Flows in Networks: The Ford-Fulkerson Algorithm

Daniel Kane

Department of Computer Science and Engineering
University of California, San Diego

Advanced Algorithms and Complexity
Data Structures and Algorithms

Learning Objectives

- Compute maximum flows in networks.

Algorithm

Idea:

- Start with zero flow.
- Repeatedly add flow.
- Stop when you cannot add more.

Adding Flow

- Have flow f.
- Compute residual G_f. Represents ways in which flow can be added
- Any new flow $f + g$, where g a flow for G_f.
- Need to find flow for G_f.
- See if there's a source-sink path. because we want to add slightly larger flow f+g to our flow f

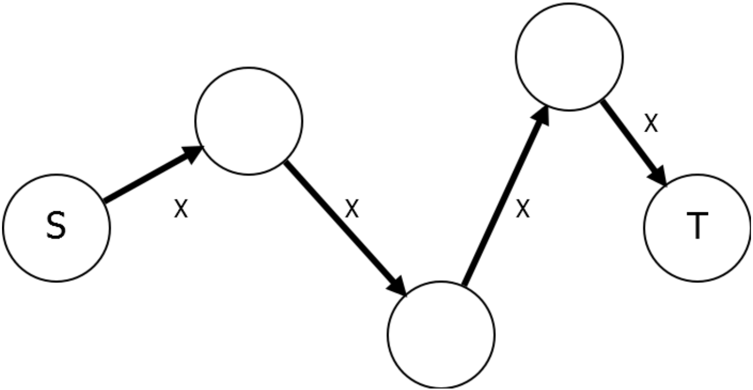
No Path

If there's no source-sink path in G_f:

- Reachable vertices define cut of size 0.
- No g of positive size.
- $|f + g| \leq |f|$.
- f is a maxflow. Flow is already max flow hence we are done

Path

If there is a path, add flow along path.



Add x units of flow along each edge if there is a path from S to T (This also maintains law of conservation of flow). Here x is min flow along the path from S to T.

Need $X \leq \min_{e \in \mathrm{path}} C_e$.

Adding Flow

- Find flow g for G_f with $|g| > 0$.
- Replace f by $f + g$.
- $|f + g| > |f|$.

by adding g we are left with f+g which is bigger then f and hence we keep on doing this untill we cannot add any more flow thus achieving max flow.

Pseudocode

Ford-Fulkerson(G)

$f \leftarrow 0$

repeat:

  Compute G_f

  Find $s - t$ path P in G_f To do this we have to go through all the paths hence time is O(|E|)

  if no path: return f

  $X \leftarrow \min_{e \in P} C_e$ X = Minimum flow amongst all edges from S to T

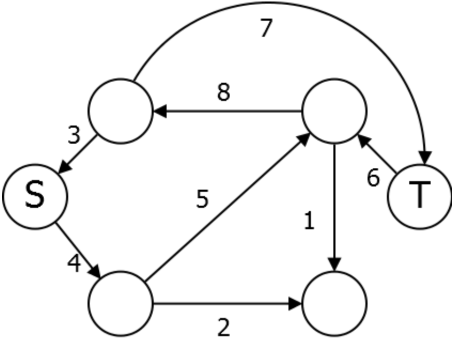
  g flow with $g_e = X$ for $e \in P$

Thus the total runtime is O(|E||f|)

  $f \leftarrow f + g$ Hence the max time we can do this is the max flow |f| times

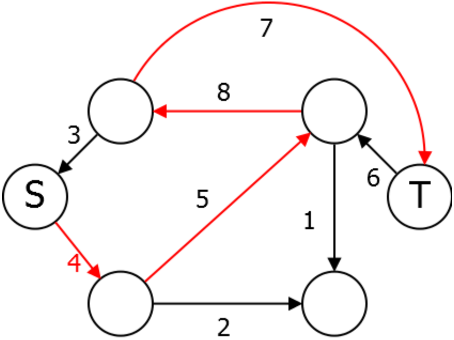
Problem

How much flow is added in one step?

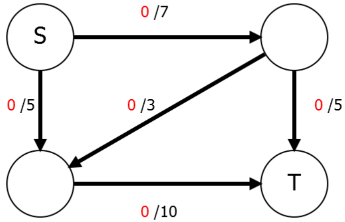


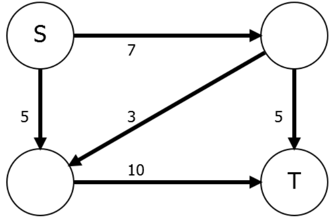
Solution

Bounded by 4.

