7.3 Choosing Good Augmenting Paths

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, Dinic 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 $G_\Delta$ be the subgraph of the residual graph consisting of only arcs with capacity at least $\Delta$. 

Ford-Fulkerson: Exponential Number of Augmentations

Q. Is generic Ford-Fulkerson algorithm polynomial in input size?

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

Max Flow Recap

- Max-Flow Problem, Min Cut Problem
- Definition of a $s$-$t$ flow $f(x)$ and a $s$-$t$ cut $(A,B)$
- Value of a flow $f$
- Capacity of a $s$-$t$ cut $(A,B)$

Weak Duality Lemma: For any flow $f$ and $s$-$t$ cut $A,B$ we have $\text{val}(f) \leq \text{cap}(A,B)$ (i.e., capacity of minimum cut is upper bound on max flow)

Finding a Max-Flow:
- Greedy algorithm failed
- Residual Graph
- Ford-Fulkerson Algorithm
  - Iteratively find augmenting path in residual graph
  - Proof of Correctness
  - Max-Flow Min-Cut Equivalence
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
  Gₙ ← residual graph
  while (Δ ≥ 1) {
    Gₙ(Δ) ← Δ - residual graph
    while (there exists augmenting path P in Gₙ(Δ)) {
      f ← augment(f, c, P)
      update Gₙ(Δ)
    }
    Δ ← Δ / 2
  }
  return f
}

Capacity Scaling: Correctness

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 Δ = 1 ⇒ Gₙ(Δ) = Gₙ.
  ▪ Upon termination of Δ = 1 phase, there are no augmenting paths.

Capacity Scaling: Running Time

Lemma 1. The outer while loop repeats 1 + \lfloor \log₂ C \rfloor times.

Pf. Initially C ≤ Δ < 2C. Δ decreases by a factor of 2 each iteration.

Lemma 2. Let f be the flow at the end of a Δ-scaling phase. Then the value of the maximum flow is at most v(f) + m Δ.

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

Theorem. The scaling max-flow algorithm finds a max flow in O(m log C) augmentations. It can be implemented to run in O(m² log C) time.

Dinic's Max Flow Min-Cut Algorithm

Use Breadth First Search to Compute Level Graph

Use Breadth First Search to Compute Level Graph
Dinic's Max Flow Min-Cut Algorithm

Use Breadth First Search to Compute Level Graph

Create Residual Graph $G_f$

Remark: Number of levels increased. This is not a coincidence!
Dinic’s Max Flow Min-Cut Algorithm

New Residual Graph $G_f$

Finding a Blocking Flow in $G_{f_1}$

**Definition:** $c_{f_1}(e)$ denotes the capacity of the edge $e$ in $G_{f_1}$

**Definition:** Given an augmenting flow $f^*$ for the level graph $G_{f_1}$ and a path $P$ in $G_{f_1}$ we define $R(P, f^*) = \min\{c_{f_1}(e) - f^*(e) : e \in P\}$

**Finding Blocking Flow($G_{f_1}$)**

1. **Initialize**
   - $\text{InitCap}(e) = c_{f_1}(e)$ and $f^*(e) = 0$ for each edge $e$ in $G_{f_1}$
   - While there is a path $P$ with $R(P, f^*) > 0$
     - Update $f^*(e) = f^*(e) + R(P, f^*)$ for each edge $e \in P$
     - Update $\text{InitCap}(e) = \text{InitCap}(e) - R(P, f^*)$ for each edge $e \in P$

**Analysis:** Each iteration of the "while" loop eliminates an edge

**Implication:** Terminates after $O(n)$ iterations of while loop.

Finding a Blocking Flow in $G_{f_2}$

**Definition:** $c_{f_2}(e)$ denotes the capacity of the edge $e$ in $G_{f_2}$

**Definition:** Given an augmenting flow $f^*$ for the level graph $G_{f_2}$ and a path $P$ in $G_{f_2}$ we define $R(P, f^*) = \min\{c_{f_2}(e) - f^*(e) : e \in P\}$

**Finding Blocking Flow($G_{f_2}$)**

1. **Initialize**
   - $\text{InitCap}(e) = c_{f_2}(e)$ and $f^*(e) = 0$ for each edge $e$ in $G_{f_2}$
   - While there is a path $P$ with $R(P, f^*) > 0$
     - Update $f^*(e) = f^*(e) + R(P, f^*)$ for each edge $e \in P$
     - Update $\text{InitCap}(e) = \text{InitCap}(e) - R(P, f^*)$ for each edge $e \in P$

**Analysis:** Each iteration of the "while" loop eliminates an edge

**Implication:** Terminates after $O(n)$ iterations of while loop.

Reachable

Dinic’s Algorithm

1. Start with empty flow $f$
2. Construct $G_f$
3. Repeat until $s$ and $t$ are disconnected (no augmenting path)
   - (Level Graph) Run BFS on $G_f$ to build $G_{f_1}$
   - (Blocking Flow) Find blocking flow $f^*$ in $G_{f_1}$
4. (Augment) Let $f \leftarrow f + f^*$ and Construct $G_f$
5. Output $f$

**Analysis:**
- Global: Each time we iterate the loop we increase the depth of $G_f$
- Implication: Stop iteration in at most $n$ iterations.

