Estudo Comparativo de Algoritmos para o Problema do Caixeiro Viajante - PCV Orientados a um Serviço de Entregas

Jean Brito Araújo[1], Francisco de Paula Santos de Araujo Junior[2]

[1]Ciência da Computação – Universidade Estadual do Piauí (UESPI)

Caixa Postal 64215-710 – Parnaíba – PI – Brasil

[2]Professor orientador – Universidade Estadual do Piauí (UESPI)

Caixa Postal 64215-710 – Parnaíba – PI – Brasil

otirbnaej@hotmail.com, pjhatata@hotmail.com

Abstract. *This article seeks to select and explore some algorithms aimed to solving the Travelling Salesman Problem - TSP. The objective of this work is to verify which algorithm from the chosen ones best fits and can be used in future in an application directed to delivery services to draw an optimized route, thus providing an economy for the establishments. For this, the algorithms were implemented in the C++ language and executed using some datasets representing the requests made by the clients. Finally, the data was systematized and analyzed.*

Resumo. *Este artigo busca selecionar e explorar alguns algoritmos voltados para resolver o Problema do Caixeiro Viajante – PCV. O trabalho objetiva-se em verificar qual algoritmo dentre os escolhidos melhor se encaixa e que possa vir a ser utilizado futuramente em uma aplicação direcionada aos serviços de entrega (delivery) para traçar uma rota otimizada, proporcionando assim uma economia para os estabelecimentos. Para isso, os algoritmos foram implementados na linguagem C++ e executados utilizando alguns conjuntos de dados representando os pedidos realizados pelos clientes. Por fim, os dados foram sistematizados e analisados.*

1. Introdução

O Problema do Caixeiro Viajante (PCV) ou *Traveling Salesman Problem* (TSP) consiste em traçar uma rota mínima onde um caixeiro deve visitar todos os pontos (cidades) passando apenas uma vez por cada uma delas, retornando para o local de partida ao fim do trajeto.

O PCV faz parte de um dos temas mais estudados no meio da análise combinatória e da pesquisa operacional e trata-se de um problema NP-Completo, onde não há um algoritmo capaz de resolvê-lo, em tempo polinomial. Dessa forma, são considerados problemas combinatórios complexos.

Por conta de sua complexidade, vários algoritmos foram criados para tentar alcançar uma melhor solução para o Problema do Caixeiro Viajante. Os diversos métodos

podem ser divididos em exatos, que são aqueles que possuem complexidade fatorial e heurísticos, que nem sempre retornam a melhor solução possível, mas que podem apresentar um resultado aproximado satisfatório. Quando existem diversos pontos a serem percorridos, o algoritmo pode apresentar um tempo de execução elevado, impossibilitando sua utilização por aplicações onde a instantaneidade é essencial. Nesses casos é recomendada a utilização de uma heurística para sua solução. Segundo Oliveira (2015), elas podem ainda ser divididas em heurísticas construtivas, heurísticas de melhoria iterativa e meta-heurísticas.

A aplicação do PCV está presente em diversas áreas do conhecimento. Muitas vezes em situações onde é necessário minimizar os custos envolvidos em determinados processos como operações envolvendo manufatura, roteamento de arquivos, roteamento de veículos e suas variações, problemas de sequenciamento, problemas de telecomunicação, entre outros.

O serviço de entrega, ou *delivery*, tem crescido constantemente. Muitas empresas se utilizam dessa atividade. Supermercados, lanchonetes, restaurantes e até lojas de cosméticos são alguns exemplos. Um ponto em destaque para todo estabelecimento que visa crescimento é ter um bom serviço de entregas. Por isso, faz-se necessário que os pedidos realizados sejam entregues em tempo hábil e de forma correta. A otimização do percurso desempenhado no momento das entregas proporciona economia de recursos para o estabelecimento.

Esse problema envolve desde pequenos empreendimentos de vários ramos até mesmo grandes empresas que fazem entregas em diversos estados. Visto que estes estabelecimentos precisam realizar muitas entregas em um curto espaço de tempo. Atualmente, na maioria desses empreendimentos, essa tarefa ocorre de forma desordenada. Não há um planejamento para otimizar o processo e o entregador é quem decide qual caminho percorrer.

A utilização de algoritmos pode contribuir com o serviço de entregas agilizando essa atividade. Para que isso ocorra, faz-se necessário um levantamento e análise dos algoritmos encarregados de resolver o PCV avaliando seus tempos de execução assim como o caminho obtido pelos mesmos. Um bom algoritmo para essa problemática deve apresentar tempo de execução e solução de boa qualidade.

Os algoritmos disponíveis para resolução do PCV variam de acordo com o seu desempenho e eficiência na tentativa de aproximação para uma melhor solução, o que está diretamente ligado ao tempo de execução do mesmo. Uma possibilidade para agilizar o processo de *delivery* é integrá-lo a um bom algoritmo que resolva esse problema, localizando uma rota que contemple os pontos de entrega de forma otimizada sem utilizar muitos recursos computacionais e que não leve muito tempo para ser executado.

No seguinte trabalho foram selecionados e implementados quatro algoritmos com base na literatura para realizar uma avaliação de sua eficiência para este problema. Os algoritmos foram Força Bruta, Vizinho Mais Próximo (Nearest Neighbor Search), Variable Neighbourhood Descent (VND) e Algoritmo Genético.

Três conjuntos de dados foram escolhidos para realizar os testes: o primeiro contendo quatro pontos; o segundo sete pontos; e o terceiro dez pontos. Para cada conjunto, os algoritmos foram executados cinco vezes a fim de avaliar a variação na qualidade da solução e tempo de execução.

Nas últimas seções são apresentados os resultados obtidos após a realização dos testes, assim como a apresentação de ideias para a realização de trabalhos futuros que possam somar a contribuição dada por este artigo. Expandindo, assim, o estudo na área do Problema do Caixeiro Viajante direcionada para os serviços de entrega.

2. Problema do Caixeiro Viajante (PCV)

Sua origem, assim como dito por Dantzig et al. (1954) não é completamente certa. Sua aplicação teria sido primeiramente estudada por conta de um jogo criado por Willian Rowan Hamilton chamado de Around The World e que consistia em encontrar em um dodecaedro o menor caminho que passasse por todos os vértices e que finalizasse na cidade de partida [Goldbarg e Luna 2000]. A solução para esse jogo passou a ser conhecida como ciclo hamiltoniano devido ao seu criador. Seu trabalho foi citado por diversos estudiosos, incluindo Hassler Whitney e Merril Flood em 1934.

