CEEL - ISSN 2596-2221
Universidade Federal de Uberlândia
25 a 29 de novembro de 2019

CRIAÇÃO DE UM SISTEMA DE DISTRIBUIÇÃO DAS ROTAS DE ENTREGADORES DELIVERY UTILIZANDO ALGORITMOS GENÉTICOS

Alexandre Henrique Pereira Tavares*[1], Gustavo Carvalho Santos.

FEELT – Universidade Federal de Uberlândia

***Resumo* - Com o avanço e desenvolvimento de novas tecnologias, algumas ações se tornam mais fáceis e rápidas de serem feitas. Com as longas jornadas de trabalho, muitas pessoas optam por opções alimentícias mais cômodas, ao pedir comida fora de casa e pedir para que entreguem a mesma a domicilio. Porém, a insatisfação dos clientes ainda é muito grande com alguns serviços de entrega, devido ao tempo que é gasto no processo.**

Dessa forma, foi desenvolvido um algoritmo que calcula as menores rotas para o número de entregadores e destinos desejados. A aplicação usa algoritmos genéticos e é ligada aos servidores de um mapa online, obtendo as menores distâncias para trajetos utilizando as ruas. Assim, o tempo e distância média para as entregas diminuirão, trazendo menos gastos para quem oferece o serviço e mais satisfação para o cliente, que recebe o produto mais rápido

***Palavras-Chave* – Algoritmos Genéticos; Mapas; Rotas;**

DELIVERT ROUTE DISTRIBUTION SYSTEM

***Abstract* - With the advancement and development of new technologies, some actions become easier and faster. With long working journeys, many people choose for more comfortable food options when ordering take away food from home. However, customer dissatisfaction is still very high with some delivery services mainly due to delays.**

Thus, an algorithm that calculates the shortest routes for the desired number of deliverers and destinations was developed. The application uses genetic algorithms and is linked to the servers of an online map, obtaining the shortest distances for routes using the streets. In that way, the time and average distance to deliveries will decrease, reducing expenses to the service provider and bringing more satisfaction to the customer, who receives the product faster.

***Keywords* - Genetic algorithms; Maps; Routes;**

alexandre.tavares@ufu.br

I. INTRODUÇÃO

O problema do caixeiro viajante é um dos problemas mais estudados de otimização combinatória e tem uma enorme variedade de aplicações. O problema é descrito como um viajante que pretende passar por um conjunto de cidades uma única vez e, após visitar todas as cidades, ele deve voltar à cidade inicial, percorrendo a menor distância possível.

Este problema pode ser desenvolvido de forma a torná-lo mais complexo. Ao invés de apenas um viajante buscando percorrer o melhor (menor) percurso, podem ser incrementados n viajantes. Nesse caso, busca-se o menor percurso possível para que todos os viajantes percorram a menor distância possível sem que nenhum repita a mesma cidade e nem passe por uma cidade já visitada por outro viajante. Por fim, é incrementado o terceiro desafio. Este consiste em que todos os viajantes saiam e voltem para a mesma localidade.

Ao trazer o problema do caixeiro viajante múltiplo à realidade descrita no trabalho, é preciso encontrar uma maneira de distribuir o mais homogeneamente possível as rotas entre os entregadores de delivery, para que a distância percorrida por eles, individualmente, seja a mais próxima possível.

O problema do caixeiro viajante é um problema NP-difícil, ou seja, não pode ser resolvido em tempo polinomial. Isto quer dizer que computar todas as possíveis soluções do problema (todas as possíveis rotas que o viajante pode fazer) necessita de um tempo computacional bastante elevado, o que o torna impraticável para o caso de muitos pontos de parada.

O número de soluções possíveis R para o problema do caixeiro viajante único é mostrado da Equação 1 [1].

$$R = n! \tag{1}$$

Onde:

R - Número de soluções possíveis.
n - Número de pontos de parada.

Para o problema com múltiplos viajantes, o espaço de busca R é multiplicado pelo número de viajantes N, como mostrado na Equação 2:

$$R = N \times n! \tag{2}$$

Onde:

N - Número de viajantes

Assim, devido à sua complexidade computacional, a forma mais frequente de solucionar este problema é utilizando técnicas heurísticas. Essa técnica usa algoritmos que permitem obter boas soluções (não necessariamente ótimas) em tempo consideravelmente inferior. Elas não percorrem todas as soluções possíveis, mas vão caminhando em direção ao ponto ótimo, até que o algoritmo seja encerrado [2].