**Time Per Iteration:** $O(n^2m)$ to find first blocking flow $f^*$

**Total Time:** $O(n^3)$
Circulation with Demands

Circulation with demands
- Directed graph $G = (V, E)$
- Edge capacities $c(e), e \in E$
- Node supply and demands $d(v), v \in V$

Demand if $d(v) > 0$ supply if $d(v) < 0$ transshipment if $d(v) = 0$

Def. A circulation is a function that satisfies:
- For each $e \in E$: $0 \leq f(e) \leq c(e)$ (capacity)
- For each $v \in V$: $\sum_{e\in\delta^{-}(v)} f(e) - \sum_{e\in\delta^{+}(v)} f(e) = d(v)$ (conservation)

Circulation problem: given $(V, E, c, d)$, does there exist a circulation?

Necessary condition: sum of supplies = sum of demands.

$\sum_{v\in V} d(v) = \sum_{v\in V} d(v) = D$

Pf. Sum conservation constraints for every demand node $v$.

Circulation with Demands

Circulation with Demands: Circulation problem: given $(V, E, c, d)$, does there exist a circulation?

Necessary condition: sum of supplies = sum of demands.

$\sum_{v\in V} d(v) = \sum_{v\in V} d(v) = D$

Pf. Sum conservation constraints for every demand node $v$. 

Key Claim: $\text{depth}(G_f) > \text{depth}(G)$ (depth always increases)

Proof: Suppose (for contradiction) $\text{depth}(G_f) \leq \text{depth}(G)$.
- Then $G_{f}$ contains an $s$-$t$ path of length $\leq \text{depth}(G_f)$.
- This path corresponds to an augmenting path for the flow $f = f_{\text{inf}} - f_{\text{from}}$.
- But since the augmenting path has length $\text{depth}(G_f)$ it is also an augmenting path in the level graph $G_{f_{\text{inf}}}$.
- This contradicts the claim that $f$ is a blocking flow in $G_{f_{\text{inf}}}$.
Circulation with Demands

Max flow formulation.

- Add new source $s$ and sink $t$.
- For each $v$ with $d(v) > 0$, add edge $(v, t)$ with capacity $d(v)$.
- Claim: $G$ has circulation iff $G'$ has max flow of value $D$.

G:

<table>
<thead>
<tr>
<th></th>
<th>3</th>
<th>10</th>
<th>6</th>
<th>-7</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

saturates all edges leaving $s$ and entering $t$

G':

<table>
<thead>
<tr>
<th></th>
<th>3</th>
<th>10</th>
<th>6</th>
<th>-7</th>
<th>11</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>10</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>11</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

demand by nodes in $B$ exceeds supply of nodes in $B$ plus max capacity of edges going from $A$ to $B$

Circulation with Demands

Integrality theorem. If all capacities and demands are integers, and there exists a circulation, then there exists one that is integer-valued.

Proof. Follows from max flow formulation and integrality theorem for max flow.

Characterization. Given $(V, E, c, d)$, there does not exist a circulation iff there exists a node partition $(A, B)$ such that $\sum_{v \in B} d(v) > \text{cap}(A, B)$

Proof idea. Look at min cut in $G'$.

Circulation with Demands and Lower Bounds

Feasible circulation.
- Directed graph $G = (V, E)$.
- Edge capacities $c(e)$ and lower bounds $\ell(e)$, $e \in E$.
- Node supply and demands $d(v)$, $v \in V$.

Def. A circulation is a function that satisfies:
- For each $e \in E$: $\ell(e) \leq f(e) \leq c(e)$
- For each $v \in V$: $\sum_{v \leftarrow e} f(e) - \sum_{v \rightarrow e} f(e) = d(v)$

Circulation problem with lower bounds. Given $(V, E, \ell, c, d)$, does there exist a circulation?

7.8 Survey Design
Survey Design

Survey design.
- Design survey asking $n_1$ consumers about $n_2$ products.
- Can only survey consumer $i$ about product $j$ if they own it.
- Ask consumer $i$ between $c_i$ and $c_i'$ questions.
- Ask between $p_j$ and $p_j'$ consumers about product $j$.

Goal. Design a survey that meets these specs, if possible.

Bipartite perfect matching. Special case when $c_i = c_i' = p_j = p_j' = 1$.

Algorithm. Formulate as a circulation problem with lower bounds.
- Include an edge $(i, j)$ if consumer $j$ owns product $i$.
- Integer circulation $\iff$ feasible survey design.