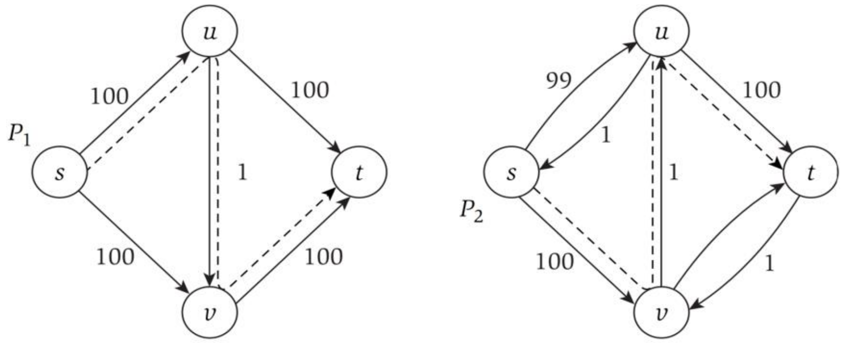
Algorithm Design & Analysis (CSE222)

Lecture-22

Recap

- Network Flow
  - Min Cut
  - Max-Flow Min-Cut Theorem



- Scaling Max-Flow

CAPACITY-SCALING(G)

FOREACH edge $e \in E$: $f(e) \leftarrow 0$.

$\Delta \leftarrow$ largest power of 2 $\leq C$.

WHILE ($\Delta \geq 1$)

  $G_f(\Delta) \leftarrow \Delta$-residual network of G with respect to flow f.

  WHILE (there exists an $s \rightsquigarrow t$ path P in $G_f(\Delta)$)

    $f \leftarrow$ AUGMENT(f, c, P).

    Update $G_f(\Delta)$.

  $\Delta \leftarrow \Delta / 2$.

Δ-scaling phase

RETURN f.

Outline

- Scaling Max-Flow
- Shortest Path

Scaling Max Flow

Observation

- If the capacities are integers then the flow and the residual capacities are also integers.
- At $\Delta = 1$ we get $G_f(\Delta) = G_f$.
- When it terminates, the flow f is the maximum flow.

It is important to analyze the number of augmentation done in each phase, fixed Δ

Claim: There are at most $1+\log C$ scaling phases in the algorithm.

Scaling Max Flow

Claim: There are at most 1+log C scaling phases in the algorithm.

Proof:

- The total capacity is C.
- Starting with Δ’ = C, in every subsequent phase we update Δ = Δ’/2.
- So at max the the algorithm goes over 1 + log C scaling phases.

Scaling Max Flow

Claim: Let f be a flow at the end of Δ-scaling phase. There is an s-t cut (A,B) such that $c(A,B) \leq |f| + m\Delta$. So possible max flow is $|f| + m\Delta$.

Proof: Let A be the set of all nodes such that there is path s-v in $G_f(\Delta)$ & $B = V \setminus A$.

(A,B) is a cut: $s \in A$ as there is always a path s-s.
As, no path s-t in $G_f(\Delta)$ so, $t \notin A$ hence, $t \in B$.

