APLICAÇÃO DE ALGORITMO GENÉTICO PARA O PROBLEMA DE TRANSPORTE DE CARGA

Wallaks Cardoso da Silva[1]
Prof º Maycon Guedes Cordeiro[2]

[1]Graduando em Análise e Desenvolvimento de Sistemas
Instituto Federal do Espírito Santo – IFES – Campus Santa Teresa

[2]Mestre em Pesquisa Operacional e Inteligência Computacional pela UCAM-Campos (2011)

Resumo: Apresenta-se neste resumo expandido a relação entre algoritmo genético e o problema de transporte de carga, em que se objetiva a encontrar uma solução ótima para dispor um maior número itens em um local, cujo espaço é limitado. Dessa forma será usada como base, a figura de um caminhão que possui um espaço interno em metros cúbicos, levando consigo produtos com valores em reais. Para o desenvolvimento será utilizado um cromossomo representando os itens a serem selecionados. Essa representação cromossonica baseiam-se em: genes totalmente aleatórios, cruzamentos e mutações.
Palavras-chave: Algoritmo Genético; Otimização; Cromossomo.

Abstract: In this expanded summary, the relationship between genetic algorithm and the cargo transport problem is presented, in which the objective is to find an optimal solution to arrange a larger number of items in a location, whose space is limited. Thus, the figure of a truck that has an internal space in cubic meters will be used as a base, taking with it products with values in reais. For development, a chromosome representing the items to be selected will be used. This chromosomal representation is based on: totally random genes, crossovers and mutations.
Keywords: Genetic Algorithm; Optimization; Chromosome.

1 INTRODUÇÃO

O planejamento para o transporte de itens é um processo que carrega junto a si uma logística administrativa, que demanda dos profissionais um alto grau de experiência no que diz respeito a conhecer, de fato, como essa distribuição deve acontecer. Nesse caminho é perdido tempo e lucratividade deixando a experiência e improvisação do processo em posições difíceis de se contornar. A falta de informatização de alguns meios no ambiente empresarial, torna um desafio o sucesso financeiro e consequentemente deixando-a exposta para quem acompanha o cenário tecnológico.

Atualmente técnicas computacionais para análise de problemas de transporte já estão listadas em vários estudos, visto que os paises estão em trocas constantes de mercadorias. Dessa forma, esses ambientes, necessitam de alternativas automatizadas que gerem respostas satisfatórias com menor custo operacional. É de extrema importância no planejamento orçamentário das instituições, principalmente aquelas que aplicam ou trabalham a logística. Uma dessas alternativas tem sido mais estudada e, consequentemente, mais aplicada, principalmente a partir da segunda metade da década de 1980. Trata-se dos algoritmos genéticos.

Algoritmos Genéticos são algoritmos de otimização global, baseados nos mecanismos de seleção natural de Darwin – sobrevivência dos mais adaptados – e nas leis de Mendell. Eles empregam uma estratégia de busca paralela e estruturada, mas aleatória, que é voltada em direção ao reforço da busca de pontos de "alta aptidão", ou seja, pontos nos quais a função a ser minimizada (ou maximizada) tem valores relativamente baixos (ou altos). Tais características tornam os AGs mecanismos extremamente úteis no trabalho com modelos que envolvem estruturas com graus elevados de complexidade, mas também trazem rezultados satisfatórios em problemas com grau de complexibilidade mais baixo.

Por esses motivos, procurarei mostrar neste resumo expandido, as relações dos algoritmos genéticos com a resolução de problemas de alocação de itens em um determinado espaço, com uma logística de atribuir variáveis aos produtos como preço e tamanho.

2 ALGORITMOS GENÉTICOS (AGs)

O algoritmo genético é um método de otimização usado para obter soluções ótimas em um determinado ciclo que se rompe quando encontra um valor previamente disposto ou quando existe um ponto de parada, por exemplo, por geração. Desenvolvida por John Henry Holland em 1975. Segundo a teoria evolucionária de Charles Darwin, os seres sofrem mutações e os mais aptos (melhores) vão sobreviver. Então investigar como ocorre tal seleção natural e efetuar simulação permite gerar dados com características desejadas e se não encontrar uma ótima solução, os AGs trazem valores próximos aos desejáveis.

