Exact minimum-cost arborescences under a single arc failure

Yilin Liu

19 November 2026

Abstract

Every rooted weighted digraph on n vertices has a subgraph with $O(n^{3/2})$ arcs that contains a minimum-cost spanning arborescence after any feasible single arc deletion, answering a question of Dey and Kavitha. An oracle of the same size reports an optimal arborescence or a shortest list of arc replacements in time linear in the output. The construction uses shortest cycles in a weighted matroid exchange graph and admits deterministic subcubic preprocessing on simple digraphs. A reduction from directed distance preservation gives an $\Omega(n^{4/3}\log n)$-bit lower bound for reporting on simple digraphs with input arc identifiers, even with constant per-query error.

hexagon:2611.00665v1

1 Introduction

Let $G = (V, E)$ be a directed multigraph with root r, n vertices, m arcs, and a distinct identifier for each arc. A spanning arborescence is a directed spanning tree reaching every vertex from r. For fixed arc costs, an exact *preserver* is a subgraph $H \subseteq G$ such that $H - F$ contains a minimum-cost arborescence of $G - F$ whenever one exists, for every allowed failure set F, including the empty set. Queries are independent.

Dey and Kavitha [7, Theorem 1.1 and Section 5] give $O(n^{3/2})$-arc preservers for factor-two approximation and ask for sparse exact preservers. We obtain exact preservation at the same size and preprocessing bounds and report optimal arborescences in linear time.

Theorem 1.1 (Arborescence preservation)**.** *For real costs, one can deterministically construct an exact single-arc-failure preserver of $O(n^{3/2})$ arcs and an oracle of $O(n^{3/2})$ records. The oracle reports an optimum arborescence or infeasibility in $O(n)$ time. For a feasible query, it can report the additions and deletions relative to a fixed initial optimum in $O(1+t)$ time, where t is the fewest replacements among minimum-cost answers. Preprocessing takes*

$$O\Big(n + m + \min\{mn + n^2\log(2n),\ mn\alpha(n) + n^2\log\log(4n),\ n^{2.9929729}\}\Big)$$

operations, where α is the inverse Ackermann function and $\log^*$ is the iterated logarithm. For polynomially bounded integer costs the first two terms in the minimum become $mn+n^2\log^*(2n)$ and $mn\alpha(n)$. The preserver itself can be constructed in $O(n+m+\min\{mn, n^{2.9929729}\})$ time.

We count exact cost operations and word-RAM operations with multiplication and division. Each identifier fits one word. For rational costs the bit complexity is polynomial. Root reachability detects initial infeasibility. At $n = 1$ use the empty arborescence. Discard loops and arcs into r.

The reduction applies to two matroids on a common ground set E.

Theorem 1.2 (Weighted matroid intersection)**.** *Given a minimum-cost common independent set of cardinality $k \geq 1$, one can deterministically retain $O(k^{3/2})$ elements preserving the optimum at cardinality k before and after every single-element deletion. An oracle of $O(k^{3/2})$ records reports an optimum or infeasibility in $O(k)$ time. Preprocessing uses $O(|E|k)$ independence tests and $O(|E|k^2)$ further operations. At $k = 0$ return the empty set.*

2 Exact replacements from exchange cycles

Truncate the matroids M_1, M_2 to rank k and fix a minimum-cost common basis B. Write $c(B) = \sum_{e \in B} c(e)$. The exchange graph has vertex set B. For $e \notin B$ and $a, b \in B$, it contains an arc $a \to b$ labeled by e if $B - a + e$ is a basis of M_1 and $B - b + e$ is a basis of M_2. The arc has length $\ell(a, b; e) = (c(e) - c(a), 1)$. Lengths are added coordinatewise and compared lexicographically. Loops are allowed.