O PCV consiste em generalizar o problema apresentado por Hamilton para a resolução em qualquer tipo de grafo. São definidos como um problema que demanda um tempo de execução de ordem fatorial para que seja resolvido de forma exata por meio de algoritmos determinísticos. A medida em que a quantidade de vértices a serem visitados cresce, o problema se torna inviável de ser resolvido devido a incapacidade de solução em tempo aceitável [Toscani e Veloso 2002].

Por essa característica, o PCV é considerado um problema NP-Completo [Garey e Johnson 1979]. Para os problemas onde existe uma grande quantidade de vértices a serem avaliados, utilizam-se heurísticas e meta-heurísticas, pois elas buscam, através do seu conjunto de regras e métodos, uma solução de forma rápida e que mais se aproxime do melhor resultado.

Classifica-se o Problema do Caixeiro Viajante em simétrico e assimétrico. No primeiro caso, leva em consideração que a distância entre dois vértices é a mesma tanto para ida quanto para a volta. Já no segundo caso, o custo entre eles difere. Os problemas simétricos têm complexidade de resolução maior que problemas assimétricos [Helsgaun 2000].

Existem inúmeras variações para o Problema do Caixeiro Viajante: PCV Múltiplo (*k-person traveling problem*), que consiste em determinar um conjunto de percursos considerando que existem n caixeiros viajantes para realizar o trajeto [Miller, Tucker e Zemlin 1960]; PCV com Janelas de Tempo (TSP with time Windows - TSPTW) onde cada cidade possui um período de tempo a ser visitada [Fox, Gavish e Graves 1970]; PCV com lucros (TSP with profits - TSPP) onde para cada cidade, o visitante recebe um prêmio a ser considerado [Feillet, Dejax e Gendreau 2005]; PCV multi-objetivo (*Multiobjective Traveling Salesman Problem* – MOTSP) que leva em consideração diversos objetivos simultaneamente, como a distância de viagem, tempo de percurso e gastos em geral, partindo do princípio que esses fatores estão intimamente ligados e que a alteração de um tem influência sobre o outro.

Ao longo dos anos, foram propostas diversas formulações matemáticas para o Problema do Caixeiro Viajante. Devido sua fácil compreensão, uma das mais utilizadas presentes na literatura é a Formulação de Dantzig-Fulkerson-Johnson (DFJ) [Goldbarg e Luna 2000]. Pode ser formulado matematicamente da seguinte forma:

$$\textbf{Minimizar z} = \sum_{j=1}^{n}\sum_{i=1}^{n} c_{ij}x_{ij}$$

Sujeito a:

$$\sum_{i=1}^{n} x_{ij} = 1 \qquad \forall j \in N$$

$$\sum_{j=1}^{n} x_{ij} = 1 \qquad \forall i \in N$$

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1 \qquad \forall S \subset N$$

$$x_{ij} \in \{0,1\} \qquad \forall i, j \in N$$

Figura 1. Formulação de Dantzig-Fulkerson-Johnson

O modelo é definido por um grafo G = (N, A), onde N representa o conjunto de nós e A o conjunto de arcos de G. A variável X_{ij} assume valor 1 se o arco (i, j) pertencer a A e for escolhido para fazer parte da solução. Caso contrário, X_{ij} recebe 0. C_{ij} representa o custo e X_{ij} a variável de decisão, enquanto um vetor que pode ser definido por $\tilde{x}$ possui a sequência da rota. Deve-se também eliminar as possíveis sub-rotas.

3. Os Serviços de Entrega (*Delivery*)

Não se sabe dizer quando e nem como exatamente surgiu a prática do serviço de entrega ou *delivery*. O que se sabe é que essa atividade tem crescido bastante desde a década de 80 e 90. No Brasil, a prática se tornou comum através de pizzarias que trabalhavam exclusivamente para entrega [Xavier 2015]. Com o avanço da tecnologia, os pedidos passaram a ser realizados por meio de aplicativos móveis, o que antes era feito apenas por telefone.

O serviço de entregas facilita a vida dos consumidores. Boa parte do faturamento dos estabelecimentos provém de pedidos realizados para *delivery* [Rodrigues 2016]. A otimização dessa atividade ao planejar o caminho a ser percorrido pelo entregador pode proporcionar diversos ganhos para os empreendimentos e para os clientes. Dentre esses benefícios pode-se citar a economia de recursos e agilidade, fazendo com que mais entregas possam ser realizadas em menos tempo.

4. Algoritmos para o Problema do Caixeiro Viajante

Os algoritmos utilizados para a resolução do Problema do Caixeiro Viajante podem ser divididos de acordo com a sua abordagem. Podem ser classificados em exatos e heurísticos ou aproximativos. Exatos são aqueles que apresentam a melhor solução para o problema, porém demandam tempo de execução elevado quando são aplicados a uma

quantidade de dados muito grande. A abordagem heurística tende a encontrar uma solução ideal, ou seja, aproximada da melhor solução possível utilizando um tempo bastante inferior em relação aos métodos exatos.

Para a realização do presente trabalho foram selecionados quatro paradigmas que trabalham na solução do PCV. Os algoritmos escolhidos foram: Força Bruta, Vizinho Mais Próximo, Variable Neighbourhood Descent (VND) e Algoritmo Genético. A escolha dos mesmos levou em consideração as três abordagens existentes, onde há pelo menos um algoritmo para cada. Determinístico: Força Bruta, Heurístico: Vizinho Mais Próximo, Metaheurístico: VND e Algoritmo Genético. Os mesmos também são bastante conhecidos na literatura e são muito utilizados em diversos problemas de combinação e roteirização incluindo o PCV.

4.1. Força Bruta

Ao se pensar em projeto de algoritmos, uma das primeiras técnicas a serem trabalhadas é o algoritmo de força bruta, também conhecido como busca exaustiva. É um algoritmo determinístico. Sua forma de execução consiste em avaliar todas as soluções para o problema envolvido a fim de buscar o melhor resultado. Apesar de apresentar a melhor opção como resposta final, sua utilização se restringe a amostras pequenas, visto que não se trata de um algoritmo eficiente, pois sua complexidade cresce exponencialmente.

Esta abordagem não adota nenhuma técnica especial para melhorar sua performance. Depende exclusivamente do poder computacional para testar as diversas possibilidades disponíveis até que uma solução seja encontrada. [Baidoo e Oppong 2016]. Em alguns casos, o algoritmo de Força Bruta é utilizado para verificar o resultado de algoritmos heurísticos, visto que a solução sempre é encontrada.

