Universidade Federal de Viçosa
Departamento de Informática
Centro de Ciências Exatas e Tecnológicas

Fluxo em Redes

(capítulo 11 do livro GTWA, 26 do livro CLRS)

Universidade Federal de Viçosa

Fluxo em Redes

- Seja G um grafo direcionado valorado.
- Uma possível interpretação para o "peso" de cada aresta é a representação de "capacidade de fluxo".
- Exemplos:
    - Rede de internet, onde o peso de cada aresta representa a largura de banda de uma conexão;
    - Rede de tubos onde o peso da aresta representa a quantidade de litros de água/óleo/gás que pode passar a cada segundo.
    - Rede elétrica onde o peso das arestas representa a quantidade de corrente elétrica que pode passar em cada fio.
    - Rede de ruas onde o peso de cada rua é o número de carros que podem passar a cada minuto nela.

Fluxo em Redes

- Em geral, considera-se que as redes são grafos **direcionados valorados**, onde o peso *c(a)* de cada arco *a* representa sua capacidade máxima de “fluxo”.
- Além disso, há dois vértices especiais na rede chamados de **fonte** (ou origem) e **sorvedouro** (ou destino).
- A **fonte** (normalmente denotada por *s*) possui grau de entrada 0 – é a partir dela que o “fluxo” se **origina**.
- O **sorvedouro** (normalmente denotado por *t*) possui grau de saída 0 – é para ele que o “fluxo” **escoa**.
- Os outros nodos são chamado de nodos de transbordo (eles não produzem nem consomem fluxo – apenas o repassam).

Fluxo em Redes

- Seja v um vértice qualquer de uma rede G:
  - $in(v)$ é o conjunto de arcos que chegam em v.
  - $out(v)$ é o conjunto de arcos que saem de v.

Fluxo em Redes

- Em uma rede *G* com fluxo viável, temos um fluxo com valor $f(a) \geq 0$ associado a cada arco *a*, onde:
  - $f(a) \leq c(a)\ \forall a \in E(G)$
    - restrição de capacidade dos arcos
  - $\sum_{\forall a \in in(v)} f(a) = \sum_{\forall a \in out(v)} f(a)$, para todo vértice $v \neq s,t$.
    - "fluxo que entra é igual ao fluxo que sai – conservação de fluxo"
  - Exemplos.

Fluxo em Redes

- Exemplo: rede de água composta por tubos e conexões entre tubos.
  - A quantidade de água que passa em um tubo deve ser menor ou igual à capacidade desse tubo.
  - A quantidade de fluxo que chega em uma conexão de tubos deve ser igual à quantidade de fluxo que sai dela (a água não deve acumular nem vazar).

Fluxo em Redes

- O **fluxo** em uma rede *G,* denotado por *val(f),* é definido como sendo a quantidade total de fluxo que sai da fonte *s*.
  - Ou seja, $val(f) = \sum_{\forall a \in out(s)} f(a)$

- Por conservação de fluxo, temos também que:
  - $val(f) = \sum_{\forall a \in in(t)} f(a)$

Fluxo em Redes

- Um problema importante relacionado a fluxo em redes é o *problema do fluxo máximo.*
- O problema do fluxo máximo consiste em determinar um fluxo viável f^* tal que $val(f^*) \geq val(f)$ para todo fluxo viável f – ou seja, encontrar o fluxo de maior valor possível em uma rede.
- **Vários** problemas de grafos podem ser reduzidos ao problema do fluxo máximo.
- Exemplo.

Fluxo em Redes

- Algoritmo para o problema do fluxo máximo: baseado na ideia de caminhos aumentantes.

- Conceitos utilizados:
  - Rede de fluxo G_f de G: Rede onde o peso de cada arco representa o fluxo corrente nesse arco.
  - Rede residual G_r de G: rede formada pela capacidade "restante" (ou residual) de cada arco de G.
    - Ou seja, o valor de cada arco representa a diferença entre a capacidade do arco e seu fluxo corrente em G_f.
    - Obs: arcos cuja capacidade restante sejam 0 são removidos de G_r.

Fluxo em Redes

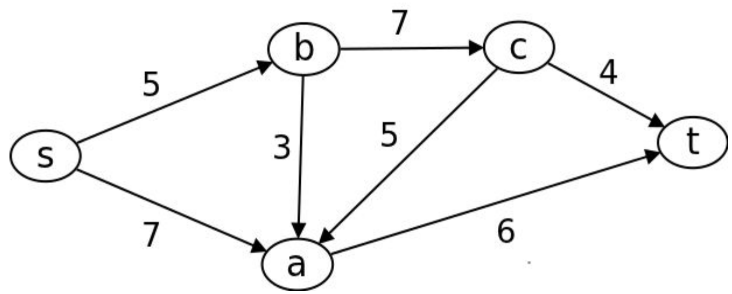
- Conceitos utilizados:
  - Caminho aumentante: qualquer caminho direcionado em G_r que sai de *s* e chega em *t*.
  - Capacidade residual *Cr* de um caminho aumentante *P*: é a menor capacidade residual de um arco do caminho *P.*
    - *Cr(P) = min(capacidade(a)* para cada aresta *a* em *P)*
    - Note que a capacidade residual é a maior quantidade de fluxo adicional que pode passar por *P* sem violar alguma restrição do problema.
  - Exemplos.

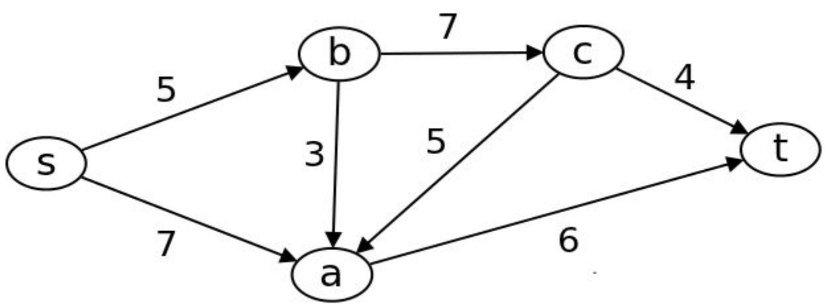
Fluxo em Redes

- **Possível** ideia para se obter o fluxo máximo: (pular para Ford-Fulkerson…)
  - Encontrar um caminho aumentante *P* em G_r.
  - Aumentar o fluxo em *Cr(P)* unidades.
  - Atualizar G_f e G_r.
  - Repetir o processo acima enquanto houver um caminho aumentante *P* em G_r.

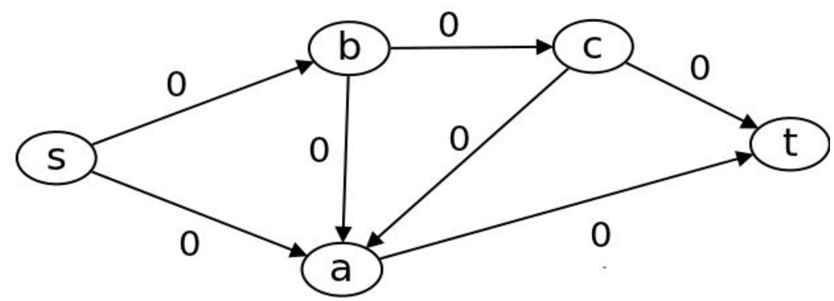
Fluxo em Redes

- Exemplo:

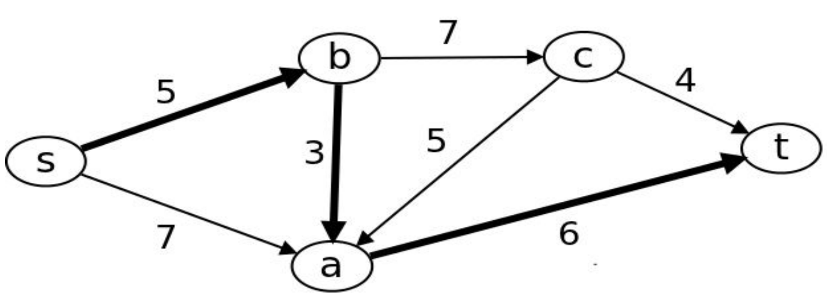




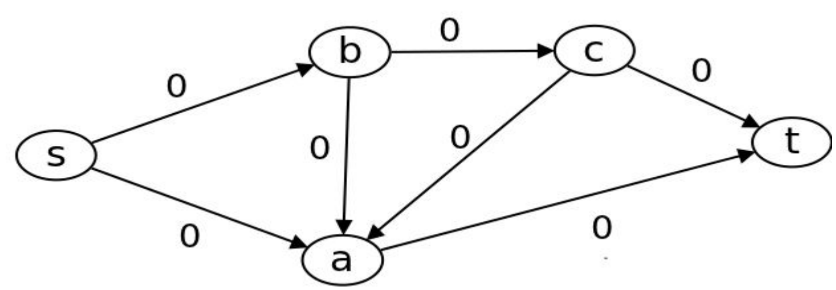
Rede residual



Rede de fluxo



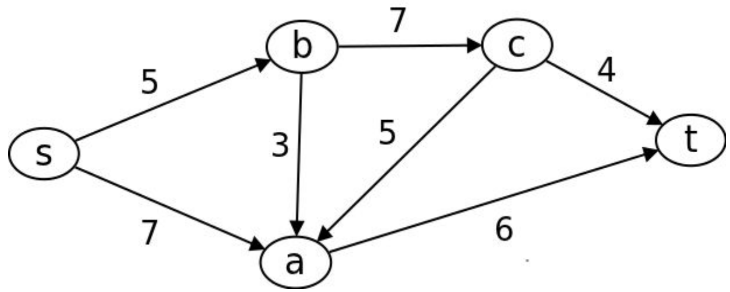
Rede residual

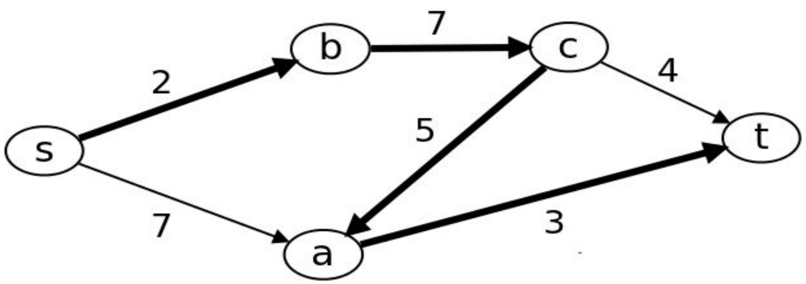


Rede de fluxo

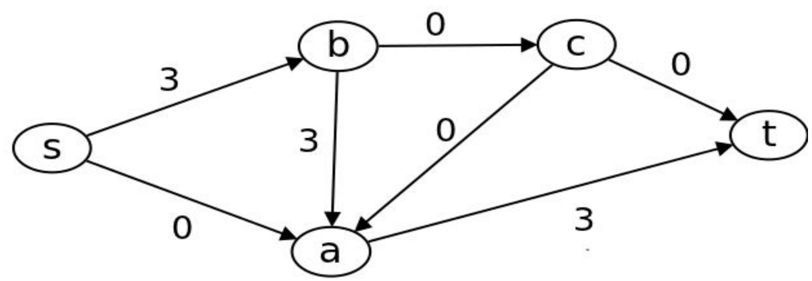
Fluxo em Redes

- Exemplo:

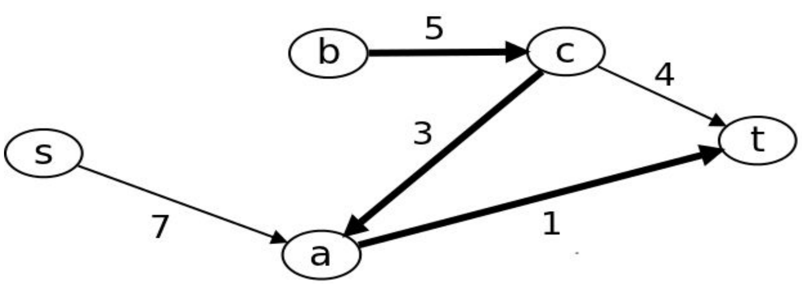




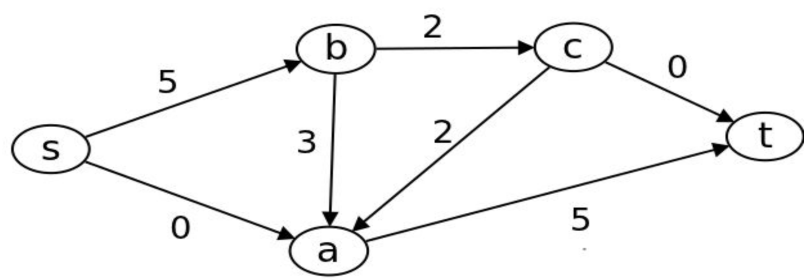
Rede residual



Rede de fluxo



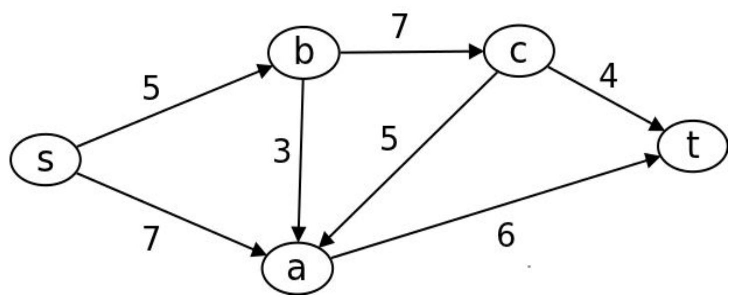
Rede residual

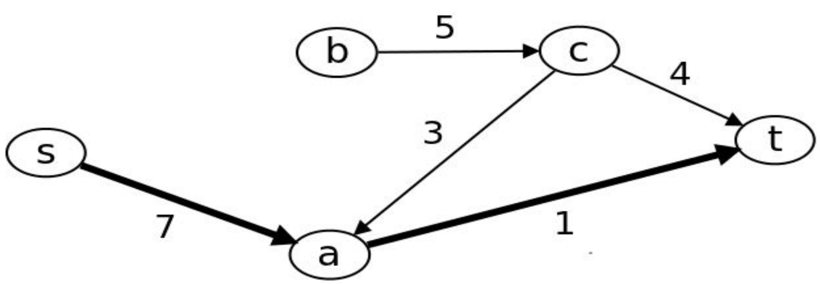


Rede de fluxo

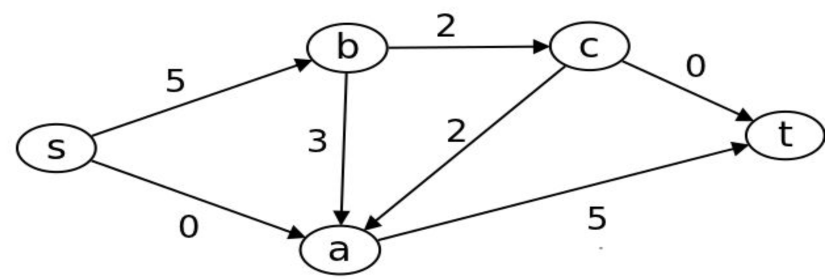
Fluxo em Redes

- Exemplo:

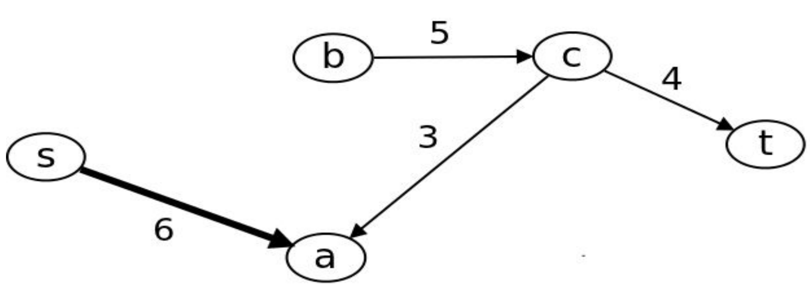




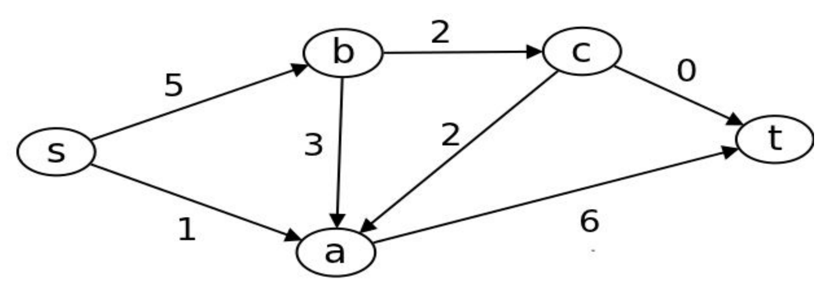
Rede residual



Rede de fluxo



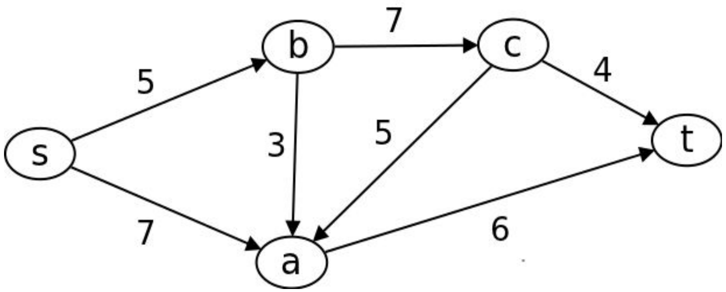
Rede residual

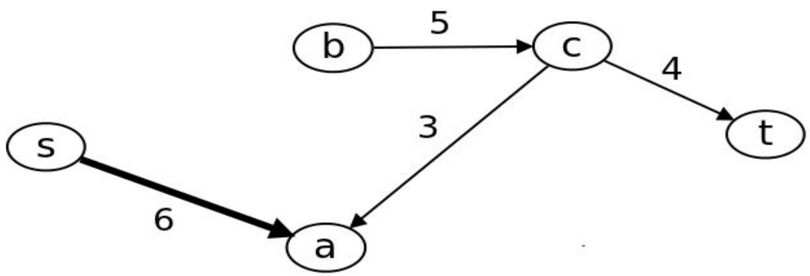


Rede de fluxo

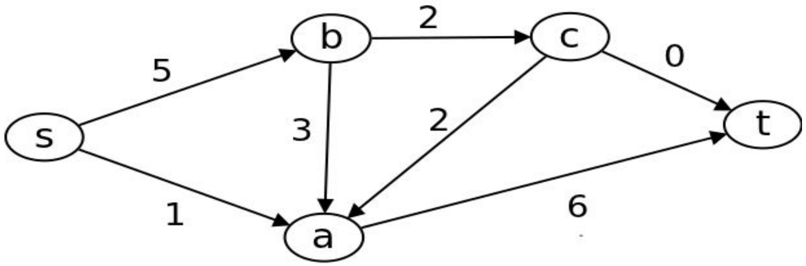
Fluxo em Redes

- Exemplo:





Rede residual

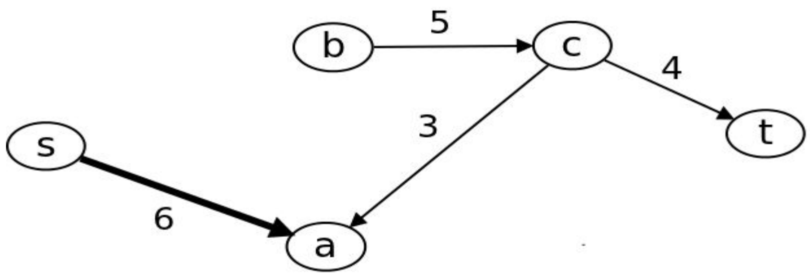


Rede de fluxo

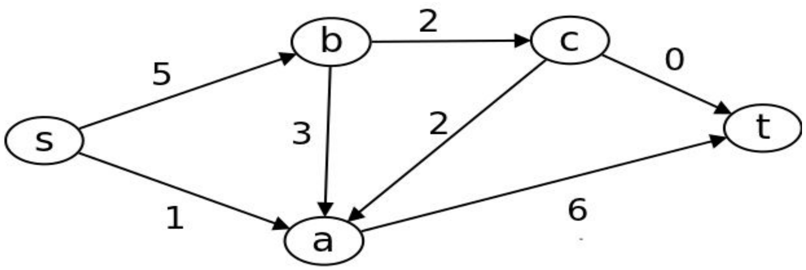
- Não há mais caminho aumentante na rede residual, mas o fluxo obtido **não é máximo**! Há um fluxo viável com valor 10!

