INTRODUÇÃO À INTELIGÊNCIA ARTIFICIAL

CAP. 5 PESQUISA INFORMADA (HEURÍSTICA)

Carlos Pereira
ISEC 22-23

Índice

2

- Índice
  - Pesquisa Sôfrega
  - Pesquisa A*
  - Variantes do A*

Motivação

3

- Os métodos de Pesquisa Não Informada são muito ineficientes:
  - Exigem Grandes requisitos de tempo
  - Exigem Grandes requisitos de memória
  - As Soluções encontradas nem sempre são óptimas

Métodos do tipo "Best-First"

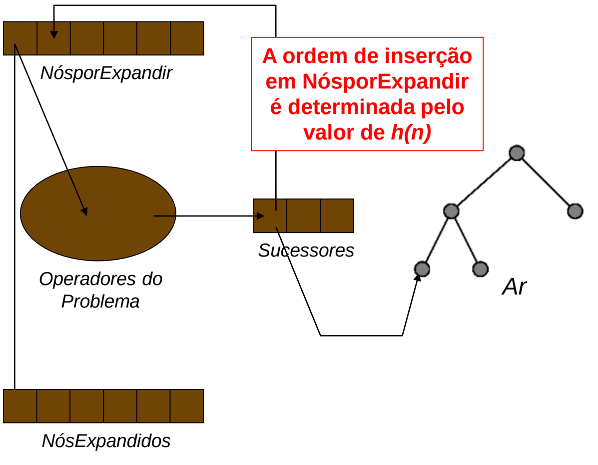
4

- Método "Best-First" – O melhor primeiro
  - Recorrem a conhecimento pericial para definir uma função de avaliação
    - Conhecimento específico acerca do domínio do problema
  - A Função de avaliação de um estado n - h(n) retorna um valor indicativo da vantagem em expandir esse estado primeiro.
    - De acordo com a estrutura do AGP, cada nó sucessor é inserido ordenadamente na lista de nós a expandir, em função do valor de h(n)

Métodos do tipo "Best-First"

5

- …



Métodos do tipo "Best-First"

6

- Pesquisa Sôfrega
  - Expande-se em primeiro lugar o nó que parece estar mais perto do objectivo.
    - em muitos problemas pode obter-se uma estimativa do custo do caminho de um dado nó até ao objectivo
    - A estimativa é calculada por h(n) - Função Heurística
      - Se h(n)=0, o nó n coincide com o nó objectivo
      - Se h(n)>=0, o nó objectivo pode ser atingido a partir do nó n, sendo o custo estimado em h(n)
      - Se h(n)=∞, o objectivo não pode ser atingido a partir do nó n

…Pesquisa Sôfrega

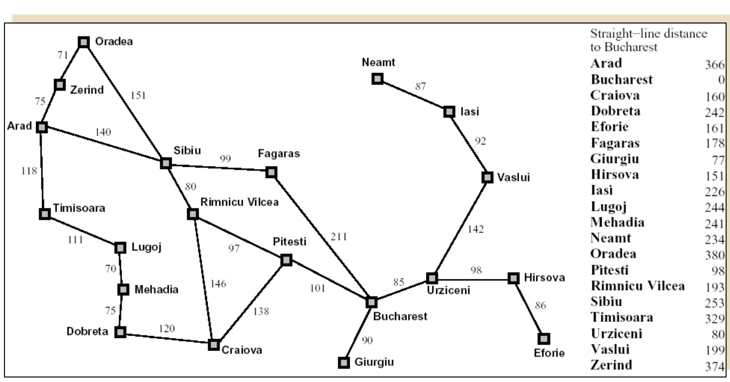
7

- …
  - Exemplo
    - Que heurística poderemos utilizar para o problema de determinar o melhor caminho entre duas cidades?
      - poderá ser a Distância em Linha Recta, de cada cidade à cidade objectivo
      - Neste caso, para calcular h(n), basta ter as coordenadas (x,y) de cada cidade

…Pesquisa Sôfrega

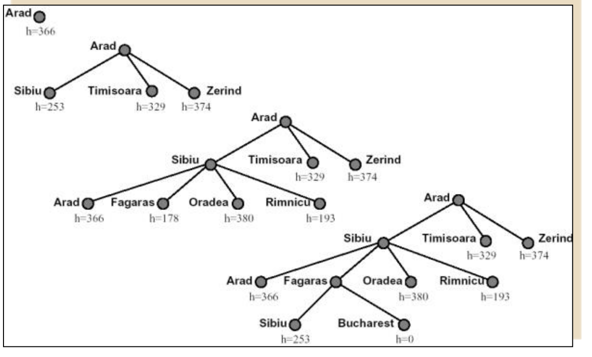
8

- …
  - ?



