



Universidade Federal Rural de Pernambuco
Departamento de Estatística e Informática
Bacharelado em Sistemas de Informação

FlyFood

Ayan Faustt

Recife

Maio de 2022

1. Introdução

Em um mundo totalmente baseado na internet, a demanda por serviços de delivery e entregas tem crescido exponencialmente e, conseqüentemente, a busca pela otimização desses processos cresceram em mesmo nível, visto que, quanto maior a eficiência, maior se é a lucratividades das empresas prestadoras desses serviços.

Os principais problemas relacionados a este setor são as questões de otimização de tempo e custo de operação, visto que o tempo hábil e um baixo de custos de operação, são características vistas como vantagem competitiva em um mercado altamente aquecido.

Com a grande interconexão e dependência entre as economias globais e os setores econômicos, a necessidade de ter um processo de transporte mais eficiente torna-se uma necessidade vital para o contínuo crescimento destes.

Este trabalho, a fim de criar uma possível solução para os problemas supracitados, tem como objetivo otimizar os processos relacionados à entregas, sejam estas de bens ou de serviços.

Para atingir este objetivo, o algoritmo deverá ser apto a ler um mapa com os pontos de entrega e, além disso, realizar os cálculos entre estes, gerando um resultado contendo a melhor rota possível entre os pontos de entregas contidos no mapa.

Este trabalho, por meio de um algoritmo, tem como objetivos específicos:

- Analisar os pontos contidos em um mapa;
- Calcular as rotas com as combinações contidas no mapa;
- Demonstrar ao usuário final a melhor rota possível.

2. Referencial Teórico

Este trabalho tem como base, para o desenvolvimento do algoritmo os seguintes tópicos:

2.1 Complexidade de Algoritmos

O custo computacional é caracterizado pelo quadro de consumo de processamento e memória em função de uma entrada de dados de tamanho n , podendo ser de complexidade polinomial ou exponencial dependendo do tipo de crescimento.

2.2 Caixeiro Viajante

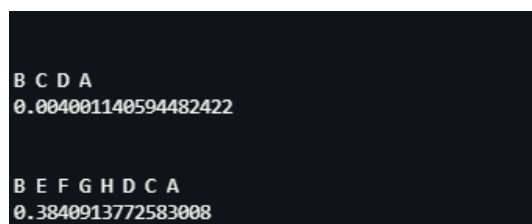
O problema do Caixeiro Viajante é um problema de otimização combinatória, é um problema muito importante, pois apresenta um vasto campo de aplicabilidade e um deles está no escopo deste projeto. O problema do Caixeiro viajante é fundamentado em um circuito hamiltoniano. O problema consiste no vendedor, partindo de uma origem,

visitar todas as cidades de forma de $c(P) > 0$ seja mínimo de modo que, no fim, tendo n cidades, há $(n-1)!$ possíveis caminhos a serem percorridos. A maior dificuldade a respeito desse problema é o crescimento exponencial do custo computacional à medida que um novo ponto é adicionado.

2.3 Algoritmo de Força Bruta

Um algoritmo de força bruta funciona através da comparação de todas as possibilidades e busca o resultado que melhor se adequa à resposta desejada. No caso do algoritmo deste trabalho, é calculado todas as permutações possíveis entre os pontos de entrega e, entre os elementos do resultado, é selecionado, entre permutações resultantes, aquela que além de ter o ponto de início fixo, o custo para percorrer todos esses pontos é o menor possível.

Apesar de ser um método de algoritmo que funciona, o cenário muda de acordo com o tamanho da entrada e da potência do hardware, podendo tornar sua execução consideravelmente mais lenta ou até impossível de se concluir.



```
B C D A
0.004001140594482422

B E F G H D C A
0.3840913772583008
```

Comparação de tempo de execução entre entradas com tamanhos diferentes

Fonte: Elaboração própria, 2022

O primeiro teste é executado sobre uma matriz 4×5 contendo 4 pontos de interesse, excluindo a origem. Enquanto no segundo teste, uma matriz 5×10 contendo 8 pontos de interesse. Observa-se um aumento para a realização dos cálculos de, aproximadamente, 9500 % no segundo teste em relação ao primeiro.

2.4 Algoritmo Genético

Os algoritmos genéticos são algoritmos com foco na otimização global. São implementados utilizando técnicas baseadas no mecanismo de seleção natural por meio da ‘reprodução’ dos indivíduos que são mais aptos como solução do problema.

