Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages
Čeština
Deutsch
Español
فارسی
Français
Polski
Русский
Српски / srpski
ไทย
Tiếng Việt
Edit links

Dinic's algorithm

From Wikipedia, the free encyclopedia

Dinitz's algorithm is a strongly polynomial algorithm for computing the maximum flow in a flow network, conceived in 1970 by Israeli (formerly Soviet) computer scientist Yefim (Chaim) A. Dinitz.[1] The algorithm runs in $O(V^2E)$ time and is similar to the Edmonds–Karp algorithm, which runs in $O(VE^2)$ time, in that it uses shortest augmenting paths. The introduction of the concepts of the *level graph* and *blocking flow* enable Dinic's algorithm to achieve its performance.

History [edit]

Yefim Dinitz invented this algorithm in response to a pre-class exercise in Adel'son-Vel'sky's (co-inventor of AVL trees) Algorithm class. At the time he was not aware of the basic facts regarding Ford-Fulkerson algorithm.[2]

Dinitz mentions inventing his algorithm in January 1969, which was published in 1970 in journal *Doklady Akademii nauk SSSR*. In 1974, Shimon Even and (his then Ph.D. student) Alon Itai at the Technion in Haifa were very curious and intrigued by the Dinitz's algorithm as well as Alexander Karzanov's idea of blocking flow. However it was hard to decipher these two papers for them, each being limited to four pages to meet the restrictions of journal *Doklady Akademii nauk SSSR*. However Even did not give up and after three days of effort managed to understand both papers except for the layered network maintenance issue. Over the next couple of years, Even gave lectures on "Dinic's algorithm" mispronouncing the name of the author while popularizing it. Even and Itai also contributed to this algorithm by combining BFS and DFS, which is the current version of algorithm.[3]

For about 10 years of time after Ford–Fulkerson algorithm was invented, it was unknown if it can be made to terminate in polynomial time in the generic case of irrational edge capacities. This caused lack of any known polynomial time algorithm that solved max flow problem in generic case. Dinitz algorithm and the Edmonds–Karp algorithm, which was published in 1972, independently showed that in the Ford–Fulkerson algorithm, if each augmenting path is the shortest one, the length of the augmenting paths is non-decreasing and it always terminated.

Definition [edit]

Let $G = ((V, E), c, s, t)$ be a network with $c(u, v)$ and $f(u, v)$ the capacity and the flow of the edge (u, v) respectively.

The **residual capacity** is a mapping $c_f \colon V \times V \to R^+$ defined as,

1. if $(u, v) \in E$,
   $c_f(u, v) = c(u, v) - f(u, v)$
   $c_f(v, u) = f(u, v)$
2. $c_f(u, v) = 0$ otherwise.

The **residual graph** is the graph $G_f = ((V, E_f), c_f|_{E_f}, s, t)$, where

$$E_f = \{(u, v) \in V \times V : c_f(u, v) > 0\}.$$

An **augmenting path** is an $s - t$ path in the residual graph G_f.

Define $\operatorname{dist}(v)$ to be the length of the shortest path from s to v in G_f. Then the **level graph** of G_f is the graph $G_L = (V, E_L, c_f|_{E_L}, s, t)$, where

$$E_L = \{(u, v) \in E_f : \operatorname{dist}(v) = \operatorname{dist}(u) + 1\}.$$

A **blocking flow** is an $s - t$ flow f such that the graph $G' = (V, E'_L, s, t)$ with $E'_L = \{(u, v) : f(u, v) < c_f|_{E_L}(u, v)\}$ contains no $s - t$ path.

Algorithm [edit]

Dinic's Algorithm

Input: A network $G = ((V, E), c, s, t)$.
Output: An $s - t$ flow f of maximum value.

1. Set $f(e) = 0$ for each $e \in E$.
2. Construct G_L from G_f of G. If $\operatorname{dist}(t) = \infty$, stop and output f.
3. Find a blocking flow f' in G_L.
4. Augment flow f by f' and go back to step 2.

Analysis [edit]

It can be shown that the number of edges in each blocking flow increases by at least 1 each time and thus there are at most $n - 1$ blocking flows in the algorithm, where n is the number of vertices in the network. The level graph G_L can be constructed by Breadth-first search in $O(E)$ time and a

blocking flow in each level graph can be found in $O(VE)$ time. Hence, the running time of Dinic's algorithm is $O(V^2E)$.

Using a data structure called dynamic trees, the running time of finding a blocking flow in each phase can be reduced to $O(E \log V)$ and therefore the running time of Dinic's algorithm can be improved to $O(VE \log V)$.