Lemma 2.1 (Exchange cycles)**.** *Every directed cycle has positive length. For $f \in B$, a common basis avoiding f exists exactly when a directed cycle through f exists. A shortest such cycle has distinct labels and replacing its vertices in B by its labels gives a minimum-cost common basis avoiding f with the fewest outside elements. Conversely, the exchange bijections from any such basis produce a single shortest cycle through f.*

Proof. For a simple cycle C of t arcs and $0 < \eta \leq 1/t$, form

$$x = (1 - t\eta)\mathbf{1}_B + \eta \sum_{a \to b \text{ in } C} \mathbf{1}_{B-a+e},$$

where e labels the corresponding arc. Replacing $B - a + e$ by $B - b + e$ leaves x unchanged, since the heads and tails form the same multiset. Thus x lies in both base polytopes and is a convex combination of common-basis indicators by matroid intersection [8].

Set

$$\Delta(x) = \left(c \cdot x - c(B), \sum_{e \notin B} x_e \right).$$

On every common basis this objective is lexicographically nonnegative. Since $\Delta(x) = \eta\ell(C)$, the cost coordinate of $\ell(C)$ is nonnegative. Its second coordinate is $t > 0$.

If $f \in V(C)$, then $x_f = 1 - \eta$, so the bases avoiding f in the combination have total coefficient η. At least one has objective at most $\ell(C)$, because the remaining bases have nonnegative objective. Every basis in the combination retains $B \setminus V(C)$ and draws its outside elements from the labels of C, by the coordinates of x equal to one and zero.

Conversely, let J be a common basis avoiding f. For each matroid there is a bijection $e \mapsto a_i(e)$ from $J \setminus B$ to $B \setminus J$ with $B - a_i(e) + e$ a basis. Indeed, after contracting $B \cap J$, the exchange neighbors of any subset of the remaining elements of J span that subset by fundamental circuits. The rank inequality gives Hall's condition. The arcs $a_1(e) \to a_2(e)$ labeled by e form vertex-disjoint cycles of total length $\Delta(\mathbf{1}_J)$. Positivity makes the cycle through f no longer than this total, with strict inequality if another cycle occurs.

Hence the shortest cycle length equals the minimum replacement objective. On a shortest t-arc cycle the supported optimum must use t outside elements, forcing all t labels to be distinct and used. For a minimizing J, the strict inequality above is impossible, so its bijections produce a single shortest cycle. □

Keeping one cheapest arc per ordered pair leaves shortest cycle lengths unchanged. We use this compression when building the exchange graph explicitly.

3 Sparsifying the replacement paths

Split each exchange vertex a into a^- and a^+, joined by a zero-length arc $a^- \to a^+$. Replace $a \to b$ by $a^+ \to b^-$, keeping its length and label. A simple path from f^+ to f^- represents a cycle through f. Every nonempty cycle in the split graph has positive length.

Lemma 3.1 (Path storage)**.** *Suppose every directed cycle in an N-vertex digraph has positive length. Given one shortest path for each of $p \le N$ finite demands, one can select at most $O(\sqrt{p})$ vertices and designate some paths for explicit storage in $O(Np)$ time when $p \ge 1$. The explicit paths and the shortest-path trees to and from these vertices use $O(N\sqrt{p})$ arcs and records. They report a shortest demand path in time linear in its length plus one. For $p = 0$ no records are needed.*

Proof. Use the short-path and hub construction [5, 3]. A phase with q remaining demands stores their paths of at most $L = \lceil N/\sqrt{q} \rceil$ arcs. Greedily select $\lceil N/L \rceil$ vertices, or stop when all longer paths are hit. Each choice hits at least an L/N fraction of the unhit longer paths, so at most $q \exp(-1)$ demands remain. The phase stores $O(qL + N^2/L + N) = O(N\sqrt{q})$ arcs and records and selects $O(\sqrt{q})$ vertices. Sum over the phases.

Each hit demand points to a selected vertex on its path. Concatenating a shortest path into that vertex and one out of it preserves the distance. A repeated vertex would yield a strictly positive cycle, so the concatenation is simple. Follow the two trees and reverse only the portion read backwards.

