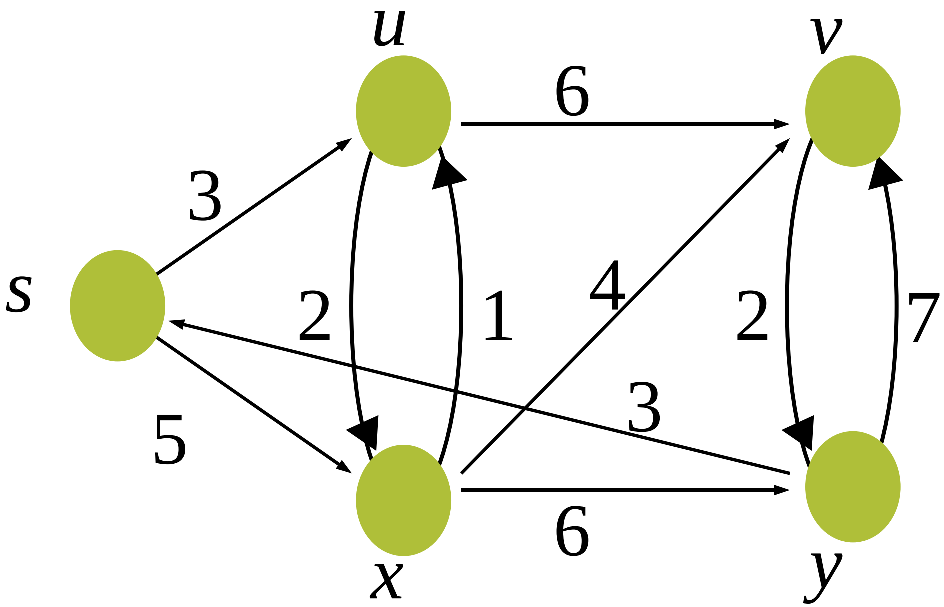
Grafos: caminhos mínimos em Listas de Adjacência

Profa. Graça Nunes

Caminhos mínimos

- O problema do caminho mínimo consiste em determinar um menor caminho entre um vértice de origem e um vértice de destino

- Qual o menor caminho entre s e y?



Caminho mínimo

- Grafo dirigido G(V,E) com função peso *w*: $E \rightarrow \Re$ que mapeia as arestas em pesos
- Peso (custo) do caminho p = <v_0, v_1, ..., v_k>

$$w(p) = \sum_{i=1}^{k} w(v_{i-1}, v_i)$$

- Caminho de menor peso entre u e v:

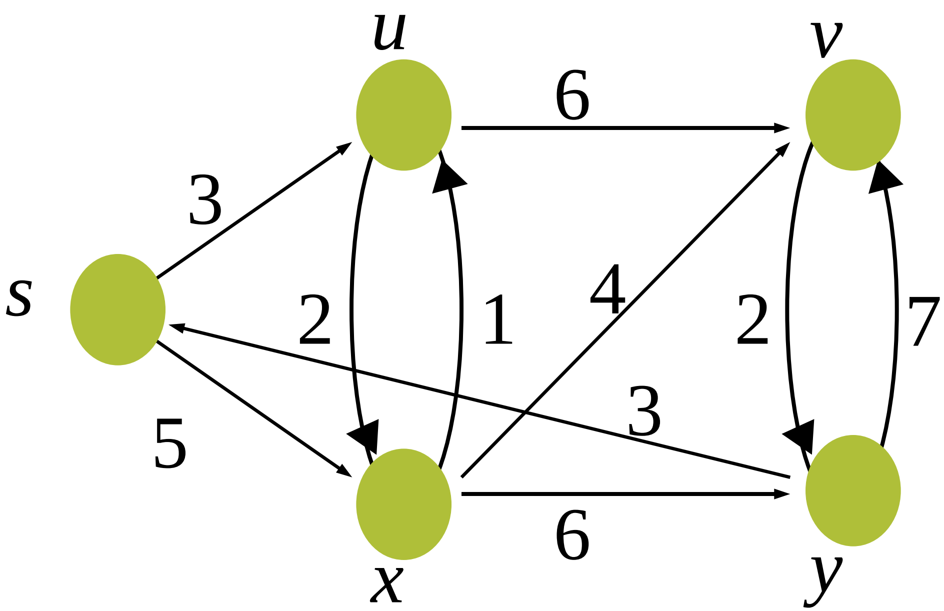
$$\delta(u,v) = \begin{cases} \min\{w(p) : u \overset{p}{\Rightarrow} v\} \; se \; \exists \; rota \; de \; u \;\; p/v \\ \infty \; cc \end{cases}$$

