65 Years After Dijkstra: An Analytical Perspective on Breaking the Sorting Barrier

Authored by: Shibani Selvakumar

**Abstract**

The single-source shortest paths (SSSP) problem has been central to algorithmic research since the 1950s. Classical deterministic methods such as Dijkstra’s algorithm have long been constrained by a sorting barrier, with runtime in sparse graphs, while Bellman–Ford offered generality at the cost of efficiency. Recent advances using randomization surpassed Dijkstra’s bound but lacked deterministic guarantees. In 2025, Duan, Mao, Mao, Shu, and Yin introduced the first deterministic algorithm to break this barrier, achieving time in the comparison–addition model. This article traces the historical development of SSSP algorithms, analyzes the conceptual significance of the breakthrough, and reflects on its broader implications for algorithm design and complexity theory.

**1. Introduction**

Shortest path algorithms are foundational in both theory and practice. They underpin applications ranging from internet routing and road navigation to compiler optimization and resource scheduling. The single-source shortest paths problem (SSSP) in particular asks: given a directed weighted graph with non-negative edge weights, how can we efficiently compute the minimum distance from a source vertex to every other vertex?

For decades, the fastest known deterministic solution for sparse graphs was Dijkstra’s algorithm, with runtime when implemented with efficient priority queues. The logn term arises from the need to maintain a perfectly ordered priority queue of tentative distances — a process fundamentally tied to sorting. This “sorting barrier” was widely believed to be unavoidable in deterministic computation.

Randomized methods in the 2020s challenged this assumption by achieving sub-performance in expectation. Yet, their reliance on probabilistic guarantees left open the fundamental question: **can deterministic algorithms do better?** The 2025 breakthrough by Duan et al. provides a compelling answer, shattering a sixty-year-old barrier.

**3. Historical Background**

The development of shortest path algorithms can be understood as a gradual evolution in balancing efficiency, generality, and determinism.

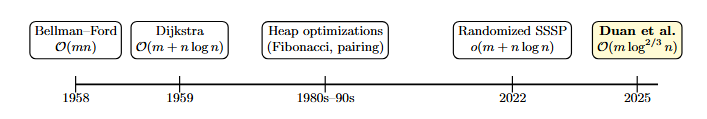


Figure 1: Milestones in single-source shortest paths (SSSP)

* **1958 — Bellman–Ford.** Introduced as a general algorithm capable of handling negative weights. Its runtime of is prohibitive in practice, particularly for sparse graphs.
* **1959 — Dijkstra.** A more efficient solution for non-negative weights, achieving Its reliance on strict priority queue ordering tied it inherently to sorting.
* **1980s–1990s — Heap Optimizations.** Advances in data structures, such as Fibonacci heaps and pairing heaps, reduced constants but failed to escape the overhead.
* **2020s — Randomized Breakthroughs.** Works presented at FOCS 2022 and STOC 2024 achieved runtimes below but only with high probability.
* **2025 — Deterministic Barrier Broken.** Duan et al. achieve , the first deterministic improvement in more than half a century.

**4. The 2025 Breakthrough**

The key result of Duan et al. (2025) is a deterministic algorithm for SSSP on directed graphs with non-negative edge weights that runs in time under the comparison–addition model.

At a high level, the algorithm avoids the full precision of global sorting. Instead, it employs a layered relaxation process that groups vertices into structured intervals. By relaxing vertices in batches rather than strict order, the algorithm ensures correctness while reducing dependence on priority queue operations.

This result is significant for two reasons. First, it breaks a barrier that had withstood sixty-five years of research, proving that shortest path computation is not inherently tied to sorting. Second, it restores balance in the field by showing that deterministic methods can compete with — and surpass — randomized approaches.

**5. Analysis**

**5.1 Conceptual Contrast**

The contrast between Dijkstra’s classical algorithm and the 2025 breakthrough is more than a matter of complexity bounds, it is a difference in **algorithmic worldview**. Dijkstra embodies a sorting-driven paradigm: every iteration hinges on extracting the globally minimum tentative distance from a priority queue, then relaxing its outgoing edges. This exact global ordering is what guarantees correctness, but it also anchors the algorithm’s runtime to the cost of maintaining a fully sorted structure.

By contrast, the new approach of Duan et al. embraces *approximate ordering*. Instead of insisting on a perfectly sorted priority queue, vertices are grouped into **layers or intervals** that represent approximate distance ranges. Within each layer, vertices are relaxed in batches rather than individually, and only when finer distinctions become necessary are groups refined. This subtle shift eliminates the need for strict global ordering at every step, while still preserving the correctness of the final distances.

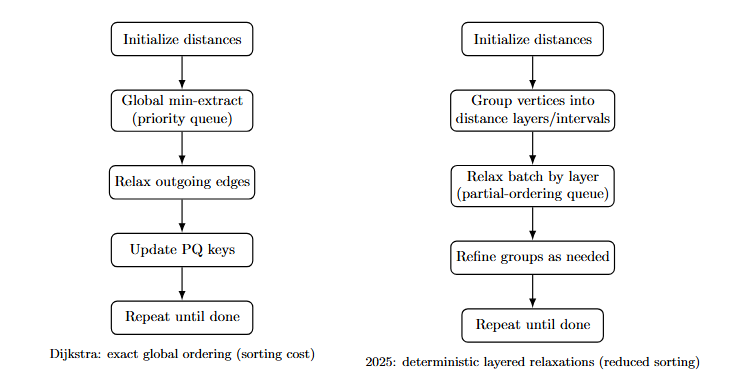


Figure 2: Conceptual contrast: Dijkstra’s global sorting vs. deterministic layered relaxations.