No presente trabalho foi desenvolvido um algoritmo genético (AG), sendo um exemplo de técnica heurística, para solucionar o problema descrito.

II. ABORDAGENS DO PROBLEMA

Como dito anteriormente, para solucionar o problema de criar as melhores rotas para os entregadores delivery foi desenvolvido um algoritmo genético.

Foi preciso criar técnicas que permitem balancear as rotas distribuídas no sentido de serem menores e serem melhor distribuídas entre todos os entregadores, de forma que um entregador não vá andar muito mais que o outro.

Diante disso, foi usada a estratégia de dividir o problema em dois objetivos distintos. O primeiro é minimizar a distância média percorrida entre todos os n entregadores, de forma que não haja percursos desnecessários, otimizando assim o tempo de entrega e custo de operação e entrega dos pedidos. O segundo objetivo que procurou-se otimizar foi a distribuição entre as n rotas de cada entregador, para realizar esse objetivo, foi feita a minimização do desvio padrão entre as distâncias da rota de cada um dos entregadores. Com isso, impõe que as rotas distribuídas mais homogeneamente são mais bem avaliadas pelo algoritmo.

A seguir é mostrado como foi desenvolvida a interface do sistema com o usuário e quais sistemas de terceiros foram utilizados no projeto.

III. O GOOGLE MAPS API

É um serviço que qualquer pessoa pode utilizar em seus *sites* e aplicações. Existe uma limitação de acessos, que ao exceder o valor máximo delimitado, o usuário deve utilizar a versão paga da API.

Na realidade, o serviço do *Google Maps* possui várias API's que podem ser inseridas na aplicação desenvolvida, dependendo de cada caso desejado.

Seu funcionamento se dá pela combinação e comunicação entre diversos códigos. Eles definem o comportamento dos componentes da interface, de acordo com a programação estabelecida pelo profissional de TI responsável.

A API pode interligar diversos códigos no mesmo *site*, o que torna possível que a aplicação reúna muitas funcionalidades em um só ambiente. A seguir são apresentadas as API's utilizadas no sistema.

A. Geocoding API

A API *Google Geocodig* é uma maneira de utilizar um geocodificador através de um serviço HTTP. A geocodificação transforma endereços em coordenadas, como: Av Rondon Pacheco N° 500, Uberlândia, em: -18.9319823, -48.2900017. Também é possível realizar a operação inversa, transformando coordenadas em endereços.

Esse serviço é utilizado geralmente para geocodificar endereços antes conhecidos para que seja possível localizá-los em um mapa. Esse serviço do Google está limitado a 2.500 consultas ao dia e quem possui o pacote de serviços empresarial da Google pode realizar até 100.000 consultas por dia.

O parâmetro output deve definir o formato da resposta da solicitação, sendo possível informar JSON ou Xml. A recomendação do Google é a utilização do formato JSON (notação de objetos Java Script), já que o tamanho da mensagem de resposta é menor do que o tamanho da mensagem do XML.

B. Distance Matrix API

Para obter a distância entre 2 endereços, a solução está na *Google Distance Matrix API*. Quando é informada uma origem (em forma de coordenadas geográficas) e um destino, a API devolve um JSON ou XML com as distâncias (formato: Dirigindo, Andando e Ciclismo) entre a origem e o destino. Dessa forma é necessário o uso da API de *geocoding* para que se tenham os endereços em forma de coordenadas para usar a *API Distance Matrix*.

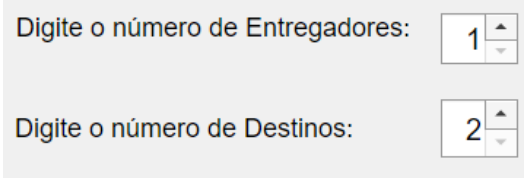
IV. PROTÓTIPO DE INTERFACE GRÁFICA

Um protótipo de interface gráfica foi proposto visando que os usuários da aplicação tenham facilidade no uso. Parâmetros como número de entregadores, número de destinos, modo de cálculo da distância são escolhidos por meio da interface construída.

