Algoritmo de Dijkstra

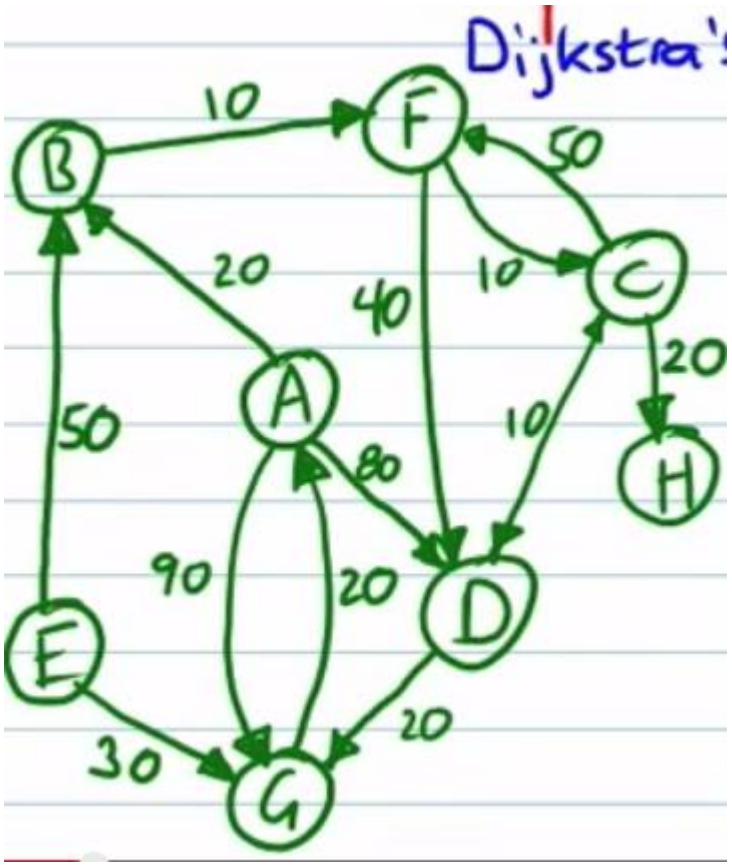
Caminhos mínimos em Grafos

- Considere um grafo orientado ponderado $G = (V,E)$ em que cada aresta possui um rótulo não negativo associado que define o custo da aresta, e um dos vértices é especificado como *origem*.
- Nosso problema é determinar quais são os caminhos mais curtos do vértice origem para cada um dos demais vértices em V e os seus custos.
- O caminho ***mais curto*** ou ***mínimo*** é definido pelo o caminho cuja soma dos custos dos vértices encontrados no caminho é mínimo.

Slide: Gustavo E.A.P.A. Batista - http://www.lcad.icmc.usp.br/~jbatista/sce183/mat/Didatica.pdf

Grafo direcionado – (arestas com ‘setas’) (**digrafo**)

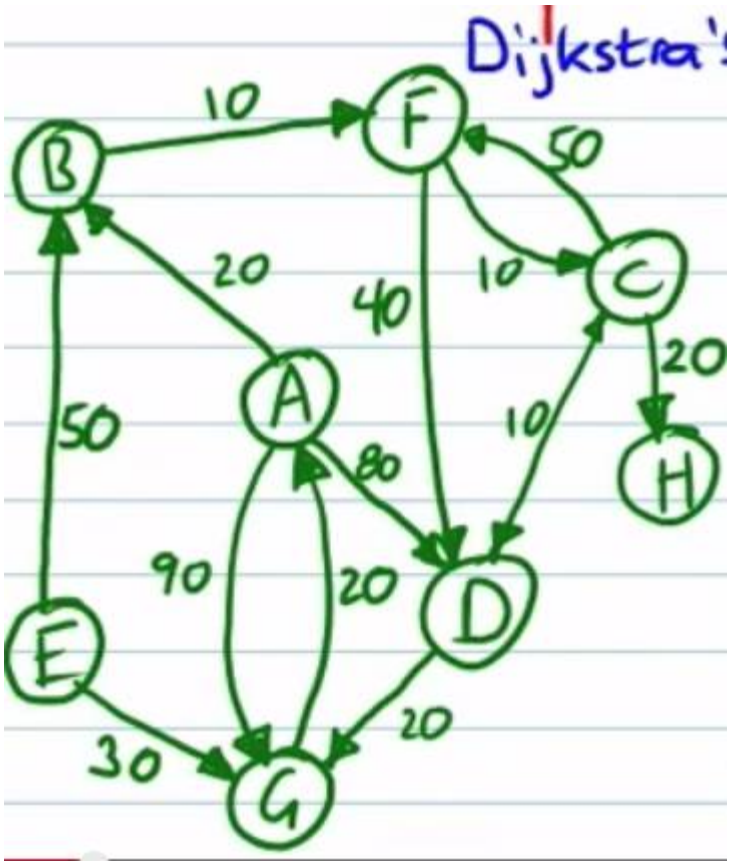
Grafo ponderado – grafos com pesos (arestas com ‘números’)



https://www.youtube.com/watch?v=8Ls1RqHCOPw

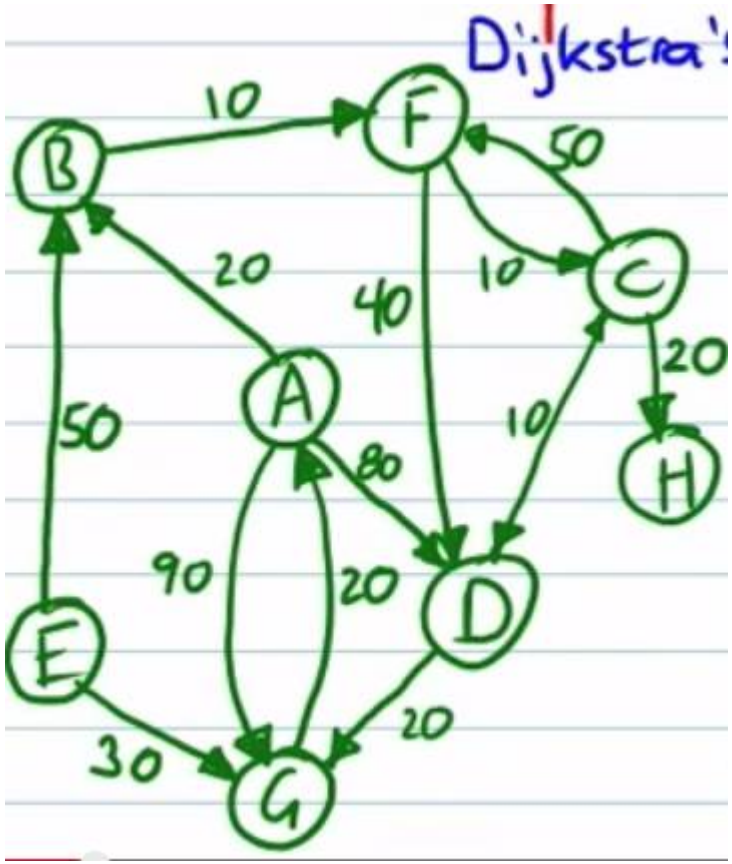
Algortimo de Dijkstra

- O algoritmo de **Dijkstra** permite encontrar o menor caminho entre um nó origem (ex. **A**) e todos os outros vértices do grafo



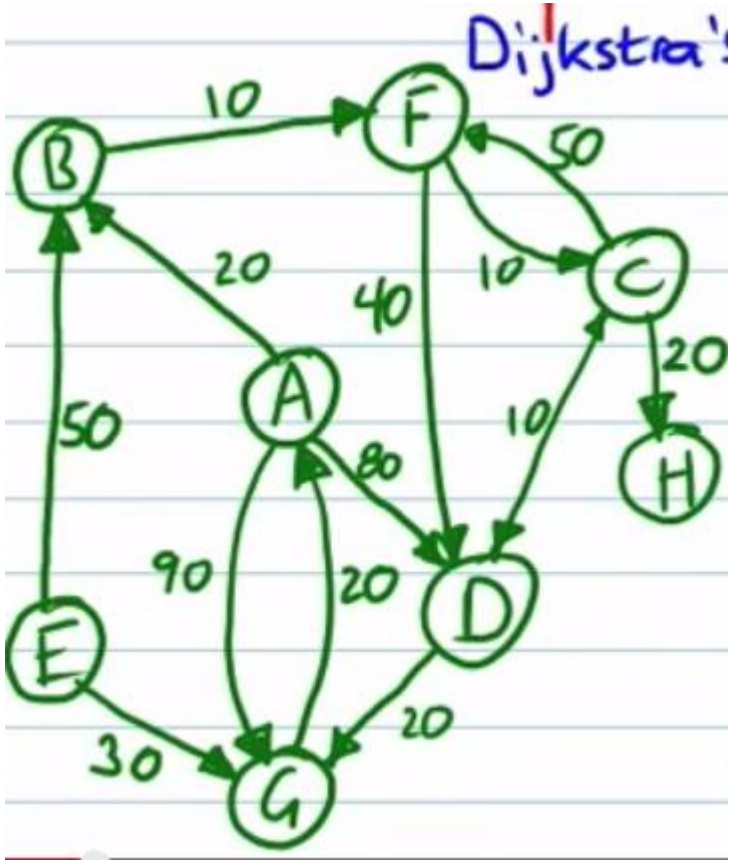
Dijkstra

- Inicia em A, e verifica as saídas de A
  - De A podemos ir para
    - B
    - D
    - G
  - Estabelecemos o custo olhando o peso das

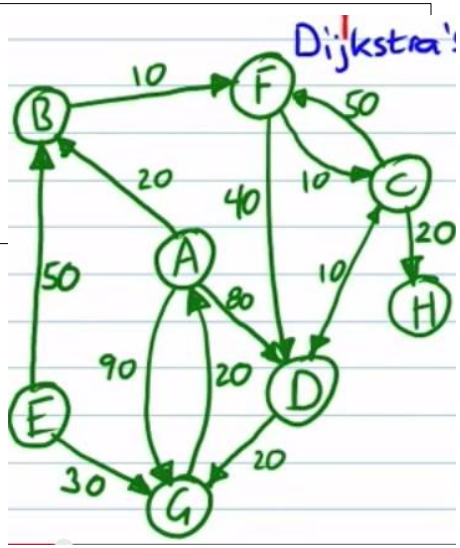


Dijkstra

- Custos
  - De A podemos ir para
    - B = 20
    - D = 80
    - G = 90
  - Outros nós:
    - Custo infinito (não é possível acessar)

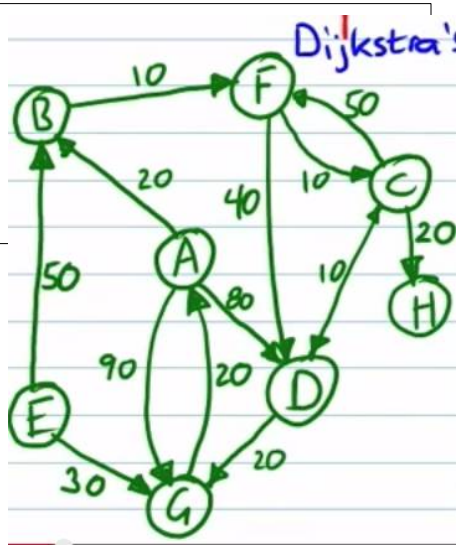


Dijkstra



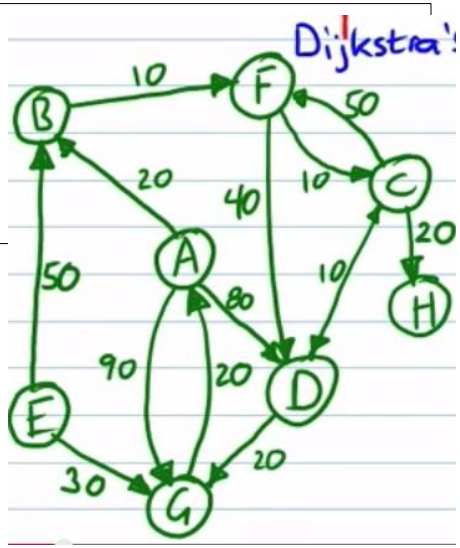
de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf

Qual o menor custo?



