Science | DOI:10.1145/3607866 Chris Edwards

Historic Algorithms Help Unlock Shortest-Path Problem Breakthrough

By revisiting key algorithms from computing, a team unlocked hidden efficiency in a long-standing computer science problem.

COMPUTER SCIENCE PIONEER Edsger Dijkstra's algorithms form the backbone of many computer subroutines, thanks to their elegant efficiency. However, seemingly subtle changes in requirements can lead to these often conceptually simple formulations failing to provide an accurate answer. The replacement algorithms provide the correct answers but are frequently orders of magnitude slower.

A recent breakthrough in combinatorial techniques has shown how these early algorithms can be revived.

Shortest-path problems provide good examples of the sensitivity of an algorithm to the specifics of their requirements. The underpinning for many applications from navigation to chip layout, these problems present a simple, basic proposition: Find the path through a network of directed paths that offers the lowest cost based on the weights applied to each hop. That weighting may reflect distance or delay, properties that are always positive in the physical world.

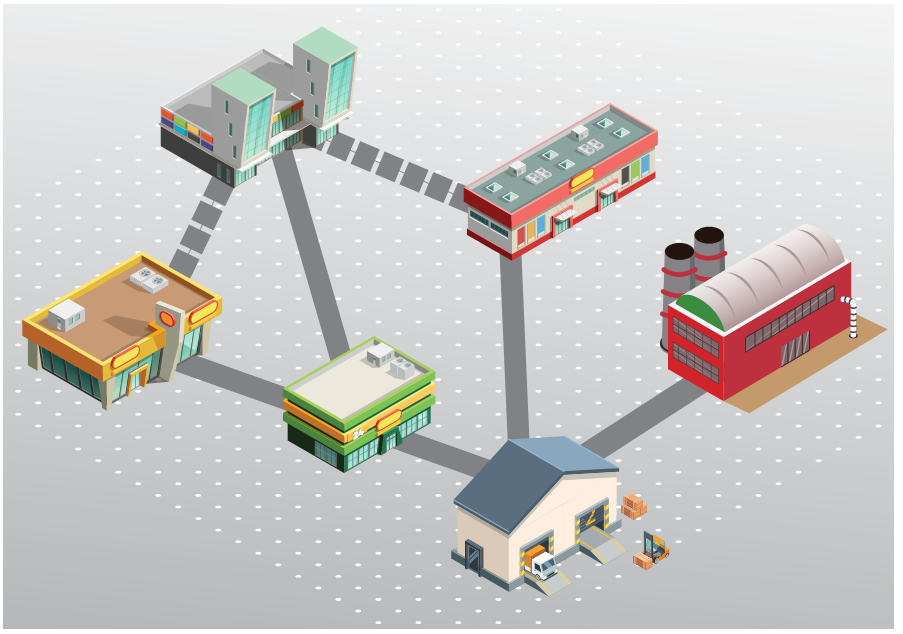

IMAGE BY ANDRIJ BORYS ASSOCIATES, USING SHUTTERSTOCK

Problems start when you encounter a similar network where paths can have negative weights. "When you're dealing with the situation when edge weights correspond to cost, negative weights are natural," says Aaron Bernstein, assistant professor of computer science at Rutgers University, New Brunswick, NJ.

In finance, for example, there may be situations in currency or options trading where buying and selling in one sequence is more profitable than taking a different path, and these can be modeled using negative weights as long as the search algorithm is fast enough. But those negative weights

can throw Dijkstra's shortest-path algorithm for a loop.

The issue lies in the efficient greediness of the algorithm developed in the mid-1950s: It will often remove negative-weight paths from consideration. It processes each vertex in turn and does not return, so the algorithm may not consider whether a high weight combined with one that carries a negative weight might result in a lower cost than a single hop with a low weight.

A revised approach, often known as the Bellman-Ford algorithm, handles the negative-weight connections correctly, but because it relies on the ability to process nodes, it lacks the raw efficiency of Dijkstra's technique, which completes in a time based on the sum of the number of nodes and connections. Bellman-Ford instead needs many more steps: The total is based on the product of the number of nodes and vertices.

