Teoria dos Grafos
Aula 9

Aula passada

- Caminho mais curto em grafos
- Algoritmo de Bellman-Ford
- Algoritmo distribuído

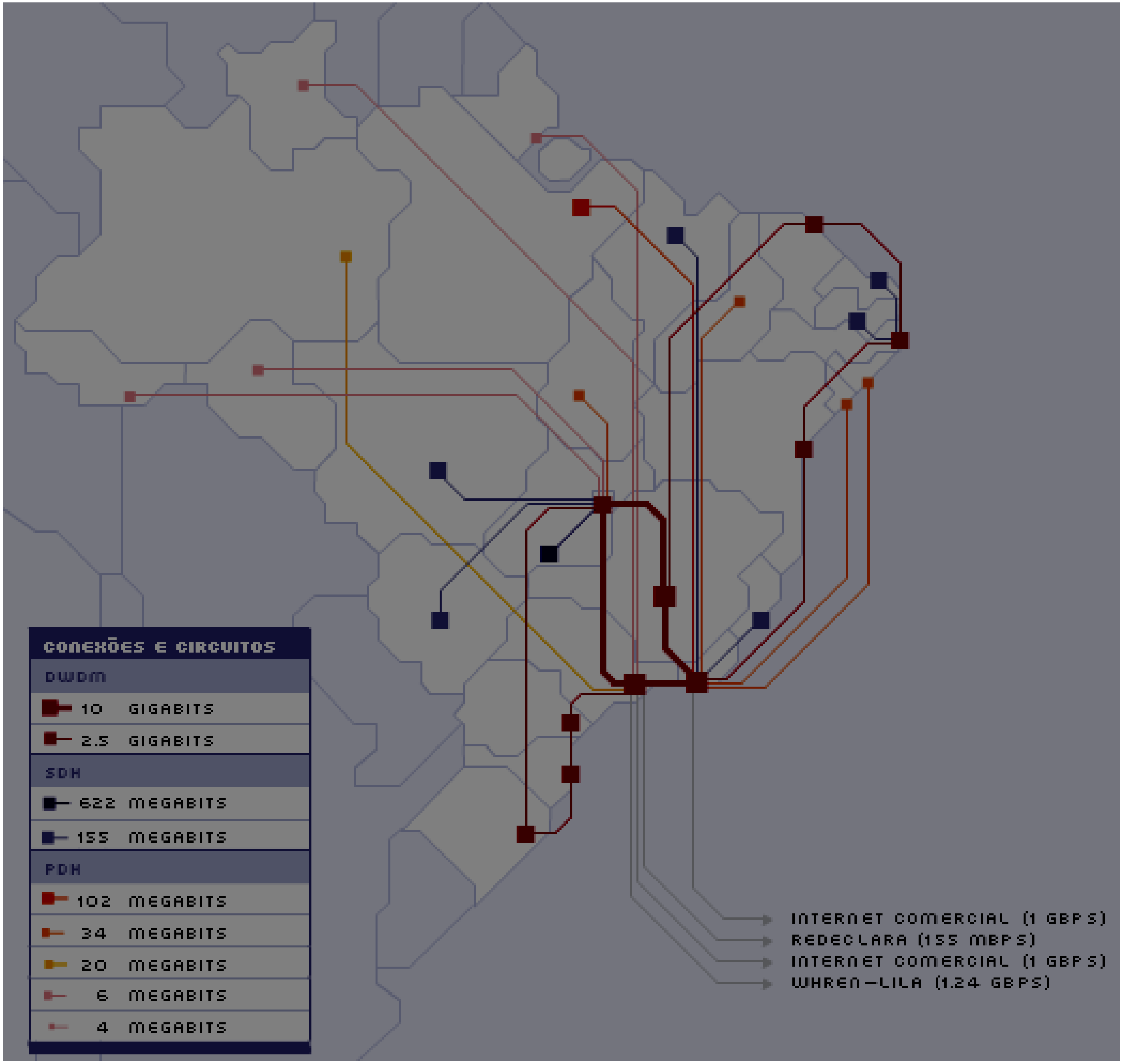
Aula de hoje

- Redes de fluxo
- Problema do fluxo máximo/corte mínimo
- Algoritmo de Ford-Fulkerson
- Aplicações

Malha Rodoviária

- Mapa das estradas brasileiras
  - capacidade das estradas (“carros por hora”)
- Problema: escoamento da produção nacional

Backbone da RNP



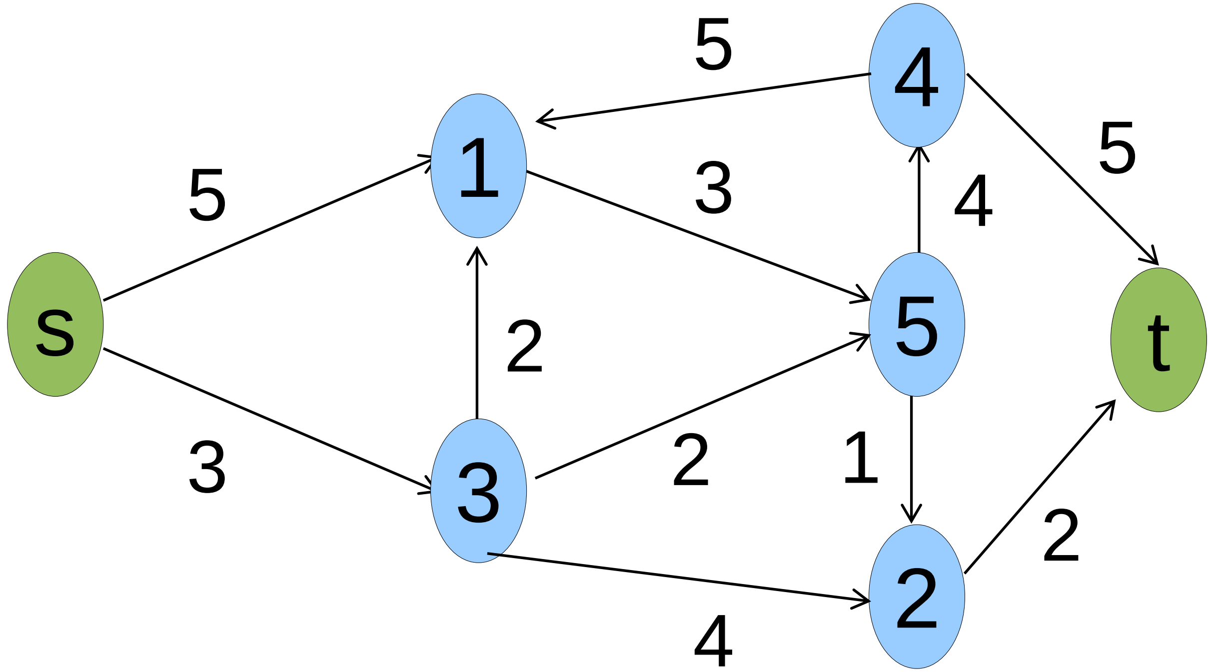
- RNP: Rede Nacional de Pesquisa em Ensino
  - ligação entre instutições nacionais
  - Capacidade dos enlaces (“bits por segundo”)

Redes de Fluxos

- Grafo direcionado
- Arestas possuem “capacidade”
  - quantidade de fluxo máximo que pode passar pela aresta
- 3 tipos de vértices
  - Origem, onde fluxo entra
  - Destino, onde fluxo sai
  - Interno, onde fluxo passa
- *Fluxo*: abstração de algo que possa escoar pelo grafo entre origem e destino
  - carros, bits, etc

