Fluxo em redes
5189-32

Rodrigo Calvo
rcalvo@uem.br

Departamento de Informática – DIN
Universidade Estadual de Maringá – UEM

1° semestre de 2016

Introdução

- Uma rede $G = (V, A)$ é um grafo direcionado no qual cada aresta $(u, v) \in A$ tem uma capacidade $c(u, v) \geq 0$
- Se G contém a aresta (u, v) ele não pode conter (v, u)
- Se $(u, v) \notin A$, então $c(u, v) = 0$
- Destaca-se dois vértice s (**fonte**) e t (**sumidouro**)
- Para cada vértice $v \in V$, temos $s \rightsquigarrow v \rightsquigarrow t$

Introdução

- Um fluxo em G é uma função $f : V \times V \rightarrow \mathbf{R}$ que satisfaz as seguintes propriedades
- **Restrição de capacidade**: Para todo $u, v \in V$, $0 \leq f(u, v) \leq c(u, v)$
- **Conservação do fluxo**: Para todo $u \in V - \{s,t\}$

$$\sum_{v \in V} f(v,u) = \sum_{v \in V} f(u,v)$$

- A quantidade $f(u, v)$ é chamada de fluxo entre u e v. Quando $(u, v) \notin A$, não pode haver fluxo de u para v e portanto $f(u, v) = 0$

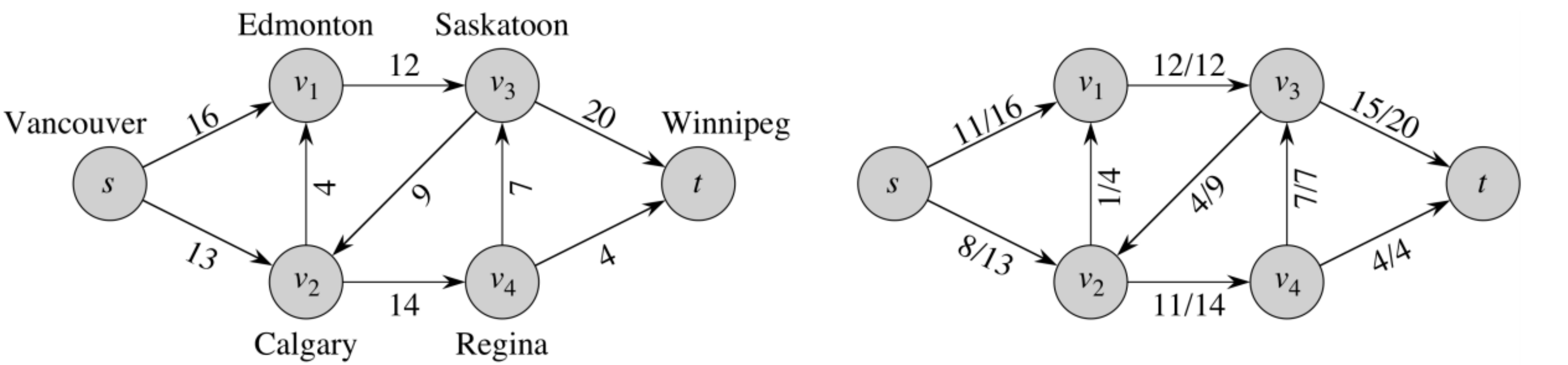
Introdução

- O valor $|f|$ do fluxo f é definido como:

$$|f| = \sum_{v \in V} f(s,v) - \sum_{v \in V} f(v,s)$$

ou seja, o fluxo total que sai de *s* menos o fluxo total que entra em *s*

Exemplo



Edmonton
Saskatoon
Vancouver
Winnipeg
Calgary
Regina
s
v_1
v_2
v_3
v_4
t
16
13
12
14
4
9
7
20
4
s
v_1
v_2
v_3
v_4
t
11/16
8/13
12/12
11/14
1/4
4/9
7/7
15/20
4/4

(a) (b)

Problema do fluxo máximo

- Dado uma rede *G*, uma **fonte** *s* e um **sumidouro** *t*, o **problema do fluxo máximo** consiste em encontrar um fluxo em *G* de valor máximo.

