Resolução de Problemas por Meio de Busca

Décio G. de Aguiar Neto

[1]Campus Quixadá - Universidade Federal do Ceará, Quixadá – CE, Brasil

decioaguiarneto@gmail.com

1. Introdução

Muitos problemas do mundo real muitas vezes podem não ser resolvidos facilmente sem a ajuda de meios computacionais, visto que muitos desses problemas apresentam infinitas possibilidades de soluções. No caso onde obter uma solução válida já é o suficiente algoritmos simples podem resolver esses tipos de problemas. Mas o que utilizar quando além de obter uma solução, essa solução deve ser ótima, visando maximizar ou minimizar uma função objetivo ? Nesse contexto agentes de resolução de problemas foram pensados como uma forma de obter soluções ótimas em problemas de otimização.

Quando pensamos na resolução de problemas por meios computacionais e usando métodos de inteligência artificial existem alguns cuidados que devemos tomar para uma melhor performance do agente a ser implementado. Dentre estes cuidados temos a formulação do problema que será abordado, quais algoritmos utilizar, no caso deste trabalho quais algoritmos de busca utilizar, e como fazer bom uso das informações fornecidas pelo problema, afim de que o agente a ser implementado possa se utilizar ao máximo desse conjunto de informações, assim alcançando o objetivo do problema.

Neste trabalho iremos abordar o problema de obter o menor custo de caminho entre duas cidades, retornando o caminho de menor custo, obtido por cada agente. Para isso um agente de busca será implementado e irá utilizar cada uma das seguintes buscas na resolução do problema, sendo elas, busca em largura(BFS), busca em profundidade(DFS) e a busca com custo uniforme(BCU). O código necessário para a reprodução dos testes neste experimento se encontram no seguinte repositório [1]

Este trabalho está organizado da seguinte forma na seção 2 é apresentado a definição do problema assim como a formulação formal do problema a ser explorado. Na seção 3 são apresentados os detalhes de implementação do agente, ferramentas e estruturas utilizadas. Na seção 4 são apresentados e discutidos os resultados obtidos por cada um dos métodos de busca utilizados pelo agente. Na seção 5 é apresentado a conclusão deste trabalho e uma pequena discussão sobre os resultados obtidos.

2. Formulação do Problema

O problema a ser explorado neste trabalho é o de obter o menor custo de caminho entre duas cidades, tendo como a cidade inicial *Arad* e a cidade objetivo *Bucharest*, e tendo como informação o mapa rodoviário simplificado da Romênia como mostrado na figura 1.

Para a formulação formal desse problema, devemos definir algumas informações sobre ele, que são o estado inicial, as ações que o agente pode executar, o modelo de transição de estados, o teste de objetivo e o custo de caminho.

[1]https://github.com/DecioAguiar/InteligenciaArtificial/blob/master/Search/

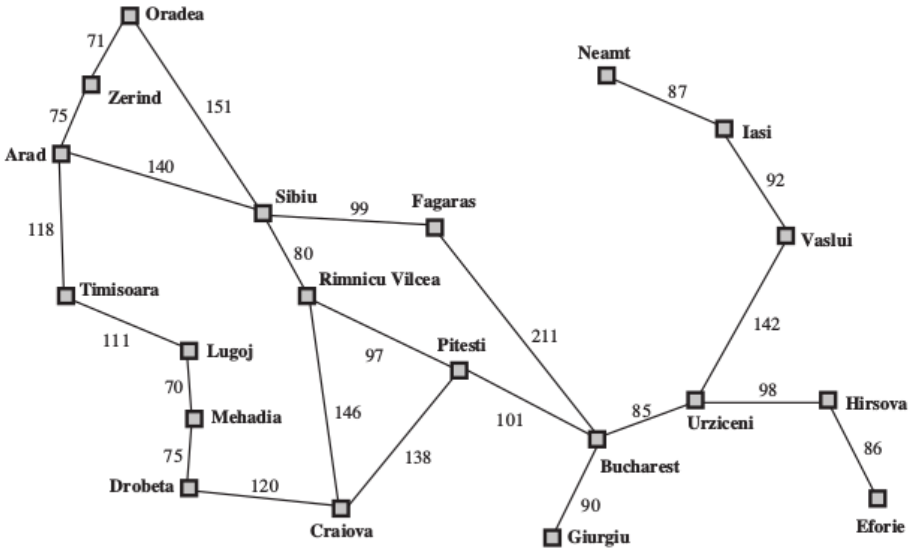

Figura 1. Mapa rodoviário simplificado de parte da Romênia [Norvig and Russell 2014]

Para o nosso problema, sabendo que o agente inicia sua busca em *Arad* e tem como objetivo chegar em *Bucharest*, sua formulação fica da seguinte forma:

1. **Estado inicial:** Sabendo que agente se encontra na cidade de *Arad* o estado inicial pode ser descrito como Em(*Arad*).
2. **Ações:** Sabendo o estado em que o agente se encontra, chamaremos de s, O método Ações(s) retorna o conjunto de ações possíveis a partir do estado s. Por exemplo, se o estado é Em(*Oradea*), as ações a serem retornadas pelo método Ações(*Oradea*) seria o conjunto irPara(Zerind), irPara(*Sibiu*).
3. **Modelo de transição de estados:** Dado o estado que o agente se encontra e uma ação, o estado seguinte que o agente irá, será o estado que corresponde a cidade para qual o agente se move. Exemplo, dado o estado Em(*Zerind*) e a ação irPara(*Oradea*) o resultado dessa transição é o novo estado do agente que será Em(*Oradea*).
4. **Custo de caminho:** O custo de caminho é referente ao custo de ir do estado atual para o novo estado. Seja s o estado atual a a ação, a função custo seria determinada por custo(s,a) onde o custo seria os KM percorridos do estado inicial até o estado atual. Por exemplo custo(*Arad*, Sibiu) = 140.
5. **Teste de objetivo:** Verificar se a cidade que o agente se encontra é Bucharest. Por exemplo Objetivo(*Bucharest*) retorna que o objetivo foi alcançado.