Origem/Destino Únicos

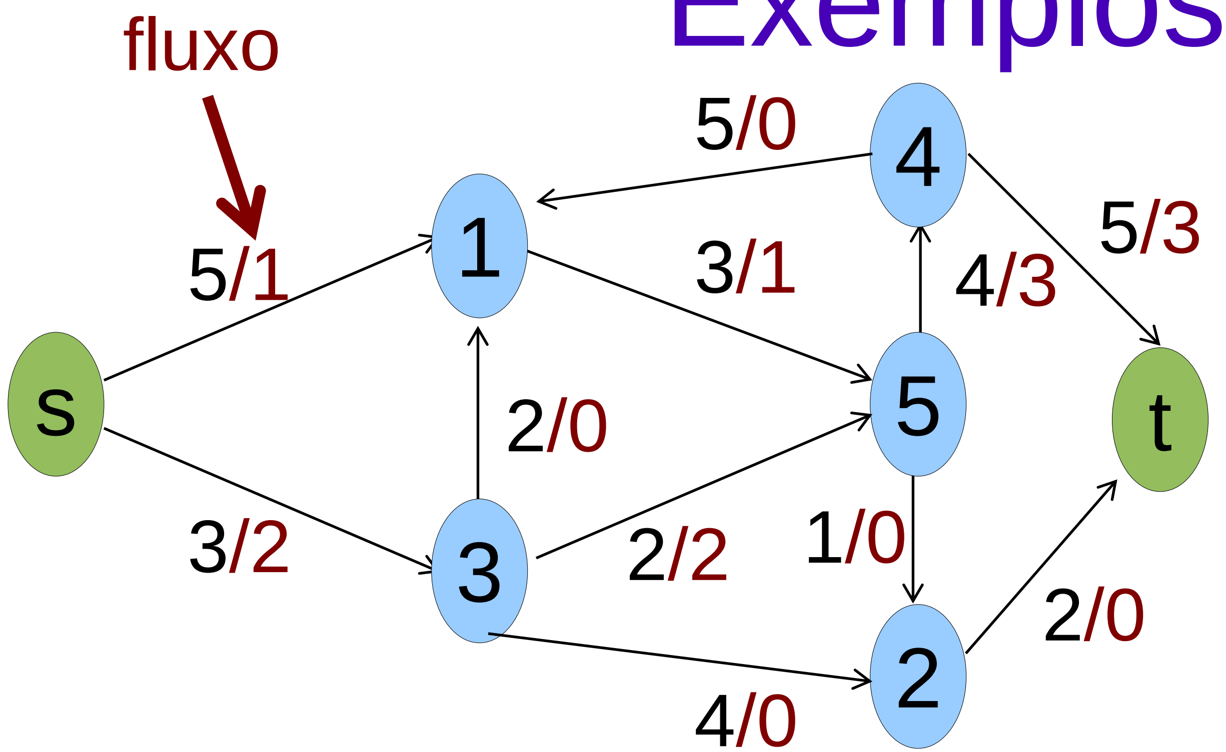
- Rede de fluxos simples
  - 1 vértice origem, 1 vértice destino
  - todos os outros vértices são internos
- Origem não possui arestas de entrada
- Destino não possui arestas de saída



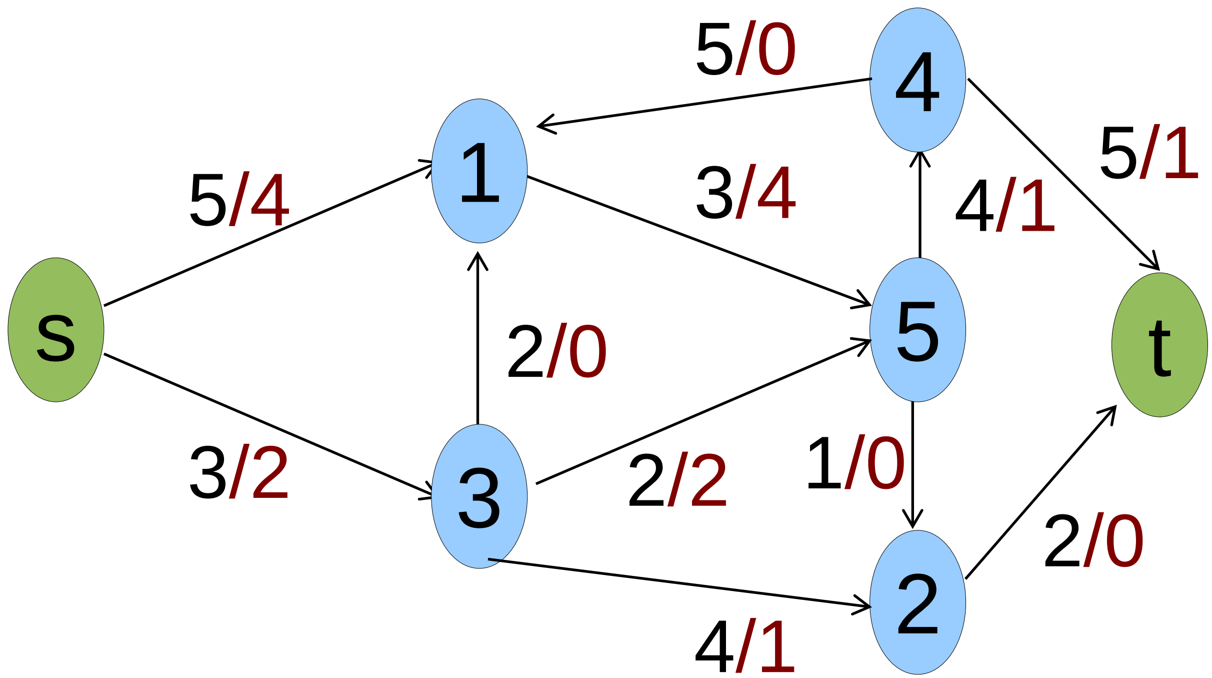
Fluxo na Rede

- Como definir um *fluxo* na rede?
- Determinar o fluxo de cada arestas
- Função $f : E \rightarrow R$, com restrições
- 1) Capacidade
  - Fluxo em uma aresta menor que capacidade
- 2) Conservação
  - Fluxo que entra igual ao fluxo que sai
  - Fluxo que sai da origem igual fluxo que entra no destino
- Valor do fluxo *f*
  - quantidade de fluxo saindo da origem

Exemplos



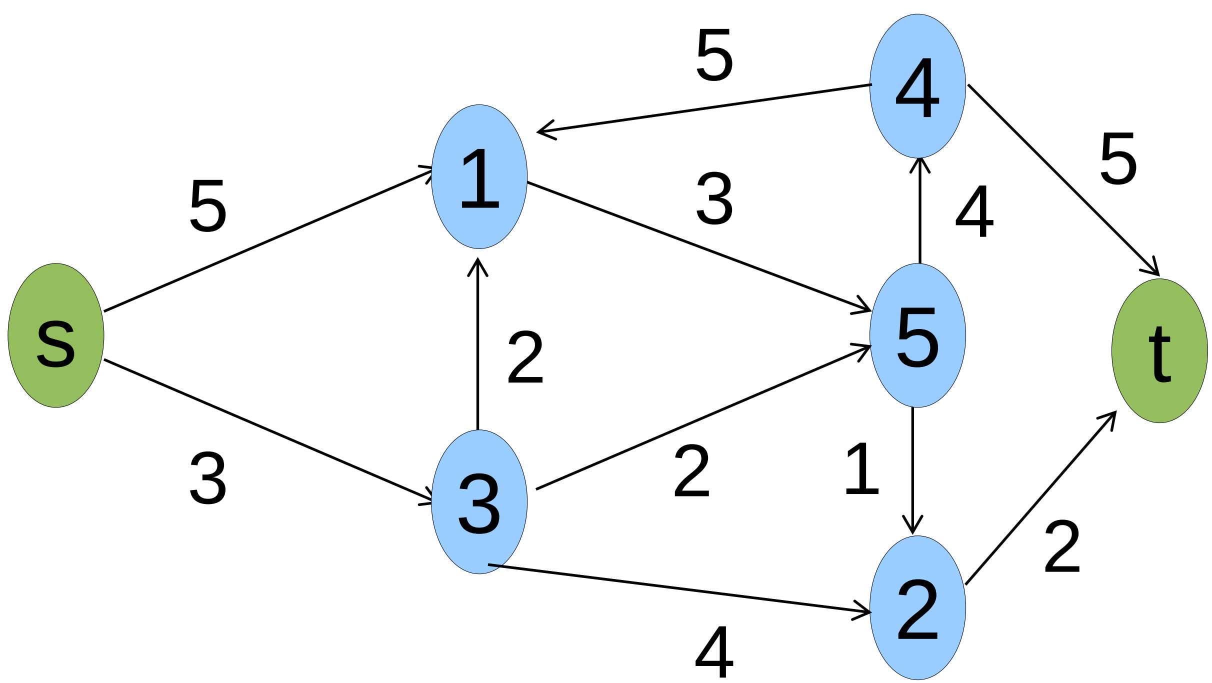
- Fluxo válido?
- Valor?



- Fluxo válido?