Caminho mínimo

- Várias possibilidades de caminhos
  - Caminhos mínimos de origem única (Diykstra)
  - Caminhos mínimos de destino único
  - Caminho mínimo de par único
  - Caminhos mínimos de todos os pares

Caminhos mínimos de origem única,s

- Conceitos
  - Associa-se a cada vértice, x, um número d(x) indicando o menor custo entre s e x
  - P.ex., quando chegamos ao vértice v, na figura abaixo, d(v) será min(d(u)+6, d(x)+4, d(y)+7)



Caminhos mínimos de origem única, s

- Conceitos
  - *Relaxamento* de arestas
    - Faz-se uma estimativa pessimista para o caminho mínimo até cada vértice: d(v)=∞
    - O processo de relaxar uma aresta consiste em verificar se é possível melhorar esta estimativa passando-se pelo vértice u

Caminhos mínimos de origem única, s

- Conceitos
  - *Relaxamento* de arestas
    - Faz-se uma estimativa pessimista para o caminho mínimo até cada vértice: d(v)=∞
    - O processo de relaxar uma aresta consiste em verificar se é possível melhorar esta estimativa passando-se pelo vértice u

Caminhos mínimos de origem única, s

- Sub-rotina para relaxamento de estastas

```
relax(u, v, w) /* recálculo de d(v) quando alcançado via
   lista de adjacência de u - w é o peso da aresta (u,v)*/
   início
   se d[v] > d[u]+w(u,v) então
         d[v] = d[u]+w(u,v)
         antecessor[v]=u /* registra que passou por u */
   fim
```

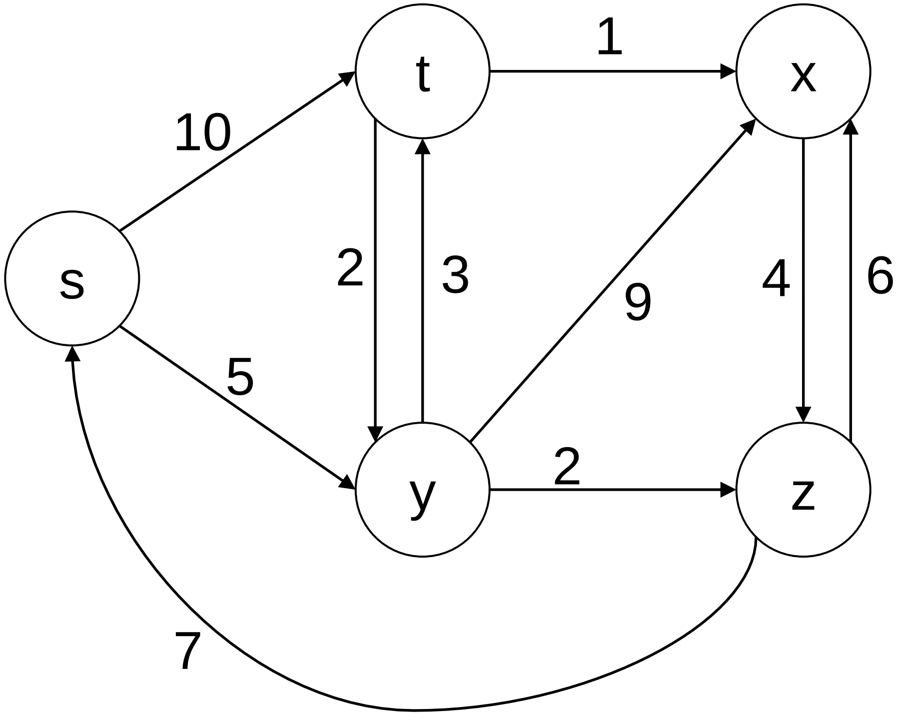
Algoritmo de Dijkstra

- Características
  - Calcula os custos dos caminhos mínimos de origem única e todos os destinos
  - Dígrafo pode ser cíclico
  - Somente pesos positivos

- Método
  - A cada passo, adiciona um vértice u, de menor estimativa de caminho mínimo (fila de prioridade), a um conjunto S inicialmente vazio
  - Relaxam-se as arestas adjacentes a u (portanto, percurso em profundidade!)
  - Cada vez que muda a estimativa, registra o caminho, alterando o antecessor

