Network Flow

- Maximum Flow: Problem Formulation
- Maximum Flow: Upper Bound
- Maximum Flow: Adding flow along paths
- Residual Network and Augmenting Path
- Ford-Fulkerson Algorithm – Max-Flow-Min-Cut Theorem
- Edmond-Karp Algorithm
- Maximum Flow: Variants and Applications

IMDAD ULLAH KHAN

Max Flow : Problem Formulation

Input: A flow network $G = (V, E, c)$, $c : E \to \mathbb{R}^+$

$s \in V$ a source and $t \in V$ a sink

$f : E \to \mathbb{R}^+$ ($f_e = f(e)$) is a flow if it satisfies

1. capacity constraints $\forall\, e \in E \;:\; 0 \leq f_e \leq c_e$
2. flow conservation constraints $\forall\, v \in V\,, v \neq s, t \quad f^{out}(v) = f^{in}(v)$

$$size(f) \;=\; f^{out}(s) \;=\; f^{in}(t)$$

Output: A flow f of maximum size

Let f be a flow in G and let $\left[A, \overline{A}\right]$ be any $s - t$ cut in G, then

$$size(f) \;\leq\; c\left(\left[A, \overline{A}\right]\right)$$

Max Flow : Algorithm

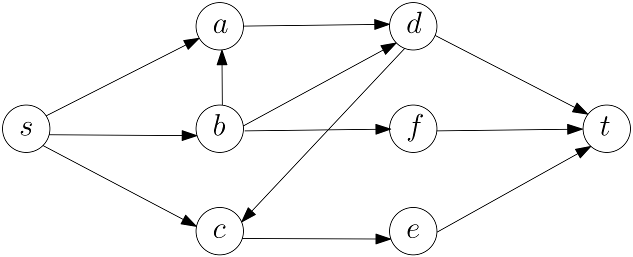
There is no known divide-and-conquer or dynamic programming algorithm for max-flow

We try a greedy strategy – build up flow little bit at a time

1. Start with a 0 flow
   - ▷ Note: the flow f with $f_e = 0$ for all $e \in E$ satisfies both constraints
2. Add more flow to f via a $s - t$ path

Max Flow : Algorithm

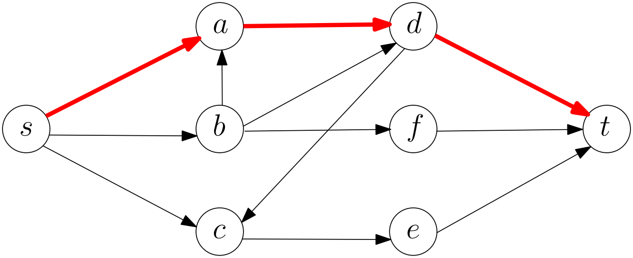
Consider the flow network – all edge capacity are 1



