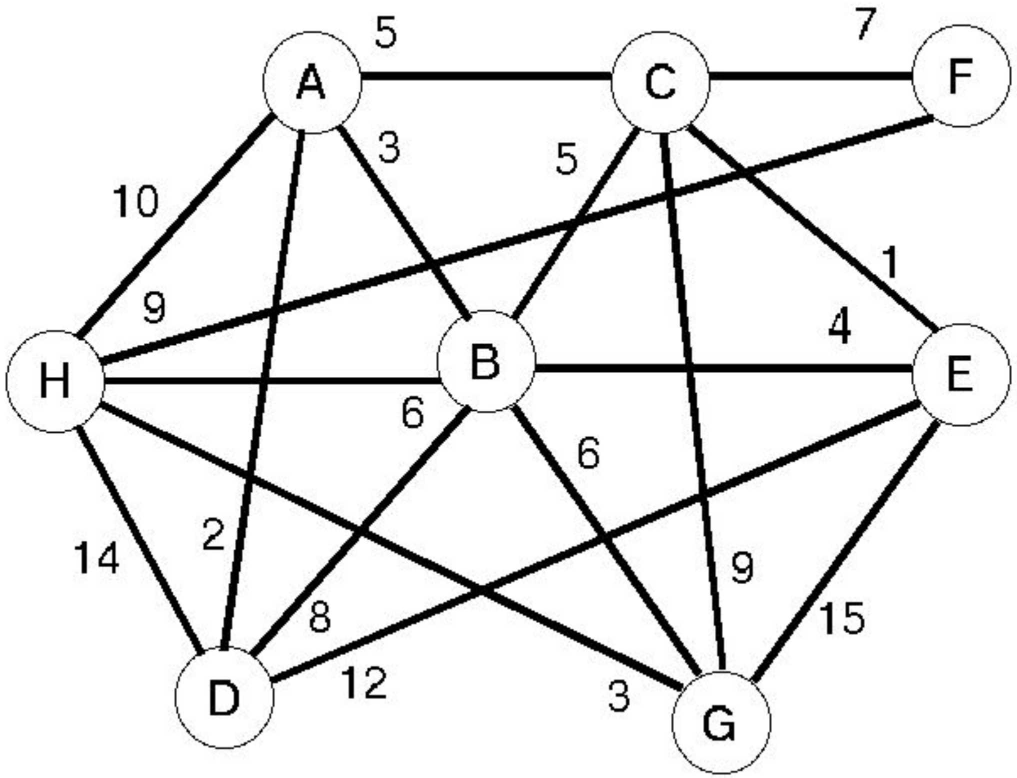
A* e Best-Fisrt Search

GRAFOS

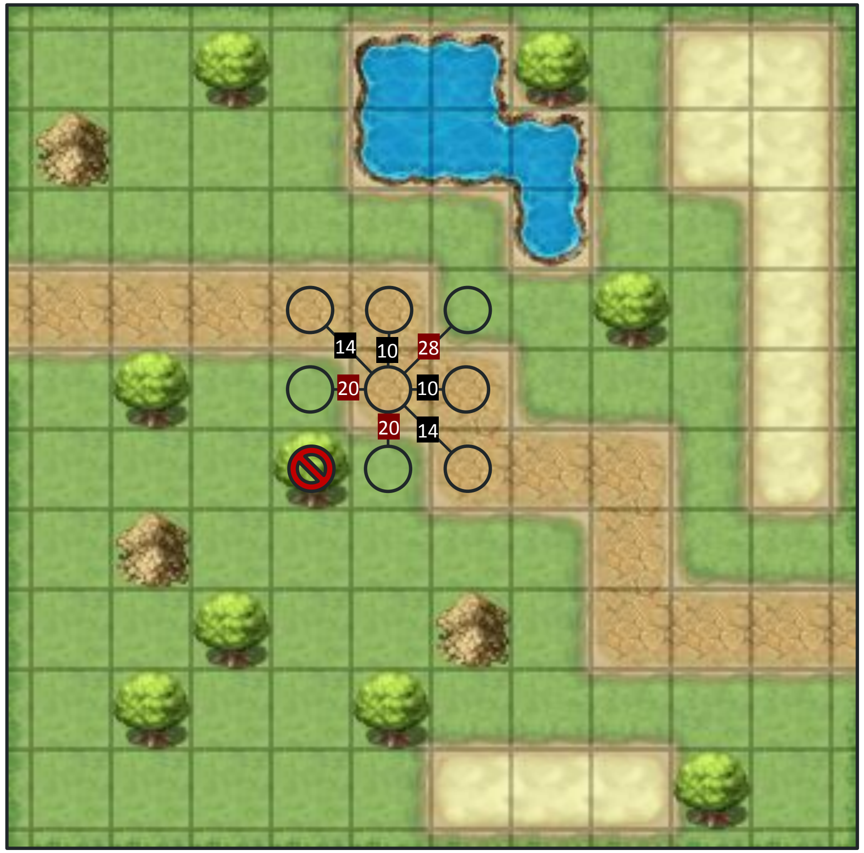
Um pouco de história

- O A* surgiu em 1968 para Shakey the Robot para andar em uma sala com obstáculos.
- Nils Nilsson sugeriu a heurística. O algoritmo foi chamado de A1.
- Bertram Raphael sugeriu aperfeiçoamentos e foi chamado de A2.
- Then Peter E. Hart fez ajustes ao algoritmo.

Grafo – Como vimos até agora