Maintain an incidence list for each vertex and place vertices in buckets indexed by their counts of unhit paths. Counts only decrease. A path's first hit decrements its incidences in constant time each, and a maximum-count pointer scans at most $q + 1$ buckets in a phase. The total incidence and initialization costs are $O(Np)$. □

Apply Lemma 3.1 to the finite demands (f^+, f^-). The preserver contains B and the original labels of the stored arcs. Store the identifiers of B in a deterministic static dictionary with constant lookup time, built in $O(k^{3/2})$ time and linear space [15]. Return B or the empty change list for no failure or an absent identifier. For $f \in B$, an infinite distance signals infeasibility. A path with t labeled arcs has $2t - 1$ arcs. The labels give the additions and their tails give the deletions. Lemma 2.1 gives the minimum change count. To report the full basis, scan B with the deletions marked and append the additions.

For Theorem 1.2, enumerate both exchange lists of each outside element in $O(|E|k)$ independence tests. Updating the cheapest arc for each eligible pair costs $O(|E|k^2)$ operations. Dense all-pairs shortest paths and their trees take $O(k^3)$ time, absorbed since $k \le |E|$. Choosing a tight predecessor or successor at each reachable vertex gives a tree, since cycles have positive length.

4 Preprocessing for arborescences

Use the graphic matroid on the underlying undirected multigraph and the partition matroid with capacity one per nonroot head. A common independent set of size $n - 1$ is a spanning tree with one incoming arc per nonroot vertex, hence a rooted arborescence. Fix an optimum T with arc b_v entering v. Each non-tree arc $e = (u, v)$ gives a transition $a \to b_v$ for each edge a of the undirected tree path $P_T(u, v)$.

Compute T by Gabow, Galil, Spencer, and Tarjan's implementation of Edmonds' algorithm [11, Section 3]. It takes $O(m + n \log(2n))$ time using comparisons and additive cost updates. A tail-indexed array retains the cheapest arc per ordered pair in a sweep by head. Cost-zero dummy arcs into r give strongly connected input to the algorithm.

4.1 Exchange graph representation

Path blocks. Subdivide T, placing an edge atom at each midpoint and the empty set at each original vertex. Apply Alon and Schieber's tree path algorithm [1, Section 3] with set union. Represent each nonempty preprocessing product by a block with zero-length arcs from its nonempty operands. Traversing the temporary gates of a query gives stored blocks whose union is the queried path. Connect each exchange copy a^+ to its singleton block with length $(-c(a), 0)$. For each non-tree arc $e = (u, v)$, connect the blocks for $P_T(u, v)$ to b_v^- with length $(c(e), 1)$ and label e. Keep the zero-length arcs $a^- \to a^+$.

The linear-space scheme with $O(\alpha(n))$ query time gives $N = O(n)$ vertices and $A = O(n + m\alpha(n))$ arcs. With $k = 4$, the constant-query scheme gives $N = O(n \log^*(2n))$ and $A = O(m + n \log^*(2n))$. Either graph takes $O(N + A)$ time to build. The tree edges that reach a block are exactly its atoms. Thus walks between exchange copies project to walks of equal length in the split exchange graph, and split-graph walks lift to the block graph.

Potentials. Record the minimum entering cost y_S subtracted from each component S when it is merged into a contracted cycle. The algorithm ends with one component, and the sets S are laminar. Subtractions are nonnegative for nonsingletons, whose entering arcs have nonnegative residual costs at formation. An arc's residual cost freezes at the first contraction containing both endpoints. During contraction, parallel arcs discarded earlier have residual cost at least that of their retained representative. Thus for every $e = (u, v)$,

$$c(e) - p_v \;\geq\; -\sum_{S \supseteq \{u,v\}} y_S, \qquad p_v = \sum_{S \ni v} y_S.$$

The arcs retained in T freeze at zero, so they attain equality. Expansion keeps $T[S]$ connected by dropping one arc of each contracted cycle. If a lies on $P_T(u, v)$, every component containing u, v contains both endpoints of a. Therefore $c(e) - p_v \geq c(a) - p_{\mathrm{head}(a)}$. Use potential $p_{\mathrm{head}(a)}$ on both copies of a and $\min_{a \in Q}(p_{\mathrm{head}(a)} - c(a))$ on block Q. All reduced lengths are lexicographically nonnegative. One traversal of the contraction hierarchy and a sweep of the block DAG compute these potentials in $O(n + N)$ time.