A. Número de entregadores e destinos

O grande diferencial do algoritmo desenvolvido em relação aos já existentes no mercado é a possiblidade do usuário obter múltiplas rotas de acordo com suas necessidades. Dessa forma, a interface desenvolvida permite ao usuário escolher o número de entregadores disponíveis e a quantidade de endereços que deseja. A Figura 1 ilustra esse processo.

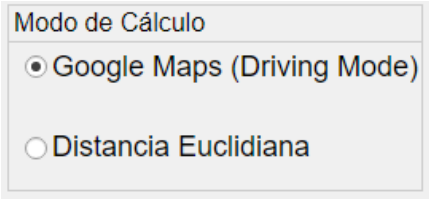
Figura 1: Seleção dos parâmetros



B. Modos de cálculo da distância

Foi verificado na aplicação que o tempo de cálculo da distância entre dois pontos usando o *Distance Matrix* é muito longo. Dessa forma, foi acrescentada a forma de cálculo usando distância euclidiana, a qual obtém a menor distância entre dois pontos. A desvantagem desse último método é que ele não representa a real distância entre os pontos, pois ignora as ruas, obstáculos e curvas do trajeto. A Figura 2 ilustra as opções de cálculo.

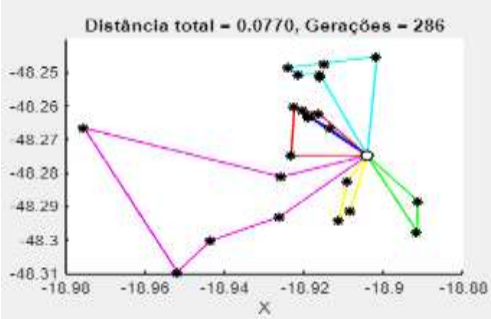
Figura 2: Modos de cálculo da distância



C. Plotagem das rotas

A fim de mostrar a rota obtida, é plotado na interface as melhores rotas para cada entregador, mostrando a ordem de viagem aos destinos ligados por linhas retas, além disso, são salvos os endereços na sequência correta, em que cada entregador deverá percorrer. A Figura 3 ilustra esse gráfico.

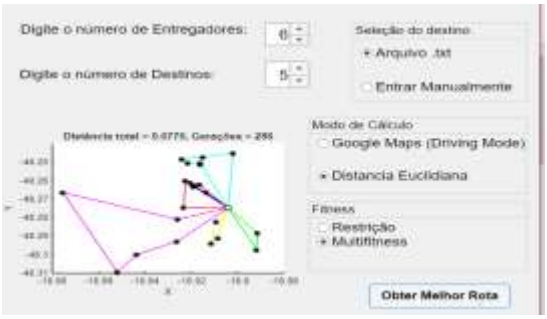
Figura 3: Plotagem das rotas



D. Visão Geral da aplicação

A Figura 4 ilustra a visão geral da aplicação desenvolvida, com todos os parâmetros possíveis de serem escolhidos pelo usuário. Há um botão para a inicialização do processo, deixando a execução mais intuitiva.

Figura 4: Visão Geral da interface gráfica



V. ALGORITMO GENÉTICO

Como dito anteriormente, o Algoritmo Genético (AG) foi a técnica utilizada para a análise e modificação das soluções do problema, de modo a convergir para uma boa solução.

Os algoritmos genéticos baseiam-se em conceitos da biologia e da genética, aliados à matemática e estatística para solucionar problemas de otimização, geralmente, de problemas com alta complexidade, em que o cálculo e análise de todas as soluções possíveis é inviável em questões de gasto computacional e tempo.

No presente trabalho, o AG foi responsável por analisar e modificar as rotas (que são as soluções) de cada um dos entregadores. A otimização das rotas consiste em minimizar a distância média entre as rotas de todos os entregadores e minimizar o desvio padrão entre as distâncias medias, a fim de que as rotas sejam mais homogêneas quanto à distância percorrida.

