Faster Negative-Weight Shortest Paths and Directed Low-Diameter Decompositions

Jason Li[1] Connor Mowry[2] Satish Rao[3]

[1]Carnegie Mellon University

[2]University of Illinois Urbana-Champaign

[3]UC Berkeley

SODA 2026

Negative-Weight Single-Source Shortest Paths

Input: Directed graph $G = (V, E, w)$ with $w : E \to \mathbb{Z}$ and source $s \in V$

Goal: Compute shortest paths from s to all vertices

Assumptions:

- No negative-weight cycles

Challenge: Dijkstra's algorithm requires non-negative edge weights

Our Result

$$O\Big((m + n \log\log n) \cdot \log(nW) \cdot \log n \log\log n\Big)$$

W = maximum absolute value of a negative edge weight

Algorithm	**Running Time**
Bellman-Ford [1958]	$O(mn)$
Gabow-Tarjan [1989]	$O(m\sqrt{n}\log(nW))$
Bernstein-Nanongkai-Wulff-Nilsen [2022]	$O(m\log^8 n \log W)$
Bringmann-Cassis-Fischer [2023]	$O((m + n\log\log n)\log(nW) \cdot \log^2 n)$
This paper	$O((m + n\log\log n)\log(nW) \cdot \log n \log\log n)$

Nearly $\log n$ factor improvement over [BCF'23]

Idea: Remove Negative Edges

If all edges are non-negative: Run Dijkstra in $O(m + n \log \log n)$

Johnson's reweighting: Transform edge weights using a potential function

Given $\phi : V \to \mathbb{Z}$, define:

$$w_\phi(u, v) = w(u, v) + \phi(u) - \phi(v)$$

Idea: Remove Negative Edges

If all edges are non-negative: Run Dijkstra in $O(m + n \log\log n)$

Johnson's reweighting: Transform edge weights using a potential function

Given $\phi : V \to \mathbb{Z}$, define:

$$w_\phi(u, v) = w(u, v) + \phi(u) - \phi(v)$$

Key property: For any path P from s to t:

$$w_\phi(P) = w(P) + \phi(s) - \phi(t)$$

$\Rightarrow$ Shortest paths are preserved!

Making All Edges Non-Negative

Observation: If $\phi(v) =$ shortest path distance from s to v, then:

$$w_\phi(u, v) = w(u, v) + \phi(u) - \phi(v) \geq 0$$

Why? Triangle inequality: $\phi(u) + w(u, v) \geq \phi(v)$

Making All Edges Non-Negative

Observation: If $\phi(v)$ = shortest path distance from s to v, then:

$$w_\phi(u, v) = w(u, v) + \phi(u) - \phi(v) \geq 0$$

Why? Triangle inequality: $\phi(u) + w(u, v) \geq \phi(v)$

The catch: Computing ϕ *is* the shortest path problem!

Our approach: Make incremental progress

- Halve the most negative weight
- Make only some edges non-negative

Outer Problem: Halving the Most Negative Weight [BNW'22]

Let $-W$ be the most negative edge weight in G

Define: $G_+ = G$ with:

- All weights increased by $W/2$
- Source s added with 0-weight edges to all vertices

Outer Problem: Halving the Most Negative Weight [BNW'22]

Let $-W$ be the most negative edge weight in G

Define: $G_+ = G$ with:

- All weights increased by $W/2$
- Source s added with 0-weight edges to all vertices

Key insight: If ϕ makes $(G_+)_\phi$ non-negative, then:

$$w_{G_\phi}(e) = w_{(G_+)_\phi}(e) - W/2 \geq -W/2$$

Outer Problem: Halving the Most Negative Weight [BNW'22]

Let $-W$ be the most negative edge weight in G

Define: $G_+ = G$ with:

- All weights increased by $W/2$
- Source s added with 0-weight edges to all vertices

Key insight: If ϕ makes $(G_+)_\phi$ non-negative, then:

$$w_{G_\phi}(e) = w_{(G_+)_\phi}(e) - W/2 \geq -W/2$$

Algorithm:

1. Compute ϕ making G_+ non-negative [Inner problem]
2. Apply ϕ to G [Most negative weight halved!]
3. Repeat $O(\log(nW))$ times until negative weights can be rounded to 0

Inner Problem: Making G_+ Non-Negative [BCF'23]

Goal: Compute ϕ such that $(G_+)_\phi$ has all non-negative edges

Recursive parameter: Diameter bound Δ

Decomposition Lemma

Delete few edges so that each SCC either:

- Has $\leq 3/4$ of the vertices, or
- Has diameter $\leq \Delta/2$

Inner Problem: Making G_+ Non-Negative [BCF'23]

Goal: Compute ϕ such that $(G_+)_\phi$ has all non-negative edges

Recursive parameter: Diameter bound Δ

Decomposition Lemma

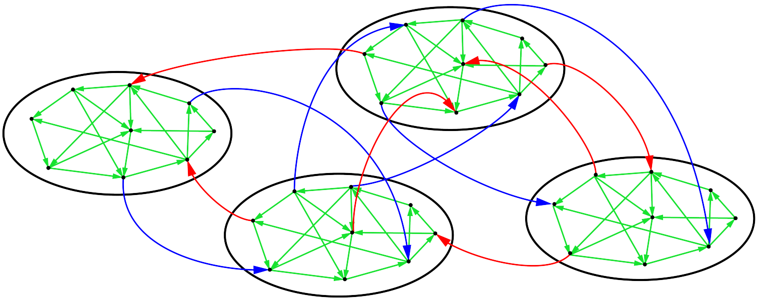
Delete few edges so that each SCC either:

- Has $\leq 3/4$ of the vertices, or
- Has diameter $\leq \Delta/2$

Algorithm:

1. Decompose
2. Recurse on SCCs to fix edges within SCCs
3. Fix DAG edges [Linear time]
4. Fix cut edges via Bellman-Ford/Dijkstra

Inner Problem: Decomposition Structure



Algorithm:

1. Decompose
2. Recurse on SCCs to fix edges within SCCs
3. Fix DAG edges [Linear time]
4. Fix cut edges via Bellman-Ford/Dijkstra

Bellman-Ford/Dijkstra Hybrid [BCF'23]

After DAG edges are non-negative: Only cut edges can be negative

BF/Dijkstra hybrid:

- Alternates Dijkstra iterations with Bellman-Ford relaxations
- After i iterations: found shortest paths using $\leq i$ negative edges

Bellman-Ford/Dijkstra Hybrid [BCF'23]

After DAG edges are non-negative: Only cut edges can be negative

BF/Dijkstra hybrid:

- Alternates Dijkstra iterations with Bellman-Ford relaxations
- After i iterations: found shortest paths using $\leq i$ negative edges

Running time depends on $\#$ cut edges on shortest paths

Loss factor $\ell(n)$: Each edge cut with probability $\leq w(e) \cdot \ell(n)/\Delta$

$\Rightarrow$ Expected cuts on path P: at most $w_{\geq 0}(P) \cdot \dfrac{\ell(n)}{\Delta}$ ($w_{\geq 0}$ = negative edges set to 0)

Bellman-Ford/Dijkstra Hybrid [BCF'23]

After DAG edges are non-negative: Only cut edges can be negative

BF/Dijkstra hybrid:

- Alternates Dijkstra iterations with Bellman-Ford relaxations
- After i iterations: found shortest paths using $\leq i$ negative edges

Running time depends on # cut edges on shortest paths

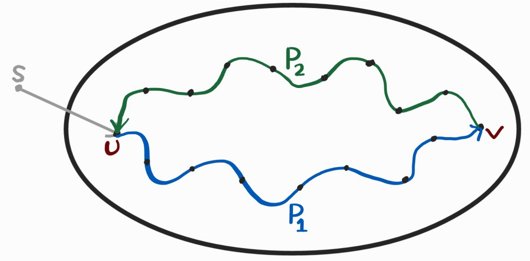
Loss factor $\ell(n)$: Each edge cut with probability $\leq w(e) \cdot \ell(n)/\Delta$