4.2 Sparse preprocessing

Demand paths. Test reachability in $G - f$ for each $f \in T$. On feasible instances minimize $(c(e), \mathbf{1}[e \notin T])$ to obtain J in $O(m + n \log(2n))$ time, retaining the cheapest surviving arc per ordered pair under these costs.

The partition exchanges match arcs by head. For the graphic bijection, maintain a tree S initially equal to J, with fixed edges $F = T \cap J$. Process edges $a = (p, x) \in T \setminus J$ in postorder of T. The F-component of x is its descendant subtree in T, since all missing edges below x have been processed. Choose the first edge $e \notin F$ on the x-to-p path of S. It crosses the cut of a in T, so $T - a + e$ is a tree. Match a to e, replace e by a in S, and add a to F. Each removed edge is a distinct member of $J \setminus T$. A link-cut tree [16] supports the search and swap in $O(\log(2n))$ amortized time. Expose the path, descend using a subtree flag for nonfixed edges, and splay the edge found.

Lemma 2.1 turns the two bijections for J into a shortest demand path. All demand paths take $O(mn + n^2 \log(2n))$ time, and Lemma 3.1 selects $O(\sqrt{n})$ vertices. Run Dijkstra with Fibonacci heaps [10] in both directions from those vertices in the reweighted block graph with constant-time path queries. Contract each tree's block portions to labeled transitions. A traversal tracks the nearest exchange copy and the intervening label, leaving $O(n)$ records in $O(N)$ time. All selected trees take $O(\sqrt{n}\,(A + N \log(2N)))$ time, within the same bound.

Distance computation. Compute the demand paths and selected trees from distances in a block graph. Append a hop coordinate of one to each reduced length and add a directed cycle of arcs with length $(1+NC_{\max}, 0, 1)$, where $C_{\max}$ is the largest reduced cost coordinate. Merge parallel arcs in $O(N + A)$ time. Pettie's comparison-addition algorithm [14] on the linear-size graph takes $O(nA + n^2 \log\log(4n))$ time. Coordinatewise additions and lexicographic comparisons implement the infinitesimal tie break. Below the added cost threshold, the first two coordinates give the reweighted distances. Tight-arc scans and contraction recover paths and trees within the same time.

For integer costs, pack a length $(a, b, 1)$ as $(N + 1)^2 a + (N + 1)b + 1$. This preserves the order of simple paths, whose lengths use at most $N - 1$ arcs. If $|c(e)| \le C$, singleton subtractions have magnitude at most C and each nonsingleton subtraction is at most $2C$. Thus polynomial input costs give polynomial packed lengths. Orlin and Végh [13] spend $O(A\sqrt{N}\log(2N))$ time on preprocessing and $O(A)$ per source. Run from the exchange copies in both directions. For either block graph this costs $O(nA)$, giving the two integer bounds.

4.3 Dense preprocessing

For each head $v \neq r$, root the undirected tree T at v. At u, place the cheapest non-tree arc from u to v, if any. For an edge a, the candidate tails for $a \to b_v$ form the subtree cut off from v by a. A bottom-up pass finds the cheapest arc for each transition. Grouping by head gives the compressed graph in $O(m + n^2)$ time. Dense all-pairs shortest paths with witnesses and Lemma 3.1 give $O(m + n^3)$ time.

Subcubic multiplication. The min-plus algorithm of Essayag and Zabokritskiy [9, Theorem 1.1] extends to lexicographic lengths and returns minimizing indices. For input pairs A_{ih}, B_{hj}, compare candidates h, q via

$$(A_{ih} - A_{iq},\, h - q) < (B_{qj} - B_{hj},\, 0).$$

This orders the sums lexicographically by cost, count and index. The sorted keys supply the rank and counting interfaces of their Sections 2 and 5 with $O(\log N)$-bit integer coordinates. Their final minimization identifies the winning index and length pair.