Cada indivíduo, que sofre as operações genéticas de seleção, cruzamento e mutação, é composto por um conjunto de duas informações, sendo dois vetores. O primeiro vetor contém o índice das posições das coordenadas dos endereços, na ordem em que os entregadores irão visitar. O segundo contém os índices da última parada de cada entregador. Todos os operadores genéticos são efetuados nos dois vetores.

No algoritmo genético criado, foi implementada a função de elitismo com apenas um indivíduo. Para a seleção dos indivíduos, foi utilizada a técnica de torneio (com 8 indivíduos) e depois foi feito um ranking entre os indivíduos do torneio após as operações genéticas e foram escolhidos os 8 melhores para substituírem os outros 8 indivíduos anteriores. Após serem feitas as operações, as soluções sofriam uma correção pelo método 2-opt [3], que elimina as rotas cruzadas, reduzindo assim a distância em cada rota.

A. Operadores genéticos

Os operadores genéticos utilizados na implementação do AG foram o *Partially Mapped Crossover* (PMX) e as mutações *Swap, Flip e Slide*.

O PMX consiste em um crossover de dois pontos de corte em que só há a troca das posições dos genes do indivíduo, ou seja, no problema do caixeiro viajante, após a operação do PMX, há a garantia de que nenhuma cidade se repita na rota gerada.

A mutação *Swap* consiste apenas em trocar de posição dois genes do indivíduo. A mutação *Flip* é a inversão da ordem da rota, sendo que a última posição passa a ser a primeira, e a primeira passa a ser a última. Por fim, a mutação Slide desloca todos os genes em uma casa para a direita, o ultimo gene da direita passa a ser o primeiro da esquerda.

Todas estas operações são realizadas após a seleção dos indivíduos para o torneio, que são selecionados aleatoriamente. Entre os 8 indivíduos selecionados, o melhor é mantido, depois é feito o PMX entre os 2 melhores indivíduos. Após realizado o crossover, são feitas as 3 operações de mutação diferentes, cada uma em 25% da população dentro do torneio, ou seja, cada uma em 2

indivíduos. Depois de ocorrerem estas operações, era feito um ranking e os indivíduos primários do torneio eram substituídos pelos oito melhores gerados após as operações genéticas.

B. Função de Aptidão

Para a função de aptidão, foram usadas duas técnicas, em que o usuário poderia escolher entre uma (apenas uma de cada vez) delas.

A primeira função de aptidão implementada foi a junção de dois objetivos em um só, com o uso da função de penalidade. Os dois objetivos (distância média entre as rotas e desvio padrão) deveriam ser o menor possível. A Equação 3 mostra a primeira função fitness.

$$Fitness = W_{avg} \times D_{avg} + W_{std} \times Std_D \qquad (3)$$

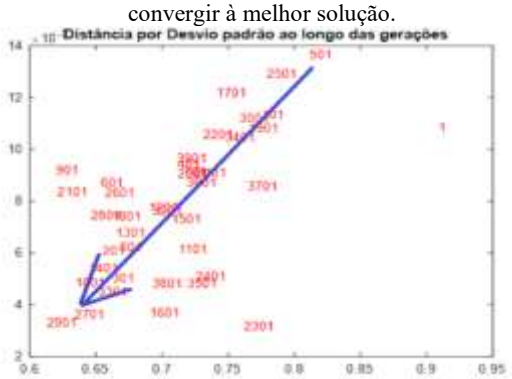
Onde:

W_{avg} - Peso utilizado para a distância média.
D_{avg} - Distância média entre as rotas.
W_{std} - Peso utilizado para o desvio padrão.
Std_D - Desvio padrão entre as distâncias.

Os pesos W_{avg} e W_{std} são analisados após diversos testes e foram definidos para o valor de 1 e 5, respectivamente. Para a segunda função de aptidão, foram mantidos os dois objetivos (caracterizando-se uma função multiobjetivo), sendo visado a diminuição dos dois parâmetros descritos anteriormente. Para isso, foram selecionados apenas os 15 melhores indivíduos de cada população em cada um dos parâmetros, ou seja, foram escolhidas as 15 rotas com menor distância média e as 15 rotas com menor desvio padrão. Esta escolha de apenas as 15 melhores foi feita com o intuito de diminuir o gasto computacional que seria necessário para analisar toda a população.

