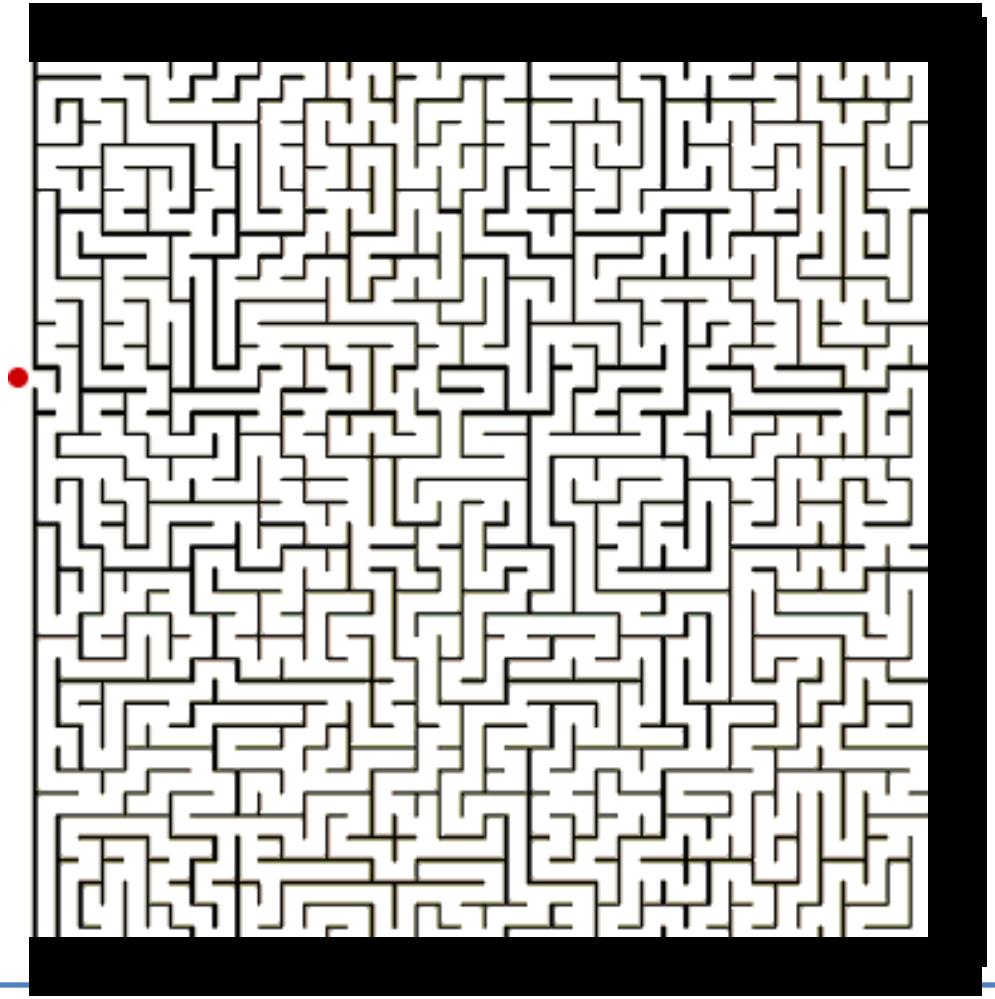
Inteligência Artificial

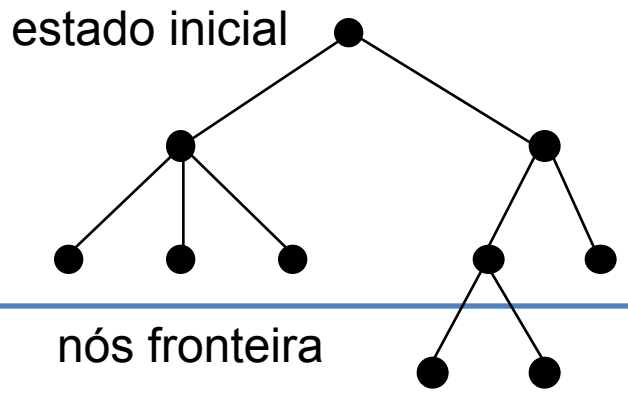
Métodos de procura

Métodos de procura

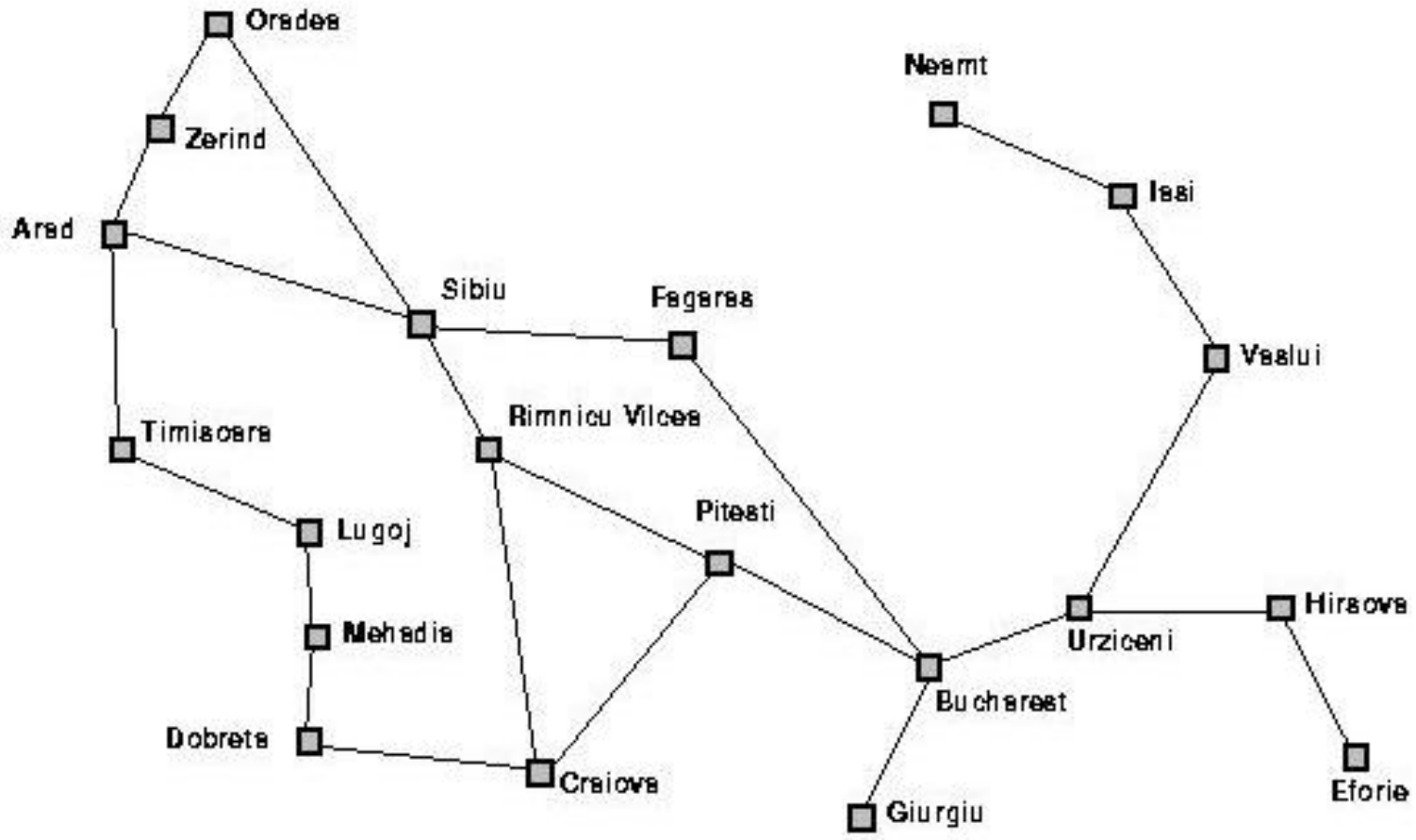


Métodos de procura

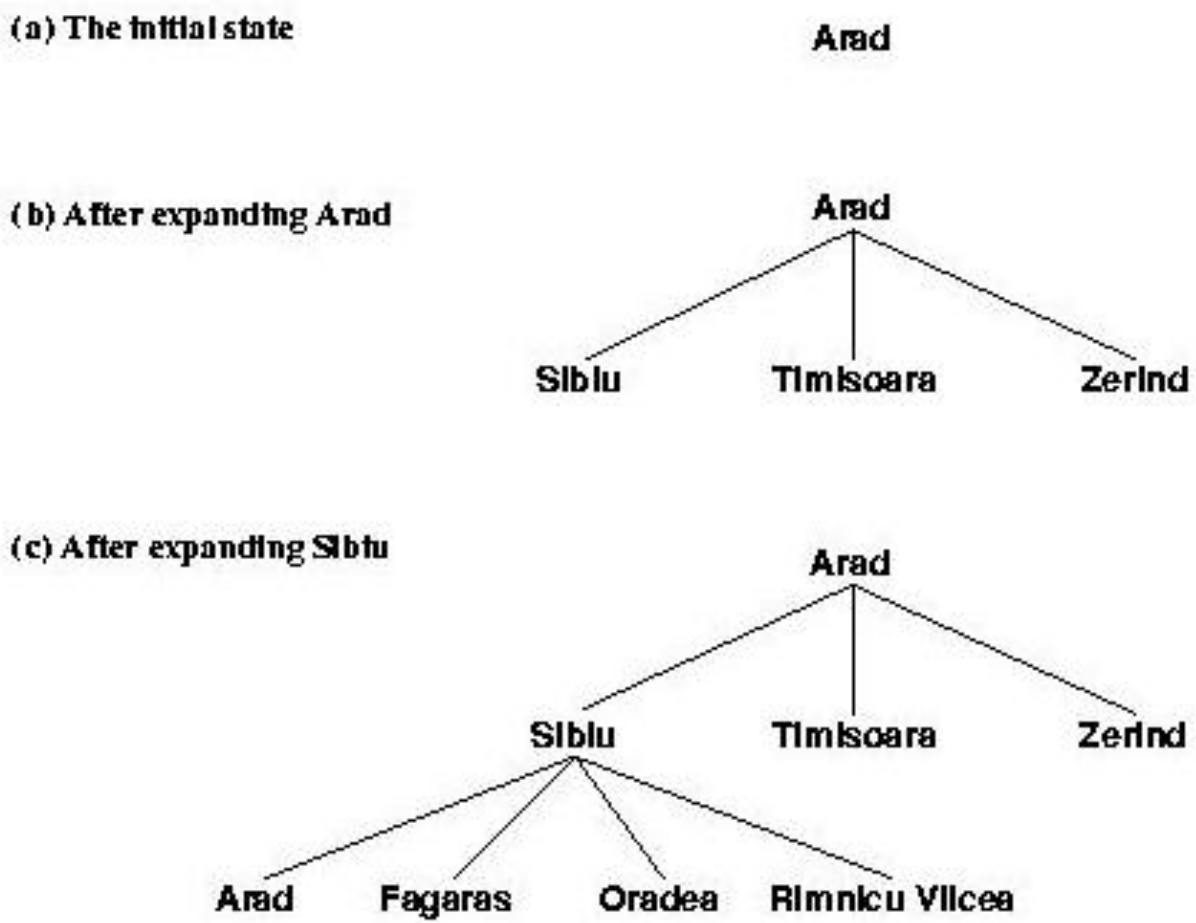
- O processo de pesquisa constrói uma árvore onde:
  - Raiz é o estado inicial
  - Os nós folha são nós que ainda não foram expandidos ou que não têm sucessores
- Ao conjunto de nós que ainda não foram expandidos dá-se o nome de **fronteira**



Métodos de procura



Métodos de procura



Métodos de procura

- O próximo nó (estado) a ser expandido depende da estratégia que é seguida por cada método de procura
  - Dependendo do método também temos diferentes estruturas de dados implementadas:
    - pilha, fila, etc.
- A evitar pelos métodos de procura:
  - Voltar ao estado anterior
  - Gerar ciclos
  - Gerar estados repetidos

Métodos de procura

- Critérios de avaliação de um método de procura:
  - COMPLETO : Encontra a solução quando ela existe.
  - ÓPTIMO : Encontra a de melhor qualidade quando existem várias soluções.
  - COMPLEXIDADE TEMPORAL : Quanto tempo demora a encontrar a solução?
  - COMPLEXIDADE ESPACIAL : Quanta memória é necessária para efectuar a procura ?

