Lecture 19 – Network Flow

Reading: KT Chapter 7

Part of this content has been obtained from the official lecture slides that accompany the textbook. A complete set of slides can be found at: http://www.cs.princeton.edu/~wayne/kleinberg-tardos/

Network flow

• Examples:
  • Water pipe network
  • Computer network
  • Road network

Maximum Network flow

Maximum network flow problem

• Input:
  • Directed graph \( G = (V, E) \)
  • Each edge has a non-negative capacity \( c(e) \)
  • Source and target vertices on the graph

• Output:
  • Maximum flow that can be pushed from \( s \) to \( t \)

• Constraints:
  • Capacity constraint
  • Flow conservation
Minimum cut problem

• Input:
  • Directed graph $G = (V, E)$
  • Each edge has a non-negative capacity $c(e)$
  • Source and target vertices on the graph

• Output:
  • Minimum cut of $G$ whose capacity is minimum over all cuts of $G$
  • The net flow across a cut is the same as the maximum flow
Ford-Fulkerson algorithm

- It's simple, and practical *
- Idea behind it:
  - Find the different paths that the flow can be decomposed into

```
Flow decompositions:
s,a,b,t
s,c,a,b,t
s,c,d,b,t
s,c,d,t
```

Max-flow problem

Def. An \( \omega \)-flow (flow) \( f \) is a function that satisfies:

- For each \( e \in E \):
  \[ 0 \leq f(e) \leq \omega(e) \]  
  [capacity]
- For each \( v \in V \setminus \{ s, t \} \):
  \[ \sum_{e \in \delta^+(v)} f(e) = \sum_{e \in \delta^-(v)} f(e) \]  
  [flow conservation]

Augmenting path

Def. An augmenting path is a simple \( s \rightarrow t \) path in the residual network \( G_f \).

Def. The bottleneck capacity of an augmenting path \( P \) is the minimum residual capacity of any edge in \( P \).

Key property. Let \( f \) be a flow and let \( P \) be an augmenting path in \( G_f \). Then, after calling \textsc{AUGMENT}, the resulting \( f' \) is a flow and \( \text{val}(f') = \text{val}(f) + \text{bottleneck}(G_f, P) \).

```
\textsc{AUGMENT}(f, c, P)

\begin{algorithm}
  \STATE \textbf{b} \leftarrow \text{bottleneck capacity of path } P.
  \STATE \textbf{FOREACH} edge \( e \in P \):
  \STATE \textbf{IF} \( \omega(e) \geq f(e) \)
  \STATE \hspace{1em} \textbf{f}[e] \leftarrow f[e] + b.
  \STATE \textbf{ELSE}
  \STATE \hspace{1em} \textbf{f}[\omega(e) \rightarrow e] = f[\omega(e)] - b.
  \STATE \textbf{RETURN } f'.
\end{algorithm}
```

Ford-Fulkerson algorithm

```
\textsc{FORD–FULKERSON}(G)

\begin{algorithm}
  \STATE \textbf{FOREACH} edge \( e \in E \) \textbf{: } f[e] \leftarrow 0.
  \STATE G_f \leftarrow \text{residual network of } G \text{ with respect to } f.
  \STATE \textbf{WHILE} \{ \text{there exists an } s \rightarrow t \text{ path } P \text{ in } G_f \} \textbf{: }
  \STATE \hspace{1em} f \leftarrow \textsc{AUGMENT}(f, c, P).
  \STATE \text{Update } G_f.
  \STATE \textbf{RETURN } f.
\end{algorithm}
```

 augmenting path
Residual graph with augmenting path $p$

Augment flow $f$ along $p$
Residual graph with augmenting path $p$

Augment flow $f$ along $p$

Residual graph with augmenting path $p$

Final result