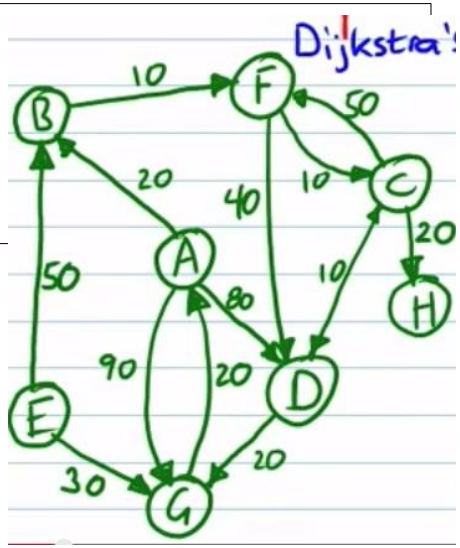
de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf

Qual o menor custo?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf

Qual o menor custo?



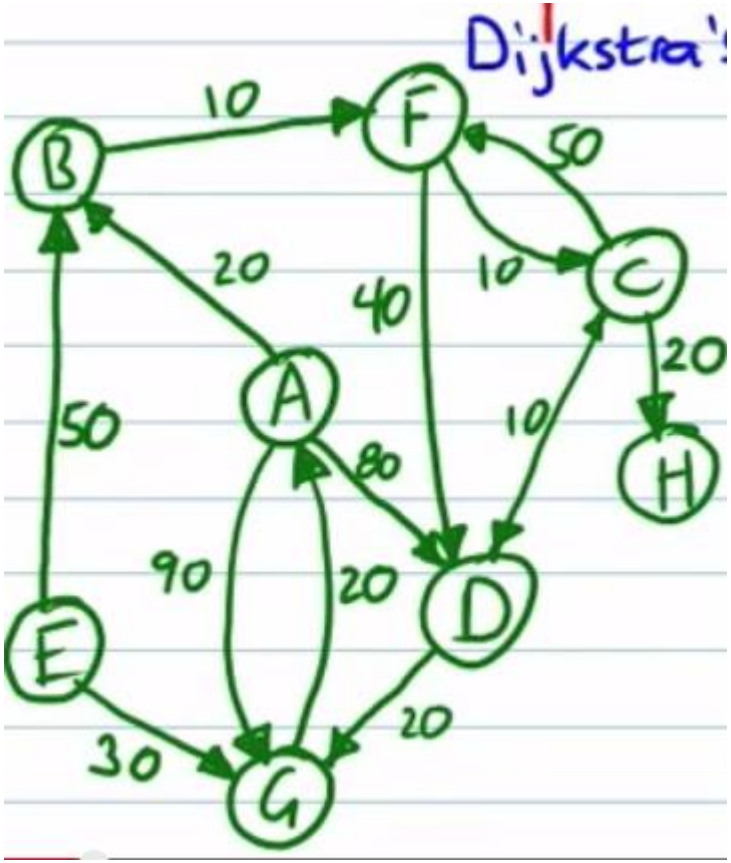
de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf

Já sabemos então que o menor caminho entre A e B é de custo 20, e é formado pelo seguinte caminho: A,B

- Como já sabemos que o menor caminho saindo de A é ir para B, vamos no próximo passo analisar o menor caminho entre B e os outros nós da rede.

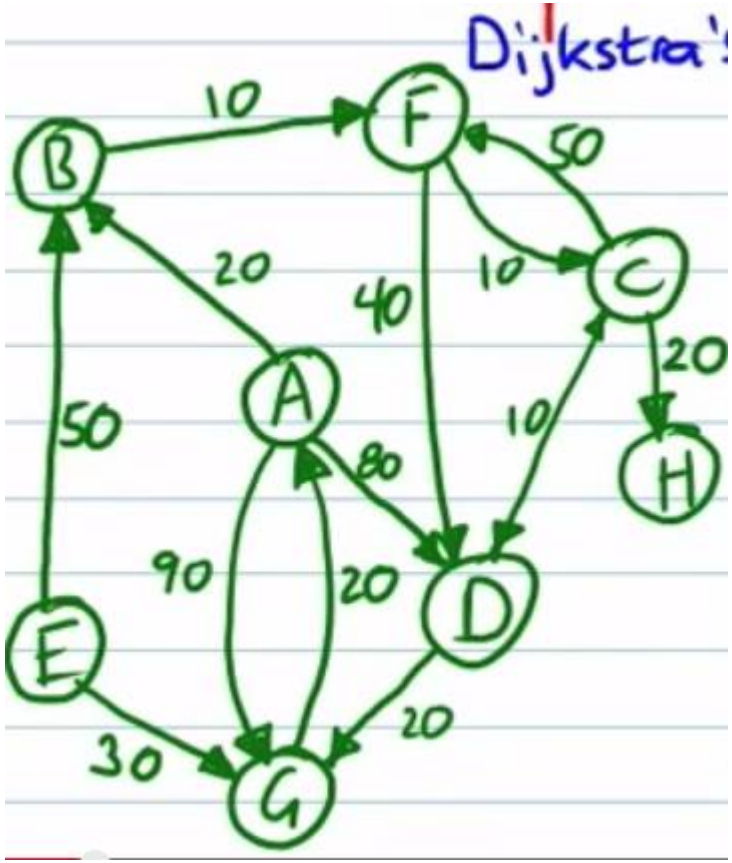
Dijkstra

- Inicia em B, e verifica as saídas de B
  - De B podemos ir para
    - F
  - Estabelecemos o custo olhando o peso das arestas



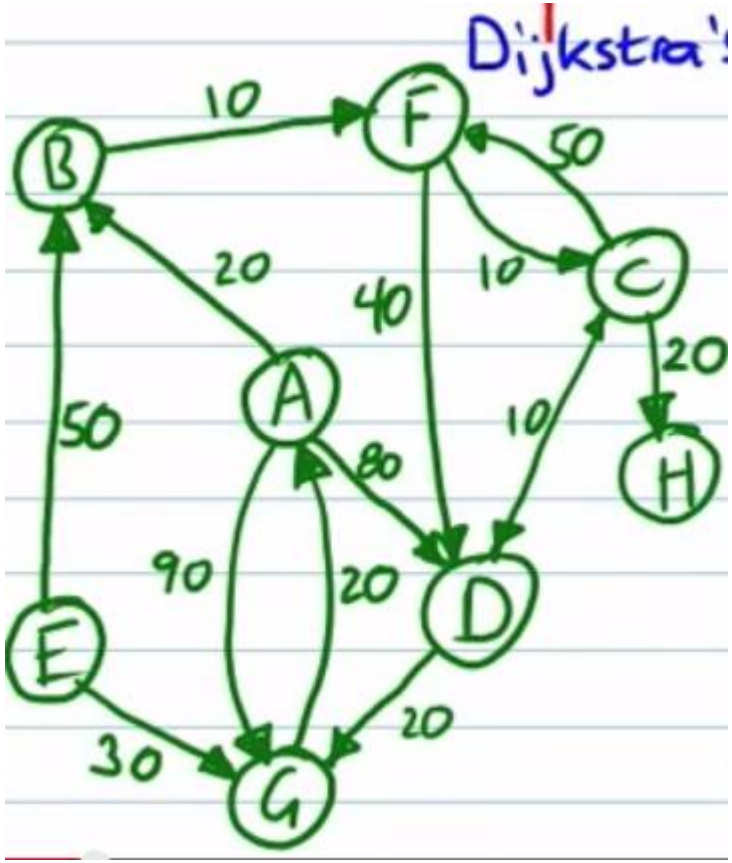
Dijkstra

- Custos
  - De B podemos ir para
    - F = 10
  - Outros nós:
    - Custo infinito (não é possível acessar)

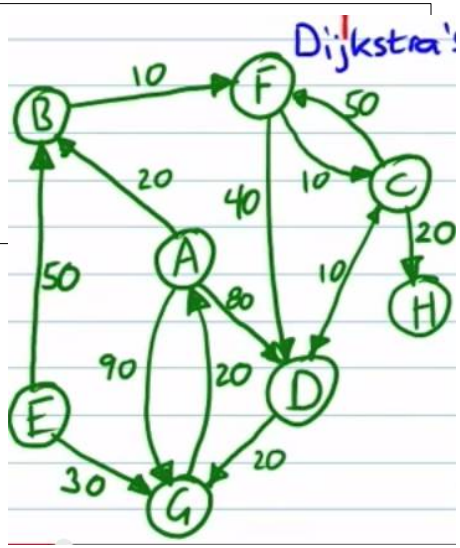


Dijkstra

- Custos
  - De B podemos ir para
    - F = 10
  - O custo para chegar em F a partir de A era infinito. Agora podemos chegar em F por meio de B
  - O custo para chegar em F é então somado ao custo para chegar em B