Métodos de procura

- Tipos de métodos de procura:
  - CEGOS OU NÃO-INFORMADOS
    - Não existe informação sobre o custo do caminho do estado actual para o objectivo.
  - HEURISTICOS OU INFORMADOS
    - Caso contrário

Métodos de procura

- Os métodos de procura informados são, geralmente mais eficazes do que os métodos não informados
- No entanto, uma vez que existem muitos problemas onde não existe informação adicional disponível, os métodos não-informados são também importantes

Métodos de procura

- Procura **CEGA**
  - Largura Primeiro
  - Custo Uniforme
  - Profundidade Primeiro
  - Profundidade Limitada
  - Aprofundamento Progressivo
  - Bidirecional

Métodos de procura

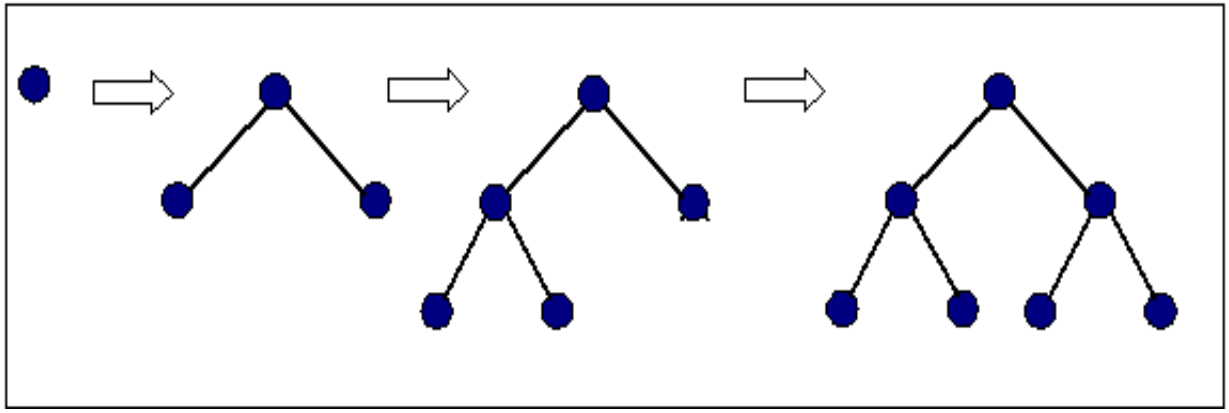
- Procura **Heurística**
  - Diferentes heurísticas :
    - Custo caminho parcial
    - Custo estimado para a solução
  - Famílias

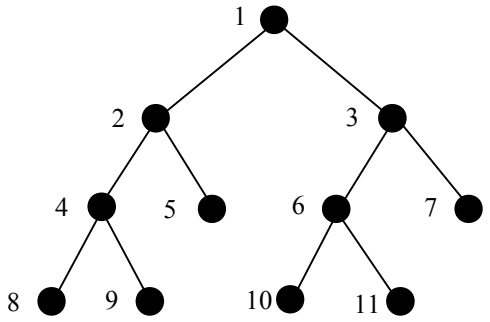
Métodos de procura

- Procura **Heurística**
  - O Melhor Primeiro
    - Procura Sôfrega
    - A*
  - Memória Limitada
    - IDA*
    - SMA*
  - Melhoria Progressiva
    - Trepa Colinas
    - Esfriamento Simulado

Métodos de procura

- **Procura Em Largura Primeiro:**
  - **Principio : Os nós do nível n são todos expandidos antes dos nós do nível n+1**





Métodos de procura

- **Características:**
  - **Completo :** Sim
  - **Óptimo :** Geralmente Não (Só é óptimo se o custo de um caminho for uma uma função não decrescente do nível do nó)
  - **Complexi. Esp. :** $O(b^n)$ $1+b+b^2$
  - **Complexi. Temp :** $O(b^n)$
    - b- factor de ramificação
    - n- profundidade da árvore

Métodos de procura

- Notas as retirar em relação a árvores extensas com um grande factor de ramificações:
  - A quantidade de memória necessária é um problema maior do que o tempo de execução.
  - O tempo de execução mesmo assim ainda é considerável.
  - Tempo elevado se a solução tem muitos passos, pois verifica todas as possibilidades mais curtas
  - Factor de ramificação grande pode tornar impossível a sua utilização

Métodos de procura

Nível	Nós	Tempo	Espaço
0	1	1ms	10 bytes
5	4681	4,68s	45 Kbytes
10	$153*10^6$	1,9 Dias	1,5 Gbytes
15	$5*10^{12}$	175 Anos	50 Tbytes

b=8, 1ms por nó, 10 bytes por nó

Métodos de procura

- A pesquisa em Largura Primeiro encontra o nó objectivo que está mais à superfície na árvore, mas pode não ser a solução de menor custo para uma função de custo de caminho geral.

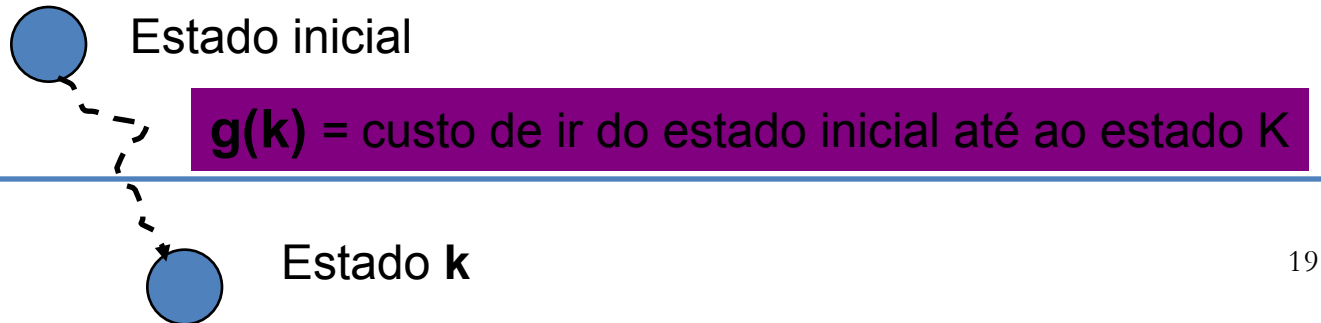
Métodos de procura

Métodos de procura

- Procura por custo Uniforme:
  - **Princípio :** Expandir sempre o nó na fronteira da árvore com menor custo.
  - Função g(n): custo do caminho percorrido
  - Quando na procura por custo uniforme

    g(n) = profundidade(n)

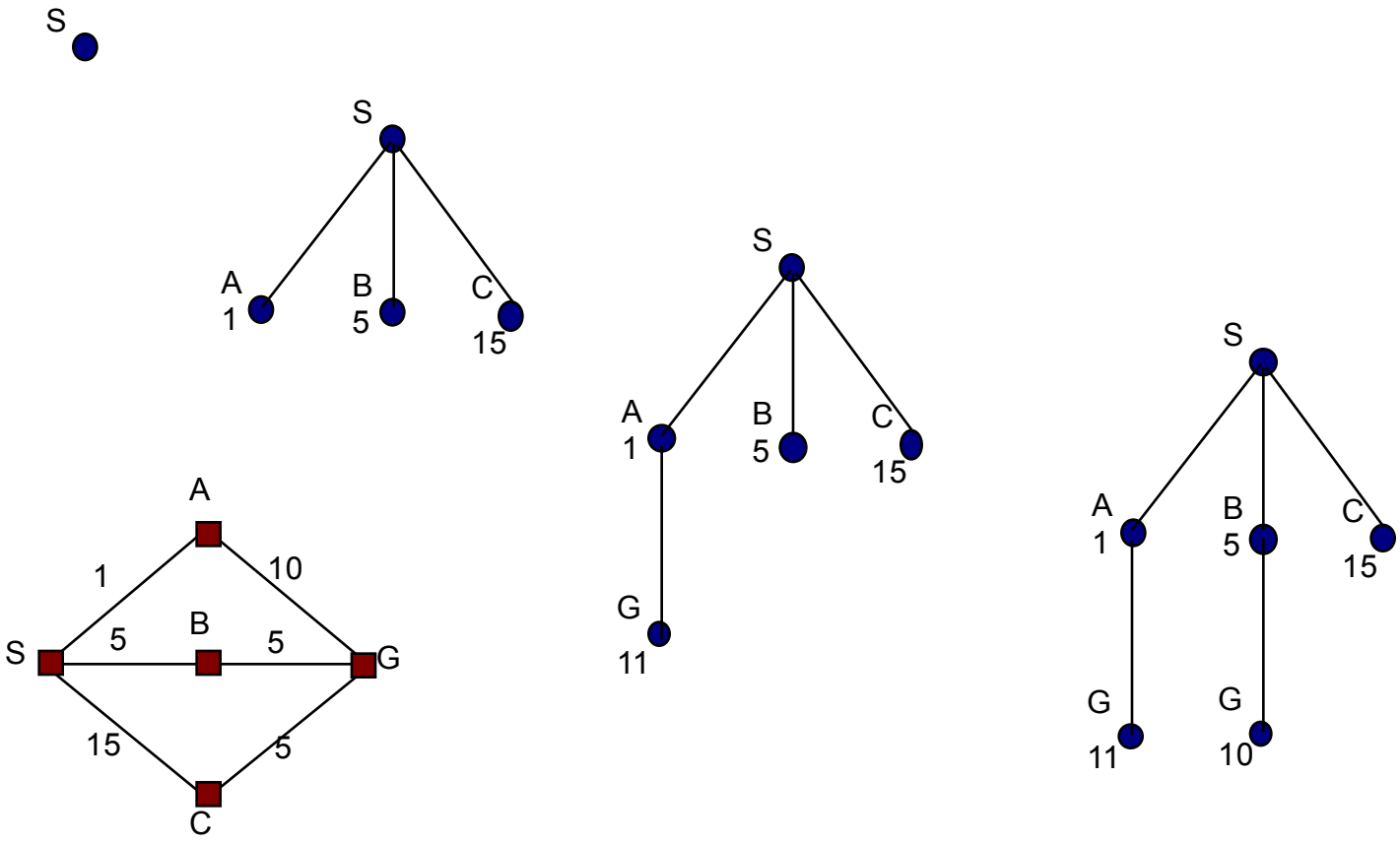
    temos procura em Largura Primeiro.



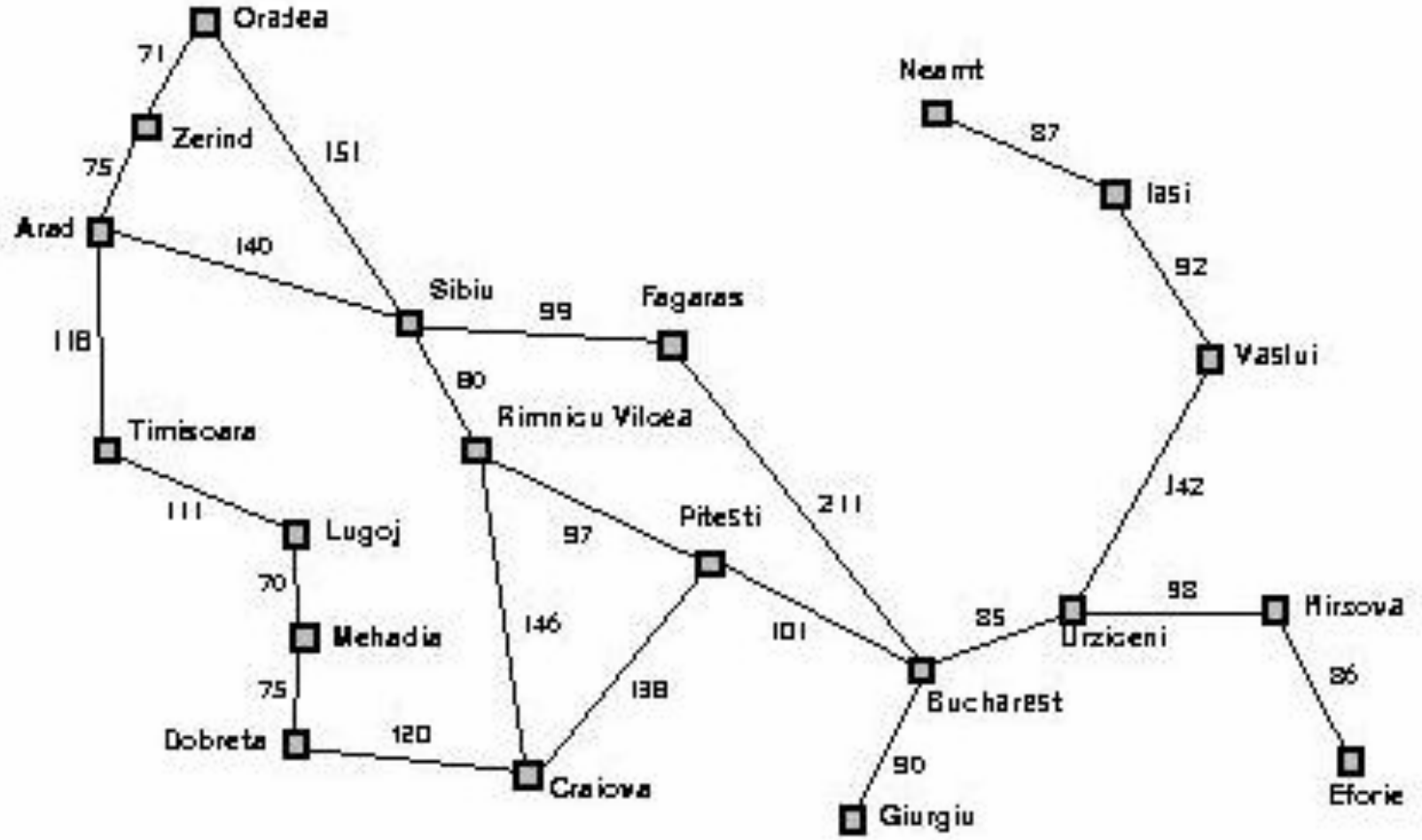
Métodos de procura

- Características:
  - **Completo:** Sim
  - **Óptimo :** Sim c/ g(succ(n)) >= g(n)
    - Se existir algum operador que tenha custo negativo deixa de ser óptimo
  - Complexidade temporal: $O(b^d)$
  - Complexidade espacial: $O(b^d)$
    - b - factor de ramificação
    - d - nível que se encontra a solução na árvore