...Pesquisa Sôfrega

- ...



  - Solução: Arad → Sibiu → Fagaras → Bucharest

...Pesquisa Sôfrega

- ...
  - Características
    - Complexidade temporal exponencial - O(b^d)
      - com b=Factor de Ramificação e d=Número de Níveis da Árvore (máxima profundidade do espaço)
    - Complexidade espacial exponencial - O(b^d)
      - Complexidades temporal e espacial podem ser substancialmente reduzidas se h(n) for adequada
    - Não óptima
      - não garante que se encontre o caminho de menor custo
    - Incompleta
      - pode seguir caminhos infinitos

Pesquisa A*

11

- Pesquisa A*
  - Combina Pesquisa Uniforme com Pesquisa Sôfrega

  - Pesquisa Uniforme
    - Guiada pelo custo mínimo da origem a cada nó n, expande primeiro o nó efectivamente mais perto da origem
    - Óptima se o custo aumentar com a profundidade
    - Complexidade $O(b^d)$

Pesquisa A*

12

- …
  - Pesquisa Sôfrega:
    - Guiada pelo custo mínimo aparente de cada nó n até ao objectivo, expande primeiro o nó que parece mais perto do objectivo
    - Não óptima
    - De complexidade $O(b^d)$ mas muito reduzida para boas funções h(n)

Pesquisa A*

13

- …
  - Estes dois tipos de pesquisa são complementares:
    - A Uniforme "mede" a parte inicial do percurso - g(n)
    - A Sôfrega "mede" a aparente parte restante - h(n)
    - Os custos do caminho provenientes de ambas podem combinar-se numa simples soma: f(n)=g(n)+h(n)
      - "mede" o custo estimado da solução que passa pelo nó n
      - No AGP, a inserção na lista de Nós a Expandir é feita por ordem crescente de f(n)

Pesquisa A*

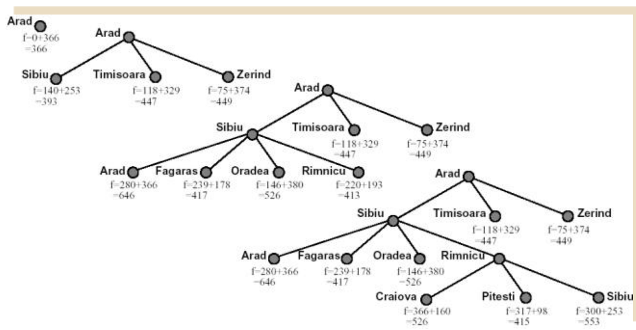
14

- …
  - A pesquisa A* é óptima e completa desde que:
    - A heurística utilizada nunca sobrestime o custo do caminho do nó n até ao objectivo (isto é, nunca possa assumir um valor superior ao do custo real).
    - Se assim for, constitui uma Heurística Admissível.
      - Será a distância em linha recta entre cidades uma heurística admissível?

Pesquisa A*

15

- …
  - Exemplo:

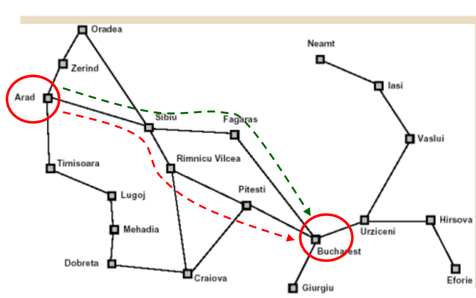


- $f(n)=g(Arad,n) + h_{SLD}(n,Bucharest)$

Pesquisa A*

16

- …
  - Solução:
    - Arad → Sibiu → Riminicu → Pitesti → Bucharest



Pesquisa A*

17

- …
- Outros exemplos:
  - http://aispace.org/search/
  - http://aigamedev.com/open/interviews/mario-ai/

Funções Heurísticas

18

- Considere-se o Puzzle de 8 peças:

  - Factor de ramificação médio: b=3 (quadrado vazio no meio=4, no canto=2, junto a um lado=3)
  - Tipicamente, uma solução ocorre em 20 passos, Portanto, uma pesquisa exaustiva procuraria, em média, num espaço de 3^{20} estados
    - Evitando a passagem por estados repetidos, este número pode reduzir-se a 362.880
  - O recurso a uma heurística efectua uma redução significativa do número de estados!