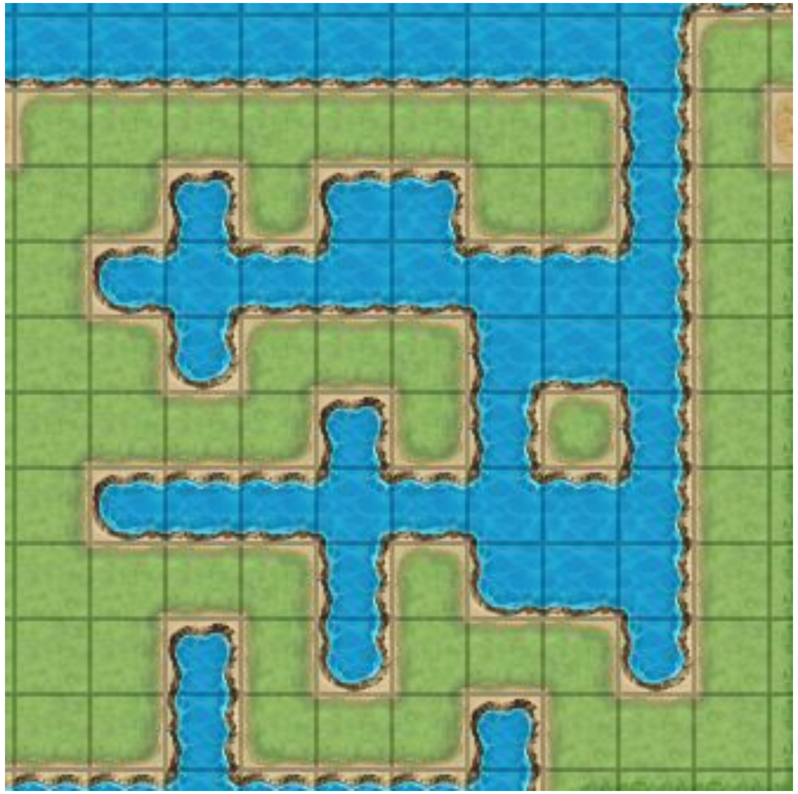


Grafo – Outra abordagem



Grafo – Outra abordagem

Grafo – Outra abordagem



Best-First Search

- O A* busca se aproximar de um destino estimando sua atual distância, mas para isto temos um outro algoritmo, o Best-First Search.
- Este algoritmo é simples e rápido, ele avalia poucas opções para chegar ao destino.
- Ele tende a ser rápido, porém nem sempre encontra o menor caminho.