Problema do Fluxo Máximo

- Dado G=(V,E) com capacidade nas arestas
  - E dois vértices *s* e *t*
- **Problema**: Determinar fluxo máximo entre *s* e *t*



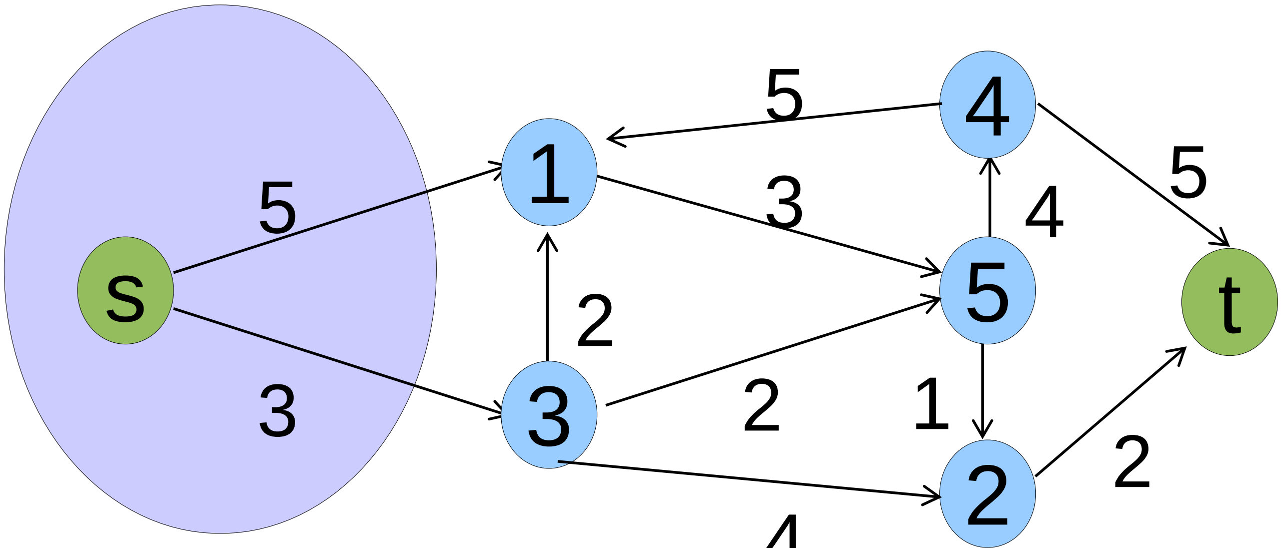
- Fluxo máximo?
- Limitante para fluxo máximo?

Corte em Redes de Fluxo

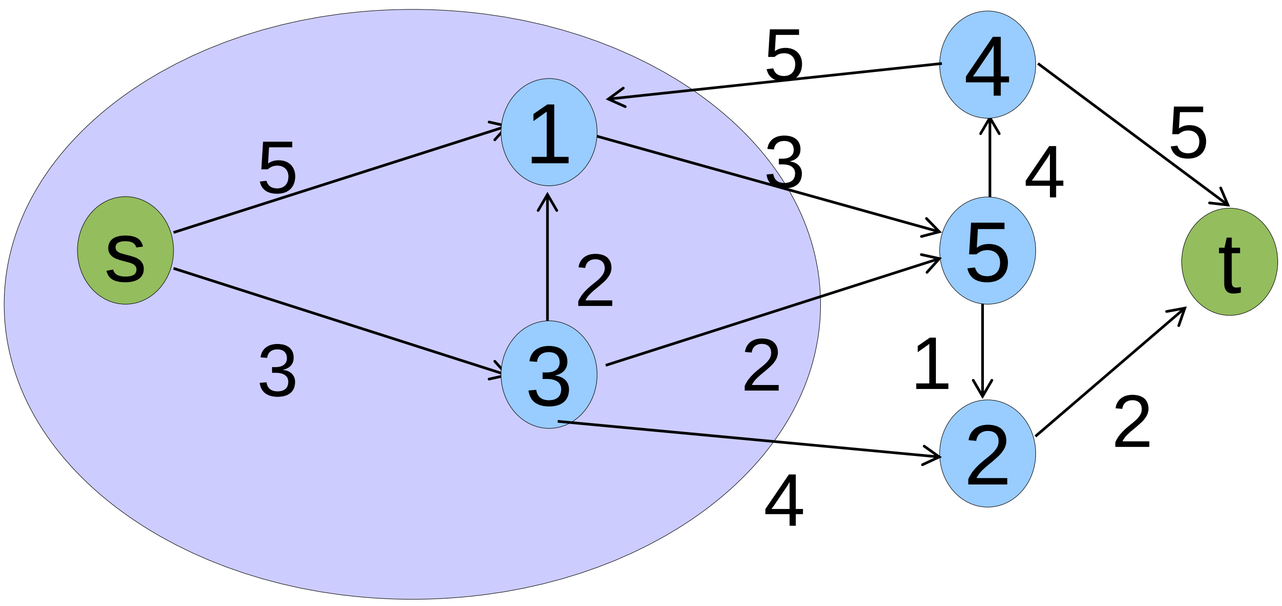
- Generalização da definição de corte
- Corte s-t (A, B) é uma partição dos vértices nos conjuntos A e B tal que *s* está em A e *t* está em B
- Custo do corte (ou capacidade do corte)
  - soma das capacidades da arestas do corte

$$c(A, B) = \sum_{e=(a,b), a \in A, b \in B} c\,(e)$$

Exemplos



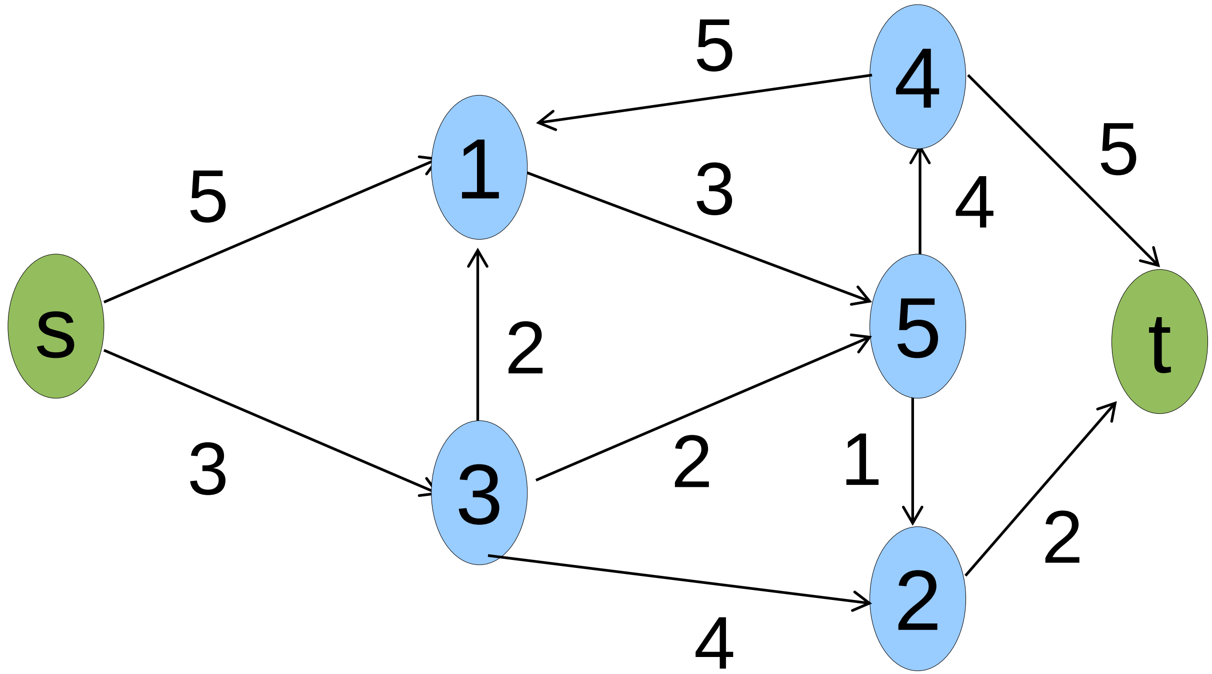
- c(A,B) = ?



- c(A,B) = ?

Problema do Corte Mínimo

- Dado G=(V,E) com capacidade nas arestas
  - E dois vértices *s* e *t*
- **Problema**: Determinar corte s-t de capacidade mínimo?



- Corte mínimo?