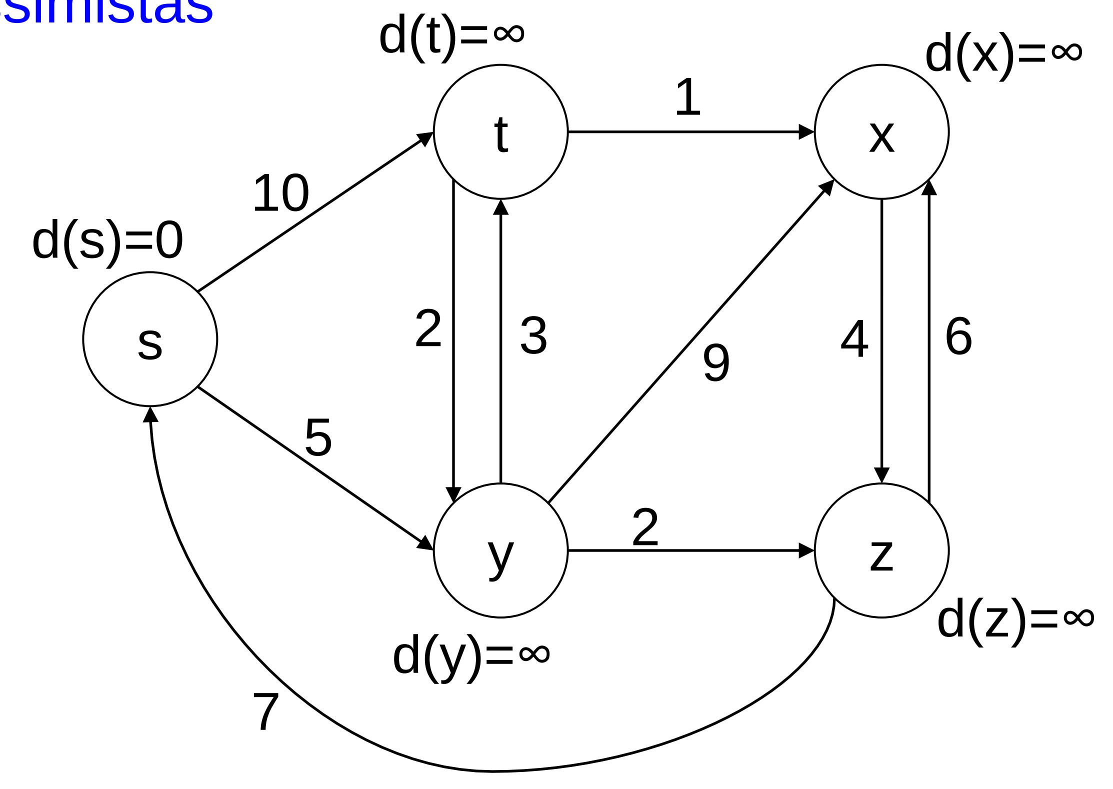
Algoritmo de Dijkstra

- Exemplo (a partir de s)