Métodos de procura

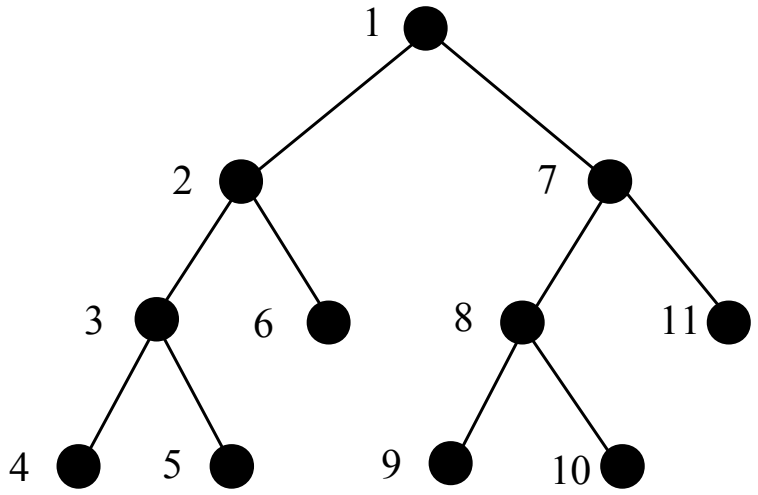


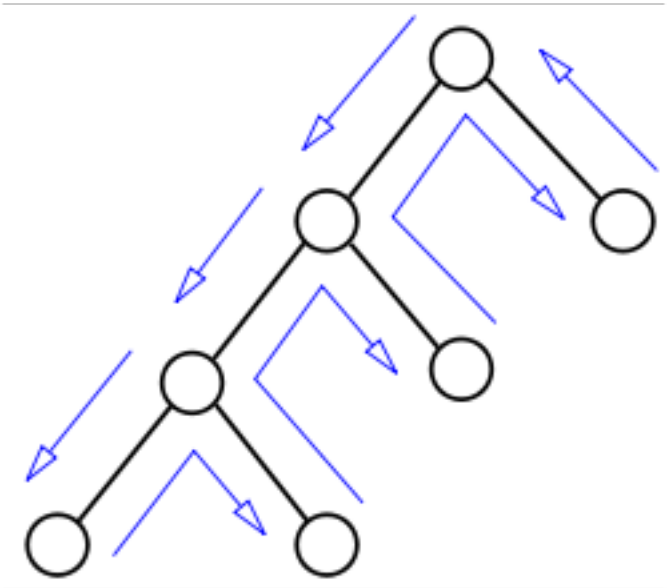
Métodos de procura



Métodos de procura

- Procura em Profundidade
  - **Princípio**: expandir sempre o nó o mais profundo possível





Métodos de procura

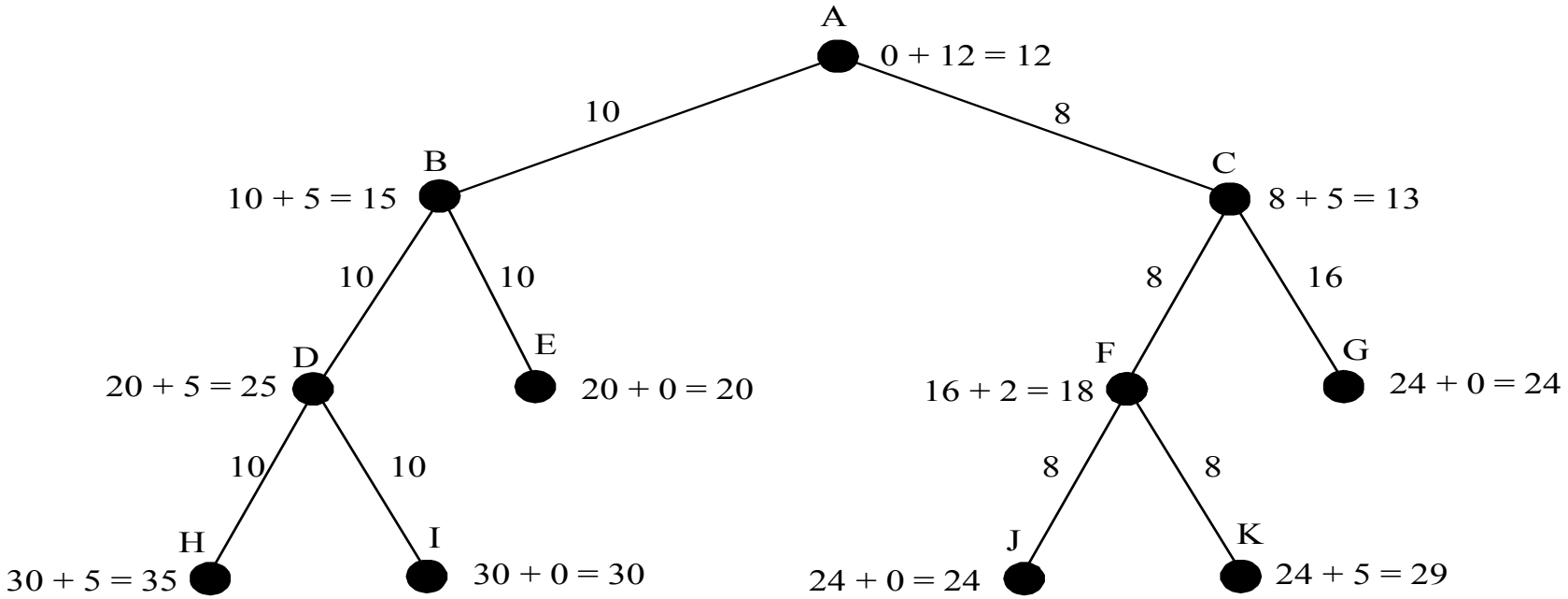
- Estrutura de Dados : Pilha
- **Características :**
  - **Completo : Não**
  - **Óptimo : Não**
  - **Complexidade Espacial:** O(b*n) - Este método é muito modesto em exigências de memória.
    - Se existirem muitas soluções, este método pode ser mais rápido do que o de pesquisa em Largura Primeiro, porque tem a possibilidade de achar a solução explorando somente uma pequena porção do espaço de pesquisa.
    - Mas contudo, este método tem de passar por todos os caminhos até ao nível d-1 antes de considerar algum do nível d.

Métodos de procura

- **Características :**
  - **Complexidade Temporal :** $O(b^n)$
    - **Problemas :**
      - **com arvores de pesquisa muito profundas ou infinitas este algoritmo pode seguir por um caminho abaixo na árvore e não conseguir regressar ao topo**
      - **podendo inclusivamente ficar preso em ciclos não podendo assim encontrar uma solução**
      - **ou eventualmente encontrar um caminho para uma solução que não seja o caminho ideal.**

Métodos de procura

- Exercício



Métodos de procura

- Procura em Profundidade Limitada:
  - Visa resolver o problema da não-completude do método de pesquisa em profundidade primeiro.
  - **Princípio :** Expansão dos nós em profundidade primeiro, até um nível ***l*** imposto

Métodos de procura

- No exemplo de procura de um percurso entre duas cidades, apresentado anteriormente o nº de cidades é de 20. Assim se existir uma solução esta deve ter um comprimento máximo de 19.

Métodos de procura

- Características:
  - **Completo:** Sim com l >= d
    - d- nível a que se encontra a solução na árvore.
    - l- limite de aprofundamento da pesquisa
  - **Óptimo :** Não
  - Complexidade temporal: $O(b^l)$
  - Complexidade espacial: $O(b * l)$
    - b- factor de ramificação

Métodos de procura

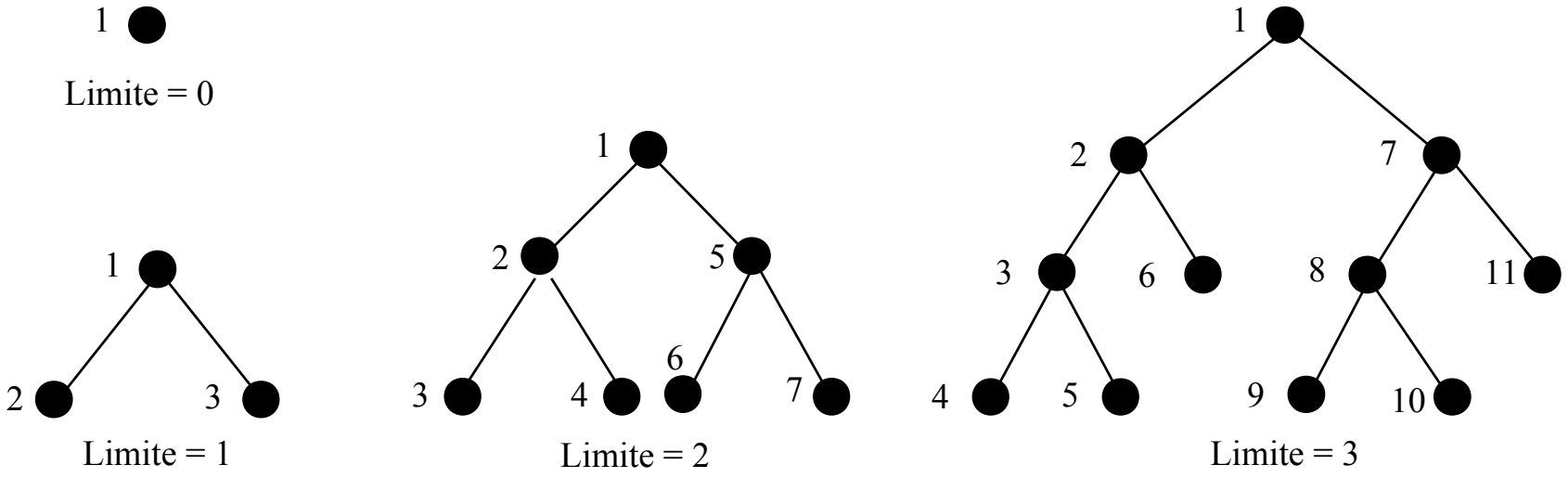
- A parte complicada acerca do método de Aprofundamento Limitado é:

  Estabelecer um bom limite de profundidade de pesquisa .
- Escolhemos **19 como um limite** de profundidade óbvio para o problema do mapa da Roménia, mas de facto, se olharmos o mapa atenciosamente, vemos que cada cidade pode ser alcançada **a partir de qualquer outra cidade, no máximo em 9 passos**.
- Este número, conhecido como **diâmetro** do espaço de estados, é um melhor limite de profundidade, que leva a uma maior eficiência na pesquisa em profundidade.
- Contudo para a maior parte dos problemas, nós não conseguimos estabelecer um bom limite até termos resolvido o problema.

Métodos de procura

- Procura por Aprofundamento Progressivo
  - Principio : Para contornar o problema da escolha do melhor limite de profundidade este algoritmo, experimenta todos os limites possíveis : Primeiro até ao nível 0, depois nível 1, nível 2 , ...

Métodos de procura



Métodos de procura

- Com efeito a pesquisa por aprofundamentos progressivo combina os benefícios de:
  - procura em profundidade primeiro
  - largura primeiro.
- É Completo e poderá ser óptimo, tal como a pesquisa em largura primeiro., mas é modesto em necessidade de memória tal como o de em profundidade primeiro.

Métodos de procura

- Este método pode parecer penoso e repetitivo em termos de tempo porque muitos nós são expandidos várias vezes. Contudo, para a maior parte dos problemas, a diferença de preço deste processo para o processo anterior não é assim tão significante como isso.
  - Intuitivamente, a razão é que numa árvore de pesquisa exponencial, maior parte dos nós estão no nível inferior, e assim, não é muito significante que sejam os níveis superiores a serem expandidos múltiplas vezes.

Métodos de procura

- Características:
  - **Completo :** Sim
  - **Óptimo :** Não* (*Sim, se custo caminho entre nós =1)
  - Complexidade temporal: $O(b^d)$
  - Complexidade espacial: $O(b * d)$
    - b- factor de ramificação
    - d- nível a que se encontra a solução na árvore.

