

DCC206 – Algoritmos 1

Aula 08 – Fluxo em redes

Professor Renato Vimieiro

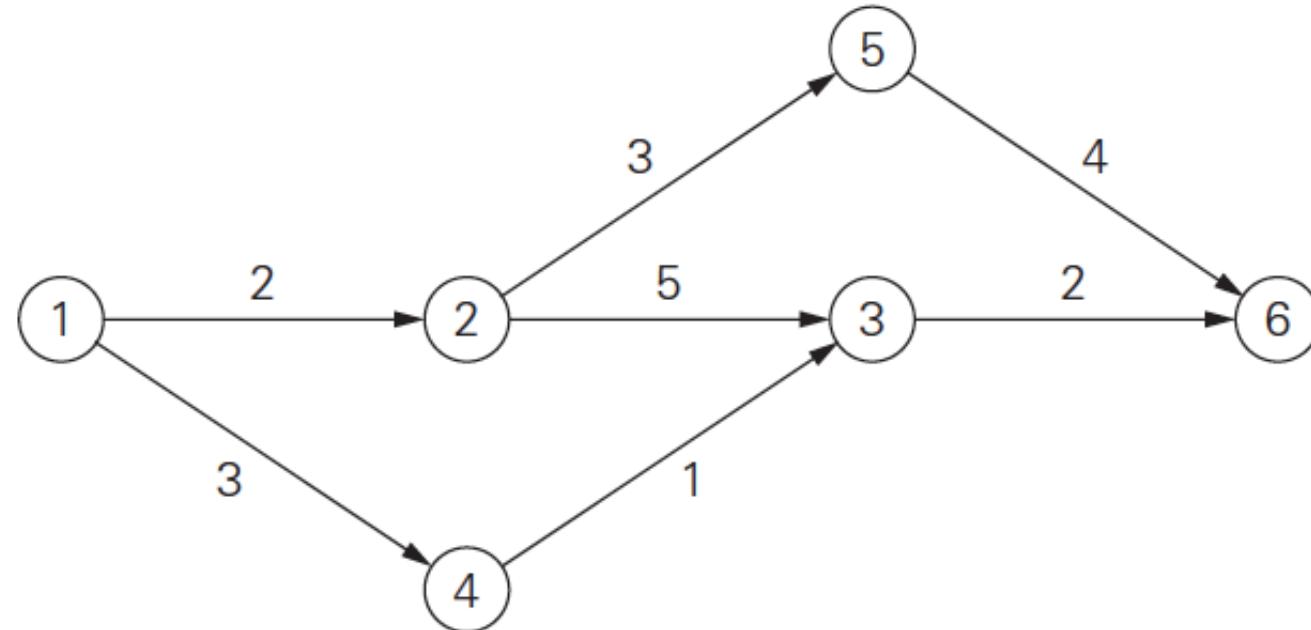
DCC/ICEx/UFMG

Introdução

- Dois grupos de pesquisa localizados, um na Europa, e o outro em BH querem compartilhar os dados coletados em alguns experimentos realizados no laboratório europeu.
- O volume de dados coletados nos experimentos é da ordem de petabytes
- Os dados poderiam ser enviados pelo correio
 - Risco de extravio dos discos
 - Corrupção dos arquivos
- Não existe link direto entre os dois institutos, mas ambos estão conectados a uma rede de universidades que permite compartilhamento dos dados

Introdução

- Dadas as diferentes situações financeiras das instituições envolvidas, os links entre pares de instituições têm largura de banda distintas
- O objetivo é maximizar o uso da banda disponível para enviar os arquivos desde a origem até o destino



Introdução

- Esse e outros problemas similares podem ser modelados usando grafos
 - Vértices: instituições conectadas à rede de institutos de pesquisa
 - Areias: links diretos entre instituições
 - Peso: largura de banda dos links
- Como estamos interessados em enviar dados de uma instituição origem para outra destino, é mais natural enxergar o grafo como direcionado
 - Também é natural supor que o grafo seja conectado e que exista caminho entre a origem e destino
- O problema em questão é encontrar um '**fluxo**' **máximo** nessa rede