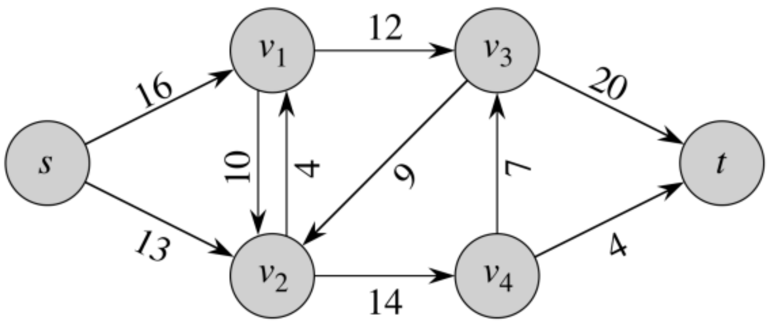
Criando modelos

- Arestas antiparalelas
  - Suponha que a firma de caminhões oferecesse para Lucky Puck a oportunidade de transportar 10 caixotes nos caminhões indo de Edmonton para Calgary
  - Parece uma boa oportunidade para inserir uma aresta no grafo
  - O problema é que isto viola a restrição de que se $(v1, v2) \in A$, então $(v2, v1) \notin A$ (as arestas $(v1, v2)$ e $(v2, v1)$ são chamadas de **antiparalelas**)

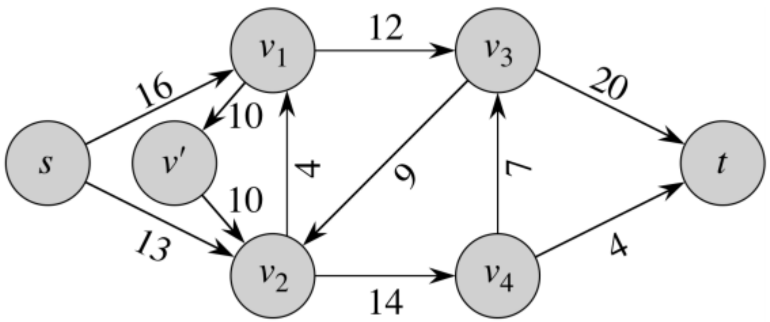
Criando modelos

- Arestas antiparalelas
  - Suponha que a firma de caminhões oferecesse para Lucky Puck a oportunidade de transportar 10 caixotes nos caminhões indo de Edmonton para Calgary
  - Parece uma boa oportunidade para inserir uma aresta no grafo
  - O problema é que isto viola a restrição de que se ($v1$, $v2$) $\in A$, então ($v2$, $v1$) $\notin A$ (as arestas ($v1$, $v2$) e ($v2$, $v1$) são chamadas de **antiparalelas**)
  - Podemos transformar a rede em uma rede equivalente sem arestas antiparalelas
    - Escolhemos uma aresta, neste caso ($v1$, $v2$), e a dividimos adicionando um v' para substituir aresta ($v1$, $v2$) pelas arestas ($v1$, v') e (v', $v2$)

Criando modelos



(a)

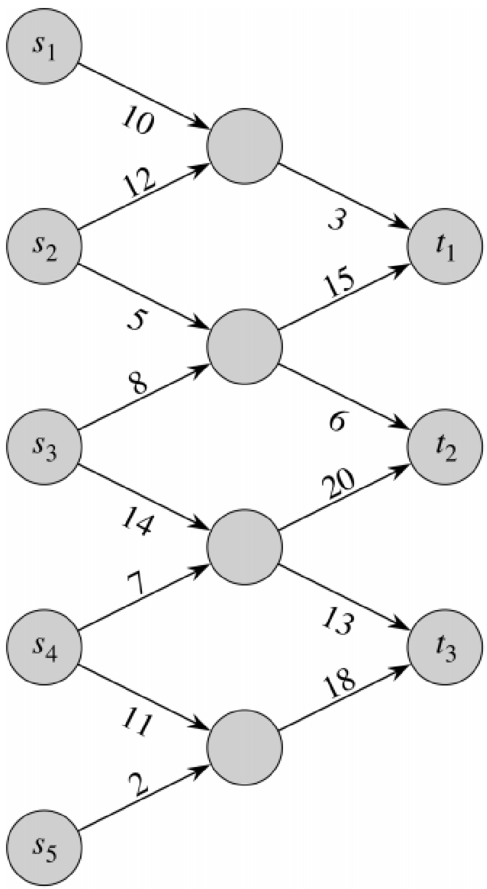


(b)