Em relação ao PCV, o algoritmo utiliza uma combinação para avaliar todos as possibilidades de caminho retornando ao final da execução o melhor resultado para o problema. Nesse caso existem $(n - 1)!$ hipóteses a serem testadas, onde n representa o número de arestas. O algoritmo possui complexidade fatorial $O(n!)$ [Freitas 2010].

4.2. Vizinho Mais Próximo

O Algoritmo do Vizinho Mais Próximo trata-se de uma heurística gulosa simples que consiste em deslocar-se do ponto de partida para o ponto mais próximo e retornando ao início quando todos os pontos tiverem sido contemplados [Goldbarg e Luna 2000]. Foi um dos primeiros algoritmos utilizados para resolver o problema do Caixeiro Viajante e ainda é bastante utilizado porque, apesar de não apresentar o melhor resultado, gera uma solução aceitável.

O primeiro passo a ser observado é o cálculo da distância entre todos os pontos, armazenando essa informação numa matriz. Iniciando do vértice de partida, deve-se observar qual ponto possui o menor caminho a ser percorrido para que então a rota comece a ser gerada. Os pontos visitados devem ser marcados. Esse procedimento é repetido até que não sobre nenhum ponto. O último passo é retornar para o local de início, encerrando assim o funcionamento do algoritmo. Um caminho otimizado é então gerado.

4.3. Variable Neighbourhood Descent

O Método de Descida em Vizinhança Variável – VND (Variable Neighbourhood Descent) consiste de uma busca local onde o seu funcionamento baseia-se em diversas

trocas sistemáticas no ambiente da vizinhança, armazenando uma solução sempre que houver otimização em relação à solução encontrada a princípio. Esse método foi proposto por Mladenovic & Hansen (1997).

Trata-se de um método de refinamento aplicado sob uma solução inicial (adotou-se uma solução inicial aleatória) e seu diferencial consiste em avaliar várias estruturas de vizinhança, ao invés de analisar uma única estrutura. Por esse motivo, difere-se do modelo de Busca Local tradicional [Vianna 2004].

Para o caso do Problema do Caixeiro Viajante, o algoritmo VND utilizado foi aplicado sob uma rota gerada aleatoriamente. Fundamenta-se na seleção de um dos pontos onde é realizada a troca entre os vértices vizinhos, um de cada vez, até que a condição de parada seja alcançada. Sempre que uma rota otimizada for encontrada, é realizada a substituição da solução atual. Em caso de não haver melhora em nenhuma das vizinhanças pesquisadas, o método é interrompido.

4.4. Algoritmo Genético

Os Algoritmos Genéticos baseiam-se na teoria da seleção natural proposta por Charles Darwin onde os indivíduos mais preparados sobrevivem. A primeira grande obra na área é atribuída a John H. Holland com sua publicação "Adaptação em Sistemas Naturais e Artificiais" em 1975. Porém, de acordo com Pila (2006), diversos outros autores já haviam introduzido a ideia de algoritmos genéticos em seus trabalhos anteriormente, como Bagley (1967), Rosenberg (1967), Cavicchio (1970), Hollstien (1971) e Bosworth (1972).

O Algoritmo Genético faz uma analogia de um caso do mundo real (evolução) para encontrar soluções aproximadas para problemas como o do Caixeiro Viajante. Guedes (2005) os define como sendo uma técnica exploratória que busca uma população de soluções viáveis (cromossomos), onde os operadores genéticos de cruzamento e mutação seriam responsáveis por gerar a evolução dos indivíduos. Em cada ciclo evolutivo do algoritmo, os indivíduos da população são submetidos a uma avaliação para definir um valor a cada um deles (fitness).

De forma geral, os Algoritmos Genéticos operam seguindo alguns passos. Na inicialização é gerada aleatoriamente uma população inicial com N cromossomos, determinando o fitness de cada um; uma nova população é então criada após a sequência de três etapas: seleção, crossover e mutação; é feita então uma avaliação da população gerada avaliando o fitness de cada cromossomo dessa população. Se a condição de parada for satisfeita, o algoritmo é encerrado retornando o melhor resultado obtido, caso contrário, novas populações são geradas até que essa condição seja alcançada.

Para o algoritmo utilizado, considerou-se o limite de 10 gerações como condição de parada. A população inicial é determinada através de um parâmetro chamado gerações, onde são feitas permutações sobre os dados de entrada. Consiste de uma forma de gerar aleatoriamente a população inicial. O valor utilizado para essa medida foi de 1000 permutações. Outro parâmetro relevante é a taxa de mutação que indica a possibilidade dos filhos sofrerem mutação após o processo de crossover. A taxa utilizada foi de 5%.

Para cada geração são selecionadas uma quantidade de rotas onde serão escolhidas as duas melhores (rotas pai) para a realização do crossover onde são combinadas para a geração de rotas filhas. Nesse momento é observado a taxa de mutação. De acordo com a

probabilidade, as rotas filhas sofrem uma mutação, evitando que todas as rotas da população sejam iguais. As rotas geradas substituem as duas rotas maiores da população. Esse processo é repetido até alcançar o critério de parada.

5. Materiais e Métodos

Os quatro algoritmos selecionados (Força Bruta, Vizinho Mais Próximo, Variable Neighbourhood Descent e Algoritmo Genético) foram implementados na linguagem de programação de alto nível C++ e utilizou-se o ambiente de desenvolvimento Codeblocks para a execução dos mesmos. Devido à dificuldade em se encontrar conjuntos de dados já prontos com uma quantidade de pontos que refletisse o problema com fidelidade, foram criados três conjuntos para realização dos testes.

A quantidade de pontos selecionados baseou-se na inferência do número de entregas realizadas por vez por um entregador de um estabelecimento comercial (lanchonetes, restaurantes, entre outros). Em todos os conjuntos pode-se considerar o primeiro ponto como sendo o estabelecimento e os demais pontos os locais de entrega. Os conjuntos de dados foram classificados como pequeno, médio e grande em relação a sua quantidade de pontos, respectivamente quatro (PONTOS4), sete (PONTOS7) e dez (PONTOS10). Com relação aos dados utilizados, as distâncias entre os pontos são equivalentes às distâncias encontradas em uma cidade como Parnaíba – Piauí, local onde a pesquisa foi realizada.