$\Rightarrow$ Expected cuts on path P: at most $w_{\geq 0}(P) \cdot \dfrac{\ell(n)}{\Delta}$ ($w_{\geq 0}$ = negative edges set to 0)

Key observation: In G_+, all shortest paths have $w_{\geq 0}(P) \leq \Delta$ (see next slide)

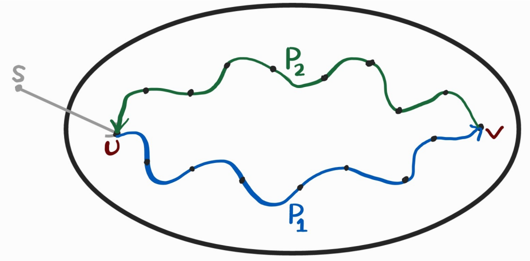
$\Rightarrow$ Expected cuts $\leq \ell(n)$

Key Observation: Bounding Positive Weight



Claim: In G_+, all shortest paths P have $w_{\geq 0}(P) \leq \Delta$

Key Observation: Bounding Positive Weight



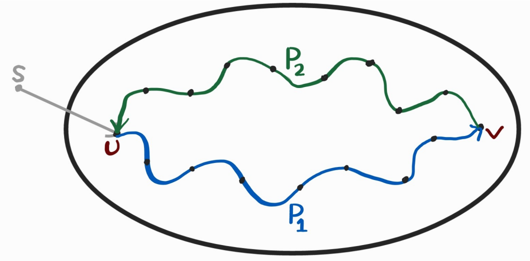
Claim: In G_+, all shortest paths P have $w_{\geq 0}(P) \leq \Delta$

Proof: Suppose $w_{\geq 0}(P) > \Delta$

- Write $P = s \to u \xrightarrow{P_1} v$
- $w(P) \leq 0$ (0-wt edge $s \to v$)
- So $w(P_1) \leq 0$ with $w_{\geq 0}(P_1) > \Delta$

$\Rightarrow$ Negative weight of $P_1 < -\Delta$

Key Observation: Bounding Positive Weight



Claim: In G_+, all shortest paths P have $w_{\geq 0}(P) \leq \Delta$

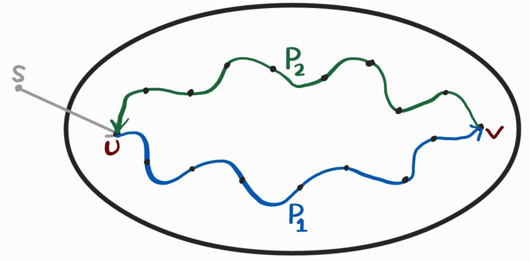
Proof: Suppose $w_{\geq 0}(P) > \Delta$

- Write $P = s \to u \xrightarrow{P_1} v$
- $w(P) \leq 0$ (0-wt edge $s \to v$)
- So $w(P_1) \leq 0$ with $w_{\geq 0}(P_1) > \Delta$

$\Rightarrow$ Negative weight of $P_1 < -\Delta$

In G: negative weights $\times 2$
$\Rightarrow w_G(P_1) < -\Delta$

Key Observation: Bounding Positive Weight



Claim: In G_+, all shortest paths P have $w_{\geq 0}(P) \leq \Delta$

Proof: Suppose $w_{\geq 0}(P) > \Delta$

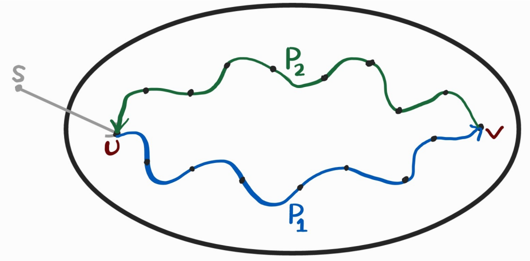
- Write $P = s \to u \xrightarrow{P_1} v$
- $w(P) \leq 0$ (0-wt edge $s \to v$)
- So $w(P_1) \leq 0$ with $w_{\geq 0}(P_1) > \Delta$