Os AGs podem ser usados para diversos problemas com parâmetros variados. O problema de Corte *cutting stock problem* (CSP), o Problema do Cacheiro Viajante (The Travelling Salesman Problem-TSP) e o problema da Mochila (Knapsack problem), são exemplos clássicos de NP-difíceis, ou seja, não aproximáveis – em termos absolutos – além de certas constantes, a menos que P=NP. Os parâmetros podem ser representados através de cadeias de caracteres, números reais ou sequências de bits. Pela própria forma como os computadores trabalham, a codificação binária é a preferida.

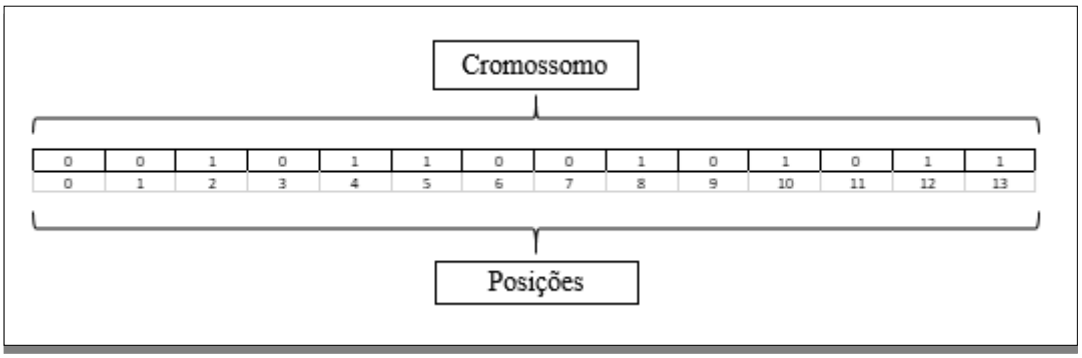

Figura 1: Representação cromossômica binária em um AG.

O tratamento de um problema com algoritmos genéticos inicia-se com a determinação de um conjunto de soluções iniciais (escolhidas aleatoriamente, mas dentro de certas restrições) que forma o espaço de busca. Cada possível resposta dentro desse conjunto é um cromossomo, o qual é composto por genes. Esses genes representam parâmetros que devem ser ajustados para melhorar a solução e, consequentemente, aperfeiçoar a função analisada.

Para o desenvolvimento dos melhores resultados do AG ultiliza-se alguns operadores do própio algoritmo. Esses operadores são basicamente conhecidos como cruzamento, mutação e seleção.

A mecanica do projeto gira em torno do seguinte processo: cria-se uma população inicial de indivíduos de forma aleatória, esses individuos são avaliados e assim se obtem o fitness de casa um. Feito isso, os individuos são ordenados e os melhores ficam nas posições iniciais da lista. Dessa forma eu garanto que mesmo em uma geração inicial aleatória, os melhores já serão separados para realizar o cruzamento. O próximo passo será o cruzamento desses dois indivíduos melhores na lista, denominados pai um e pai dois. O pai um troca genes com o pai dois a partir de um ponto de corte no cromossomo, também de forma aleatória, e obtem-se os filhos, que serão denominados de filho um e filho dois. Esses filhos serão inseridos na população que foi gerada aleatória passando por uma nova ordenação.

Esse sistema de crossover representa a reprodução sexuada, pois na reprodução assexuada cada filho é idêntico a seu genitor e tem as mesmas habilidades, o que não cria diversidade e acaba

prejudicando o desenvolvimento do código, uma vez que diversidade é um ponto chave para a evolução.