Fluxo Máximo e Corte Mínimo

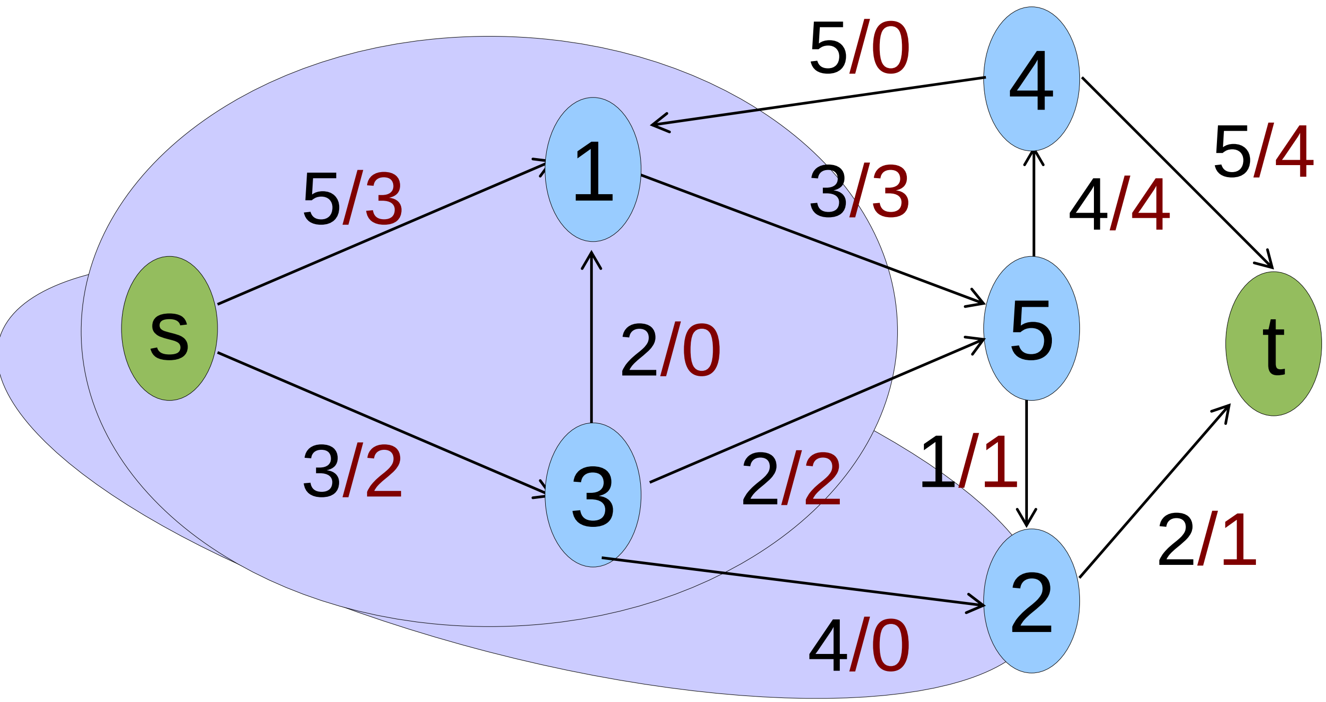
- Problemas duais
  - Muitas aplicações, mesma solução
- Lema da valor do fluxo
  - Seja f um fluxo, e (A, B) um corte s-t
  - Então o fluxo total entre A e B é igual ao fluxo saindo de s

$$\sum_{e saindo de A} f(e) - \sum_{e entrando em A} f(e) = v(f)$$

Valor do fluxo f

Exemplo

- Lema da valor do fluxo



- Correto?

Prova do Lema

- Seja f um fluxo, e (A, B) um corte s-t
- Então o fluxo total entre A e B é igual ao fluxo saindo de s

$$\sum_{e saindo de A} f(e) - \sum_{e entrando em A} f(e) = v(f)$$

Conservação de fluxo: todos os termos são zero menos s e t

- Prova

$$v(f) = \sum_{e saindo de s} f(e)$$

$$v(f) = \sum_{v \in A} \left(\sum_{e saindo de v} f(e) - \sum_{e entrando em v} f(e) \right)$$

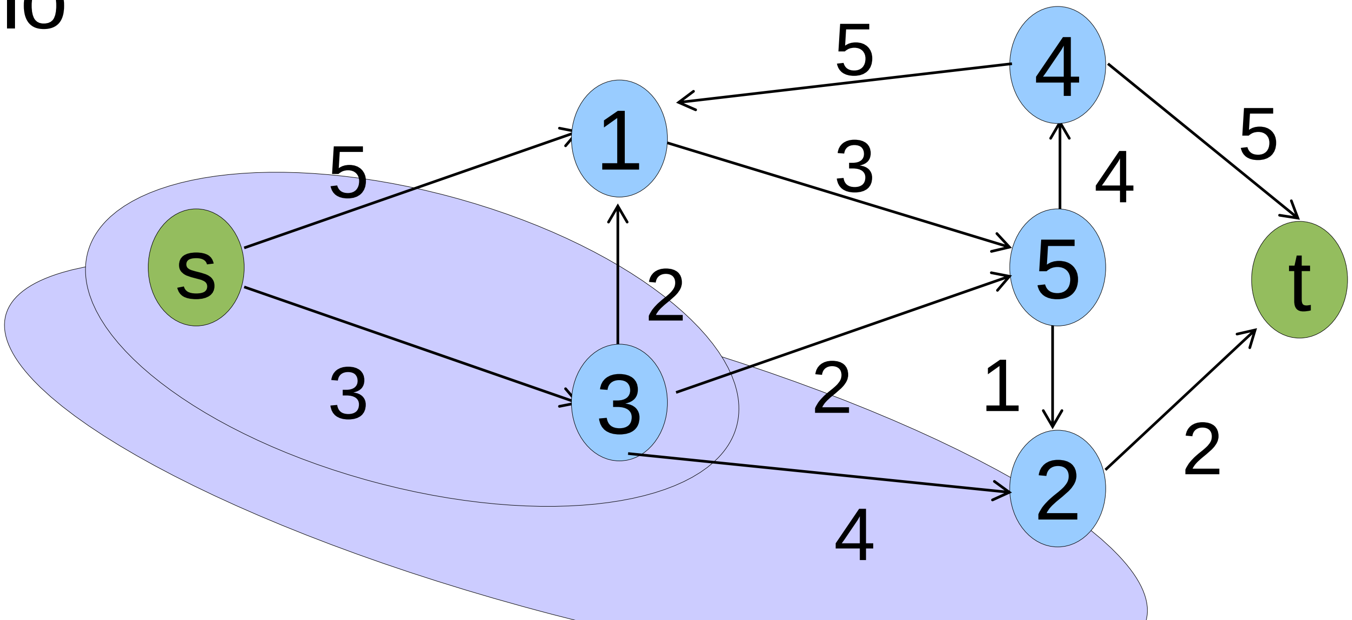
$$v(f) = \sum_{e saindo de A} f(e) - \sum_{e entrando em A} f(e)$$

Fluxo e Corte

- Dualidade fraca
- Seja f um fluxo qualquer e (A, B) um corte s-t qualquer
- Então o valor do fluxo v(f), é no máximo a *capacidade* do corte

$$v(f) \leq c(A, B)$$

- Exemplo



Dualidade Fraca

- Seja f um fluxo qualquer e (A, B) um corte s-t qualquer, então $v(f) \leq c(A, B)$

- Prova

$$v(f) = \sum_{esaindodeA} f(e) - \sum_{eentrandoemA} f(e)$$

$$. \leq \sum_{esaindodeA} f(e)$$

$$. \leq \sum_{esaindodeA} c(e)$$

$$. = c(A, B)$$

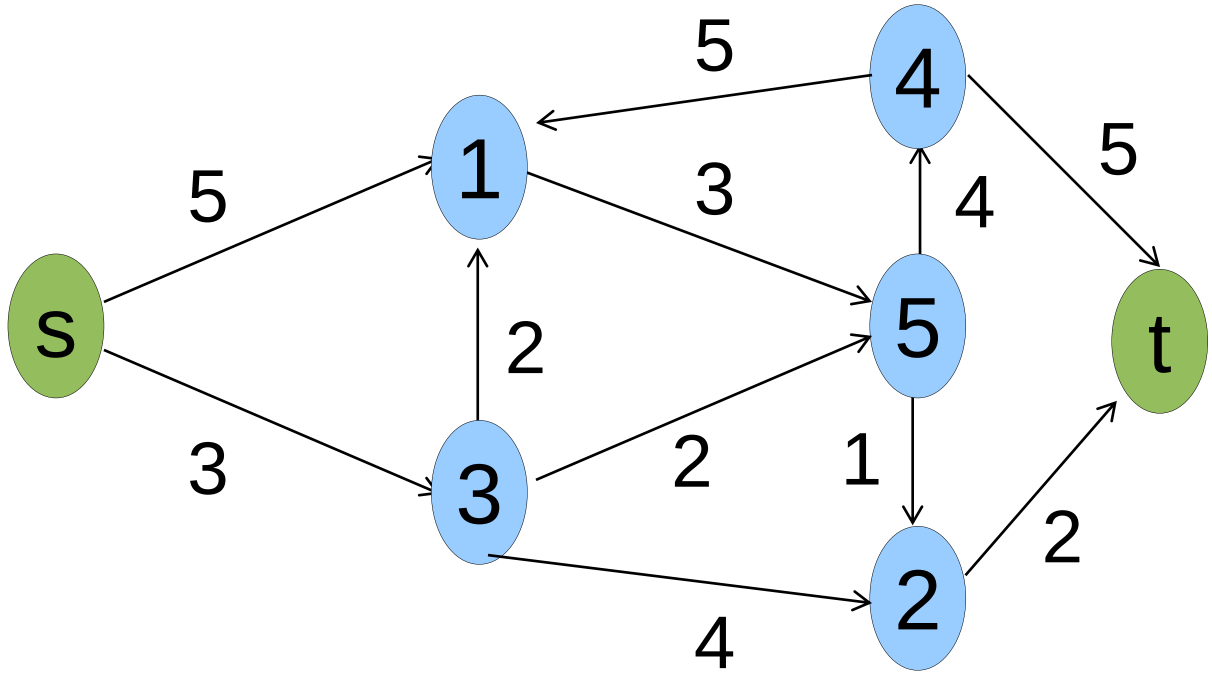
Buscando um Algoritmo

- Algoritmo guloso
- Começar com $f(e) = 0$, para todo e
- Procurar caminho P entre s-t com $f(e) < c(e)$, para todo e em P
- Aumentar fluxo em P
- Repetir até não conseguir mais