Métodos de procura

Preço do processo de expansões múltiplas (exemplo: $b = 10$; $d = 5$)

- **Profundidade primeiro**

  $1 + b + b^2 + b^3 + b^4 + b^5 =$

  $1 + 10 + 100 + 1000 + 10000 + 100000 = 111.111$

- **Aprofundamento progressivo**

  $(d + 1) * 1 + d * b + (d - 1)\, b^2 + \ldots + 1 * b^5 =$

  $6 + 50 + 400 + 3000 + 20000 + 100000 = 133.456$

- **Em geral**, o aprofundamento progressivo é o método a utilizar quando o espaço de pesquisa é extenso e a profundidade a que se encontra a solução é desconhecida

Métodos de procura

- Procura Bidireccional
  - **Principio :** Expansão dos nós a partir do estado inicial e do(s) estado(s) objectivo.
  - A pesquisa é efectuada , simultaneamente a partir do estado inicial e para trás a partir do estado objectivo, e pára quando as duas pesquisas se encontram ao meio.

Métodos de procura

- Para problemas em que o factor de ramificação b é o mesmo em ambas as direcções a procura bidireccional pode fazer uma grande diferença em relação aos algoritmos anteriores.
- Se nós assumirmos, como é habitual, que existe uma solução de ordem d, então a solução será encontrada em O($2b^{d/2}$) = O($b^{d/2}$)passos, porque as pesquisas nos 2 sentidos têm que fazer somente meio-caminho .
- Isto soa bem, mas existem certas condições a ter em conta

Métodos de procura

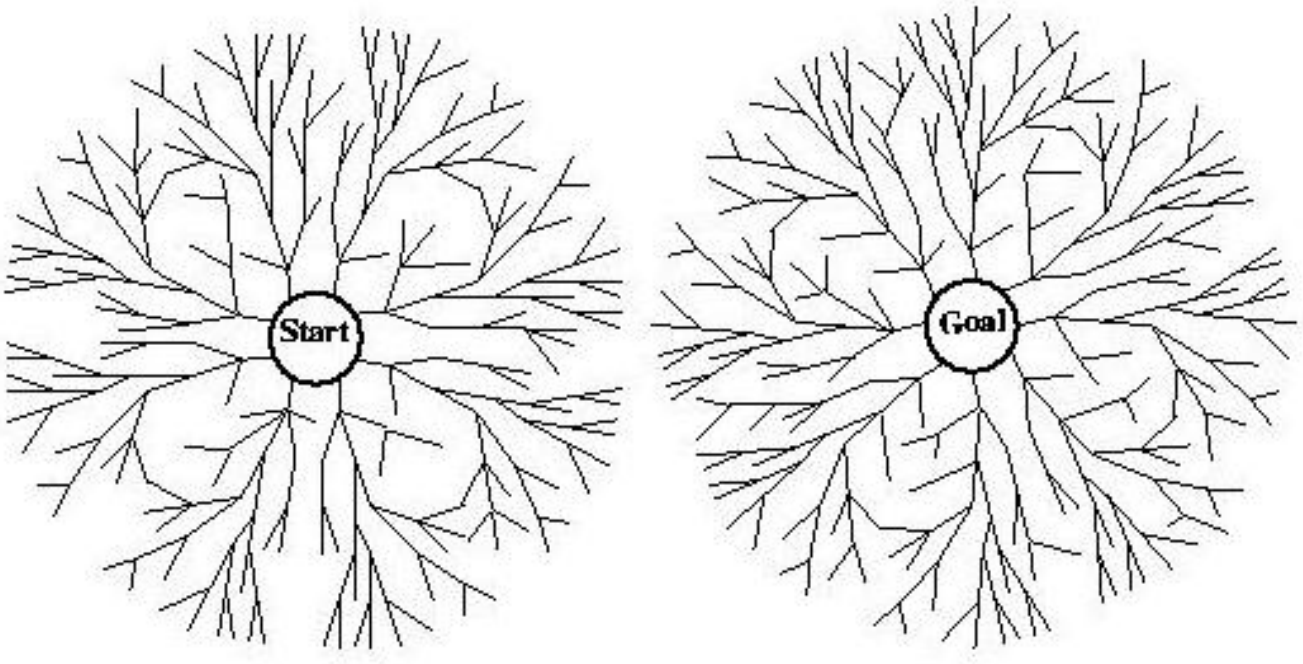
- Neste método é preciso ter em conta:
  - A principal questão : O que significa andar para trás partindo do objectivo ?
    - Definimos predecessores de um nó **n** como sendo todos os nós que têm **n** como sucessor.
    - Procura para trás significa gerar predecessores sucessivos começando do nó objectivo.
  - Quando todos os operadores estiverem invertidos, os conjuntos de predecessores e sucessores são idênticos. Contudo, por vezes é difícil calcular os predecessores .

Métodos de procura

- Neste método é preciso ter em conta:
  - E se existirem muitos possíveis estados objectivos ?
    - Se existir uma lista explicita de estados objectivos, então nós podemos aplicar a função predecessor ao conjunto de estados tal como aplicamos a função sucessor na pesquisa de vários estados.
    - Se só tivermos uma descrição do conjunto, pode ser possível definir um conjunto de estados que poderiam gerar o conjunto de objectivos, mas é uma coisa muito complicada de fazer.
    - Por exemplo quais são os estados que são predecessores do objectivo "xeque-mate" no xadrez?
  - Tem que existir uma maneira eficiente de verificar a ocorrência do mesmo nó em ambas as direcções de procura.
  - Precisamos de decidir que tipo de pesquisa efectuar em cada metade da árvore.

Métodos de procura

- De maneira a garantir que as duas pesquisas se encontram, os nós de uma delas devem ser retidos em memória (Pesquisa em Largura Primeiro).



Métodos de procura

- Características:
  - **Completo:** Sim (quando aplicável)
  - **Óptimo :** Sim (quando aplicável)
  - Complexidade espacial: $O(b^{d/2})$
  - Complexidade temporal: $O(b^{d/2})$
    - b- Factor de ramificação
    - d- nível a que se encontra a solução na árvore

Métodos de procura

- As estratégias de pesquisa não-informadas conseguem achar soluções para problemas gerando sistematicamente novos estados e testando-os para ver se correspondem ao objectivo.
- Infelizmente, estas estratégias são incrivelmente ineficientes na maior parte dos casos.
- As estratégias de procura informadas (aquelas que utilizam conhecimento específico do problema) conseguem encontrar soluções, de um modo mais eficiente.
- Mostra também como podem ser resolvidos os problemas de optimização.

Métodos de procura

- Diferentes heurísticas :
  - Custo caminho parcial
  - Custo estimado para a solução

Métodos de procura

- Procura **Heurística**
  - O Melhor Primeiro
    - Procura Sôfrega
    - A*
  - Memória Limitada
    - IDA*
    - SMA*
  - Melhoria Progressiva
    - Trepa Colinas
    - Esfriamento Simulado

Métodos de procura

- O melhor primeiro:
  - A família de Algoritmos “O melhor primeiro”, tem diferentes funções de avaliação.
  - Porque estes algoritmos tentam encontrar soluções de baixo custo, logo usam valores estimados de custo da solução e tentam minimiza-los.

Métodos de procura

- O melhor primeiro:
  - Já vimos uma dessa funções de avaliação: A função de custo de caminho *g()* que decide qual o caminho a estender.
  - Esta medida contudo, não leva a pesquisa directa ao objectivo.
  - Com vista à optimização da pesquisa, a avaliação deve incorporar uma estimativa do custo do caminho de um estado para o estado objectivo mais próximo *h()*.

Métodos de procura

- O melhor primeiro:
  - Estes algoritmos ordenam a lista de nós a expandir por ordem crescente do valor de uma função de avaliação *f()* que incorpora informação específica do domínio, e escolhem depois o melhor nó, isto é, o primeiro nó da lista

Métodos de procura

- Procura sôfrega
  - Uma das melhores estratégia de procura do tipo "o melhor primeiro" é minimizar o custo estimado para se atingir o objectivo.
  - Isto é o nó cujo estado é julgado estar mais perto do objectivo é expandido primeiro.
  - Para muitos problemas, o custo para se atingir o objectivo a partir de um determinado estado pode ser estimado mas não pode ser determinado exactamente.

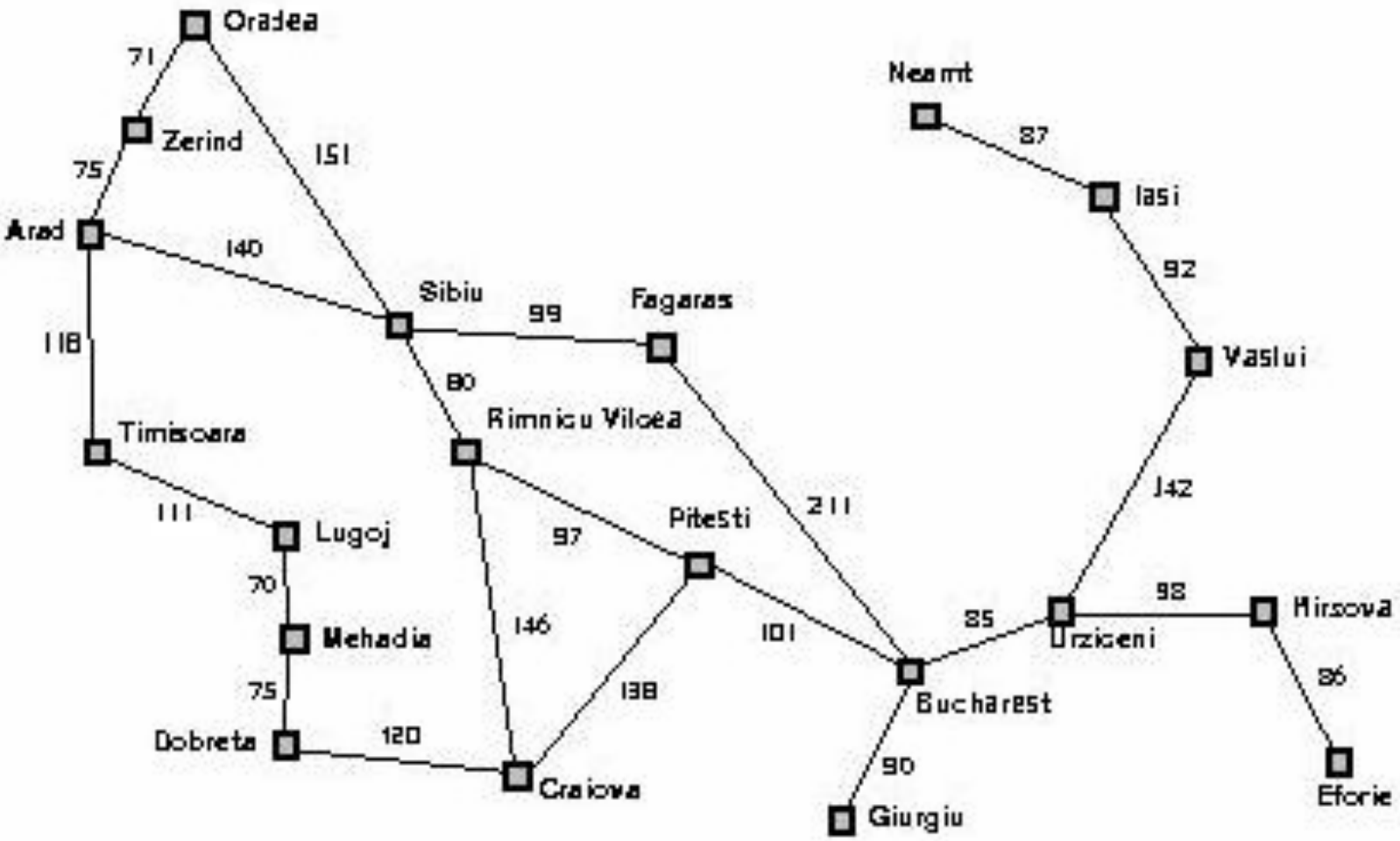
Métodos de procura

- Procura sôfrega
  - A função que calcula tal estimativa de custo é denominada de **função heurística**.
  - *h(n)* = custo estimado do caminho mais barato a partir do estado no nó *n* para o estado objectivo.
  - *h(n)* = 0 se *n* é objectivo.
  - Uma função heurística é especifica para cada problema.