Best-First Search

Definir um vértice inicial como atual

Marcar o vértice atual como fechado

Verificar todos os vizinhos

- Se não está na lista de vértices abertos
    - Calcular H (Estimativa da distância até o destino)
    - Adicionar na lista aberta com o atual como anterior

Colocar o vértice da lista aberta com menor valor H como vértice atual

Repetir o processo enquanto o vértice atual não é o objetivo

A* (A Estrela)

- O A* tem como objetivo encontrar um destino a partir do ponto de partida pelo menor caminho possível, ele procura unir as forças do Dijkstra e do Best-First-Search:
  - A garantia que o Dijkstra possui
  - A velocidade que o Best-First Search Possui
- O algoritmo procura se aproximar do destino utilizando uma estimativa, como o Best-First Search, porém ele tenta se o afastar o mínimo possui do início, como o Dijkstra.

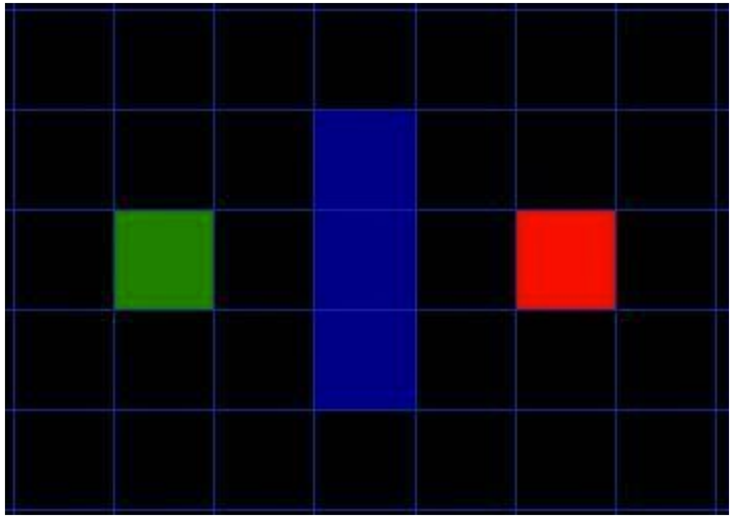
Função de Estimativa

- Tenta prever a distância até o destino
- Situações diferentes, resultados diferentes
- Equilibrar custo/benefício
- Diferentes formas:
  - Método Manhattan
  - Atalho Diagonal
  - Distância Euclidiana

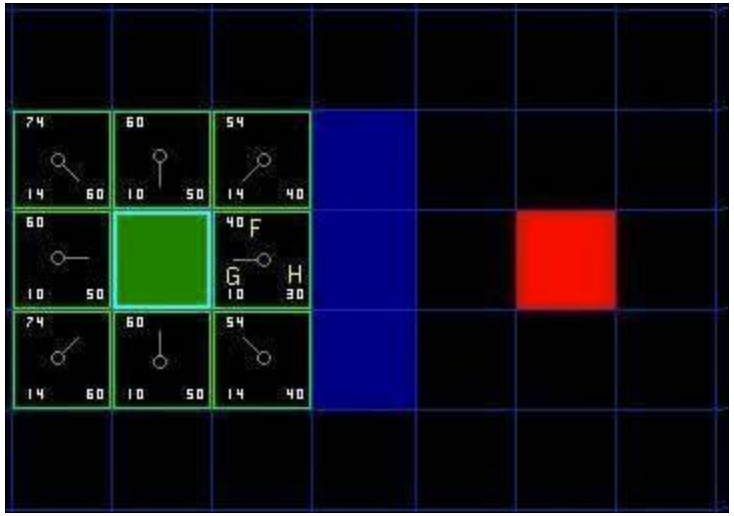
A* - Algoritmo

- Definir o vértice inicial como atual
- Adicionar o vértice atual como fechado
- Verificar todos os vizinhos
  - Se não está na lista de vértices abertos
    - Adicionar em uma lista aberta, adicionar o atual como anterior
    - Calcular o H (estimativa do destino)
    - Definir G (distância do início) como no Dijkstra
  - Se já está na lista
    - Substituir o G caso o valor pelo vértice atual seja menor, mudar o vértice anterior
- Colocar o vértice da lista aberta com menor valor F(G + H) como vértice atual
- Repetir o processo enquanto o vértice atual não é o objetivo