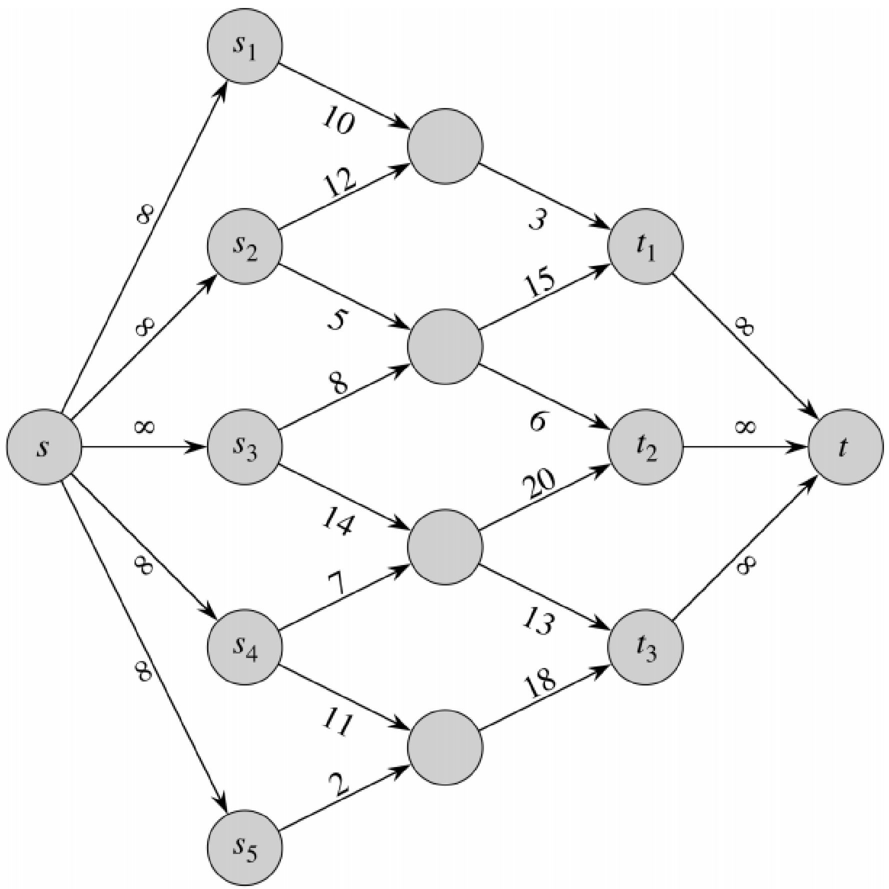
Criando modelos

- Redes com múltiplas **fontes** e **sumidouros**
  - A empresa poderia ter mais que uma fábrica e mais que um depósito
  - Não está de acordo com a definição de rede
  - Transformamos em uma rede equivalente
    - Adicionamos uma **superfonte** s e arestas com capacidade ∞ de s para cada fonte original
    - Adicionamos um **supersumidouro** e arestas com capacidade ∞ de cada sumidouro original para o **supersumidouro**

Criando modelos



(a)



(b)

O Método de Ford-Fulkerson

- Chamamos de método e não algoritmo pois engloba diversas implementações com tempo de execução diferentes
- Utiliza os conceitos: rede residual, caminho aumentante e corte. Estes conceitos são importantes para muitos algoritmos e problemas de fluxo em rede
- Ideia
  - Incrementar iterativamente o valor do fluxo
  - Inicialmente, têm-se $f(u, v) = 0$ para todo $u, v \in G$, o que gera um fluxo de valor 0
  - A cada iteração, o valor do fluxo é incrementado, determinando um "caminho aumentante" na "rede residual" G_f associada a G
  - O processo continua até que nenhum caminho aumentante é encontrado
  - O teorema do fluxo máximo e corte mínimo garante que este processo produz o fluxo máximo no término

Método de Ford-Fulkerson

```
ford-fulkerson-method(G, s, t)
Iniciar o fluxo f com 0
while existe um caminho aumentante p na rede residual Gf
  aumente f ao longo de p
return f
```

- A fim de implementar e analisar o método de Ford-Fulkerson, precisamos de vários conceitos

Redes residuais

- Intuitivamente, uma rede residual G_f de uma rede G e um fluxo f consiste de arestas com capacidades que representam como o fluxo das arestas de G podem ser alterados
  - O fluxo em uma aresta pode aumentar ou diminuir

Redes residuais

- Seja $G = (V, E)$ uma rede com fonte s e sumidouro t, f um fluxo em G e $u, v \in V$
- A capacidade residual $c_f(u, v)$ é definida como