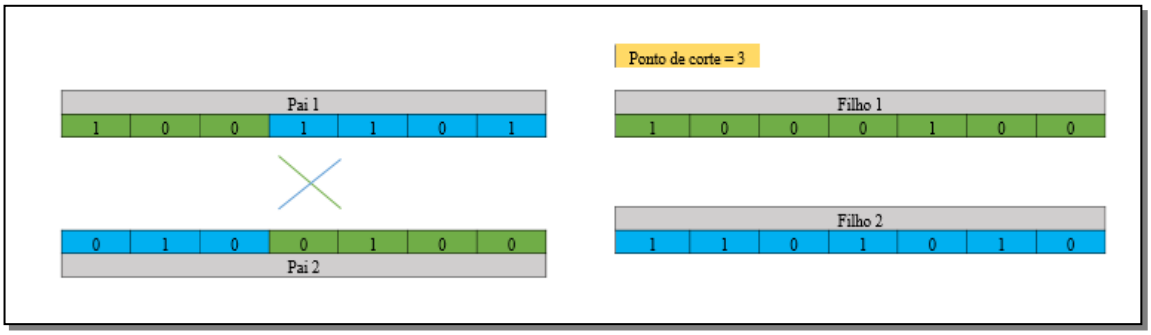

Figura 2: Operação de cruzamento (*crossover*) em um AG.

Pai 1	1	0	0	1	1	0	1
Pai 2	0	1	0	0	1	0	0
Filho 1	1	0	0	0	1	0	0
Filho 2	1	1	0	1	0	1	0

Tabela 1: Exemplo de um resultado no processo de cruzamento.

Para garantir que o processo seja variado e evitar que as soluções convirjam rapidamente, é aplicado o processo de mutação, como descrito acima, em que selecionamos aleatoriamente uma gene e trocamos a sua propriedade para o oposto, ou seja, caso na posição selecionada tenha um gene com o valor de 1, o mesmo é trocado para 0. Caso contrario é feito o mesmo processo, porém trocando para 1. Assim podemos ver uma alteração no cromossomo agregando a ele uma pequena variação. É importante destacar que esse gene será um novo membro da população e poderá ser avaliado e cruzado novamente. Sendo assim, digamos que o indivíduo selecionado aleatóriamente foi o da posição três do cromossomo. A figura a seguir mostra como ficou o resultado.

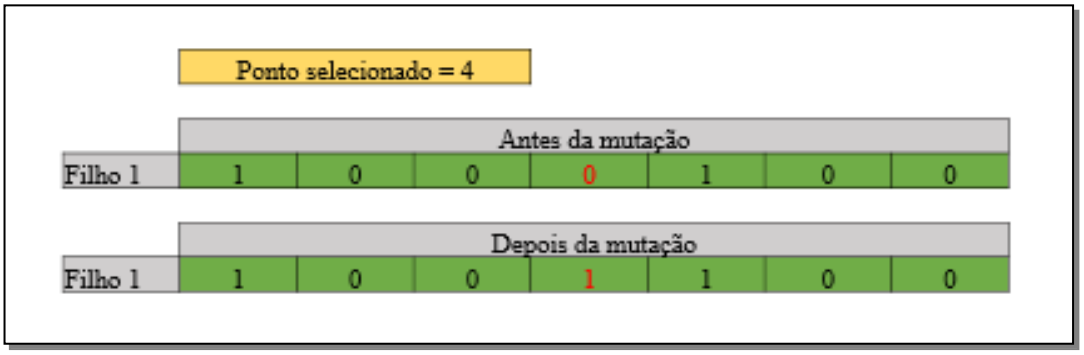

Figura 3: Processo de mutação em um AG.

Ao elaborar um processo de resolução que envolva algoritmos genéticos, as taxas de cruzamento e mutação devem ser cuidadosamente analisadas. Como o AG é primordialmente estocástico, ou seja, aleatório, essas taxas são tratadas como probabilidades. A probabilidade de cruzamento pode situar- se entre 0 e 100%, onde no primeiro caso cada nova geração é uma cópia exata da anterior e no segundo, a nova geração é totalmente formada por cruzamento entre o primeiro e segundo indivíduo da lista de indivíduos. Estudos mostram que uma probabilidade de crossover em torno de 95% é a ideal na maioria dos casos, uma vez que, como dito anteriormente, manter algumas cópias das melhores soluções melhora a otimização. No caso da probabilidade

de mutação, esta deve ser mantida num patamar relativamente baixo – menor que 5% –, pois sua utilidade é garantir que o AG não caia num extremo local muito rapidamente.

