Universidade Federal Rural de Pernambuco

Departamento de Estatística e Informática

Bacharelado em Sistemas de Informação

Projeto FlyFood

Diego Diniz da Cruz Gomes

Recife

abril de 2022

1. Introdução

No ano de 2025, com o grande crescimento da população e do número de veículos nas ruas, tornou-se praticamente impossível realizar entregas em um curto espaço de tempo, pois, além de tudo, as ofertas de emprego estão muito altas e a mão de obra cada vez mais cara. Diante disso, para minimizar os custos e o tempo de entrega, se fez necessário pensar em novas formas para melhorar a mobilidade do serviço de delivery, surgindo assim uma empresa chamada FlyFood, criada por um ex-aluno do curso de BSI da UFRPE, tendo o objetivo de fazer entregas utilizando drones, que possuem a capacidade de sair com vários pedidos e levá-los para diversos pontos espalhados pela cidade, otimizando de forma significativa o tempo de entrega. Porém, apesar de todas as vantagens que a utilização desses dispositivos oferecem, a duração de suas baterias são bastante limitadas, consequentemente tendo que adicionar paradas para recarga no meio do percurso. Portanto, o projeto consiste em criar um algoritmo que consiga escolher o melhor caminho para que seja feito o maior número de entregas no menor espaço de tempo antes que seja necessário parar em uma estação de carregamento.

O projeto tem como objetivo geral propor solução algorítmica que retorne a solução ótima para o problema, ou seja, a solução da melhor rota a ser percorrida pelo drone. Essa implementação será feita em python e a representação em forma de grafos.Para obter as melhores rotas e coordenadas GPS será usada uma matriz contendo pontos de entrega, pontos de partida e pontos de retorno, considerando que os drones não podem andar na diagonal, além das fórmulas para calcular a menor distância, ou seja, a distância ideal que deverá ser percorrida pelo drone ao passar pelo menor percurso e fórmulas para fazer a permutação de todos os caminhos possíveis que serão percorridos pelo programa .

2. Referencial Teórico

2.1 O problema do caixeiro viajante

Os caixeiros viajantes referem-se aos comerciantes viajantes que vendem seus produtos fora do local de origem, ou seja, circulam pelas ruas para vender seus produtos, principalmente manufaturados. Profissão milenar que se tornou fundamental em uma época em que o transporte entre cidades era inconveniente, naquela época o

caixeiro-viajante era a única forma de transportar produtos entre diferentes regiões fora das grandes cidades.

O problema do caixeiro viajante é um exemplo clássico de um problema de otimização combinatória. Para resolver esse tipo de problema, o primeiro passo é reduzi-lo a um problema de enumeração: encontrar todas as rotas possíveis, usar um computador para calcular o comprimento de cada rota e ver qual rota é a mais curta. É possível dividi-lo em 2 soluções diferentes: algoritmo de força bruta e algoritmo genético.

No algoritmo de força bruta será feita a pemurtação de todas as rotas possíveis

2.2 Classes de Completude

Na teoria da complexidade computacional, a classe de complexidade é um conjunto de problemas relacionados a recursos computacionais baseados em complexidade. Por exemplo, a classe NP é o conjunto de problemas decidíveis que podem ser resolvidos por uma Máquina de Turing Não-Determinística em tempo polinomial, onde a classe PSPACE é o conjunto de problemas decidíveis que podem ser resolvidos por uma Máquina de Turing Determinística em espaço polinomial. A classe de complexidade é o subconjunto dos problemas NP de tal modo que todo problema em NP se pode reduzir, com uma redução de tempo polinomial, a um dos problemas NP-completo, que por sua vez são os problemas mais difíceis de NP e muito provavelmente não formem parte da classe de complexidade P. A razão é que se conseguisse encontrar uma maneira de resolver qualquer problema NP-completo rapidamente (em tempo polinomial) , então poderiam ser utilizados algoritmos para resolver todos problemas NP rapidamente. Na prática, o conhecimento de NP-completo pode evitar que se desperdice tempo tentando encontrar um algoritmo de tempo polinomial para resolver um problema quando esse algoritmo não existe.