Funções Heurísticas

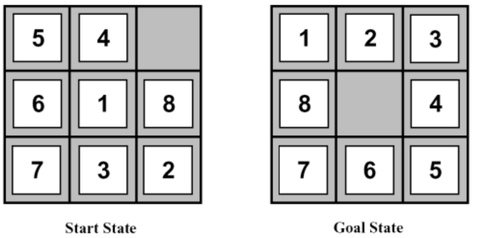
19

- …
  - Qual a melhor heurística?
    - A heurística e escolher nunca pode sobre-estimar o custo do caminho até à solução.
    - Uma possibilidade:
      - h1 = Número de peças que, em cada estado, estão numa posição não coincidente com a posição que devem ocupar na solução
        - nunca sobre-estima o custo do caminho, porque cada peça fora de ordem terá de ser movida pelo menos uma vez: Assim, 7 é o custo mínimo do caminho até à solução

Funções Heurísticas

20

- …
  - Exemplo:

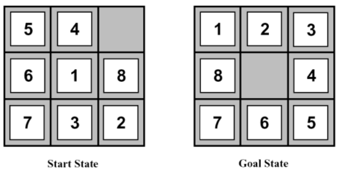


Start State Goal State

- Estado inicial: h=7
- Estado Final: h=0

Funções Heurísticas

- …
  - Outra Possibilidade
    - h2 = Soma das distâncias das peças até à sua posição final.
      - Soma das distâncias horizontais com as verticais (Manhattan Distance)



      - No exemplo, para o Estado Inicial, h=2+3+3+2+4+2+0+2=18
      - h nunca sobre-estima o custo do caminho, porque cada peça fora de ordem terá de ser movida pelo menos x vezes até atingir a sua posição final (não podem mover-se na diagonal)

Funções Heurísticas

- …
  - Uma heurística é caracterizada pelo Factor de Ramificação Efectivo, b* :
    - Seja N o número total de nós expandidos pela pesquisa A*, Então: $N=1+b^*+(b^*)^2+\ldots+(b^*)^d$
    - Uma boa heurística apresenta um b* próximo de 1
      - b*=1 significa que a pesquisa escolheria sempre o nó correcto a expandir, progredindo sempre, sem erros, em direcção à solução!

Funções Heurísticas

- …
  - 100 puzzles gerados aleatoriamente
  - Soluções a profundidades que variam entre 2 e 24
    - Nós expandidos e factor de ramificação efectivo (b*) para as heurísticas h1 e h2 do puzzle 8:

	Search Cost			Effective Branching Factor		
d	IDS	A*(h_1)	A*(h_2)	IDS	A*(h_1)	A*(h_2)
2	10	6	6	2.45	1.79	1.79
4	112	13	12	2.87	1.48	1.45
6	680	20	18	2.73	1.34	1.30
8	6384	39	25	2.80	1.33	1.24
10	47127	93	39	2.79	1.38	1.22
12	364404	227	73	2.78	1.42	1.24
14	3473941	539	113	2.83	1.44	1.23
16	–	1301	211	–	1.45	1.25
18	–	3056	363	–	1.46	1.26
20	–	7276	676	–	1.47	1.27
22	–	18094	1219	–	1.48	1.28
24	–	39135	1641	–	1.48	1.26

Funções Heurísticas

- …
  - Uma função heurística que calcule o custo exacto da solução para um problema com menos restrições, embora derivado do original (relaxed problem) constitui geralmente uma boa solução

  - A função heurística cujo valor se aproxima mais do custo da solução real, é habitualmente a melhor

Funções Heurísticas

- …
  - Puzzle-8 com menos restrições:
    - Exemplo 1
      - Um puzzle em que as peças se podem mover logo para a posição final (em vez de uma só posição de cada vez e apenas para a posição livre)
    - Exemplo 2
      - Um puzzle em que as peças se possam mover uma só posição de cada vez, mas para qualquer posição, e não apenas para a livre
    - Qual a melhor heurística para cada um dos exemplos? h1 ou h2?

Funções Heurísticas

- …
  - No exemplo 1
    - h1 daria o custo exacto da solução!
  - No exemplo 2,
    - h2 daria o custo exacto da solução!