É importante preservar a boas soluções levando elas para as próximas gerações. Esse sistema ocorre no projeto, quando os dois melhores indivíduos são dispostos na posição zero e posição um da lista. Assim é possível realizar o cruzamento entre eles e obter um resultado ainda melhor, elevando o grau de aptidão do cromossomo. A isso dá o nome de elitismo.

Devem ser feitas algumas escolhas iniciais para implementar nossos AGs. Primeiramente devemos considerar o tamanho de nossa população. Populações muito pequenas dão pouca margem para cruzamentos e o espaço de busca será pouco explorado. Usando populações muito grandes teremos um elevado consumo de processamento, visto que o número de interações cresce exponencialmente com a quantidade de cromossomos em cada geração. Pesquisas revelam que uma população composta por aproximadamente 30 membros são uma boa escolha para a maioria dos problemas.

A forma como são escolhidos os cromossomos a serem cruzados em cada população varia de acordo com o problema, mas podemos citar seleção por roleta, Boltzman, campeonato, dentre outros. Analisaremos os métodos roleta e classificação.

No projeto em questão, a escolha dos próximos pais e definida da seguinte forma: primeiro é feito o somatório das avaliações, ou seja, cada cromossomo possui um preço, que é quanto ele está levando consigo. Esses preços são todos somados e usado para realizar uma multiplicação de um valor entre zero e um. Feito isso, encontramos um x valor que será comparado com a soma do cromossomo na posição zero da lista. Se esse valor da multiplicação for menor ou igual ao valor da soma do cromossomo, o cromossomo é selecionado e assim encontramos o pai um. Caso o valor da multiplicação for maior, seguramos o valor do primeiro cromossomo e somamos com o valor do segundo cromossomo. Então faço a mesma verificação listada acima, ou seja, se o valor for menor ou igual da multiplicação selecionamos esse cromossomo que será, no caso, o pai dois.

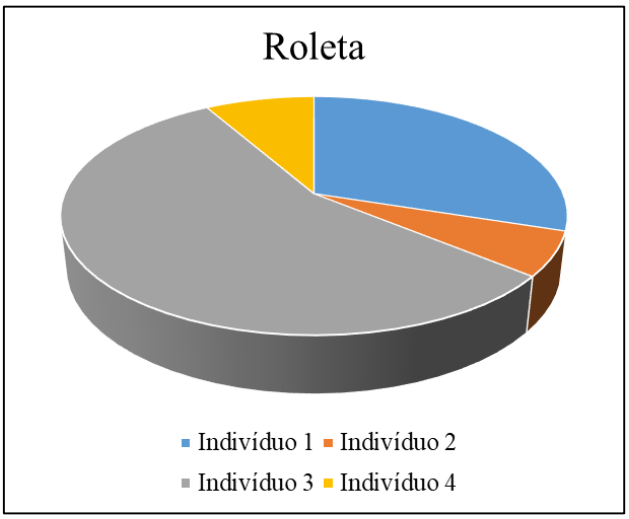

Figura 4: Processo da roleta em um AG.

Indivíduo 1	11.876,00
Indivíduo 2	2.369,00
Indivíduo 3	22.098,00
Indivíduo 4	3.289,00

Tabela 2: Dados do gráfico.

Um problema enfrentado no uso do método da roleta surge quando há grande diferença de aptidão entre os grupos de indivíduos. Isso faz com que certos cromossomos tenham uma chance excessivamente baixa de serem escolhidos para cruzamento. Uma alternativa é usar o método de classificação para realizar o sorteio dos elementos dentro de cada geração. Nessa metodologia, os grupos são classificados em ordem decrescente de acordo com sua adequação. Assim, o grupo com menor adequação recebe a classificação 1, o segundo recebe 2 e assim sucessivamente até a última classe. As fatias de cada grupo são proporcionais às suas classificações.