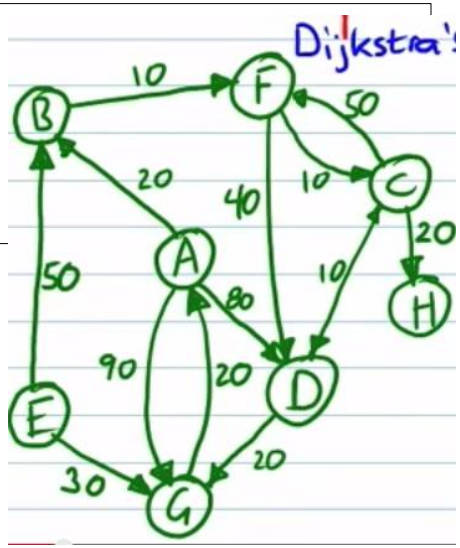


Dijkstra



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B					20+10 de: B		

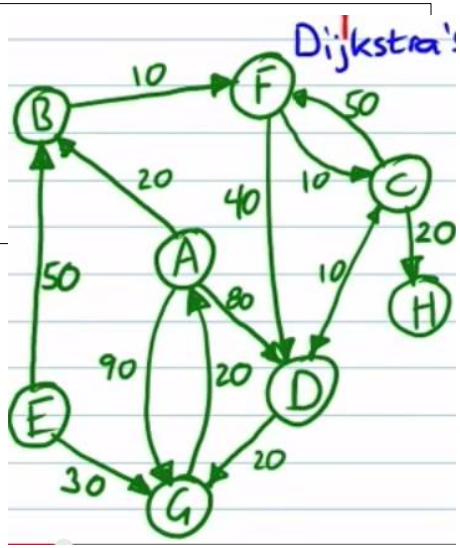
Dijkstra



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B					30 de: B		

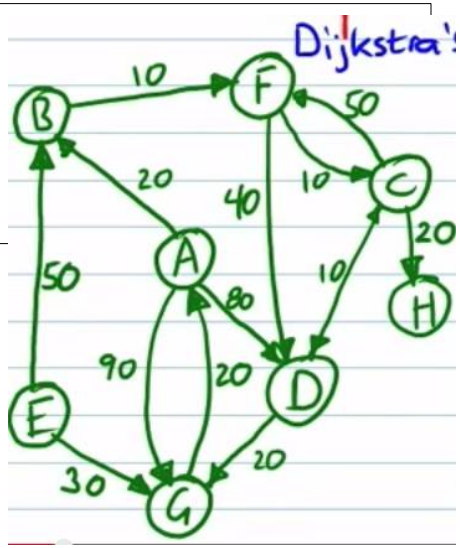
- E o custo dos outros nós não acessíveis por B?
  - Nada muda, copiamos então o custo anterior
  - Exemplo:
    - Custo para ir de A até E: inf – mantém infinito
    - Custo para ir de A até G: 90 – mantém 90

Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf

Dijkstra

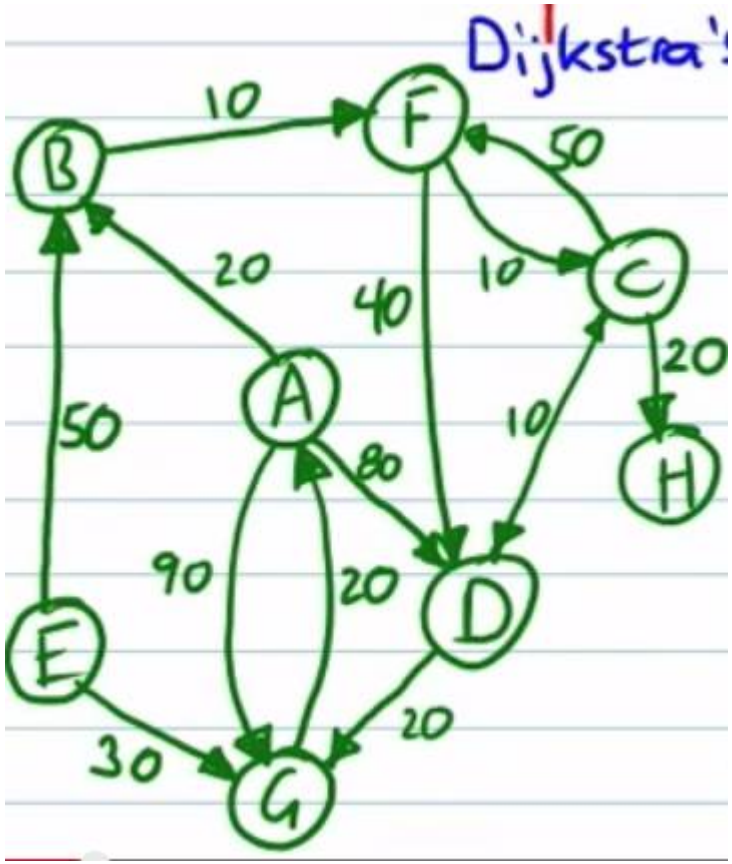


de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf

Já sabemos então que o menor caminho entre A e F é de custo 30, e é formado pelo seguinte caminho: A,B,F

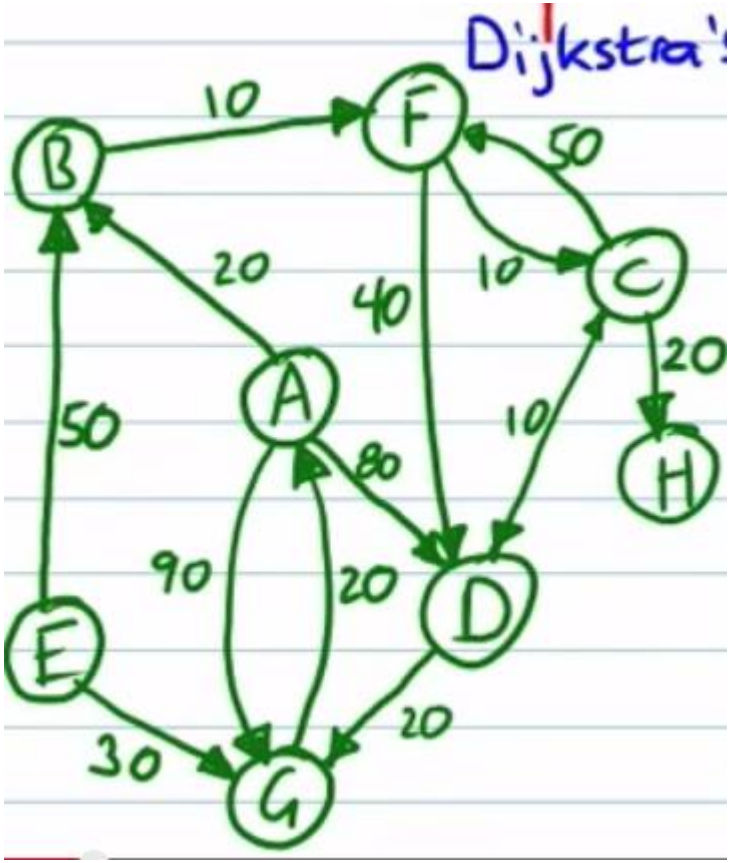
Dijkstra

- Inicia em F, e verifica as saídas de F
  - De F podemos ir para
    - C
    - D
  - Estabelecemos o custo olhando o peso das arestas



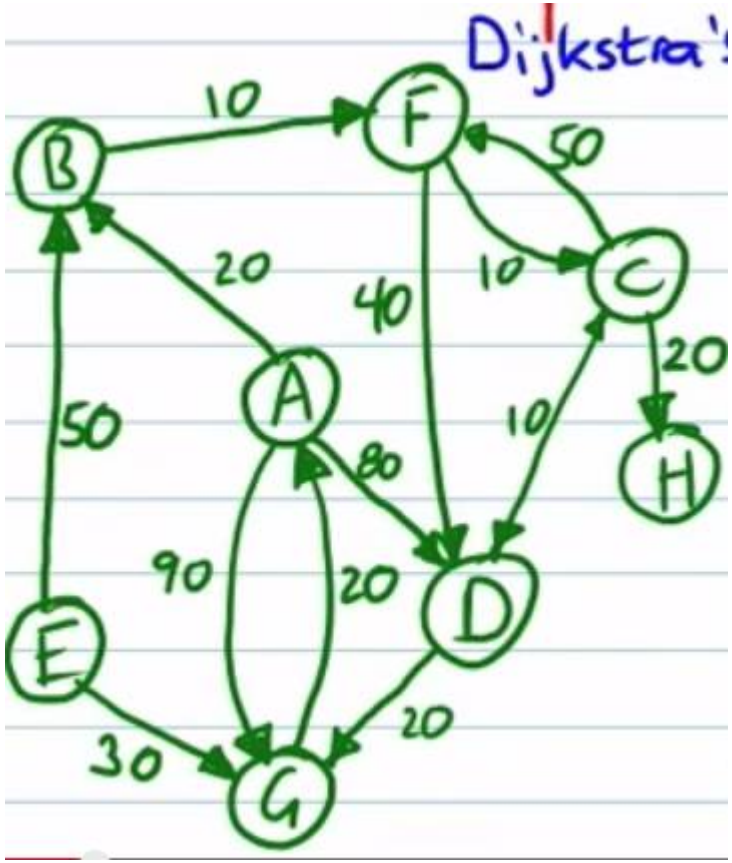
Dijkstra

- Inicia em F, e verifica as saídas de F
  - De F podemos ir para
    - C = 10
    - D = 40
  - Outros nós:
    - Custo infinito (não é possível acessar)

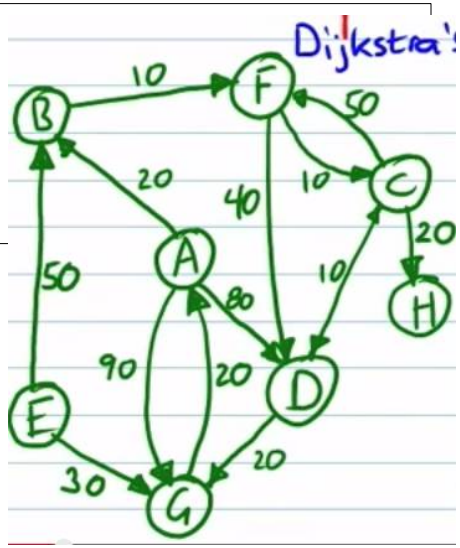


Dijkstra

- Custos
  - De F podemos ir para
    - C = 10
  - O custo para chegar em C a partir de A era infinito. Agora podemos chegar em C por meio de F
  - O custo para chegar em C é então somado ao custo para chegar em F