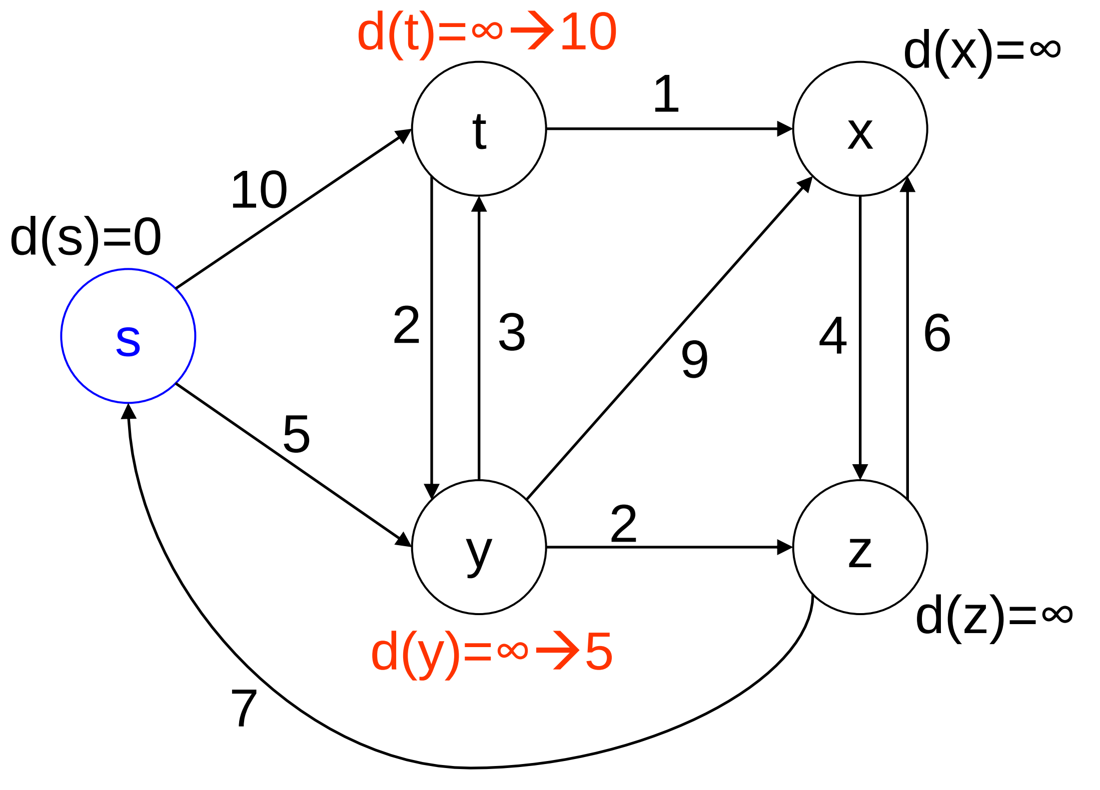
Algoritmo de Dijkstra

- Exemplo (a partir de s)
  - Estimativas pessimistas
  - S=Ø



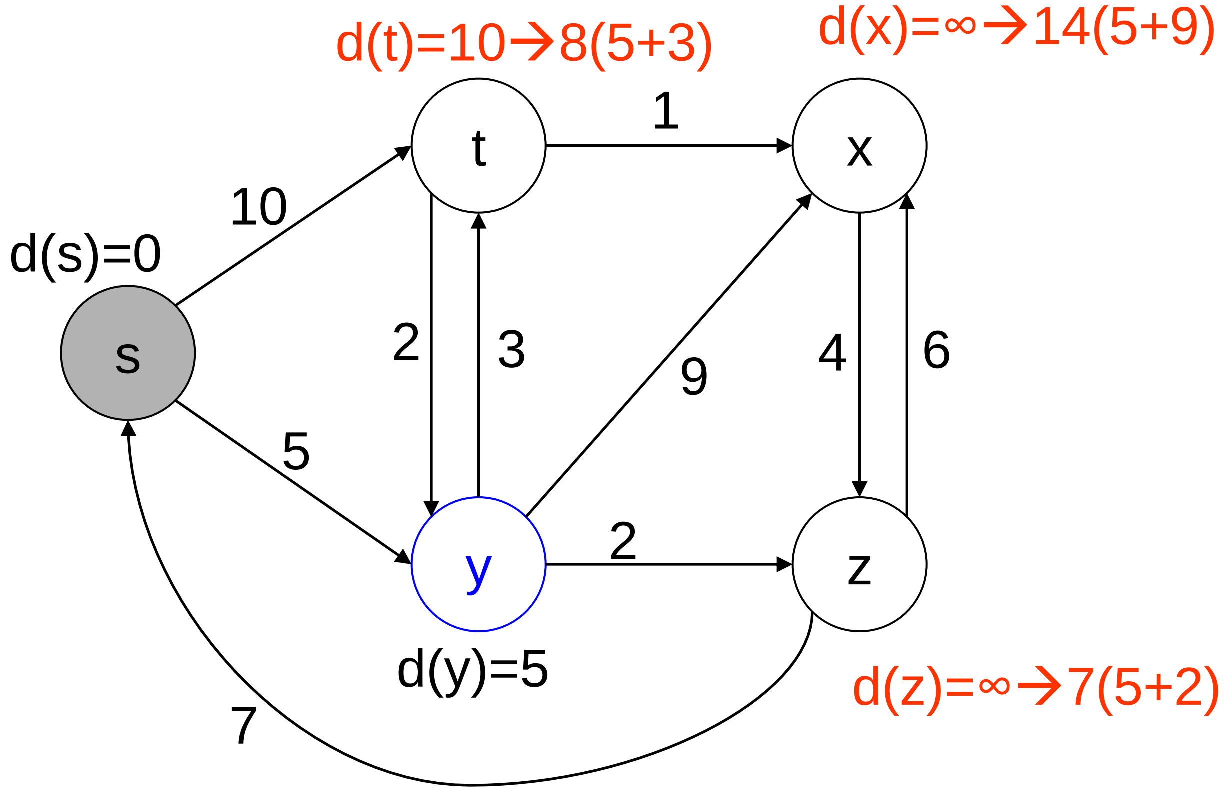
Algoritmo de Dijkstra

- Exemplo (a partir de s)
  - Adiciona s a S
  - S={**s**}
  - Antecessor(t)=s
  - Ant(y)=s