Salles Viana Gomes de Magalhães
Departamento de Informática - UFV

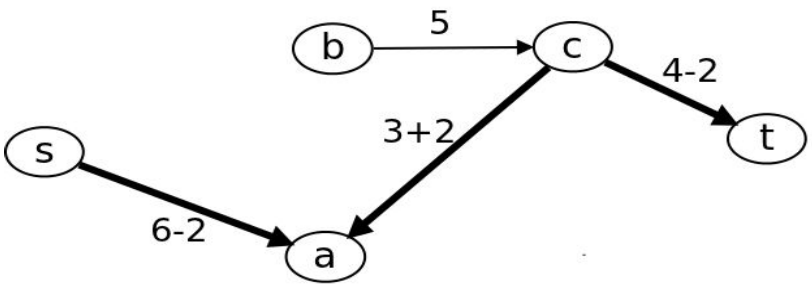
Fluxo em Redes



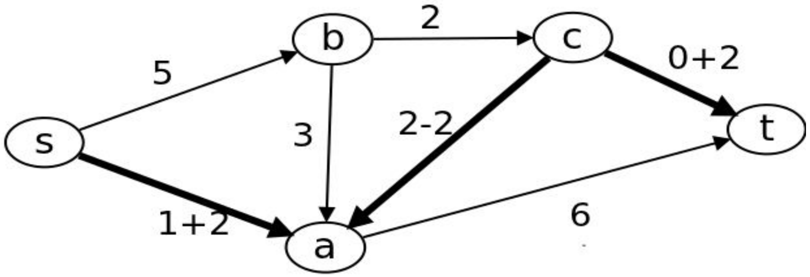
Rede residual



Rede de fluxo



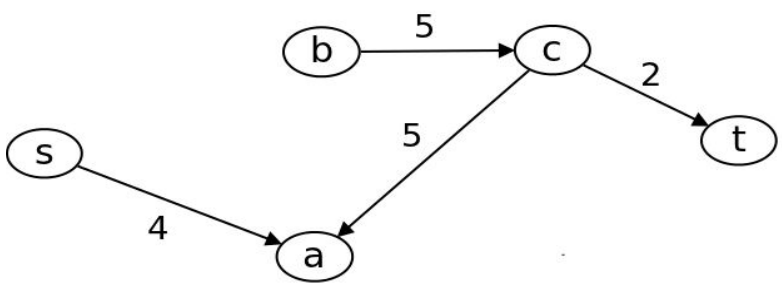
Rede residual



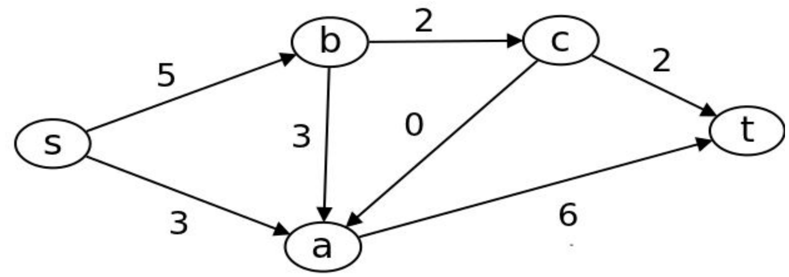
Rede de fluxo

“Desviando” o fluxo...

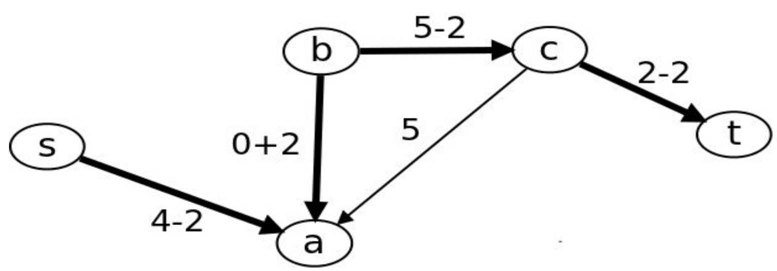
Fluxo em Redes



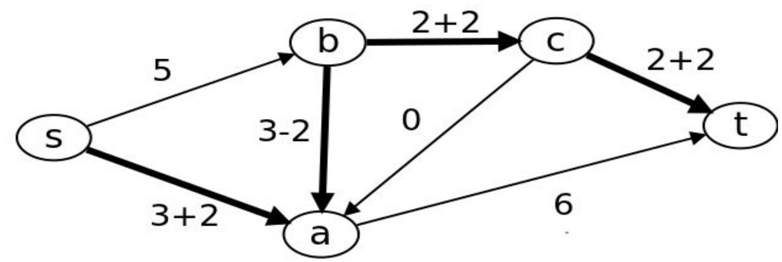
Rede residual



Rede de fluxo

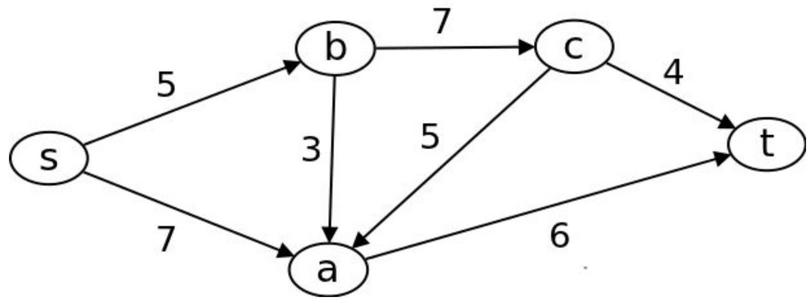


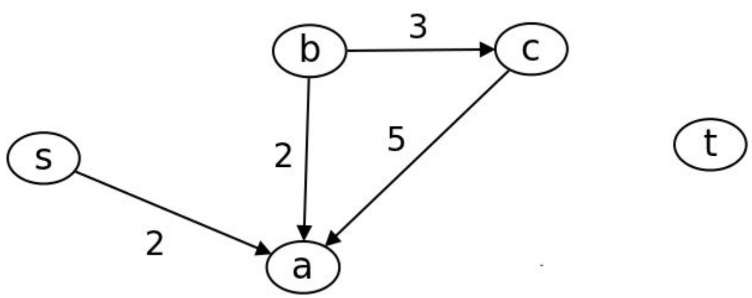
Rede residual



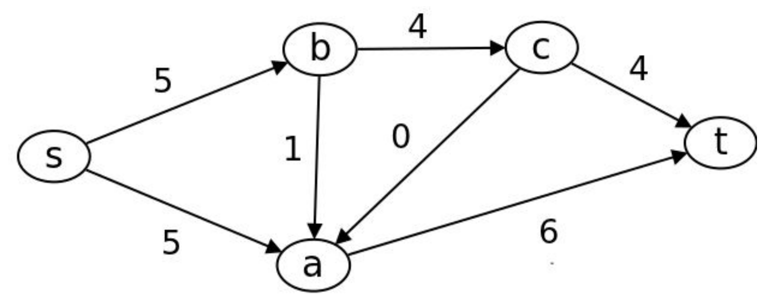
Rede de fluxo

Fluxo em Redes





Rede residual



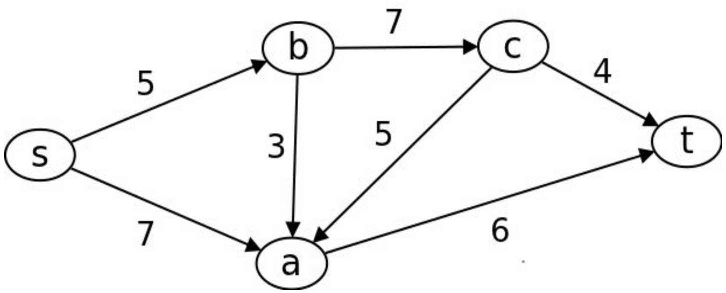
Rede de fluxo

Fluxo em Redes

- Como simular esse "desvio de fluxo"? Uma ideia consiste em criar arcos na rede residual com direção contrária à do fluxo. A capacidade desses arcos é igual à do fluxo.
- Algoritmo de Ford-Fulkerson (baseado em caminhos aumentantes):
- Inicialmente o grafo residual Gr é igual a G
- Para cada aresta *uv* em E(G):
  - fluxo(u,v) ← fluxo(v,u) ← 0
- Enquanto existir um caminho aumentante *P* em *Gr*
  - *f* ← capacidade residual de P
  - Para cada aresta uv em P:
    - fluxo(*u*,*v*) += f //fluxo(u,v) é o peso da aresta (u,v) em *Gf*
    - capacidade(*u*,*v*) -= f //capacidade(u,v) é o peso da aresta (u,v) em Gr
    - capacidade(*v*,*u*) += f
- Para cada aresta <u>*uv*</u> em *Gf*
  - Se fluxo(u,v) > fluxo(v,u)
    - fluxo(u,v) ← fluxo(u,v) – fluxo(v,u)
    - fluxo(v,u) ← 0
  - Senão
    - fluxo(v,u) ← fluxo(v,u) – fluxo(u,v)
    - fluxo(u,v) ← 0

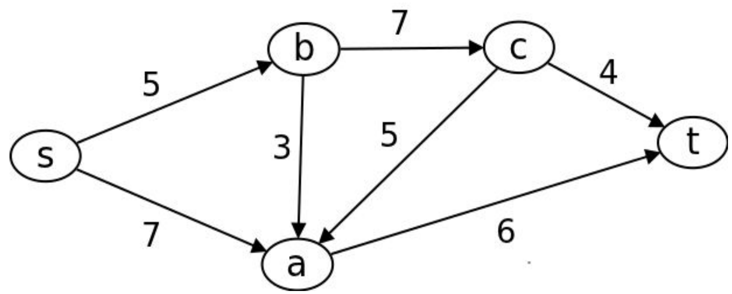
Fluxo em Redes

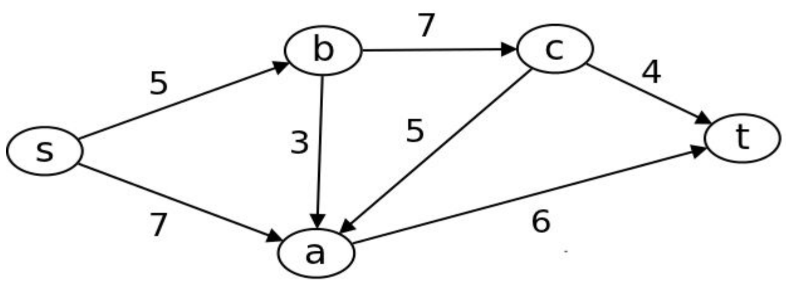
- Exemplo do algoritmo de Ford-Fulkerson:



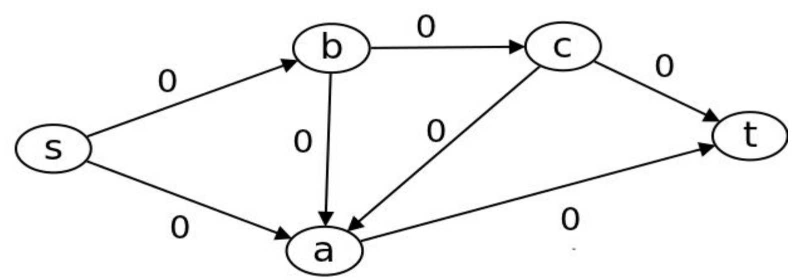
Fluxo em Redes

- Exemplo:

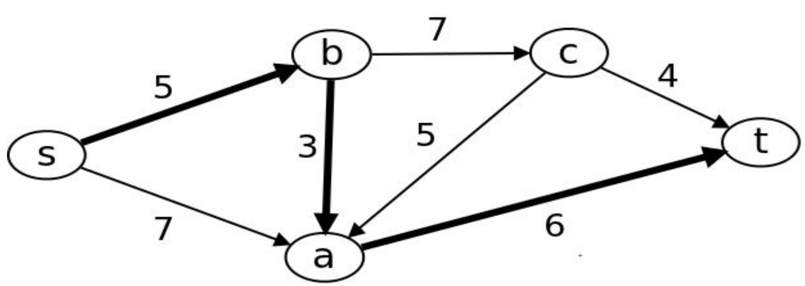




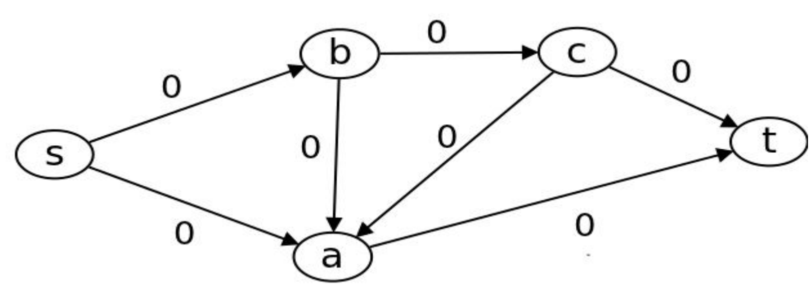
Rede residual



Rede de fluxo



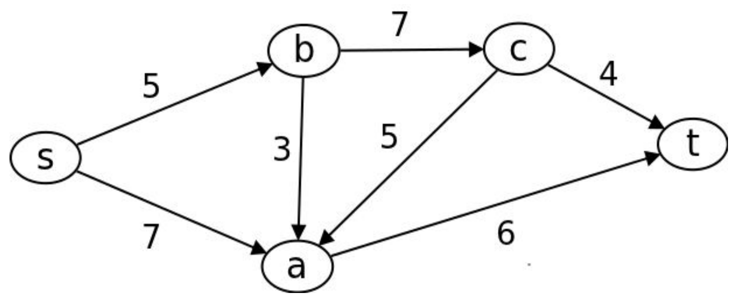
Rede residual

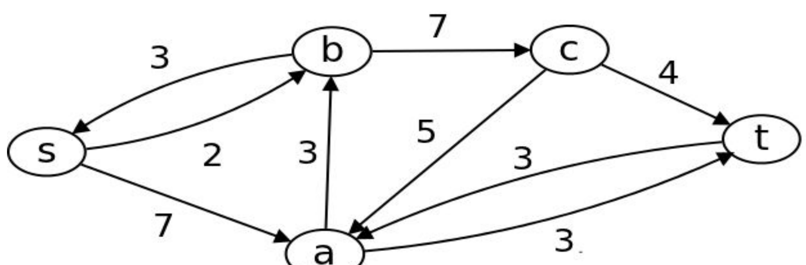


Rede de fluxo

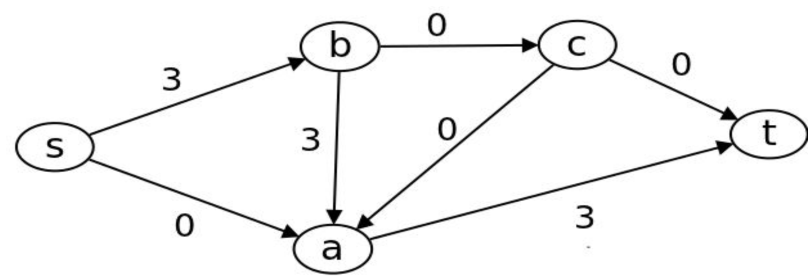
Fluxo em Redes

- Exemplo:

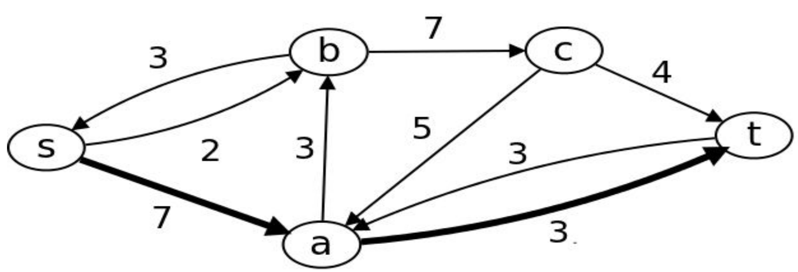




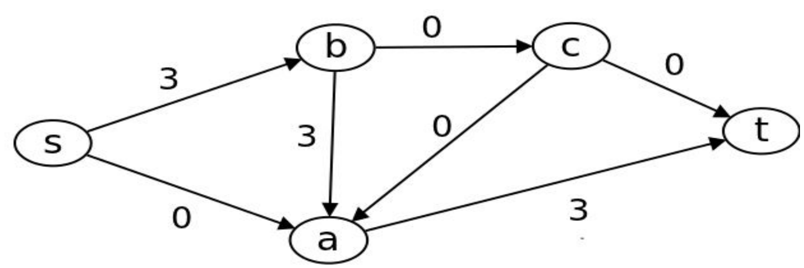
Rede residual



Rede de fluxo



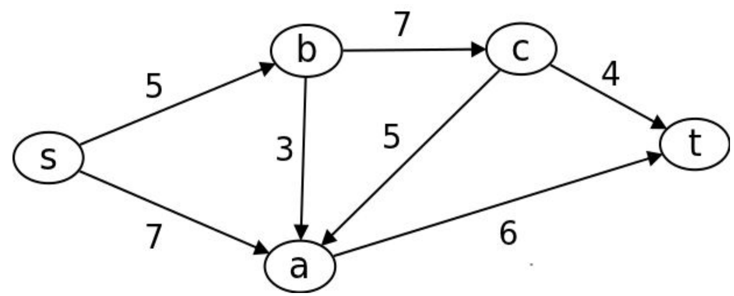
Rede residual

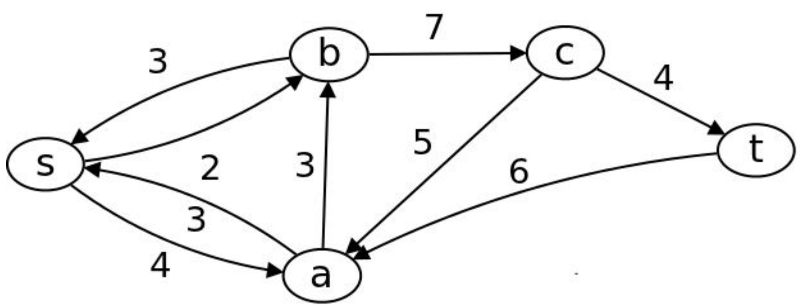


Rede de fluxo

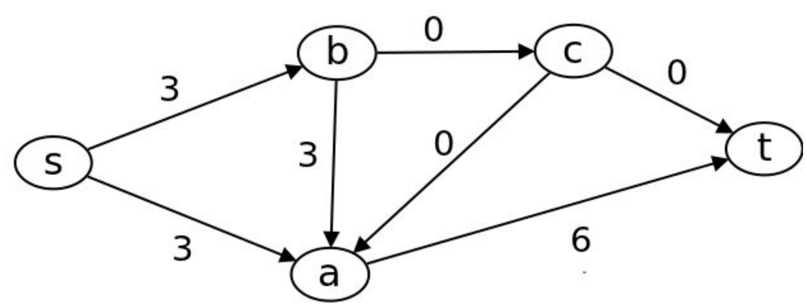
Fluxo em Redes

- Exemplo:

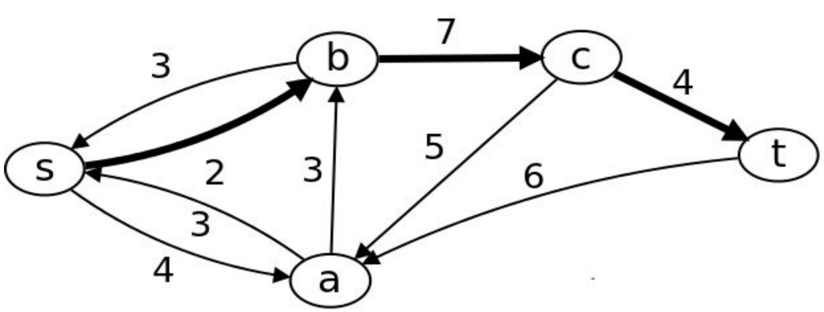




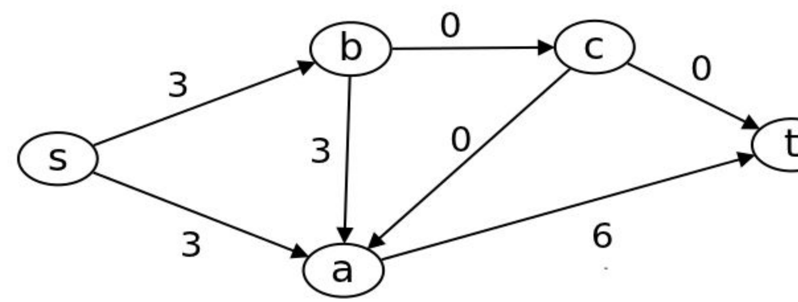
Rede residual



Rede de fluxo



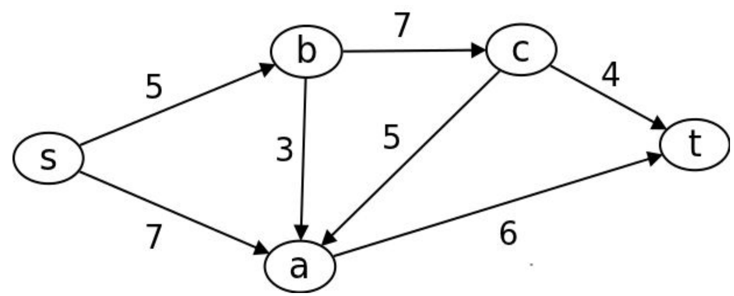
Rede residual

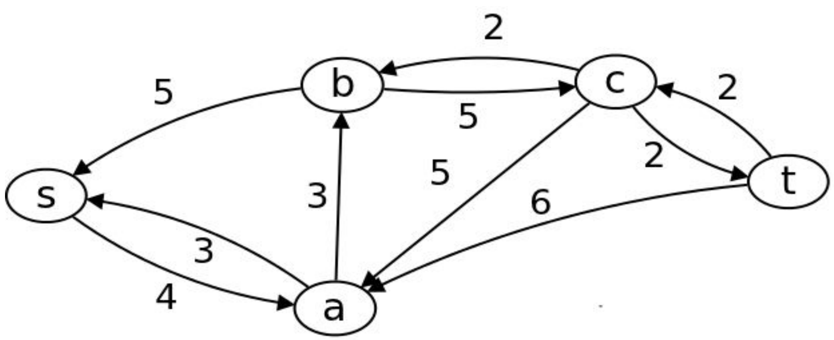


Rede de fluxo

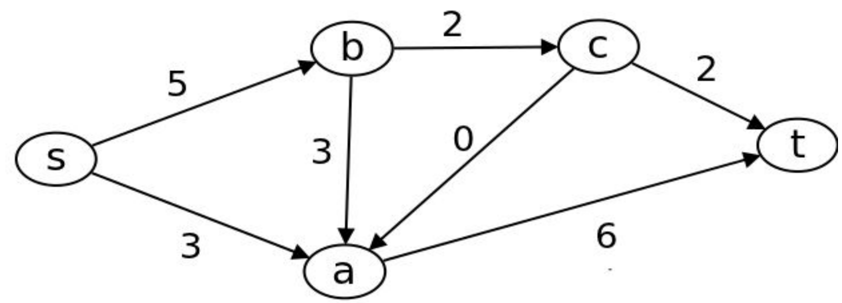
Fluxo em Redes

- Exemplo:

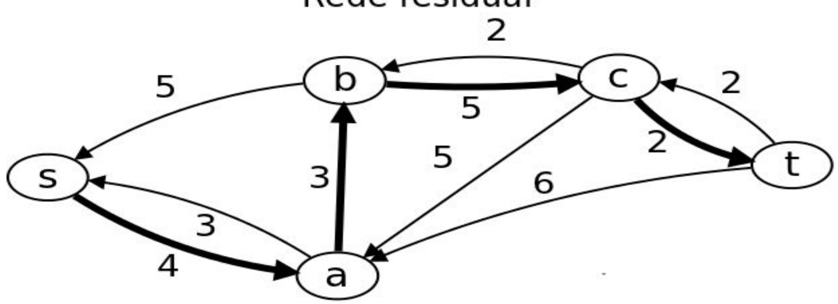




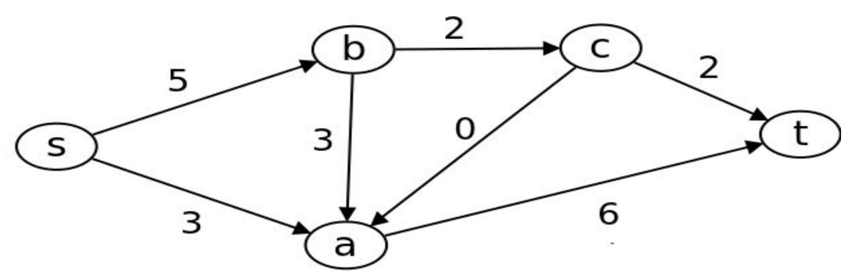
Rede residual



Rede de fluxo



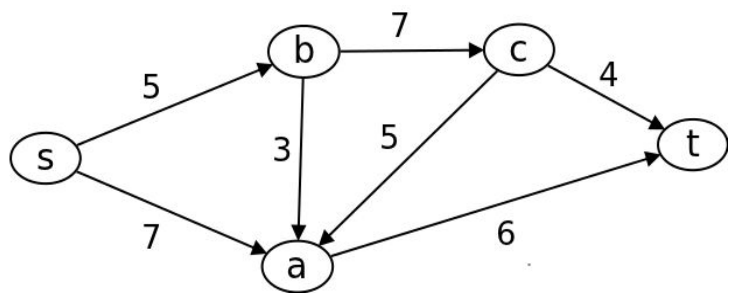
Rede residual

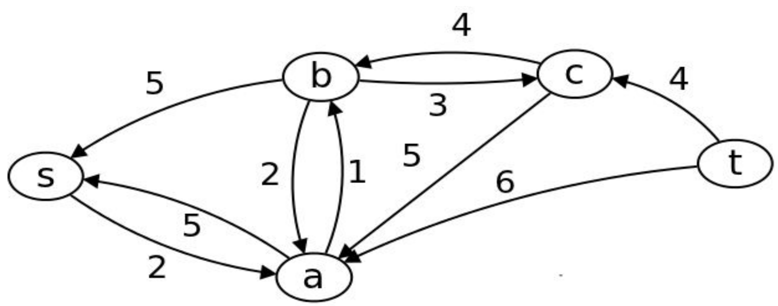


Rede de fluxo

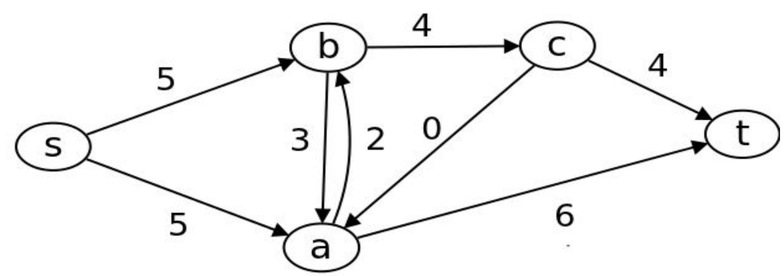
Fluxo em Redes

- Exemplo:

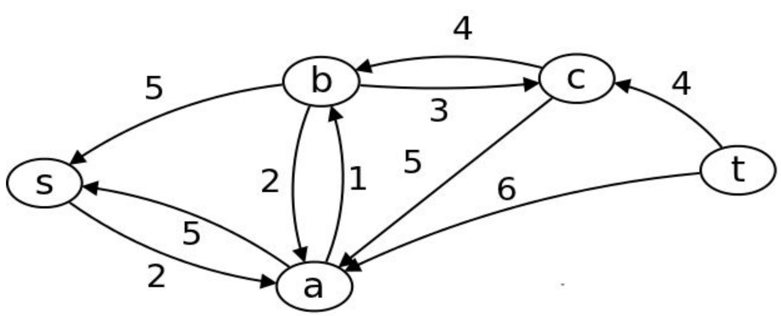




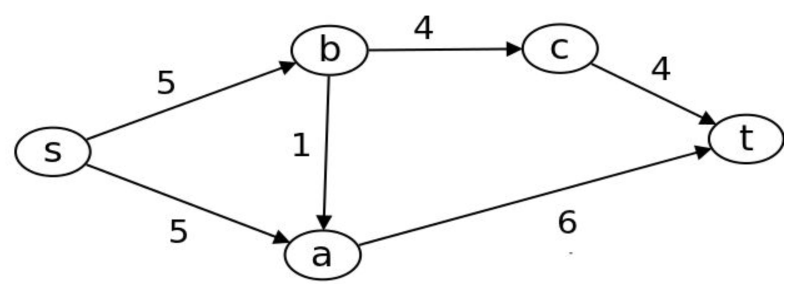
Rede residual



Rede de fluxo



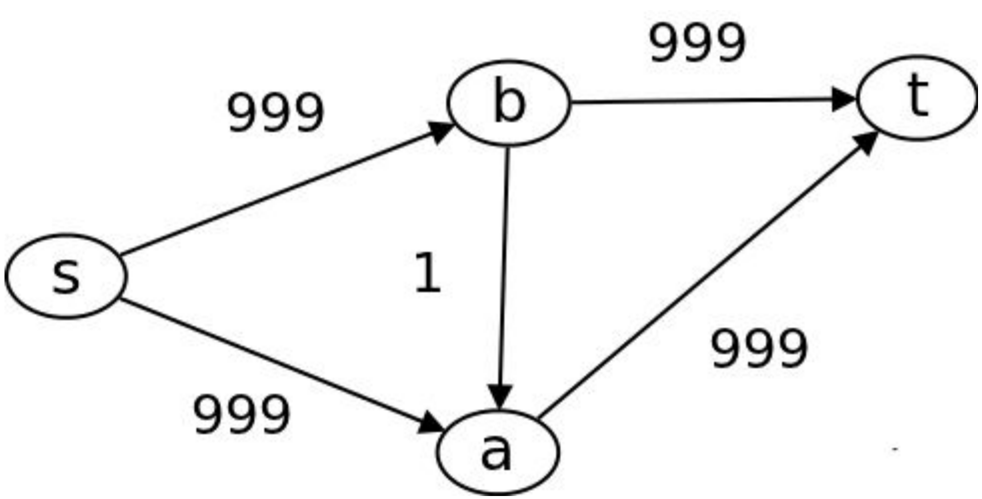
Rede residual



Rede de fluxo

Fluxo em Redes

- O algoritmo de Ford-Fulkerson pode ficar bem lento em algumas situações, tendo complexidade O(|E|f*) no pior caso. Isso pode acontecer devido à estratégia utilizada para se encontrar os caminhos aumentantes.
- Exemplo:

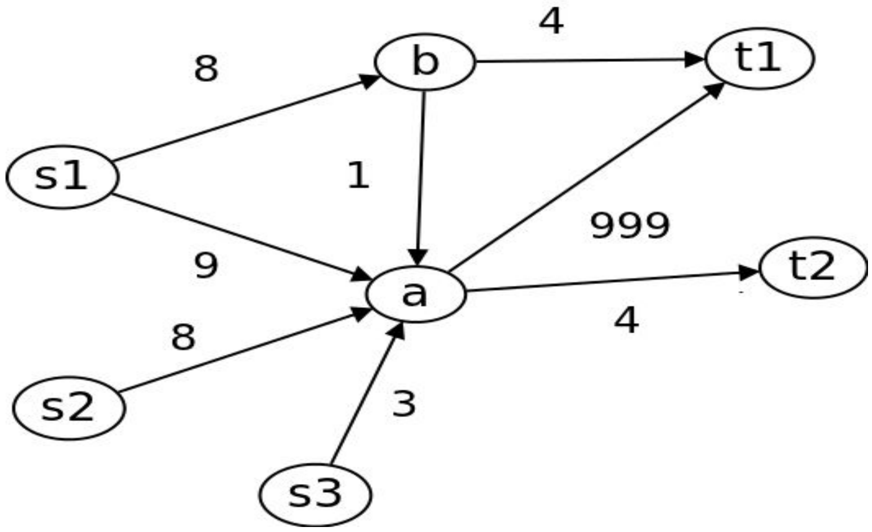


Fluxo em Redes

- Um algoritmo **assintoticamente** mais eficiente para o cálculo do fluxo máximo é o algoritmo de Edmons-Karp. Tal algoritmo é similar ao algoritmo de Ford-Fulkerson, porém utiliza uma **busca em largura** para se encontrar os caminhos aumentantes (de distância mínima…).
- Pode-se provar que a complexidade do algoritmo de Edmons-Karp é $O(VE^2)$.
- Na prática, dependendo do grafo Ford-Fulkerson pode ser mais rápido do que Edmons-Karp.
- Existem outros algoritmos mais eficientes para o cálculo do fluxo máximo em redes (tais algoritmos são mais complexos do que o algoritmo de Ford-Fulkerson e Edmons-Karp): um exemplo é o algoritmo de Relabel-to-Front, cuja complexidade é $O(V^3)$

