Fine-Grained Complexity - Hardness in P

Lecture 2: APSP

Karl Bringmann
Landscape of Polytime Problems

- **SETH-hard**
  - **SAT** $2^n$
    - $k$-DomSet $n^k$
    - Frechet $n^2$
    - Diameter $n^2$
    - Longest Palindromic Subsequence $n^2$
    - LCS $n^2$

- **3SUM-hard**
  - **3SUM** $n^2$
    - X+Y $n^2$
    - Colinear $n^2$
    - GeomBase $n^2$
    - Separator $n^2$
    - PlanarMotion Planning $n^2$

- **APSP-hard**
  - **APSP** $n^3$
    - Radius $n^3$
    - Metricity $n^3$
    - Betweenness Centrality $n^3$
    - Tree Edit Distance $n^3$
    - NegTriangle $n^3$

- **Other Problems**
  - SubsetSum $n + t$
  - Dynamic Time Warping $n^2$
  - Edit Distance $n^2$
  - NFA-Acceptance $n^2$
  - RegExp Matching $n^2$

- **Other**
  - LCS $n^2$
  - Frechet $n^2$
Landscape of Polytime Problems

- SAT $2^n$
- SubsetSum $n + t$
- Dynamic Time Warping $n^2$
- Edit Distance $n^2$
- NFA-Acceptance $n^2$
- RegExp Matching $n^2$
- k-DomSet $n^k$
- Frechet $n^2$
- Diameter $n^2$
- LCS $n^2$
- Longest Palindromic Subsequence $n^2$

**SETH-hard**
- [Patrascu, Williams’10]
- [B’14]
- [V-Williams, Roditty’13]
- [B, Künnemann’15, Abboud, Backurs, V-Williams’15]
- [Backurs, Indyk’15]
- [Gajentaan, Overmars’98]
- [B, Künnemann’15, Abboud, Backurs, V-Williams’15]
- [Backurs, Indyk’15]
- [Backurs, Indyk’16]
- [Impagliazzo]
- [Backurs, Dikkala, Tzamos’16]
- [B, Gawrychowski, Mozes, Weimann’18]
- [Backurs, Indyk’15]

- 3SUM $n^2$
- X+Y $n^2$
- Colinear $n^2$
- GeomBase $n^2$
- Separator $n^2$
- Planar Motion Planning $n^2$

**3SUM-hard**
- [Gajentaan, Overmars’98]

- APSP $n^3$
- Radius $n^3$
- Metricity $n^3$
- Betweenness Centrality $n^3$
- NegTriangle $n^3$
- Maximum Submatrix $n^3$
- Tree Edit Distance $n^3$

**APSP-hard**
- [V-Williams, Williams’10]
**Subcubic Reductions**

A **subcubic reduction** from $P$ to $Q$ is an algorithm $A$ for $P$ with **oracle** access to $Q$ s.t.:

- for any instance $I$, algorithm $A(I)$ correctly solves problem $P$ on $I$
- $A$ runs in time $r(n) = O(n^{3-\gamma})$ for some $\gamma > 0$
- for any $\varepsilon > 0$ there is a $\delta > 0$ s.t. $\sum_{i=1}^{k} n_i^{3-\varepsilon} \leq n^{3-\delta}$

**Properties:**

- for any instance $I$, algorithm $A(I)$ correctly solves problem $P$ on $I$
- $A$ runs in time $r(n) = O(n^{3-\gamma})$ for some $\gamma > 0$
- for any $\varepsilon > 0$ there is a $\delta > 0$ s.t. $\sum_{i=1}^{k} n_i^{3-\varepsilon} \leq n^{3-\delta}$
Problem Definitions

Problem All-Pairs-Shortest-Paths (APSP):
given a weighted directed graph $G$, compute the (length of the)
shortest path between any pair of vertices

APSP-Hypothesis:
$\forall \varepsilon > 0$: APSP has no $O(n^{3-\varepsilon})$-time algorithm
there exists $c > 0$ such that

Algorithms:

$O(n^3)$  
[Floyd‘62,Warshall‘62]

... 

