Grafos – Aula 8

Roteiro

- Grafos com pesos
- Caminhos mínimos
- Dijkstra – a ideia
- Dijkstra – o algoritmo
- Dijkstra – o próprio

Diferenciando Relacionamentos

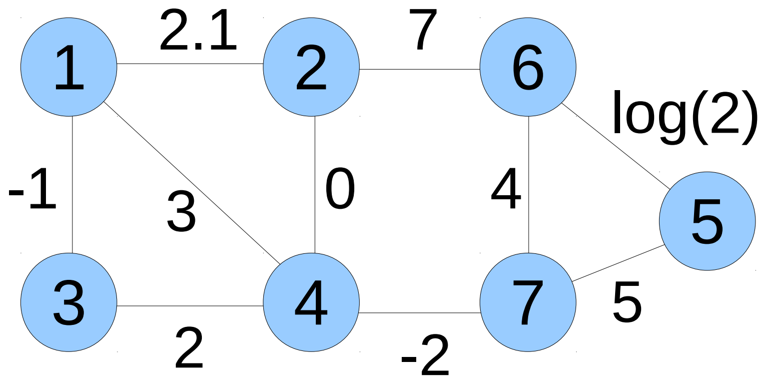
- Relacionamento entre objetos nem sempre são idênticos
- Amizade
  - mais ou menos amigo
- Distância física
  - perto ou longe
- Tempo de translado
  - mais ou menos tempo
- Conteúdo
  - mais ou menos parecido

Como representar tais relacionamentos?

Grafos com Pesos

- Anotar arestas do grafo com “intensidade” do relacionamento
  - peso da aresta (*weight*)
  - função w(*e*) retorna peso da aresta *e*
  - Ex. $w : E \rightarrow \Re$

- Graficamente



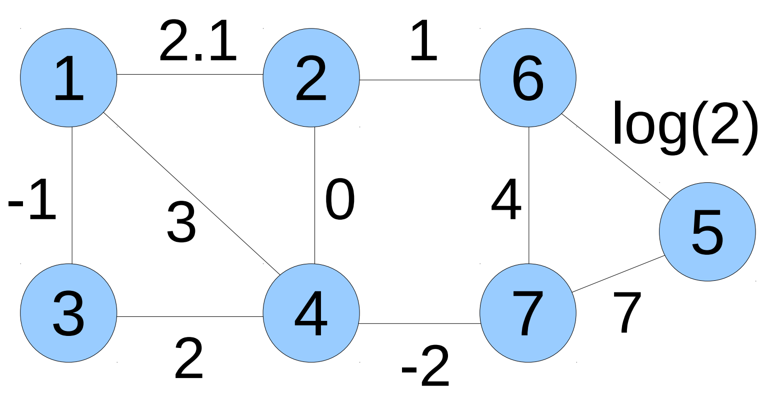
- w(2,6) = 7
- w(5,6) = log(2)
- w(2,4) = 0
- w(3,1) = -1

Comprimento com Pesos

- Comprimento de um caminho
  - **soma** dos pesos das arestas que definem caminho
- Caminho $p = (v_1, v_2, \ldots, v_k)$

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

- Exemplo



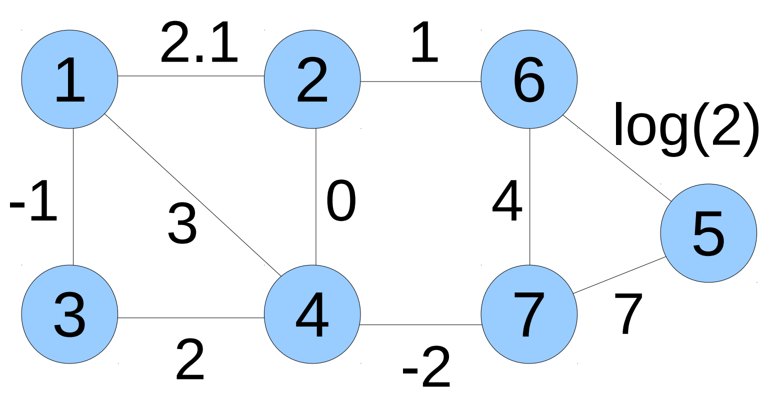
- $p = (1,2,6,7) \rightarrow C(p) = 7.1$
- $p = (1,3,4,7) \rightarrow C(p) = -1$
- $p = (2,4,7,6,5) \rightarrow C(p) = 2 + \log(2)$

Distância com Pesos

- Comprimento do **menor** caminho simples entre dois vértices
- *P*(*u*,*v*) : conjunto com todos os caminhos simples entre *u* e *v*

$$d(u,v) = min_{p \in P(u,v)} C(p)$$

- Exemplo



- d(4,1) = 1
- d(1,7) = -1
- d(3,5) = 2.1 + log(2)
- d(5,7) = log(2) - 1

- menor caminho não necessariamente é o mais curto em arestas!

