Universidade de São Paulo

SEL0611- Fundamentos de Controle

Lista de Exercícios No.1

Pedro Morello Abbud

Número USP 8058718

Disciplina minsitrada por
Professor Doutor B.J.Mass

15 de Maio de 2017

1 Introducao

Este trabalho foi desenvolvido para a disciplina SMEXXX, Inteligencia Artificial, ministrada pela Professora Solange Sobrenome Sobrenome. O objetivo deste eh aprofundar e confirmar o conhecimento de nos , alunos, a cerca de buscas cegas e buscas informadas, assim como modelagem de problemas e solucao dos mesmos. Como o projeto tem vies tambem didatico, escolhemos utilizar Prolog como seu motor principal, pois este foi a linguagem de programacao de escolha ao ministrar-se as aulas da disciplinas. Dividimos a documentacao deste projeto em 3 partes:

Modelagem do Problema :

Consiste em definir o problema, suas dificuldades e qual abordagem foi utilizada para isso.

Solucao do Problema :

Como abordamos e resolvemos o problema.

Apresentacao do Problema :

As escolhas decididas por este grupo para apresentar de forma clara e interessante os resultados da Modelagem e Solucao do problema.

Foi escolhido para este projeto o problema do caixeiro-viajante, ou em ingles, TSP, Travelling Salesman Problem. Este é um problema clássico em computação: dado um mapa e um número de cidades, encontrar o caminho de menor distância para percorrer todas as cidades, passando uma única vez por cada uma delas. O problema do caixeiro-viajante é um problema de alta complexidade computacional e encontrar a solução ótima para um número elevado de cidades é muito custoso, crescendo de forma exponencial.

Definindo de uma forma formal, podemos dizer que o problema se resume em achar o menor caminho hamiltoniano em um grafo, o que caracteriza um problema NP-HARD.

Neste trabalho, é analisado o uso de dois métodos de busca não informada (busca em profundidade e em largura) e um método de busca informada (A*) para a solução do problema.

O problema abordado neste trabalho foi modificado em, relação ao problema original, para simplificar as implementações. As modificações feitas foram:

- É definida uma cidade inicial, informada no início do programa para fazer a busca;
- A busca é feita sem considerar o retorno à cidade original.

2 Modelagem

Figura 1: Name

A figura 1 mostra uma representação das cidades e suas interligações modeladas em um grafo. Cada nó do grafo representa uma cidade e as arestas são os caminhos, cada qual possui um custo associado.

Para uma abordagem mais realista, foi utilizada a API do Google Maps em um script Python para obter as distâncias reais entre cidades reais. A partir deste script, são exportadas as regras para o prolog, onde foi feita a implementação da busca. Desta forma, pode-se modelar o problema de forma dinamica.

Ao longo deste documento usaremos as seguintes definicoes:

Estados: Um estado é o caminho percorrido da cidade inicial até a cidade atual (com exceção do estado inicial, que não possui caminho mas apenas a cidade inicial).

Transição: Viagem de uma cidade para a outra. Estado final: Caminho percorrido depois de visitar todas as cidades uma única vez (e que apresenta o menor custo).

Custo: Distância entre as cidades.

3 Solucao

3.1 Busca Cega

Para a busca cega, foi testada tanto a busca em profundidade quanto a busca em largura para avaliar qual apresenta o melhor resultado.

3.2 Busca Informada

Para fazer a busca informada foi utilizada a estratégia A*. Essa estratégia foi utilizada por trabalhar com caminhos, e não apenas com um resultado final, o que é ideal para o problema. Também é interessante por não retornar um resultado aproximado, mas sim um resultado ótimo, desde que seja utilizada uma função de avaliação admissível. Um ponto negativo desta estratégia é a quantidade de memória utilizada já que todos os caminhos tentados são armazenados (embora apenas um caminho esteja sendo desenvolvido por

vez). Em nossos testes, não foi possível encontrar a solução para mais de 7 cidades. A heurística utilizada para guiar a busca A* foi a de vizinho mais próximo. Nesta, a função de avaliação é sempre zero (e portanto é admissível) e a função de custo é a distância entre as cidades. (Foi implementada também a heurística de inserção mais barata, fazendo a função de avaliação ser a distância em linha reta entre as cidades, no entanto essa apresentou uma performance pior, certamente devido à maneira como foi implementada.)