$O\left(\frac{n^3}{2^{\Omega(\log n)^{1/2}}}\right)$  
[Williams‘14]
Problem Definitions

Problem All-Pairs-Shortest-Paths (APSP):
given a weighted directed graph $G$, compute the (length of the) shortest path between any pair of vertices

APSP-Hypothesis:
$\forall \varepsilon > 0$: APSP has no $O(n^{3-\varepsilon})$-time algorithm

Problem Min-Plus Matrix Product:
given $n \times n$-matrices $A, B$, define their min-plus product as the $n \times n$-matrix $C$ with
\[
C_{i,j} = \min_{1 \leq k \leq n} A_{i,k} + B_{k,j}
\]
each entry in $\{1, \ldots, n^c, \infty\}$

Naive algorithm: $O(n^3)$
compute all pairwise distances in a graph

compute matrix $C$ with

$$C_{i,j} = \min_{1 \leq k \leq n} A_{i,k} + B_{k,j}$$

**Problem NegTriangle:**

Given a weighted directed graph $G$

Decide whether there are vertices $i, j, k$ s.t.

$$w(j, i) + w(i, k) + w(k, j) < 0$$

$$\text{APSP} \iff \text{Min-Plus Product} \iff \text{All-Pairs-Negative-Triangle} \iff \text{Negative Triangle}$$

- non-trivial $O(n^3)$-algorithm, output size $n^2$
- trivial $O(n^3)$-algorithm, output size $n^2$
- trivial $O(n^3)$-algorithm, output size 1

[Vassilevska-Williams, Williams’10]
[Abboud, Grandoni, Vassilevska-Williams’15]
Easy Application: Minimum Weight Cycle

Given a weighted directed graph $G$, find the smallest weight of any (directed) cycle.

**MinWeightCycle** $\rightarrow$ **APSP**

compute all pairwise distances $d(u, v)$, the minimum weight of any cycle is $\min_{(u,v) \in E} w(u, v) + d(v, u)$

**NegTriangle** $\rightarrow$ **MinWeightCycle**

Can assume that there are no double edges, since input graph is tripartite

Let $M \geq w(i, j)$ for all $i, j$

Add $10 \cdot M$ to each edge weight

Then a cycle with $k$ edges has length in $[9 \cdot M \cdot k, 11 \cdot M \cdot k]$.

So any triangle has smaller length than any 4-cycle, 5-cycle, ...

So minimum weight of any cycle is $< 30 \cdot M$ iff there is a negative triangle.
More Applications

- APSP
- Min-Plus Product
- All-Pairs-Negative-Triangle
- Negative Triangle
- 2nd Shortest Path
- Maximum Submatrix
- Metricity
- Betweenness Centrality
- Radius
- Median

this is surprising!
this is useful!

[Vassilevska-Williams, Williams’10]
[Abboud, Grandoni, Vassilevska-Williams’15]
I. Equivalence of APSP and NegTriangle

II. Example Applications

III. Further Topics

IV. Conclusion
Subcubic Equivalences

- APSP
- Min-Plus Product
- All-Pairs-Negative-Triangle
- Negative Triangle

Subcubic Equivalences
APSP ⇔ Min-Plus-Product

**Thm:** If APSP is in time $T(n)$ then Min-Plus Product is in time $O(T(n))$.

**Proof:** Given matrices $A, B$, construct graph:

$w(i, k) = A[i, k]$  $w(k, j) = B[k, j]$
APSP $\iff$ Min-Plus-Product

**Thm:** If APSP is in time $T(n)$ then Min-Plus Product is in time $O(T(n))$.

**Thm:** If Min-Plus Product is in time $T(n)$ then APSP is in $O(T(n) \log n)$.

**Proof:** Given graph $G$ with adjacency matrix $A$

Add selfloops with cost 0, this yields adjacency matrix $\hat{A}$

Square $[\log n]$ times using Min-Plus Product:

$$ B := \hat{A}^{[\log n]} $$

Then $B_{i,j}$ is the length of the shortest path from $i$ to $j$

**Property:** $(\hat{A}^k)_{i,j} = \text{length of shortest path from } i \text{ to } j \text{ using } \leq k \text{ hops}$
Cor: APSP has an $O(n^{3-\varepsilon})$ algorithm for some $\varepsilon > 0$ if and only if Min-Plus Product has an $O(n^{3-\delta})$ algorithm for some $\delta > 0$.