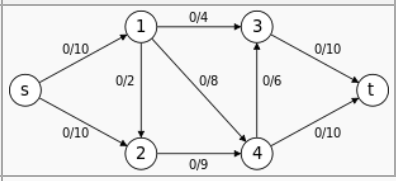
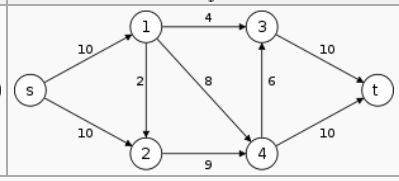
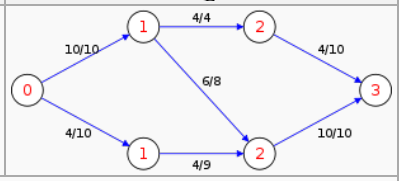
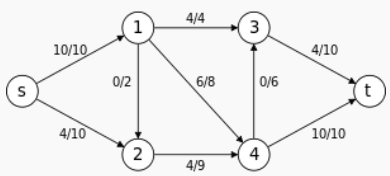
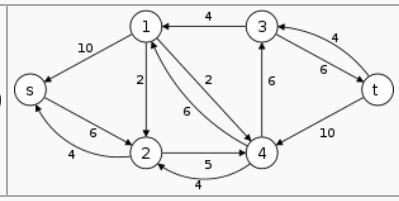
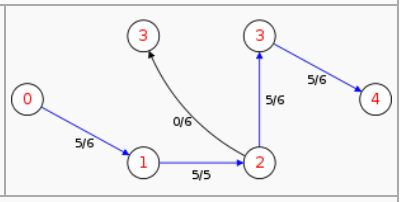
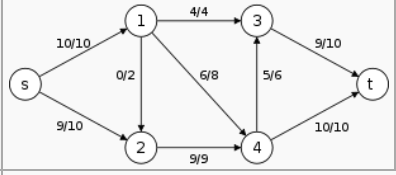
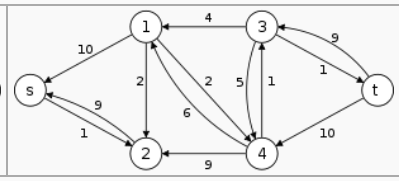
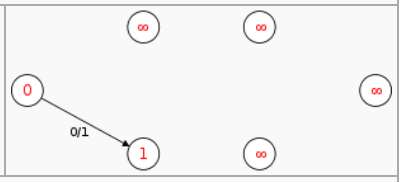
Special cases [edit]

In networks with unit capacities, a much stronger time bound holds. Each blocking flow can be found in $O(E)$ time, and it can be shown that the number of phases does not exceed $O(\sqrt{E})$ and $O(V^{2/3})$. Thus the algorithm runs in $O(\min\{V^{2/3}, E^{1/2}\}E)$ time.

In networks arising during the solution of bipartite matching problem, the number of phases is bounded by $O(\sqrt{V})$, therefore leading to the $O(\sqrt{V}E)$ time bound. The resulting algorithm is also known as Hopcroft–Karp algorithm. More generally, this bound holds for any *unit network* — a network in which each vertex, except for source and sink, either has a single entering edge of capacity one, or a single outgoing edge of capacity one, and all other capacities are arbitrary integers.[4]

Example [edit]

The following is a simulation of Dinic's algorithm. In the level graph G_L, the vertices with labels in red are the values $\text{dist}(v)$. The paths in blue form a blocking flow.

	G	G_f	G_L
1.			
	The blocking flow consists of 1. $\{s, 1, 3, t\}$ with 4 units of flow, 2. $\{s, 1, 4, t\}$ with 6 units of flow, and 3. $\{s, 2, 4, t\}$ with 4 units of flow. Therefore the blocking flow is of 14 units and the value of flow $\lvert f \rvert$ is 14. Note that each augmenting path in the blocking flow has *3* edges.		
2.			
	The blocking flow consists of 1. $\{s, 2, 4, 3, t\}$ with 5 units of flow. Therefore the blocking flow is of 5 units and the value of flow $\lvert f \rvert$ is 14 + 5 = 19. Note that each augmenting path has 4 edges.		
3.			
	Since t cannot be reached in G_f. The algorithm terminates and returns a flow with maximum value of 19. Note that in each blocking flow, the number of edges in the augmenting path increases by at least 1.		

See also [edit]

- Ford–Fulkerson algorithm
- Maximum flow problem

Notes [edit]

1. ^ Yefim Dinitz (1970). "Algorithm for solution of a problem of maximum flow in a network with power estimation" (PDF). *Doklady Akademii nauk SSSR* **11**: 1277–1280.
2. ^ Ilan Kadar, Sivan Albagli (2009-11-27). "Dinitz's algorithm for finding a maximum flow in a network".
3. ^ Yefim Dinitz. "Dinitz's Algorithm: The Original Version and Even's Version" (PDF).
4. ^ Tarjan 1983, p. 102.

References [edit]

- Yefim Dinitz (2006). "Dinitz' Algorithm: The Original Version and Even's Version" (PDF). In Oded Goldreich, Arnold L. Rosenberg, and Alan L. Selman. *Theoretical Computer Science: Essays in Memory of Shimon Even*. Springer. pp. 218–240. ISBN 978-3-540-32880-3.
- Tarjan, R. E. (1983). *Data structures and network algorithms*.
- B. H. Korte, Jens Vygen (2008). "8.4 Blocking Flows and Fujishige's Algorithm". *Combinatorial Optimization: Theory and Algorithms (Algorithms and Combinatorics, 21)*. Springer Berlin Heidelberg. pp. 174–176. ISBN 978-3-540-71844-4.