Após selecionados estes indivíduos, foi feita uma comparação de um a um, e caso um deles tivesse os dos dois parâmetros (distância média e desvio padrão) com valores menores que o do outro indivíduo, este primeiro seria selecionado como o melhor indivíduo. Esta comparação era encerrada ao final da verificação de cada par de indivíduos entre os 15 melhores de cada parâmetro. Com esta técnica, garante-se que o melhor indivíduo seja aquele que esteja mais para baixo e a esquerda na curva de distância por desvio padrão, como é mostrado na Figura 8.

Figura 8: Caminho que as soluções tendem a percorrer para convergir à melhor solução.

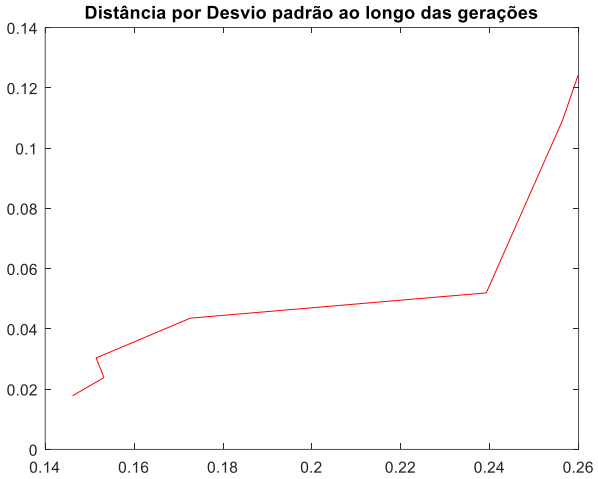


VI. RESULTADOS

A seguir são mostrados alguns resultados obtidos em testes do Algoritmo Genético desenvolvido. Para isto, foi utilizada uma população de 80 indivíduos e o número de gerações foi fixado em 1000.

A Figura 9 mostra a evolução dos indivíduos ao longo das gerações pelo método multiobjetivo, o gráfico mostra os pontos das coordenadas x e y, sendo a distância média e o desvio padrão, respectivamente, do melhor indivíduo de cada geração.

Figura 11: Evolução dos indivíduos ao longo das gerações pelo método multiobjetivo.



As Figuras 10 e 11 mostram a evolução de cada obtida ao longo das gerações pelo método multiobjetivo. Na Figura 10 é mostrada a distância média do melhor individuo de cada geração e na Figura 11, é mostrado o desvio padrão do melhor individuo de cada geração. A Figura 12 mostra o gráfico com as rotas de cada entregador, após a execução do algortimo genético.

Figura 10: Distância média do melhor indivíduo de cada geração pelo método multiobjetivo.

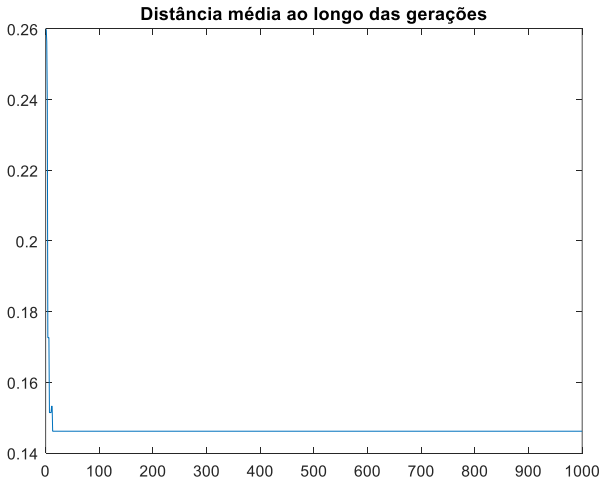


Figura 11: Desvio padrão do melhor indivíduo de cada geração pelo método multiobjetivo.