A forma básica de um algoritmo genético é a seguinte:

1 Início: Criar uma população com *n* indivíduos (adequados ao problema) aleatoriamente.
2 Aptidão: Avaliar a aptidão de cada cromossomo da população.
3 Nova geração: Até que se tenha uma nova população, executar os passos seguintes.
3.1 Seleção: Escolher dois cromossomos para serem pais de acordo com sua aptidão.
3.2 *Crossover:* Com a probabilidade, cruzar os pais para obter-se a geração seguinte. Caso não haja cruzamento, a descendência será uma cópia dos pais.
3.3 Mutação: Com a probabilidade, modificar os genes de um cromossomo para obter indivíduo alterado
3.4 Aceitação: Por os novos indivíduos na nova geração.
4 Substituição: Usar a nova geração na próxima execução do algoritmo.
5 Critério de parada: Se a condição final for atingida, pare e devolva a melhor solução encontrada.
Repetir Retornar ao passo 2.

Abaixo está o fluxograma de um AG, conforme a figura 5:

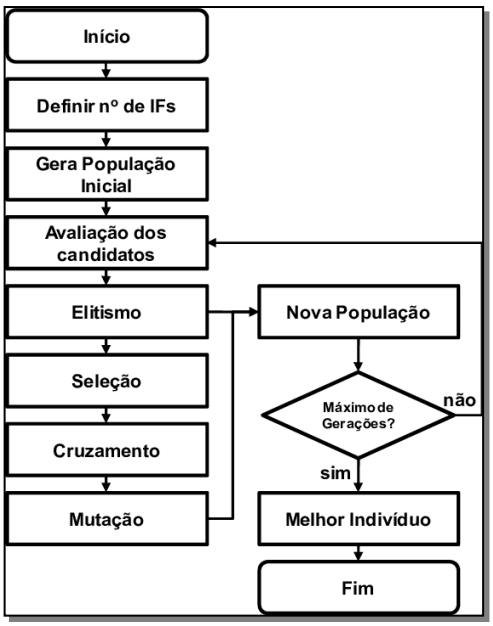

Figura 5: Fluxograma de execução de um AG.
Fonte:https://www.researchgate.net/figure/Fluxograma-do-algoritmo-genetico desenvolvido_fig1_268394114

3 METODOLOGIA

Há diversos tipos de problemas de transporte no âmbito da logística, todos de grande importância por influenciarem os custos envolvidos no fornecimento de um produto e/ou serviço. Por exemplo, qual o melhor tamanho de frota de veículos de entrega que deve ser mantido para garantir o abastecimento dos clientes de um certo fornecedor, ou qual a melhor rota a ser seguida para que o custo da entrega seja minimizado, ou como preencher uma carga com o máximo de produtos possíveis.

Ambos os exemplos citados anteriormente são descritos como de complexidade de tempo não polinomial e, por isso, não há ainda uma ferramenta eficiente que os resolva de forma exata. Além disso, incorre-se na maioria absoluta das vezes em situações em que as funções que representam cada situação não podem ser representadas analiticamente ou, quando o podem, não são lineares.

Nesse estudo, proponho apresentar melhorias no processo de transporte de carga, com intuito de apresentar uma solução para que o caminhão baú leve o máximo de itens respeitando o

tamanho em m³ interno e levando em consideração, o preço de cada ítem. Desta forma o projeto deverá obdecer às tres variáveis princípais que são: tamanho do ítem, preço e tamanho máximo que o baú comporta, que, nesse exemplo, será de 3m³.

Com o propósito de obter a otimização dos resultados foram criados cromossomos em que se baseiam-se em: genes totalmente aleatórios determinados entre zero e um, onde zero representa que o báu não tem o item, já o um representa que há o item no bau, logo depois que os genes foram gerados eles se juntam formando um cromossomo. O tamanho do cromossomo é determinado pela quantidade de itens que está implementado no código. Logo depois é feita uma avalição se o cromossomo está dentro do valor em m³ determinado. Se estiver, ele entra na lista de população, caso não, ele recebe o valor de um e é ignorado.

O projeto foi criado em Java e obedece as seguintes etapas: cria-se uma população de cromossomos como descrito acima, de onde é selecionado dois pais, realiza-se um cruzamento entre eles, gerando dois filhos, e com eles, é realizada uma mutação e inseridos na população.