- **Problemas???**

Problema do Fluxo Máximo

- Dado G=(V,E) com capacidade nas arestas
  - e dois vértices *s* e *t*
- **Problema**: Determinar fluxo máximo entre *s* e *t*



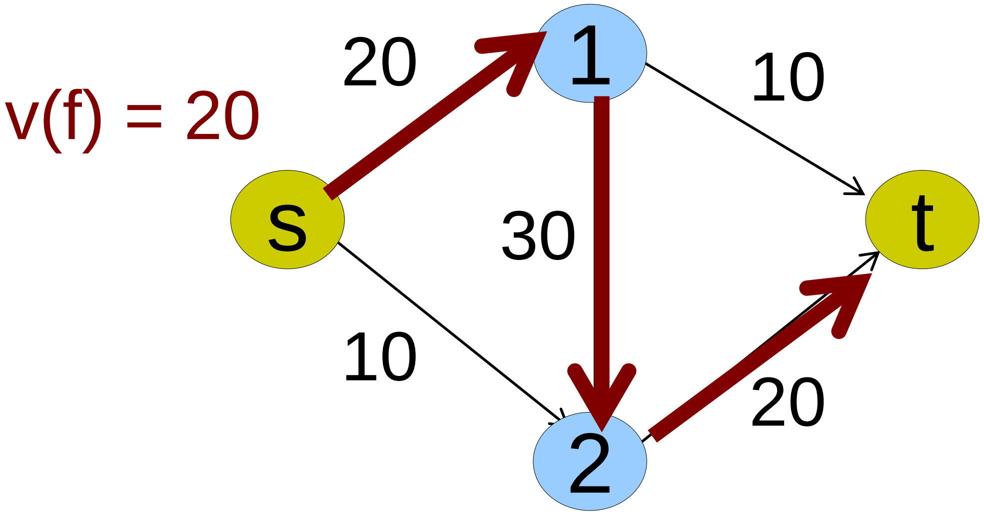
- Fluxo máximo?
- Limitante para fluxo máximo?

Fluxo na Rede

- Como definir um *fluxo* na rede?
- Determinar o fluxo de cada aresta da rede
- Função $f : E \rightarrow R$, com restrições
- 1) Capacidade
  - Fluxo em uma aresta menor que capacidade
- 2) Conservação
  - Fluxo que entra igual ao fluxo que sai
  - Fluxo que sai da origem igual fluxo que entra no destino
- Valor do fluxo *f*
  - quantidade de fluxo saindo da origem

Buscando um Algoritmo

- Algoritmo guloso
- Começar com f(e) = 0, para todo e
- Procurar caminho P entre s-t com f(e) < c(e), para todo e em P
- Aumentar fluxo em P
- Repetir até não conseguir mais



- Problema: fluxo não volta atrás!

Grafo Residual

- **Idéia**: dar chance do fluxo voltar atrás!
- Construir um grafo onde isto é possivel
  - Grafo residual
- Arestas (direcionadas) originais: $e = (u, v)$
  - capacidade $c(e)$, fluxo $f(e)$
- Arestas residuais (do grafo residual)
  - dois tipos: originais e reversas
  - $e = (u,v)$, $e^R = (v,u)$
  - manter os dois tipos no grafo residual

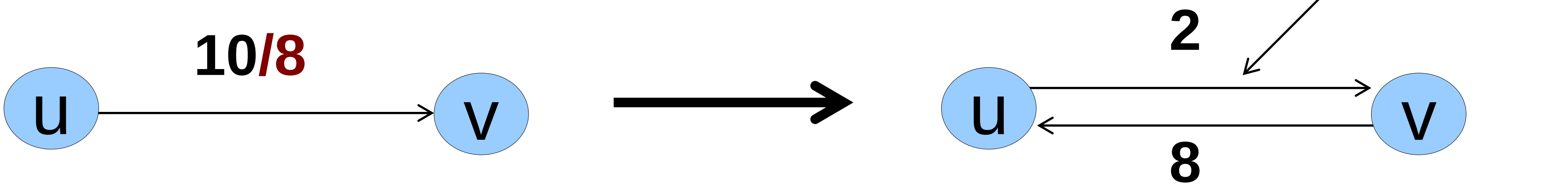
Arestas do Grafo Residual

- Capacidade das arestas
  - dado grafo original e um fluxo f

$c_f(e) = c(e) - f(e)$, quando e for original

$c_f(e) = f(e)$, quando e for reversa

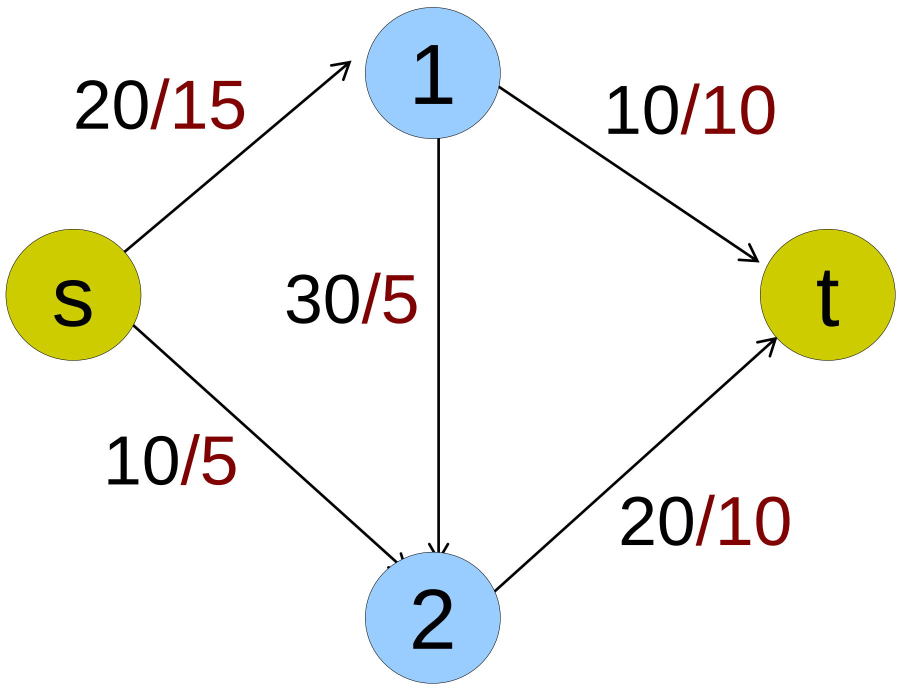
- Exemplo



- Grafo residual
  - Todos os vértices, mais arestas (originais e reversas) com capacidade

Construindo Grafo Residual

- Vértices iguais ao original
- Arestas originais e reversas com capacidade
  - apenas quando capacidade > 0
- Exemplo



Aumentando Fluxo

- **Idéia**: aumentar o fluxo de um caminnho
  - “saturar” o caminho
- Dado um caminho P, entre s e t no grafo residual
  - atualizar fluxo do caminho
- Encontrar gargalo do caminho, b
  - capacidade da aresta de menor capacidade
- Atualizar fluxos
  - $f(e) = f(e) + b$, se e for aresta original
  - $f(e^R) = f(e^R) - b$, se e for aresta reversa

