Inteligência Artificial

Thiago Henrique Leite da Silva, RA: 139920

Aula 03 - Exercício Prático

1) Mostre um passo-a-passo da ideia que você explorou no exercício do URI. Você conseguiu aplicar busca (BFS, DFS), ou usou outra estratégia. Explane as ideias que você explorou para resolver o exercício.

Primeiramente, optei pela representação da malha por matriz, pois acredito que facilitou visualizar em qual posição estaria cada um de seus pontos.

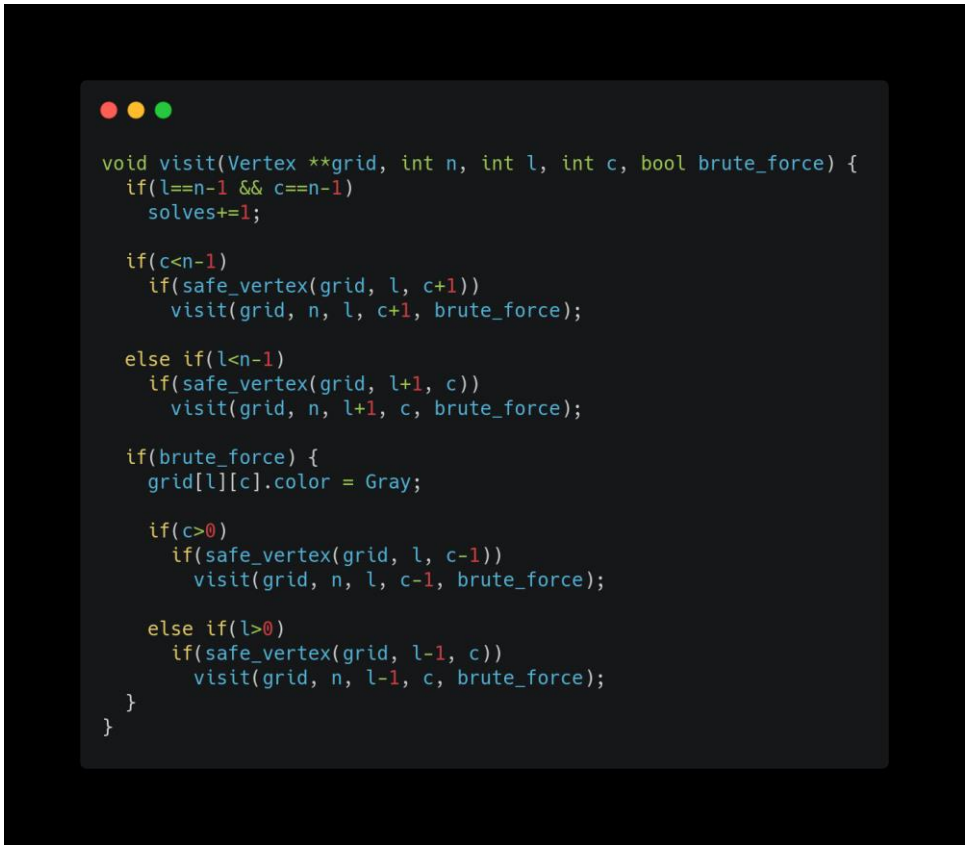
A partir daí, precisávamos dividir o problema em duas partes, uma em que realizamos uma busca a partir do ponto (0,0) só indo para direita e para baixo, sendo que precisamos encontrar todas as possibilidades de chegar no ponto (n-1, n-1) só indo nestas duas direções.

A segunda parte do problema é o que chamei de força bruta, quando precisamos se existe alguma solução de partir do ponto s (0,0) ao ponto t (n-1, n-1) podendo ir também para a esquerda e direita.

Juntei os dois subproblemas em um único método, que possui um parâmetro chamado "brute_force", caso ele seja false, realizamos, para cada um dos vértices da matriz, a busca para direita e para baixo, sem colorir nenhum dos vértices. Caso o parâmetro seja true, buscamos também para esquerda e para cima, e neste caso, como só precisamos de uma solução, vamos colorindo os vértices ao qual já passamos de cinza, pois desse jeito, no pior caso, vamos percorrer todos os vértices da matriz para garantir que não existe solução possível.

Lembrando também que temos as paredes que os robôs não podem atravessar as paredes, que são representadas por (#), então passamos percorrendo todos os vértices da matriz e antes de cada movimento (direita, esquerda, cima e baixo), verificamos, quando necessário, se a cor dele é branca (cor inicial), e se ele não é uma parede (#).

Outra tratativa que adotei, é verificar se não estamos nas bordas, pois se estamos em (0,0), por exemplo, não podemos ir nem pra cima, nem pra esquerda, para não sair da malha.



```
void visit(Vertex **grid, int n, int l, int c, bool brute_force) {
  if(l==n-1 && c==n-1)
    solves+=1;

  if(c<n-1)
    if(safe_vertex(grid, l, c+1))
      visit(grid, n, l, c+1, brute_force);

  else if(l<n-1)
    if(safe_vertex(grid, l+1, c))
      visit(grid, n, l+1, c, brute_force);

  if(brute_force) {
    grid[l][c].color = Gray;

    if(c>0)
      if(safe_vertex(grid, l, c-1))
        visit(grid, n, l, c-1, brute_force);

    else if(l>0)
      if(safe_vertex(grid, l-1, c))
        visit(grid, n, l-1, c, brute_force);
  }
}
```

Professora, uma coisa que me deixou bem intrigado é porque ao submeter meu código fora da disciplina, ele deu 75% de acerto, e submetendo na disciplina, ele não rodou, então submeti por último uma solução que acertei apenas 45%.

Consigo reconhecer que o problema do meu código está na hora de buscar todas as soluções possíveis indo pra direita e para baixo, ele leva muito tempo para fazer isso, mas passei horas e horas trabalhando em cima deste problema, mas não consegui pensar uma solução, gostaria muito de saber uma solução ótima pra este problema.

E confesso que para conseguir passar do Time Limit Exceeded, comentei algumas linhas, pois não estava rodando.

Em resumo, consigo fazer um código que funciona, mas que é lento, o que não serviria na prática.