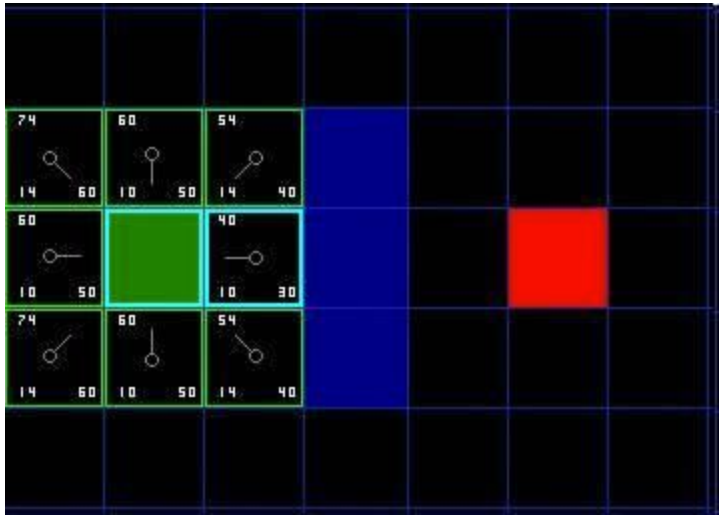
A* - Algoritmo



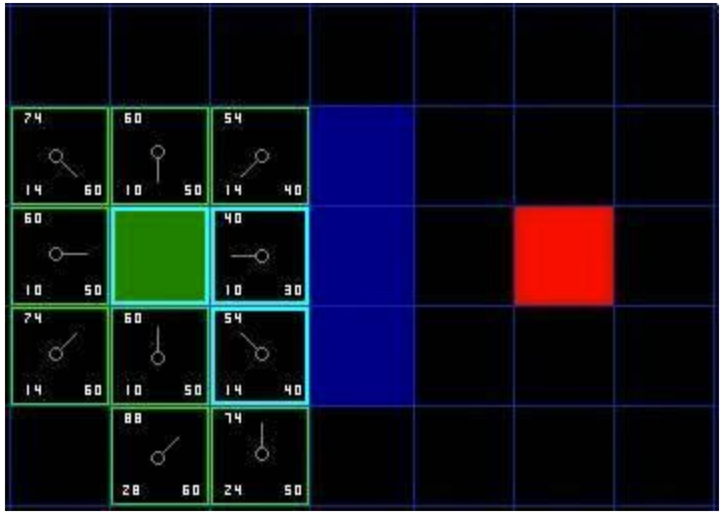
A* - Algoritmo


74
60
54
14
60
10
50
14
40
60
40
F
G
H
10
50
10
30
74
60
54
14
60
10
50
14
40

A* - Algoritmo

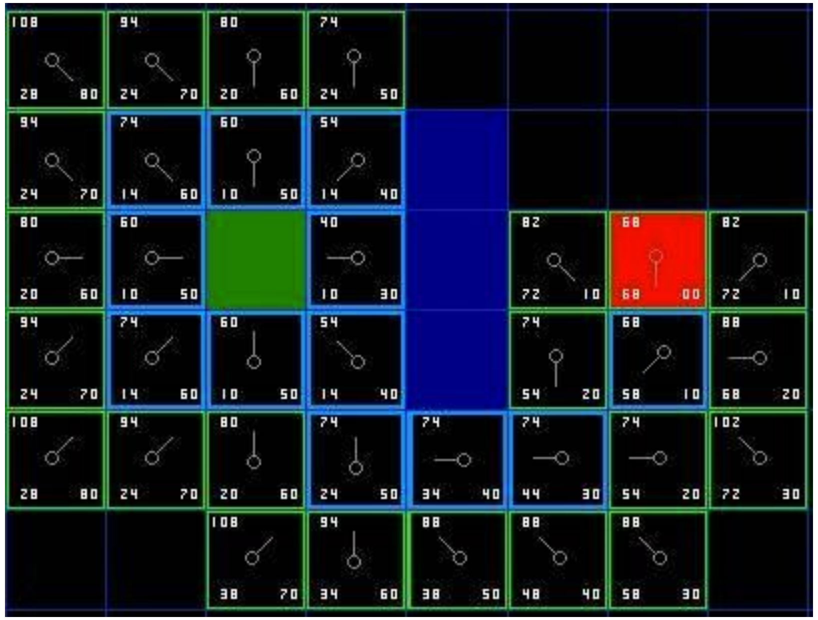


A* - Algoritmo



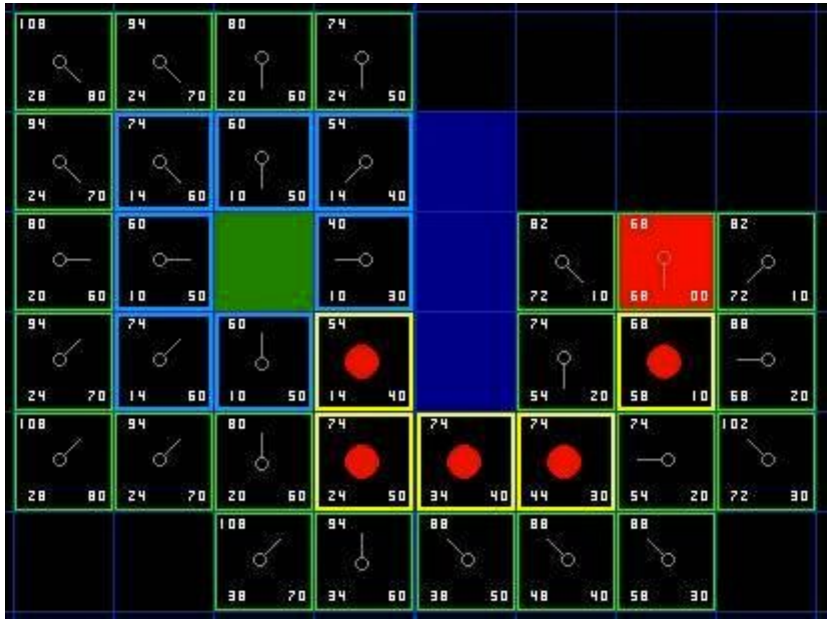
74
14 60
60
10 50
54
14 40
60
10 50
40
10 30
74
14 60
60
10 50
54
14 40
88
28 60
74
24 50

A* - Algoritmo



108
28 80
94
24 70
80
20 60
74
24 50
94
24 70
74
14 60
60
10 50
54
14 40
80
20 60
60
10 50
40
10 30
82
72 10
68
68 00
82
72 10
94
24 70
74
14 60
60
10 50
54
14 40
74
54 20
68
58 10
88
68 20
108
28 80
94
24 70
80
20 60
74
24 50
74
34 40
74
44 30
74
54 20
102
72 30
108
38 70
94
34 60
88
38 50
88
48 40
88
58 30

A* - Algoritmo


108
28 80
94
24 70
80
20 60
74
24 50
94
24 70
74
14 60
60
10 50
54
14 40
80
20 60
60
10 50
40
10 30
82
72 10
68
68 00
82
72 10
94
24 70
74
14 60
60
10 50
54
14 40
74
54 20
68
58 10
88
68 20
108
28 80
94
24 70
80
20 60
74
24 50
74
34 40
74
44 30
74
54 20
102
72 30
108
38 70
94
34 60
88
38 50
88
48 40
88
58 30