Definições

- Esse tipo de modelagem se aplica a vários problemas práticos envolvendo transporte
 - Transporte de mercadorias entre localidades
 - Fluxo de energia entre produtores e consumidores
 - Fluxo de gás/petróleo entre produtores e consumidores
- A modelagem segue como descrito no exemplo anterior
- Entre os vértices existem dois especiais:
 - A **origem** que contém somente arestas saindo (chamado de **s**)
 - O **destino** que contém somente arestas chegando (chamado de **t**)
 - Os outros vértices são chamados de **intermediários**

Definições

- Os pesos associados às arestas são chamados de **capacidade** por representarem o volume máximo de itens que podem ser transportados por uma conexão
- Como o objetivo é determinar um fluxo máximo entre a origem e destino, os vértices intermediários não retém fluxo
 - Em todo vértice intermediário, o fluxo de entrada é igual ao fluxo de saída
 - Essa propriedade é chamada de **conservação de fluxo**
- Naturalmente, o fluxo enviado por uma aresta não pode ultrapassar a capacidade máxima de uma aresta
 - Essa propriedade é chamada **restrição de capacidade**

Definições

- Um **fluxo** é uma função $f: V \times V \rightarrow \mathbb{R}^+$ que mapeia as arestas de um grafo $G = (V, E)$ a valores reais não-negativos
- A **capacidade** de uma aresta (u, v) é $c(u, v)$
- Assim, as propriedades de restrição de capacidade e conservação de fluxo podem ser formalmente definidas por
 - $0 \leq f(u, v) \leq c(u, v)$ (**restrição de capacidade**)
 - $\sum_{v \in V} f(u, v) = \sum_{v \in V} f(v, u)$ para todo u diferente de s e t (**conservação de fluxo**)
- O fluxo ‘transmitido’ numa rede é $F = \sum_{v \in V} f(s, v)$

Definições

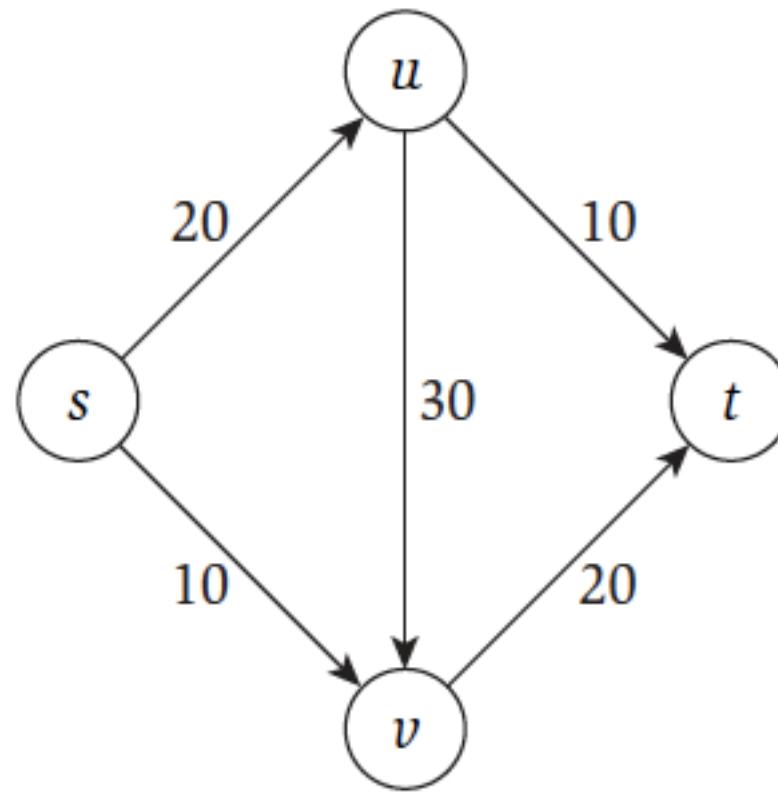
- Para simplificar as notações, definimos:
 - $f^+(v) = \sum_{u \in V} f(u, v)$ (fluxo de entrada em v)
 - $f^-(v) = \sum_{u \in V} f(v, u)$ (fluxo de saída de v)
 - $f^+(X) = \sum_{v \in X} f^+(v)$ (fluxo de entrada nos vértices de $X \subseteq V$)
 - $f^-(X) = \sum_{v \in X} f^-(v)$ (fluxo de saída dos vértices de $X \subseteq V$)
- Logo,
 - $F = f^-(s)$
 - $f^+(v) = f^-(v)$

