CS 512 FINAL PROJECT

Project_group 1

Archit Bansal	Shreesh Keskar	Devarata Oza	Saransh Sharma
ab2465	skk139	do309	ss4368

Project area: Deterministic Algorithm Animation and Algorithm Snippets

Topic: The Ford-Fulkerson Augmenting Path Algorithm for the Maximum Flow Problem

Theory:

Flows in graphs

$G = (V, E)$ is a directed graph.

$c: e \rightarrow R^+$ is a capacity function on edges.
$c(e)$ is capacity of edge δ_{in}

Capacity constraint: flow is a function $f: E \rightarrow R^+$ such that $f(e) \leq c(e) \; \forall \; e \in E$.

Conservation constraint: $\forall \; v \in V \setminus \{s, t\} \quad \sum_{e \in \delta_{in}(v)} f(e) = \sum_{e \in \delta_{out}(v)} f(e)$

Value of flow = flow that starts from source
= flow that reaches sink

Examples of flow network models:

- fluid in a network of pipes
- traffic on a network of roads
- electricity in network of circuits

Our objective is to find a flow whose value is maximum.

What is a Cut?

A cut is a partition of the vertex set V into two parts $(S, \overline{S})$.

Edges of the cut are the edges which have one endpoint in S and the other in $\overline{S}$.

$s - t$ **cut:** A cut in which vertices s, t are in different parts. For an $s - t$ cut, the s-side is the set that contains vertex s and t-side is the set containing vertex t.

Capacity of an $s - t$ **cut:** The total capacity of edges going from s-side to t-side.

max-flow min-cut theorem: max $s - t$ flow = min capacity $s - t$ cut.

Ford–Fulkerson Algorithm:

residual capacity of an edge $e = c(e) - f(e)$

repeat

find a path from s to t (of positive residual capacity), say P

let δ be the minimum residual capacity of any edge in P

send δ units along P [increase flow of every edge on P by δ]

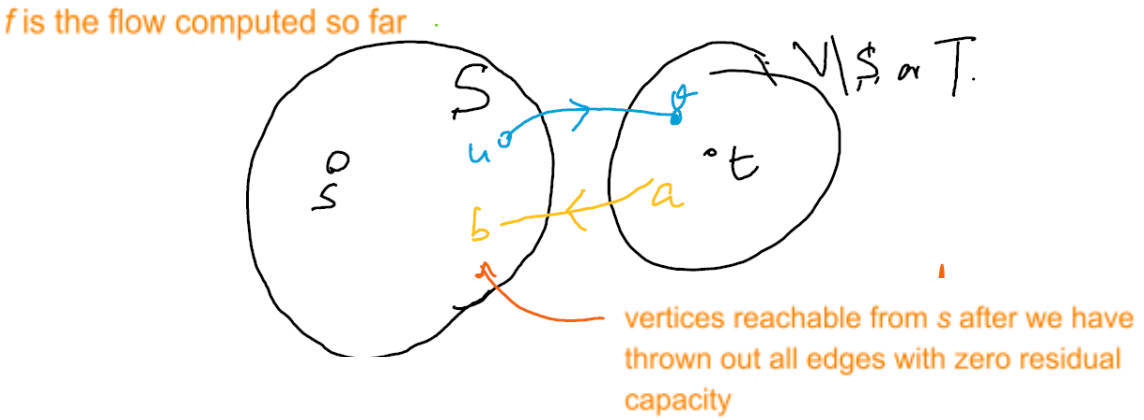
update residual capacities of edges along P [reduce residual capacity of edge along P by δ and increase residual capacity of each reverse edge by δ]

until we cannot find a path of positive residual capacity

Correctness of Ford-Fulkerson algorithm:

We will show that the flow computed equals the capacity of a certain s-t cut, i.e. the flow computed is maximum and the s-t cut found is minimum.

s-t flow $\leq$ *capacity of any s-t cut*



Blue edges have zero residual capacity $f(e) = c(e) \Rightarrow e$ is saturated

Yellow edges carry zero flow. If (a, b) has non-zero flow then we would have an edge (b, a) of non-zero residual capacity. Then a would have been reachable from s.

The net flow going out of S equals total capacity of blue edges
= capacity of cut (S, T)
= flow from source s
= flow from s to t

$\Rightarrow (S, T)$ cut is the minimum capacity s-t cut and flow f is the maximum flow.

max-flow min-cut theorem: maximum flow from s to t equals the min capacity cut separating s, t.

Running time: When edge capacities are integral, if F is the maximum flow from s to t then the Ford-Fulkerson algorithm takes at most F steps. In each step we find a path from s to t and this takes $O(m)$ time, where m is the number of edges.
So total time is $O(mF)$.

Space complexity: Total space consumed is $O(n)$, where n is the number of vertices.

***Note on the Python code:** A directed weighted sample graph is provided as input in the form of an adjacency matrix. However, any other directed graph with positive edge weights can be passed in the form of an adjacency matrix. The* s *and* t *vertices have been taken as user input. The function prints the maximum flow from* s *to* t.