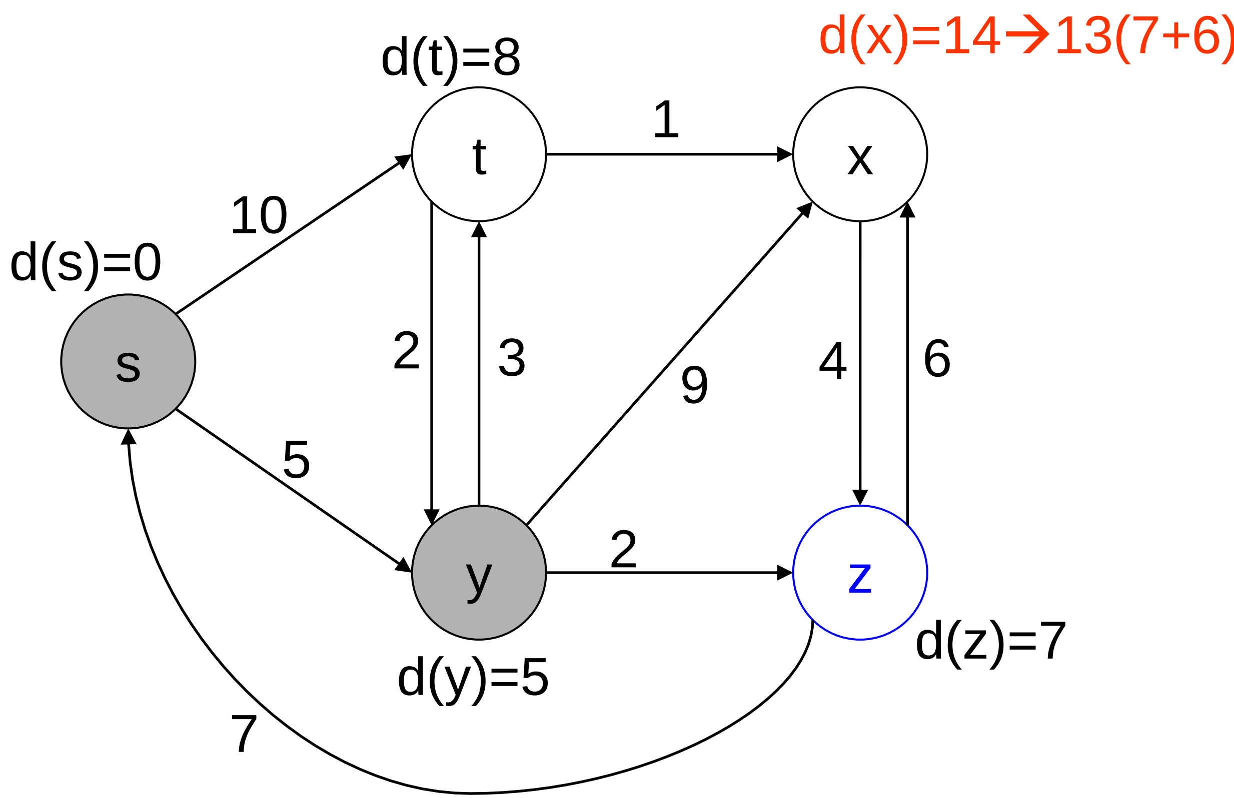
Algoritmo de Dijkstra

- Exemplo (a partir de s)
  - Adiciona y a S
  - S={s,**y**}
  - Ant(t)=y
  - Ant(y)=s
  - Ant(z)=y
  - Ant(x)=y