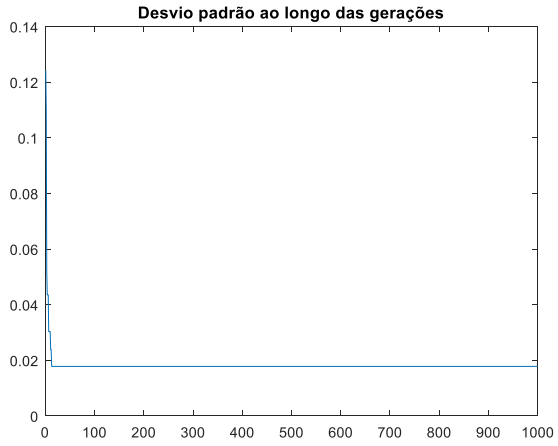
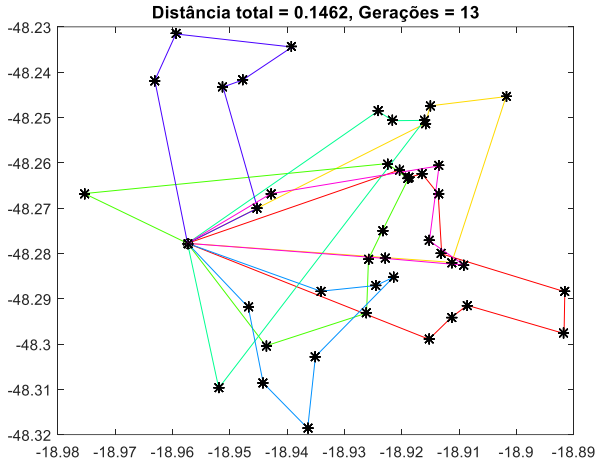


Figura 12: Rotas de cada entregador (cada cor é uma rota) e os pontos são os endereços de parada. Método multiobjetivo.



As Figuras 13 a 16 mostram o mesmo resultado, mas desta vez pelo método monoobjetivo com penalidade no fitness.

Figura 13: Evolução dos indivíduos ao longo das gerações pelo método monoobjetivo com penalidade no fitness.

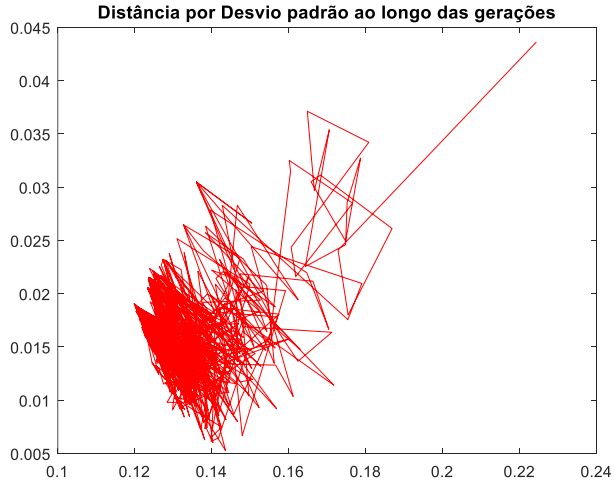


Figura 14: Distância média do melhor indivíduo de cada geração pelo método monoobjetivo

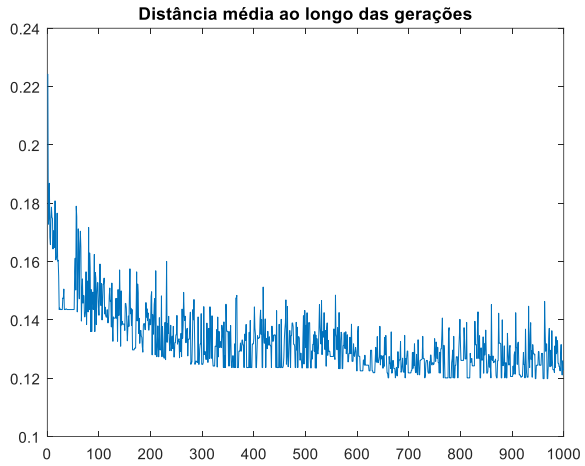


Figura 15: Desvio padrão do melhor indivíduo de cada geração pelo método monoobjetivo