Min-plus squaring gives the lexicographic distance matrix D, using zero diagonal entries to allow shorter walks. For the arc matrix W of the N-vertex split exchange graph, $W \star D$ and $D \star W$ give first and last hops for finite distances between distinct vertices. Encode absent entries of W and infinite distances in D by $((4N + 1)U, 0)$, where $U \ge 1$ bounds the absolute cost coordinates in W. A walk with such an entry and at most $2N$ steps costs more than NU, which bounds all finite shortest simple path costs. The same threshold separates the outputs of the witness products. Cycle positivity makes the selected hops form shortest-path trees. The squarings and witness products retain the cited time exponent, giving $O(m + n^{2.9929729})$ preprocessing for real costs.

To construct the preserver alone, return G when $m \le n^{3/2}$. For larger m, the first sparse bound is $O(mn)$.

5 Size equivalence and lower bounds

Let $\mathsf{D}(N, p)$ be the worst-case minimum size of a subgraph preserving exact distances for at most p designated ordered pairs in an N-vertex digraph with nonnegative real weights. Let $\mathsf{A}(n)$ be the worst-case minimum size of a single-failure arborescence preserver with real costs.

Theorem 5.1 (Equivalence of preserver sizes)**.** *For $n \geq 2$ and $N, p \geq 1$,*

$$\mathsf{A}(n) \leq n - 1 + \mathsf{D}(2n - 2, n - 1), \qquad \mathsf{D}(N, p) \leq \mathsf{A}(N + p + 1).$$

Proof. For the first inequality, use the compressed split exchange graph. The simple paths form a finite set, so $(a, b) \mapsto a + \varepsilon b$ preserves their order for sufficiently small $\varepsilon > 0$. Cycles stay positive, and a shortest-path potential makes the scalar weights nonnegative. Retain the labels in a distance preserver for the $n - 1$ demand pairs, together with T. The empty subgraph suffices when the initial instance is infeasible.

For the reverse inequality, adapt the backbone construction of [4]. For the $q \leq p$ finite demands (s_i, t_i), add a disjoint zero-cost path $x_0 \to \cdots \to x_q$. Choose M larger than the cost of any simple path in the original graph. Add $x_{i-1} \to s_i$ of cost $(q + 1 - i)M$ and $t_i \to x_i$ of cost iM.

When $x_{i-1} \to x_i$ fails, an x_0-to-x_q path must make an excursion from x_{a-1} to x_b through the original graph, with $a \leq i \leq b$. Its connectors cost $(q + 1 + b - a)M$. Each further excursion costs at least $2M$. Thus any path of cost below $(q + 2)M$ uses just the i-th pair of connectors. The minimum is $(q + 1)M + \operatorname{dist}(s_i, t_i)$.

Root at x_0 and add zero-cost arcs from x_q to all other vertices except the root. They complete any simple x_0-to-x_q path to an arborescence at the same cost. Conversely, nonnegative costs bound the tree cost below by its path cost. Restricting an arborescence preserver to the original arcs therefore preserves the demand distances. The case $q = 0$ is trivial, and zero-cost leaves at x_q pad the instance to $N + p + 1$ vertices. □

Theorem 5.2 (Space lower bounds)**.** *On simple digraphs with polynomially bounded nonnegative integer costs, single-failure preservers can require $\Omega(n^{4/3})$ arcs. An oracle reporting from its stored state and the failed identifier can require $\Omega(n^{4/3} \log n)$ bits on simple digraphs with fixed topology and costs and varying arc identifiers. With fixed identifiers and varying costs, the reporting bounds are $\Omega(n^{4/3})$ bits for simple digraphs and $\Omega(n^{4/3} \log n)$ bits for multigraphs. These bounds apply to full and replacement-list reporting with per-query error at most $1/3$. For two arc failures, exact preservers have worst-case size $\Theta(n^2)$. With arbitrary nonnegative integer costs, the same size is necessary for every fixed multiplicative approximation factor.*