Cor: Min-Plus Product is in time $O\left(\frac{n^3}{2^{\Omega(\log n)^{1/2}}}\right)$. 
Subcubic Equivalences

- APSP
- Min-Plus Product
- All-Pairs-Negative-Triangle
- Negative Triangle
Triangle Problems

Negative Triangle

Given a weighted directed graph $G$

Decide whether there are vertices $i, j, k$ s.t.

$$w(j, i) + w(i, k) + w(k, j) < 0$$

Naive algorithm: $O(n^3)$

Intermediate problem:

All-Pairs-Negative-Triangle

Given a weighted directed graph $G$ with vertex set $V = I \cup J \cup K$

Decide for every $i \in I, j \in J$ whether there is a vertex $k \in K$ s.t.

$$w(j, i) + w(i, k) + w(k, j) < 0$$
Subcubic Equivalences

- APSP
- Min-Plus Product
- All-Pairs-Negative-Triangle
- Negative Triangle

\[ \text{APSP} \iff \text{Min-Plus Product} \iff \text{All-Pairs-Negative-Triangle} \iff \text{Negative Triangle} \]
Given a weighted directed graph $G$ on vertex set $\{1, \ldots, n\}$

Adjacency matrix $A$:

$A_{i,j} =$ weight of edge $(i, j)$, or $\infty$ if the edge does not exist

1. Compute Min-Plus Product $B := A \ast A$:

$B_{i,j} = \min_k A_{i,k} + A_{k,j}$

2. Compute $\min_{i,j} A_{j,i} + B_{i,j}$

$= \min_{i,j,k} A_{j,i} + A_{i,k} + A_{k,j}$

$= \text{the smallest weight of any triangle}$

thus we solved Negative Triangle

Running Time: $T_{\text{NegTriangle}}(n) \leq T_{\text{MinPlus}}(n) + O(n^2)$

$\rightarrow \text{subcubic reduction}$
Subcubic Equivalences

APSP \iff \text{Min-Plus Product} \iff \text{All-Pairs-Negative-Triangle} \iff \text{Negative Triangle}
Add all edges from $J$ to $I$ with (carefully chosen) weights $w(j, i)$

Run All-Pairs-Negative-Triangle algorithm

Result: for every $i, j$, is there a $k$ such that $w(j, i) + w(i, k) + w(k, j) < 0$?

$\Leftrightarrow w(i, k) + w(k, j) < -w(j, i)$

WANTED: Min-Plus: for every $i, j$: $\displaystyle \min_k w(i, k) + w(k, j)$

$= \text{minimum number } z \text{ s.t. there is a } k \text{ s.t. } w(i, k) + w(k, j) < z + 1$

binary search via $w(j, i)$! simultaneous for all $i, j$!
Min-Plus to All-Pairs-Neg-Triangle

\[
\begin{array}{ccccccc}
3 & 1 & \infty & \infty & & & \\
\infty & \infty & 4 & \infty & & & \\
\infty & \infty & \infty & 2 & & & \\
\infty & \infty & \infty & 1 & & & \\
\end{array}
\]

\[
\begin{array}{ccccccc}
5 & \infty & \infty & \infty & & & \\
7 & \infty & \infty & \infty & & & \\
\infty & 2 & \infty & \infty & & & \\
\infty & \infty & \infty & 4 & & & \\
\end{array}
\]

\[
\begin{array}{ccccccc}
3 & 1 & 5 & & & & \\
3 & 1 & 5 & & & & \\
4 & 7 & 2 & & & & \\
2 & 4 & 1 & & & & \\
2 & 4 & 1 & & & & \\
-2 & 4 & 1 & & & & \\
-7 & & & & & & \\
\end{array}
\]

binary search via \(w(j, i)\)! simultaneous for all \(i, j\)!

need that all (finite) weights are in \([-n^c, \ldots, n^c]\)
each entry of Min-Plus Product is in \([-2n^c, \ldots, 2n^c, \infty]\)
binary search takes \(\log_2 (4n^c + 1) = O(\log n)\) steps
Min-Plus to All-Pairs-Neg-Triangle

\[
\begin{array}{ccccccc}
3 & 1 & \infty & \infty & & & 5 & \infty & \infty & \infty \\
\infty & \infty & 4 & \infty & & & 7 & \infty & \infty & \infty \\
\infty & \infty & \infty & 2 & & & \infty & 2 & \infty & \infty \\
\infty & \infty & \infty & \infty & 1 & & & \infty & \infty & \infty & 4 \\
\end{array}
\]

\[ A \]

\[ I \]

\[ J \]

\[ B \]

\[ K \]

\[
\begin{array}{cccc}
3 & 1 & & \\
& 5 & & \\
& & 7 & \\
& & & 2 \\
& & & \infty \\
\end{array}
\]

\[
\begin{array}{cccc}
4 & & & \\
& & & 2 \\
& & & \infty \\
& & & \infty \\
\end{array}
\]

\[
\begin{array}{cccc}
1 & & & \\
& & & \infty \\
& & & \infty \\
& & & \infty \\
\end{array}
\]

\[
\begin{array}{cccc}
& & & \\
& & & \infty \\
& & & \infty \\
& & & \infty \\
\end{array}
\]

\[
\begin{array}{cccc}
& & & \\
& & & \infty \\
& & & \infty \\
& & & \infty \\
\end{array}
\]

\[
\begin{array}{cccc}
& & & \\
& & & \infty \\
& & & \infty \\
& & & \infty \\
\end{array}
\]

binary search via \( w(j, i) \)! **simultaneous** for all \( i, j \)!