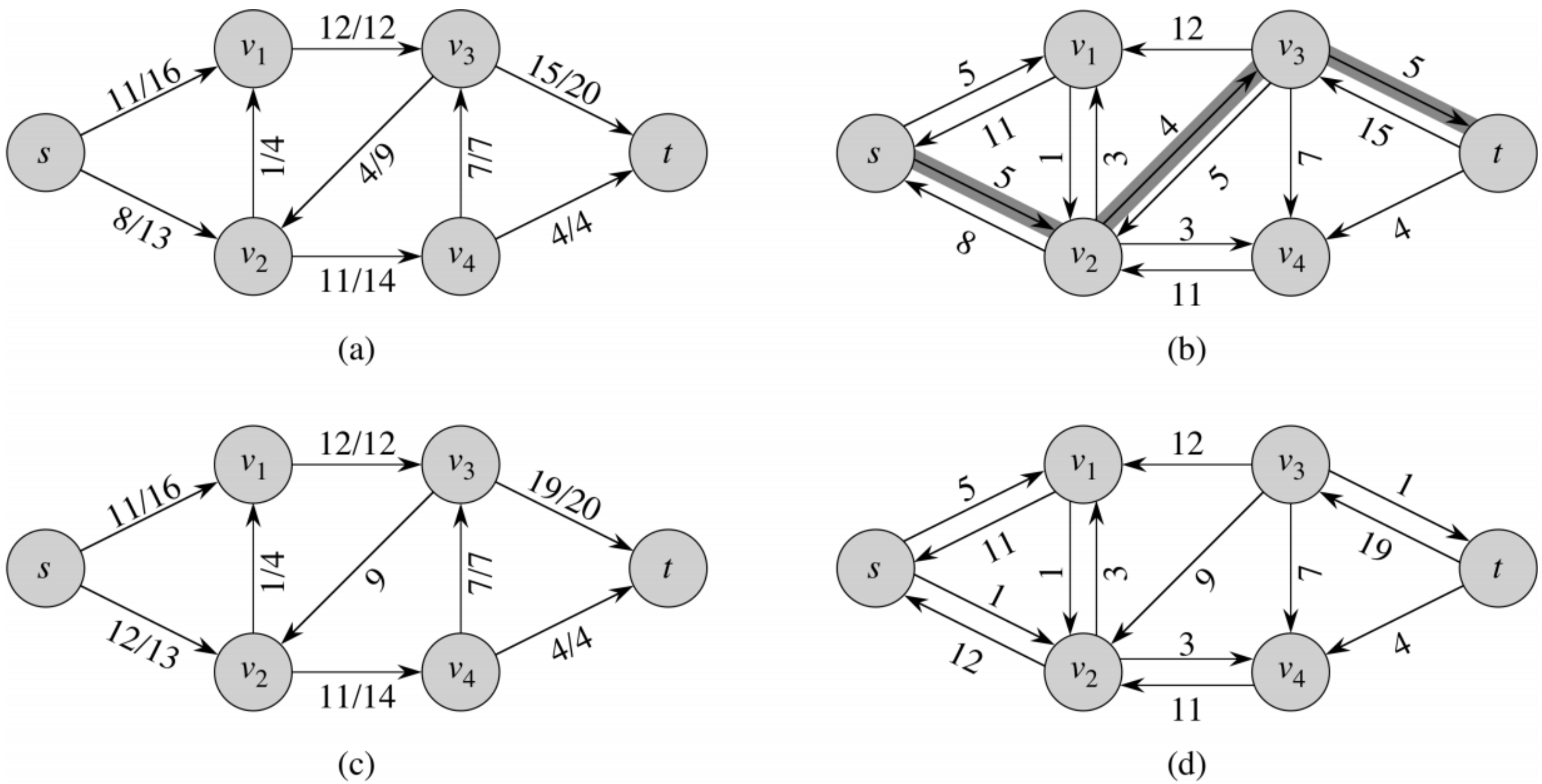
$$c_f(u,v) = \begin{cases} c(u,v) - f(u,v) & ,\text{se } (u,v) \in A \\ f(v,u) & ,\text{se } (v,u) \in A \\ 0 & ,\text{caso contrário} \end{cases}$$

- A rede residual de G induzida por f é $G_f = (V, A_f)$, onde

$$A_f = \{(u,v) \in V \times V : c_f(u,v) > 0\}$$

$$|A_f| \leq 2|A|$$

Exemplo de rede residual



Redes residuais

- Se *f* é um fluxo em *G* e *f'* é um fluxo na rede residual G_f correspondente, definimos $f \uparrow f'$, o aumento do fluxo *f* por *f'*, como sendo a função $V \times V \rightarrow \mathbf{R}$

$$(f \uparrow f')(u,v) = \begin{cases} f(u,v) + f'(u,v) - f'(v,u) & \text{, se } (u,v) \in A \\ 0 & \text{, caso contrário} \end{cases} \qquad (26.4)$$

Caminho aumentante

- Dado uma rede $G = (V, A)$ e um fluxo f, um **caminho aumentante** p é um caminho simples de s para t na rede residual G_f
- O valor máximo que pode ser aumentado no fluxo de cada aresta no caminho aumentante p é chamado **capacidade residual** de p, e é dado por:

$$c_f(p) = \min \{ c_f(u,v) : (u,v) \in$$

Caminho aumentante

- Lema 26.2
    - Seja $G = (V, A)$ um rede, f um fluxo em G e p um caminho aumentante em G_f. Seja a função $f_p : V \times V \to \mathbf{R}$, definida como