Métodos de procura

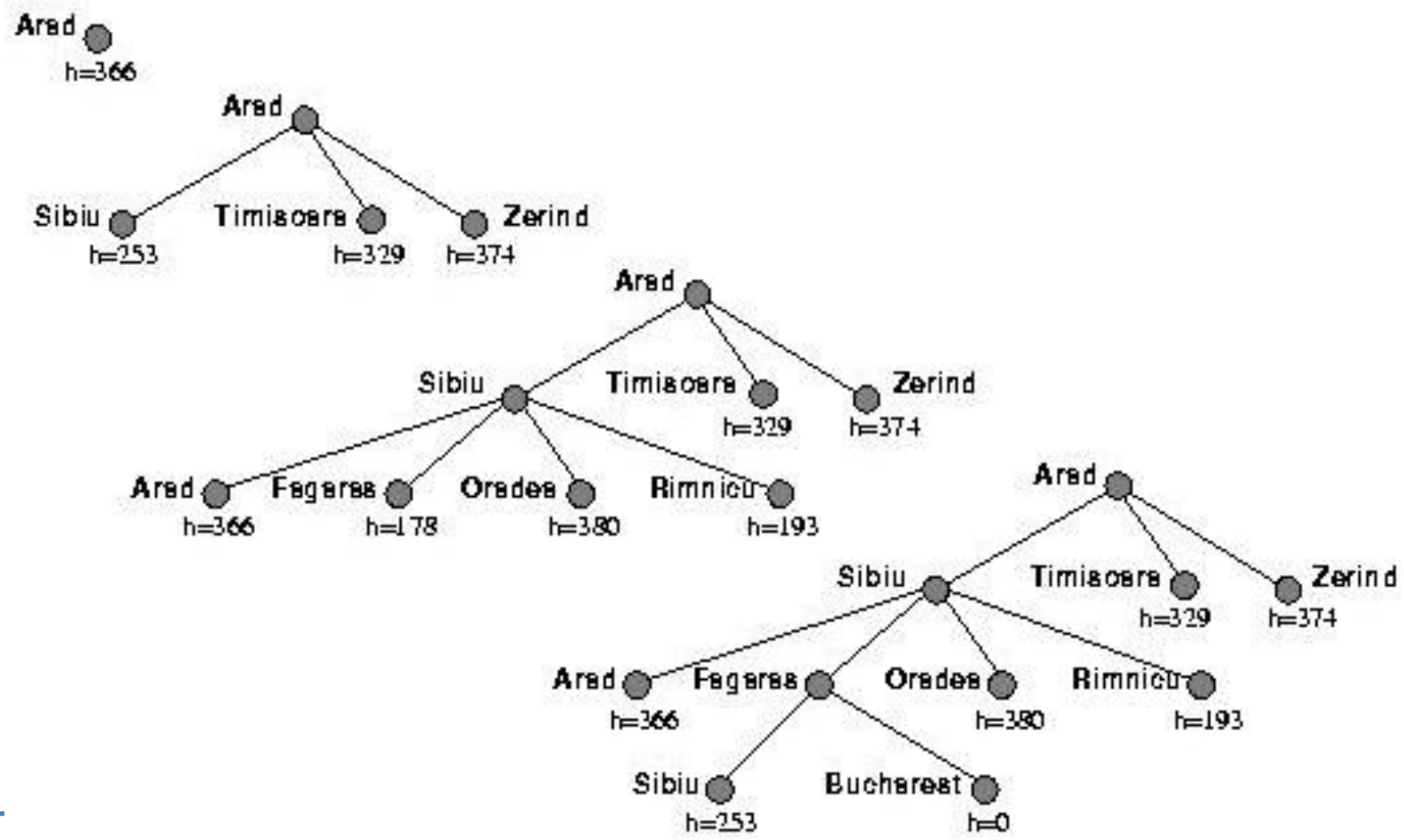
- Procura sôfrega
  - **Principio:** Expandir o nó cujo o custo estimado à solução é o menor.
  - *f(n)=h(n)*
  - h(n) = custo estimado do caminho mais barato a partir do estado no nó n até um nó objectivo.
  - Exemplo Estradas da Roménia

Métodos de procura



Distancia para Bucareste	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Métodos de procura



Métodos de procura

- Procura sôfrega
  - Para este problema, em particular, a heurística utilizada levou a um custo de procura mínimo: encontrou a solução sem ter que expandir algum dos nós que não estivesse no caminho encontrado.
  - Contudo, não é óptimo: havia um caminho mais curto para o objectivo.

Métodos de procura

- Procura sôfrega
  - Óptimo : Não
  - Completo : Não (pode andar em ciclos ou andar por um caminho infinito).
  - Complexidade Espacial: $O(b^m)$
    - pior dos caso- retêm todos os nós em memória
  - Complexidade Temporal: $O(b^m)$
    - Porque é semelhante ao algoritmo de pesquisa em profundidade primeiro, se encontrar um nó sem saída, volta para trás no caminho e reinicia a pesquisa escolhendo outro caminho.

m – profundidade máxima do espaço de pesquisa

Métodos de procura

- Procura sôfrega
  - Com uma boa heurística, a complexidade espacial e temporal, podem ser substancialmente reduzidas, dependendo esta redução da qualidade da função h.
  - Como a gula, é considerado como uns dos sete pecados mortais, este adjectivo aplica-se bastante bem a este algoritmo.
    - Pois ele tenta encontrar rapidamente a solução mas contudo, nem sempre encontra a solução óptima.
    - Este algoritmo de procura sôfrega é susceptível de falsas partidas, seguindo por caminhos sem saída
      - No caso de querermos ir de Iasis a Fagaras, a primeira escolha seria Neamt, a solução seria Iasi-Vaslui-Urziceni-Bucarest-Fagaras, assim , neste caso a heurística utilizada iria provocar a expansão desnecessária de nós.
    - E mais, se o algoritmo não estiver preparado para detectar estados repetidos, a solução andará em ciclos entre Neamt e Iasi.

Métodos de procura

	Faro
Aveiro	363
Braga	454
Bragança	487
Beja	99
C. Branco	280
Coimbra	319
Évora	157
Faro	0
Guarda	352
Leiria	278
Lisboa	195
Portalegre	228
Porto	418
Santarém	228
Setúbal	168
Viana	473
V. Real	429
Viseu	363

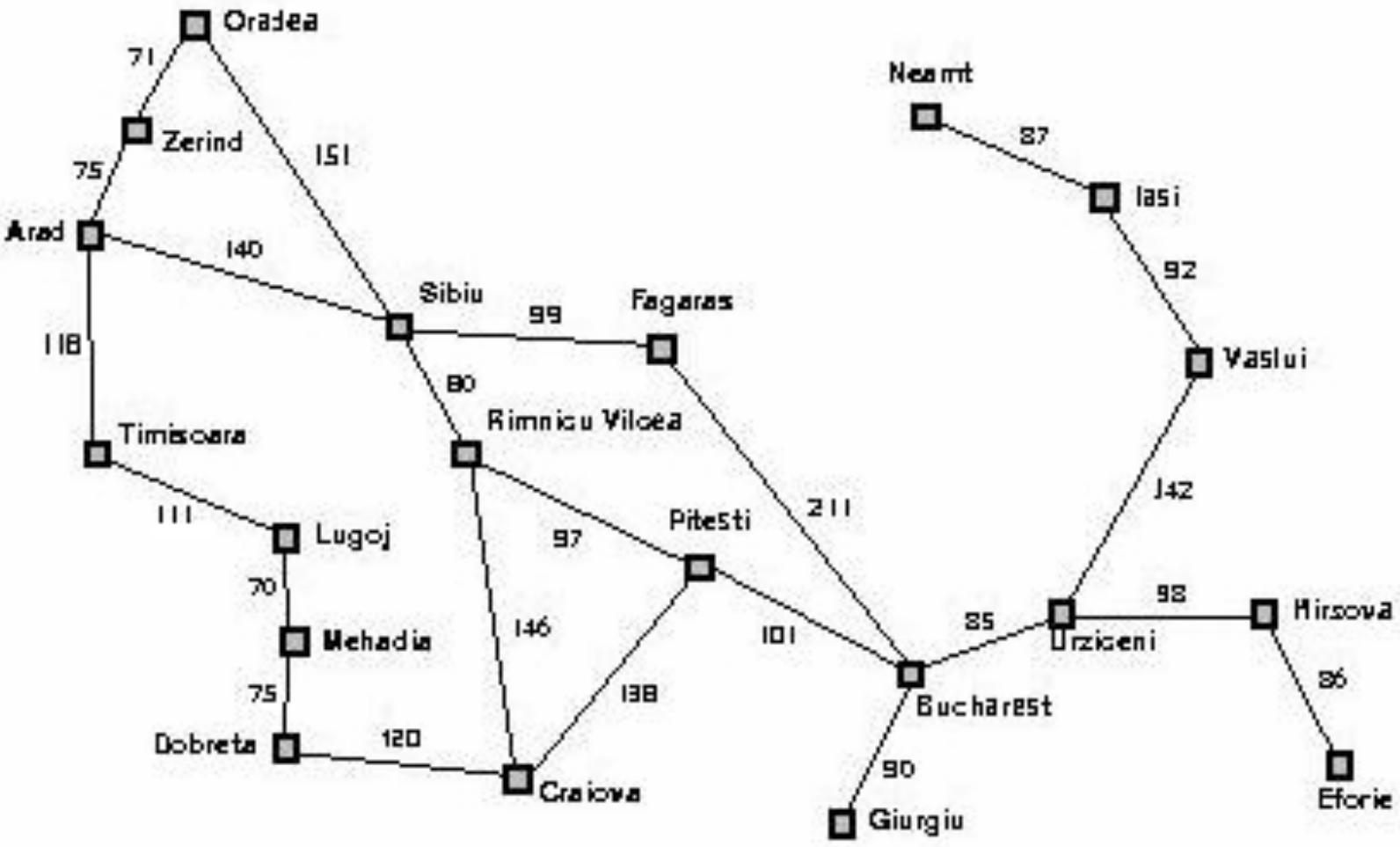
Métodos de procura

- A*
  - O algoritmo de procura sôfrega, minimiza o custo estimado para o objectivo, *h(n)*, e por isso reduz o custo de procura consideravelmente.
  - Infelizmente, não é óptimo, nem completo.
  - O algoritmo de custo uniforme, por outro lado minimiza também os custo do caminho, *g(n):* é óptimo e completo, mas pode ser bastante ineficiente.
  - Felizmente podemos combinar estas duas estratégias para conseguir as vantagens de ambas, combinado as duas funções de estimativa.

Métodos de procura

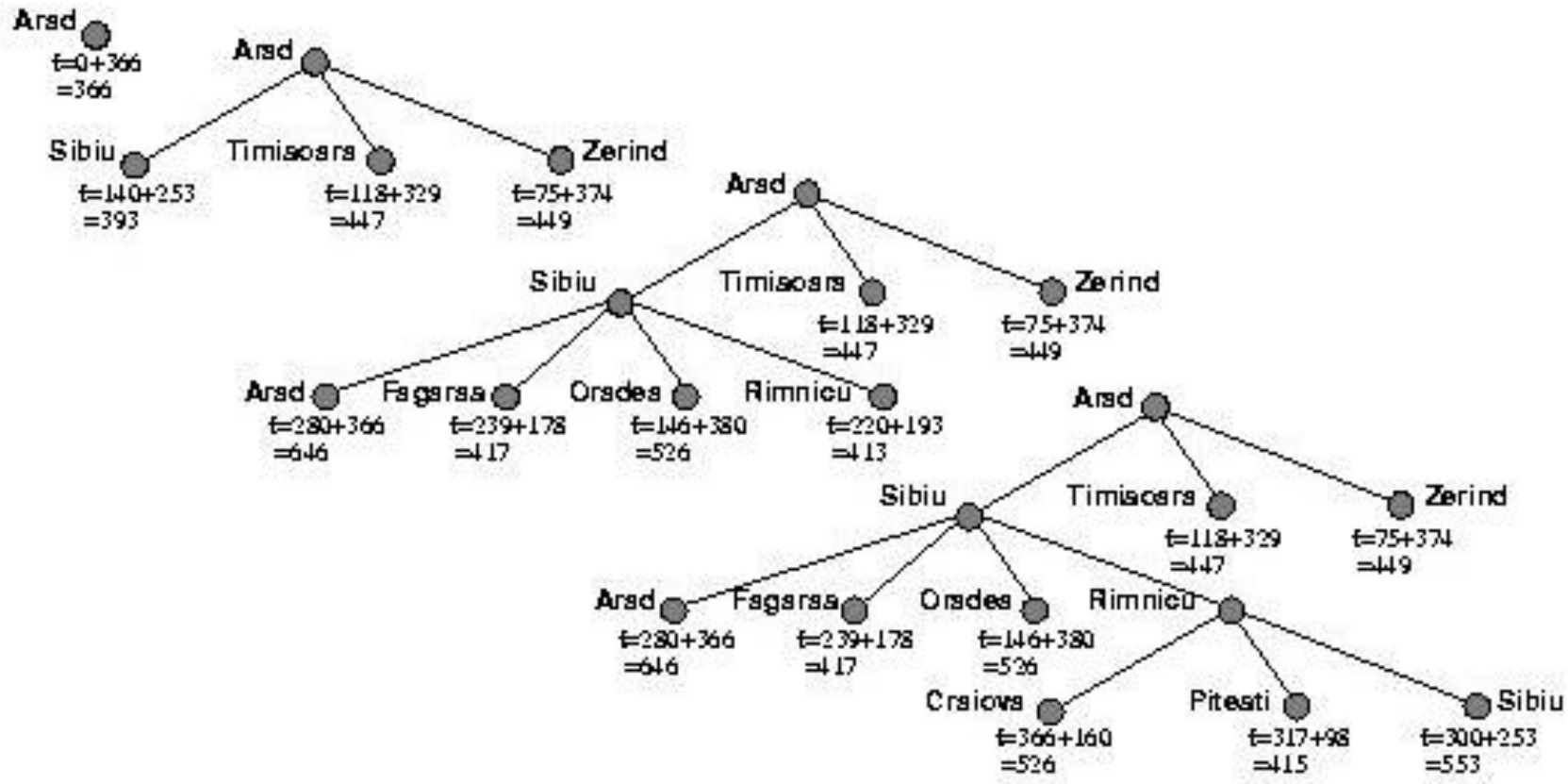
- A*
  - **Princípio**: tenta expandir o nó que pertence ao caminho com um menor custo associado
  - $f(n) = g(n) + h(n)$
  - *f(n)* = custo estimado da solução mais barata passando por *n*.
  - Assim a melhor solução passa por escolher primeiro o nó com o valor mais baixo de *f* .