O critério de parada utilizado foi a quantidade de geração, ou seja, cem gerações. O algoritmo irá rodar em um mecanismo de respetição *(for)* e quando atingir o limite máximo de gerações, irá interromper o seu funcionamento e mostrará a sua melhor solução genética e em qual geração obteve este resultado.

4 RESULTADOS

Para realização dos testes e comparação, foram utilizados resultados de pesquisa de outro resumo expandido (OLIVEIRA, Carlos Eduardo Ronconi de - *Aplicação de algoritmo genético para o problema da mochila*), para assim analisar os resultados obtidos com o projeto apresentado. Os parâmetros foram definidos seguindo o artigo de comparação: Uma população de tamanho 200, a chance de mutação de 0,05% e o mecanismo de parada definido para 100 gerações. Dessa forma todas as entradas de dados do AG são extamente iguais, inclusive os itens que serão disposto dentro do baú. A seguir será apresentado o desempenho do algorítmo e seus respectivos resultados.

ITENS	TAMANHO (M³)	PREÇO (R$)
Geladeira Dako	0.751	999.90
Iphone 6	0.000089	2911.12
TV 55'	0.400	4346.99
TV 50'	0.290	3999.90
TV 42'	0.200	2999.00
Notebook Dell	0.00350	2499.90
Ventilador Panasonic	0.496,	199.90
Microondas Electrolux	0.0424	308.66
Microondas LG	0.0544	429.90
Microondas Panasonic	0.0319	299.29
Geladeira Brastemp	0.635	849.00
Geladeira Consul	0.870	1199.89
Notebook Lenovo	0.498	1999.90
Notebook Asus	0.527	3999.00

Tabela 3: Dados usados para preencher as listas de m³ e valores dos produtos.

A tabela apresentada acima representa os itens que o caminhão baú poderá levar ou não. Nela contém os produtos com seus respectivos preços e tamanhos. Tais dados foram inseridos em dois algorítimos diferentes e a seguir podemos acompanhar os resultados dos processos a partir do número de gerações. Os algorítmos ordenam e trazem os melhores indivíduos, ou seja, aquele mais apto. As entradas de tamanho da população taxa de mutação sao de 200 e 0,05%

respectivamente. O codigo apresentado nesse projeto foi adequado para rodar até o limite de 100 gerações.

Geração: 0	Geração: 8	Geração: 15
Valor: 24385.600000000002	Valor: 24363.660000000003	Valor: 24793.560000000005
Espaço: 2.842989	Espaço: 2.7438890000000002	Espaço: 2.7982890000000005
Geração: 1	Geração: 9	Geração: 16
Valor: 23385.610000000004	Valor: 24064.370000000003	Valor: 24793.560000000005
Espaço: 2.468989	Espaço: 2.711989	Espaço: 2.7982890000000005
Geração: 2	Geração: 10	Geração: 17
Valor: 23563.660000000003	Valor: 24212.760000000002	Valor: 24343.370000000003
Espaço: 2.488889	Espaço: 2.6278889999999997	Espaço: 2.6503889999999997
Geração: 3	Geração: 11	Geração: 18
Valor: 21583.149999999998	Valor: 24494.270000000004	Valor: 24254.99
Espaço: 2.7663	Espaço: 2.766389	Espaço: 2.8204890000000002
Geração: 4	Geração: 12	Geração: 19
Valor: 23684.900000000005	Valor: 24793.560000000005	Valor: 24563.65
Espaço: 2.500889	Espaço: 2.7982890000000005	Espaço: 2.862889
Geração: 5	Geração: 13	Geração: 20
Valor: 22684.900000000005	Valor: 24363.660000000003	Valor: 24484.900000000005
Espaço: 2.753889	Espaço: 2.7438890000000002	Espaço: 2.7558890000000003
Geração: 6	Geração: 14	Geração: 21
Valor: 23694.270000000004	Valor: 24334.000000000004	Valor: 24993.550000000003
Espaço: 2.511389	Espaço: 2.639889	Espaço: 2.9172890000000002
Geração: 7		
Valor: 23694.270000000004		
Espaço: 2.511389		

Figura 6: Representação da primeira rodada chegando no melhor indivíduo na geração 21.