Though computer scientists have attempted to find more efficient solutions to the problem using similar combinatorial techniques to those used in these longstanding algorithms, they failed to narrow the computational gap between them by a significant amount. Over the past few decades, greater advances were made by representing the graph as coefficients in a matrix and using the tools of linear algebra. Such techniques have proven successful for a wide range of path-related problems, not just for identifying the shortest route but for applications such as maximizing flow through a network of pipes or transportation routes, as demonstrated in a paper presented at last year's Symposium on Foundations of Computer Science (FOCS).

Georgia Institute of Technology Ph.D. student Li Chen, working with mathematicians at Switzerland's ETH Zurich, Stanford University, and the University of Toronto, developed a mechanism based on gradient descent, the same core approach as the one used to train neural networks, to progressively move to the best answer for maximizing flow through a network. This algorithm managed to bring the computation time down to almost linear performance.

The downside to recently developed techniques based on optimization is that they are more difficult to implement and understand than traditional combinatorial algorithms. "This type of approach often uses a lot of dynamic algorithms, which tend to be complicated. They involve many moving parts that you have to put together," says Danupon Nanongkai, director and scientific member of the Max Planck Institute for Informatics in Saarbrücken, Germany.

The Bellman-Ford algorithm handles the negative-weight connections correctly but lacks the raw efficiency of Dijkstra's technique.

Bernstein, Nanongkai, and Christian Wulff-Nilsen, associate professor of computer science at the University of Copenhagen, wanted to see if they could find an efficient combinatorial solution to the single-source, shortest-path problem with negative weights.

The team first turned to an early-1990s paper by Andrew Goldberg and colleagues that had reduced the complexity to the square root of the number of vertices multiplied by the number of nodes by trying to scale the weight values to become positive and so allow the shortest path to be found using Dijkstra's approach. "The original goal was just to improve the Goldberg result by a little bit but, in the end, it turned out that once everything was in place we could go all the way," says Nanongkai.

A long-standing technique for scaling is the use of price functions, though it can be difficult to assign prices efficiently and in a way that does not distort the relationship between weights. The team found an efficient algorithm which worked on a specific type of graph: those with low diameter. That proved to be the key to a bigger breakthrough and one that may help uncover more efficient combinatorial solutions to other problems.

ACM Member News

SYSTEM SOFTWARE FOR HIGH-PERFORMANCE COMPUTING

Robert Ross is a senior computer scientist at Argonne National Laboratory in Lemont, IL, where he is also the deputy division director of the Mathematics and Computer Science Division.

He also serves as director of the U.S. Department of Energy's SciDAC (Scientific Discovery through Advanced Computing) RAPIDS Institute for Computer Science, Data and Artificial Intelligence, a virtual organization of 13 national laboratories and higher education institutions.

As a child, Ross recalls, his family had an Apple II computer, and he started programming in high school. "That early interest led me down this career path."

Ross received both his undergraduate and Ph.D. degrees in Computer Engineering from Clemson University in Clemson, SC. After earning his doctorate in 2000, Ross joined Argonne National Laboratory where he has remained since, serving in different capacities over the years.

His research focuses on communication system software for high-performance computing (HPC) systems; in particular, the design, implementation, and deployment of complex distributed systems, and libraries for I/O and message passing.

Currently, Ross is working to foster interdisciplinary teams using HPC and the sciences. On the research side, he is exploring how to advance various services running on HPC platforms and connecting those HPC capabilities to other environments, such as the cloud and edge computing.

"In high-performance computing, we don't do a good job of adapting to different workloads," Ross explains. "This notion of elasticity that we see in the cloud is valuable and needs to be adopted in HPC, and we're helping make that happen."

—John Delaney

Low-diameter graphs are structures where the paths are short and can be seen as strongly connected to each other. The low-diameter property has been a key component of finding ways to cut graphs into sections so processing can be distributed across multiple computers. Ensuring that strongly connected components are kept on the same machine helps minimize network traffic.