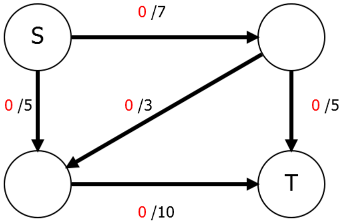


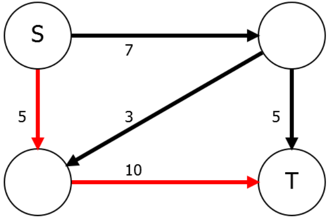
Example



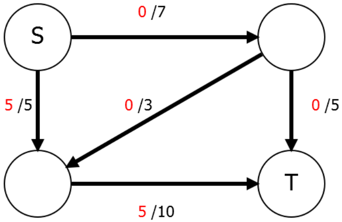


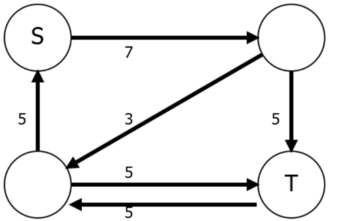
Example



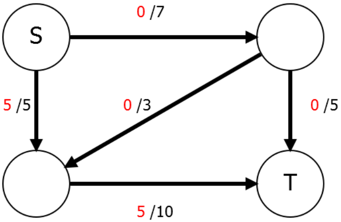


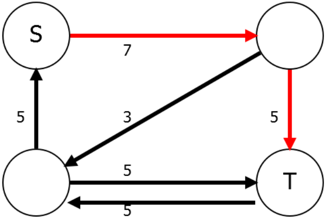
Example



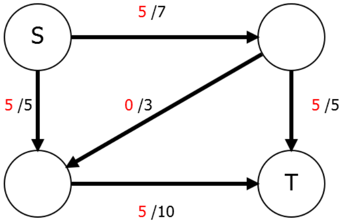


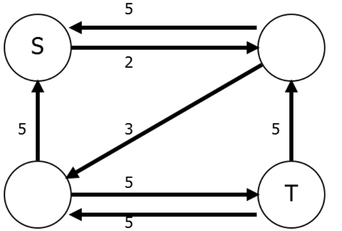
Example



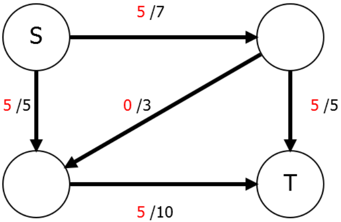


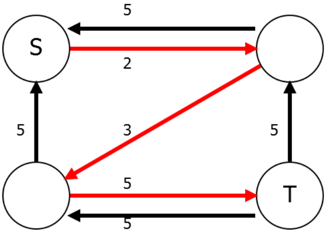
Example



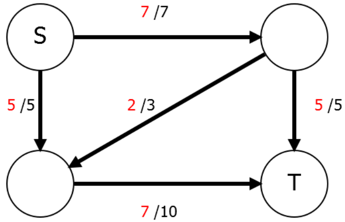


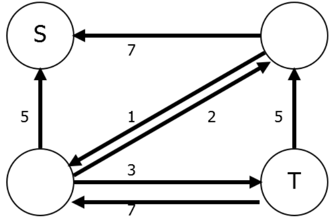
Example



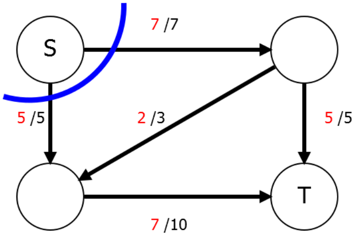


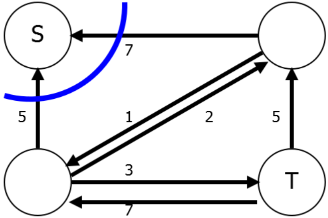
Example





Example





Observing the Cut (as no more flows can be added) we can notice that max flow is 12

Integrality

Note that if all capacities are integers, all flows we produce are integral.

Lemma

If the network G has integral capacities, there is a maximum flow with integral flow rates.

Analysis

[Assume integral capacities]

- Can compute G_f and find P in $O(|E|)$ time.
- Each time, increase total flow by at least 1.
- Total runtime: $O(|E||f|)$.

Analysis

[Assume integral capacities]

- Can compute G_f and find P in $O(|E|)$ time.
- Each time, increase total flow by at least 1.
- Total runtime: $O(|E||f|)$.

Note: Potentially quite large if flow is numerically large. Hence not just dependent on the structure of the network but also large capacities

Non-Determinacy

Note that the algorithm says to find an $s - t$ path in G_f. There might be many valid paths to choose from. Using DFS is fast, but perhaps not the best. As we will see the way we pick our path will affect the runtime of the algorithm.