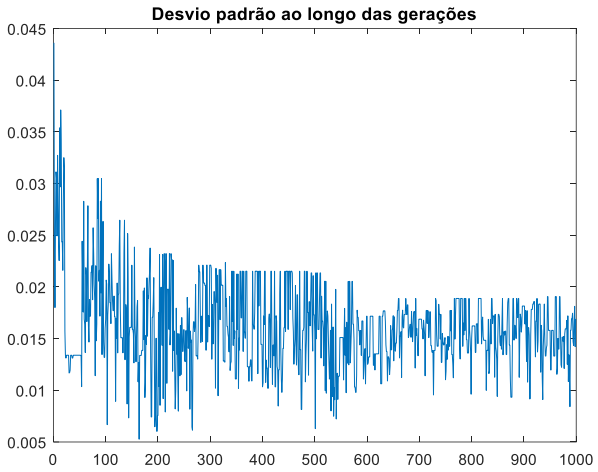
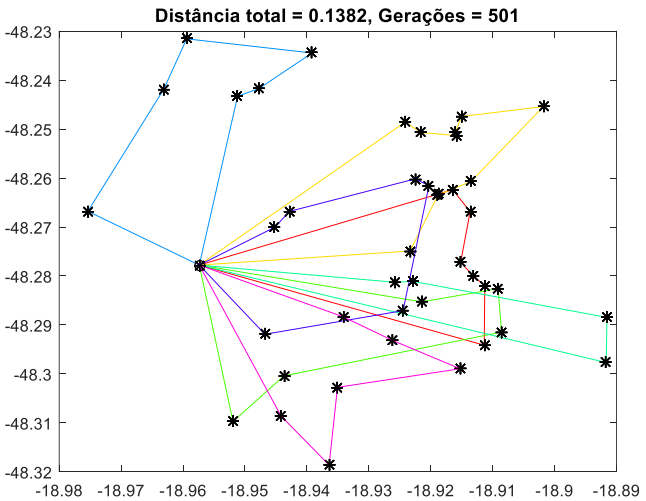


Figura 16: Rotas de cada entregador (cada cor é uma rota) e os pontos são os endereços de parada. Método monoobjetivo



VII. DISCUSSÃO E CONCLUSÕES

Diante dos resultados apresentados, é possível notar a diminuição tanto do objetivo 1 (distância média entre as rotas), quanto do objetivo 2 (desvio padrão). Os dois métodos convergiram em valores finais próximos quanto aos dois objetivos, ao final das gerações, porém é nítido que houve muito mais instabilidade no método monoobjetivo com penalidade no fitness.

Este resultado pode ser explicado pelo fato que o método multiobjetivo só reconhece um indivíduo como sendo melhor, somente se os dois objetivos sejam melhorados ao mesmo tempo, assim não há instabilidade ou aumento de nenhum parâmetro ao longo das gerações, mas somente a diminuição deles (visto que é um problema de minimização).

No método monoobjetivo com penalidade no fitness, o algoritmo pode ser confundir ao escolher o melhor indivíduo da população, pois a fórmula leva em consideração os dois parâmetros, mas com pesos diferentes, assim pode haver casos em que um objetivo seja bem melhor que o outro, mas a rota não é considerada boa.

A Aplicação se comportou bem. Nos testes realizados as rotas escolhidas pelos usuários foram calculadas da forma esperada. A utilização da técnica 2-opt aprimorou os resultados, pois evitou as rotas cruzadas. O uso da API que requisita os dados de endereço se mostrou muito lento, fazendo com que os testes que exigiam muitos endereços fossem feitos usando distância euclidiana.

Futuramente, pretende-se converter o algoritmo para uma nova linguagem, que seja open-source e que possa ser utilizada comercialmente. Uma nova interface utilizando serviço web e aplicação móvel também é um objetivo, facilitando assim a utilização pelo usuário final.

VIII. REFERÊNCIAS

[1] 1. Araújo Oliveira A. F. M., Extensões do Problema do Caixeiro Viajante, Disponível em https://estudogeral.uc.pt/bitstream/10316/31684/1/Tese_AndreOliveira.pdf, Acesso em 02/12/2018

[2] 2. Silva Pereira M. M., O problema do caixeiro viajante, Disponível em http://vigo.ime.unicamp.br/Projeto/2014-1/ms877/ms877_marie.pdf, Acesso em 02/12/2018

[3] Pedro h. s. f., Optimization with 2-Opt, Disponível em http://pedrohfsd.com/2017/08/09/2opt-part1.html.