Augmenting path theorem. Flow $f$ is a max flow iff there are no augmenting paths.

Max-flow min-cut theorem. [Ford-Fulkerson 1956] The value of the max flow is equal to the value of the min cut.

Proof strategy. We prove both simultaneously by showing the TFAE:

(i) There exists a cut $(A, B)$ such that $v(f) = \text{cap}(A, B)$.
(ii) Flow $f$ is a max flow.
(iii) There is no augmenting path relative to $f$.

(i) $\Rightarrow$ (ii) This was the corollary to weak duality lemma.

(ii) $\Rightarrow$ (iii) We show contrapositive.

- Let $f$ be a flow. If there exists an augmenting path, then we can improve $f$ by sending flow along path.
(iii) \(\Rightarrow\) (i)

- Let \(f\) be a flow with no augmenting paths.
- Let \(A\) be set of vertices reachable from \(s\) in residual graph.
- By definition of \(A\), \(s \in A\).
- By definition of \(f\), \(t \notin A\).

\[
\nu(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\
= \sum_{e \text{ out of } A} c(e) \\
= \text{cap}(A, B)
\]

original network
Running Time

Assumption. All capacities are integers between 1 and C.

Invariant. Every flow value \( f(e) \) and every residual capacities \( c_f(e) \) remains an integer throughout the algorithm.

Theorem. The algorithm terminates in at most \( v(f^*) \leq nC \) iterations.

Pf. Each augmentation increase value by at least 1. \( \Box \)

Corollary. If \( C = 1 \), Ford-Fulkerson runs in \( O(mn) \) time.

Integrality theorem. If all capacities are integers, then there exists a max flow \( f \) for which every flow value \( f(e) \) is an integer.

Pf. Since algorithm terminates, theorem follows from invariant. \( \Box \)
7.3 Choosing Good Augmenting Paths
Q. Is generic Ford-Fulkerson algorithm polynomial in input size?

A. No. If max capacity is $C$, then algorithm can take $C$ iterations.

$m, n, \text{ and } \log C$
Choosing Good Augmenting Paths

Use care when selecting augmenting paths.
- Some choices lead to exponential algorithms.
- Clever choices lead to polynomial algorithms.
- If capacities are irrational, algorithm not guaranteed to terminate!

Goal: choose augmenting paths so that:
- Can find augmenting paths efficiently.
- Few iterations.

Choose augmenting paths with: [Edmonds-Karp 1972, Dinitz 1970]
- Max bottleneck capacity.
- Sufficiently large bottleneck capacity.
- Fewest number of edges.
**Capacity Scaling**

**Intuition.** Choosing path with highest bottleneck capacity increases flow by max possible amount.
- Don't worry about finding exact highest bottleneck path.
- Maintain scaling parameter $\Delta$.
- Let $G_f(\Delta)$ be the subgraph of the residual graph consisting of only arcs with capacity at least $\Delta$.  

![Graph](image_url)
Capacity Scaling

Scaling-Max-Flow(G, s, t, c) {
    foreach e ∈ E  f(e) ← 0
    Δ ← smallest power of 2 greater than or equal to C
    G_f ← residual graph

    while (Δ ≥ 1) {
        G_f(Δ) ← Δ-residual graph
        while (there exists augmenting path P in G_f(Δ)) {
            f ← augment(f, c, P)
            update G_f(Δ)
        }
        Δ ← Δ / 2
    }
    return f
}
Assumption. All edge capacities are integers between 1 and C.

Integrality invariant. All flow and residual capacity values are integral.

Correctness. If the algorithm terminates, then f is a max flow.

Pf.
- By integrality invariant, when $\Delta = 1 \Rightarrow G_f(\Delta) = G_f$.
- Upon termination of $\Delta = 1$ phase, there are no augmenting paths. □
**Capacity Scaling: Running Time**

Lemma 1. The outer while loop repeats \(1 + \lceil \log_2 C \rceil\) times.

Proof. Initially \(C \leq \Delta < 2C\). \(\Delta\) decreases by a factor of 2 each iteration. □

Lemma 2. Let \(f\) be the flow at the end of a \(\Delta\)-scaling phase. Then the value of the maximum flow is at most \(v(f) + m \Delta\). ← proof on next slide

Lemma 3. There are at most \(2m\) augmentations per scaling phase.

- Let \(f\) be the flow at the end of the previous scaling phase.
- \(L2 \Rightarrow v(f^*) \leq v(f) + m (2\Delta)\).
- Each augmentation in a \(\Delta\)-phase increases \(v(f)\) by at least \(\Delta\). □

Theorem. The scaling max-flow algorithm finds a max flow in \(O(m \log C)\) augmentations. It can be implemented to run in \(O(m^2 \log C)\) time. □
Lemma 2. Let $f$ be the flow at the end of a $\Delta$-scaling phase. Then value of the maximum flow is at most $v(f) + m\Delta$.

Pf. (almost identical to proof of max-flow min-cut theorem)

- We show that at the end of a $\Delta$-phase, there exists a cut $(A, B)$ such that $\text{cap}(A, B) \leq v(f) + m\Delta$.
- Choose $A$ to be the set of nodes reachable from $s$ in $G_f(\Delta)$.
- By definition of $A$, $s \in A$.
- By definition of $f$, $t \notin A$.

\[
v(f) = \sum_{e \text{ out of } A} f(e) - \sum_{e \text{ in to } A} f(e) \\
\geq \sum_{e \text{ out of } A} (c(e) - \Delta) - \sum_{e \text{ in to } A} \Delta \\
= \sum_{e \text{ out of } A} c(e) - \sum_{e \text{ out of } A} \Delta - \sum_{e \text{ in to } A} \Delta \\
\geq \text{cap}(A, B) - m\Delta
\]