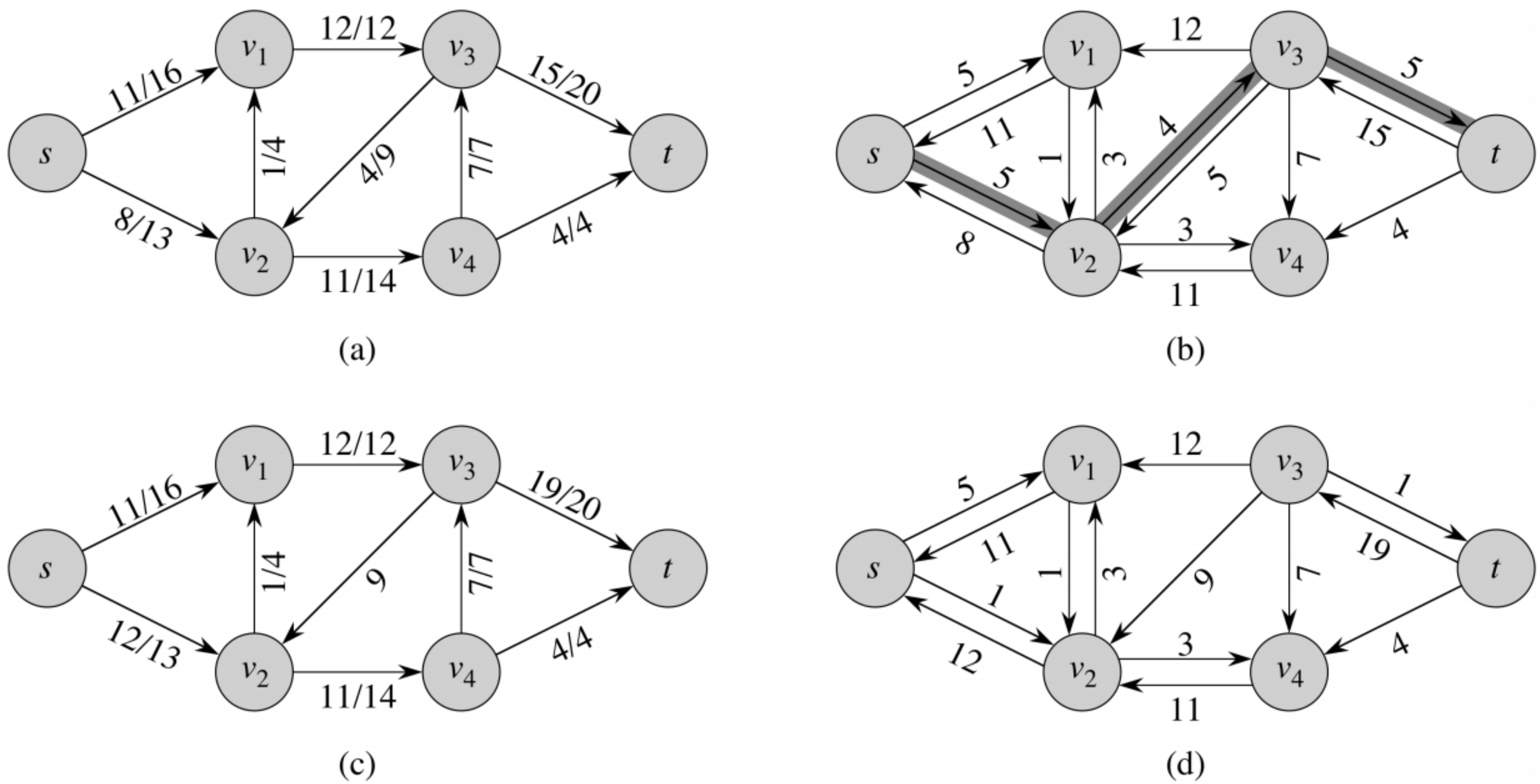
$$f_p(u,v) = \begin{cases} c_f(p) & , \text{se } (u,v) \in p \\ 0 & , \text{caso contrário} \end{cases} \qquad (26.8)$$

Então, f_p é um fluxo em G_f com valor $|f_p| = c_f(p) > 0$

Caminho aumentante

- Corolário 26.3
  - Seja G = (V, A) um rede, f um fluxo em G e p um caminho aumentante em G_f . Seja a função f_p como definido na equação (26.8) e suponha que nós aumentamos f por f_p. Então a $f \uparrow f_p$ é um fluxo em G com valor $| f \uparrow f_p| = |f| + |f_p| > |f|$
  - Prova: a partir dos lemas 26.1 e 26.2 (Cormem)

Caminho aumentante



Corte de rede

- Um corte (S, T) de uma rede G = (V, A) é uma partição de V em S e T = V − S, tal que $s \in S$ e $t \in T$
- Se f é um fluxo, então o **fluxo líquido** f(S, T) através do corte (S,T) é definido como

