recognition	$O(n^2)$	$O(n^{1+o(1)})^\dagger$
	optimization	$O(nm)$	$O(n^2 + m)$
$2k < \ell < 3k$	recognition	$O(n^3)$	$\begin{cases} O(n^2) & \text{if } \ell = 2k+1 \\ O(n^2 \log n) & \text{if } \ell > 2k+1 \end{cases}$

* Purely combinatorial version: $O(n\sqrt{n})$.

† Purely combinatorial version: $O(n\sqrt{n \log n})$.

Our papers:

- Optimization problem: Quadratic-Time Algorithm for the Maximum-Weight (k, ℓ)-Sparse Subgraph Problem (https://arxiv.org/abs/2511.20882)
- Recognition problem: in progress...

Throughout the project, no AI/ML tools were used.

Thank you for your attention!