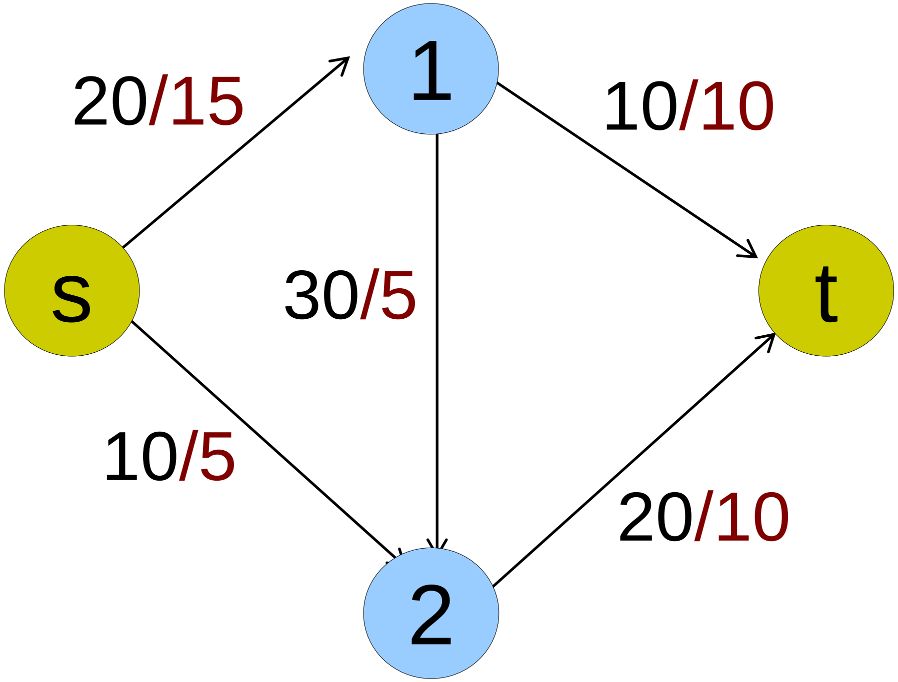
Aumentando Fluxo

```
Augment(f, c, P) {
   b ← bottleneck(P)
   foreach e ∈ P {
      if (e ∈ E) f(e) ← f(e) + b
      else       f(eR)← f(eR) - b
   }
   return f
}
```

f : fluxo
c : capacidade
P : caminho

- Exemplo: P = {s, 1, t}

P não possui
aresta reversa



Ford-Fulkerson

- **Idéia**: aumentar o fluxo dos caminhos, enquanto for possivel
- 1) Inicializar com fluxo 0
- 2) Descobrir um caminho P (no grafo residual)
  - aumentar fluxo deste caminho, via gargalo
- 3) Atualizar grafo residual
  - capacidade das arestas
- 4) Parar quando não houver mais caminho P

Ford-Fulkerson

- Algoritmo

```
Ford-Fulkerson(G, s, t, c) {
   foreach e ∈ E  f(e) ← 0
   Gf ← residual graph

   while (there exists augmenting path P) {
      f ← Augment(f, c, P)
      update Gf
   }
   return f
}
```

- Exemplo?

Análise do Término

- Assumir capacidades inteiras
- Então fluxos e capacidades residuais interias
  - Fluxo máximo é inteiro
- C : limitante para fluxo máximo
  - C = soma das capacidades de saída de *s*
- **Teorema:** algoritmo termina em no máximo $v(f^*) <= C$ iterações
- **Prova:** cada iteração aumenta valor do fluxo em ao menos uma unidade
  - Gargalo é sempre no minimo, b = 1

Complexidade

- Número de iterações = $O(C)$
- Complexidade de cada iteração?
- Encontrar caminho P
  - BFS = $O(n + m)$
- Descobrir gargalo e atualizar fluxos do caminho
  - Caminho mais comprido = $O(n)$
- Atualizar grafo residual
  - Iterar por arestas (originais + residuais) = $O(n + m)$
- Assumir grafo conexo: $m = W(n)$
- Cada passo = $O(m)$

Complexidade

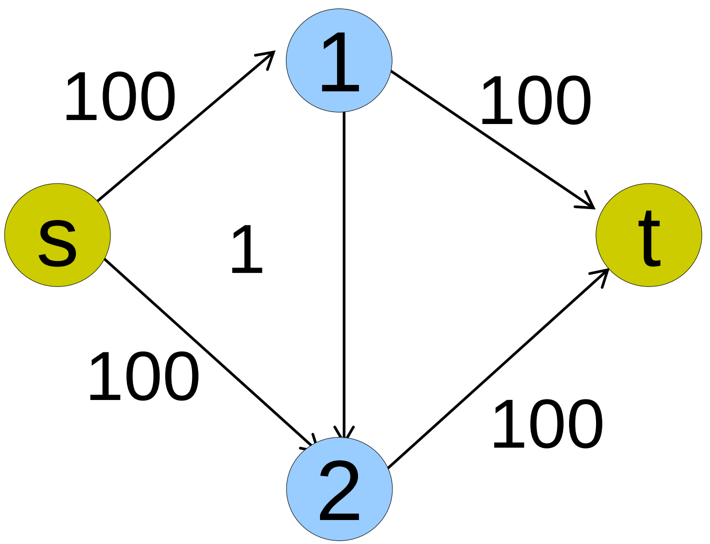
- Complexidade de cada passo: O(m)
- Total de passos: O(C)
- Complexidade: O(mC)
- Algoritmo Polinomial?
- *Pseudo-Polinomial*
  - C é um número com $\log_2$ C bits

Capacidade Reais

- Assumimos capacidade inteira
  - Onde? Para que?
- Algoritmo pode não terminar com capacidades reais
  - Incremento a cada passo pode ser arbitrariamente pequeno
  - Número de iterações pode divergir
- Modelos de problemas reais trabalham com capacidade inteiras
  - ex. bps, carros por hora, etc.

Casos Patológicos

- Número de iterações pode ser muito alto
- Escolha patológica dos caminhos para aumento de fluxo
- Exemplo
- Escolher P1 = {s, 1, 2, t}
- Escolher P2 = {s, 2, 1, t}
- Alternar entre eles...
  - 200 iterações, pois C = 200



Melhorando Algoritmo

- **Idéia**: encontrar caminhos de maior capacidade, maiores gargalos
- Problema: caminho com maior gargalo (entre todos) pode ser difícil encontrar
  - Aumento do tempo de cada iteração
- **Idéia**: caminhos com gargalos suficientemente grandes
  - reduzir restrição ao longo do algoritmo
- Restringir caminhos do grafo residual

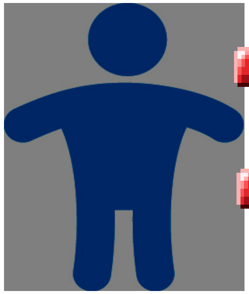
Melhorando Algoritmo

- Δ : parâmetro de escala
- $G_f(\Delta)$: grafo residual restringido
  - arestas com capacidade residual de ao menos Δ
- Algortimo modificado
  - 1) Δ = maior potência de 2, que não seja maior do que C (capacidade de saída de s)
  - 2) Trabalhar com $G_f(\Delta)$ até que não haja mais P
  - 3) Fazer $\Delta = \Delta/2$
  - 4) $G_f(1)$ = Grafo residual convencional

Complexidade

- Número de iterações que reduzem D
  - $\log_2 C$
- Número máximo de caminhos P para um dado valor de D
  - primeira iteração: 1
  - em geral, no máximo 2m (pode-se mostrar)
- Custo total: $O(m^2 \log C)$
- Polinomial?
- Outras variações que não dependem de C
  - Custo $O(mn)$

Formando Pares

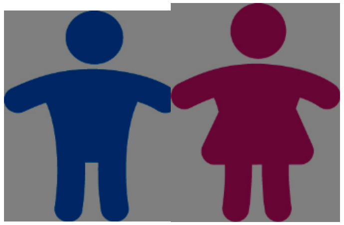


- N rapazes
- Cada rapaz declara interesse em uma ou mais moça

- N moças
- Cada moça declara interesse em um ou mais rapaz

- Casal pode “sair junto” (formar um par) se existe **interesse mútuo**
- **Problema 1:** Qual maior número de pares que podemos formar?
- **Problema 2:** Quais pares devemos formar?