São algoritmos com soluções de nível bom que possuem um custo computacional relativamente baixo. Os algoritmos genéticos trabalham com o esquema de população e indivíduo, este, geralmente composto por um array que é o seu gene e a população é composta por um conjunto de indivíduos. A cada iteração do algoritmo, utilizando o fitness como forma de mensurar a adaptação do indivíduo, e os operadores crossover e de mutação, é criada uma nova população e esta, preferencialmente, será mais apta para a resolução do problema que a anterior e, no final, é retornado o melhor

indivíduo encontrado. O operador de crossover é responsável pela criação de novos indivíduos por meio do cruzamento dos genes dos pais, e o de mutação, pela variabilidade, garantindo que o resultado obtido tenha uma maior otimização global.

3. Procedimento

3.1 Força Bruta

Procedimentos do algoritmo de força bruta:

1- Mapeamento: Após a leitura de toda a matriz, o algoritmo realiza o mapeamento de todos os pontos de interesse (destinos) da matriz. Os pontos e suas respectivas coordenadas são armazenados;

2- Roteamento: Com todos os pontos mapeados, ocorre o “roteamento” onde é feito a permutação entre os pontos armazenados no passo anterior. Após a permuta, os resultados são filtrados de maneira que os selecionados serão aqueles em que a cadeia de pontos inicia-se na origem;

3- Cálculo de custo: Com todos os trajetos possíveis computados, nesta etapa ocorrerá os cálculos de custo para cada trajeto. O custo total é contabilizado somando a distância de cada ponto em relação ao seu antecessor de forma que, ao chegar no último ponto, sua distância será comparada em relação à origem.

4- Seleção: Esta etapa ocorre concomitantemente com a etapa anterior. O primeiro trajeto a ser computado é armazenado em uma variável auxiliar de maneira que o custo dos percursos seguintes serão comparados com o custo do trajeto na variável auxiliar e, caso seja menor, o caminho com o menor custo passa a ser armazenado. Essa iteração ocorre até que o último trajeto seja verificado e após será exibido ao usuário.

3.2 Algoritmo Genético

O algoritmo inicia com uma população inicial de oito indivíduos, tendo o critério de parada atendido após o algoritmo rodar por cem vezes. E os demais parâmetros são:

3.2.1 Seleção

A seleção ocorrerá de forma que os indivíduos com maior fitness reproduzem-se entre si e o mesmo será feito com os de menor fitness.

3.2.2 Operador de cruzamento

O operador de crossover que será utilizado é o PMX, que consiste na seleção de pontos de corte nos pais e na inversão da parte seccionada.

No algoritmo deste projeto o primeiro filho manterá a primeira parte do primeiro pai preservada e receberá a segunda parte vinda do segundo pai. O contrário ocorre com o segundo filho. O ponto de secção será entre os elementos do meio e o último do gene dos pais.

3.2.3 Operador de Mutação

O operador de mutação que será utilizado é o Insertion Mutation que consiste na escolha aleatória de um elemento do gene, no caso deste trabalho, uma cidade da rota, e o insere novamente em uma posição diferente. No algoritmo desenvolvido neste trabalho a taxa de mutação será de 10%.

3.2.4 Função Fitness e Restrições

A função fitness responsável por determinar a aptidão do indivíduo, onde o indivíduo é identificado como percurso, desenvolvida neste trabalho é dada por:

$$fitness = \frac{1}{\sum_{i=0}^{len(percurso)-1} distância(percurso[i],percurso[i+1])}$$

E as restrições para a formação dos indivíduos são dadas por:

- Percurso[0] = Origem;
- Percurso[i] ≠ Percurso[i+1], Percurso[i+2]...Percurso[-1]

4. Resultados

Os resultados apresentados abaixo foram obtidos a partir da execução dos algoritmos em questão utilizando casos de testes. O tempo de execução é mensurado através da extensão coderunner que, ao fim da execução do código, retorna o seu tempo de execução. O tempo de execução considerado na análise é o valor médio obtido ao executar cada algoritmo, no mesmo teste, quatro vezes.

4.1 Tempo de execução

Em primeira análise, fica evidente que o método por força bruta é inviável para aplicações reais, principalmente em aplicações com uma maior entrada de dados e, após uma certa quantidade de pontos, o algoritmo implementado começou a falhar nos testes, pois o sistema estava encerrando sua execução.