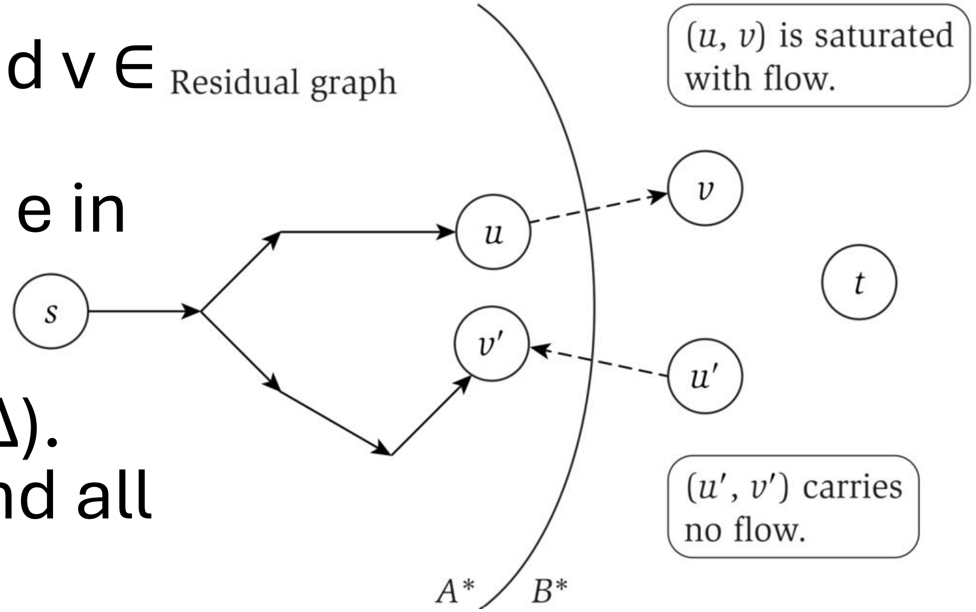
Scaling Max Flow

Claim: Let f be a flow at the end of Δ-scaling phase. There is an s-t cut (A,B) such that c(A,B) ≤ |f| + mΔ. So possible max flow is |f| + mΔ.

Proof: Let A be the set of all nodes such that there is path s-v in $G_f(\Delta)$ & B = V \ A.

(A,B) is a cut.

- Let, e = (u,v) be an edge such that u ∈ A and v ∈ B
- Then c(e) < f(e) + Δ. As no forward edge on e in $G_f(\Delta)$.
- Let e' = (u',v') edge in G, u'∈B & v'∈A.
- Then f(e') < Δ. As no backward edge in $G_f(\Delta)$.
- So all edge from A are almost saturated and all edge into A are almost unused.



So, $|f| = f^{out}(A) - f^{in}(A) > \sum_{e \text{ from } A^*}[c(e)-\Delta] - \sum_{e \text{ into } A^*} \Delta > \sum_{e \text{ from } A^*} c(e) - m\Delta = c(A,B) - m\Delta$.

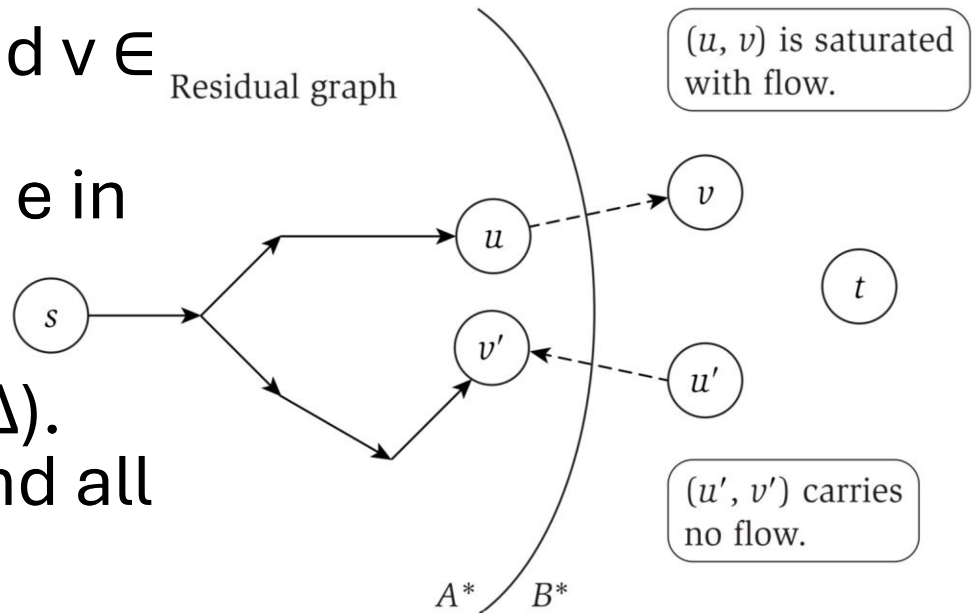
Scaling Max Flow

<u>Claim</u>: Let f be a flow at the end of Δ-scaling phase. There is an s-t cut (A,B) such that c(A,B) ≤ |f| + mΔ. So possible max flow is |f| + mΔ.

<u>Proof</u>: Let A be the set of all nodes such that there is path s-v in $G_f(\Delta)$ & B = V \ A.

(A,B) is a cut.

- Let, e = (u,v) be an edge such that u ∈ A and v ∈ B
- Then c(e) < f(e) + Δ. As no forward edge on e in $G_f(\Delta)$.
- Let e' = (u',v') edge in G, u'∈B & v'∈A.
- Then f(e') < Δ. As no backward edge in $G_f(\Delta)$.
- So all edge from A are almost saturated and all edge into A are almost unused.