The team found it was possible to divide the input graph into clusters of low-diameter subgraphs, and then to progressively rework the weights using a series of price functions to build a restructured graph that could be processed almost completely using Dijkstra's algorithm. This involved the random removal of paths with negative weights that would enable the source graph to be converted into a directed acyclic graph: one with only forward paths and no loops connecting the strongly connected clusters to each other. This form was selected because it opened the door to tools that would allow the use of the fastest algorithm. A later phase then reintroduces the missing negative-weight edges. The small number of these paths can be processed using the Bellman-Ford algorithm with comparatively little impact on runtime.

Though a form of low-diameter decomposition built for directed graphs proved fundamental to the solution, Bernstein says it was not immediately obvious as the traditional decomposition was developed for undirected graphs. "When we started to look at it, it was unclear what low-diameter decomposition in this context would mean and it didn't occur to us that it could be a relevant notion. It was only because we started working from a different direction and found we were dealing with graphs connected to the low-diameter property that we started to see how it could be used," he says.

The decomposition made it possible to break the price-function calculations into several steps, and do so in an efficient way without risking the distortion that a single-step approach would impose.

"In a sense, what our algorithm is doing is decomposing the graph, solving things with a few negative edge weights, fixing those, and then moving up. It's continually trying to make sure there are few negative edge weights, solving those and then figuring out the next, progressively trying to make things less and less negative," Bernstein adds.

The core algorithm has several elements with a logarithmic dependency on the number of vertices in the graph, each of which represents room for improvement.

Wulff-Nilsen notes, "This progressive improvement is done by changing the edges using price functions several times. So, it's not as if we just find one price point."

The result was an algorithm with near-linear performance, and with slightly better performance on its task compared to the optimization technique of maximum flow presented by Chen at the same FOCS event last year. Both shared the best-paper award at the symposium, and a number of teachers have since incorporated the combinatorial algorithm and colleagues into their classes because of its relative simplicity.

Though the core algorithm is near-linear, it has several elements that have a logarithmic dependency on the number of vertices in the graph, each of which presents room for improvement. Karl Bringmann and Alejandro Cassis of Saarland University teamed up with Nick Fischer of the Weizmann Institute of Science in Israel and managed to optimize away six of the eight log factors in a paper published in April. Some they regarded as simple, such as changing the order in which elements are presented to the underlying Dijkstra algorithm, but others were "more involved."

In their work, Bringmann and colleagues came up with what they describe as a more direct form of decomposition for their more efficient algorithm, together with a different form of low-diameter decomposition they did not use for this particular problem.

Such treatments could turn out to be as useful for directed graphs as low-diameter decomposition as they have been for work with undirected graphs. "I think people were excited to see that low-diameter decomposition could be used in this way. People have been talking to all of us about using this for other directed-graph problems," Bernstein claims.

Taking the low-diameter decomposition full circle, Bernstein, Nanongkai, and others published a paper in March demonstrating a distributed algorithm for shortest-path calculation. However, finding efficient combinatorial solutions to problems such as flow maximization remains an uphill struggle, and despite their greater complexity and reliance on techniques that call for the manipulation of dynamic data structures, the optimization-based approaches remain key tools for computer science.

Bernstein observes, "The optimization techniques really are the only way we know to solve some problems. Our algorithm showed a combinatorial solution for one problem, but it's still one particular problem. For minimum-cost flow, for example, we don't yet have an idea of how to do it combinatorially." C

Further Reading

Bernstein, A., Nanongkai, D., and Wulff-Nilsen, C.
Negative-Weight Single-Source Shortest Paths in Near-Linear Time
***Proceedings of the 63rd IEEE Annual Symp. on Foundations of Computer Science* (2022), 600–611**

Chen, L., Kyng, R., Liu, Y.P., Peng, R., Probst Gutenberg, M., and Sachdeva, S.
Maximum Flow and Minimum-Cost Flow in Almost-Linear Time
***Proceedings of the 63rd IEEE Annual Symp. on Foundations of Computer Science* (2022), 612–623**

Bringmann, K., Cassis, A., and Fischer, N.
Negative-Weight Single-Source Shortest Paths in Near-Linear Time: Now Faster!
ArXiv 2304.05279 (2023)

Linial, N. and Saks, M.
Low-Diameter Graph Decompositions
***Combinatorica 13* (4) (1993) 441–454**

Chris Edwards is a Surrey, U.K.-based writer who reports on electronics, IT, and synthetic biology.