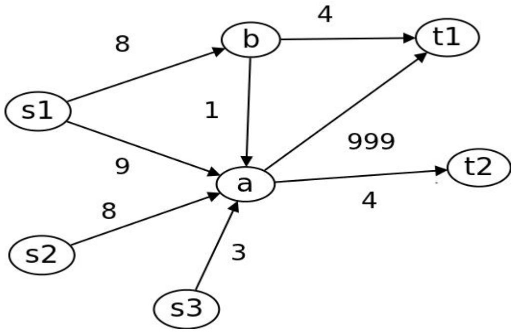
Fluxo em Redes

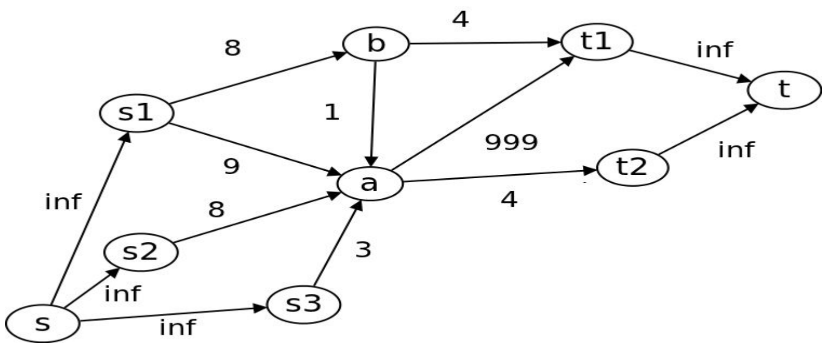
- Outras considerações sobre fluxo em redes:
- Suponha que a rede possua várias fontes e/ou sorvedores: como encontrar o fluxo máximo total no grafo ?



Fluxo em Redes

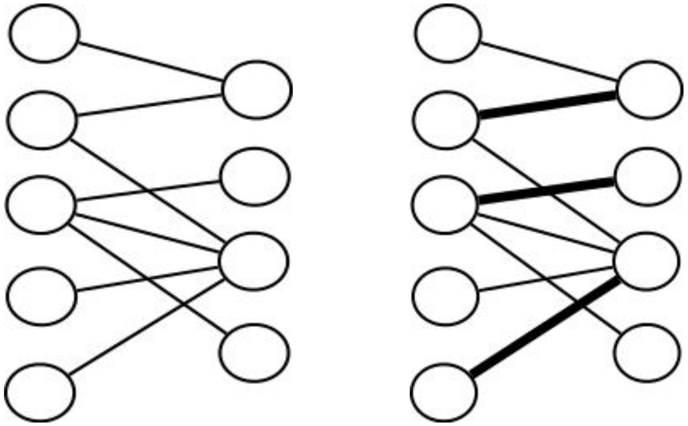
- Outras considerações sobre fluxo em redes:
- Suponha que a rede possua várias fontes e/ou sorvedores: como encontrar o fluxo máximo total no grafo ? Basta criar uma fonte e um sorvedouro “artificial” e transformar as outras fontes/sorvedouros em nodos de transbordo – utilize arcos de capacidade infinita para conectar a fonte artificial às fontes originais e faça o mesmo com o sorvedouro.





Fluxo em Redes

- Outras considerações sobre fluxo em redes:
- Uma característica interessante do problema de fluxo em redes é que outros problemas de grafos podem ser reduzidos ao problema de fluxo máximo.
- Um exemplo clássico é o problema de emparelhamento bipartido máximo: dado um grafo bipartido, encontrar o maior conjunto de arestas de forma que cada vértice do grafo esteja conectado a no máximo um outro vértice.
  - Aplicações:
    - Designar pessoas a tarefas.
    - Designar máquinas a tarefas.
    - Etc.



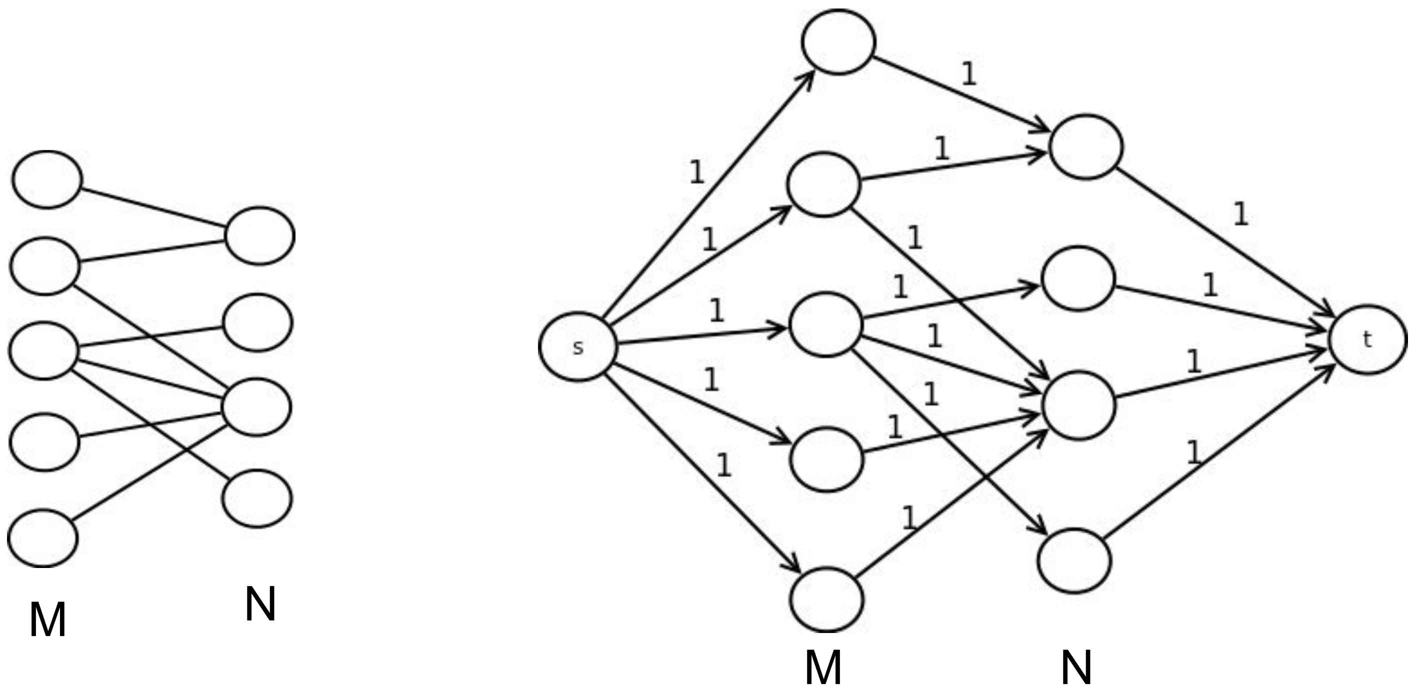
Grafo bipartido G Emparelhamento máximo de G

Fluxo em Redes

- Outras considerações sobre fluxo em redes:
- Como modelar o problema de emparelhamento máximo em um grafo bipartido G utilizando fluxo máximo em redes?
  - Basta criar uma rede contendo uma fonte s , um sumidouro t e o vértices do grafo bipartido.
  - Sejam *M* e *N* as partições de vértices do grafo bipartido.
  - Cria-se um arco saindo de *s* para cada vértice *v* de *M*.
  - Para cada aresta uv de G (onde *u* pertence a M e *v* pertence a N), cria-se um arco uv na rede.
  - Para cada vértice *v* de *N*, cria-se um arco *vt*.
  - Atribui a capacidade 1 a todos os arcos da rede.
  - Determina-se o fluxo máximo na rede *st* criada utilizando um algoritmo que encontre o fluxo máximo onde o fluxo de cada arco é um valor inteiro (**propriedade IMPORTANTE: pode-se provar que, se a capacidade dos arcos for inteira, então o fluxo obtido pelo método de Ford-Fulkerson/Edmons-Karp é inteiro**).

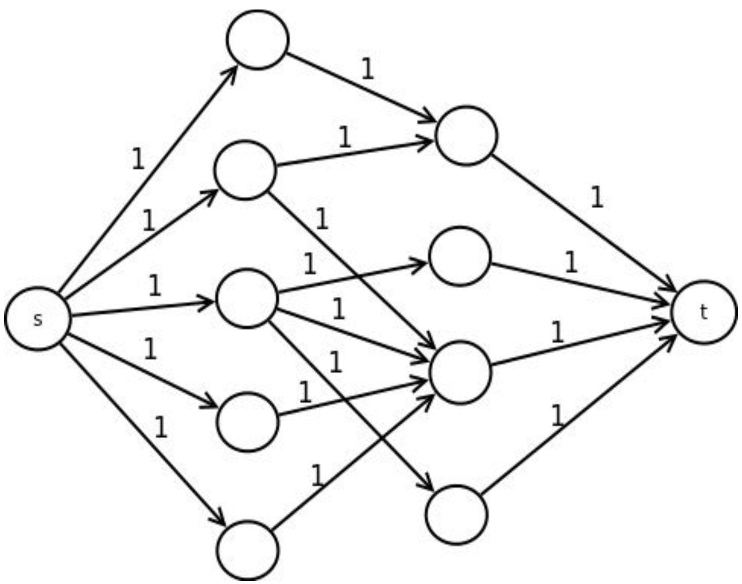
Fluxo em Redes

- Outras considerações sobre fluxo em redes:
- Após o cálculo do fluxo, o emparelhamento máximo corresponderá aos arcos com fluxo não nulo que ligam vértices de *M* a vértices de *N*.