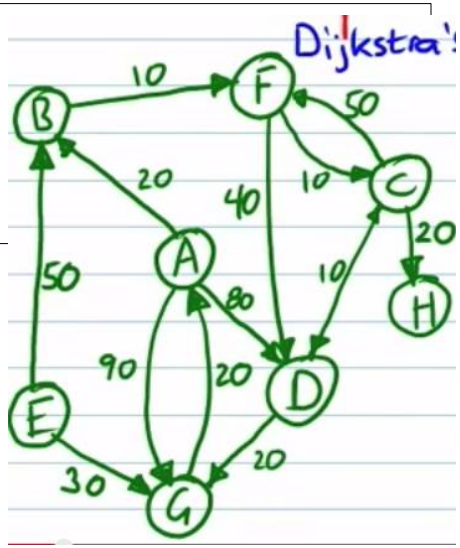


Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		30+10 de: F					

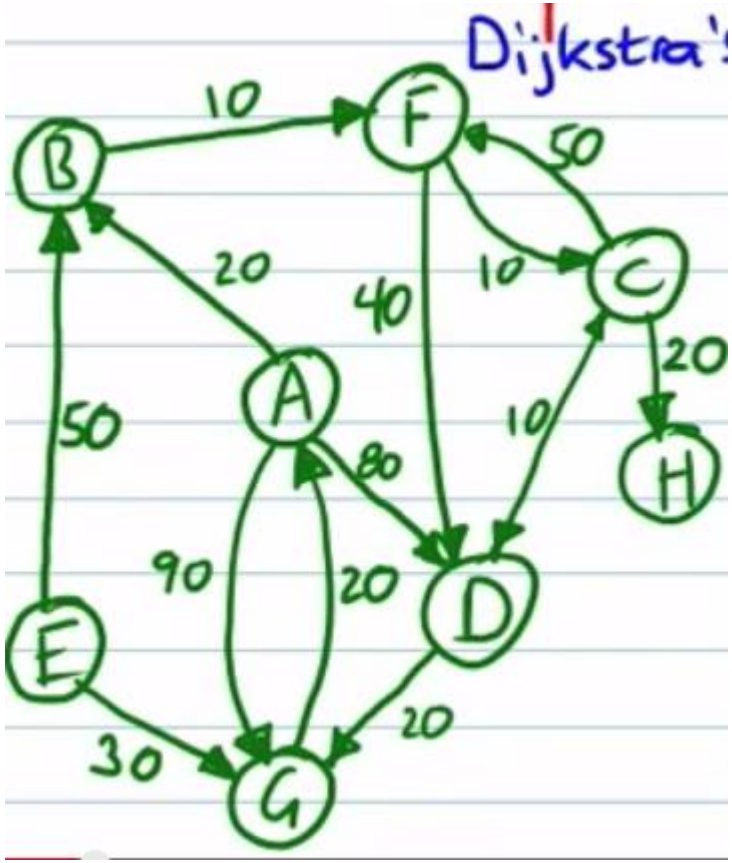
Qual o menor caminho?



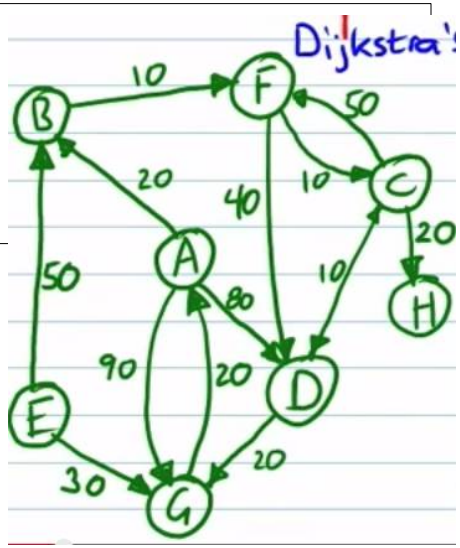
de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F					

Dijkstra

- Custos
  - De F podemos ir para
    - D = 40
  - O custo para chegar em D a partir de A era **80**. Agora podemos chegar em D por meio de F
  - O custo para chegar em D é então somado ao custo para chegar em F

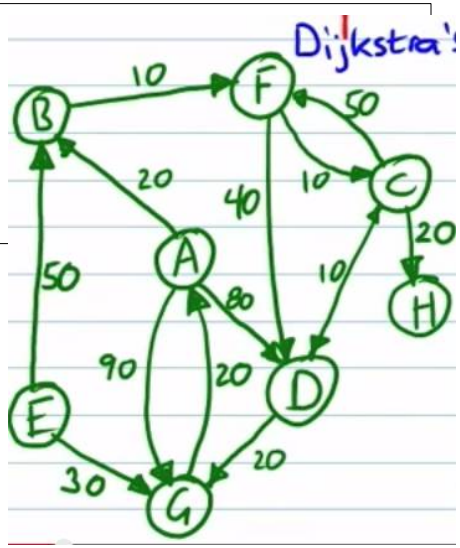


Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	30+40 de: F				

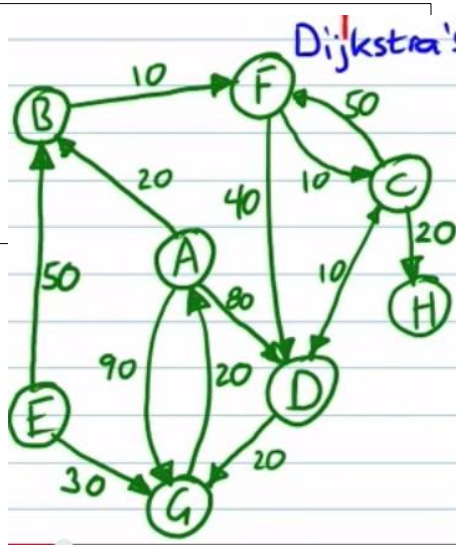
Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F				

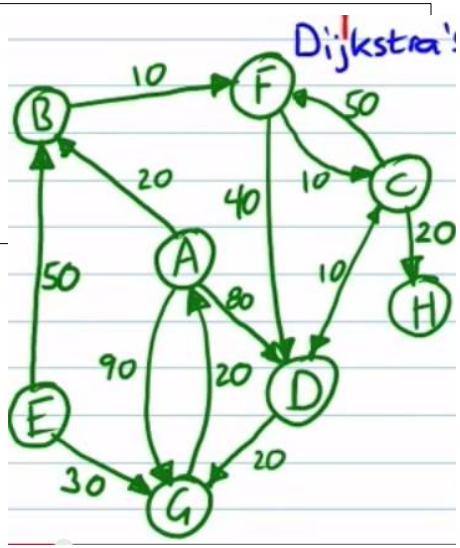
- Antes, para sair de A e chegar em D o custo era 80
- Agora, para sair de A e chegar em D, via F, o custo é 70. Então devemos manter este como sendo o menor caminho entre A e D
- Não há mudanças nos outros caminhos de A

Copiando os valores



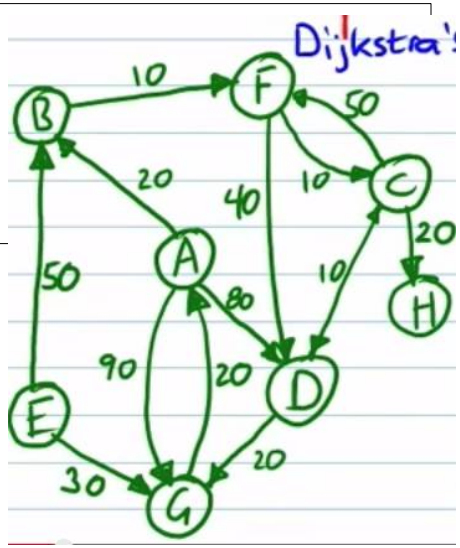
de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf

Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf

Qual o menor caminho?

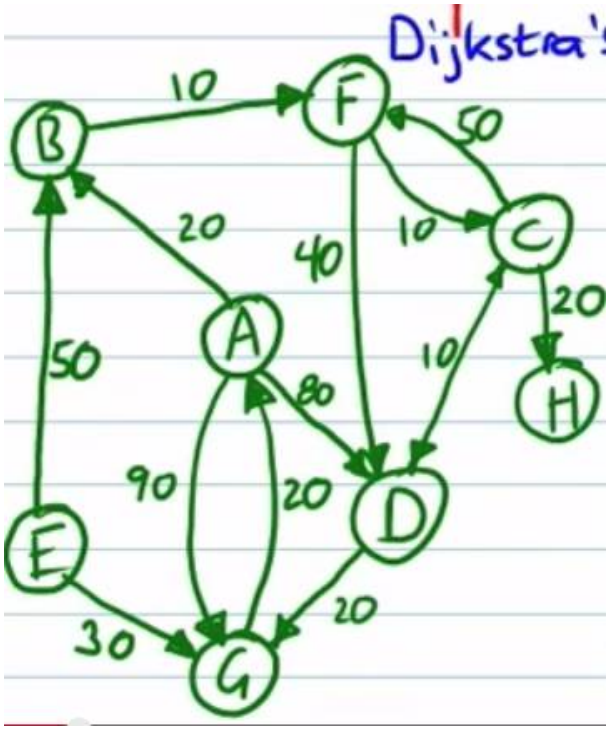


de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf

Já sabemos então que o menor caminho entre A e C é de custo 40, e é formado pelo seguinte caminho: A,B,F,C

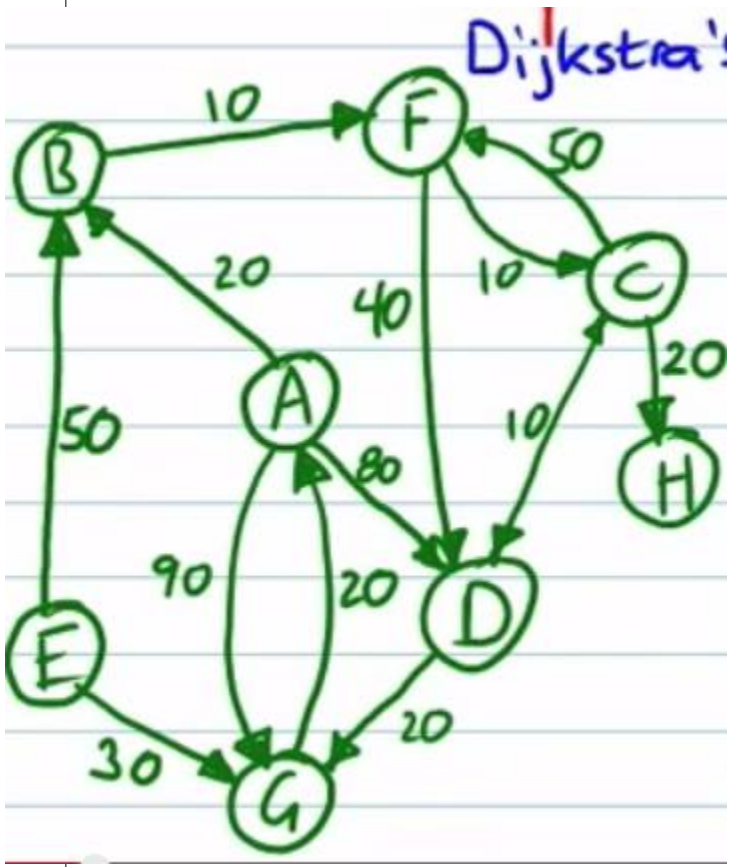
Dijkstra

- Inicia em C, e verifica as saídas de C
  - De C podemos ir para
    - H
    - D
    - F
  - Estabelecemos o custo olhando o peso das