Fluxo máximo e cortes

- Um corte em um grafo $G=(V,E)$ é uma partição dos vértices em dois conjuntos A e B
- Seja A e B uma partição de uma rede $G=(V,E)$ tal que s pertence a A, e t pertence a B
- Todo fluxo de s para t tem de obrigatoriamente passar pelo menos por uma aresta que cruza A e B
- O fluxo máximo está limitado pelas arestas que cruzam o corte
 - Um corte mínimo (com menor capacidade) é um limite superior para o fluxo máximo (a ser provado posteriormente)

Algoritmo de Ford-Fulkerson

- Vamos considerar o seguinte exemplo:



Algoritmo de Ford-Fulkerson

- Podemos inicializar todos os fluxos como zero
 - Claramente, isso não ultrapassa a capacidade das arestas e não fere a conservação de fluxo
- Em seguida, podemos buscar um caminho entre s e t cujos fluxos nas arestas que ainda não saturaram sua capacidade
 - Por exemplo, o caminho $s-u-v-t$ permite aumentar o fluxo em 20 unidades nessas arestas
 - O novo fluxo atribuído ainda preserva as propriedades da rede
- Esse novo fluxo é máximo?

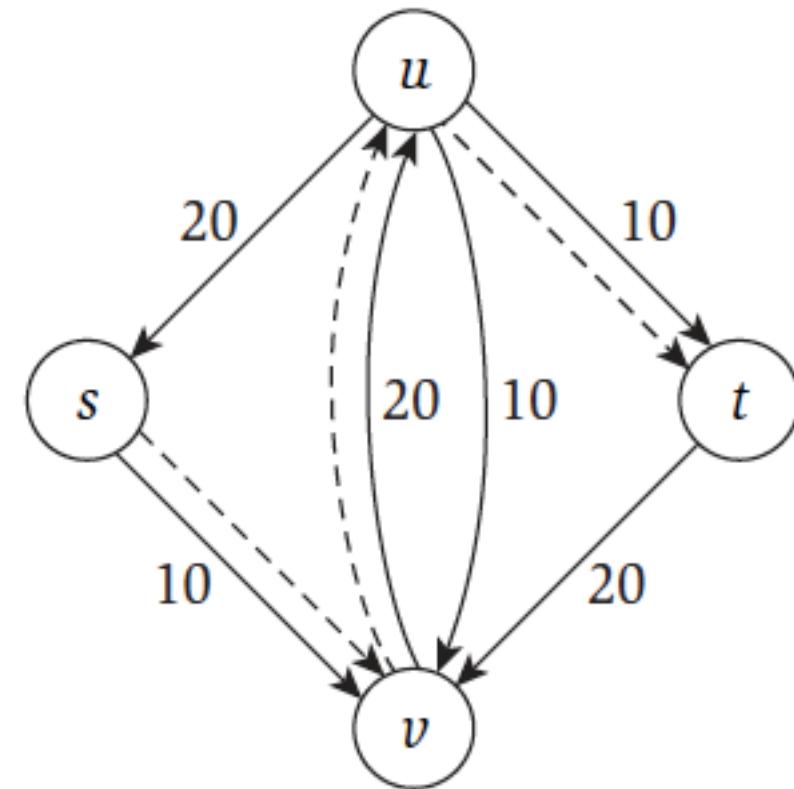
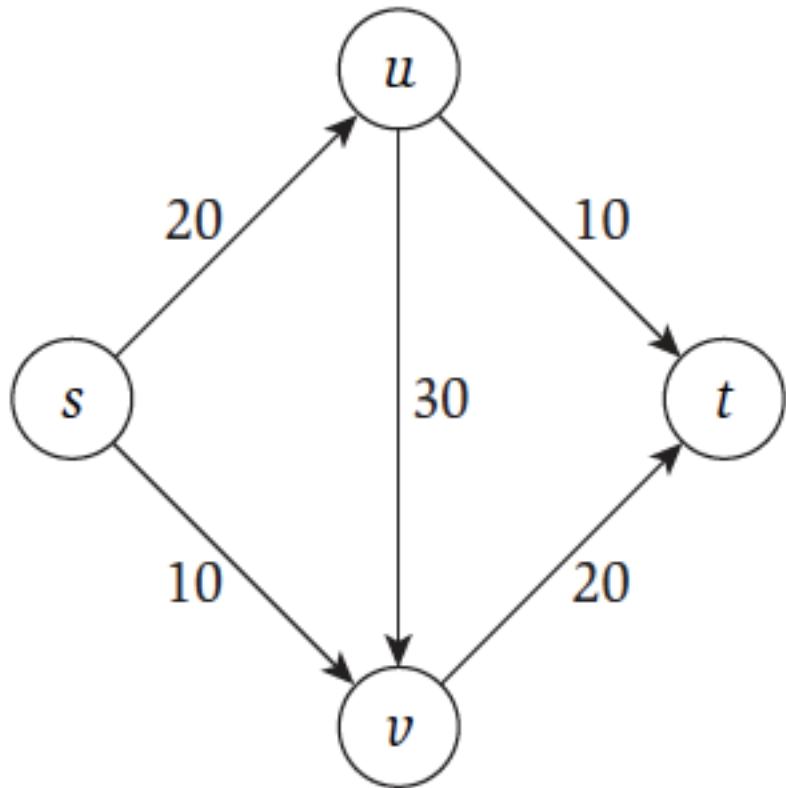
Grafos Residuais