Os gráficos relativos aos conjuntos de dados estão presentes nas Figuras 2.1, 2.2, 2.3. Os pontos da amostra com baixa quantidade de pontos (PONTOS4) foram dispostos da seguinte maneira:

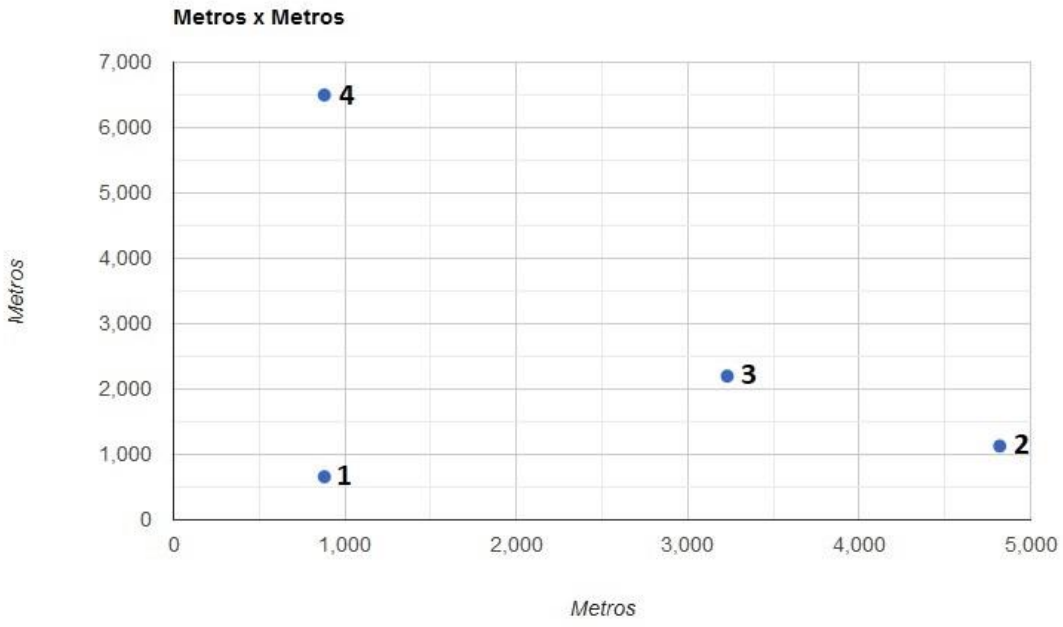

Figura 2.1. PONTOS4

A amostra com média quantidade de pontos (PONTOS7) está representada na figura 2.2 a seguir:

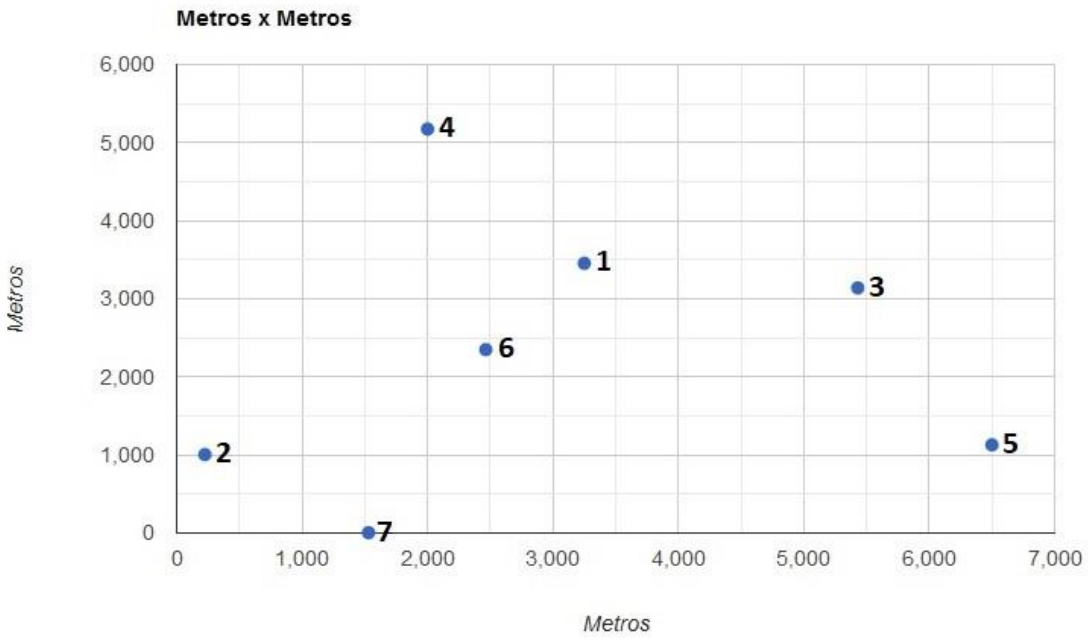

Figura 2.2. PONTOS7

O terceiro conjunto de dados contém 10 pontos (figura 2.3). Essa amostra foi rotulada como grande quantidade de dados em relação às demais.

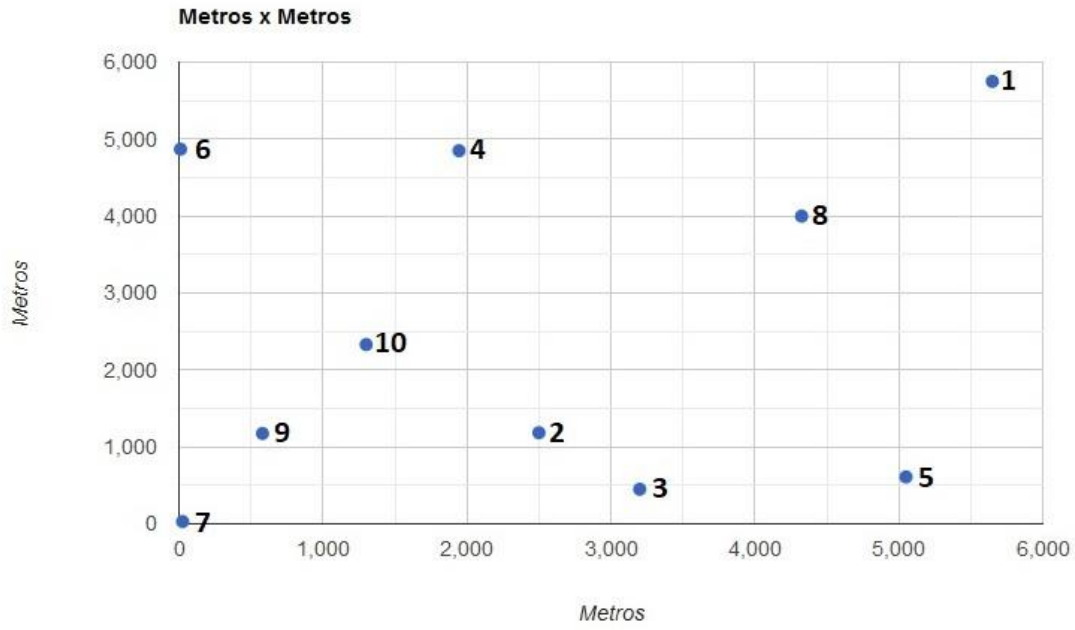

Figura 2.3 –PONTOS10.