CAIXEIRO VIAJANTE

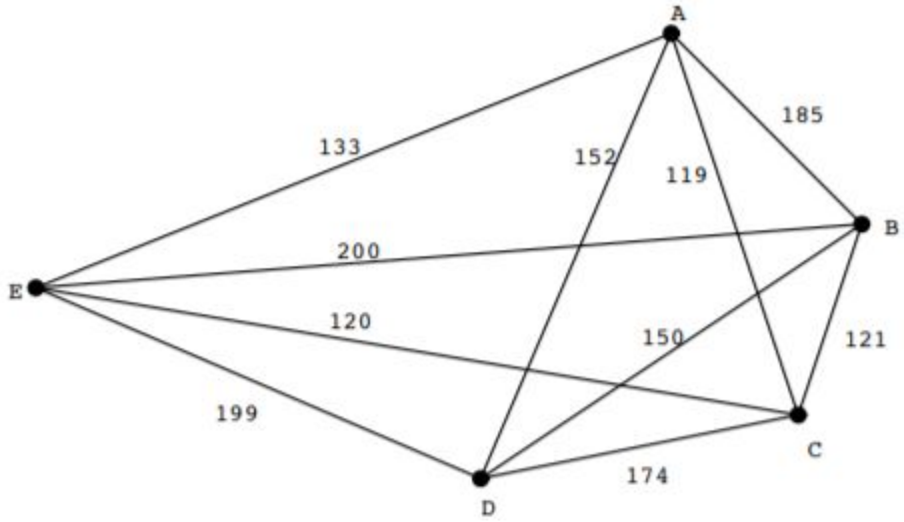
GRAFOS

Caixeiro Viajante

- O Problema do Caixeiro Viajante (PCV) é o nome que usualmente se da à uma série de problemas reais importantes que podem ser modelados em termos de ciclos Hamiltonianos em grafos completos. De seguida apresentamos três exemplos deste tipo de problema, começando com um que dá o nome a esta classe de problemas.

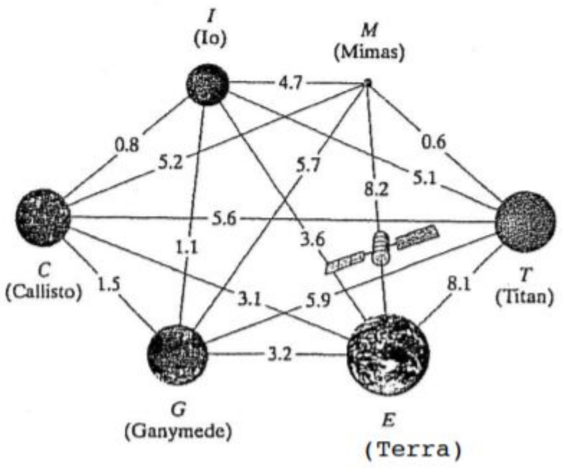
Caixeiro Viajante

- (Zé Pedro, o caixeiro viajante). O Zé Pedro é um caixeiro viajante que tem clientes em cinco cidades, que abreviamos por A, B, C, D e E. Ele precisa planejar uma viagem de negócios com cidade de partida e de destino final A (a cidade onde mora), passando por cada uma das restantes quatro cidades precisamente uma vez. O grafo abaixo representa o custo de cada viagem (em qualquer um dos sentidos) entre cada par de cidades.



Caixeiro Viajante

- (As Luas de Júpiter e Saturno). No ano 2025 será lançada da Terra uma expedição para explorar as luas de Júpiter e Saturno. Serão visitadas Calisto, Ganímedes, Io (luas de Júpiter), Mimas e Titã (luas de Saturno), onde serão recolhidas amostras com as quais a expedição voltará à Terra. A seguinte figura indica a duração da missão (em anos) entre cada par de luas. Como determinar um ciclo Hamiltoniano ótimo (de duração mais curta) no grafo representado?



Caixeiro Viajante

- (Vida em Marte). A tabela abaixo indica as distancias aproximadas, em milhas, entre sete locais de Marte, onde cientistas da NASA pensam haver maior probabilidade de encontrar vestígios de vida.