Variantes A*

27

- Variantes do A* com limitação de memória
  - IDA*
    - A* com aprofundamento progressivo
    - "IDS para A*"
  - SMA*
    - "Simplified Memory Bounded A*"
    - Desenhado para não ultrapassar o limite de memória disponível para resolver um problema.

IDA*

28

- IDA*
  - Está para a a pesquisa A*, assim como o IDS está para a pesquisa em profundidade:
    - No IDS cada iteração é limitada por um nível de profundidade crescente
    - No IDA* cada iteração é limitada por um valor crescente da função de custo, $f(n)=g(n)+h(n)$

IDA*

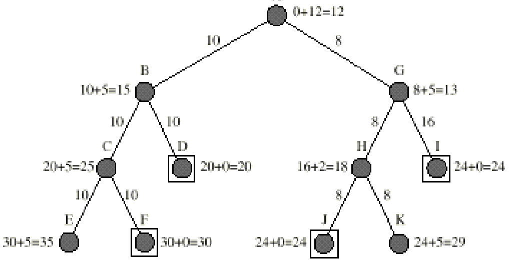
29

- ...
  - Para cada "limite de custo estimado", fi ,"exclui" os nós cujo valor f é superior
  - Pára quando atingir um nó objectivo cujo f é ≤ que o limite actual
  - Enquanto não encontrar um objectivo nestas condições, progride para o limite seguinte, fi+1. Este limite, fi+1 :
    - Pode provir de outro nó situado à mesma profundidade do que proporcionou o limite anterior, fi: O IDA* é controlado pelo valor de f e não pela profundidade do nó, d
    - É determinado na iteração i, escolhendo o menor custo estimado de entre todos os custos estimados associados aos nós por expandir

IDA*

30

- Exemplo



  - D, I, F e J representam todos o objectivo a atingir, assim, os ramos que a eles conduzem são **vários caminhos possíveis** até esse objectivo

IDA*

31

- …
  - Estado Inicial
    - Nós por Expandir: A
    - Limite Iteração Seguinte =

      = Custo Estimado (A, objectivo) =

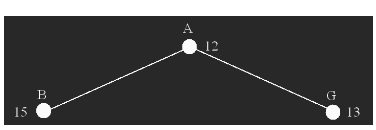
      =f1=

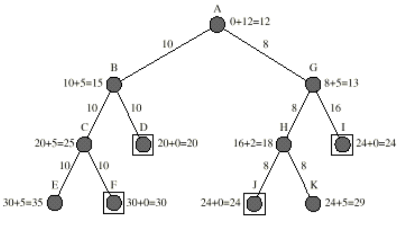
      = min(fNósPorExpandir) = g(A)+h(A) = 0+12 = 12

IDA*

32

- …
  - Iteração 1





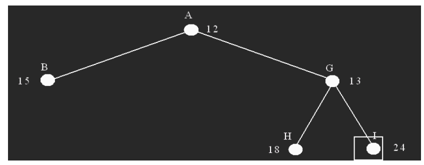
    - A foi expandido
    - Limite Actual=f1=12
    - Como f(B)=15 e f(G)=13 são >12, B e G são excluídos
    - Limite Iteração Seguinte = f2 = min(fNósNãoExpandidos) =

      = min(f(B), f(G)) = 13
    - o nó G será o próximo a expandir (por ser o 1° da lista G,B)

IDA*

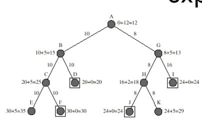
33

- …
  - Iteração 2



    - G foi expandido
    - Limite Actual = f2 = 13
    - Como f(H)=18>13, H é excluído
    - Como f(I)=24>13, I é excluído, apesar de ser Nó Objectivo
    - Como o limite 13 não conduziu a nada, tomar o limite seguinte:

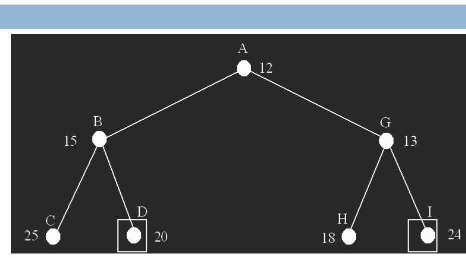
    Limite Iteração Seguinte = f3 = min(fNósNãoExpandidos) = f(B) = 15
    - Como em termos de f ao nó G se segue B (e G já foi expandido), então AB é o caminho escolhido pelo A* em seguida