Após a realização dos testes com os algoritmos selecionados construiu-se uma tabela com os valores de tempo encontrados para os algoritmos em cada uma das cinco execuções em cada entrada de dados. Posteriormente uma tabela com o tempo médio das

execuções também foi construída. Observou-se também os resultados de percurso obtidos e as distâncias encontradas a fim de verificar a eficiência dos algoritmos em questão.

6. Resultados

Com os dados de entrada definidos e a implementação dos algoritmos realizada na linguagem C++, utilizou-se o ambiente de desenvolvimento Codeblocks para a execução dos mesmos. O computador no qual os testes foram realizados foi um Core i7 – 2.40GHz de quinta geração com 8GB de memória e Windows 10 como sistema operacional.

Como citado na Seção 5, para cada conjunto de dados foram realizadas 5 execuções em cada algoritmo. A Tabela 1 dispõe dos resultados de tempo e do tamanho do percurso em cada execução. Alguns algoritmos divergem em relação ao percurso obtido a cada execução. Foi considerada a distância do percurso obtido.

Tabela 1. Tempo e distâncias obtidas

	PONTOS4	**PONTOS7**	**PONTOS10**
Força Bruta	1. 0.041 s – 16,625 km 2. 0.037 s – 16,625 km 3. 0.035 s – 16,625 km 4. 0.039 s – 16,625 km 5. 0.033 s – 16,625 km	1. 1.340 s – 18,822 km 2. 1.337 s – 18,822 km 3. 1.235 s – 18,822 km 4. 1.299 s – 18,822 km 5. 1.448 s – 18,822 km	1. 480.994 s – 22,495 km 2. 434.062 s – 22,495 km 3. 524.660 s – 22,495 km 4. 560.082 s – 22,495 km 5. 516.772 s – 22,495 km
VND	1. 0.031 s – 16,625 km 2. 0.031 s – 16,625 km 3. 0.033 s – 16,625 km 4. 0.034 s – 16,625 km 5. 0.030 s – 16,625 km	1. 0.053 s – 18,822 km 2. 0.050 s – 20,967 km 3. 0.040 s – 19,099 km 4. 0.058 s – 19,099 km 5. 0.048 s – 19,522 km	1. 0.069 s – 22,495 km 2. 0.061 s – 22,917 km 3. 0.076 s – 22,917 km 4. 0.080 s – 22,917 km 5. 0.051 s – 22,917 km
Vizinho Mais Próximo	1. 0.023 s – 16,625 km 2. 0.031 s – 16,625 km 3. 0.037 s – 16,625 km 4. 0.019 s – 16,625 km 5. 0.022 s – 16,625 km	1. 0.027 s – 20,317 km 2. 0.021 s – 20,317 km 3. 0.021 s – 20,317 km 4. 0.024 s – 20,317 km 5. 0.022 s – 20,317 km	1. 0.037 s – 22,917 km 2. 0.022 s – 22,917 km 3. 0.026 s – 22,917 km 4. 0.026 s – 22,917 km 5. 0.021 s – 22,917 km
Algoritmo Genético	1. 0.086 s – 16,625 km 2. 0.061 s – 16,625 km 3. 0.068 s – 16,625 km 4. 0.061 s – 16,625 km 5. 0.067 s – 16,625 km	1. 0.115 s – 18,822 km 2. 0.115 s – 18,822 km 3. 0.110 s – 18,822 km 4. 0.114 s – 18,822 km 5. 0.125 s – 18,822 km	1. 0.120 s – 22,495 km 2. 0.117 s – 22,495 km 3. 0.140 s – 22,495 km 4. 0.118 s – 22,495 km 5. 0.102 s – 22,495 km

O algoritmo de Força Bruta sempre retorna o melhor caminho. O resultado encontrado por essa solução serve, inclusive, como medida para comparação com os resultados obtidos pelos demais algoritmos. Observou-se, porém, que a medida em que a quantidade de pontos cresce, seu uso se torna inviável. Isso acontece pois ele realiza uma busca exaustiva em que analisa todas as possibilidades possíveis no campo amostral. Numa amostra com dez pontos pôde-se verificar que o tempo de execução pode chegar a 560,082 segundos, o equivalente a mais de 9 minutos. Sua complexidade é de tempo fatorial, O(n!).

A Figura 3.1 apresenta o melhor caminho para o conjunto de testes com baixa quantidade de pontos. É também o percurso apresentado pelo algoritmo de Força Bruta.