for all \( i, j \): initialize \( m(i, j) := -2n^c \) and \( M(i, j) := 2n^c \)

repeat \( \log(4n^c) \) times:

for all \( i, j \): set \( w(j, i) := -[(m(i, j) + M(i, j))/2] \)

compute All-Pairs-Negative-Triangle

for all \( i, j \): if \( i, j \) is in negative triangle: \( M(i, j) := -w(j, i) - 1 \)

otherwise: \( m(i, j) := -w(j, i) \)

\( n = 4 \) in the picture

(missing: handling of \( \infty \))
binary search takes $\log_2(4n^c + 1) = O(\log n)$ steps

$T(n)$ algorithm for All-Pairs-Neg-Triangle yields $O(T(n) \log n)$ algorithm for Min-Plus Product

In particular: $O(n^{3-\varepsilon})$ algorithm for All-Pairs-Neg-Triangle for some $\varepsilon > 0$ implies $O(n^{3-\varepsilon})$ algorithm for Min-Plus Product for some $\varepsilon > 0$ → subcubic reduction
Subcubic Equivalences

- APSP
- Min-Plus Product
- All-Pairs-Negative-Triangle
- Negative Triangle
All-Pairs-Neg-Triangle to Neg-Triangle

**Negative Triangle**  Given graph $G$
Decide whether there are vertices $i, j, k$ such that
\[ w(j, i) + w(i, k) + w(k, j) < 0 \]

**All-Pairs-Negative-Triangle**  Given graph $G$ with vertex set $V = I \cup J \cup K$
Decide for every $i \in I, j \in J$ whether there is a vertex $k \in K$ such that
\[ w(j, i) + w(i, k) + w(k, j) < 0 \]

Split $I, J, K$ into $n/s$ parts of size $s$:
\[ I_1, \ldots, I_{n/s}, J_1, \ldots, J_{n/s}, K_1, \ldots, K_{n/s} \]

For each of the $(n/s)^3$ triples $(I_x, J_y, K_z)$:
consider graph $G[I_x \cup J_y \cup K_z]$
All-Pairs-Neg-Triangle to Neg-Triangle

Initialize $C$ as $n \times n$ all-zeroes matrix

For each of the $(n/s)^3$ triples of parts $(I_x, J_y, K_z)$:

While $G[I_x \cup J_y \cup K_z]$ contains a negative triangle:

Find a negative triangle $(i, j, k)$ in $G[I_x \cup J_y \cup K_z]$

Set $C[i, j] := 1$

Set $w(i, j) := \infty$

$(i, j)$ is in no more negative triangles

✓ guaranteed termination:
  can set $\leq n^2$ weights to $\infty$

✓ correctness:
  if $(i, j)$ is in negative triangle, we will find one
All-Pairs-Neg-Triangle to Neg-Triangle

Find a negative triangle \((i, j, k)\) in \(G[I_x \cup J_y \cup K_z]\)

**How to find a negative triangle**
*if we can only decide whether one exists?*

Partition \(I_x\) into \(I_x^{(1)}, I_x^{(2)}\), \(J_y\) into \(J_y^{(1)}, J_y^{(2)}\), \(K_z\) into \(K_z^{(1)}, K_z^{(2)}\)

Since \(G[I_x \cup J_y \cup K_z]\) contains a negative triangle, at least one of the \(2^3\) subgraphs
\[ G[I_x^{(a)} \cup J_y^{(b)} \cup K_z^{(c)}] \]
contains a negative triangle

Decide for each such subgraph whether it contains a negative triangle

Recursively find a triangle in one subgraph
All-Pairs-Neg-Triangle to Neg-Triangle

Find a negative triangle \((i, j, k)\) in \(G[I_x \cup J_y \cup K_z]\)

*How to find a negative triangle if we can only decide whether one exists?*

Partition \(I_x\) into \(I_x^{(1)}, I_x^{(2)}\), \(J_y\) into \(J_y^{(1)}, J_y^{(2)}\), \(K_z\) into \(K_z^{(1)}, K_z^{(2)}\)