A seguir está disponível o gráfico da solução:

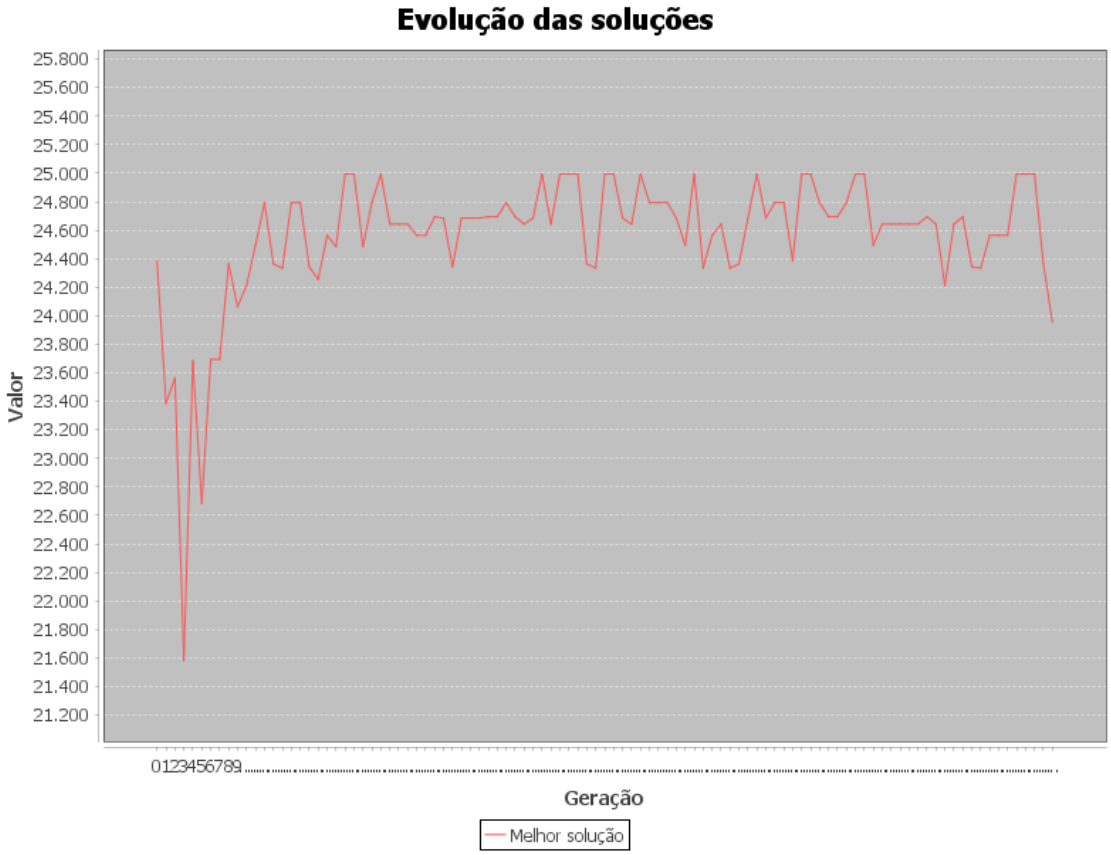

Figura 7: Representação gráfica da primeira rodada.

Geração: 0	**Geração: 3**	**Geração: 6**
Valor: 24334.000000000004	Valor: 23185.710000000003	Valor: 24264.36
Espaço: 2.639889	Espaço: 1.972989	Espaço: 2.8309889999999998
Geração: 1	**Geração: 4**	**Geração: 7**
Valor: 22184.99	Valor: 23642.660000000003	Valor: 24264.36
Espaço: 2.8713889999999997	Espaço: 2.935289	Espaço: 2.8309889999999998
Geração: 2	**Geração: 5**	**Geração: 8**
Valor: 23055.100000000002	Valor: 24385.600000000002	Valor: 24993.550000000003
Espaço: 1.9504890000000001	Espaço: 2.842989	Espaço: 2.9172890000000002

Figura 8: Representação da segunda rodada chegando no melhor indivíduo na geração 8.