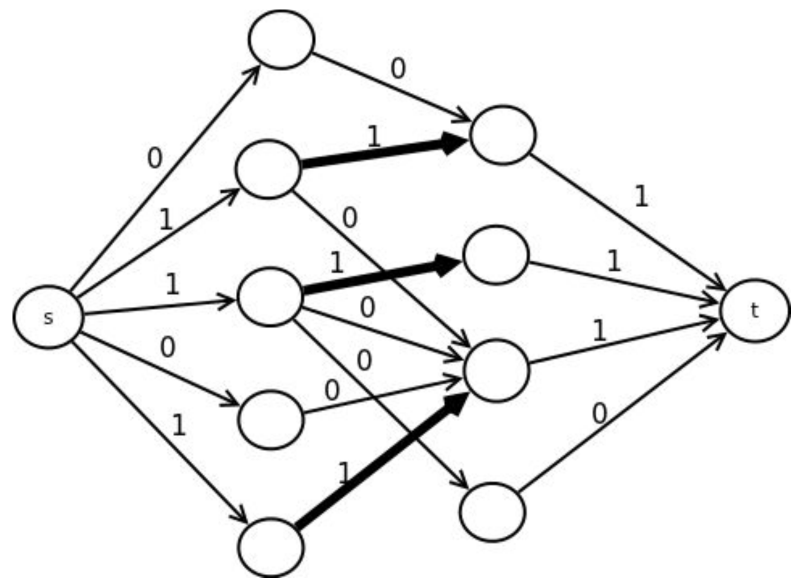


Fluxo em Redes

- Outras considerações sobre fluxo em redes:
- Após o cálculo do fluxo, o emparelhamento máximo corresponderá aos arcos com fluxo não nulo que ligam vértices de *M* a vértices de *N*.



Rede com as capacidades



Grafo de fluxo (máximo)

Fluxo em Redes

- Exercício: dado um conjunto de pessoas onde cada pessoa é capaz de realizar algumas determinadas tarefas, atribuir pessoas a tarefas de modo que:
  - Cada tarefa seja realizada por no máximo uma pessoa.
  - Cada pessoa pode realizar no máximo k tarefas.
  - Queremos maximizar o número de tarefas realizadas.
- Como você modelaria o problema acima?
- Exemplo de entrada:

Pessoa	Consegue realizar..		
P1	T1	T2	T3
P2	T2	T3	T4
P3	T3		

Fluxo em Redes

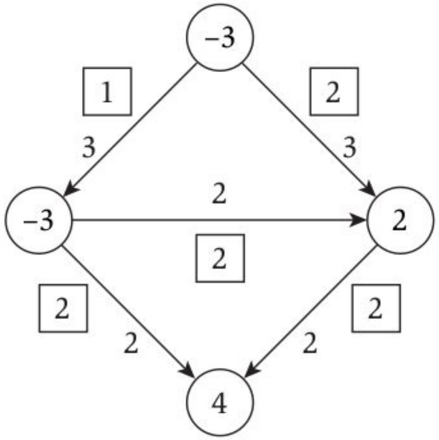
- Exercício: dado um grafo direcionado onde vértices representam cidades e arestas estradas (todas estradas possuem direção), encontrar o número de caminhos ligando duas cidades de modo que dois caminhos não possuam estradas em comum (podem possuir cidades em comum).

Fluxo em Redes

- Exercício: dado um conjunto de cidades e estradas entre elas (todas estradas possuem direção), encontrar o número de caminhos ligando duas cidades de modo que dois caminhos não possuam estradas em comum (podem possuir cidades em comum).
  - Solução:
    - Criar uma rede onde estradas são arestas de capacidade 1, cidades são vértices.
    - O número de caminhos é igual ao fluxo máximo entre as duas cidades.
      - Isso funciona porque, como as capacidades são inteiras, existirá um fluxo máximo que utiliza apenas valores inteiros nas arestas.

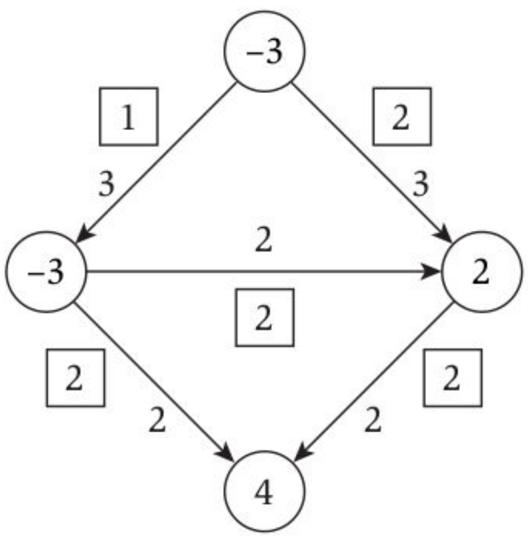
Fluxo em Redes

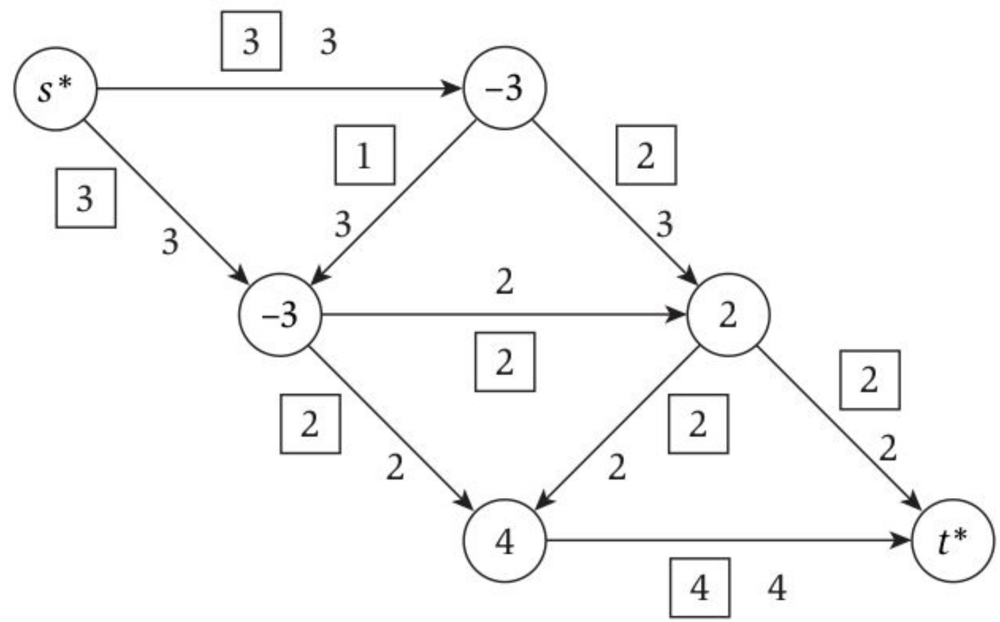
- Outro problema: circulação com demandas.
- E se tivermos várias fontes e vários sorvedouros, onde o objetivo seria não maximizar o fluxo, mas sim escoar a oferta das fontes e atender a demanda dos sorvedouros?
  - Exemplo de aplicação: fábricas produzem produtos, cidades possuem demanda fixa por produtos e as estradas possuem capacidade máxima de roteamento.
- Exemplo de entrada (fonte: Algorithm Design, Kleinberg e Tardos):
  - Rótulos nas arestas: capacidade
  - Rótulos em quadrados: fluxo na solução
  - Rótulos nos vértices: demanda (negativa → fábrica)
  - (note que qualquer quantidade de fluxo pode passar por um vértice)
- Como modelar isso?



Fluxo em Redes

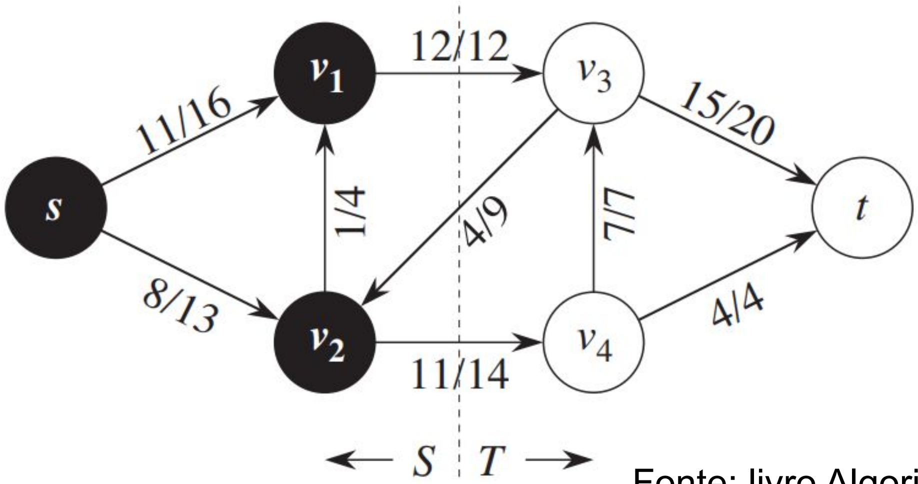
- Como modelar isso? Basta reduzir ao problema do fluxo máximo em redes.
  - Criar fonte artificial s* e sumidouro artificial t*
  - Se uma fábrica tem oferta -x ⟶ criar uma aresta de capacidade x de s* para o vértice da fábrica.
  - Se uma cidade possui demanda x ⟶ criar uma aresta de capacidade x do vértice da cidade para t*.
  - O fluxo obtido na rede é a solução para a circulação de fluxo original (veja a figura à direita).





Corte minimo

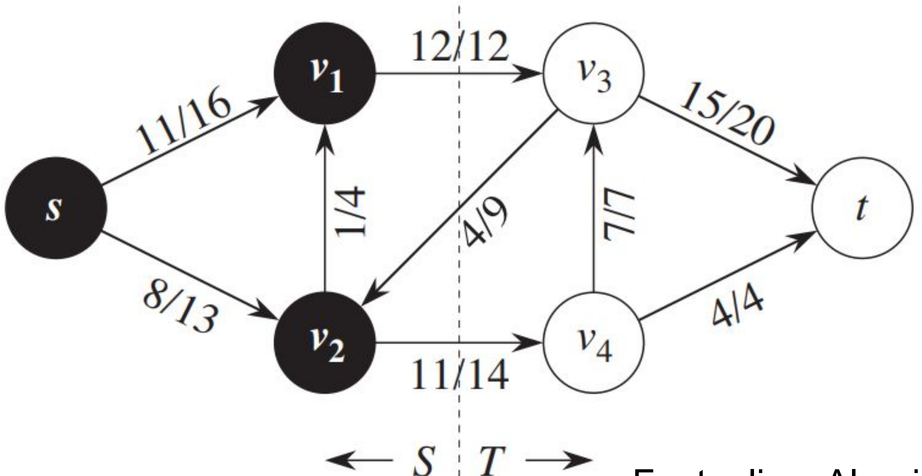
- Um corte (S,T) de um fluxo em rede G divide os vértices de G em dois conjuntos disjuntos S e T, onde s está em S e t está em T.
- A capacidade c(S,T) de um corte (S,T) é a soma da capacidade das arestas saindo de S.
- Se f é um fluxo em G, o fluxo líquido f(S,T) é a diferença entre o fluxo saindo de S e o fluxo chegando a S.
- Exemplo abaixo (arestas apresentam o fluxo/capacidade): S = {s,v1,v2} e T={v3,v4,t}. f(S,T) = 12+11-4 = 19 e c(S,T) = 12+14 = 26 (saltar para slide 44)