Formando Pares

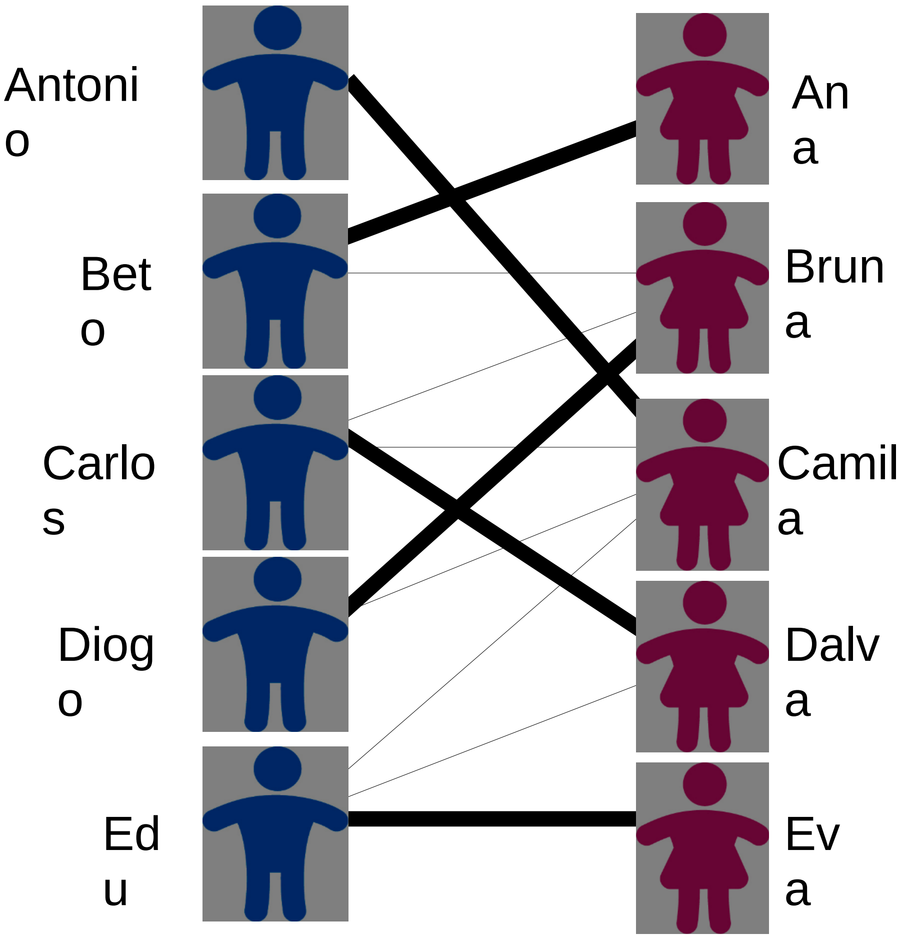


- Como abstrair o problema (usando grafos)?
- Objeto: pessoas (rapazes e moças)
- Relacionamento: interesse mútuo em sair

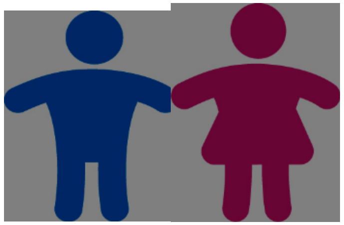
Exemplo:

Ana e Beto têm interesse mútuo!

Podemos formar 5 pares?



Problema Genérico



- Emparelhamento em grafos bipartites
- Grafo bipartite (ou bipartidos)
  - dois tipos de vértices, arestas apenas entre tipos diferentes
- Determinar maior *emparelhamento*
  - emparelhamento: formação de pares
  - perfeito: todos vértices estão emparelhados

Algoritmo para problema genérico?

Aplicação de Fluxo Máximo

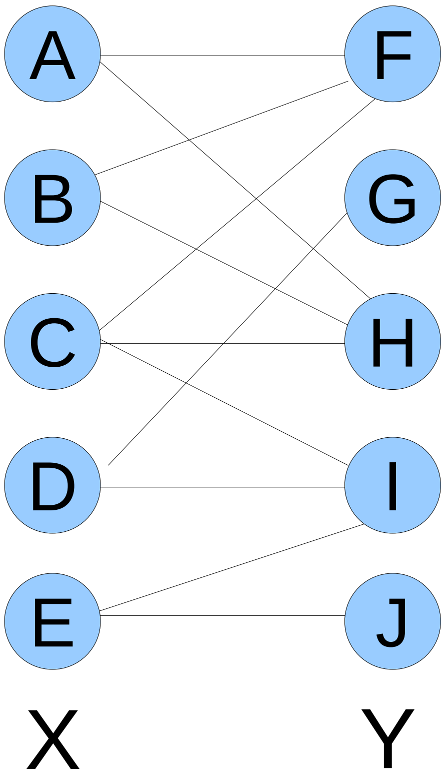
- **Idéia**: Converter problema original em problema de fluxo máximo

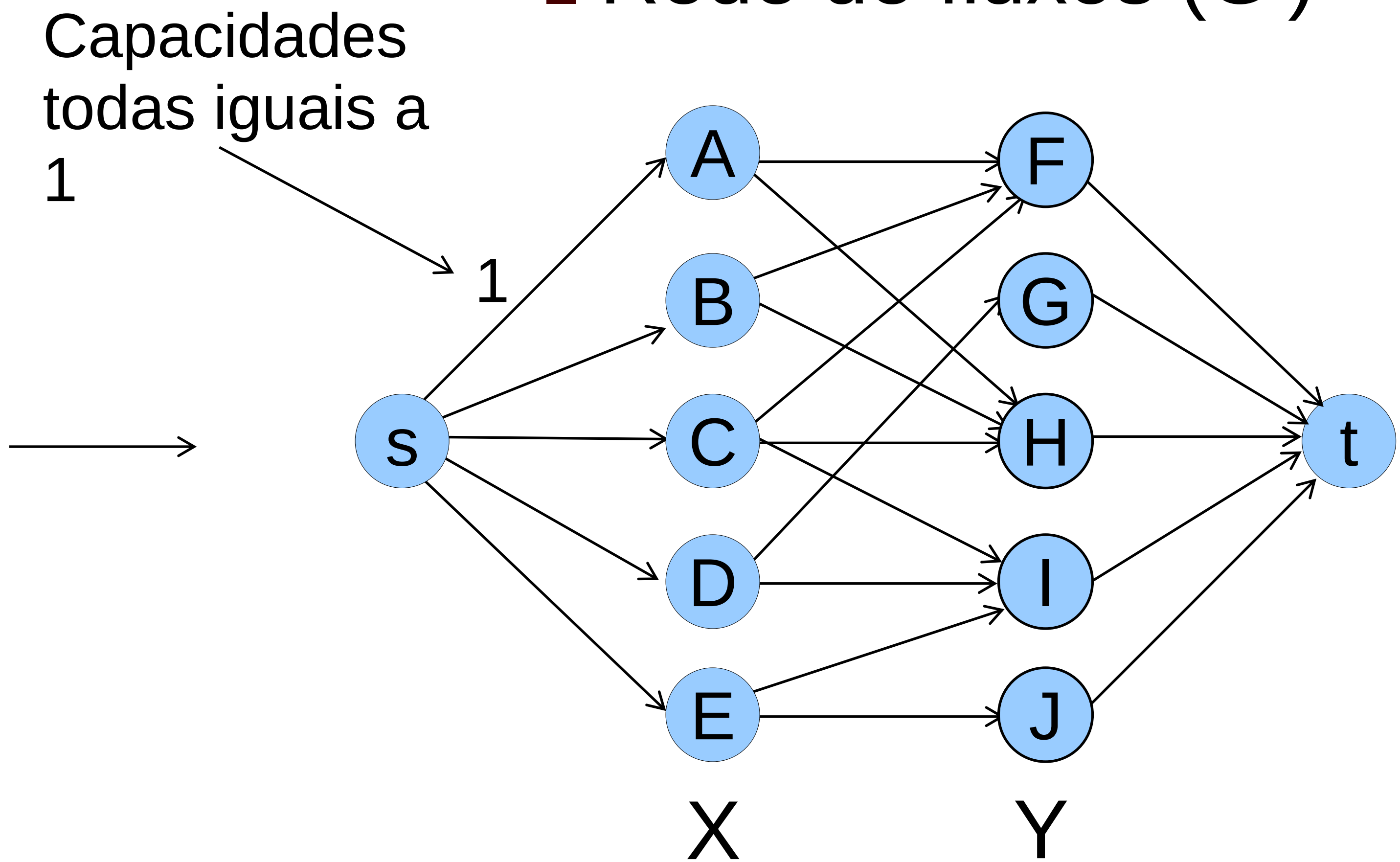
Como?

- Construir redes de fluxo direcionada
- Adicionar vértice s
  - conectar ao vértices do conjunto X
- Adicionar vértice t
  - conectar vértices do conjunto Y a t
- Direcionar arestas do grafo original X -> Y
- Capacidade 1 em todas as arestas

Exemplo

- Original (G)
- Rede de fluxos (G')





- Hipótese: Fluxo máximo em G' é igual ao maior emparelhamento em G!