	A	G	H	I	N	P	W
A	-	7500	5000	2800	3500	1500	2200
G	7500	-	3000	6000	8000	6500	5000
H	5000	3000	-	4000	4800	3500	2800
I	2800	6000	4000	-	2000	3000	2900
N	3500	8000	4800	2000	-	4000	3200
P	1500	6500	3500	3000	4000	-	1300
W	2200	5000	2800	2900	3200	1300	-

- Quer-se planejar uma viagem que colocará um robô em Marte, aterrizando em A. O robô deverá percorrer cada um dos locais, recolher amostras de solo e regressar até A, de onde um foguete trará as amostras para Terra de forma a serem analisadas. Como encontrar um ciclo Hamiltoniano ótimo no grafo correspondente?

Caixeiro Viajante

- Estes três exemplos tem algumas coisas em comum, temos um problema que é modelado por um grafo completo cujas arestas têm pesos. A solução do problema consiste em encontrar um ciclo Hamiltoniano no grafo, de forma a que a soma dos pesos das arestas escolhidas seja o menor possível.
- Aqui não se põe a questão da existência de ciclo Hamiltoniano, já que o grafo subjacente é completo, logo Hamiltoniano.
- A questão que se propõe é a da procura de uma solução ótima.

Caixeiro Viajante

Outros problemas que é comum encontrar na categoria de problemas PCV são os seguintes:

- Rotas de ônibus escolares;
- Redes de distribuição de encomendas;
- Construção de placas de circuitos integrados;
- Programação de máquinas de linhas de montagem.

Caixeiro Viajante – Modo Exaustivo

- Este método consiste em fazer uma lista de todos os ciclos Hamiltonianos do grafo, calcular o peso de cada um e escolher um de peso mínimo.
- Um caminho em Kn é determinado por uma sequência de vértices distintos. Para ser um ciclo Hamiltoniano, todos os vértices devem ocorrer na sequência, e o último vértice deverá ser igual ao primeiro.
- Logo, existem n! sequências distintas. No entanto, num ciclo não interessa qual é o vértice inicial, já que podemos começar em qualquer vértice. Assim, podemos fixar o vértice inicial e obtemos um total de (n – 1)! ciclos Hamiltonianos distintos em.

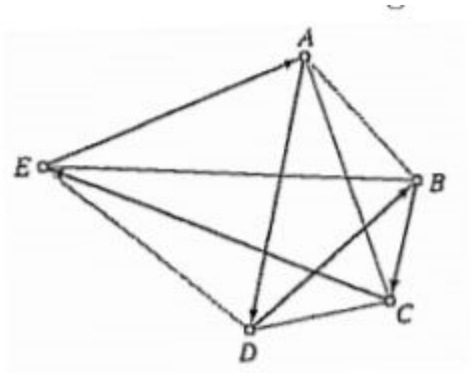
Caixeiro Viajante – Modo Exaustivo

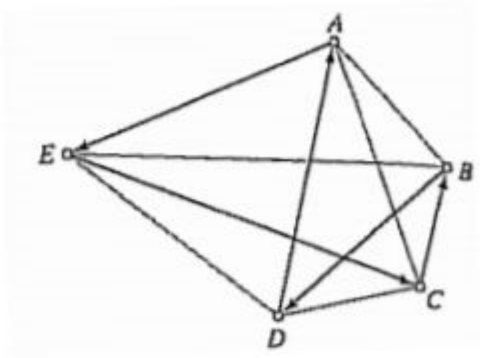
- (Zé Pedro, o caixeiro viajante). Uma forma de resolver o problema do Zé Pedro é usar o método de exaustão em que se calculam os pesos dos 4! = 24 ciclos Hamiltonianos possíveis em um K5:

ciclo Hamiltoniano	custo total	ciclo inverso
A-B-C-D-E-A	185+121+174+199+133=812	A-E-D-C-B-A
A-B-C-E-D-A	185+121+120+199+152=777	A-D-E-C-B-A
A-B-D-C-E-A	185+150+174+120+133=762	A-E-C-D-B-A
A-B-D-E-C-A	185+150+199+120+119=773	A-C-E-D-B-A
A-B-E-C-D-A	185+200+120+174+152=831	A-D-C-E-B-A
A-B-E-D-C-A	185+200+199+174+119=877	A-C-D-E-B-A
A-C-B-D-E-A	119+121+150+199+133=722	A-E-D-B-C-A
A-C-B-E-D-A	119+121+200+199+152=791	A-D-E-B-C-A
A-C-D-B-E-A	119+174+150+200+133=776	A-E-B-D-C-A
A-C-E-B-D-A	119+120+200+150+152=741	A-D-B-E-C-A
A-D-B-C-E-A	152+150+121+120+133=676	A-E-C-B-D-A
A-D-C-B-E-A	152+174+121+200+133=780	A-E-B-C-D-A