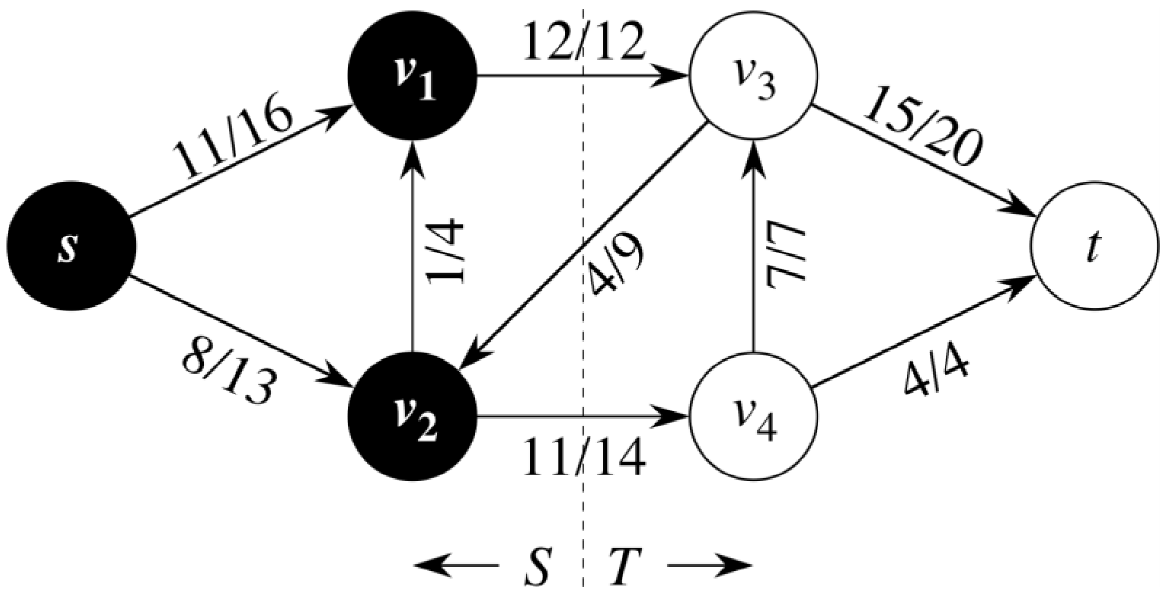
$$f(S,T) = \sum_{u \in S} \sum_{v \in T} f(u,v) - \sum_{u \in S} \sum_{v \in T} f(v,u)$$

- A **capacidade** do corte (S, T) é

$$c(S,T) = \sum_{u \in S} \sum_{v \in T} c(u,v)$$

- Um **corte mínimo** de uma rede é um corte que tem capacidade mínima entre todos os cortes da rede

