Relatório sobre o EP de IA - ACH2016 - 1º Sem 2019

Alunos:

Denise Keiko Ferreira Adati - 10430962

Fernando Karchiloff Gouveia de Amorim - 10387644

a. **Tabela com as estatísticas das buscas implementadas em termos de nós expandidos e tamanho de plano**

	Busca em Profundidade	**Busca em Largura**	**Busca de Custo Uniforme**	**Busca Heurística**
Tiny	15	15	15	14
Small	59	92	92	53
Medium	146	269	269	221
Big	390	620	620	549

b. **Discutir os méritos e desvantagens de cada método, no contexto dos dados obtidos**

Pelos dados obtidos em 4 tamanhos de planos diferentes, podemos observar que em quase todos os casos o algoritmo de busca heurística (A*) expande menos nós do que os outros 3 algoritmos de busca, tornando esse algoritmo mais rápido.

O algoritmo de busca em profundidade também expande menos nós em alguns dos planos testados. Porém, isso nem sempre é uma constante, pois o algoritmo de busca em profundidade retorna o primeiro caminho para o nó objetivo encontrado, ou seja, a quantidade de nós expandidos até a solução ser encontrada pode ser tanto mínima quanto máxima.

Como os custos do caminho de um nó a outro são idênticos em todos os nós, então podemos notar que os algoritmos de busca em largura e busca

de custo uniforme não apresentam diferenças com relação a quantidade de nós expandidos.

Vale lembrar que o melhor algoritmo para ser usado depende do tamanho do problema, caso ele cresça muito, a busca em profundidade pode acabar perdendo por ter de expandir nós demais para achar a solução.

c. QUESTÕES

1. Busca em Profundidade

No algoritmo de busca em profundidade, a ordem de exploração do espaço de estados segue a ordem de remoção da pilha dos elementos sucessores na pilha de caminho, eles são inseridos na ordem de exploração, porém são removidos para serem utilizados inversamente, do Oeste (West) para o Norte (North), lembrando que a ordem é: North, South, East, West. A ordem de exploração é inversa.

O Pacman só se move pelo caminho que encontrou até o objetivo através da execução do algoritmo de busca, o algoritmo vasculha pelo objetivo, caso encontre, retorna o caminho correto a ser feito para o Pacman seguir.

A busca em profundidade nem sempre encontra a solução ótima, ela retorna a primeira solução encontrada ao expandir os nós, sendo assim, pode ser que ela encontre uma solução que não é de custo mínimo, porém foi a primeira que encontrou. Dificilmente ela encontra a solução de custo mínimo.

2. Busca em Largura

A busca em largura sempre encontra a solução mais rasa (de menor custo) desde que o custo de todos os caminhos seja igual. A nossa implementação da busca em largura encontra a solução de menor custo, por ser um dos princípios do próprio conceito da busca em largura.

3. Busca de Custo Uniforme

No primeiro comando com o agente *StayEastSearchAgent*, Pacman tenta se manter do lado Leste do labirinto, pois ele é penalizado caso se mova para o Oeste, consequentemente ele acaba comendo algumas bolinhas durante seu percurso até encontrar o caminho.

No segundo comando com o agente *StayWestSearchAgent*, Pacman tenta se manter do lado Oeste do labirinto, pois ele é penalizado caso se mova para o Leste, ao encontrar o objetivo ele trava, pois não coletou as bolinhas que estavam do lado Leste do labirinto, ficando infinitamente preso no objetivo sem se mexer.

4. Busca Heurística

O algoritmo A* utiliza uma função heurística admissível. Para ela ser admissível a heurística não pode ultrapassar o custo real de um nó. Se a função heurística for consistente a busca A* se torna ótima desde que ela tenha detecção de estados repetidos.

Se h(n), função da heurística, for consistente, f(n) (custo de caminho + heurística) ao longo do caminho nunca será decrescente. Sempre que o algoritmo seleciona um nó para expansão o caminho ótimo para esse nó já foi encontrado, então é impossível ter outro caminho melhor do que o que já foi encontrado pois f(n) é crescente. Tendo isso em vista, o primeiro nó objetivo selecionado é a solução ótima.