Simulador de Estacionamento

Juliana Araújo

Universidade Federal de São João Del Rei, Outubro 2016

1 Introdução

Esse trabalho tem como objetivo a implementação de um simulador de estacionamento, a fim de levar um determinado carro à saída, movimentando os veículos necessários paralelamente ao eixo X ou ao eixo Y. Para isso, serão elaborados dois planejamentos de manobras, sendo o primeiro um algoritmo tentativa e erro e o segundo uma heurística.

2 Especificações do problema

O estacionamento é quadrado e tem dimensões 6 por 6. As dimensões são identificadas de X1 a X6 e de Y1 a Y6. O estacionamento comporta dois tipos de veículos: carros e caminhões. Os carros têm dimensão 2 por 1 e os caminhões têm dimensão 3 por 1. Os movimentos realizados pelos veículos são paralelos ao eixo X ou eixo Y, de forma que os veículos não fazem curva. O objetivo é levar um determinado carro Z à saída, que se encontra nas posições X5Y4 e X6Y4, fazendo as movimentações necessárias. A exemplificação do estacionamento se encontra no próximo tópico.

2.1 Entradas

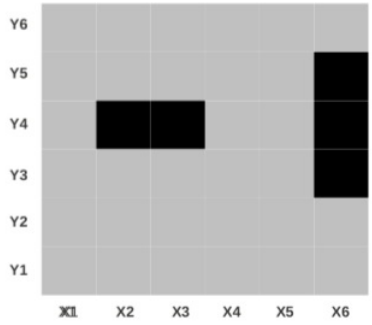
As entradas proveem de dois arquivos base. No primeiro, encontra-se os veículos contidos no estacionamento. Os veículos podem estar posicionados paralelo ao eixo X ou ao eixo Y. Cada linha informa sobre um veículo: identificador, tamanho, direção e posição. A posição do veículo é aquela que ele ocupa e que for mais próxima de X1Y1. Por exemplo, o arquivo a seguir descreve dois veículos:

Z 2 X X2Y4
A 3 Y X6Y3

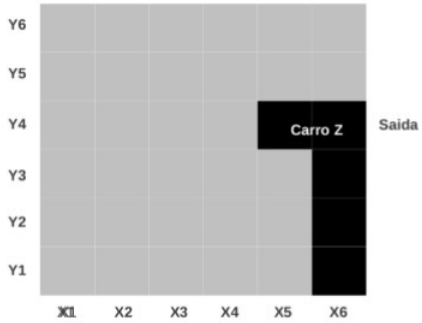
O segundo arquivo possui as sequências de manobras realizadas para movimentar Z em direção à saída. Cada movimento indica o veículo, a dimensão onde o movimento ocorre e a amplitude do movimento. Para o exemplo anterior, as manobras de saída seriam:

A Y -2
Z Y 3

O estacionamento, portanto, é preenchido da seguinte maneira:



A configuração final deve conter o carro Z ocupando as posições X5Y4 e X6Y4:



3 Desenvolvimento

O programa foi dividido em duas partes. A primeira parte realiza a leitura das entradas, preenchendo o estacionamento e executando as manobras lidas no outro arquivo. A segunda parte está dividida entre criar duas estratégias: tentativa e erro, que visita sistematicamente todas as posições da matriz até que o carro Z esteja na saída, e uma heurística desenvolvida, que nem sempre precisa levar a uma solução.

3.1 Parte 1

Para implementação do algoritmo, foi utilizada uma matriz de estruturas. Tendo em vista as inúmeras variáveis relacionadas aos veículos, foi criada uma estrutura que contêm todos os dados necessários a serem guardados. Uma matriz de

dimensão 6x6 simula o estacionamento. Cada posição contêm uma estrutura, e caso não haja um veículo ocupando-a, todas as variáveis da estrutura referente a essa posição estão preenchidas com 0. Diversas funções foram criadas para preencher a matriz e realizar os movimentos, sendo explicadas a seguir:

Funções que preenchem a matriz: Primeiramente, verifica se todas as posições ocupadas pelo veículo, carro ou caminhão, estão vagas. O veículo será disposto na matriz de acordo com seu tamanho e eixo.

Funções responsáveis pela movimentação: Cada movimento é muito particular, portanto foi necessário a criação de diversas funções para atender cada caso. Eles dependem do eixo do veículo a ser movimentado, do eixo do movimento e seu sentido. Também deve ser verificado se os movimentos não ultrapassam a área delimitada ao estacionamento, e se todo o caminho até a conclusão do movimento está disponível. Para essa última verificação, é utilizado um laço de repetição, que percorre da posição do veículo até seu destino final.

As manobras lidas serão guardadas em um vetor do tipo da estrutura criada, para que quando todas forem lidas, possa-se voltar a primeira. Isso se tornou necessário para que o algoritmo tentativa e erro leia o arquivo repetidamente até que se encontre a solução do problema. Porém, quando a heurística apresentada não encontra a solução, o algoritmo parte 1 entrará em loop.

3.2 Parte 2

Nessa parte, há o desenvolvimento efetivo das manobras que serão impressas no arquivo e utilizadas na parte 1. Foi implementada dois tipos de estratégia: tentativa e erro e uma heurística.