- Observamos que o novo fluxo ainda não é máximo
- Podemos obter um fluxo de valor 30 escolhendo outro caminho
- Ao invés de mandar 20 unidades de u para v, poderíamos mandar somente 10 e a sobra ser enviada para u-t
- Além disso, essa nova configuração permitiria enviar 10 pela aresta s-v sem saturar v
- De uma maneira geral, temos duas possibilidades em relação ao fluxo atual
 - Aumentar o fluxo na direção da aresta (u,v) (chamada de aresta para frente)
 - ‘Devolver’ o fluxo na aresta (u,v) para u (criar uma aresta (v,u) com fluxo $f(u,v)$)

Grafos Residuais

- De fato, podemos construir um grafo que modelo os resíduos nas arestas
- Seja $G=(V,E)$ uma rede e f um fluxo dessa rede, o grafo de resíduos para G e f é construído como a seguir
- $G_f = (V, E_f)$
- Toda aresta (u,v) tal que $c(u,v)-f(u,v) > 0$ é incluída em E_f com capacidade $c(u,v)-f(u,v)$. Essas arestas são chamadas de **arestas para frente (forward edges)**.
- Para toda aresta (u,v) tal que $f(u,v) > 0$, (v,u) é incluída em E_f com capacidade $f(u,v)$. Essas arestas são chamadas de **arestas para trás (backward edges)**.

Grafos residuais



Caminhos aumentantes

- Seja P um caminho simples entre s e t no grafo residual de $G=(V,E)$ e f .
- P é chamado de **caminho aumentante**
- Vamos definir o **gargalo** de P como a menor capacidade residual em G_f
 - $\text{gargalo}(P, f) = \min c(u, v)$
- Agora podemos definir a função aumentarFluxo

Caminhos aumentantes

aumentarFluxo(f, P)

$r = \text{gargalo}(P, f)$

$f' = f$

para (u, v) em P

se (u, v) é uma aresta para frente

$$f'(u, v) = f(u, v) + r$$

senão ((u, v) é uma aresta para trás)

$$f'(v, u) = f(v, u) - r$$

retornar f'

Caminhos aumentantes

- O novo fluxo f' é obtido aumentando o fluxo f com o menor resíduo
- **Lema:** O fluxo aumentado f' é um fluxo em G
- **Prova:** Devemos mostrar que as propriedades de restrição de capacidade e conservação de fluxo são mantidas
- **Restrição de capacidade:** Como f e f' diferem apenas nas arestas do caminho aumentante P , precisamos apenas verificar se essas arestas preservam a restrição de fluxo.
 - Se (u,v) é uma aresta para frente, $0 \leq f(u,v) \leq f'(u,v) = f(u,v) + r \leq f(u,v) + (c(u,v)-f(u,v)) = c(u,v)$
 - Se (u,v) é uma aresta para trás, $c(u,v) \geq f(u,v) \geq f'(u,v) = f(u,v) - r \geq f(u,v) - f(u,v) = 0$