3. Implementação dos Algoritmos

Para a implementação do agente de busca foram utilizados a ferramenta Jupyter Notebook e a linguagem *Python* na sua versão 3.6.1. Conforme a descrição do trabalho proposto foram implementadas as classes base, *State*, *Node* e *Action*.

Afim de ajudar facilitar a manipulação na implementação foi utilizado um método *Series* da biblioteca Pandas onde cada cidade foi enumerada como um valor inteiro. As

ações foram modeladas como uma lista de adjacência, onde para cada cidade a sua lista de adjacência contém objetos do tipo *Action* que contém a cidade adjacente e o peso de ir até ela.

Já os estados continham apenas a informação em qual cidade se encontrava o agente, o objeto *Node* guardava as informações de onde o agente estava, assim como a lista de possíveis ações que poderiam ser aplicáveis no estado atual do problema.

Para a implementação das buscas que foram utilizadas pelo agente, foram utilizados os pseudocódigos fornecidos em [Norvig and Russell 2014] para os BFS, DFS e BCU.

Os experimentos foram realizados em um computador com as seguintes configurações: Intel Core i5-4210U CPU @ 1.70GHz 4, 8gb de memória RAM, sistema operacional Ubuntu 16.04.3 LTS 64-bit.

4. Resultados

Afim de analisar o desempenho de cada um dos métodos de busca utilizados pelo agente, alguns testes foram realizados.

No teste utilizando como problema de entrada o estado inicial foi definido sendo a cidade de *Arad* e como estado objetivo a cidade de *Bucharest*, os resultados obtidos por cada busca utilizada podem ser vistos na tabela 1

Tabela 1. Custo total obtido por cada busca

Busca	BFS	DFS	BCU
Custo total	450	733	450

Onde o caminho gerado pela BFS foi: *Arad - Sibiu - Fagaras - Bucharest*. O caminho gerado pela DFS foi: *Arad - Timisoara - Lugoj - Mehadia - Drobeta - Craiova - Pitesti - Bucharest*. E o caminho gerado pela BCU foi: *Arad - Sibiu - Fagaras - Bucharest*.

Outro teste realizado foi o tempo gasto pelos algoritmos com várias entradas distintas do problema, o gráfico da 2 mostra os tempo em microsegundos de cada um dos algoritmos para diferentes entradas. O problem1 é a entrada do problema de ir de *Arad* a *Bucharest*, como pode ser visto o DFS tem o menor tempo, mas como visto na tabela 1 retorna o pior custo de caminho dentre os outros dois algoritmos. Já a BCU consegue retornar um bom custo de caminho e executa em menor tempo que as outras duas buscas.

Para as outras entradas do problema, a BCU não apresenta menor tempo de execução, comparado as outras buscas.

5. Conclusão

Como esperado das buscas utilizadas, a busca em largura e a busca de custo uniforme irão retornar um bom resultado de custo de caminho, em relação a busca em profundidade, que não é completa e nem ótima [Norvig and Russell 2014], mas que em certos casos pode acabar achando a solução ótima do problema. Em relação a complexidade de tempo, a busca de custo uniforme acabou apresentando piores resultados em relação a tempo na maioria dos testes realizados, sendo a busca em profundidade a mais rápida mas não garantindo o ótimo.

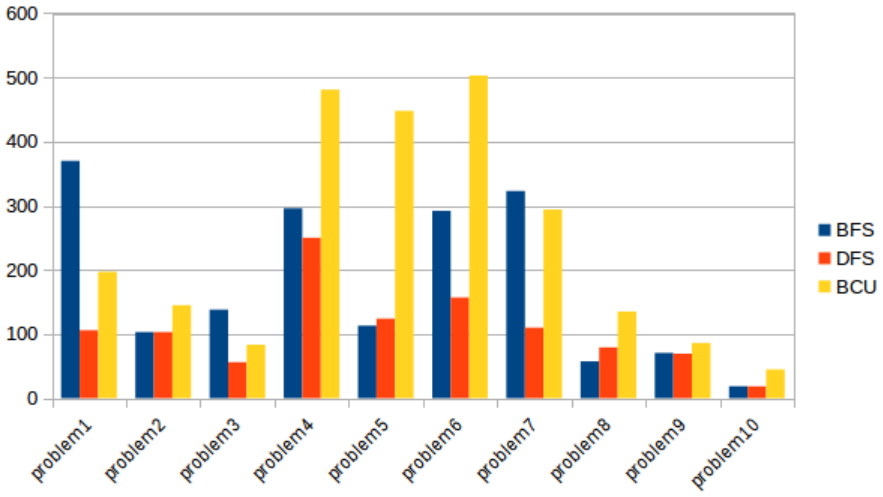

Figura 2. Resultados dos testes com diferentes problemas

Referências

Norvig, P. and Russell, S. (2014). Inteligência artificial, 3a edição.