Since \(G[I_x \cup J_y \cup K_z]\) contains a negative triangle, at least one of the \(2^3\) subgraphs \(G[I_x^{(a)} \cup J_y^{(b)} \cup K_z^{(c)}]\) contains a negative triangle

Decide for each such subgraph whether it contains a negative triangle

Recursively find a triangle in one subgraph

Running Time:

\[
T_{\text{FindNegTriangle}}(n) \leq 2^3 \cdot T_{\text{DecideNegTriangle}}(n) + T_{\text{FindNegTriangle}}(n/2)
\]

\[
= O(T_{\text{DecideNegTriangle}}(n))
\]
All-Pairs-Neg-Triangle to Neg-Triangle

Initialize $C$ as $n \times n$ all-zeroes matrix

For each of the $(n/s)^3$ triples of parts $(I_x, J_y, K_z)$:

While $G[I_x \cup J_y \cup K_z]$ contains a negative triangle:

Find a negative triangle $(i, j, k)$ in $G[I_x \cup J_y \cup K_z]$

Set $C[i, j] := 1$

Set $w(i, j) := \infty$

Running Time:

\[(*) = O(T_{\text{FindNegTriangle}}(s)) = O(T_{\text{DecideNegTriangle}}(s))\]

Total time: \[\left((\#\text{triples}) + (\#\text{triangles found})\right) \cdot (*)\]

\[\leq (n/s)^3 + n^2 \right) \cdot T_{\text{DecideNegTriangle}}(s)\]

Set $s = n^{1/3}$ and assume

\[T_{\text{DecideNegTriangle}}(n) = O(n^{3-\varepsilon})\]

Total time: \[O(n^2 \cdot n^{1-\varepsilon/3}) = O(n^{3-\varepsilon/3})\]
Subcubic Equivalences

APSP

Min-Plus Product

All-Pairs-Negative-Triangle

Negative Triangle
I. Equivalence of APSP and NegTriangle

II. Example Applications

III. Further Topics

IV. Conclusion
Subcubic Equivalences

- 2nd Shortest Path
- Maximum Submatrix
- Metricity
- APSP
- Min-Plus Product
- All-Pairs-Negative-Triangle
- Negative Triangle
- Betweenness Centrality
- Radius
- Median
Radius

$G$ is a weighted directed graph

d$(u, v)$ is the distance from $u$ to $v$ in $G$

**Radius:** \( \min_u \max_v d(u, v) \)

$u$ is in some sense the *most central vertex*

compute all pairwise distances,
then evaluate definition of radius in time \( O(n^2) \)

\( \rightarrow \) subcubic reduction

\( \Rightarrow \) Radius is in time \( O\left(\frac{n^3}{2^{\Omega(\log n)^{1/2}}}\right) \)
Negative Triangle to Radius

**Negative Triangle** instance:
graph $G$ with $n$ nodes,
edge-weights in $\{-n^c, \ldots, n^c\}$

**Radius** instance:
graph $H$ with $O(n)$ nodes,
edge-weights in $\{0, \ldots, O(n^c)\}$

1) Make four layers with $n$ nodes
2) For any edge $(i, j)$: Add $(i_A, j_B)$,
   $(i_B, j_C), (i_C, j_D)$ with weight $M + w(i, j)$

$M := 3n^c$
Negative Triangle to Radius

**Negative Triangle** instance:
graph $G$ with $n$ nodes, edge-weights in $\{-n^c, ..., n^c\}$

(i, j, k) has weight $W$

1) Make four layers with $n$ nodes
2) For any edge (i, j): Add (i$_A$, j$_B$), (i$_B$, j$_C$), (i$_C$, j$_D$) with weight $M + w(i,j)$

**Radius** instance:
graph $H$ with $O(n)$ nodes, edge-weights in $\{0, ..., O(n^c)\}$

$M := 3n^c$

$\Leftrightarrow$ path has length $3M + W$

$\Rightarrow \exists i_A, j_B, k_C, i_D$-path of length $\leq 3M - 1$?
Negative Triangle to Radius

**Negative Triangle** instance:

- graph $G$ with $n$ nodes,
- edge-weights in $\{-n^c, \ldots, n^c\}$

$(i, j, k)$ has weight $W$

1) Make four layers with $n$ nodes
2) For any edge $(i, j)$: Add $(i_A, j_B)$, $(i_B, j_C), (i_C, j_D)$ with weight $M + w(i, j)$
3) Add edges of weight $3M - 1$ from any $i_A$ to all nodes except $i_D$

**Radius** instance:

- graph $H$ with $O(n)$ nodes,
- edge-weights in $\{0, \ldots, O(n^c)\}$

$M := 3n^c$

$\Leftrightarrow$ path has length $3M + W$

$\rightarrow \exists i_A, j_B, k_C, i_D$-path of length $\leq 3M - 1$?