Provando Hipótese (1/2)

- Emparelhamento com k pares em G tem fluxo k em G'
  - cada par contribui unidade de fluxo
- Fluxo com valor k em G' emparelha k pares em G
  - assumir integridade de fluxo (fluxo inteiros): $f(e) = 0$ ou $f(e) = 1$
  - arestas com fluxo são arestas emparelhadas
- Seja M' conjunto de arestas com $f(e) = 1$
  - M' possui k arestas

Provando Hipótese (2/2)

- Cada vértice em X incidente em no máximo uma aresta em M'
  - Conservação de fluxo
- Cada vértice em Y incidente em no máximo uma aresta em M'
  - Conservação de fluxo

- Valor do fluxo máximo em G' é igual ao maior emparelhamento em G
- Vértices com f(e) = 1 correspondem aos vértices do emparelhamento

Complexidade

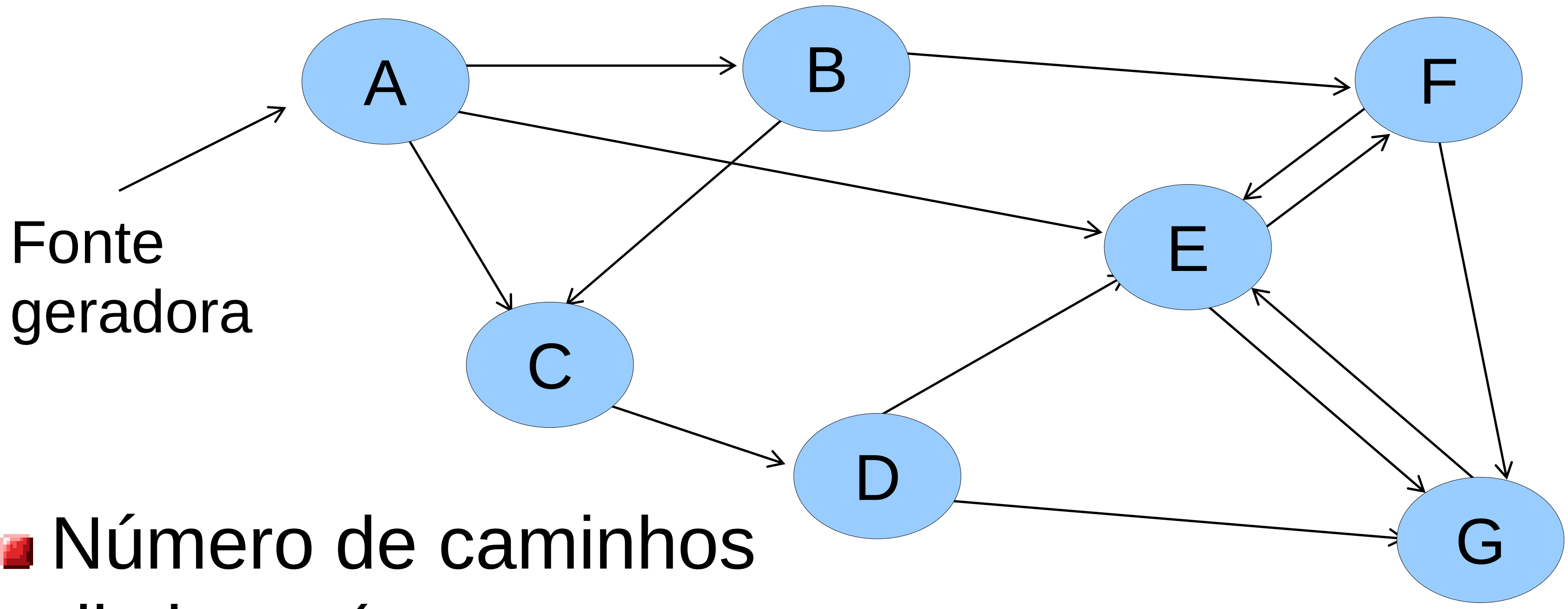
- Algoritmo original: $O(mC)$
- Algoritmo modificado: $O(m^2 \log C)$
- Quanto vale C?
- $C = O(n)$
- Custo via algoritmo original: $O(mn)$
- Custo menor via algoritmo original!
- Porque?

Robustez da Malha Elétrica

- Malha elétrica (distribuição de energia)
- Torres e linhas de transmissão
- **Robustez:** número de caminhos distintos (em linhas)
- **Problema:** Determinar robustez entre fonte geradora e ponto consumidor

Robustez da Malha Elétrica

- Objetos: torres de transmissão
- Arestas: linhas entre torres



- Número de caminhos distintos (em arestas)?

Problema Genérico

- Dado grafo direcionado G
  - e dois vértices s, t quaisquer
- Encontrar número de caminhos distintos
  - em termos de arestas
  - podemos repetir vértices
- Encontrar os caminhos em si
  - não só o número

Algoritmo para problema genérico?

Aplicação de Fluxo Máximo

- **Idéia**: Converter problema original em problema de fluxo máximo

Como?

- Construir rede de fluxos
  - s e t são origem/destino da rede
  - remover arestas entrada s, saída t
  - capacidade 1 em todas outras arestas
  - assumir fluxo inteiro

Hipótese e Prova

- Fluxo máximo s-t é igual número de caminhos distintos entre s e t

- Se existirem k caminhos distintos s-t, então fluxo máximo será ao menos k
  - cada caminho carrega fluxo 1, independente
- Se existir fluxo máximo k, então temos k caminhos distintos
  - Assumir fluxo inteiro
  - Intuição: cada aresta com f(e) = 1 pertence a exatamente 1 caminho (algum)

Complexidade

- Algoritmo original: O(mC)
- Quanto vale C?
- C = O(n)
- Custo via algoritmo original: O(mn)
- Custo menor via algoritmo original
  - De novo!

Grafos Não-Direcionados

- Considerar grafos não-direcionados
  - E dois vértices quaisquer s, t
- Número de caminhos distintos entre eles?
  - Em termos de arestas
- Caminho entre eles?
  - Não só o número

Como extender modelo anterior?

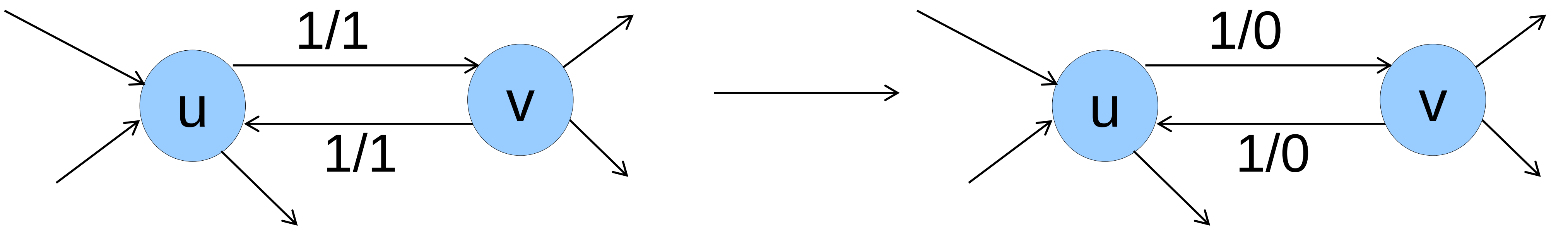
Aplicação de Fluxo Máximo

- Construir rede de fluxos direcionada
- Para cada aresta (u, v) original (não direcionada)
  - Adicionar aresta (u, v) direcionada
  - Adicionar aresta (v, u) direcionada
- Continuar como caso direcionado
  - remover arestas entrando s e saindo t, capacidade 1 em todas arestas, assumir fluxos inteiros
- **Problema**: apenas uma das arestas pode ser utilizada
  - aresta é única no grafo não-direcionado

Aplicação de Fluxo Máximo

- Problema não será problema!
- Fluxo máximo existe utilizando aresta em apenas **uma** das direções

- Supor duas direções sendo utilizadas
- Novo fluxo onde nenhuma das duas é utilizada tem mesmo valor e é um fluxo



Conectividade na Internet

- Internet: rede de redes
- Conectividade entre AS (sistemas autônomos)
- **Robustez**: conectividade global

- **Problema:** Determinar robustez da Internet
  - Número mínimo de arestas que precisam falhar para desconectar o grafo

Problema Genérico

- Problema do corte mínimo em grafos não-direcionados
- **Corte:** conjunto de arestas que se removidas “desconectam” o grafo
  - geram mais de um componente conexo
- Tamanho do corte: número de arestas que o compõem
- Corte mínimo: corte com o menor número de aresas possível

Algoritmo para problema genérico?

Aplicação de Fluxo Máximo

- **Idéia**: Converter problema original em problema de fluxo máximo

Como?

- Escolher dois vértices s e t quaisquer
- Construir rede de fluxos s-t
  - Como no caso anterior, caminhos distintos
- Obter corte mínimo entre s-t
  - corte mínimo é igual ao fluxo máximo, que é igual ao número de caminhos distintos
- Corte mínimo global?

Aplicação de Fluxo Máximo

- Problema: corte mínimo s-t não necessariamente é corte mínimo global?

Solução?

- Fixar s, variar t para cada vértice do grafo
- Determinar corte mínimo para cada t
  - via redes de fluxo
- Corte mínimo global é o menor deles
- **Obs:** corte mínimo global necessariamente separa s de t para algum par s e t

Complexidade

- Fixar s, variar t
  - Total de n-1 rodadas
- Algoritmo original: O(mC)
- Quanto vale C?
- C = O(n)
- Custo de cada rodada: O(mn)
- Custo total: $O(mn^2)$
- Mas existem algortimos mais eficientes específicos para este problema