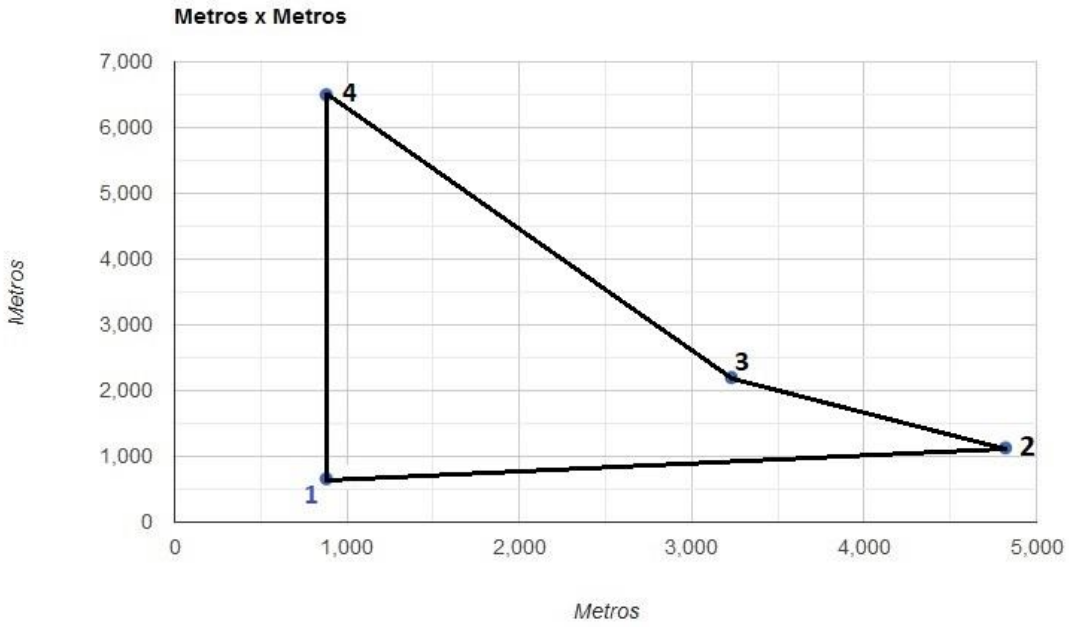

Figura 3.1 – Melhor solução para o conjunto de dados PONTOS4

A figura 3.2 representa o melhor caminho possível para o conjunto de dados com média quantidade de pontos.

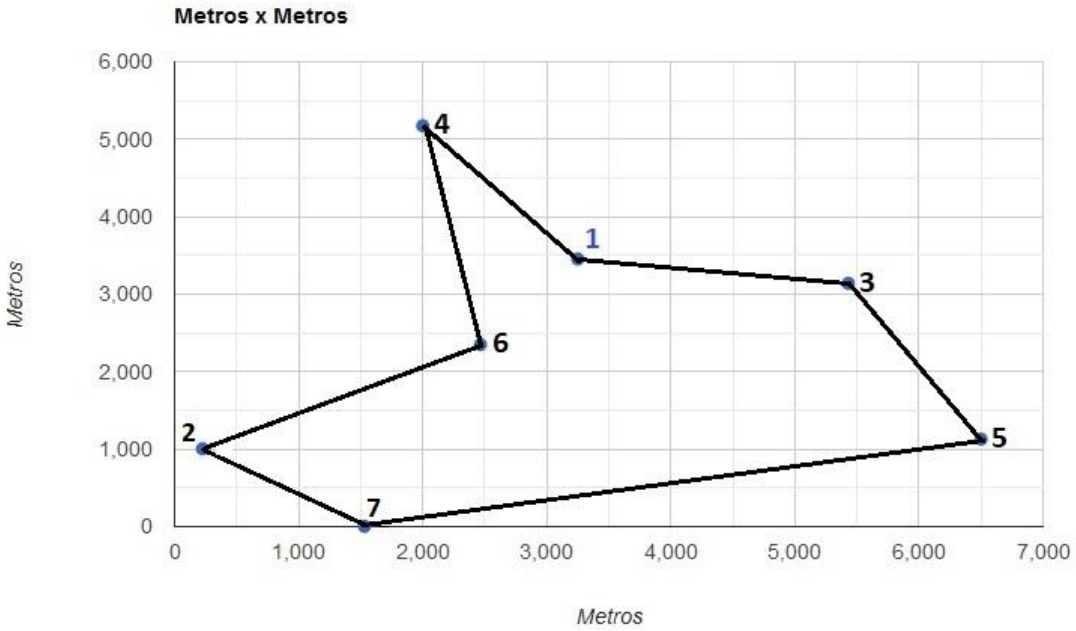

Figura 3.2 – Melhor solução para o conjunto de dados PONTOS7

Na figura 3.3 está representado o melhor caminho para o conjunto de entrada com grande quantidade de pontos (PONTOS10). Pode-se observar que por se tratarem de caminhos não direcionados, os conjuntos de dados têm pelo menos dois caminhos que

satisfazem a condição de menor percurso. Seguindo o exemplo em questão, o menor caminho pode ser representado pela sequência 1-8-5-3-2-7-9-10-6-4-1, assim como pelo seu inverso 1-4-6-10-9-7-2-3-5-8-1.

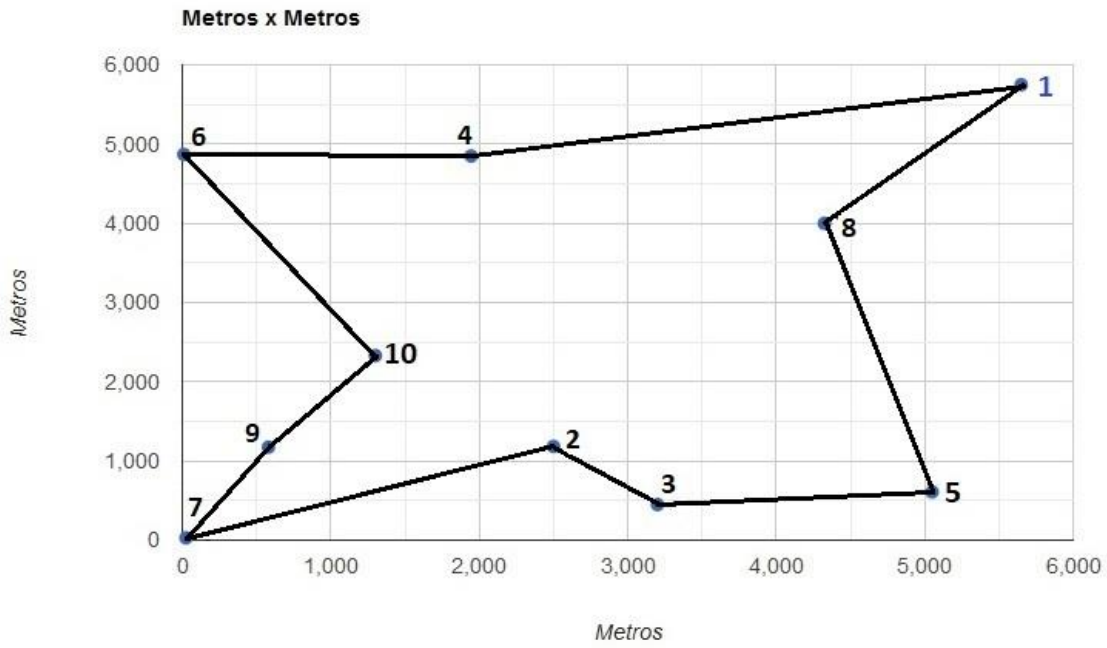

Figura 3.3 – Melhor solução para o conjunto de dados PONTOS10

Assim como a figura 3.1, as figuras 3.2 e 3.3 representam o caminho obtido utilizando o algoritmo de Força Bruta.