Grafos Direcionados com Peso

- Relacionamentos assimétricos com pesos (diferentes intensidades)
  - ruas em uma cidade
  - similaridade entre seguidores do twitter

- Mesma ideia: arestas possuem “intensidade” do relacionamento
  - função w(*e*) retorna peso da aresta *e*
  - aresta direcionada, pesos potencialmente diferentes nas duas direções

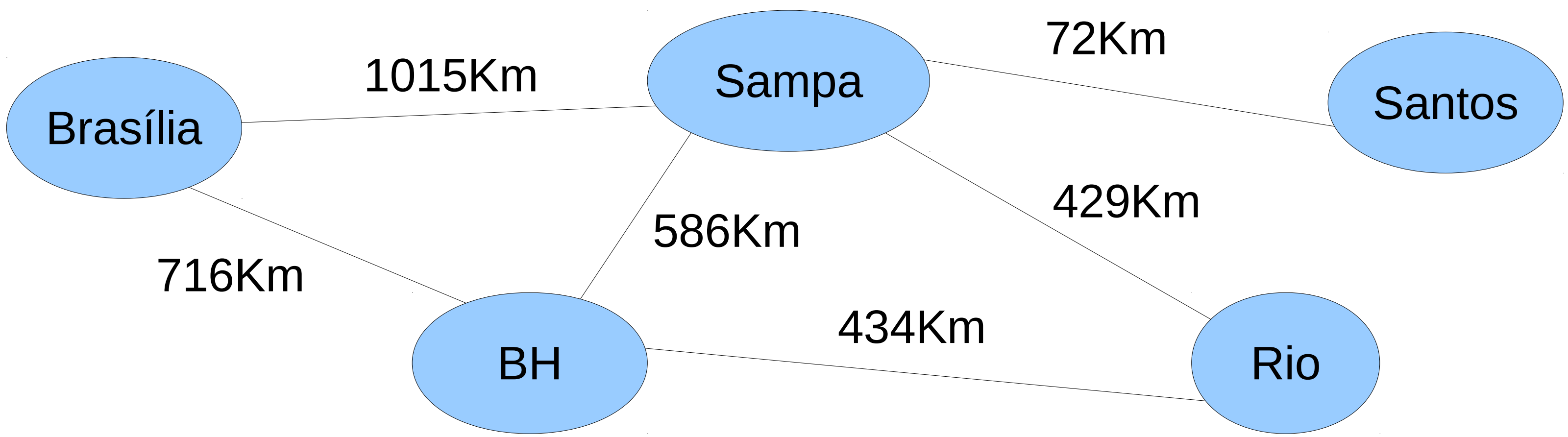
Viagem entre Cidades

- Cidades brasileiras
- Estradas entre cidades

- **Problema 1:** Como saber se duas cidades estão “conectadas” por estradas?
- **Problema 2:** Qual é o menor (melhor) caminho entre duas cidades?

Viagem entre Cidades

- Abstração via grafos com pesos



- **Problema 1:** Como ... des estão “cone... das?

Resolvido!

- **Problema 2:** Qual é o menor (melhor) caminho entre duas cidades?

Distância em Grafos com Peso

- Calcular caminho mais curto entre cidades é calcular a distância em grafos com peso
  - assumir pesos positivos

- Dado G, com pesos
- Determinar a distância do vértice *s* ao *d*

Como resolver este problema?

- Como resolvemos o problema sem pesos?
- Podemos adaptar algumas ideias?