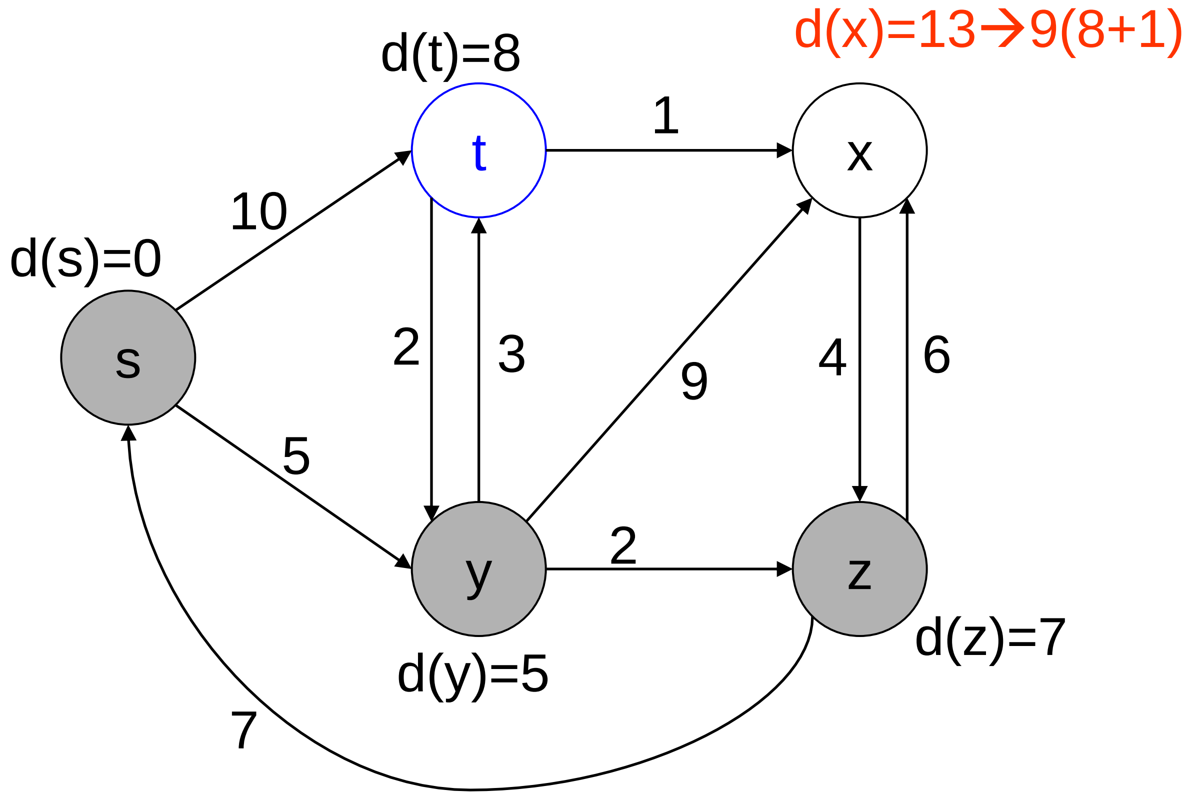
Algoritmo de Dijkstra

- Exemplo (a partir de s)
  - Adiciona z a S
  - S={s,y,**z**}
  - Ant(t)=y
  - Ant(y)=s
  - Ant(z)=y
  - Ant(x)=z