Métodos de procura



Distancia para Bucareste	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Métodos de procura



Métodos de procura

- A*
  - **Óptimo :** Sim
    - Prova
  - **Completo:** Sim

Prova :
Seja G um nó objectivo óptimo com custo de caminho f*.
Seja G2 um nó objectivo sub-óptimo, ou seja, o nó objectivo com um custo de caminho g(G2) > f*
Imaginemos que A* seleccionou G2 da lista. Como G2 é um nó objectivo, o algoritmo terminaria a procura com uma solução sub-óptima. Mostremos que isso não é possivel: Consideremos que um nó n que é uma folha de um caminho para G.

Temos então f* >= f(n)
Mais, se o n não foi escolhido para a expansão para G2, temos de ter

f(n) >= f(G2)

combinando as duas anteriores temos :
f* >= f(G2)

mas como G2 é objectivo tem h(G2) = 0, assim f(G2)= g(G2) obtemos assim f* >= g(G2) Contradição.

Métodos de procura

- A*
  - O algoritmo de procura A* é completo, óptimo, e optimamente eficiente entre os algoritmos anteriores e ainda mais satisfatório.
  - Contudo, não significa que seja a resposta a todas as nossas necessidades de pesquisa.
  - Continuam a existir problemas de espaço de memória.
  - Surgem assim os algoritmos de memória limitada.

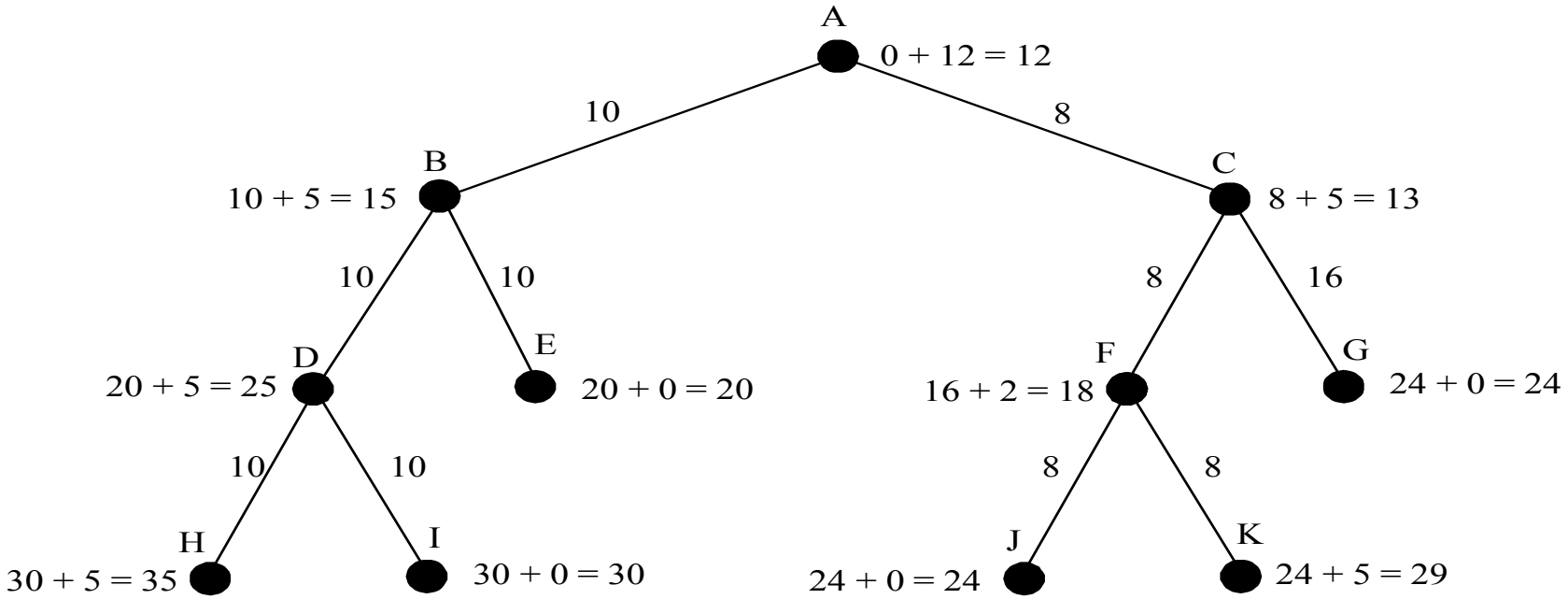
Métodos de procura

Aveiro	Porto (68)	Viseu (95)	Coimbra(58)	Leiria (115)
Braga	Viana C.(48)	V. Real (106)	Porto (53)	
Bragança	V. Real (137)	Guarda (202)		
Beja	Évora (78)	Faro (152)	Setúbal (142)	
C. Branco	Coimbra (159)	Guarda (106)	Portalegre (80)	Évora(203)
Coimbra	Viseu (96)	Leiria (67)		
Évora	Lisboa (150)	Santarém (117)	Portalegre (131)	Setúbal (103)
Faro	Setúbal (249)	Lisboa (299)		
Guarda	V. Real (157)	Viseu (85)		
Leiria	Lisboa (129)	Santarém (70)		
Lisboa	Santarém (78)	Setúbal(50)		
Porto	V. Castelo (71)	V. Real (116)	Viseu (133)	
V. Real	Viseu (110)			

	Faro
Aveiro	363
Braga	454
Bragança	487
Beja	99
C. Branco	280
Coimbra	319
Évora	157
Faro	0
Guarda	352
Leiria	278
Lisboa	195
Portalegre	228
Porto	418
Santarém	228
Setúbal	168
Viana	473
V. Real	429
Viseu	363

Métodos de procura

- Exercício



Métodos de procura

- Memória Limitada
  - IDA*
  - SMA*

Métodos de procura

- IDA*
  - Anteriormente mostramos que o aprofundamento progressivo é uma boa técnica para reduzir os custos de memória.
  - Podemos tentar o mesmo truque novamente, transformando a pesquisa A* em aprofundamento progressivo A* : IDA*

Métodos de procura

- IDA*
  - **Princípio:** funciona de modo semelhante ao algoritmo de pesquisa por aprofundamento progressivo, mas em vez de impor um limite na profundidade, impõe limites sucessivos (contornos) no custo das soluções que se podem calcular

Métodos de procura

- IDA*
  - A pesquisa em profundidade é modificada para usar a função *f* de limite de custo em vez de um limite de profundidade.
  - Assim cada iteração expande todos os nós dentro de um contorno definido para o corrente *f* - custo , procurando dentro do contorno, corrente o valor que limita o próximo contorno.
  - Uma vez que a pesquisa dentro de um certo contorno é completada, é iniciada uma nova iteração usando o novo valor da função custo para o próximo contorno.

Métodos de procura

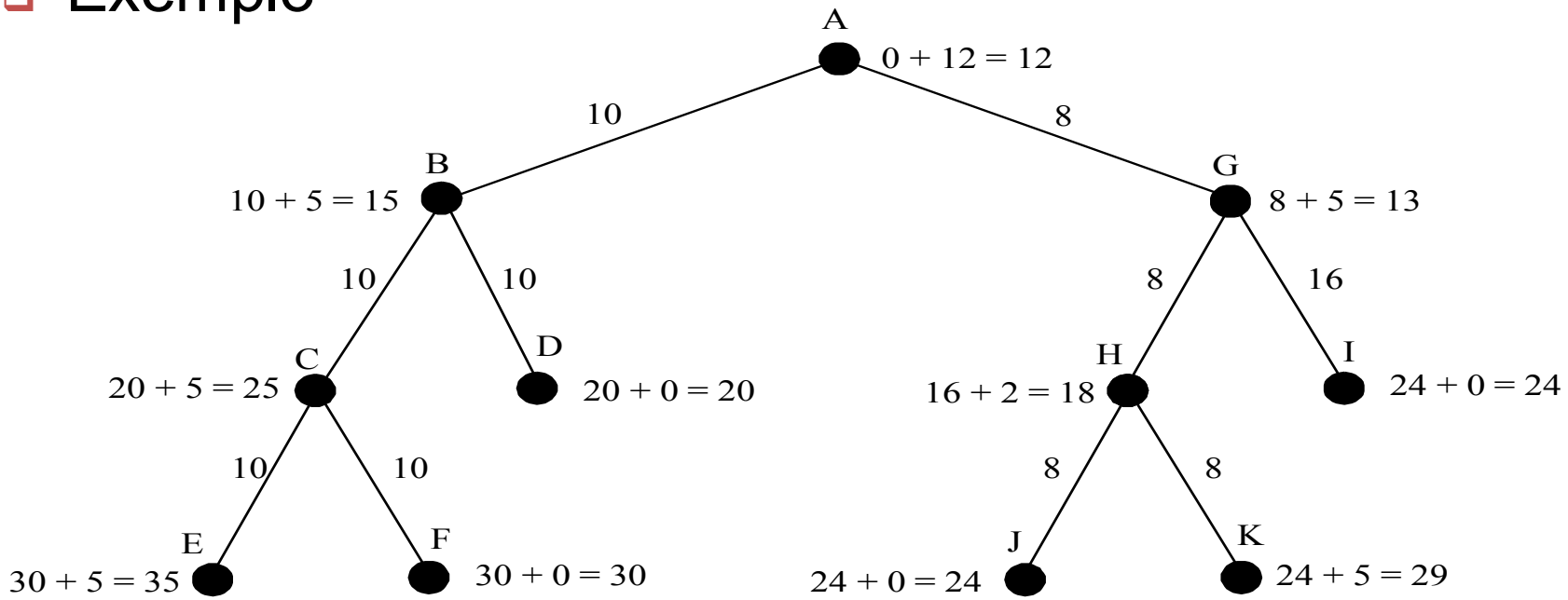
- **IDA***
  - **Óptimo :** Sim
  - **Completo :** Sim
  - Complexidade espacial: estimada em b*d.
  - A complexidade temporal depende fortemente do diferente numero de valores que a função heurística pode tomar, porque este número, vai limitar o número de iterações a efectuar.

Métodos de procura

- IDA*
  - Observações:
    - Soluções óptimas para muitos problemas, foram durante muito tempo encontradas pelo algoritmo IDA*
      - O qual foi durante muitos anos considerado como o único algoritmo heurístico óptimo, a nível de gestão de memória.
    - Problemas em domínios mais complexos.
      - Por exemplo, nos casos em que o valor da função heurística é diferente para cada estado(nó). Isto significa que cada contorno só inclui um estado mais que o contorno anterior. Se o A* expande N nós, IDA* tem que efectuar N iterações, e expandirá 1+2+3 + ... +N = $O(N^2)$ nós.
    - Assim se o N for demasiado grande para memória do computador, então será demasiado grande para causar uma grande espera de processamento, aumentando a complexidade temporal.

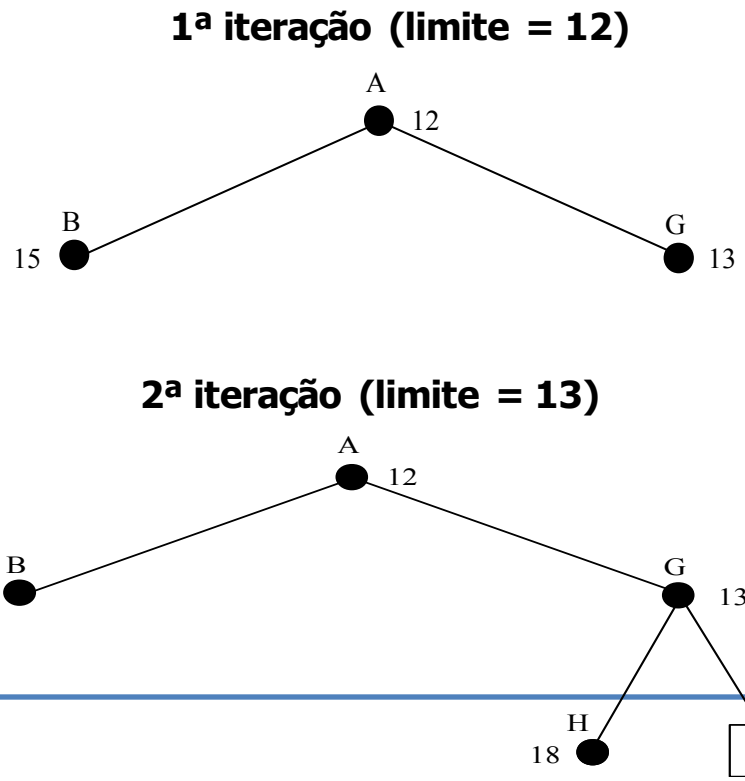
Métodos de procura

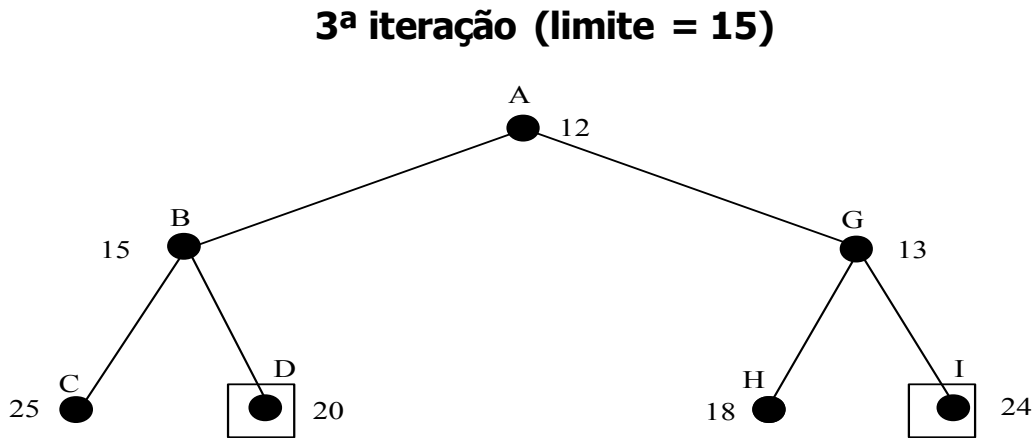
- **IDA***
  - Exemplo



Métodos de procura

- Exemplo (cont.)