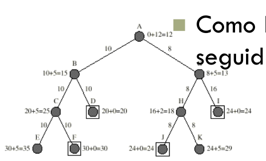
IDA*

34

- …
  - Iteração 3



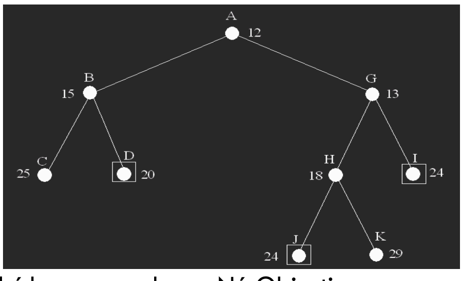
    - B foi expandido
    - Limite Actual = f3 = 15
    - Como f(C)=25>15, C é excluído
    - Como f(D)=20>15, D é excluído, apesar de ser Nó Objectivo
    - Como o limite 15 não conduziu a nada, tomar o limite seguinte:
    - Limite Iteração Seguinte = f4 = min(fNósNãoExpandidos) = f(H) = 18
    - Como H é o nó de menor f ainda por expandir, AGH é usado em seguida



IDA*

35

- ...
  - Iteração 4
    - H foi expandido
    - Limite Actual = f4 = 18

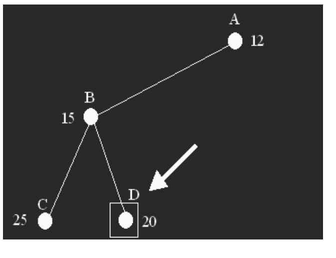


    - Como f(J)=24>18, J é excluído, apesar de ser Nó Objectivo
    - Como f(K)=29>18, K é excluído
    - Como o limite 18 não conduziu a nada, tomar o limite seguinte:
    - Limite Iteração Seguinte = f5 = min(fNósNãoExpandidos) = f(D) = 20
    - Como a ordem de expansão imposta pelo A* era (H,D,C,I) e o nó H já foi expandido, em seguida usa-se o caminho ABD

IDA*

36

- ...
  - Iteração 5
    - Limite Actual = f5 = 20



    - Mas, antes de se expandir D, verifica-se que ele é nó objectivo ...

... e que se encontra dentro do limite actual, porque f(D)=20 e f5=20

... Assim, D é a solução (óptima) procurada

IDA*

37

- …
  - Características da Pesquisa IDA*:
    - É completa
    - É óptima
    - Por ser baseada na pesquisa em profundidade, o requerimento de memória é baixo e pode ser **aproximado por b.d** (b=branching factor, d=profundidade da solução)

SMA*

38

- SMA*
  - tenta utilizar apenas a memória disponível para resolver um problema.
    - É completo e óptimo desde que a memória possibilite a sua execução completa
    - Se a memória estiver toda utilizada devido às expansões efectuadas, "esquece" os nós menos promissores (os de valor de f mais elevado), usando o espaço assim libertado para o resultado de outras expansões
    - …

SMA*

39

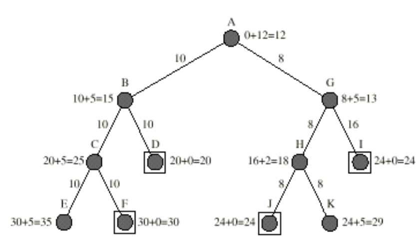
- …
  - O nó a expandir é o de menor valor de f
  - Porém, **quando se expande esse nó, adiciona-se-lhe apenas um sucessor em cada iteração**

  - Quando um nó se encontrar completamente expandido, o seu custo estimado, f, é actualizado com o mínimo dos valores de f dos seus nós filhos da iteração.

SMA*

40

- Exemplo

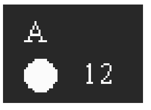


  - D, F I e J são nós que contêm o estado objectivo
  - Suponha-se que a **memória só tem capacidade para 3 nós.**