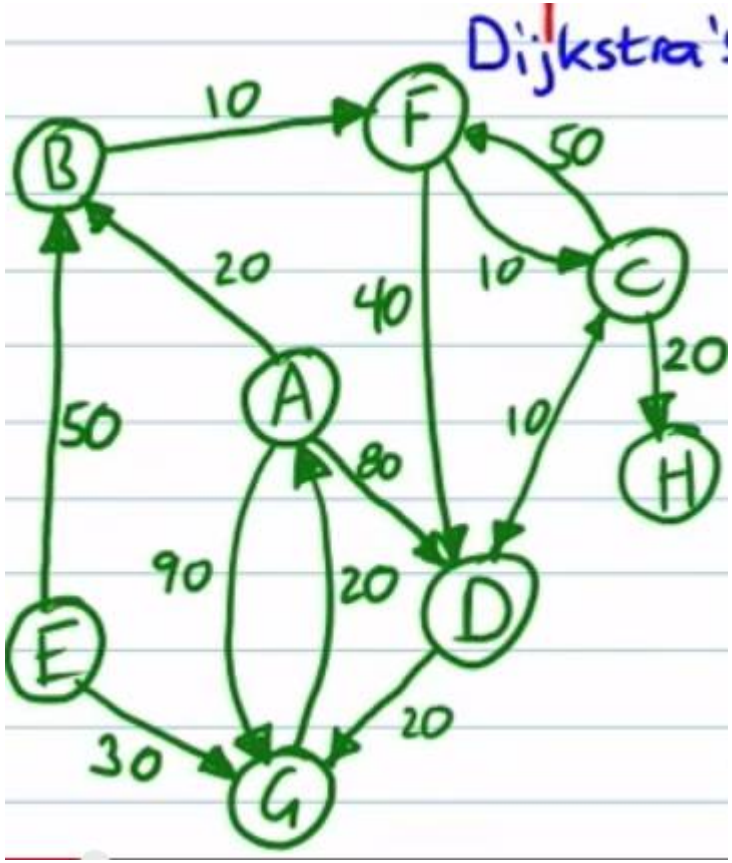
Dijkstra

- Custos
  - De C podemos ir para
    - H = 20
    - D = 10
    - F = 50

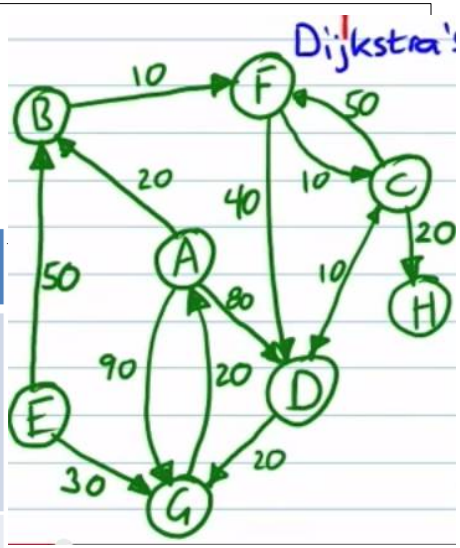


Dijkstra

- Custos
  - De C podemos ir para
    - H = 20
  - O custo para chegar em H a partir de A era infinito. Agora podemos chegar em H por meio de C
  - O custo para chegar em H é então somado ao custo para chegar em C

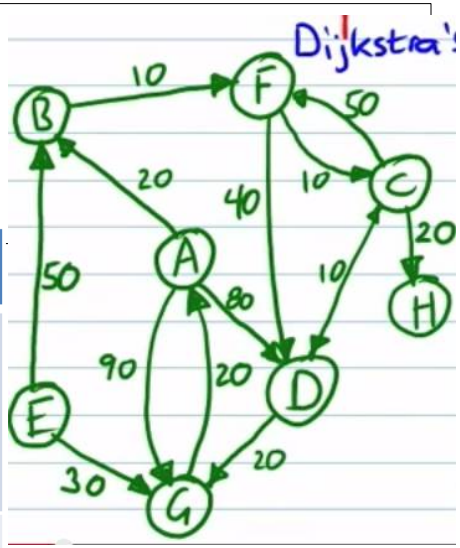


Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C							40+20 de:C

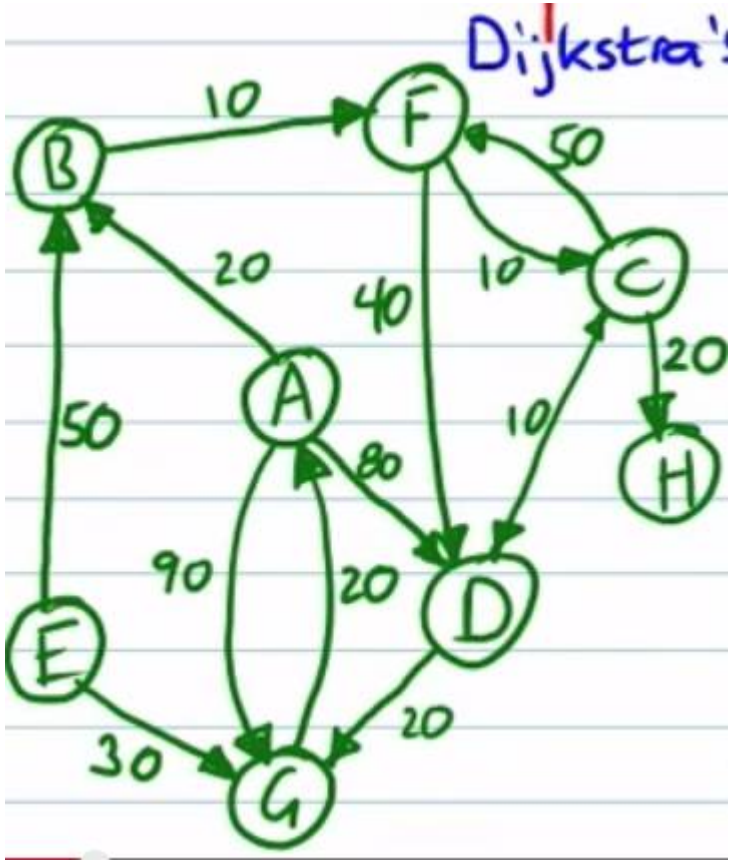
Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C							60 de:C

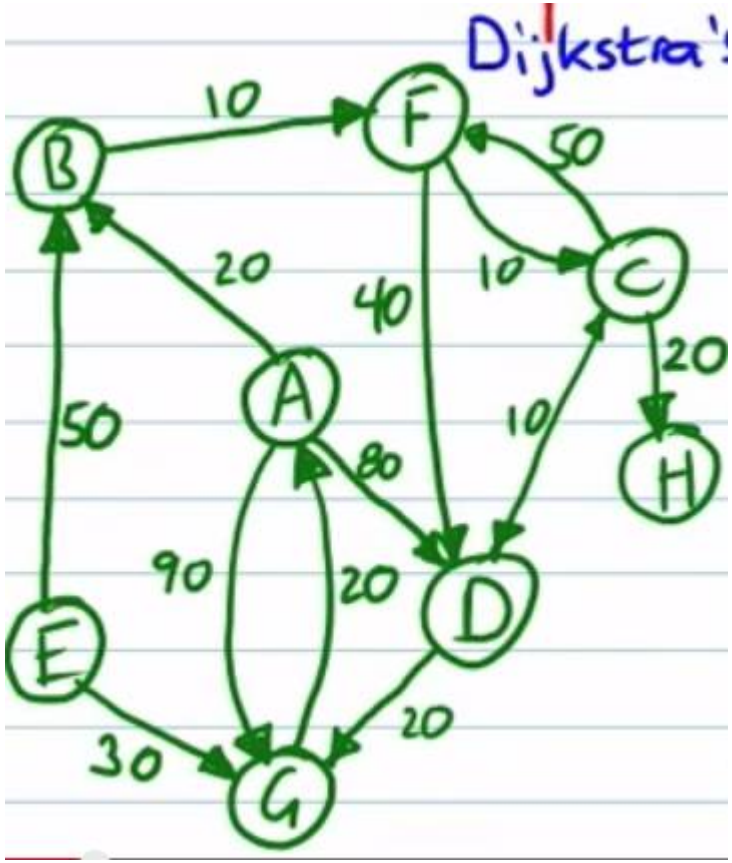
Dijkstra

- Custos
  - De C podemos ir para
    - H = 20
    - D = 10
    - F = 50

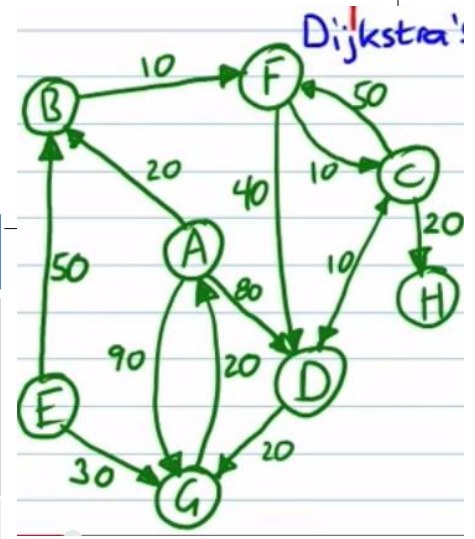


Dijkstra

- Custos
  - De C podemos ir para
    - D = 10
  - O custo para chegar em D a partir de A era **70**. Agora podemos chegar em D por meio de C
  - O custo para chegar em D é então somado ao custo para chegar em C

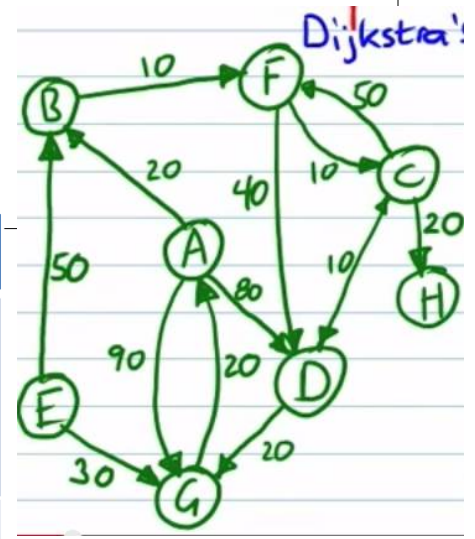


Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C							60 de:C

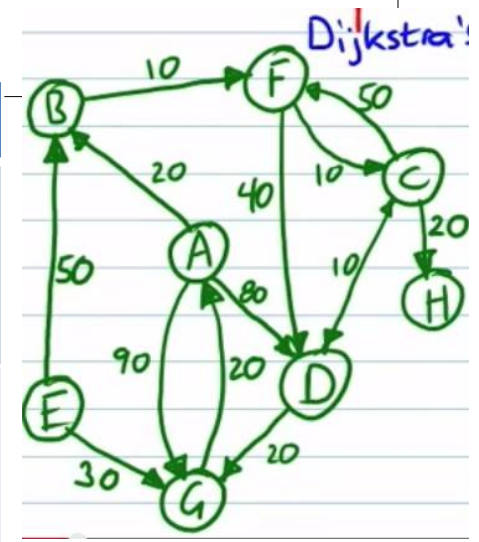
Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			40+10 de: C				60 de:C