**Claim:** Radius of $H$ is $\leq 3M - 1$ iff there is a negative triangle in $G$
Subcubic Equivalences

- 2nd Shortest Path
- Maximum Submatrix
- Metricity
- APSP
- Min-Plus Product
- All-Pairs-Negative-Triangle
- Betweenness Centrality
- Radius
- Median
- Negative Triangle

\[ \text{Maximum Submatrix} \Leftrightarrow \text{2nd Shortest Path} \]
\[ \text{2nd Shortest Path} \Leftrightarrow \text{Min-Plus Product} \]
\[ \text{Min-Plus Product} \Leftrightarrow \text{All-Pairs-Negative-Triangle} \]
\[ \text{All-Pairs-Negative-Triangle} \Leftrightarrow \text{Betweenness Centrality} \]
\[ \text{Betweenness Centrality} \Leftrightarrow \text{Radius} \]
\[ \text{Radius} \Leftrightarrow \text{Median} \]
\[ \text{Median} \Leftrightarrow \text{Negative Triangle} \]
\[ \text{Negative Triangle} \Leftrightarrow \text{2nd Shortest Path} \]

\[ \text{Maximum Submatrix} \Leftrightarrow \text{Negative Triangle} \]
MaxSubmatrix:

given an $n \times n$ matrix $A$ with entries in $\{-n^c, \ldots, n^c\}$

$\Sigma(B) := \text{sum of all entries}$ of matrix $B$

compute maximum $\Sigma(B)$ over all submatrices $B$ of $A$

**Thm:** MaxSubmatrix is subcubic equivalent to APSP

there are $O(n^4)$ possible submatrices $B$

computing $\Sigma(B)$: $O(n^2)$

trivial running time: $O(n^6)$

**Exercise:** design an $O(n^3)$ algorithm

[Tamaki, Tokuyama’98]
[Backurs, Dikkala, Tzamos’16]
MaxSubmatrix:

given an $n \times n$ matrix $A$ with entries in $\{-n^c, \ldots, n^c\}$

$\Sigma(B) := \text{sum of all entries}$ of matrix $B$

compute maximum $\Sigma(B)$ over all submatrices $B$ of $A$

**Thm:** MaxSubmatrix is subcubic equivalent to APSP

[Tamaki, Tokuyama’98]
[Backurs, Dikkala, Tzamos’16]

MaxCenteredSubmatrix:

compute maximum $\Sigma(B)$ over all submatrices $B$ of $A$ containing the center of $A$
i.e. we require $x_1 \leq n/2 < x_2$ and $y_1 \leq n/2 < y_2$

**Thm:** MaxCenteredSubmatrix is subcubic equ. to APSP

we only prove: NegativeTriangle $\leq$ MaxCenteredSubmatrix

**Exercise:** MaxCenteredSubmatrix $\leq$ APSP
**NegTriangle to MaxCentSubmatrix**

**Positive Triangle** instance:
graph $G$ with $n$ nodes,
edge-weights in $\{-n^c, ..., n^c\}$

**MaxCenteredSubmatrix:**
$2n \times 2n$-matrix $A$
entries in $\{-n^{O(c)}, ..., n^{O(c)}\}$

**Claim:** MaxCentSubmatrix of $A$ is $> M$
iff $G$ has a **positive** triangle

In quadrant II we want for any $k, i$:
$$
\sum_{y=k}^{n} \sum_{x=i}^{n} A_{y,x} = w(k, i)
$$

this is satisfied by defining:
$$
A_{k,i} := w(k, i) - w(k + 1, i) + w(k + 1, i + 1) - w(k, i + 1) + w(k + 1, i + 1)
$$

(where $w(x, y) := 0$ for $x > n$ or $y > n$)
I. Equivalence of APSP and NegTriangle

II. Example Applications

III. Further Topics

IV. Conclusion
Weighted k-Clique

Problem Negative-\(k\)-Clique:
Given weighted directed graph \(G\)
is there a \(k\)-Clique with negative total edge-weight?