Distância em Grafos com Peso

- **Ideia:** partindo de *s*, expandir os caminhos, incluindo vértices

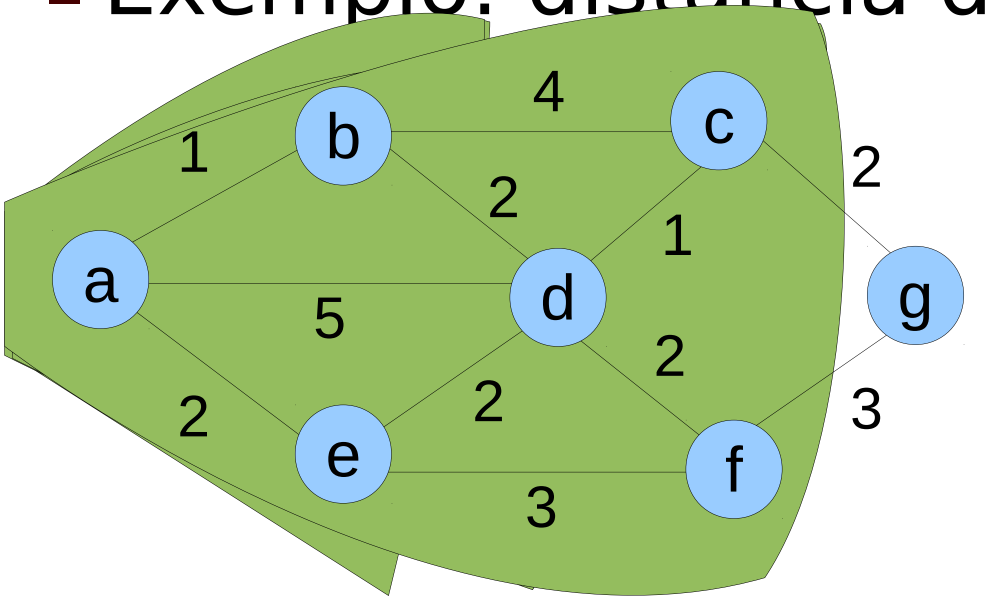
Mas em que ordem?

  - na ordem que garanta que iremos passar por caminhos mínimos!

- Expandir caminhos mínimos até chegar em *d* de maneira **gulosa**!

Distância em Grafos com Peso

- Exemplo: distância de *a* à *g*?



Algoritmo de Dijkstra!

- Começar em a, expandir
- Qual próximo vértice?
- Qual vértice nos dá caminho mínimo garantido?

Algoritmo de Dijkstra

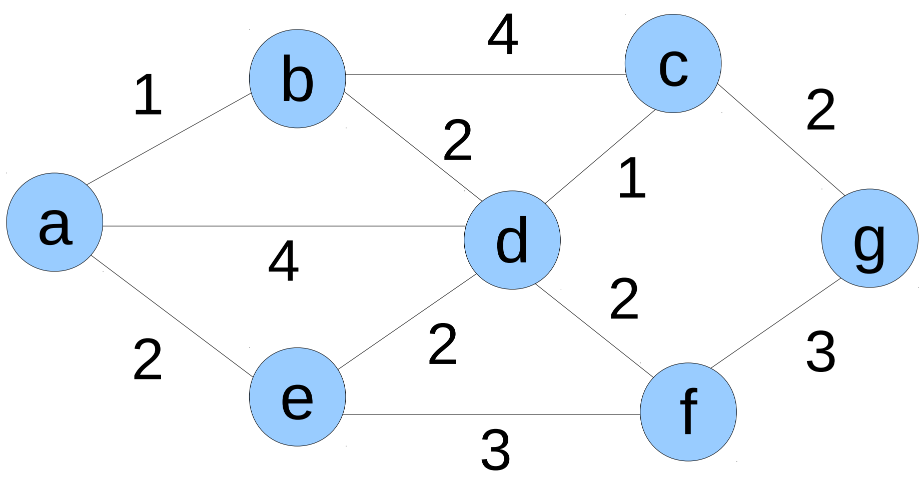
- Tornando a ideia em algoritmo
  - a cada passo, adicionar o vértice para o qual temos o menor caminho
- **Ideias:**
  - Manter dois conjuntos de vértices (descobertos, explorados)
  - Manter comprimento do menor caminho conhecido até o momento para cada vértice descoberto
  - Adicionar o vértice de menor caminho ao conjunto explorado
  - Atualizar distâncias através deste vértice, descobrindo novos vértices

Algoritmo de Dijkstra

```
Dijkstra(G, s)
  Para cada vértice v
    dist[v] = infinito
  Define conjunto S = 0 // inicia vazio
  dist[s] = 0
  Enquanto S != V
    Selecione u em V-S, tal que dist[u] é mínima
    Adicione u em S
    Para cada vizinho v de u faça
      Se dist[v] > dist[u] + w(u,v) então
        dist[v] = dist[u] + w(u,v)
Retorna dist[]
```

