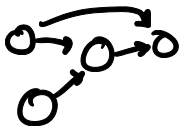


DAG:



≡ ord. top

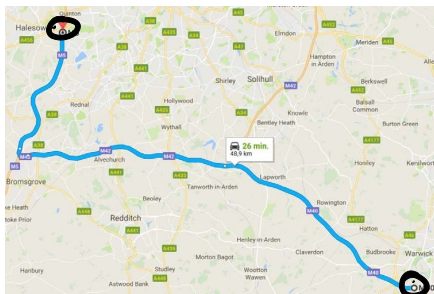
Caminho mais Curto

Caminho mais curto

# Caminho mais curto



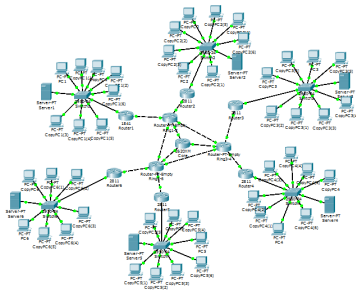
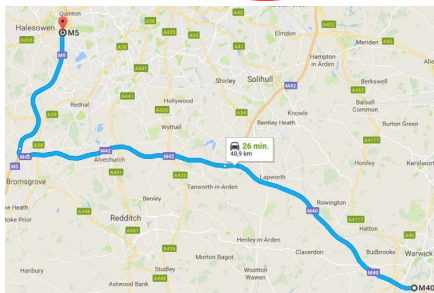
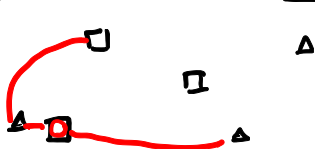
Como Google Maps encontra o melhor caminho até o meu destino?



# Caminho mais curto

Como Google Maps encontra o melhor caminho até o meu destino?

Como pacotes são roteados na internet até o destino?



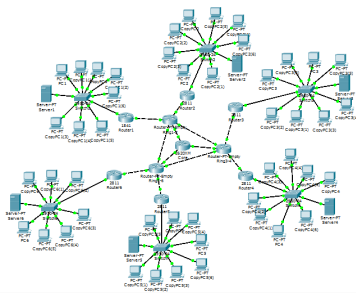
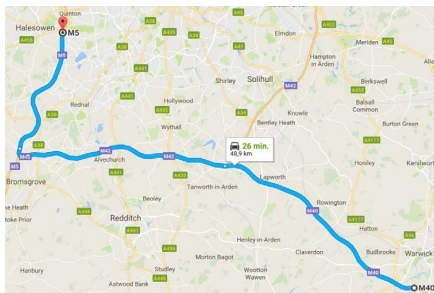
# Caminho mais curto

Como *Google Maps* encontra o melhor caminho até o meu destino?

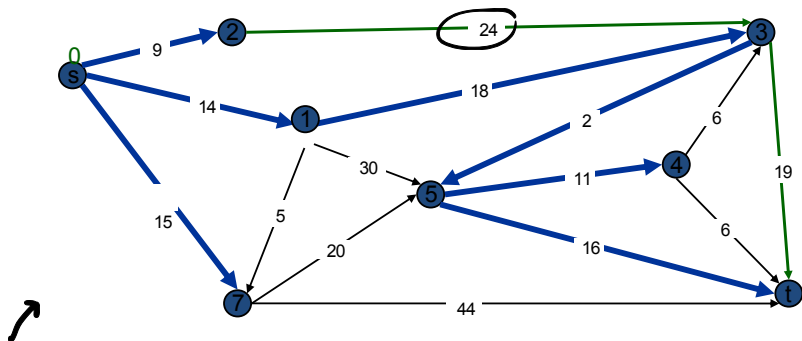
Como pacotes são roteados na internet até o destino?

...

Problemas de **caminho mais curto** em grafos

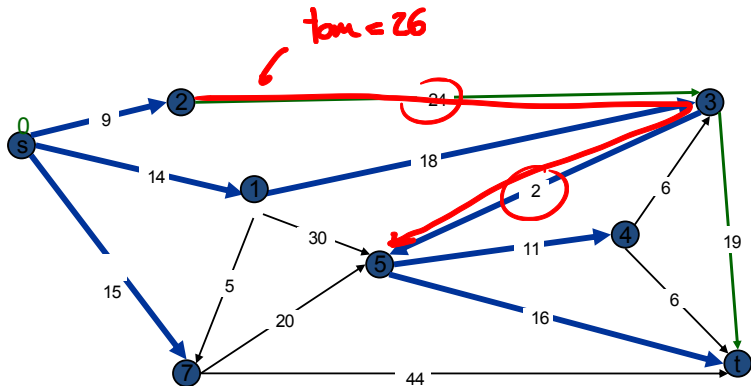


Vamos considerar grafos direcionados onde cada aresta  $e$  tem um tamanho  $w(e)$  positivo