Neg-\(k\)-Clique-Hypothesis:
\(\forall \varepsilon > 0, k \geq 3: \) Neg-\(k\)-Clique has no \(O(n^{k-\varepsilon})\) algorithm

“Yields more lower bounds since the input is sparser”
Tree Edit Distance

on two rooted ordered trees $T, T'$ with nodes labeled by $\Sigma$

determine minimum cost of edit operations transforming $T$ into $T'$

edit operations:

- relabel a node, $\text{cost}_{\text{rel}}(x, y)$
- insert a node, $\text{cost}_{\text{ins}}(x)$
- delete a node, $\text{cost}_{\text{del}}(x)$

Input size: $O(n + |\Sigma|^2)$
Tree Edit Distance

on two rooted ordered trees \(T, T'\) with nodes labeled by \(\Sigma\)

determine minimum cost of edit operations transforming \(T\) into \(T'\)

first algorithm: \(O(n^6)\) \[Tai'79\]

a series of papers improved to: \(O(n^3)\) \[Demaine, Mozes, Rossman, Weimann'07\]

Thm: \[B., Mozes, Gawrychowski, Weimann'18\]

For alphabet \(|\Sigma| = \Omega(n)\), a truly subcubic algorithm for tree edit distance implies a truly subcubic algorithm for \textbf{APSP}.

For \(|\Sigma| = O(1)\), a truly subcubic algorithm for tree edit distance implies an \(O(n^{k-\varepsilon})\) algorithm for \textbf{Neg}-\(k\)-\textbf{Clique}.

other applications: Max-Weight-Rectangle, Viterbi, ...

\[Backurs, Dikkala, Tzamos'16\] \[Backurs, Tzamos'17\]
Weighted k-Clique

Problem Negative-$k$-Clique:
Given weighted directed graph $G$ is there a $k$-Clique with negative total edge-weight?

Neg-$k$-Clique-Hypothesis:
$\forall \varepsilon > 0, k \geq 3$: Neg-$k$-Clique has no $O(n^{k-\varepsilon})$ algorithm

“Neg-$k$-Clique unifies OV and APSP”
(Weighted) $k$-Clique in Hypergraphs

$r$-hypergraph:

\[ G = (V, E) \text{ with } E \subseteq \binom{V}{r} \]

note: $2$-hypergraph = graph

$k$-Clique in $r$-hypergraph:

vertices $v_1, \ldots, v_k$ s.t.
for any $e \subseteq \{v_1, \ldots, v_k\}$ of size $r$ we have $e \in E$

\[ O(n^{0.79k}) \] known for $k$-Clique in graphs \[ \text{[NP'85]} \]

\[ O(n^{k-\varepsilon}) \] not known for Neg-$k$-Clique in graphs or $k$-Clique in $3$-hypergraphs

OVH fails:

\[ \Rightarrow \]

\[ O(n^{k-\varepsilon}) \] for Neg-$k$-Clique in $r$-hypergraphs

for any $k \gg r$ and weights bounded by $n^{f(k)}$
Proof Outline

1. **Neg-2k-Clique**
   \( r \)-hypergraphs
   - From testing \( \leq \)
     to testing \( = \)

2. **ExactWeight-2k-Clique**
   \( r \)-hypergraphs
   - Removing weights by increasing the arity

3. **2k-Clique**
   \( 2r \)-hypergraphs
   - Implementing Constraint Satisfaction Problems by Orthogonal Vectors
Proof Outline – Step (2)

Removing weights by increasing the arity

**ExactWeight-\(k\)-Clique**

Given target \(t\), graph \(G\), weights \(w\), is there a \(k\)-clique of weight \(t\)?

assume weights bounded by \(W = O(n^{f(k)})\)

Consider \(k\)-clique \(C\) with \(\sum_{e \subseteq C} w(e) = t\)

Base-\(B\) expansion: \(t = \sum_{\ell} t_\ell \cdot B^\ell, \ w(e) = \sum_{\ell} w_\ell(e) \cdot B^\ell\)

we have \(\sum_{e \subseteq C} w(e) = t\)

\(\iff\) \(\exists\) carries \(c_\ell \in \{0, ..., O(k^2)\}\) such that \(c_\ell + \sum_{e \subseteq C} w_\ell(e) = t_\ell + c_{\ell+1} \cdot B\ \ \forall \ell\)

**guess carries:** blowup of \(O(k^2)^{\log W / \log B} = n^{o(1)}\) for \(B := \log n\)
Proof Outline – Step (2)

Removing weights by increasing the arity

\[ W = O(n^{f(k)}) \]
\[ B := \log n \]

**ExactWeight-\(k\)-Clique**

Given target \(t\), graph \(G\), weights \(w\), is there a \(k\)-clique of weight \(t\)?