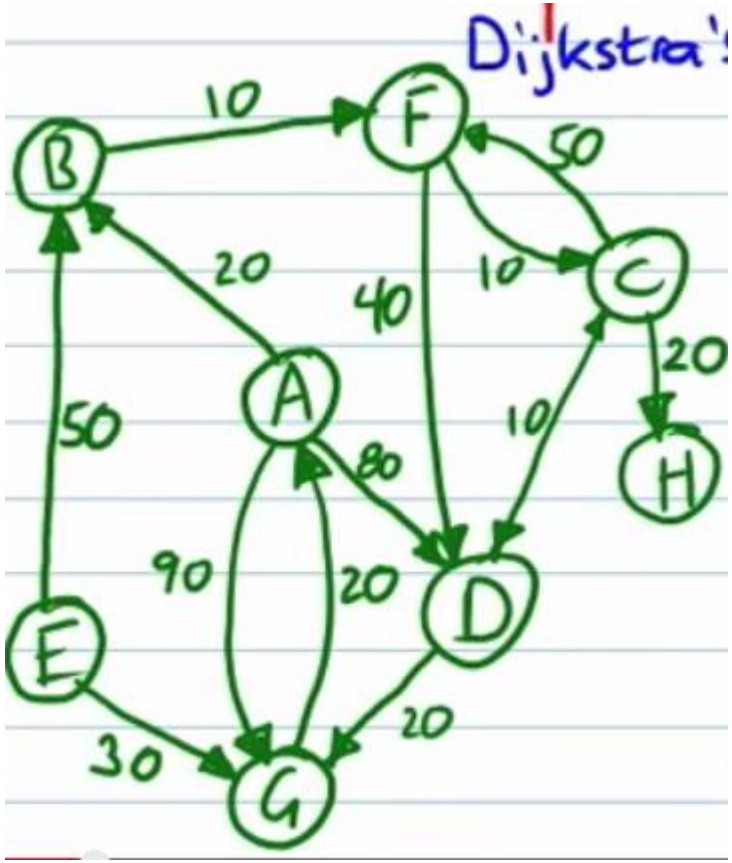
Qual o menor caminho?

de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C				60 de:C



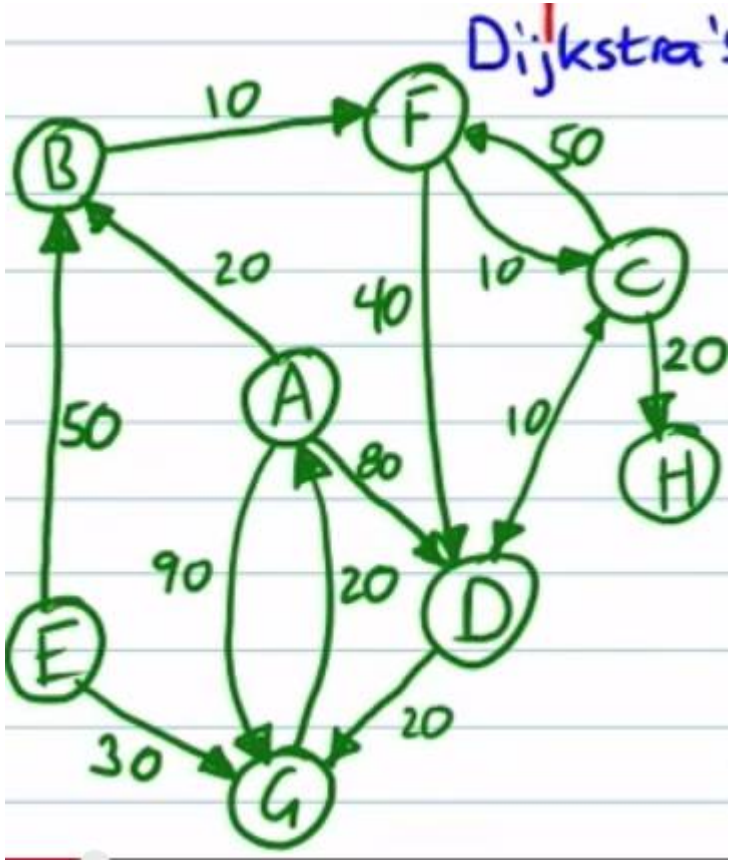
Dijkstra

- Custos
  - De C podemos ir para
    - H = 20
    - D = 10
    - F = 50

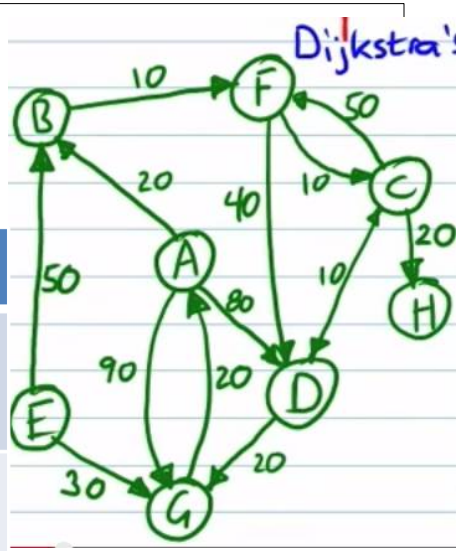


Dijkstra

- Custos
  - De C podemos ir para
    - F = 50
  - O custo para chegar em F a partir de A era **30**. Agora podemos chegar em F por meio de C
  - O custo para chegar em F é então somado ao custo para chegar em C

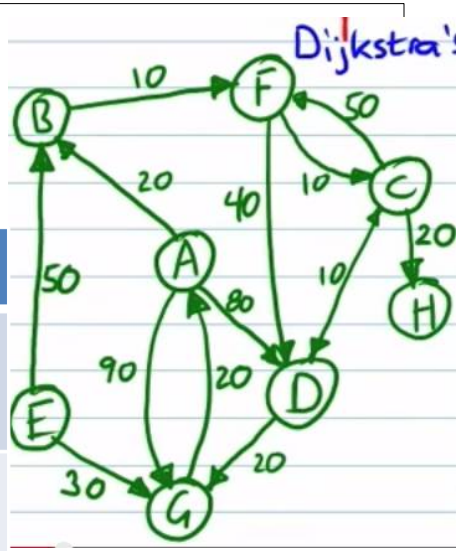


Qual o menor caminho?



de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf	40+50 de: C	90 de: A	60 de:C

Qual o menor caminho?

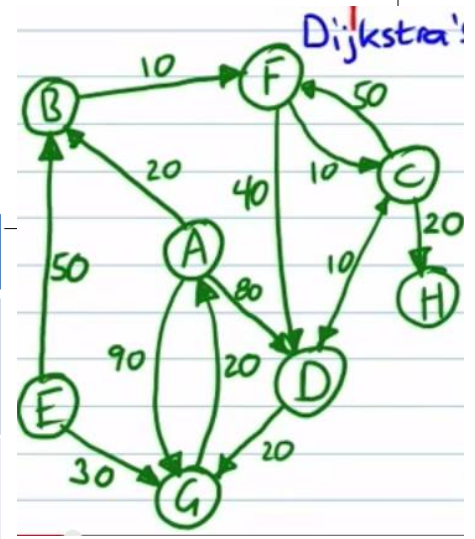


de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf	~~90 de: €~~	90 de: A	60 de:C

Com certeza o custo vindo de C será maior, pois já tínhamos determinado o menor caminho até F

Na verdade, F nem seria verificado por já temos calculado o caminho mínimo

Qual o menor caminho?

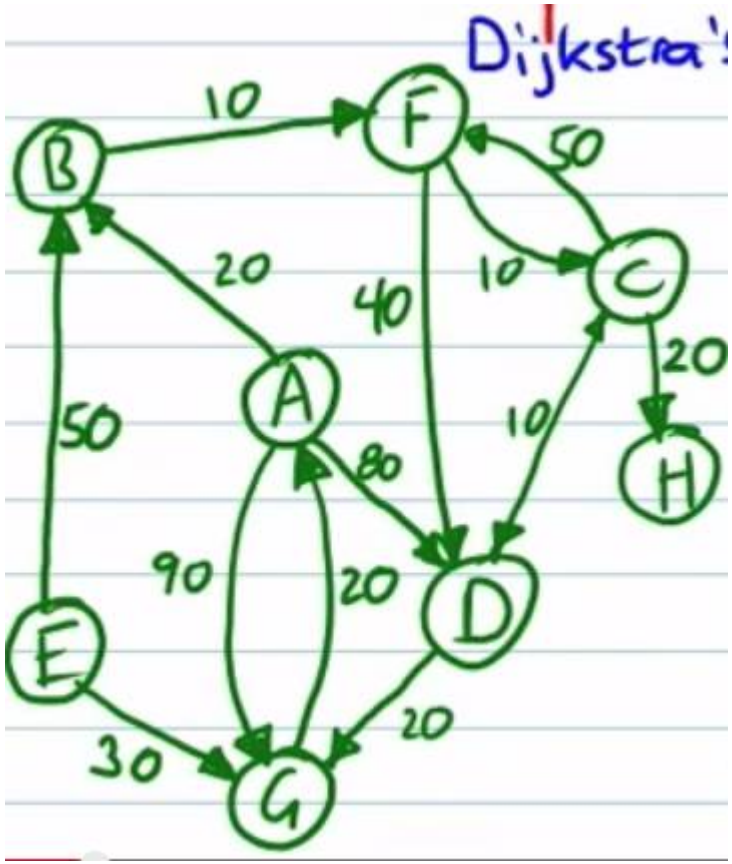


de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf		90 de: A	60 de:C

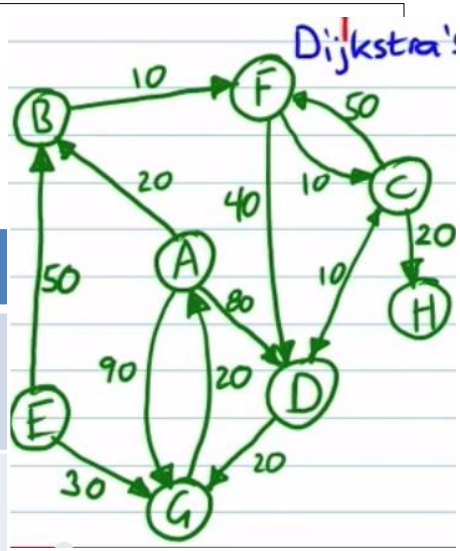
Já sabemos então que o menor caminho entre A e D é de custo 50, e é formado pelo seguinte caminho: A,B,F,C,D

Dijkstra

- Inicia em D, e verifica as saídas de D
  - De D podemos ir para
    - G
    - C já foi determinado o menor caminho
  - Estabelecemos o custo olhando o peso das arestas



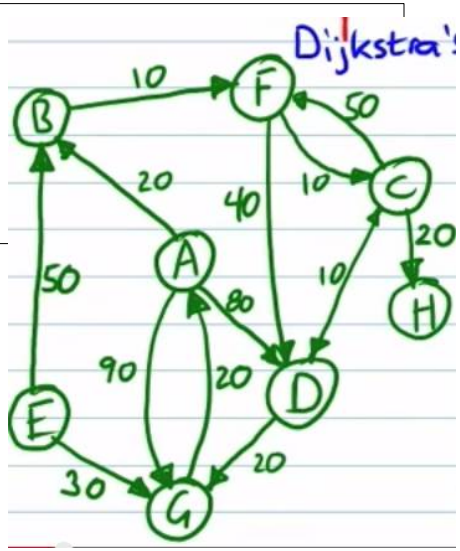
Qual o menor caminho?