Proof. Use the layered distance-preserver construction [5, 4]. For $L \geq 2$ take vertices (ℓ, x) with $0 \leq \ell \leq L$ and $0 \leq x < 2L^2$. Put an arc of cost j^2 from (ℓ, x) to $(\ell + 1, x + j)$ whenever both endpoints exist and $1 \leq j \leq L$. The demand from $(0, x)$ to $(L, x + Lj)$ has a unique shortest path with all increments j, by strict convexity of the square. The L^3 demands indexed by $0 \leq x < L^2$ and $1 \leq j \leq L$ have arc-disjoint paths, since each of their arcs determines x, j. Their L^4 arcs are forced in the backbone reduction, which has $\Theta(L^3)$ vertices.

For each forced arc, reserve a private pool of L identifiers. Choose its identifier independently from that pool. The universe of identifiers has polynomial size, with all other identifiers fixed. The optimum before failure is the common zero-cost tree of the backbone and its terminal completion. After a backbone failure, the root-to-terminal path of any optimum reveals the identifiers on its demand path. Applying a replacement list to the common optimum recovers the same tree.

Let X_i be the vector of independent uniform choices revealed by demand i, with s options per arc, and let U and R denote the oracle state and input-independent random coins. The success guarantee and Fano's inequality give $H(X_i \mid U, R) \leq 1 + \frac{1}{3} L \log_2 s$. By independence and subadditivity, $H(U \mid R) \geq L^3(\frac{2}{3} L \log_2 s - 1)$. Taking $s = L$ gives the identifier bound.

For fixed identifiers in a simple digraph, give each vertex two clones with unit-cost arcs to it. Replace a transition to that vertex by two arcs to its clones. Choose one of cost j^2 and give the other cost $j^2 + 1$. This adds a common unit cost per transition and encodes one bit in the leading arc of each forced transition. In a multigraph use L parallel arcs with one of cost j^2 and all others of cost $j^2 + 1$. Both constructions keep the common zero optimum and use a fixed M exceeding all simple path costs. Their demand paths recover the choices with $s = 2$ and $s = L$.

For two failures, take zero-cost backbones $u_0 \to \cdots \to u_L$ and $v_L \to \cdots \to v_0$. Add $u_a \to v_b$ of cost $2L - a - b$ for $0 \le a, b < L$. After deleting $u_i \to u_{i+1}$ and $v_{j+1} \to v_j$, a surviving u_0-to-v_0 path can use only $a \le i$ and $b \le j$. Its unique cheapest crossing is $u_i \to v_j$. Complete paths to arborescences by zero-cost arcs out of v_0 as in Theorem 5.1. All L^2 crossing arcs are forced by varying the two failures. This is the weighted replacement-path construction of [4] with the terminal completion. For a fixed approximation factor $\alpha \ge 1$, use crossing costs K^{2L-a-b} for an integer $K > \alpha$. Every other crossing available to a root-to-terminal path costs at least K times the optimum, so the same arcs are forced by an α-approximate preserver. These costs use $O(n \log K)$ bits per arc. For the matching upper bound, retain the three cheapest arcs of each ordered pair. After at most two failures, every discarded arc in a surviving optimum can be replaced by a surviving retained arc of no greater cost with the same endpoints. □

Theorem 5.3 (Preprocessing lower bound)**.** *Under the directed k-Cycle Hypothesis [6, Section 5], the worst-case deterministic preprocessing exponent of an exact oracle with $O(1+t)$ replacement reporting is at least two, even on simple digraphs with $O(n)$ arcs and polynomially bounded nonnegative integer costs.*

Proof. Use deterministic color-coding [2] for each fixed k. For a coloring with k colors, keep arcs advancing to the next color in cyclic order. Split each vertex v of color 1 into a source s_v and a terminal t_v. In this layered graph a path from s_v to t_v has exactly k unit-cost arcs and certifies a directed k-cycle. Use a family of polylogarithmic size that colors every k-set injectively and try all color orders.

