CSE 421
Algorithms

Lecture 21
Network Flow, Part 1

Outline

• Network flow definitions
• Flow examples
• Augmenting Paths
• Residual Graph
• Ford Fulkerson Algorithm
• Cuts
• Maxflow-MinCut Theorem

Network Flow Definitions

• Capacity
• Source, Sink
• Capacity Condition
• Conservation Condition
• Value of a flow

Flow Example

Flow assignment and the residual graph
Network Flow Definitions

• Flowgraph: Directed graph with distinguished vertices $s$ (source) and $t$ (sink)
• Capacities on the edges, $c(e) \geq 0$
• Problem, assign flows $f(e)$ to the edges such that:
  – $0 \leq f(e) \leq c(e)$
  – Flow is conserved at vertices other than $s$ and $t$
    • Flow conservation: flow going into a vertex equals the flow going out
    • The flow leaving the source is as large as possible

Flow Example

Flow assignment and the residual graph

Residual Graph

• Flow graph showing the remaining capacity
• Flow graph $G$, Residual Graph $G_R$
  – $G$: edge $e$ from $u$ to $v$ with capacity $c$ and flow $f$
  – $G_R$: edge $e'$ from $u$ to $v$ with capacity $c - f$
  – $G_R$: edge $e''$ from $v$ to $u$ with capacity $f$
Augmenting Path Algorithm

• Augmenting path
  – Vertices \( v_1, v_2, \ldots, v_k \)
  – \( v_1 = s, \ v_k = t \)
  – Possible to add \( b \) units of flow between \( v_j \) and \( v_{j+1} \) for \( j = 1 \ldots k-1 \)

Build the residual graph

\[
\begin{array}{cccc}
S & D & E & T \\
10/20 & 10/30 & 15/20 & 5/10 \\
5/10 & 15/20 & 0/10 & 5/10 \\
10/30 & 10/20 & 5/10 & 0/10 \\
15/20 & 10/30 & 5/10 & 0/10 \\
\end{array}
\]

Find two augmenting paths

Augmenting Path Lemma

• Let \( P = v_1, v_2, \ldots, v_k \) be a path from \( s \) to \( t \) with minimum capacity \( b \) in the residual graph.
• \( b \) units of flow can be added along the path \( P \) in the flow graph.

Proof

• Add \( b \) units of flow along the path \( P \)
• What do we need to verify to show we have a valid flow after we do this?
  –
  –

Ford-Fulkerson Algorithm (1956)

\[
\text{while not done}
\]
\[
\text{Construct residual graph } G_r \\
\text{Find an s-t path } P \text{ in } G_r \text{ with capacity } b > 0 \\
\text{Add } b \text{ units along in } G
\]
\[
\text{If the sum of the capacities of edges leaving } S \text{ is at most } C, \text{ then the algorithm takes at most } C \text{ iterations}.
\]