$\Rightarrow$ Negative weight of $P_1 < -\Delta$

In G: negative weights $\times 2$
$\Rightarrow w_G(P_1) < -\Delta$

But: $\exists$ path $P_2 : v \to u$ with $w_G(P_2) \leq \Delta$
(diameter bound)

Key Observation: Bounding Positive Weight



Claim: In G_+, all shortest paths P have $w_{\geq 0}(P) \leq \Delta$

Proof: Suppose $w_{\geq 0}(P) > \Delta$

- Write $P = s \to u \xrightarrow{P_1} v$
- $w(P) \leq 0$ (0-wt edge $s \to v$)
- So $w(P_1) \leq 0$ with $w_{\geq 0}(P_1) > \Delta$

$\Rightarrow$ Negative weight of $P_1 < -\Delta$

In G**:** negative weights $\times 2$

$\Rightarrow w_G(P_1) < -\Delta$

But: $\exists$ path $P_2 : v \to u$ with $w_G(P_2) \leq \Delta$ (diameter bound)

$P_1 + P_2$ is a negative cycle in G $\Rightarrow\Leftarrow$

[BCF'23] Running Time

Two sources of $O(\log^2 n)$ overhead: [[BCF'23] has $\ell(n) = O(\log n)$]

- **Decomposition:** $O(\log n)$ per level $\times$ $O(\log n)$ levels
- **BF/Dijkstra:** $O(\ell(n))$ expected cuts $\times$ $O(\log n)$ levels

[BCF'23] Running Time

Two sources of $O(\log^2 n)$ overhead: [[BCF'23] has $\ell(n) = O(\log n)$]

- **Decomposition:** $O(\log n)$ per level $\times$ $O(\log n)$ levels
- **BF/Dijkstra:** $O(\ell(n))$ expected cuts $\times$ $O(\log n)$ levels

To improve: Must reduce each to $O(\log n \log \log n)$

Low-Diameter Decomposition (LDD)

Definition

Delete random edges such that:

1. Each SCC has diameter $\leq \Delta$
2. Each edge cut with probability $\leq w(e) \cdot \ell(n)/\Delta$

Low-Diameter Decomposition (LDD)

Definition

Delete random edges such that:

1. Each SCC has diameter $\leq \Delta$
2. Each edge cut with probability $\leq w(e) \cdot \ell(n)/\Delta$

Our LDD achieves:

- Runtime $O((m + n \log\log n) \cdot \log n \log\log n)$
- Loss $\ell(n) = O(\log n \log\log n)$

Our New LDD

Two key improvements:

- CKR instead of geometric ball-growing
  - Process balls in random order [Calinescu-Karloff-Rabani]
- Preprocessing: heavy vertex elimination
  - Ensure all balls contain $\leq 75\%$ of edges

Results

Theorem

Directed LDD with loss $O(\log n \log\log n)$ *in expected time*

$$O((m + n \log\log n) \log n \log\log n)$$

Results

Theorem
Directed LDD with loss $O(\log n \log\log n)$ in expected time

$$O((m + n\log\log n)\log n \log\log n)$$

Theorem
Negative-weight SSSP in time

$$O((m + n\log\log n) \cdot \log(nW) \cdot \log n \log\log n)$$

Bonus: Direct negative cycle finding (no noisy binary search [BCF'23])

Summary

Main contribution: Faster directed LDD

- CKR ball-growing with random ordering
- Heavy vertex elimination preprocessing
- Loss $O(\log n \log\log n)$, matching Bringmann-Fischer-Haeupler-Latypov [2025]
- $O(\log^3 n)$ faster than [BFHL'25]

Application: Nearly $\log n$ factor speedup for negative-weight SSSP

Summary

Main contribution: Faster directed LDD

- CKR ball-growing with random ordering
- Heavy vertex elimination preprocessing
- Loss $O(\log n \log \log n)$, matching Bringmann-Fischer-Haeupler-Latypov [2025]
- $O(\log^3 n)$ faster than [BFHL'25]

Application: Nearly $\log n$ factor speedup for negative-weight SSSP

Open questions:

- **Directed LDD:** $O(\log n)$ loss? (matching undirected)
- **Negative-weight SSSP:** Near-linear time for non-integer weights?

Thank you!