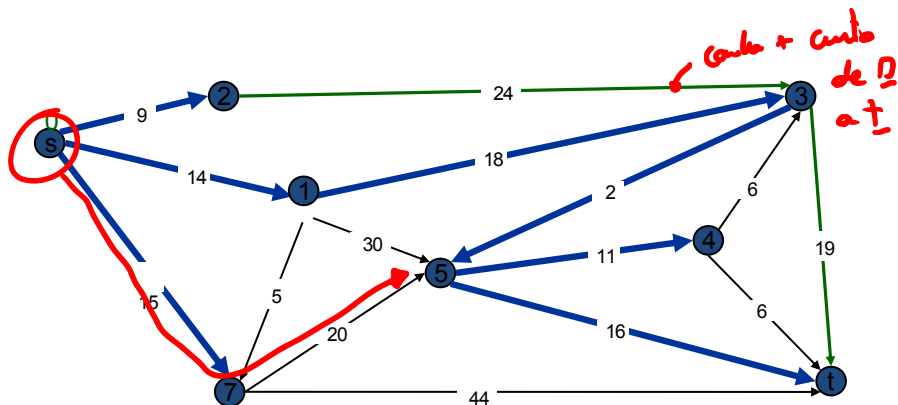
Vamos considerar grafos direcionados onde cada aresta  $e$  tem um tamanho  $w(e)$  **positivo**

O tamanho de um caminho é a soma dos tamanhos de suas arestas



**Problema de Caminho mais Curto:** Dado um grafo direcionado com tamanhos positivos nas arestas, origem  $s$  e destino  $t$ , encontre um menor caminho de  $s$  a  $t$ .

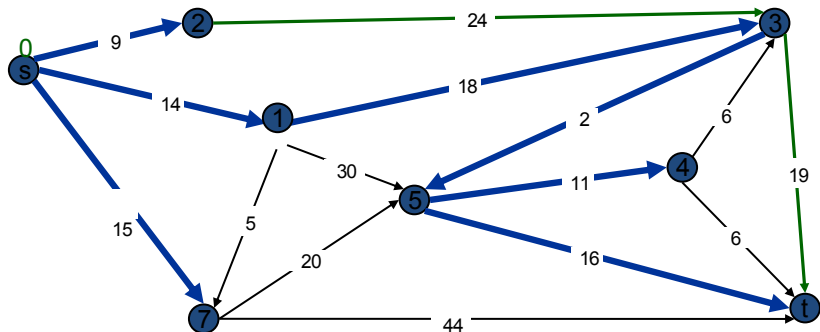
CNC de 1 a 5:  $1-7-5=35$   
^





**Problema de Caminho mais Curto:** Dado um grafo direcionado com tamanhos positivos nas arestas, origem  $s$  e destino  $t$ , encontre um menor caminho de  $s$  a  $t$ .

Pra encurtar: **CMC** = caminho mais curto



**P:** Algoritmo?

**P:** Algoritmo?

**R:** Força bruta: gerar todos os caminhos entre  $s$  e  $t$ , e selecionar o menor

**P:** Algoritmo?

**R:** Força bruta: gerar todos os caminhos entre  $s$  e  $t$ , e selecionar o menor

**Problema:** Muito **ineficiente**, o número de caminhos pode ser **exponencial** no número de vértices do grafo:

**P:** Algoritmo?

**R:** Força bruta: gerar todos os caminhos entre  $s$  e  $t$ , e selecionar o menor

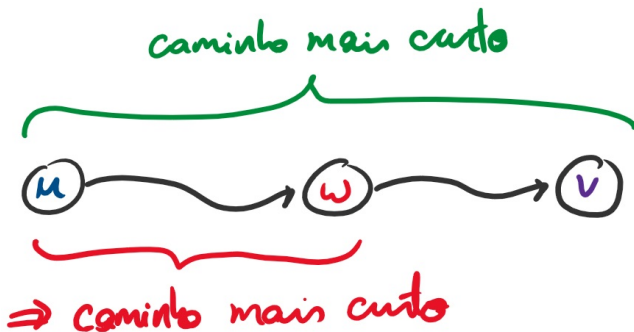
**Problema:** Muito **ineficiente**, o número de caminhos pode ser **exponencial** no número de vértices do grafo:

Vamos desenvolver um algoritmo muito mais eficiente: **Dijkstra**

Começamos com a seguinte observação

### Lema

Se  $P$  é um CMC de  $u$  a  $v$ , e  $w$  um vértice nesse caminho, então o subcaminho de  $P$   $u \rightsquigarrow w$  é um CMC entre  $u$  e  $w$

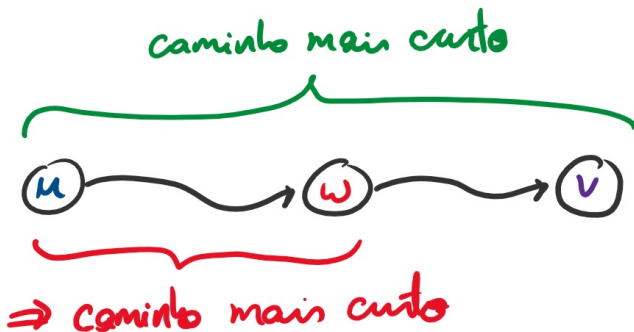


Começamos com a seguinte observação

### Lema

Se  $P$  é um CMC de  $u$  a  $v$ , e  $w$  um vértice nesse caminho, então o subcaminho de  $P$   $u \rightsquigarrow w$  é um CMC entre  $u$  e  $w$

**Prova.** Se isso não fosse verdade, poderíamos obter um caminho mais curto que  $P$  substituindo o subcaminho de  $P$   $u \rightsquigarrow w$  por um menor

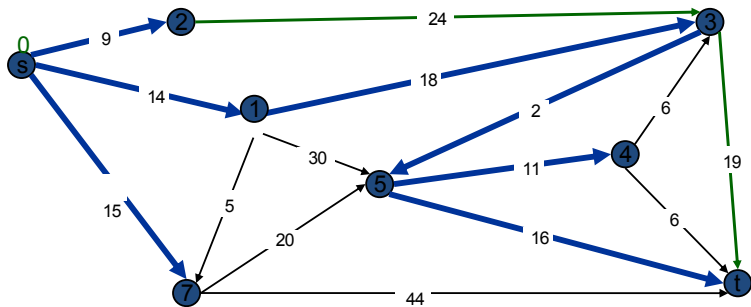


**Notação:** Vamos usar  $P_{u,v}$  pra denotar um CMC de  $u$  a  $v$

**Def:** A **distância** entre  $u$  e  $v$  é o tamanho do CMC entre eles

**Ex:** Distância de  $(s)$  a  $(7)$  é 15

Distância de  $(s)$  a  $(3)$  é 32





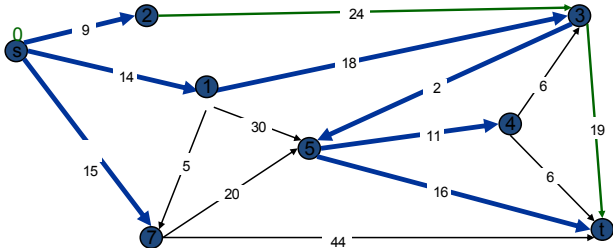
Ideia do **Dijkstra**:

## Ideia do **Dijkstra**:

- Encontrar o CMC até o **vértice mais próximo** da origem  $s$
- Encontrar o CMC até o **segundo vértice mais próximo** da origem  $s$
- ... até encontrar o destino  $t$

**Def:**  $v_k$  o  $k$ -ésimo vértice mais próximo de  $s$

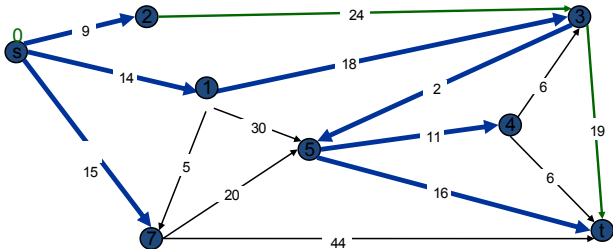
(**ponta do  $k$ -ésimo menor caminho desde  $s$** )



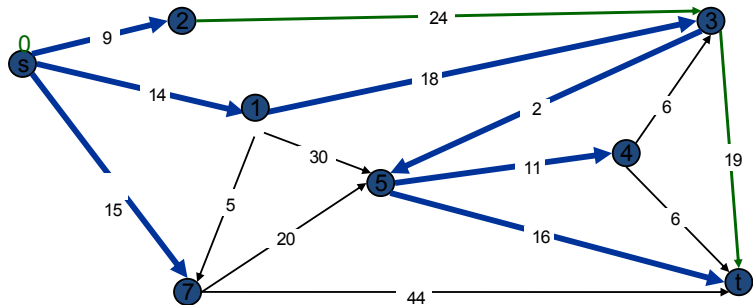
## Ideia do Dijkstra:

- Encontrar o CMC até o **vértice mais próximo** da origem  $s$  (**menor caminho desde  $s$** )
- Encontrar o CMC até o **segundo vértice mais próximo** da origem  $s$  (**segundo menor caminho desde  $s$** )
- ... até encontrar o destino  $t$

**Def:**  $v_k$  o  $k$ -ésimo vértice mais próximo de  $s$   
(**ponta do  $k$ -ésimo menor caminho desde  $s$** )

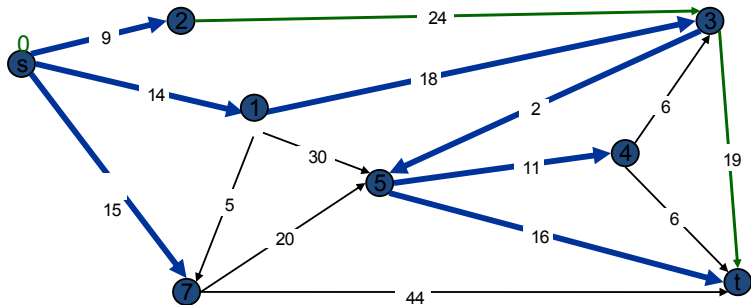


Seja  $v_k$  o  $k$ -ésimo vértice mais próximo de  $s$



Seja  $v_k$  o  $k$ -ésimo vértice mais próximo de  $s$

(ponta dos  $k$ -ésimo menor caminho a partir de  $s$ )



Vamos construir os CMCs **indutivamente**:

Vamos construir os CMCs **indutivamente**:

- Conhecemos os  $k - 1$  nós mais pertos  $v_1, \dots, v_{k-1}$  e CMC desde  $s$   
**( $k - 1$  menores caminhos desde  $s$ )**

Vamos construir os CMCs **indutivamente**:

- Conhecemos os  $k - 1$  nós mais pertos  $v_1, \dots, v_{k-1}$  e CMC desde  $s$   
**( $k - 1$  menores caminhos desde  $s$ )**
- Queremos encontrar  $v_k$  e seu CMC desde  $s$   
**( $k$ -ésimo menor caminho desde  $s$ )**



Vamos construir os CMCs **indutivamente**:

- Conhecemos os  $k - 1$  nós mais perto  $v_1, \dots, v_{k-1}$  e CMC desde  $s$   
**( $k - 1$  menores caminhos desde  $s$ )**
- Queremos encontrar  $v_k$  e seu CMC desde  $s$   
**( $k$ -ésimo menor caminho desde  $s$ )**

**P:** Considere o CMC a  $v_k$ , seja  $u$  o penúltimo nó nesse caminho.  
Quem está mais **perto** de  $s$ ,  $u$  ou  $v_k$ ?

Vamos construir os CMCs **indutivamente**:

- Conhecemos os  $k - 1$  nós mais pertos  $v_1, \dots, v_{k-1}$  e CMC desde  $s$   
**( $k - 1$  menores caminhos desde  $s$ )**
- Queremos encontrar  $v_k$  e seu CMC desde  $s$   
**( $k$ -ésimo menor caminho desde  $s$ )**

**P:** Considere o CMC a  $v_k$ , seja  $u$  o penúltimo nó nesse caminho.  
Quem está mais **perto** de  $s$ ,  $u$  ou  $v_k$ ?

**A:**  $u$  está mais perto

Vamos construir os CMCs **indutivamente**:

- Conhecemos os  $k - 1$  nós mais pertos  $v_1, \dots, v_{k-1}$  e CMC desde  $s$   
**( $k - 1$  menores caminhos desde  $s$ )**
- Queremos encontrar  $v_k$  e seu CMC desde  $s$   
**( $k$ -ésimo menor caminho desde  $s$ )**

**P:** Considere o CMC a  $v_k$ , seja  $u$  o penúltimo nó nesse caminho.  
Quem está mais **perto** de  $s$ ,  $u$  ou  $v_k$ ?