SMA*

41

- …
  - Estado Inicial



SMA*

42

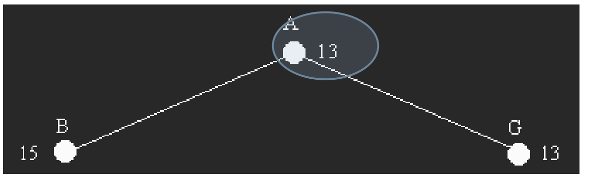
- …
  - Iteração 1

• Iniciar a expansão do nó A adicionando-lhe o primeiro dos seus filhos, B

SMA*

43

- ...
  - Iteração 2



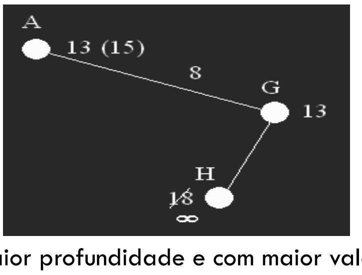
Como ainda só há 2 nós em memória (< limite = 3), continuar a expansão do nó A, adicionando-lhe o segundo dos seus filhos, G

• O nó A encontra-se agora completamente expandido: actualiza-se o valor f do nó pai com o mínimo dos valores f dos seus filhos, f(A)=f(G)=13

SMA*

44

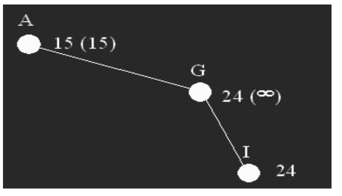
- ...
  - Iteração 3



- Elimina-se o nó de maior profundidade e com maior valor de f (nó B) e o seu valor, f(B)=15, é guardado em A (entre parêntesis)
- H não é objectivo!
  - Por outro lado, para expandir H seria preciso mais memória, uma vez que em H já há 3 nós ocupados
  - Portanto, o caminho nunca poderá ser construído através de H: Marca-se H com f(H)=∞

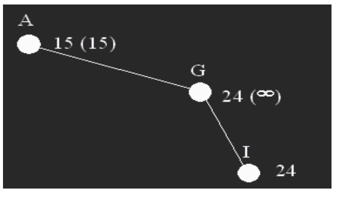
SMA*

- …
  - Iteração 4



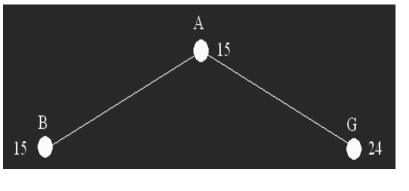
SMA*

- …
  - Iteração 5



I é um estado objectivo, mas como f(I)=24 > f(A)=15, é possível que haja uma solução melhor, e portanto a solução AGI não é considerada! Portanto, há que expandir A “para o outro lado”.

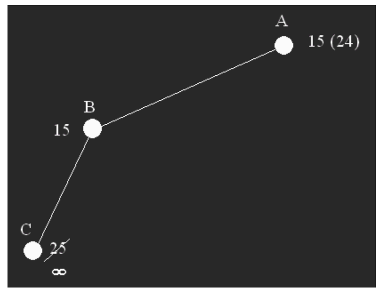
O nó B pode é gerado (pela 2ª vez) e adicionado a A



SMA*

47

- …
  - Iteração 6

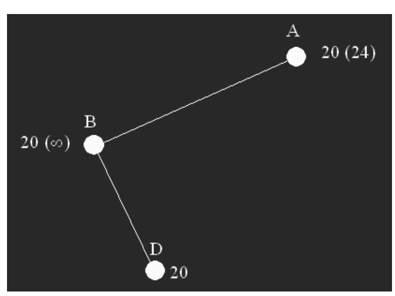


- C (tal como H no 3º passo) não é objectivo e origina a presença de 3 nós em memória, portanto, o caminho nunca poderá ser construído através de C:
- Marca-se C com f(C)=∞

SMA*

48

- …
  - Iteração 7

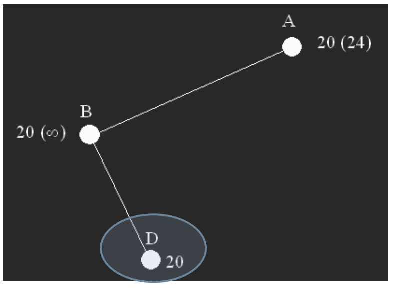


o nó B foi completamente expandido. Quando isto acontece, actualiza-se o valor f do nó pai com o mínimo dos valores f dos seus filhos , f(B)=min(∞,20)=20

Como f(B) muda, f(A) também tem de ser actualizado. Como f(G)=24 ficou guardado em A entre parêntesis, o mínimo f entre os filhos de A é min(20,24)=20

SMA*

- ...
  - Iteração 8



- D é agora o nó seleccionado para expansão, porém, como é nó objectivo e f(D) não é maior que nenhum dos valores de f noutros nós da árvore, **o caminho ABD é a solução**