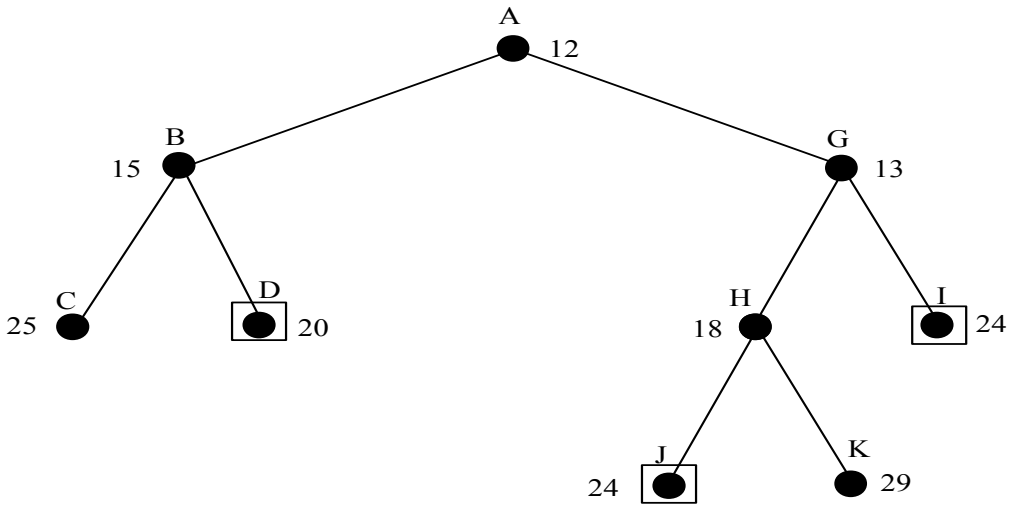




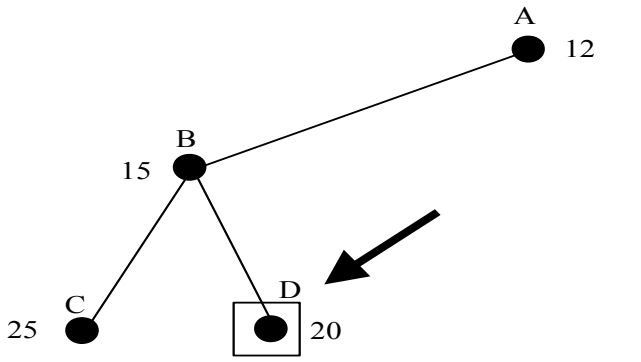
Métodos de procura

- Exemplo (cont.)

4ª iteração (limite = 18)



5ª iteração (limite = 20)



Métodos de procura

- **SMA***
  - O problema do IDA* em certos problemas de espaço podem resultar do uso de pouca memória. Entre iterações o IDA* retêm somente o valor limite da *f-cost* , função de custo. Como não pode recordar o caminho realizado até aí, ele é obrigado a repeti-lo.
  - Isto é duplamente verdade em espaços de estado que são grafos em vez de árvores. Aqui IDA* pode ser modificado para verificar o caminho corrente para estados repetidos, mas não consegue evitar estados repetidos pelos caminhos alternativos.

Métodos de procura

- **Princípio**: usa a memória disponível e quando necessita de expandir um sucessor, mas não tem memória suficiente, esquece os nós com *f* mais alto
- Faz uso de toda a memória disponível para efectuar a pesquisa.
- Usando mais memória, só pode melhorar a eficiência da pesquisa porque geralmente é melhor recordar um nó do que ter que reproduzi-lo quando necessário.

Métodos de procura

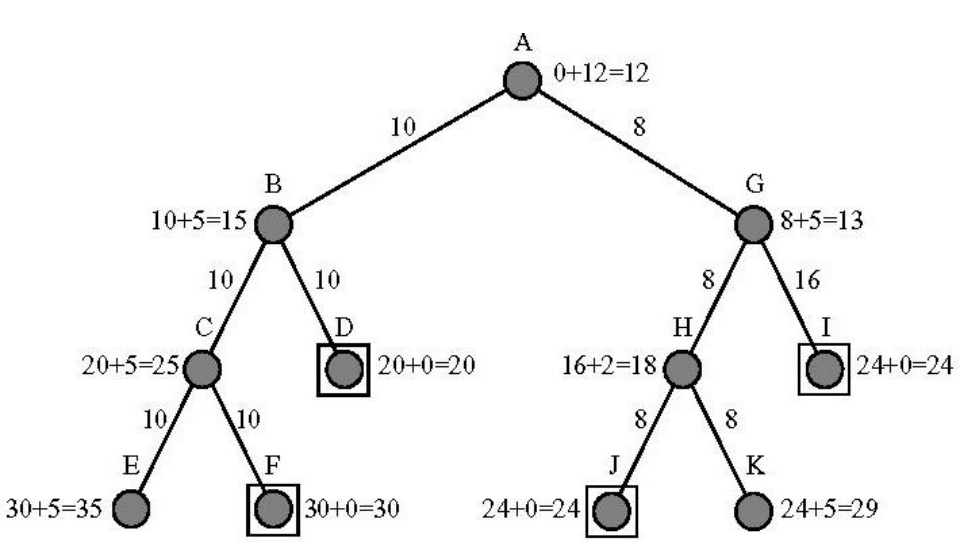
- SMA*
  - É completo, se a memória disponível for suficiente armazenar o caminho da solução encontrada.
  - É óptimo, se houver memória suficiente para armazenar o caminho da solução óptima encontrada. Caso contrário ele desenvolve a melhor solução que pode ser encontrada com a memória disponível.
  - Quando é disponibilizada memória suficiente para toda a árvore de pesquisa, a pesquisa é optimamente eficiente.

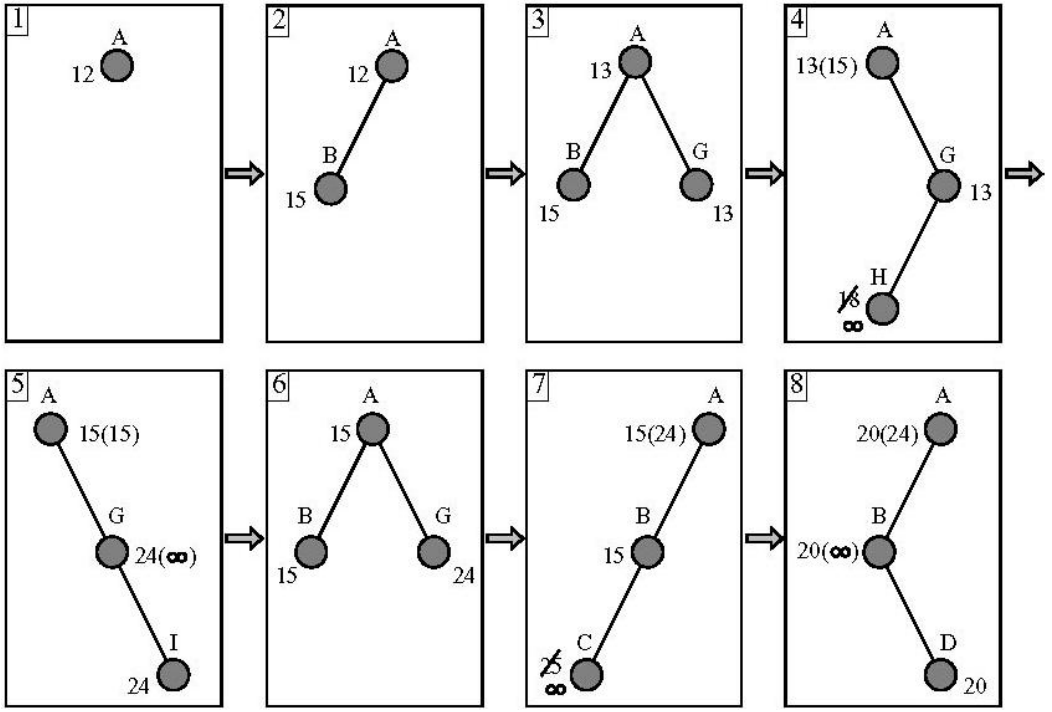
Métodos de procura

- SMA*
  - Quando o SMA* necessita de gerar um sucessor mas não tem memória suficiente, necessita de libertar espaço na pilha. Assim, retira um nó da pilha.
  - Os nós que são retirados são denominados nós esquecidos. Ele prefere retirar nós que não são prometedores, ou seja, com um alto valor de função custo (f-cost).
  - Ele retém a informação do nó antecessor que se refere à qualidade do melhor caminho na sub árvore esquecida.
  - Assim, ele regenera a sub árvore quando todos os outros caminhos mostrarem serem piores do que o caminho que ele esqueceu.