A tabela 2 apresenta o tempo médio e a distância média obtida após a realização das cinco execuções.

Tabela 2. Tempo médio obtido

	PONTOS4	**PONTOS7**	**PONTOS10**
FORÇA BRUTA	0.037 s – 16,625 km	1.3318 s – 18,822 km	503.314 s – 22,495 km
VND	0.0318 s – 16,625 km	0.0498 s – 19,5018 km	0.0674 s – 22,8326 km

VIZINHO MAIS PRÓXIMO	0.0264 s – 16,625 km	0.023 s – 20,317 km	0.0264 s – 22,917 km
ALGORITMO GENÉTICO	0.0686 s – 16,625 km	0.1158 s – 18,822 km	0.1194 s – 22,495 km

Para o conjunto PONTOS4, por se tratar de uma quantidade muito pequena de pontos, todos os algoritmos retornaram o melhor percurso (1-3-2-4-1 e 1-4-2-3-1). Para esse caso, os algoritmos tiveram tempos bastante similares, o que pode ser verificado na tabela 1 e na tabela 2.

Para o conjunto PONTOS7, a média de execução do algoritmo de Força Bruta ultrapassou 1 segundo. O VND e o Vizinho Mais Próximo mantiveram tempos satisfatórios (médias: 0.0498 s e 0.023 s, respectivamente), porém o resultado do percurso não foi o menor possível. Em apenas uma das execuções o VND encontrou o menor percurso (1-4-6-2-7-5-3-1), enquanto nas demais apresentou rotas com caminhos maiores. O Vizinho Mais Próximo apresentou uma rota com aproximadamente 1,5 km acima da menor rota possível. Sua complexidade é $O(N^2)$. Em todas as execuções, o Vizinho Mais Próximo apresentará a mesma rota. Já o algoritmo genético apresentou uma média de tempo um pouco acima dos dois últimos algoritmos citados (0.1158 s), mas bastante satisfatório pois em todas as cinco execuções o melhor caminho foi encontrado (1-4-6-2-7-5-3-1 e 1-3-5-7-2-6-4-1).

No conjunto com grande quantidade de pontos (PONTOS10) o algoritmo de Força Bruta teve um tempo bastante elevado, tornando a sua utilização inviável. A média de tempo nas cinco execuções para esse algoritmo foi de 503.314 segundos, um pouco mais que 8 minutos. O algoritmo do Vizinho Mais Próximo apresentou a melhor média de tempo, porém retornou o pior caminho (22,917 km). O VND em apenas uma ocasião conseguiu localizar o menor percurso e obteve média de 22,8326 km. Já o Algoritmo Genético foi o que obteve melhores resultados. Mesmo com o tempo um pouco acima do VND e do Vizinho Mais Próximo (média: 0.1194 segundos), em todas as execuções conseguiu encontrar a melhor solução para amostra PONTOS10 (1-8-5-3-2-7-9-10-6-4-1 e 1-4-6-10-9-7-2-3-5-8-1).

A Figura 4 representa a variação do tempo médio obtido por cada algoritmo para os conjuntos de dados utilizados. O crescimento do tempo no algoritmo de Força Bruta é tão elevado que não pode ser representado no gráfico utilizando números em escala proporcional. Observa-se também que os outros três algoritmos apresentaram tempos similares.

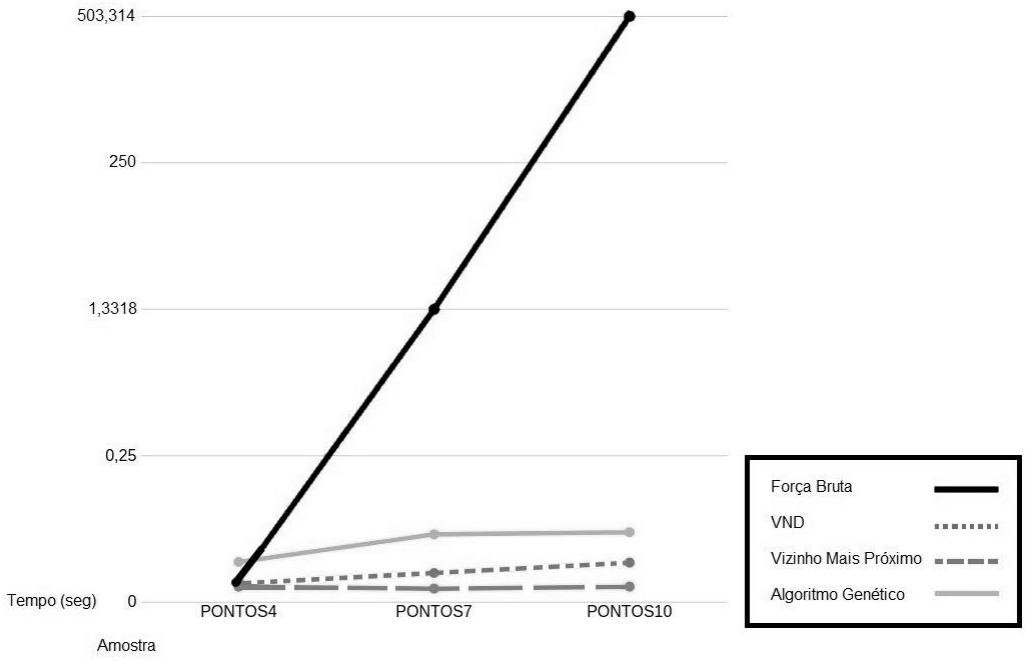

Figura 4. Tempo médio

Após a realização de todas as testagens e análise dos dados, observou-se a prevalência do Algoritmo Genético que, mesmo não obtendo os melhores tempos, apresentou um retorno eficaz, praticamente instantâneo nos três conjuntos de dados. Porém, o principal fator a ser observado é o caminho gerado. Considerando todos os conjuntos de dados, o Algoritmo Genético retornou sempre a melhor solução possível. Dessa forma, dentre os modelos estudados, o AG demonstra ser o paradigma mais indicado para ser aplicado com a finalidade de otimizar a criação de rotas num serviço de entregas.

7. Considerações Finais

Neste trabalho foram realizados experimentos com quatro algoritmos onde observou-se a prevalência do Algoritmo Genético tanto nas questões do tempo quanto na questão da qualidade obtida. Dentre as soluções escolhidas, o AG se mostrou o melhor paradigma para a resolução do problema em questão, sendo assim o mais recomendado para ser aplicado em ferramentas voltadas para criação de rotas otimizadas.