New problem after guessing carries:

Find \(k\)-clique \(C\) with

\[ c_\ell + \sum_{e \subseteq C} w_\ell(e) = t_\ell + c_{\ell+1} \cdot B \quad \forall \ell \]

\[ \iff \sum_{e \subseteq C} w'_\ell(e) = 0 \quad \forall \ell \quad \text{with } w'_\ell(e) := c_\ell + \binom{k}{2} w_\ell(e) - t_\ell - c_{\ell+1} \cdot B \]

\[ \iff \sum_\ell (\sum_{e \subseteq C} w'_\ell(e))^2 = 0 \]

\[ \iff \sum_{e_1, e_2 \subseteq C} \sum_\ell w'_\ell(e_1) \cdot w'_\ell(e_2) = 0 \]

\[ \iff \sum_{h \subseteq C, |h|=4} w''''(h) = 0 \quad \text{with weights bounded by } O\left( B^2 \frac{\log W}{\log B} \right) = \text{polylog } n \]

**Guess all weights:** \((\text{polylog } n)^{O(k^4)} = n^{o(1)}\) blowup
Proof Outline – Step (2)

Removing weights by increasing the arity

\[ W = O(n^{f(k)}) \]
\[ B := \log n \]

**ExactWeight-\(k\)-Clique**

Given target \(t\), graph \(G\), weights \(w\), is there a \(k\)-clique of weight \(t\)?

**\(k\)-Clique**

4-hypergraph \(G'\)

2\(r\)-hypergraph

New problem after guessing carries:

Find \(k\)-clique \(C\) with \(c_\ell + \sum_{e \subseteq C} w_\ell(e) = t_\ell + c_{\ell+1} \cdot B\) \(\forall \ell\)

\[ \iff \sum_\ell (c_\ell + \sum_{e \subseteq C} w_\ell(e) - t_\ell - c_{\ell+1} \cdot B)^2 = 0 \]

\[ \iff \sum_\ell \left( \sum_{e \subseteq C} w'_\ell(e) \right)^2 = 0 \quad \text{with} \quad w'_\ell(e) := c_\ell + \binom{k}{2} w_\ell(e) - t_\ell - c_{\ell+1} \cdot B \]

\[ \iff \sum_{e_1, e_2 \subseteq C} \sum_\ell w'_\ell(e_1) \cdot w'_\ell(e_2) = 0 \]

\[ \iff \sum_{h \subseteq C, |h|=4} w''(h) = 0 \quad \text{with weights bounded by} \quad O \left( B^2 \frac{\log W}{\log B} \right) = \text{polylog } n \]

**guess all weights:** \((\text{polylog } n)^{O(k^2)} = n^{o(1)}\) blowup
Proof Outline – Putting it together

\begin{align*}
\text{Neg-}2k\text{-Clique} & \\ r\text{-hypergraphs} & \quad (1) \quad \text{ExactWeight-}2k\text{-Clique} & \\ r\text{-hypergraphs} & \quad (2) \quad 2k\text{-Clique} & \\ 2r\text{-hypergraphs} & \quad (3) \quad \text{OV} \\
\text{OV-Hypothesis: } (\text{moderate dimension}) & \\
\forall \varepsilon, \delta > 0: \text{OV in } d = n^\delta \text{ has no } O(n^{2-\varepsilon})\text{-time algorithm} & \\
\text{If OVH fails, then for some } \varepsilon, \delta \text{ OV is in time } O(n^{2-\varepsilon}) \text{ in } d = n^\delta & \\
\text{Then for any } r \text{ and } k \geq 2r/\delta, \text{ Neg-}2k\text{-Clique in } r\text{-hypergraphs} \text{ is in } O(|V|^{2k-\varepsilon k}) & 
\end{align*}
I. Equivalence of APSP and NegTriangle

II. Example Applications

III. Further Topics

IV. Conclusion
Conclusion

$2^{nd}$ Shortest Path ⇐ Maximum Submatrix

Min-Plus Product ⇐ All-Pairs-Negative-Triangle

APSP ⇐ Negative Triangle

Betweenness Centrality

Radius

Median

Open: Diameter equivalent to APSP?

Unifying hypothesis that implies OV-H, APSP-H and 3SUM-H?