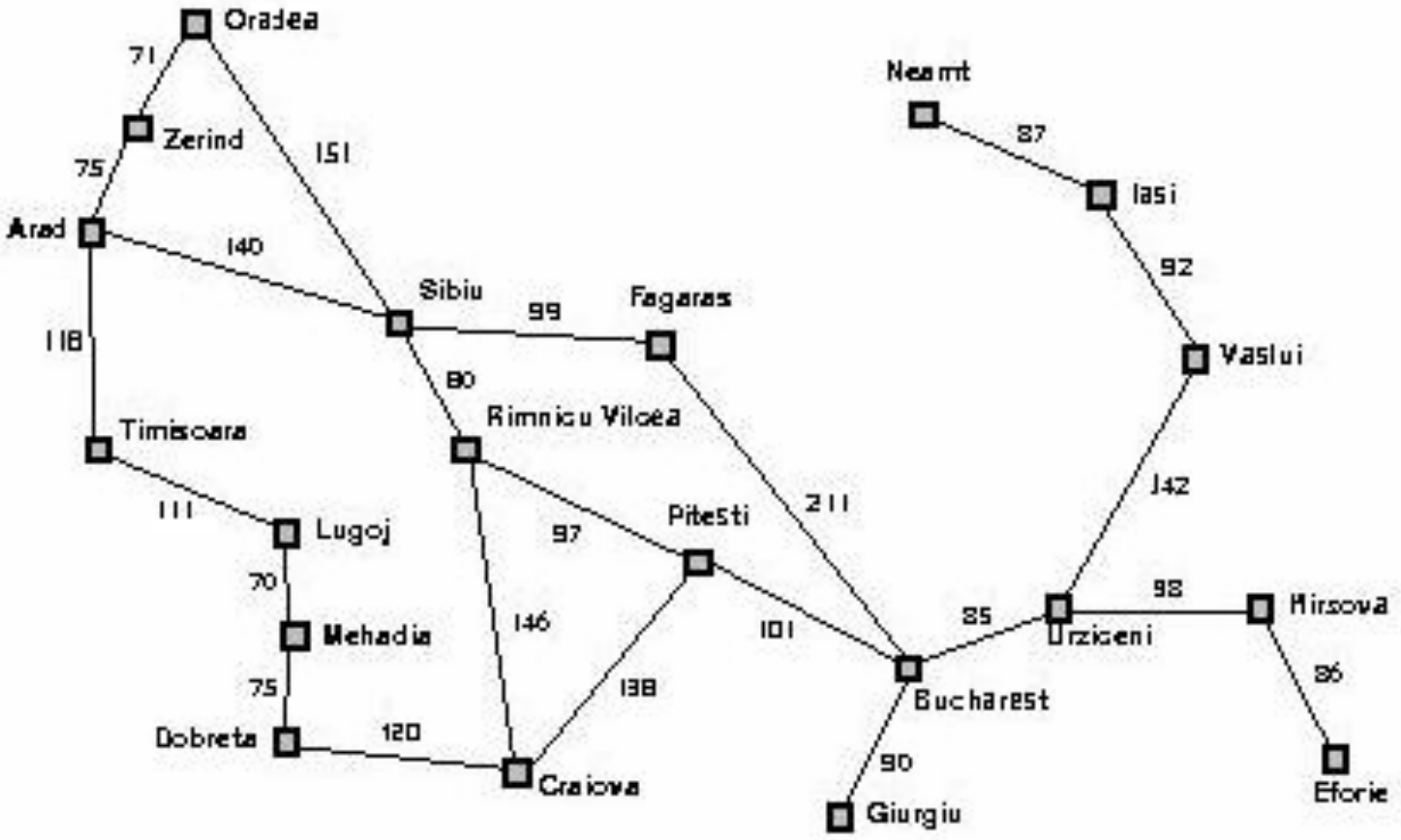
Métodos de procura

- SMA* - exemplo



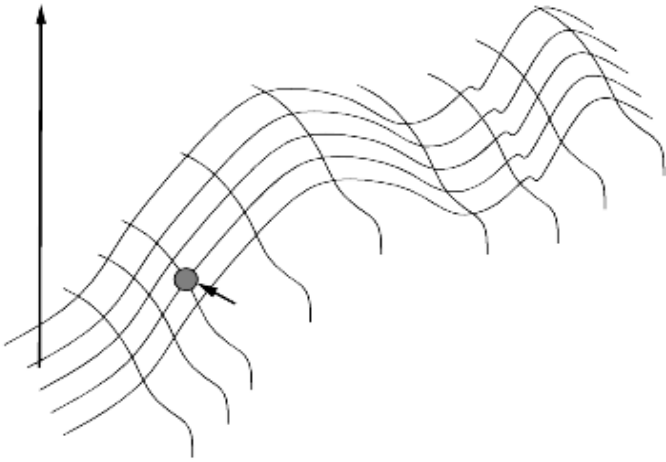


Métodos de procura



Distancia para Bucareste	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Métodos de procura



- Melhoria Progressiva
  - **Trepa-Colinas**
  - **Esfriamento Simulado**
- **Analogia**: as várias configurações sao representadas por pontos numa superfície que determina um relevo

Métodos de procura

- Melhoria Progressiva
  - A altitude de cada ponto na superfície corresponde ao valor da função de avaliação da configuração
  - A partir do ponto inicial (configuração inicial) pretende-se atingir o pico mais alto da superfície (configuração óptima)

Métodos de procura

- Trepa colinas (hill climbing)
  - **Princípio**: escolhe, de entre os sucessores de um nó, o que tiver valor mais alto e considera apenas esse para continuar a pesquisa
  - **Fragilidades**: pode ficar preso em
    - máximos locais
    - planaltos
    - linhas de encontro de encostas (arestas)
  - Nestas situações o procedimento usual é reiniciar o algoritmo aleatoriamente noutro ponto do espaço

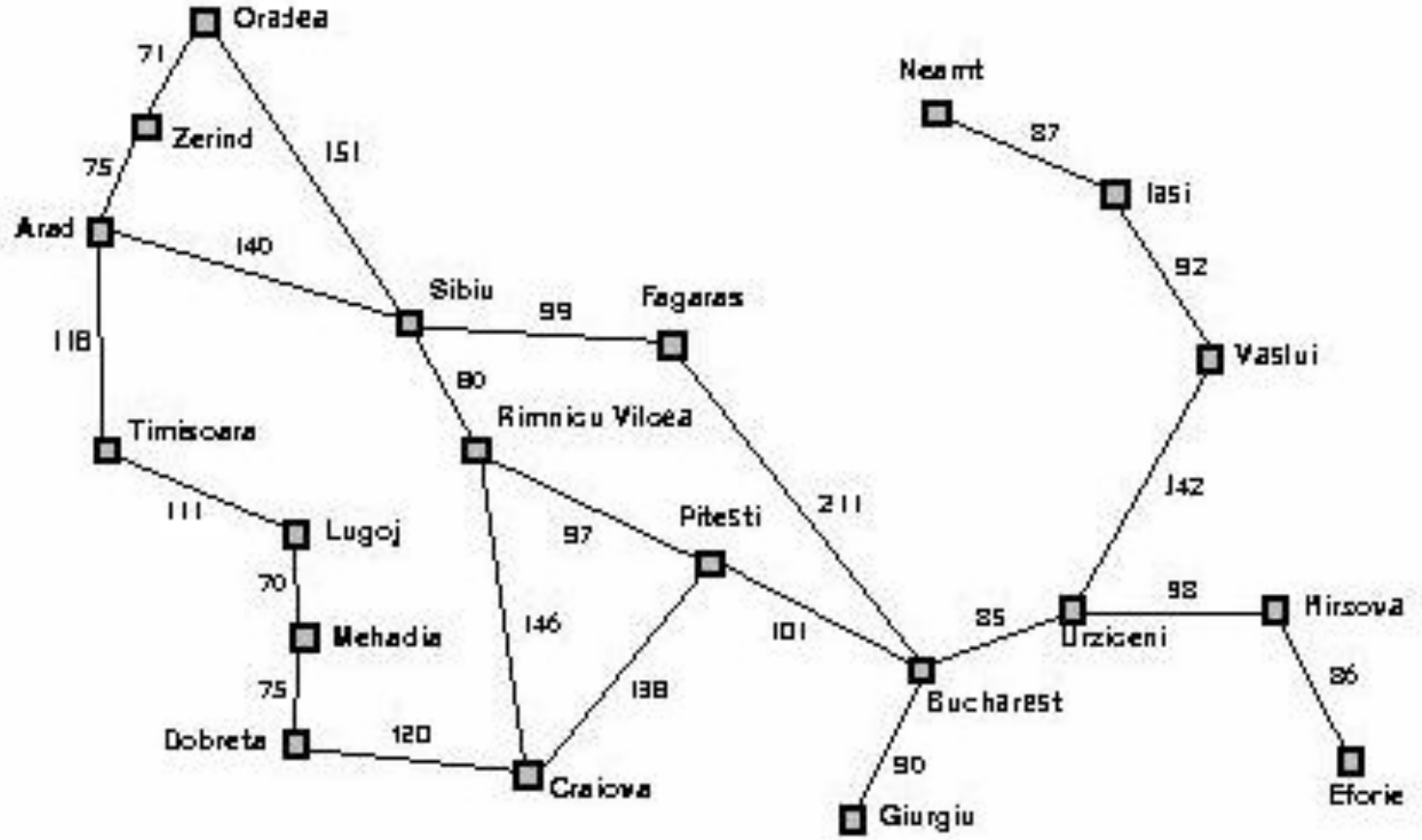
Métodos de procura

- Esfriamento simulado
  - **Princípio**: permite que, de vez em quando, se possa escolher um nó com valor *f* menor que o valor de *f* do nó actual
  - **Objectivo**: escapar a máximos locais

Métodos de procura

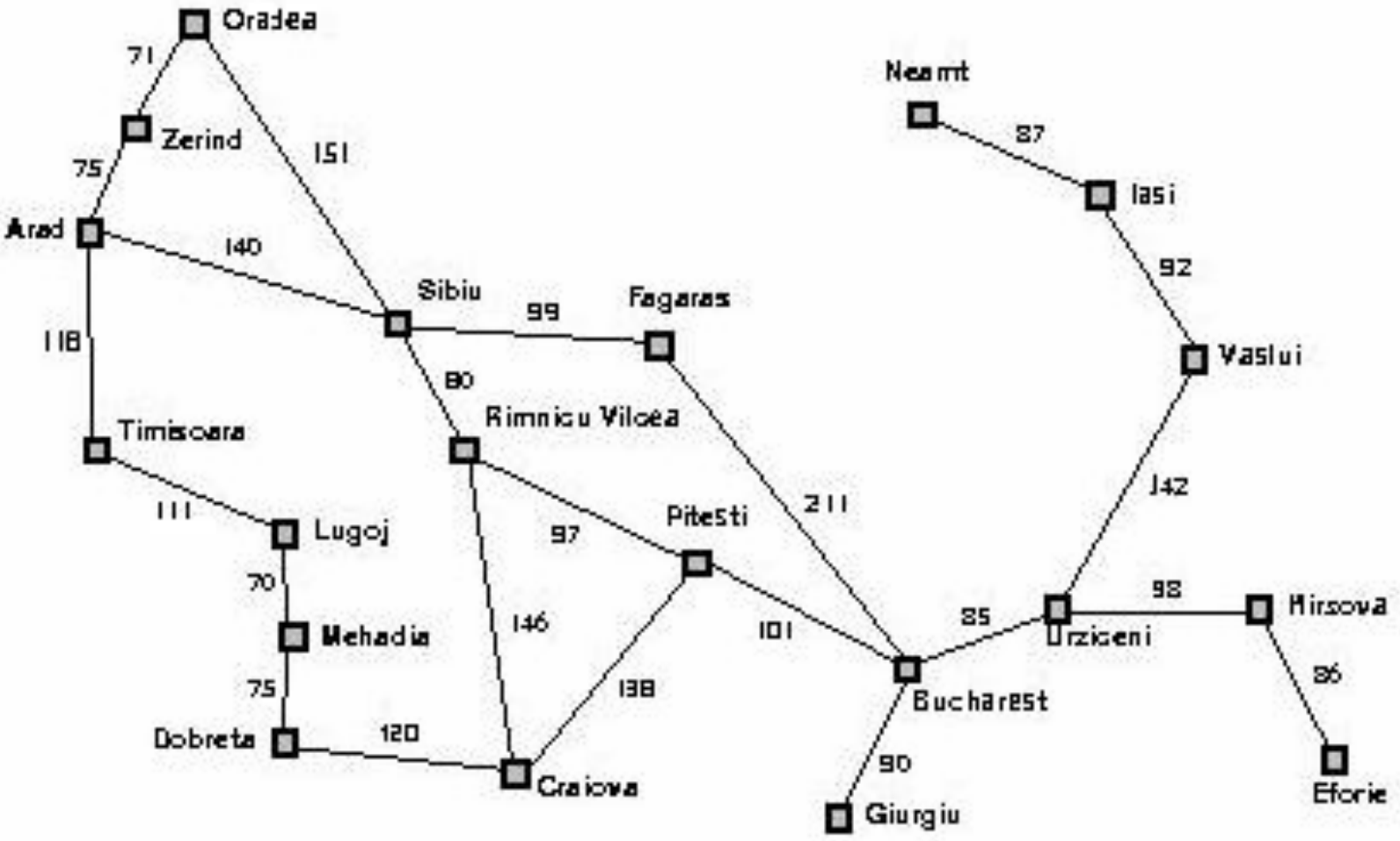
- Esfriamento simulado
  - Igual ao trepa colinas mas, em vez de escolher sempre o melhor nó, escolhe aleatoriamente o sucessor *s*
  - *s* é sempre escolhido se for melhor que o nó actual
  - Se *s* for pior que o nó actual, é escolhido com uma determinada probabilidade que decresce exponencialmente com o valor de *f* de *s*
  - A probabilidade de *s* ser escolhido depende também do parâmetro *T*. Quanto maior for o valor de *T*, maior a probabilidade de *s* ser escolhido. O valor de *T* vai diminuindo ao longo do processo de pesquisa

Métodos de procura



Métodos de procura

Métodos de procura



Distancia para Bucareste	
Arad	366
Bucharest	0
Craiova	160
Dobreta	242
Eforie	161
Fagaras	178
Giurgiu	77
Hirsova	151
Iasi	226
Lugoj	244
Mehadia	241
Neamt	234
Oradea	380
Pitesti	98
Rimnicu Vilcea	193
Sibiu	253
Timisoara	329
Urziceni	80
Vaslui	199
Zerind	374

Métodos de procura

Aveiro	Porto (68)	Viseu (95)	Coimbra(58)	Leiria (115)
Braga	Viana C.(48)	V. Real (106)	Porto (53)	
Bragança	V. Real (137)	Guarda (202)		
Beja	Évora (78)	Faro (152)	Setúbal (142)	
C. Branco	Coimbra (159)	Guarda (106)	Portalegre (80)	Évora(203)
Coimbra	Viseu (96)	Leiria (67)		
Évora	Lisboa (150)	Santarém (117)	Portalegre (131)	Setúbal (103)
Faro	Setúbal (249)	Lisboa (299)		
Guarda	V. Real (157)	Viseu (85)		
Leiria	Lisboa (129)	Santarém (70)		
Lisboa	Santarém (78)	Setúbal(50)		
Porto	V. Castelo (71)	V. Real (116)	Viseu (133)	
V. Real	Viseu (110)			

	Faro
Aveiro	363
Braga	454
Bragança	487
Beja	99
C. Branco	280
Coimbra	319
Évora	157
Faro	0
Guarda	352
Leiria	278
Lisboa	195
Portalegre	228
Porto	418
Santarém	228
Setúbal	168
Viana	473
V. Real	429
Viseu	363

Métodos de procura

- Exercício