Exemplo corte



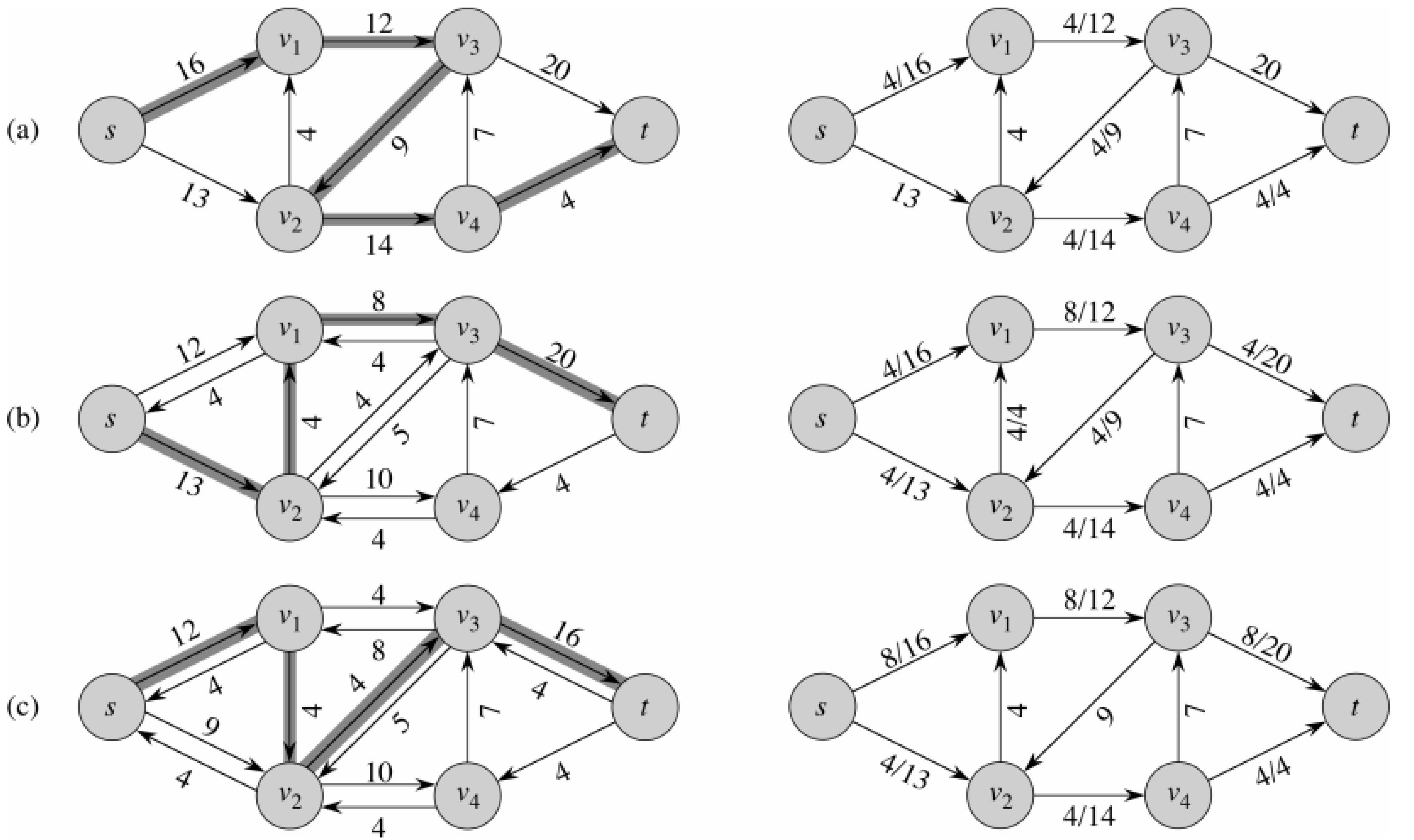
Algoritmo básico de Ford-Fulkerson

- Em cada iteração do método de Ford-Fulkerson é encontrado algum caminho aumentante *p* que é utilizado para modificar o fluxo *f*
- Como o Lema 26.2 e o Corolário 26.3 sugerem, o fluxo *f* pode ser substituído por $f \uparrow f_p$, gerando um novo fluxo com valor $|f| + |f_p|$
- Vamos ver uma implementação
  - Cada aresta residual em *p* é uma aresta na rede original ou uma aresta contrária na rede original
  - Fluxo é adicionado se a aresta é a original
  - Fluxo é removido se a aresta é contrária
  - Quando não existe mais caminho aumentante, *f* é máximo

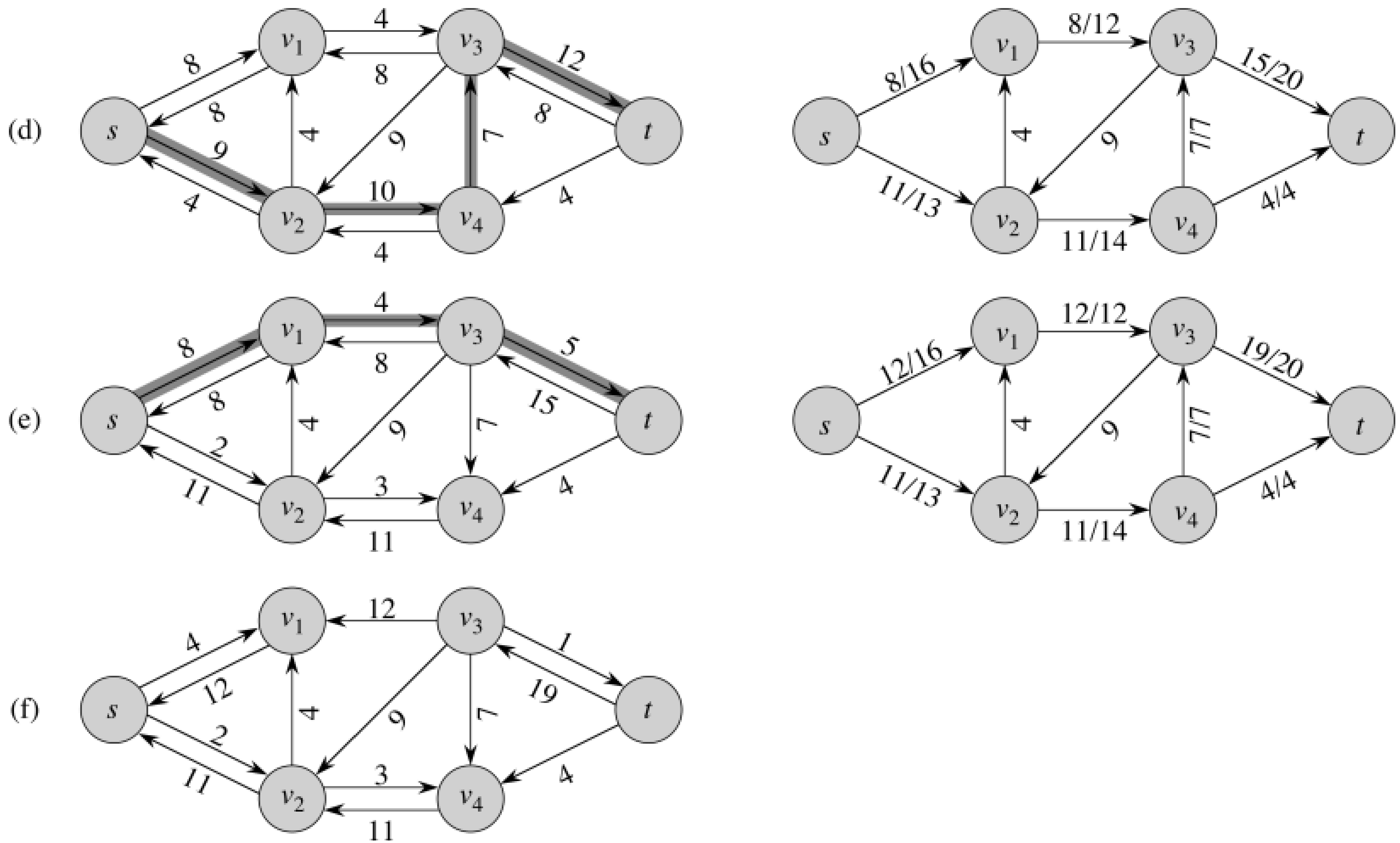
Algoritmo básico de Ford-Fulkerson

```
ford-fulkerson(G,s,t)
for cada aresta (u, v) ∈ G.A
  (u, v).f = 0
while existe um caminho p de s até t na rede residual G_f
  c_f(p) = min{c_f(u, v) : (u, v) ∈ p}
  for cada aresta (u, v) ∈ p
    if (u, v) ∈ A
      (u, v).f = (u, v).f + c_f(p)
    else
      (v, u).f = (v, u).f − c_f(p)
```