Caixeiro Viajante – Modo Exaustivo

- Verificamos assim que há exatamente dois ciclos ótimos, o ciclo A – D – B – C – E – A, e o seu inverso, o ciclo A – E – C – B – D – A. Em qualquer um dos casos, o Zé Pedro gasta 676 Reais na sua viagem de trabalho e esta é a melhor solução.





Caixeiro Viajante – Modo Exaustivo

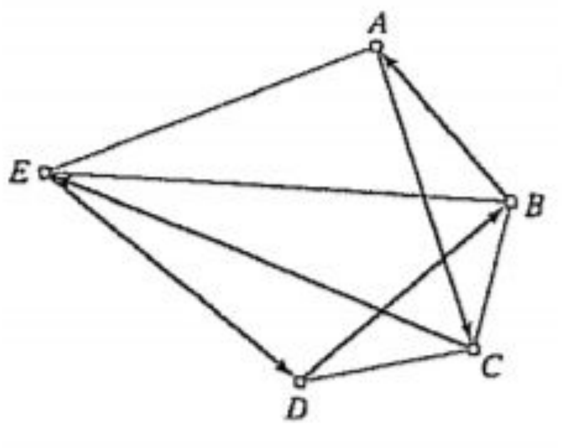
- Este método é, em geral, impossível de implementar já que o número de operações a se fazer é da ordem de $(n-1)!$. Em um (super) computador capaz de determinar o peso de mil bilhões (isto é, 1 seguido de 15 zeros) de ciclos Hamiltonianos por segundo (atualmente tal computador não existe!), este método demoraria 10 meses para $n = 24$, 20 anos para $n = 25$, 500 anos para $n = 26$ e 284 milhões de anos para $n = 30$.

Caixeiro Viajante – Método do Vizinho Mais Próximo

- Escolhe-se um vértice e a aresta de menor peso incidente nesse véertice. Esta aresta determina um outro vértice. De cada novo vértice escolhe-se a aresta de menor peso, entre as arestas que são incidentes nesse vértice e em um vértice que ainda não foi escolhido. No final, retorna-se ao vértice inicial.

Caixeiro Viajante – Método do Vizinho Mais Próximo

- (Zé Pedro, o caixeiro viajante). No caso do Zé Pedro, por este algoritmo ele começa pelo vértice A. De A vai para C, de C para E, depois para D e finalmente para B, de onde retorna ao A.
- O custo desta viagem é de 773 Reais.



Caixeiro Viajante – Método do Vizinho Mais Próximo

- O método do vizinho mais próximo é muito mais rápido do que que o método exaustivo, embora não produza, em geral, uma solução ótima. No caso do exemplo anterior, o custo adicional é de 97 Reais, e o erro relativo é de 14.3%.
- Será que este método guloso é suficientemente eficiente? Se considerarmos como uma única operação a procura de entre todas as arestas incidentes num dado vértice da aresta de menor peso, então temos de efetuar uma operação para o primeiro vértice, uma operação para o segundo, e assim sucessivamente. Quando chegarmos ao último vértice ainda não escolhido, a única hipótese é voltar ao vértice inicial. Logo, o custo deste algoritmo é da ordem de $n - 1$ operações.
- Por exemplo, se $n = 30$ e nos permitirmos demorar um minuto por cada operação, conseguimos aplicar este algoritmo em meia hora, algo que, usando o método exaustivo demoraria em um (super) computador mais de 200 milhões de anos para executar. Este é o ponto forte deste e de muitos outros algoritmos de tipo guloso.