O trabalho em questão se mostrou bastante significativo por realizar um levantamento da literatura e avaliar os algoritmos para o Problema do Caixeiro Viajante, além de ampliar os estudos nessa área, possibilitando assim sua aplicação nos serviços de entrega e demais problemas que apresentam a mesma lógica.

Como proposta para trabalhos futuros existe a possibilidade de desenvolvimento de uma aplicação mobile em que se utilize o algoritmo genético e que possa ser utilizado por um estabelecimento que realize entregas para que haja otimização dos seus serviços. Deve-se observar a questão da integração com alguma API de geolocalização para definir as questões das distâncias, rotas, ruas com sentido único, dentre outros pontos.

8. Referências

Baidoo, E.; Oppong, S. (2016). “Solving the TSP using Traditional Computing Approach.” International Journal of Computer Applications. 152. 13-19. 10.5120/ijca2016911906.

Cazetta, P. P. (2015) “Abordagens heurísticas para tratar o problema do Caixeiro Viajante Preto e Branco” / Paôla Pinto Cazetta. Viçosa, MG.

Colares, F. M.; Silva, J. L. C.; Silva, J. R. O.; Carvalho, M. do S. (2005) “Uma Heurística aplicada ao problema do caixeiro viajante.” In: SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL, 37., 2005, Gramado-RS. Anais... Gramado-RS.

Codeblocks. The open source, cross platform, free C, C++ and Fortran IDE. Disponível em: <http://www.codeblocks.org>. Acesso em: 09 jul. 2019.

Dal-Farra, R. A. P.; Lopes, T. C. (2013) “Métodos Mistos de Pesquisa em Educação: pressupostos teóricos.” Nuances: estudos sobre Educação, Presidente Prudente SP, v. 24, n. 3, p. 67-80, set/dez. 2013. Disponível em: <http://revista.fct.unesp.br/index.php/Nuances/article/view/2698>. Acesso em: 26 nov 2018.

Dantzig, G. B.; Fulkerson, D. R.; Johnson, S. M. (1954) “Solution of a Large-Scale Traveling Salesman Problem”, Operations Research, vol. 2, pp. 393-410.

Feillet, D.; Dejax, P.; Gendreau, M. (2005) “Traveling salesman problems with profits.” Transportation Science, v. 2, n. 39, p. 188-205.

Freitas, F. (2010) “Análise de Algoritmos – Aula 8”, 13 slides. Disponível em: <http://blog.unifimes.edu.br/fernando/wp-content/uploads/sites/2/2010/09/Aula08.pdf> Acesso em: 13 jun 2019.

Fox, K.; Gavish, B., Graves, S. C. (1980) “An n-constraint formulation of the (time dependant) traveling salesman problem.” Operations Research 28 1019_ 1021.

Garey, M. R., Johnson, D. S. (1979) “Computers and Intractability: A guide to the Theory of NP Completeness.” W. H. Freeman, San Francisco.

Goldbarg, M. C.; Luna, H. P. L. (2000) “Otimização combinatória e programação linear: modelos e algoritmos.” Rio de Janeiro. Campus.

Guedes, A da C. B.; Leite, J. N. de F.; Aloise, D. J. (2005) “Um algoritmo genético com infecção viral para o problema do caixeiro viajante.” Disponível em: <http://www.propesq.ufrn.br/publica/artigos-1edicao/et/MSIC-ET-011.pdf>. Acesso em: 09 jul 2019.

Helsgaun, K. (2000) “An effective implementation of the Lin-Kernigham Traveling Salesman Heuristic”, European Journal of Operational Research, v.126, p.106-130.

Holland, J. H. (1975) “Adaptation in Natural and Artificial Systems.” Ann Arbor: University of Michigan Press.

Marques, M. P.; Angélico, B. A.; Abrão, T. (2014) “Semina: Ciências Exatas e Tecnológicas”, Londrina, v. 35, n. 1, p. 63-76, jan/jun.

Miler, C. E.; Tucker, A. W., Zemlin, R. A. (1960) “Integer programming formulations and traveling salesman problems.” Journal of the Association for Computing Machinery 7 326_329.

Mladenovic, N.; Hansen, P. (1997) “Variable neighborhood Search. Computers and Operations Research.” v. 24, p. 1097 – 1100.

Nery, S. W. L. (2017) “Análise de operadores de cruzamento genético aplicados ao problema do Caixeiro Viajante” [manuscrito] / Samuel Wanberg Lourenço Nery.

Oliveira, A. F. M. A. (2015) “Extensões do Problema do Caixeiro Viajante”. Coimbra: UC, 2015. 71 f. Dissertação (Mestrado) - Curso de Mestre em Matemática, Departamento de Matemática, Faculdade de Ciências e Tecnologia, Universidade de Coimbra, Coimbra, 2015. Disponível em: <https://estudogeral.sib.uc.pt/bitstream/10316/31684/1/Tese_AndreOliveira.pdf>. Acesso em: 16 julho 2019.

Pila, A. D. (2006) “História e Terminologia a Respeito da Computação Evolutiva.” Revista de Ciências Exatas e Tecnologia, Valinhos, SP, v. 1, n.1, p. 1-25.

Prodanov, C. C.; Freitas, E. C. D. (2013) “Metodologia do trabalho científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico.” 2ª. ed. Novo Hamburgo: Universiade Freevale.

Rodrigues, M. (2016) “Pesquisa indica que 56% das pessoas fazem pedidos semanais via delivery.” O Estado de S. Paulo, 2016. Disponível em: https://economia.estadao.com.br/noticias/seu-dinheiro,pesquisa-indica-que-56-das-pessoas-fazem-pedidos-semanais-via-delivery,10000060142. Aceso em: 04 dez 2018.

Toscani, L. V, Veloso, P. A. S. (2002) Complexidade de Algoritmos. Porto Alegre: Instituto de Informática da UFRGS: Editora Sagra Luzzatto.

Vianna, D. S. (2004) “Heurística Híbridas para o Problema da Filogenia.” Tese de Doutorado, PUC-Rio, Departamento de Informática, Rio de Janeiro.

Xavier, M. (2018) “A moda de pedir pizza pelo telefone.” Veja São Paulo, 2015. Disponível em: https://vejasp.abril.com.br/blog/30-anos/a-moda-de-pedir-pizza-pelo-telefone/. Acesso em: 04 dez 2018.