Caminhos aumentantes

- **Conservação de fluxo:** Existem quatro possibilidades para atualização do fluxo em um vértice i . É fácil perceber que em todas as configurações o fluxo é conservado.

$$\xrightarrow{+r} i \xrightarrow{+r}, \quad \xrightarrow{+r} i \xleftarrow{-r}, \quad \xleftarrow{-r} i \xrightarrow{+r}, \quad \xleftarrow{-r} i \xleftarrow{-r}$$

Algoritmo de Ford-Fulkerson

Fluxo-Maximo(G, s, t)

para $(u, v) \in E$

$f[u, v] = 0$

enquanto existir caminho aumentante em G_f

$P = \text{caminho}(s, t, G_f)$

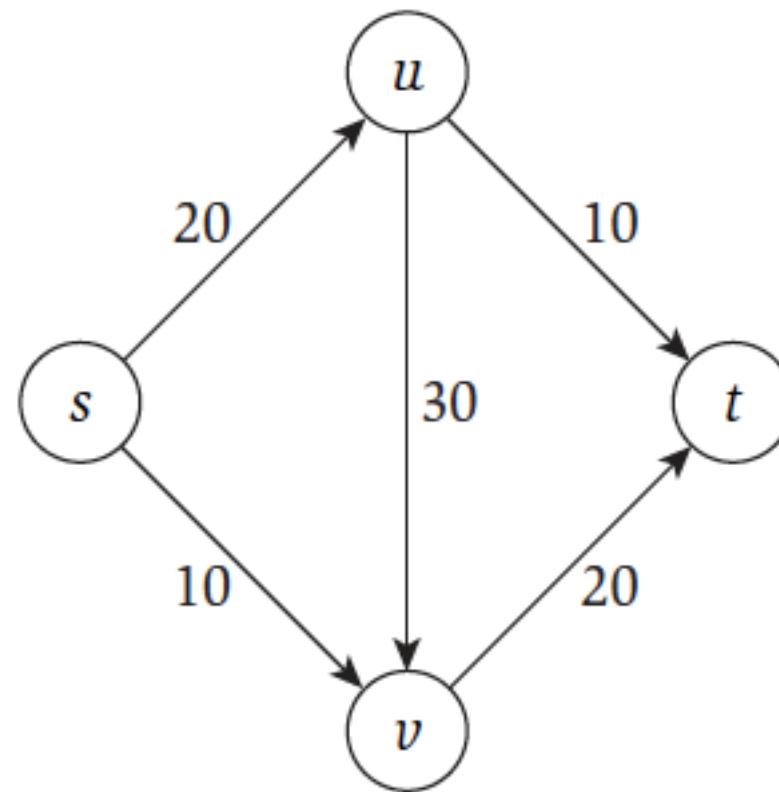
$f' = \text{aumentarFluxo}(f, P)$

$f = f'$

$G_f = \text{grafoResidual}(G, f)$

retornar f

Algoritmo de Ford-Fulkerson



Análise da complexidade do algoritmo

- O tempo de execução do algoritmo de Ford-Fulkerson é muito dependente de como o caminho aumentante é computado
- Em alguns casos, o algoritmo pode inclusive não terminar (se as capacidades fossem número irracionais – resultado de interesse teórico)
- Se as capacidades forem inteiros, então o algoritmo converge
- Usaremos esse caso para analisar o algoritmo

Análise da complexidade do algoritmo

- A inicialização do algoritmo tem custo $O(E)$
- Suponha que o fluxo máximo do grafo seja F . Como o caminho aumentante deve gerar um novo fluxo maior, os incrementos devem ser sempre positivos. Portanto, ele será no mínimo 1.
- Dessa forma, o ciclo principal é executado, no máximo, F vezes

Análise da complexidade do algoritmo

- Computar um caminho em G_f tem custo $O(|E|)$ (busca em largura ou profundidade, grafo conectado e $|E_f| \leq 2|E|$)
- A função aumentarFluxo avalia todas as arestas do caminho simples P. Logo o custo da função é $O(|V|)$
- A construção do grafo residual custa $O(|E|)$
- Portanto, o custo total do ciclo principal é $O(F|E|)$

Leitura

- Seção 7.1 (Kleinberg e Tardos)