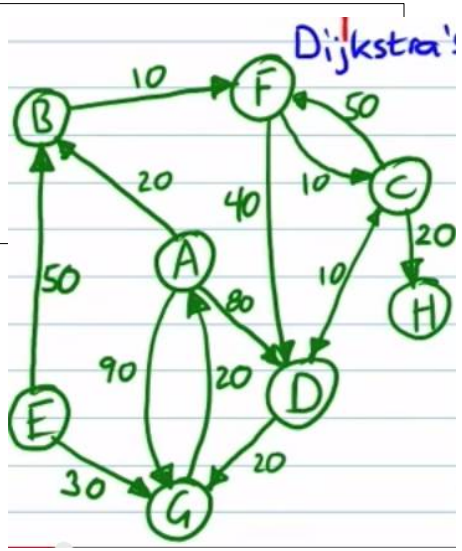
de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf		90 de: A	60 de:C
D						50+20 de: D	

Qual o menor caminho?

de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf		90 de: A	60 de: C
D				inf		70 de: D	60 de: C



Qual o menor caminho?

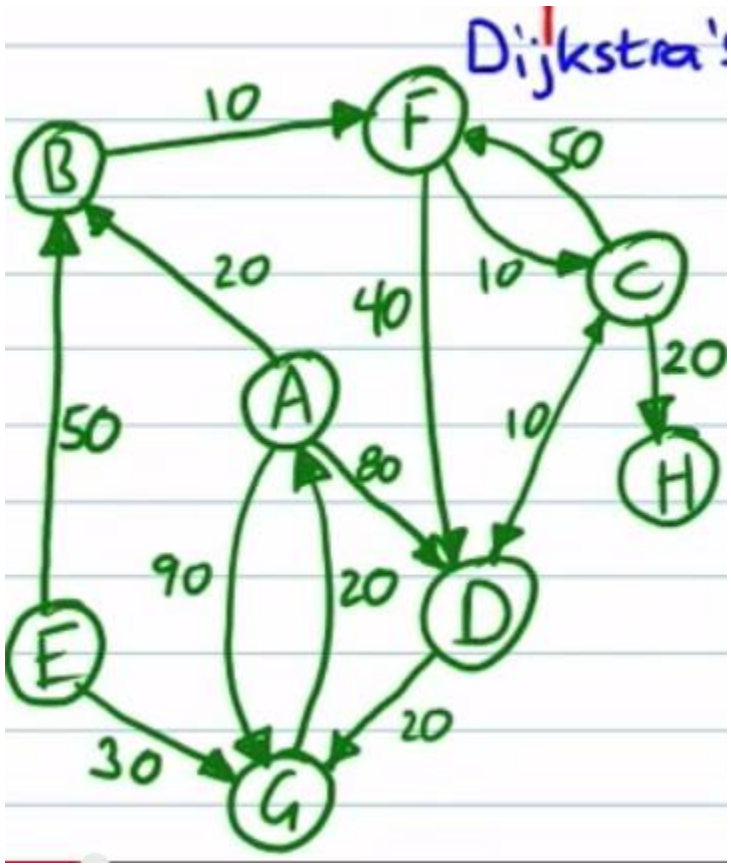


de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf		90 de: A	60 de: C
D				inf		70 de: D	60 de: C

Já sabemos então que o menor caminho entre A e H é de custo 60, e é formado pelo seguinte caminho: A,B,F,C,H

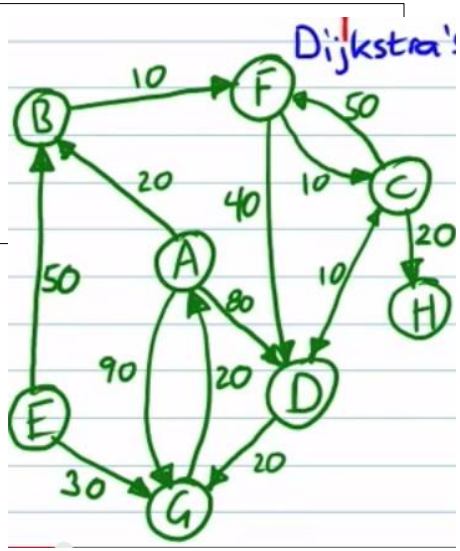
Dijkstra

- Inicia em H, e verifica as saídas de H
  - De H não podemos ir para nenhum nó.
  - Copia a tabela

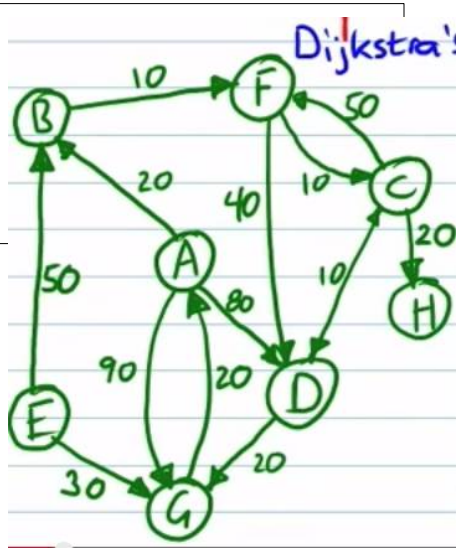


Qual o menor caminho?

de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf		90 de: A	60 de: C
D				inf		70 de: D	60 de: C
H				inf		70 de: D	



Qual o menor caminho?

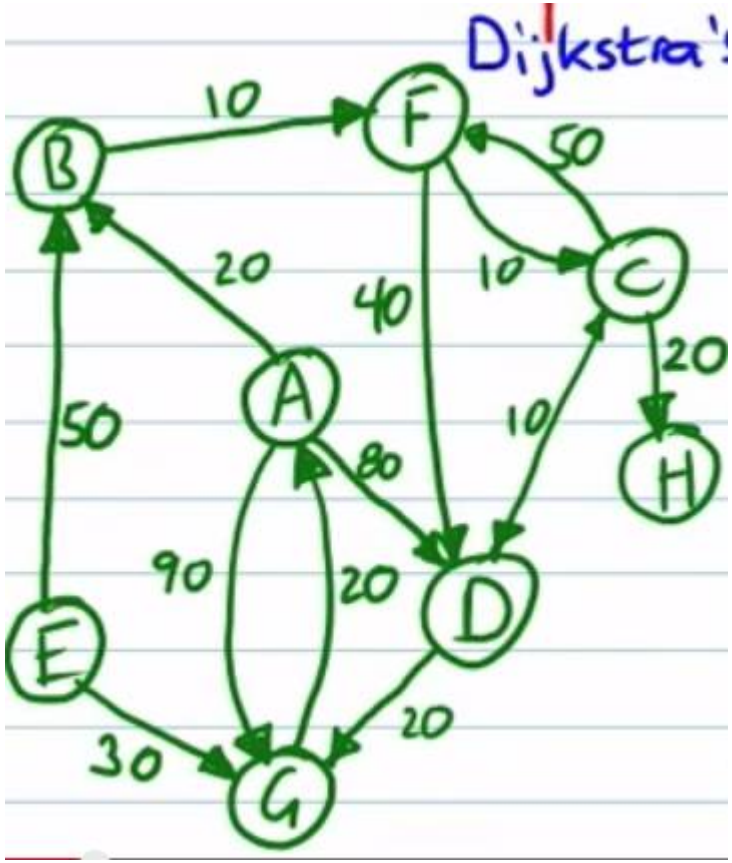


de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf		90 de: A	60 de: C
D				inf		70 de: D	60 de: C
H				inf		70 de: D	

Já sabemos então que o menor caminho entre A e G é de custo 70, e é formado pelo seguinte caminho: A,B,F,C,D,G

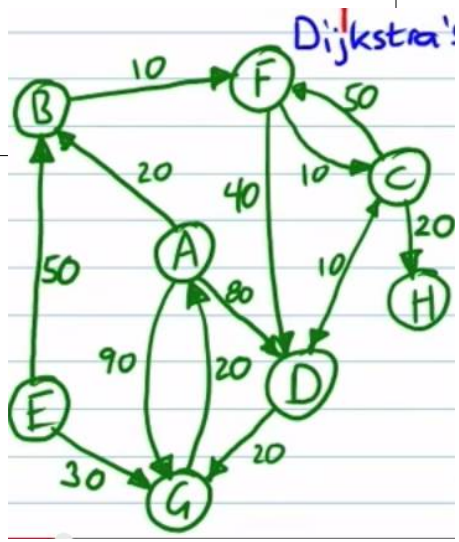
Dijkstra

- Inicia em G, e verifica as saídas de G
  - De G podemos ir para
    - A
  - A é a origem
  - Basta copiar a tabela