Fonte: livro Algoritmos, CLRS

Corte minimo

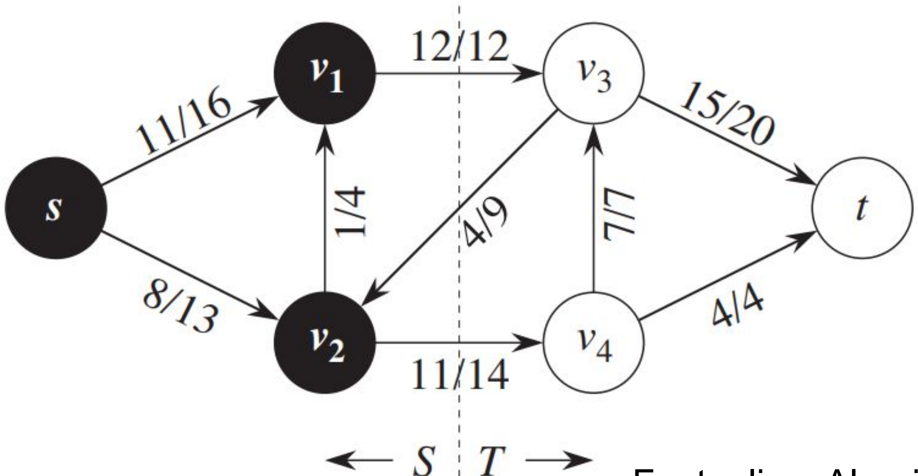
- O fluxo líquido em qualquer corte (do mesmo fluxo) é sempre o mesmo.
- Assim, o fluxo líquido sempre é igual ao fluxo s-t. Por que?



Fonte: livro Algoritmos, CLRS

Corte minimo

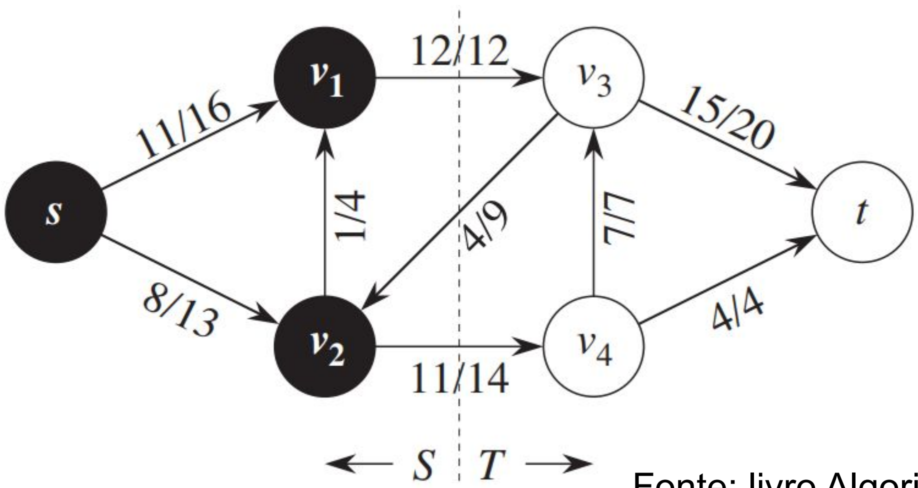
- O valor de qualquer fluxo f em uma rede G é limitado pela capacidade de qualquer corte (S,T) de G.
- Exemplo:
  - O corte abaixo tem capacidade 26 (soma das arestas saindo de S).
  - Como o fluxo f é igual ao fluxo f(S,T), se ele fosse maior que 26 a capacidade de alguma aresta teria que ser ultrapassada (o fluxo não seria viável).



Fonte: livro Algoritmos, CLRS

Corte minimo

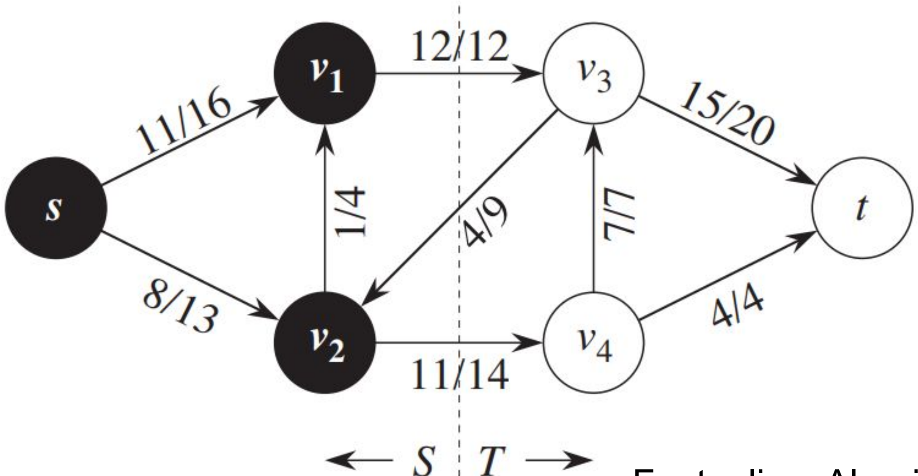
- O valor de qualquer fluxo f em uma rede G é limitado pela capacidade de qualquer corte (S,T) de G.
- Consequência: o fluxo máximo em uma rede G é menor ou igual à capacidade mínima (conhecido como *corte mínimo*) de um corte em G.
- No grafo abaixo, qual o limite para o fluxo máximo (devido ao corte mínimo)?



Fonte: livro Algoritmos, CLRS

Corte minimo

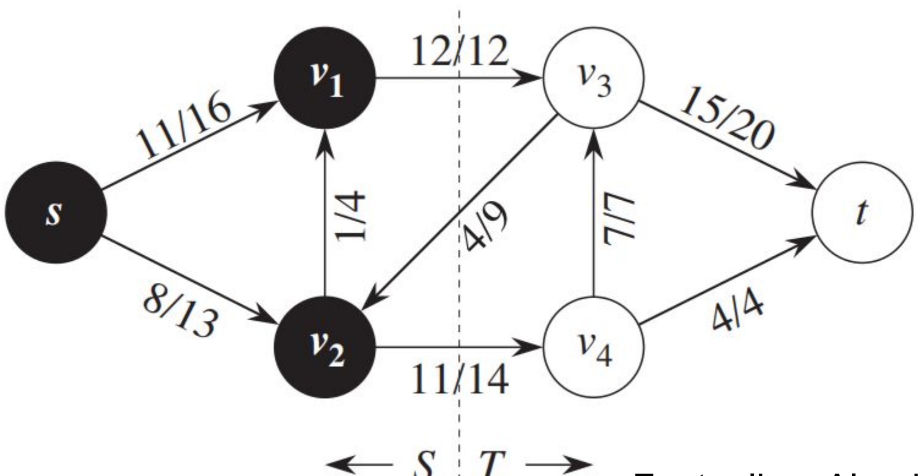
- Mais ainda: o fluxo máximo em uma rede é igual à capacidade do corte mínimo! (23 no grafo abaixo)
- Teorema do fluxo máximo e corte mínimo (Teorema 26.7 do livro CLRS): Se f é um fluxo em um fluxo em rede G, então as 3 condições abaixo são equivalentes:
  - f é um fluxo máximo em G
  - A rede residual de f não contém nenhum caminho aumentante
  - |f| (valor do fluxo) = c(S,T) para algum corte (S,T) de G (tal corte será mínimo).



Fonte: livro Algoritmos, CLRS

Corte minimo

- Assim o problema de encontrar o corte mínimo em um grafo está diretamente ligado ao problema do fluxo máximo.
- Mais especificamente, o problema do corte mínimo é chamado de dual do problema do fluxo máximo (primal).



Fonte: livro Algoritmos, CLRS

Fluxo máximo de custo minimo

- Há algumas variações do problema de fluxo em redes.
- Por exemplo, no problema do fluxo de custo mínimo, o objetivo é enviar uma determinada quantidade de fluxo (ou o máximo de fluxo) pela rede tendo o menor custo possível (além da capacidade, cada aresta possui um custo, que é multiplicado pela quantidade de fluxo que passa por ela).
    - Se o objetivo for enviar uma unidade de fluxo, o problema se reduz ao de caminho minimo entre s e t.
    - Se os custos forem zero, o problema (no caso do fluxo máximo de custo mínimo) se reduz ao de fluxo máximo em redes.
    - Tipicamente resolvido com Edmons-Karp + Dijkstra
- Outro problema interessante: circulação de fluxo, onde cada aresta possui também um limitante inferior para o fluxo ("na aresta x deve passar no mínimo y unidades de fluxo").

Fluxo em Redes

- Desafio: problema "Gasolina" da regional da Maratona de Programação 2018.
- https://www.urionlinejudge.com.br/judge/pt/problems/view/2882

"Terminada a greve dos caminhoneiros, você e os demais especialistas em logística da Nlogônia agora têm a tarefa de planejar o reabastecimento dos postos da cidade. Para isso, foram coletadas informações sobre os estoques das R refinarias e sobre as demandas dos P postos de gasolina. Além disso, há restrições contratuais que fazem com que algumas refinarias não possam atender alguns postos; quando uma refinaria pode fornecer a um posto, sabe-se o menor tempo de percurso para transportar o combustível de um lugar ao outro.

A tarefa dos especialistas é minimizar o tempo de abastecimento de todos os postos, satisfazendo completamente suas demandas. As refinarias têm uma quantidade suficientemente grande de caminhões, de modo que é possível supor que cada caminhão precisará fazer no máximo uma viagem, de uma refinaria para um posto de gasolina. A capacidade de cada caminhão é maior do que a demanda de qualquer posto, mas pode ser necessário usar mais de uma refinaria para atender a demanda de um posto. Seu programa deve encontrar o tempo mínimo no qual é possível abastecer totalmente todos os postos, respeitando os estoques das refinarias."

Fluxo em Redes

- Desafio: problema “Gasolina” da regional da Maratona de Programação 2018.
- https://www.urionlinejudge.com.br/judge/pt/problems/view/2882
- Observacao:
  - Para resolver esse problema foi utilizado o algoritmo de fluxo máximo de Dinic, que tem complexidade $O(V^2E)$ (Ford-Fulkerson e Edmond-Karp não sao eficientes o suficiente para resolve-lo)
  - A implementação do algoritmo de Dinic foi extraída de: https://www.geeksforgeeks.org/dinics-algorithm-maximum-flow/