2.3 Classificando o Projeto

Problemas NP-completo são estudados porque a habilidade de rapidamente verificar soluções para um problema (NP) parece correlacionar-se com a capacidade de resolver rapidamente esse problema, por isso, é possível afirmar que o projeto FlyFood se encaixa nessa classificação, já que ele é, em sua essência, equivalente ao problema do caixeiro viajante, que por sua vez é considerado um problema decisório

3. Procedimento

Nesse projeto será utilizado a meta-heurística, sendo considerado o método mais promissor para resolver problemas de programação inteira. Enquanto algoritmos exatos(ou de força bruta) garantem soluções ótimas para certos tipos de problemas, heurísticas não podem provar a otimalidade de suas soluções, mas podem fornecer soluções aceitáveis mesmo para problemas complexos e grandes, que são computacionalmente caros, com baixo custo. Dentre os algoritmos metaheurísticos, os Algoritmos Genéticos (AG), inspirados em mecanismos de evolução natural e genética, surgiram como uma técnica de otimização robusta e eficiente.

3.1 Algoritmos genéticos

Os algoritmos genéticos são uma família de modelos computacionais inspirados na evolução que incorporam soluções potenciais para problemas específicos em estruturas semelhantes a cromossomos e aplicam operadores de seleção e cruzamento a essas estruturas para preservar informações relevantes importantes. Os AG são subclasse de Algoritmos Evolutivos baseados em teorias biológicas, frequentemente vistos como um otimizador funcional e replicam os conceitos biológicos em códigos, embora o número de problemas aos quais o AG é aplicável seja bastante abrangente. Esse algoritmo tem como objetivo principal a otimização, seja ele, um conjunto de testes para identificar os indivíduos mais adequados, ou até mesmo uma “caixa preta” onde só conhecemos o formato da entrada e retornamos o valor que queremos otimizar. A grande vantagem dos algoritmos genéticos é que não precisamos saber como funciona essa função objetivo, basta aplicá-la a indivíduos e comparar os resultados.

O seu funcionamento ocorre de forma em que há indivíduos em uma população inicial, onde esse indivíduo é uma possível solução para o problema. Esses indivíduos são avaliados por meio de uma função de avaliação específica para o problema, e os melhores indivíduos são selecionados e "criados" por meio de um processo de recombinação (crossover) e mutação, resultando em uma nova geração de indivíduos. Por fim, é realizado um processo de ajuste populacional para manter o número inicial de indivíduos na população. Esse processo se repetirá na nova geração até atingirmos o critério de parada, que será quando o algoritmo for executado por determinado número de vezes .

Inicialmente será gerada uma população de indivíduos (rotas) aleatoriamente, posteriormente, sendo feito o rankeamento para a forma de seleção de pais através de torneio, que ocorrerá entre 2 indivíduos de forma randômica, chamando a função de fitness para determinar os vencedores. Quanto mais próximo de 1 for o fitness, melhor será o indivíduo.