3.2.1 Tentativa e Erro

A entrada é lida, e todos os nomes de veículos são guardados em um vetor. Com ajuda desse vetor, será escrito no arquivo de manobras o nome do veículo e cada movimento que será testado, em cada direção possível. Isso será feito para cada veículo. Intercalando cada movimento, está os movimentos necessários para levar o carro Z à saída, calculados anteriormente. O programa principal irá guardar todos esses movimentos em um vetor, e iŕa executá-los repetidamente até que o carro Z esteja na posição da saída.

3.2.2 Heurística

A heurística pode ser considerada um algoritmo guloso, pois busca solucionar colisões pela ordem que precedem. O programa irá tentar realizar a movimentação do carro Z, e caso não seja possível, irá retornar onde houve colisão. Feito isso, haverá a tentativa de mover o veículo que colidiu com Z, em todas as posições possíveis. Assim como na tentativa e erro, a posição de Z será guardada a fim de calcular sua distância à saída. Feito isso, enquanto o carro Z não estiver na saída, irá tentar movimentar Z à saída, e caso não seja possível, a colisão que ocorreu será retornada e então será feito todas as movimentações possíveis com

a mesma, intercalando-as com a tentativa de movimentar Z novamente à saída. Antes da execução de cada movimento, independentemente se ele é válido ou não, o mesmo será impresso no arquivo de manobras.

4 Análise de resultados

Os dados distribuídos nos gráficos abaixo foram obtidos da média de execução de três entradas diferentes aleatórias, contendo 2, 4, 6, 8 e 10 carros.

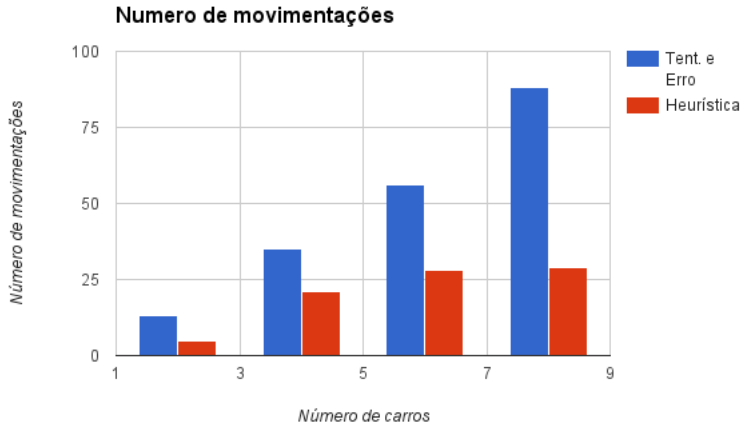


Figure 1: Distância Percorrida

O número de movimentações se altera mais devido a posição dos veículos do que o número presente na entrada. Ainda assim, a heurística apresentou melhores resultados, pois não procura qualquer veículo para fazer sua movimentação, e sim o que houve colisão com Z. Porém, não foi possível obter resultados com número superior a 10 carros, devido ao grande número de colisões. Apesar da estratégia de tentativa e erro apresentada fazer um maior número de movimentações, sua chance de garantir sucesso é maior, desde que haja uma combinação de movimentos possível.

Os tempos de usuário e sistema também foram calculados, porém não ouve alterações para que fosse feita uma análise. Caso o programa encontre um resultado, este é apresentado quase instantaneamente, seja pra qualquer um dos algoritmos. Se não, o algoritmo entra em loop.

5 Análise de complexidade

5.1 Tentativa e Erro

O algoritmo tentativa e erro apresentado irá imprimir no arquivo de manobras cada veículo e cada direção que ele pode movimentar em 1 unidade. O total de movimentos que um veículo pode fazer em uma direção será 1 - m, de modo que m é a dimensão da matriz. No caso da matriz de dimensão 6x6, será feito um movimento de 5 unidades, pois é o máximo que um veículo poderá movimentar nesse caso. Sendo dois loops aninhados, de 1 até o n, e outro até m - 1, a função de complexidade é dada por O(n * (m - 1)). Considerando m uma constante, que é caso do problema proposto, a ordem de complexidade é O(n), onde n é a quantidade de veículos da entrada.

5.2 Heurística

Apesar da heurística tentar primeiramente mover a colisão, todas os movimentos possíveis serão impressos, assim como na tentativa e erro. Porém, a posição de Z será calculada nesse loop, para que a colisão seja atualizada. O custo da heurística não será dependente do número de veículos e sim do tamanho da matriz. Os movimentos que podem ser realizados novamente serão calculado em loop de 1 a m - 1, onde m é a dimensão da matriz. Dentro deste loop, a função que encontra a posição de Z será chamada 9 vezes e sua função de complexidade é dependente do tamanho da matriz. Em um caso onde a matriz não é de dimensão constante, a função de complexidade do algoritmo seria O (m - 1 * 9(i * j)), onde i e j é a dimensão da matriz e m o maior desses dois números.

Por se tratar de um problema com dimensões constantes no estacionamento, a ordem de complexidade da heurística apresentada é O(1).

6 Conclusão

O algoritmo tentativa e erro possui sua estratégia menos elaborada, porém sua solução pode ser considerada razoavelmente boa, tendo em vista que o tempo de espera dos ocupantes dos veículos não será mínimo. Na heurística houve a tentativa de diminuir os movimentos desnecessários, e isso foi realizado de maneira considerável desde que ela consiga apresentar um resultado. Ainda assim, para grande número de a colisões não foi possível testar cada possibilidade de movimento, deixando o algoritmo ineficiente para grande número de veículos na entrada.