**A:**  $u$  está mais perto  $\Rightarrow u$  é um dos  $v_i$ 's já conhecidos!

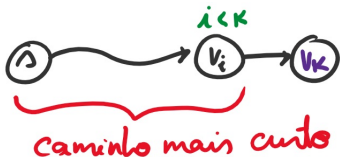
Vamos construir os CMCs **indutivamente**:

- Conhecemos os  $k - 1$  nós mais perto  $v_1, \dots, v_{k-1}$  e CMC desde  $s$   
( $k - 1$  menores caminhos desde  $s$ )
- Queremos encontrar  $v_k$  e seu CMC desde  $s$   
( $k$ -ésimo menor caminho desde  $s$ )

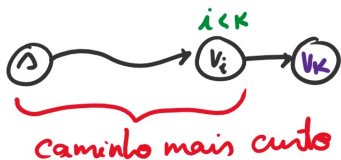
**P:** Considere o CMC a  $v_k$ , seja  $u$  o penúltimo nó nesse caminho.  
Quem está mais perto de  $s$ ,  $u$  ou  $v_k$ ?

**A:**  $u$  está mais perto  $\Rightarrow u$  é um dos  $v_i$ 's já conhecidos!

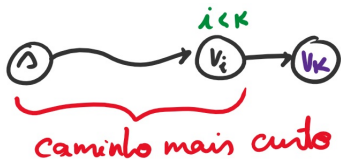
Os **candidatos** a  $k$ -ésimo caminho mais curto tem a forma



Os candidatos a  $k$ -ésimo caminho mais curto tem a forma

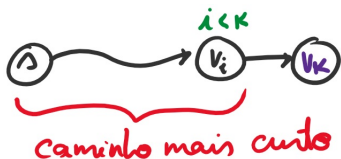


Os **candidatos** a  $k$ -ésimo caminho mais curto tem a forma



O  $k$ -ésimo caminho mais curto é o **melhor** (mais curto) candidato

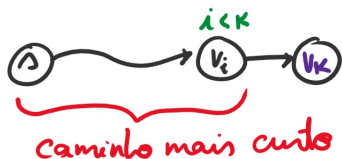
Os **candidatos** a  $k$ -ésimo caminho mais curto tem a forma



O  $k$ -ésimo caminho mais curto é o **melhor** (mais curto) candidato

O nó  $v_k$  é a ponta desse caminho

Os **candidatos** a  $k$ -ésimo caminho mais curto tem a forma



O  $k$ -ésimo caminho mais curto é o **melhor** (mais curto) candidato

O nó  $v_k$  é a ponta desse caminho

**Conseguimos computar o que queremos!**



Dijkstra( $s, t$ )

**For**  $k = 2$  **to**  $n$

$\mathcal{C} = \emptyset$     # candidatos a  $k$ -ésimo menor caminho desde  $s$

**For** todo  $i < k$     # nós mais próximos, já conhecidos

**For** todo vértice  $x$  não conhecido\* adjacente a  $v_i$

        Inclua o caminho ( $P_{s,v_i} \rightarrow x$ ) na lista  $\mathcal{C}$  dos caminhos candidatos

**Fim For**

**Fim For**

$P \leftarrow$  o menor caminho da lista  $\mathcal{C}$

$v_k \leftarrow$  último vértice do caminho  $P$

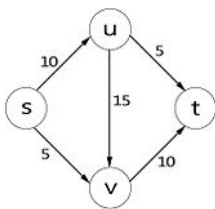
$P_{s,v_k} \leftarrow P$

**Fim For**

Retorne caminho  $P_{s,t}$

\* Não conhecido quer dizer diferente dos nós  $v_1, \dots, v_{k-1}$

**Ex:**



Em *Análise de Algoritmos* veremos uma implementação bastante eficiente desse algoritmo

# Exercícios

**Exercício 1:** Construa um grafo com aresta com comprimento **negativo** onde Dijkstra **não** retorna o caminho mais curto de  $s$  a  $t$

**Exercício 2:** Considere um problema diferente: agora os “custos” estão **nos nós**, e **não nas arestas**

O custo de um caminho é a soma dos custos de seus nós.

Nessa situação, como você pode encontrar o caminho mais barato de  $s$  a  $t$  usando um algoritmo de caminho mais curto?

# Modelando como caminhos mais curtos

Podemos modelar **diversos problemas** como problema de caminho mais curto

**Exercício 3:** Você tem uma pequena empresa de consultoria, com clientes no Rio e São Paulo

Em cada mês você pode escolher basear seu escritório no Rio ou SP (Começa no Rio)

No fim do mês  $i$  você decide se mantém o escritório no lugar atual ou muda pra próxima cidade, e paga **custo**. Os custos estão na tabela abaixo

Você quer decidir onde o escritório deve estar em cada momento de forma a **minimizar** o custo total. **Modele usando caminho mais curto**

movimento	mes 1	mes 2	mes 3	mes 4
SP-SP	-	3	20	30
SP-R	-	30	12	14
R-R	50	20	2	4
R-SP	11	33	30	40

**Dica:** Use um grafo com 4 “colunas” de nós, cada uma com 2 nós