Max Flow : Algorithm

Consider the flow network – all edge capacity are 1

Add flow of size 1 via the $s - t$ path s, a, d, t

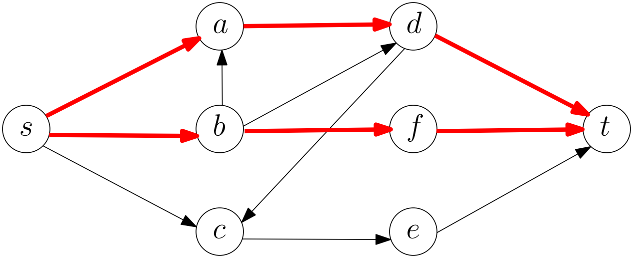


Max Flow : Algorithm

Consider the flow network – all edge capacity are 1

Add flow of size 1 via the $s - t$ path s, a, d, t

Add flow of size 1 via the $s - t$ path s, b, f, t



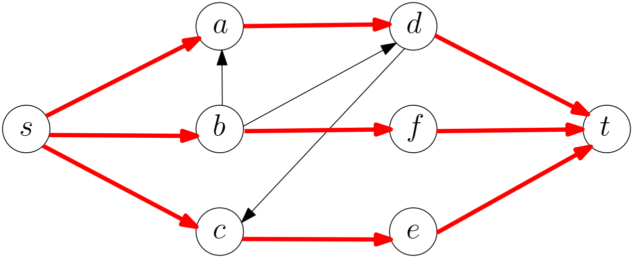
Max Flow : Algorithm

Consider the flow network – all edge capacity are 1

Add flow of size 1 via the $s - t$ path s, a, d, t

Add flow of size 1 via the $s - t$ path s, b, f, t

Add flow of size 1 via the $s - t$ path s, c, e, t



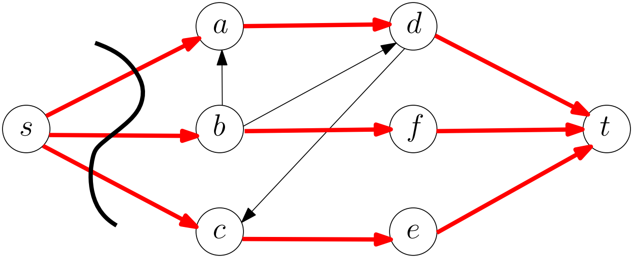
Max Flow : Algorithm

Consider the flow network – all edge capacity are 1

Add flow of size 1 via the $s - t$ path s, a, d, t

Add flow of size 1 via the $s - t$ path s, b, f, t

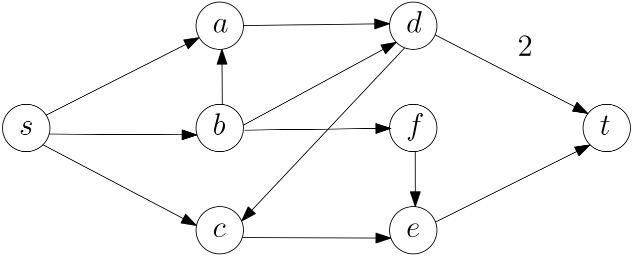
Add flow of size 1 via the $s - t$ path s, c, e, t



There is an $s - t$ cut of capacity 3, hence this flow is maximum possible

Max Flow : Algorithm

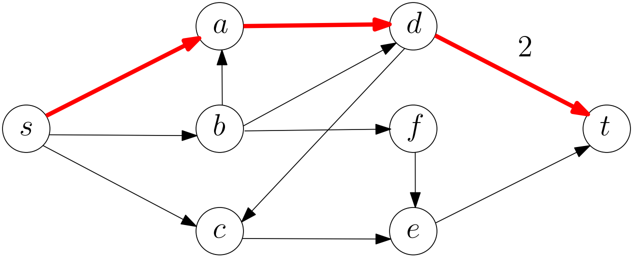
Consider the flow network – with not all capacities $= 1$



Max Flow : Algorithm

Consider the flow network – with not all capacities $= 1$

Add flow of size $1 =$ path-bottleneck via the $s - t$ path s, a, d, t

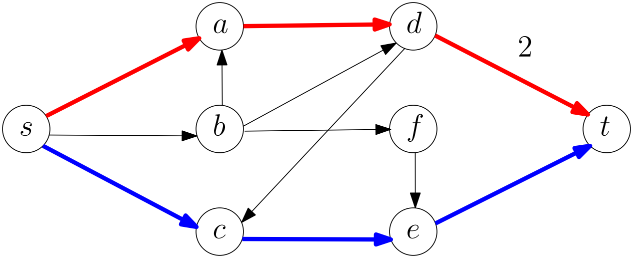


Max Flow : Algorithm

Consider the flow network – with not all capacities $= 1$

Add flow of size $1 =$ path-bottleneck via the $s - t$ path s, a, d, t

Add flow of size 1 via the $s - t$ path s, b, f, t



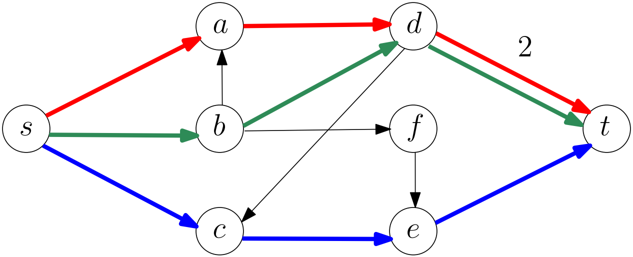
Max Flow : Algorithm

Consider the flow network – with not all capacities = 1

Add flow of size 1 = path-bottleneck via the $s - t$ path s, a, d, t

Add flow of size 1 via the $s - t$ path s, b, f, t

Add flow of size 1 via the $s - t$ path s, c, e, t



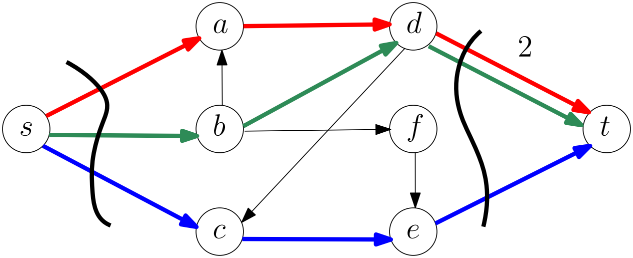
Max Flow : Algorithm

Consider the flow network – with not all capacities = 1

Add flow of size 1 = path-bottleneck via the $s-t$ path s, a, d, t

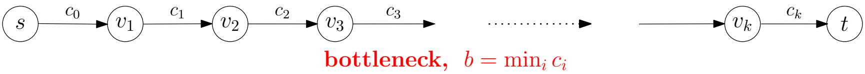
Add flow of size 1 via the $s-t$ path s, b, f, t

Add flow of size 1 via the $s-t$ path s, c, e, t

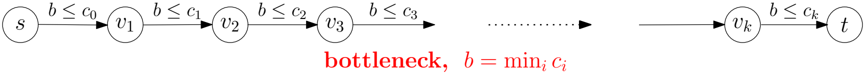


There is an $s-t$ cut of capacity 3, hence this flow is maximum possible

Max Flow : Adding Flow along a path



Adding flow along a path ensures that flow conservation constraints remain satisfied



Adding a flow along a path equal to the bottleneck of the path doesn't violate the capacity constraints

These two facts ensure that in any intermediate step of such a greedy algorithm the flow indeed is a valid flow