Execução x Quantidade de pontos

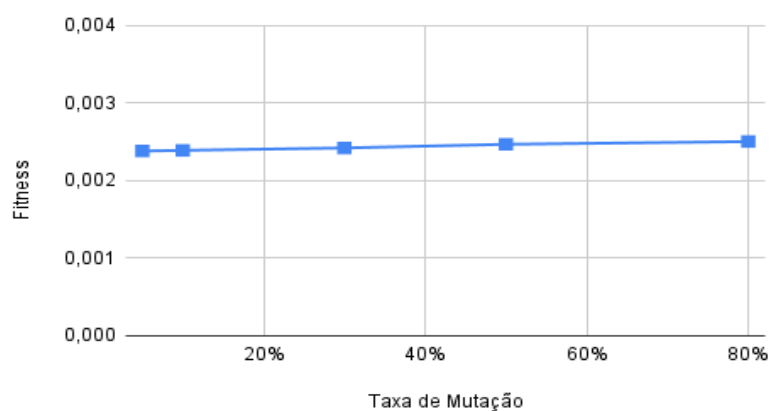


Enquanto isso, o algoritmo genético, junto do algoritmo com a heurística do vizinho mais próximo, tem um tempo de execução menor em todos os testes e quase constante em relação a entrada.

Pode-se haver dúvidas sobre a usabilidade do algoritmo genético, principalmente quando compara-se com a heurística do vizinho mais próximo. A heurística do vizinho mais próximo é do tipo determinística, ou seja, sua saída sempre será a mesma desde que a entrada seja a mesma, enquanto isso o algoritmo genético é probabilístico podendo apresentar saídas diferentes para a mesma entrada. Nesta característica encontra-se a vantagem dos algoritmos genéticos, de modo que é possível analisar as saídas a fim de escolher a melhor com um fitness mais interessante para a resolução do problema.

4.2 Algoritmo Genético

Fitness versus Taxa de Mutação



O gráfico acima demonstra o diferencial do algoritmo genético. Através da mutação, é garantido uma maior variabilidade da população, de forma que há uma

tendência de haver a maior chance de obter um bom fitness. E, quanto maior a taxa de mutação, observa-se um aumento médio no fitness. Os dados em questão foram obtidos através de um valor médio do fitness em cada faixa da taxa de mutação em um indivíduo com um gene de vinte e cinco cromossomos.

5. Conclusão

O objetivo deste trabalho é encontrar uma solução que satisfaça o problema do caixeiro viajante de modo menos custoso possível.

Os resultados obtidos cumprem totalmente com o objetivo proposto no início deste trabalho, ficando evidente nos dados analisados anteriormente.

Com isso, conclui-se que o algoritmo genético é uma solução viável para problema apresentado, sendo rápido, eficiente e tendo um custo computacional extremamente baixo. Em suma, os próximos passos serão de aperfeiçoamento do algoritmo através de atualizações à medida que os conhecimentos sobre algoritmos genéticos aumentam de escopo.

Referências Bibliográficas

- MundoGEO,(2020). Hardware Resolver Histórico Problema do Caixeiro Viajante. Disponível em: <https://mundogeo.com/2020/02/11/hardware-resolve-historico-problema-do-caixeiro-viajante/#:~:text=O%20%E2%80%9Cproblema%20do%20caixeiro%2Dviajante,diferentes%20cidades%20em%20uma%20lista>. Acesso em 27 mar. 2022
- Conte, Nelson. (2002). O Problema do Caixeiro Viajante. Teoria e Aplicações. Disponível em: <https://lume.ufrgs.br/bitstream/handle/10183/118198/000339835.pdf?sequence=1&isAllowed=y>. Acesso em 28 mar. 2022
- Barbosa, Felipe R. S. ; Souza, Flávio R. J. ; Vilela, Leonardo dos Reis. Trabalho de Análise de algoritmos. Disponível em: http://www.ic.uff.br/~jsilva/artigo_problema_da_mochila.pdf. Acesso em 28 mar. 2022
- Malaquias, Neli G. L. Uso dos Algoritmos Genéticos para a Otimização de Rotas de Distribuição. Disponível em: <https://repositorio.ufu.br/bitstream/123456789/14632/1/NGLMalaquiasDISPRT.pdf>. Acesso em 20 abril. 2022
- Pacheco, Marco A. C. Algoritmos Genéticos: Princípios e Aplicações. Disponível em: http://www.inf.ufsc.br/~mauro.roisenberg/ine5377/Cursos-ICA/CE-intro_apost.pdf . Acesso em 21 abril 2022.