Exemplo de execução



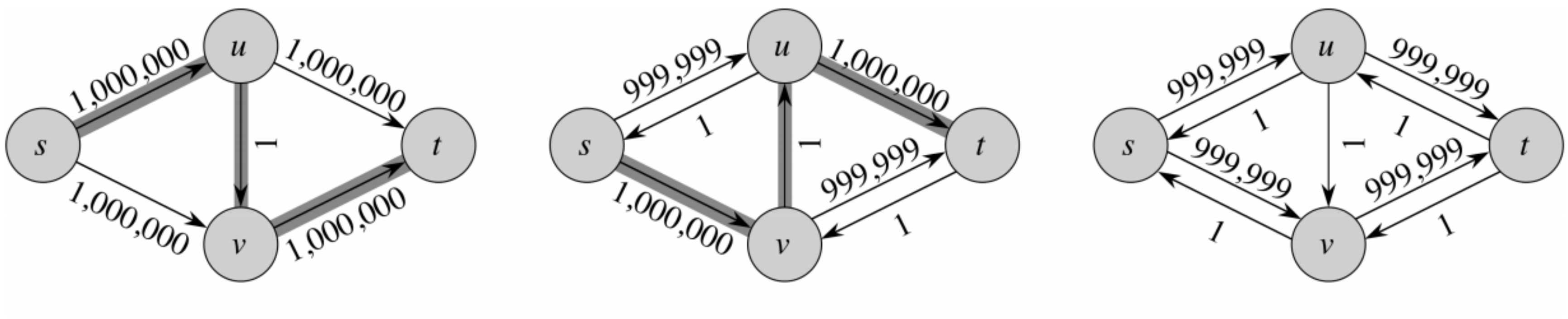
Exemplo de execução



Análise do algoritmo de Ford-Fulkerson

- Depende de como o caminho p é escolhido
- Vamos supor que todas as capacidades sejam inteiras
- Seja f^* o fluxo máximo na rede residual
- Então, o laço while das linhas 3-8 executa no máximo $|f^*|$, isto porque o valor do fluxo aumenta em pelo menos uma unidade em cada iteração
- O conteúdo dentro do while pode ser executado de forma eficiente se escolhermos a estrutura correta para representar a rede e se o caminho aumente for encontrado em tempo linear
    - Manter um grafo $G' = (V, A')$, onde $A' = \{(u, v) : (u, v) \in A \text{ ou } (v, u) \in A\}$
    - Encontrar o caminho aumentante com dfs ou bfs, tempo $O(V + A') = O(A)$
    - Cada iteração demora $O(A)$
- Portanto, o tempo de execução do algoritmo é $O(A\ |f^*|)$

Exemplo ruim



(a) (b) (c)

Algoritmo de Edmonds-Karp

- Encontrar o caminho aumentante *p* com a busca em largura
- Escolher o menor caminho entre *s* e *t*, sendo que o tamanho do caminho é o número de arestas no caminho
- Executa em O(VA^2)

Bibliografia

- Thomas H. Cormen et al. Introduction to Algorithms. 3rd edition. Capítulo 26.