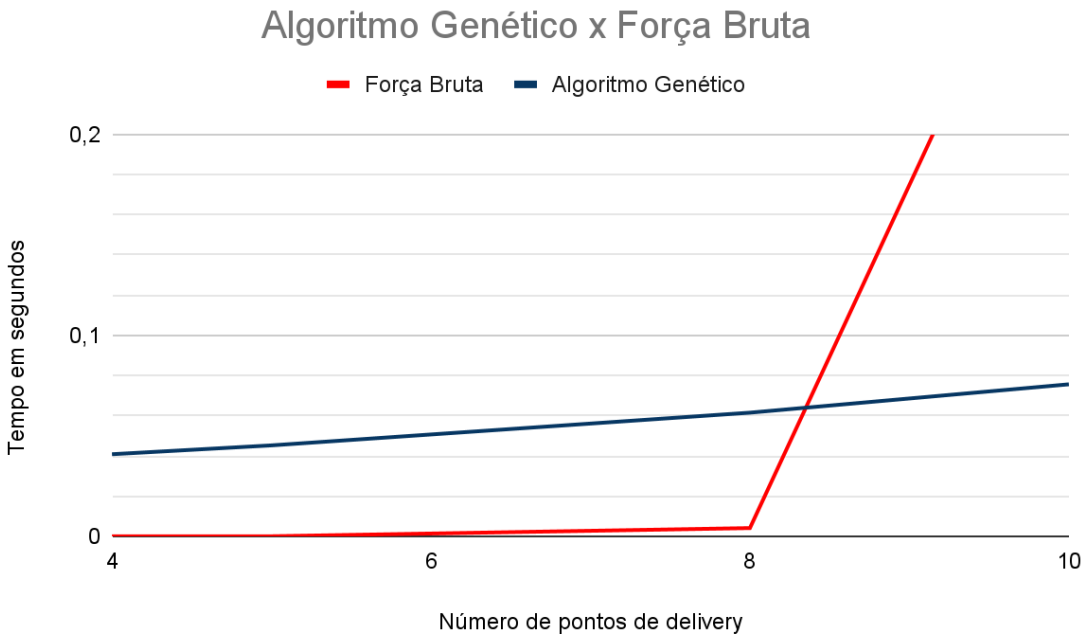
Em seguida, após feita a seleção de pais, no crossover haverá a seleção dos melhores indivíduos para cruzamento, que ocorre de forma aleatória, no qual uma certa quantidade dos genes do pai A ou pai B são selecionados aleatoriamente e inseridos sequencialmente na composição dos cromossomos dos filhos. O ponto de corte ocorrerá de forma aleatória, não podendo ser localizado no primeiro gene nem no último.

Feito o cruzamento, é realizado o processo de mutação para aumentar a diversidade da população e reduzir a convergência genética. A taxa de mutação por cromossomo foi ajustada em 5%. O processo de mutação ocorre inicialmente gerando valores aleatórios para definir se o gene sofre mutação ou não, ocorrendo caso esses valores estejam de acordo com a taxa de mutação. Posteriormente, é escolhido um gene aleatoriamente e é feita a troca com o seu próximo.

Em seguida, é feito um controle populacional para reduzir o tamanho da população de acordo com um valor predeterminado. Serão escolhidos dois indivíduos aleatoriamente e caso eles sejam iguais, um deles será removido, caso contrário será removido da população aquele que tiver o menor fitness.

4. Resultados

Para analisar os resultados e comparar força bruta e algoritmos genéticos, serão utilizados casos de teste. A medição será feita pelo tempo (em segundos) que leva para executar o script, esta medição será feita pelo módulo *time*. Além disso, será analisada a qualidade da solução do algoritmo genético com diferentes parâmetros na taxa de mutação de cada gene.



5. Conclusão

O objetivo deste trabalho foi desenvolver um algoritmo que satisfizesse as demandas do contexto atual, sendo ele entregar a solução ótima e de forma pouco custosa. De acordo com os dados analisados, o objetivo foi alcançado de forma clara.

Analisando o gráfico, pode-se observar que o algoritmo de força bruta só é viável para um número limitado de pontos de queda, pois quando o limite de pontos de queda é ultrapassado, mesmo um pequeno aumento resultará em uma explosão combinatória. Além disso, os algoritmos genéticos são sustentáveis quando existem múltiplos pontos de entrega e seu tempo de execução permanece o mesmo. Portanto, é muito caro usar algoritmos de força bruta quando há um grande número de entregas, o que torna os algoritmos genéticos mais eficazes nessas situações.

Referências Bibliográficas

Beraldi, L. C., & Escrivão Filho, E. (2000). Impacto da tecnologia de informação na gestão de pequenas empresas. Revista Ciência da Informação, Brasília, 29(1), 46-50.

Bishop, C. M. (2006). Pattern recognition and machine learning. springer.

BRASIL (2017). Lei Complementar nº 123, de 14 de dezembro de 2006. Institui o Estatuto Nacional da Microempresa e da Empresa de Pequeno Porte. Disponível em: http://www. planalto. gov. br/ccivil_03/leis/LCP/Lcp123. htm. Acesso em 07 dez. 2017

Conselho Regional de Contabilidade do Paraná (2017) . Balanço Patrimonial Disponível em: https://goo.gl/Lb4jzw. Acesso em 08 dez. 2017