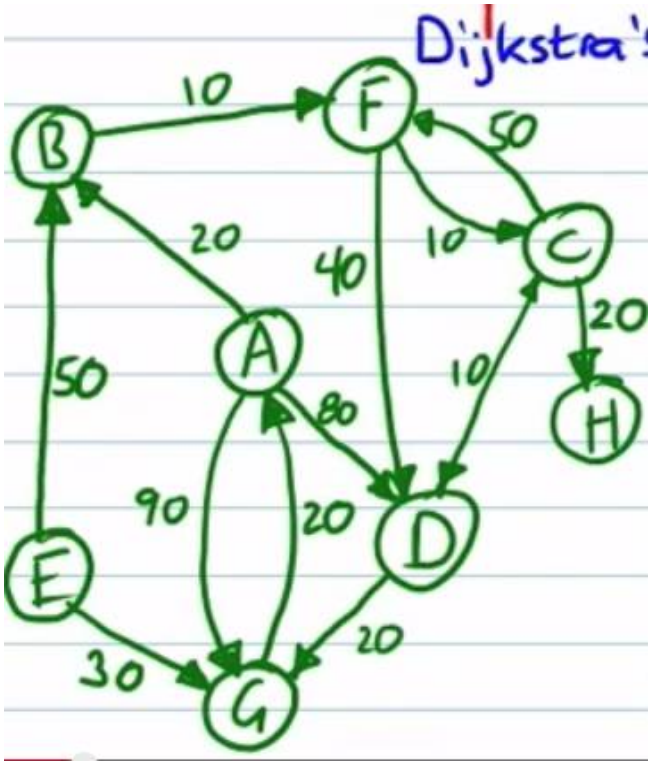


Qual o menor caminho?

de	B	C	D	E	F	G	H
A	20 de: A	inf	80 de: A	inf	inf	90 de: A	inf
B		inf	80 de: A	inf	30 de: B	90 de: A	inf
F		40 de: F	70 de: F	inf		90 de: A	inf
C			50 de: C	inf		90 de: A	60 de: C
D				inf		70 de: D	60 de: C
H				inf		70 de: D	
G				inf			



Não há caminho para o vértice E a partir de A


Dijkstra's
B
F
C
A
H
D
E
G
10
50
20
40
10
20
50
10
80
90
20
20
30

O Algoritmo de Dijkstra

Require: $G = (V, E)$, um grafo orientado ponderado
Require: C, uma matriz de custos associados aos vértices E
Ensure: D um vetor com as custos mínimos entre cada vértice em E e o vértice origem 1

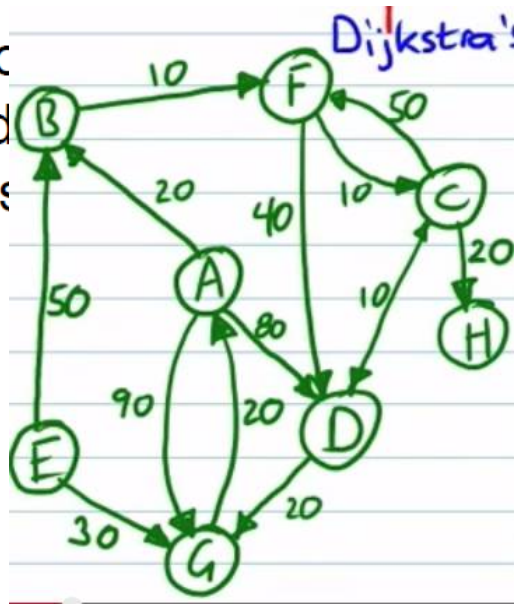
```
1:  S ← {1}
2:  for i ← 2 to n do
3:      D[i] ← C[1, i]
4:  end for
5:  for i ← 2 to n do
6:      encontre um vértice w ∈ V − S tal que D[w] é mínimo
7:      S ← S ∪ {w}
8:      for all v ∈ V − S do
9:          D[v] ← min(D[v], D[w] + C[w,v])
10:     end for
11: end for
```

Slide: Gustavo E.A.P.A. Batista - http://www.lcad.icmc.usp.br/~jbatista/sce183/mat/Didatica.pdf

O Algoritmo de Dijkstra

Require: $G = (V, E)$, um grafo orientado pc
Require: C, uma matriz de custos associad
Ensure: D um vetor com as custos mínimos
em E e o vértice origem 1

1: $S \leftarrow \{1\}$
2: **for** $i \leftarrow 2$ to n **do**
3: $D[i] \leftarrow C[1, i]$
4: **end for**
5: **for** $i \leftarrow 2$ to n **do**
6: encontre um vértice $w \in V - S$ tal que $D[w]$ é mínimo
7: $S \leftarrow S \cup \{w\}$
8: **for all** $v \in V - S$ **do**
9: $D[v] \leftarrow$ min(D[v], D[w] + C[w,v])
10: **end for**
11: **end for**



Slide: Gustavo E.A.P.A. Batista - http://www.lcad.icmc.usp.br/~jbatista/sce183/mat/Didatica.pdf

- Matriz C Vetor D

	A	B	C	D	E	F	G	H
A		20		80			90	
B						10		
C				10		50		20
D							20	
E		50					30	
F			10	40				
G	20							
H								

A	B	C	D	E	F	G	H
inf	20	inf	80	inf	inf	90	inf

A	B	C	D	E	F	G	H
inf	20	inf	80	inf	30	90	inf

A	B	C	D	E	F	G	H
inf	20	40	70	inf	30	90	inf

A	B	C	D	E	F	G	H
inf	20	40	50	inf	30	90	60

A	B	C	D	E	F	G	H
inf	20	40	50	inf	30	70	60

A	B	C	D	E	F	G	H
inf	20	40	50	inf	30	70	60

A	B	C	D	E	F	G	H
inf	20	40	50	inf	30	70	60

- Caso for necessário reconstruir o caminho mais curto entre o vértice origem e cada vértice, pode-se manter um vetor P de vértices, tal que $P[v]$ contém o vértice imediatamente anterior ao vértice v no caminho mais curto.
- Para isso, devemos realizar uma modificação no algoritmo anterior.

Slide: Gustavo E.A.P.A. Batista - http://www.lcad.icmc.usp.br/~jbatista/sce183/mat/Didatica.pdf

O Algoritmo de Dijkstra Modificado

Ensure: P, um vetor com os caminhos de custo mínimo entre a origem e os demais vértices

```
1:  S ← {1}
2:  for i ← 2 to n do
3:      D[i] ← C[1, i]
4:      P[i] ← 1
5:  end for
6:  for i ← 1 to n − 1 do
7:      encontre um vértice w ∈ V − S tal que D[w] é mínimo
8:      S ← S ∪ w
9:      for all v ∈ V − S do
10:         D[v] ← min(D[v], D[w] + C[w,v])
11:         if D[w] + C[w, v] < D[v] then
12:             P[v] ← w
13:         end if
14:     end for
15: end for
```

=

Slide: Gustavo E.A.P.A. Batista - http://www.lcad.icmc.usp.br/~jbatista/sce183/mat/Didatica.pdf

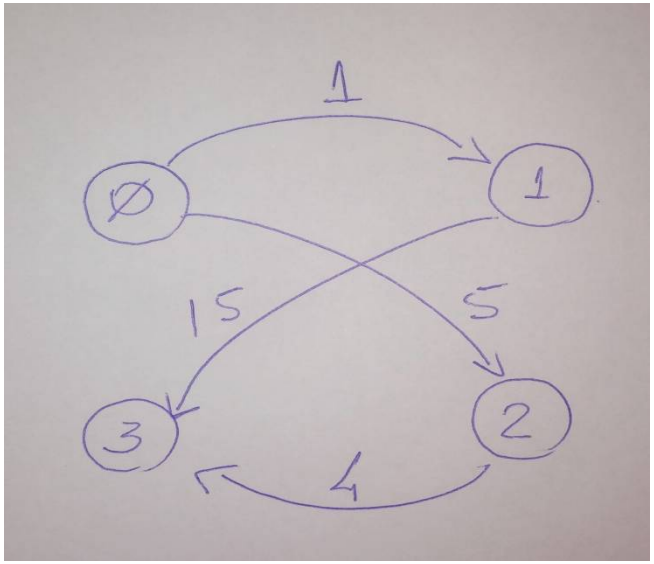
Uma modificação no algoritmo de Djikstra para determinar o caminho mais curto entre dois vértices

```
void Caminho_mais_Curto(TipoGrafo *grafo, int origem, int destino)
{

   int i, vert,k, NovaDist, min;
   int *M, *L, *A, *caminho;
   M = (int *)malloc(grafo->NumVertices*sizeof(int));
   L = (int *)malloc(grafo->NumVertices*sizeof(int));
   A = (int *)malloc(grafo->NumVertices*sizeof(int));
   caminho = (int *)malloc(grafo->NumVertices*3*sizeof(int));  vetor auxiliar

   // incializando vairiaveis

   for (i=0; i<grafo->NumVertices; i++)
   {

      M[i] = 0; //  false -- determina se um vertice ja foi visitado
      A[i] = -1;  // determina o caminho mais curto entre origem e destino
      L[i] = 300000; //infinito determina o comprimento do caminho mais curto
   }
```

```
 vert = origem;
 L[vert] = 0;
while (vert != destino && vert != -1) // não terminou ou caminho inexistente
    {
       for(i=0;i<grafo->NumVertices; i++) // percorre vertices adjacentes de vert
         if (grafo->Mat[vert][i] != 0 && M[i]==0)   // se aresta existe e ela  não foi visitada
         {
           NovaDist = L[vert] + grafo->Mat[vert][i];
           if (NovaDist < L[i])
           {
              L[i] = NovaDist;   // atualiza menor distancia
              A[i] = vert;   // atualiza caminho
           }
         }
       M[vert] = 1;   // toda a lista de adjacentes de vert já foi analisada
       min = 300000;   //infinito
       vert = -1;  // valor invalido
       for (i=0; i<grafo->NumVertices; i++)  // encontra proximo vertice do caminho
          if (M[i]==0 && L[i] < min)  //escolhe o vertice  cuja aresta possui o menor peso
          {
            min = L[i];   // atualiza min
            vert = i;      //atualiza vert
          }
    } //fim while
```

```
// listar o caminho mais curto entre origem e destino

    if (vert == destino) // encontrou um caminho
    {
       printf("caminho mais curto entre %4d e %4d tem comprimento  %4d: ",
                    origem, destino, L[destino] );
       caminho[0] = destino;
        k = 1;
       while (vert != origem)
       {
          caminho[k]= A[vert];
          vert = A[vert];
          k++;
       }
       for (i=k-1; i>=0; i--)
         printf("%4d", caminho[i]);
    }
    else printf("nao exite caminho entre %4 e %4", origem, destino);
}
```

Considere o grafo:



Execute manualmente o algoritmo dado e determine o caminho mais curto entre os vértices 0 e 3.

```
grafo criado
  0   1   5   0
  0   0   0  15
  0   0   0   4
  0   0   0   0
```

MLA

```
     1       0       0       0
     0       1       5  300000
    -1       0       0      -1
```

MLA

```
     1       1       0       0
     0       1       5      16
    -1       0       0       1
```

MLA

```
     1       1       1       0
     0       1       5       9
    -1       0       0       2
```

```
caminho mais curto entre    0 e    3 tem comprimento    9:    0   2   3
```