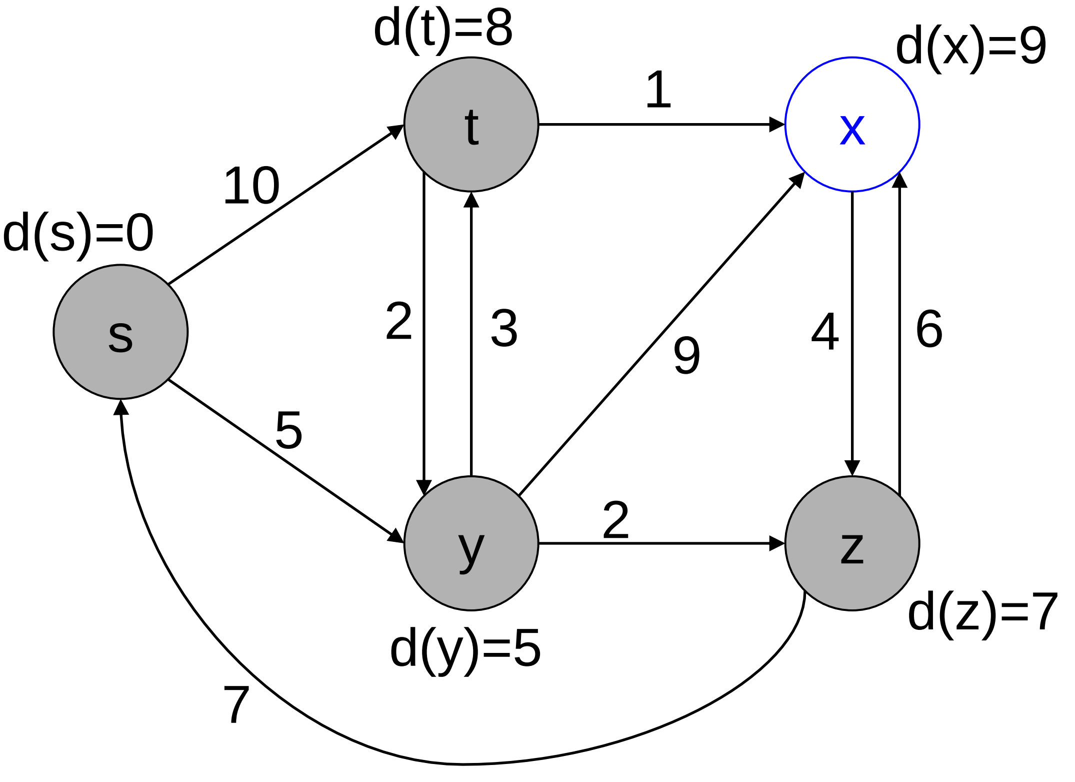
Algoritmo de Dijkstra

- Exemplo (a partir de s)
  - Adiciona t a S
  - S={s,y,z,**t**}
  - Ant(t)=y
  - Ant(y)=s
  - Ant(z)=y
  - Ant(x)=t



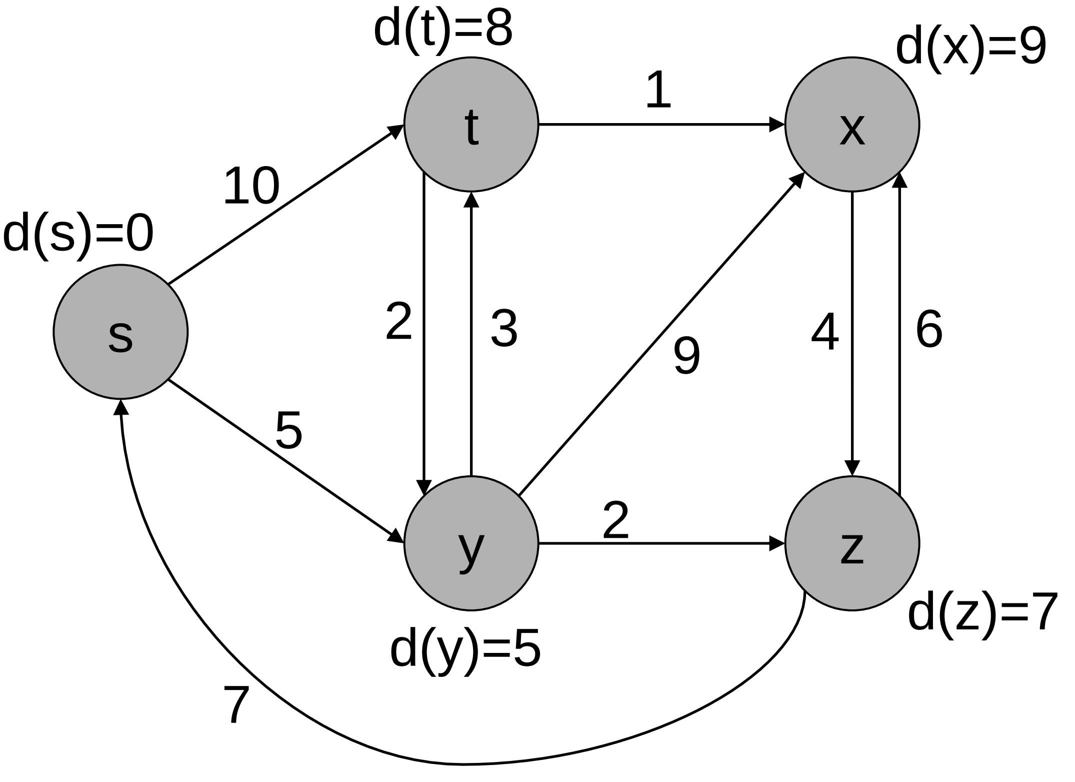
Algoritmo de Dijkstra

- Exemplo (a partir de s)
  - Adiciona x a S
  - S={s,y,z,t,**x**}
  - Ant(t)=y
  - Ant(y)=s
  - Ant(z)=y
  - Ant(x)=t



Algoritmo de Dijkstra

- Exemplo (a partir de s)
  - S={s,y,z,t,x}
  - Ant(t)=y
  - Ant(y)=s
  - Ant(z)=y
  - Ant(x)=t