- S é o conjunto dos vértices explorados
- V é o conjunto dos vértices do grafo
- w(u,v) é o peso da aresta (u,v)
- dist[v] é a melhor estimativa da distância de s a v
- se v é explorado, então dist[v] é a distância de s a v

Executando o Algoritmo



- Tabela indica passos do algoritmo, atualização do vetor de distâncias e do conjunto S

vetor dist[ ] é d[ ] na tabela abaixo

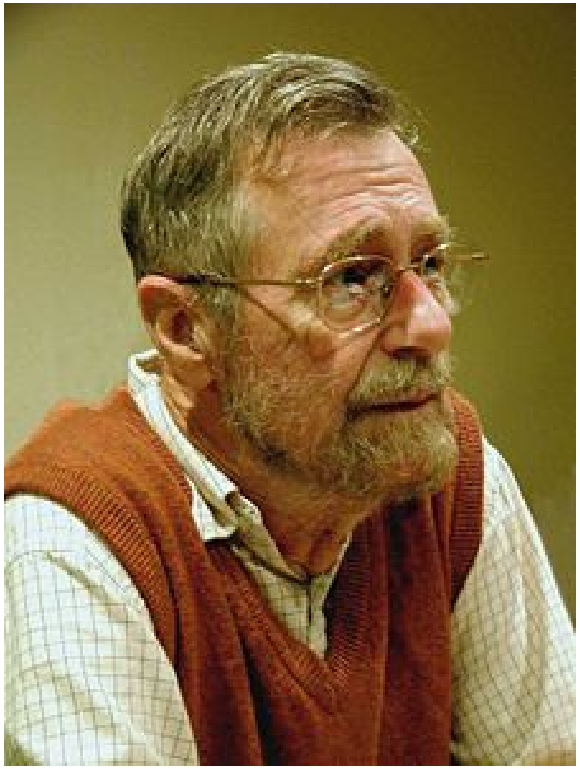
Passo	Conjunto S	d[a]	d[b]	d[c]	d[d]	d[e]	d[f]	d[g]
0	{}	0	inf	inf	inf	inf	inf	inf
1	{a}	-	1	inf	4	2	inf	inf
2	{a,b}		-	5	3	2	inf	inf
3	{a,b,e}			5	3	-	5	inf
4	{a,b,e,d}			4	-		5	inf
5	{a,b,e,d,c}			-			5	6
6	{a,b,e,d,c,f}						-	6
7	{a,b,e,d,c,f,g}							-

Complexidade ?

```
1. Dijkstra(G, s)
2.   Para cada vértice v
3.     dist[v] = infinito
4.   Define conjunto S = 0 // inicia vazio
5.   dist[s] = 0
6.   Enquanto S != V
7.      Selecione u em V-S, tal que dist[u] é mínima
8.      Adicione u em S
9.      Para cada vizinho v de u faça
10.       Se dist[v] > dist[u] + w(u,v) então
11.          dist[v] = dist[u] + w(u,v)
12.Retorna dist[]
```

- Percorre todas arestas do grafo (linha 6 + linha 9)
- Depende do tempo para escolher u (linha 7)
  - percorrer vetor e obter menor tem custo O(n)
- Complexidade $O(n^2)$
  - aula que vem seremos mais eficientes

Dijkstra, o Próprio



11/5/1930 – 6/8/2002

- Edsger Wybe Dijkstra
- Renomado professor e pesquisador em Computação
- Recebeu Prêmio Turing em 1972
- Contribuições fundamentais em ling. de programação e verificação formal
- Algoritmo de Dijkstra utilizado em vários sistemas (redes, GPS, etc)
- Documentário: *Discipline in Thought* (2000)
  - http://www.cs.utexas.edu/users/EWD/video-audio/NoorderlichtVideo.html

Discipline in Thought

- “Ciência da computação não é mais sobre computadores do que astronomia é sobre telescópios”
- “Falta coragem nas universidades para ensinar ciências duras. Elas continuarão a enganar os alunos e cada etapa na infantilização do currículo será celebrada como progresso educacional”
- “Elegância (*na arte de programar*) não é um luxo dispensável, mas fator que decide entre sucesso e falha”
- “Elegância demanda trabalho duro para produzir e uma boa educação para apreciar. Por isto encontrou poucos adeptos”

Quais são suas impressões?