Apply the backbone construction to all q pairs (s_v, t_v), with $M = k+1$. At each backbone arc add a two-arc backup through a fresh vertex, with costs $(q+2)M$ and zero, and include the fresh vertices in the terminal completion. The unique zero-cost optimum uses the backbone. After its i-th arc fails, a path of cost below $(q+2)M$ must use the i-th connector pair, so its optimum cost is $(q+1)M + k$ if the demand is reachable. Otherwise the backup gives cost $(q+2)M$. In either case the path contains all other backbone arcs and completes to a tree with at most $k+2$ replacements. Thus the sum of the added costs in each replacement report tests reachability in $O(k)$ time.

For an input with m arcs, discard isolated vertices and pad each constructed graph with zero-cost terminal leaves to $\Theta(m)$ vertices. It is simple with $O(m)$ arcs. Preprocessing in $O(n^{2-\varepsilon})$ for a fixed $\varepsilon > 0$ would decide directed k-Cycle for every fixed k in $\widetilde{O}(m^{2-\varepsilon})$ time, contradicting the hypothesis. □

Acknowledgment. The author used OpenAI models to formulate results, develop and review proofs, draft the manuscript, and implement finite checks. The source manuscripts [12] were prepared with GPT-6 Astra Pro.

References

[1] N. Alon and B. Schieber. Optimal preprocessing for answering on-line product queries. Technical Report 71/87, Tel Aviv University (1987).

[2] N. Alon, R. Yuster and U. Zwick. Color-coding. *Journal of the ACM* 42(4) (1995), 844–856.

[3] G. Bodwin. A note on distance-preserving graph sparsification. *Information Processing Letters* 174 (2022), article 106205.

[4] G. Bodwin, F. Grandoni, M. Parter and V. Vassilevska Williams. Preserving distances in very faulty graphs. *ICALP*, LIPIcs 80 (2017), 73:1–73:14.

[5] D. Coppersmith and M. Elkin. Sparse sourcewise and pairwise distance preservers. *SIAM Journal on Discrete Mathematics* 20(2) (2006), 463–501.

[6] M. Dalirrooyfard and V. Vassilevska Williams. Conditionally optimal approximation algorithms for the girth of a directed graph. *ICALP*, LIPIcs 168 (2020), 35:1–35:20.

[7] D. Dey and T. Kavitha. Low-cost arborescence under edge faults. arXiv:2605.13800v1 (2026).

[8] J. Edmonds. Submodular functions, matroids, and certain polyhedra. In *Combinatorial Structures and Their Applications* (1970), 69–87.

[9] I. B. Essayag and A. L. Zabokritskiy. Faster deterministic all-pairs shortest paths with real edge weights. arXiv:2611.17215 (2026).

[10] M. L. Fredman and R. E. Tarjan. Fibonacci heaps and their uses in improved network optimization algorithms. *Journal of the ACM* 34(3) (1987), 596–615.

[11] H. N. Gabow, Z. Galil, T. Spencer and R. E. Tarjan. Efficient algorithms for finding minimum spanning trees in undirected and directed graphs. *Combinatorica* 6(2) (1986), 109–122.

[12] Yilin Liu and GPT-6 Astra Pro. Exact single-failure preservers. *Algorithm Frontier*, source snapshot `77140c2` (2026).

[13] J. B. Orlin and L. A. Végh. Directed shortest paths via approximate cost balancing. *Journal of the ACM* 70(1) (2022), article 3.

[14] S. Pettie. A new approach to all-pairs shortest paths on real-weighted graphs. *Theoretical Computer Science* 312(1) (2004), 47–74.

[15] M. Ružić. Uniform deterministic dictionaries. *ACM Transactions on Algorithms* 4(1) (2008), article 1.

[16] D. D. Sleator and R. E. Tarjan. A data structure for dynamic trees. *Journal of Computer and System Sciences* 26(3) (1983), 362–391.