The illustration is not pseudocode, but captures the **conceptual leap**: shortest paths do not require perfect sorting. By reducing reliance on global order, the new approach cuts the logarithmic factor while preserving determinism.

**5.3 Runtime Growth**

The asymptotic implications of this shift are depicted in Figure 3. On sparse graphs (where ), Bellman–Ford’s quadratic growth quickly becomes infeasible, while Dijkstra’s runtime dominated the deterministic frontier for decades. Randomized methods dip just below Dijkstra, but only in expectation. Duan et al.’s curve lies beneath Dijkstra’s, demonstrating the first provable deterministic improvement.

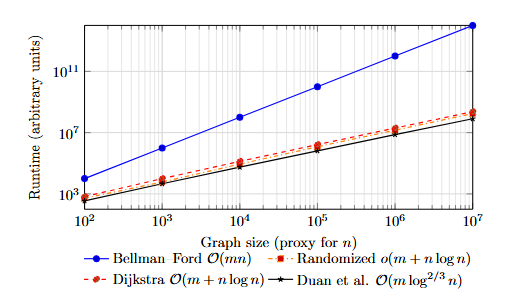


Figure 3: Schematic runtime growth (log–log) (Curves illustrate asymptotics)

This schematic underscores the long-term impact: while the absolute gain may be modest for small instances, asymptotically the separation grows. It positions the 2025 algorithm as a new **theoretical baseline** for deterministic SSSP.

**5.2 Complexity Landscape**

Traditional algorithms reflect fundamentally different philosophies. Bellman–Ford embodies brute-force relaxation, guaranteeing correctness even with negative weights but at prohibitive cost. Dijkstra epitomizes greedy optimization, but is chained to sorting through its priority queue. Randomized methods introduced in the 2020s sidestepped exact ordering, but only by gambling on probability, leaving open whether determinism could compete.

Duan et al.’s contribution is the first to show that **determinism can break the glass ceiling**: shortest paths do not require perfect sorting. By grouping vertices and processing them in carefully structured layers, the algorithm achieves progress without precise global order.

| **Algorithm** | **Year** | **Runtime** | **Handles Negatives?** | **Deterministic/ Randomized** | **Practical Relevance** |
| --- | --- | --- | --- | --- | --- |
| Bellman–Ford | 1958 |  | Yes | Deterministic | Correct but impractical for large graphs |
| Dijkstra | 1959 |  | No | Deterministic | Long-standing baseline for sparse graphs |
| Heap Optimizations | 1980s–90s |  | No | Deterministic | Improved performance but same asymptotic bound |
| Randomized SSSP | 2022–24 |  | No | Randomized | Theoretical progress, less robust in practice |
| **Duan et al.** | **2025** |  | **No** | **Deterministic** | **First deterministic breakthrough in 65+ years** |

The illustration is not pseudocode, but captures the **conceptual leap**: shortest paths do not require perfect sorting. By reducing reliance on global order, the new approach cuts the logarithmic factor while preserving determinism.

**5.2 Critical Reflections**

Despite its historic contribution, the 2025 breakthrough is not without boundaries. A deeper look reveals both strengths and caveats:

1. **Weight Restrictions:** Like Dijkstra’s algorithm, the new method requires non-negative edge weights. Extending determinism beyond this constraint — to handle negative weights efficiently — remains an open frontier.
2. **Asymptotic vs. Practical Efficiency:** The runtime improvement from to is asymptotically elegant. However, hidden constants may blunt practical gains on real-world graphs of modest size. The true test will come with implementations.
3. **Conceptual Victory:** Most importantly, the algorithm severs the assumed equivalence between deterministic shortest paths and sorting. This is a philosophical shift: shortest paths are no longer chained to global ordering.

Taken together, these reflections show that while the breakthrough’s practical impact is yet to unfold, its **theoretical significance is undeniable**. It not only surpasses a long-standing runtime barrier but also reshapes our understanding of what is possible in deterministic graph algorithms.

**7. Future Directions and Speculation**

The immediate open question is whether the exponent in can be reduced further, potentially reaching Another direction concerns extending the approach to graphs with negative weights, a capability that would unify efficiency with generality. Dynamic graphs, which better model real-world networks, present another frontier.

Beyond shortest paths, this work underscores a broader lesson: algorithmic barriers are not always fundamental. If sorting is not intrinsic to shortest path computation, what other problems, that are long believed and settled, might be waiting for re-examination?

**8. Conclusion**

The 2025 breakthrough by Duan et al. is more than an asymptotic improvement. It dismantles a conceptual barrier that shaped our understanding of shortest path algorithms for over sixty years. By achieving a deterministic runtime of , the result reaffirms that even the most classical algorithmic problems can yield new surprises. For the theory of algorithms, it is both a milestone and an invitation: to question long-held assumptions and to seek further advances where none were thought possible.

**9. References**

* Duan, R., Mao, J., Mao, X., Shu, X., & Yin, L. (2025). *Breaking the Sorting Barrier for Directed Single-Source Shortest Paths.* arXiv:2504.17033.
* Bellman, R. (1958). *On a Routing Problem.* Quarterly of Applied Mathematics.
* Dijkstra, E. W. (1959). *A Note on Two Problems in Connexion with Graphs.* Numerische Mathematik.
* Henzinger, M., Bernstein, A., Nanongkai, D., & Saranurak, T. (2022). *Randomized Improvements for Shortest Paths.* Proceedings of FOCS.