Algoritmo de Dijkstra

- Implementação
  - Uso de uma fila de prioridades com vértices organizados em função da estimativa d de caminho mínimo
  - Recupera o próprio caminho a partir da lista antecessor (i), no sentido inverso:
    - O caminho mínimo de s a x, no exemplo, é dado por:
    - (x, antecessor(x), ….., s), ou seja: (s, y, t, x)

Algoritmo de Dijkstra

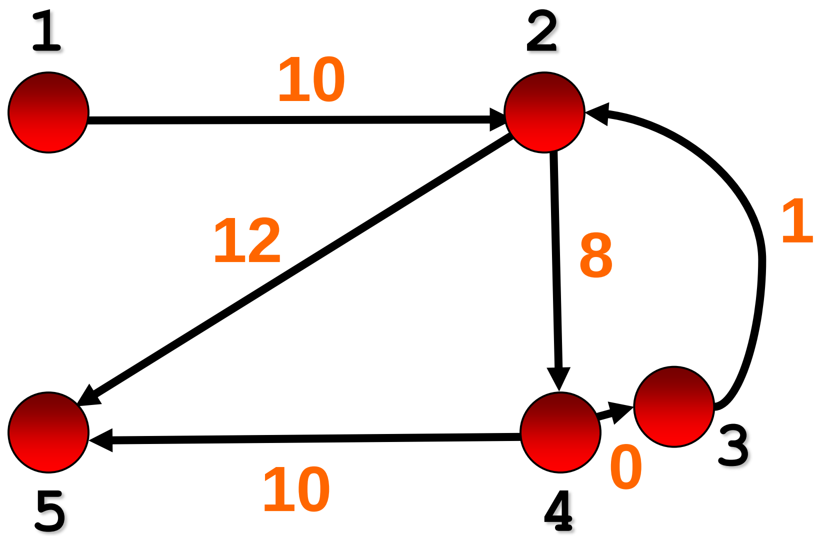
```
DIJKSTRA(G, w, s) /* Dados Grafo G, origem s e Lista de adjacência
    de s, w*/
    início
    //inicializa variáveis
    para cada vértice v faça
            d[v]=∞
            antecessor[v]=-1
    d[s]=0
    S=Ø
    cria fila de prioridade F com vértices do grafo
    enquanto F≠ Ø faça /*insere vértice u em S e faz relaxamento das
    arestas adjacentes*/
            u=retirar vértice de F, reorganizando F
            S=S+{u}
            para cada vértice v adjacente a u faça
                        relax(u,v,w)
    fim
```

Algoritmo de Dijkstra

- Complexidade de tempo: $O(|V| \log|V| + |E|)$, se fila de prioridade bem implementada
- Se não....?

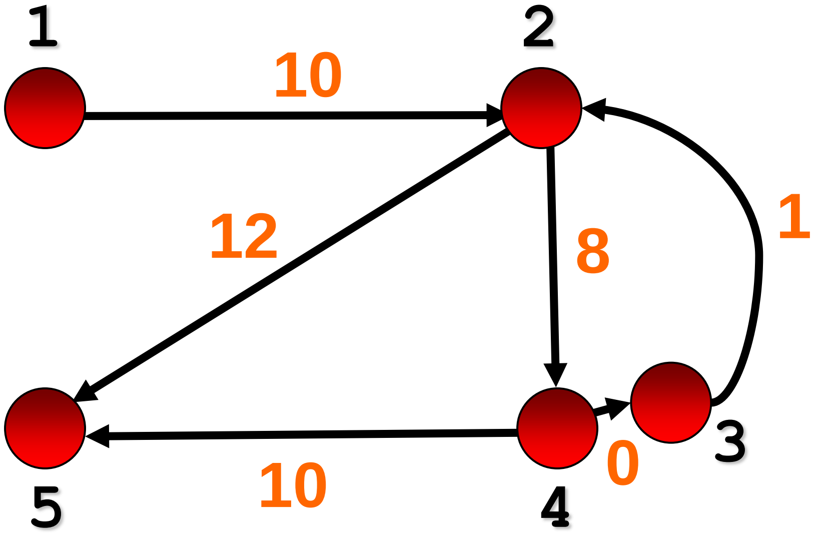
Exercícios

1. Calcule os custos - d(v) - dos caminhos mínimos para o grafo abaixo a partir do vértice 1 aplicando o algoritmo de Dijkstra



Exercícios

2. Quais são os próprios caminhos mínimos a partir do vértice 1?



Exercícios

3. Implemente o algoritmo de Dijkstra, usando o TAD Listas de Adjacência