So, c(A,B) < |f| + mΔ. Hence max flow is at most |f| + mΔ.

Scaling Max Flow

Claim: Number of augmentations in a scaling phase is at most 2m.

Proof:

- At first phase it is trivially true as every edge is used for only one path s-t.
- At any intermediate scaling phase Δ, let f_p be the flow by the end of the previous phase.
- In the previous phase we used Δ' = 2Δ.
- By previous claim $|f^*| \leq |f_p| + m\Delta' = |f_p| + 2m\Delta$
- In the Δ-scaling phase each augmentation increases the flow by at least Δ
- So, there can be at most 2m augmentations.

Scaling Max Flow

- There are $1 + \log_2 C$ scaling phases.
- In each phase there are at most $2m$ computations of augmenting paths.
- Every augmentation takes $O(m)$ time.

Claim: Total running time of Scaling Max Flow is $O(m^2 \cdot \log_2 C)$.

Is it better than Ford-Fulkerson algorithm?

Outline

- Scaling Max-Flow
- Shortest Path

Shortest Augmenting Path (Edmonds-Karp)

- In the residual graph, choose the shortest (#edges) path s-t.
- Let f_i be the flow after i^{th} augmentation step. So f_0: E → 0.
- Let G_i be the corresponding residual graph. So, G_0 = G.
- For each node v $level_i(v)$ be the simple shortest path s-v or level of v in BFS(s).
- If v is unreachable from s in G_i then $level_i(v) = \infty$.

Claim: The level of a vertex can only increase over time.

Shortest Augmenting Path (Edmonds-Karp)

<u>Claim</u>: For every node v and integer $i > 0$, $level_i(v) \geq level_{i-1}(v)$.

<u>Proof</u>: Fix a node v and prove by induction.
Inductive hypothesis: assume for every node u, $level_i(u) \geq level_{i-1}(u)$.

- If v=s then $level_i(v) = level_i(s) = 0$.
- If no path s-v then $level_i(v) = \infty \geq level_{i-1}(v)$.
- Let $s \longrightarrow \ldots \longrightarrow u \longrightarrow v$ be simple shortest path s-v in G_i. So, $level_i(v) = level_i(u) + 1 \geq level_{i-1}(u) + 1$.
    - If $u \rightarrow v$ is an edge in G_{i-1}, then $level_{i-1}(v) \leq level_{i-1}(u) + 1$. As levels are defined by BFS(s).
    - If $u \rightarrow v$ is not an edge in G_{i-1} then its reverse $v \rightarrow u$ must be an edge in i^{th} augmenting path.
    - This is simple shortest path s-t in G_{i-1}. So, $level_{i-1}(v) = level_{i-1}(u) - 1 \leq level_{i-1}(u) + 1$.
- So we conclude, $level_i(v) \geq level_{i-1}(u) + 1 \geq level_{i-1}(v) - 1 + 1 = level_{i-1}(v)$.

Shortest Augmenting Path (Edmonds-Karp)

Claim: Each edge u→v disappears from the residual graphs G_i at most |V|/2 times.

Proof:

- Let u→v edge is in two residual graph G_i and G_{j+1} but not in any intermediate G_l for $i < l < j$, for some $i < j+1$.
- So, u→v must be in the i^{th} augmenting path. So, $level_i(v) = level_i(u) + 1$.
- So, v→u must be in the j^{th} augmenting path. So, $level_j(u) = level_j(v) + 1$ or $level_j(v) = level_j(u) - 1$.

We know that,
$level_j(u) = level_j(v) + 1 \geq level_i(v) + 1 = level_i(u) + 2$.

- The level difference between appearance and disappearance is 2.
- There are at most |V| possible labels, so #disappearance is at most |V|/2.

Shortest Augmenting Path (Edmonds-Karp)

Running time

- Each edge disappears at most V/2 times.
- There are EV/2 many number of disappearance.
- Each iteration takes O(E) time to compute the simple shortest s-t path.
- Finally the running time is $O(VE^2)$.

Reference

Slides

Jeff Erickson Chp-10

Algorithms Design by Kleinberg & Tardos - Chp 7.3