A seguir está disponível o gráfico da solução:

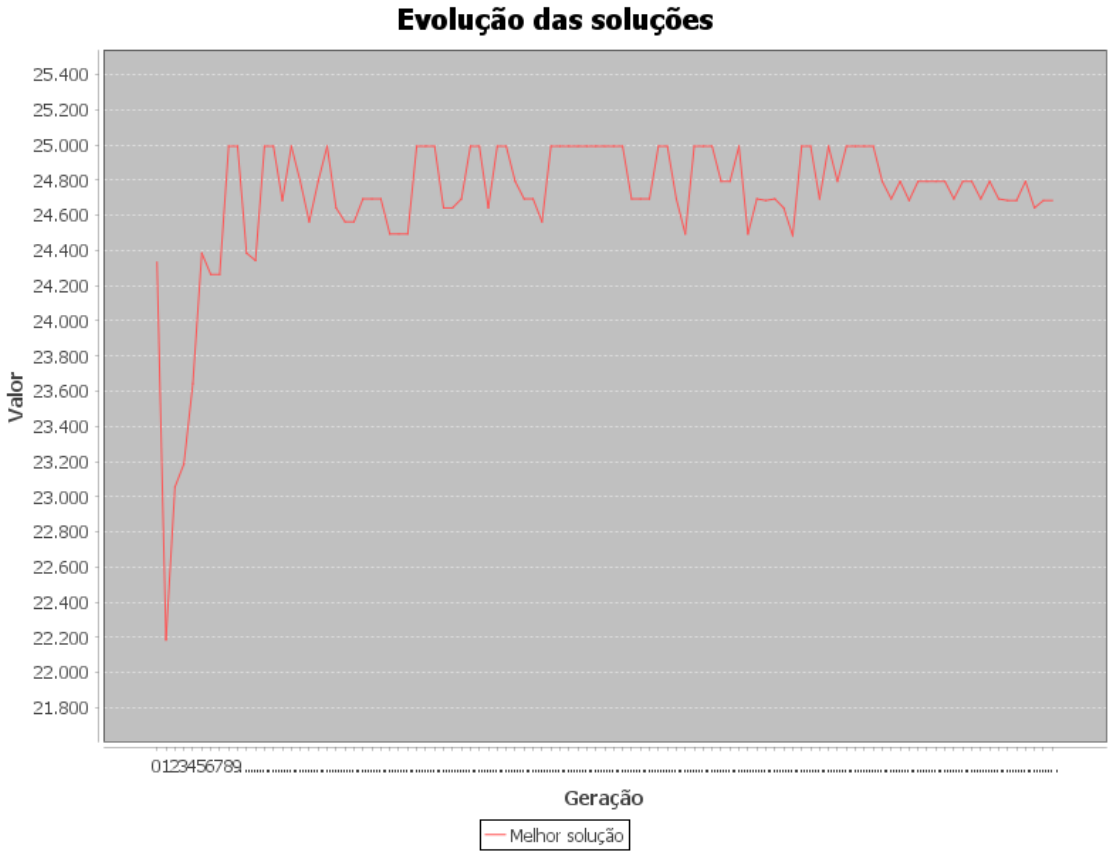

Figura 9: Representação gráfica da segunda rodada.

Geração: 0	**Geração: 4**
Valor: 24055.000000000004	Valor: 24185.610000000004
Espaço: 2.7014890000000005	Espaço: 2.7239890000000004
Geração: 1	**Geração: 5**
Valor: 24055.000000000004	Valor: 24642.660000000003
Espaço: 2.7014890000000005	Espaço: 2.682289
Geração: 2	**Geração: 6**
Valor: 24363.660000000003	Valor: 24993.550000000003
Espaço: 2.7438890000000002	Espaço: 2.9172890000000002
Geração: 3	
Valor: 24055.000000000004	
Espaço: 2.7014890000000005	

Figura 10: Representação da terceira rodada chegando no máximo fitness na geração 6.

A seguir está disponível o gráfico da solução.

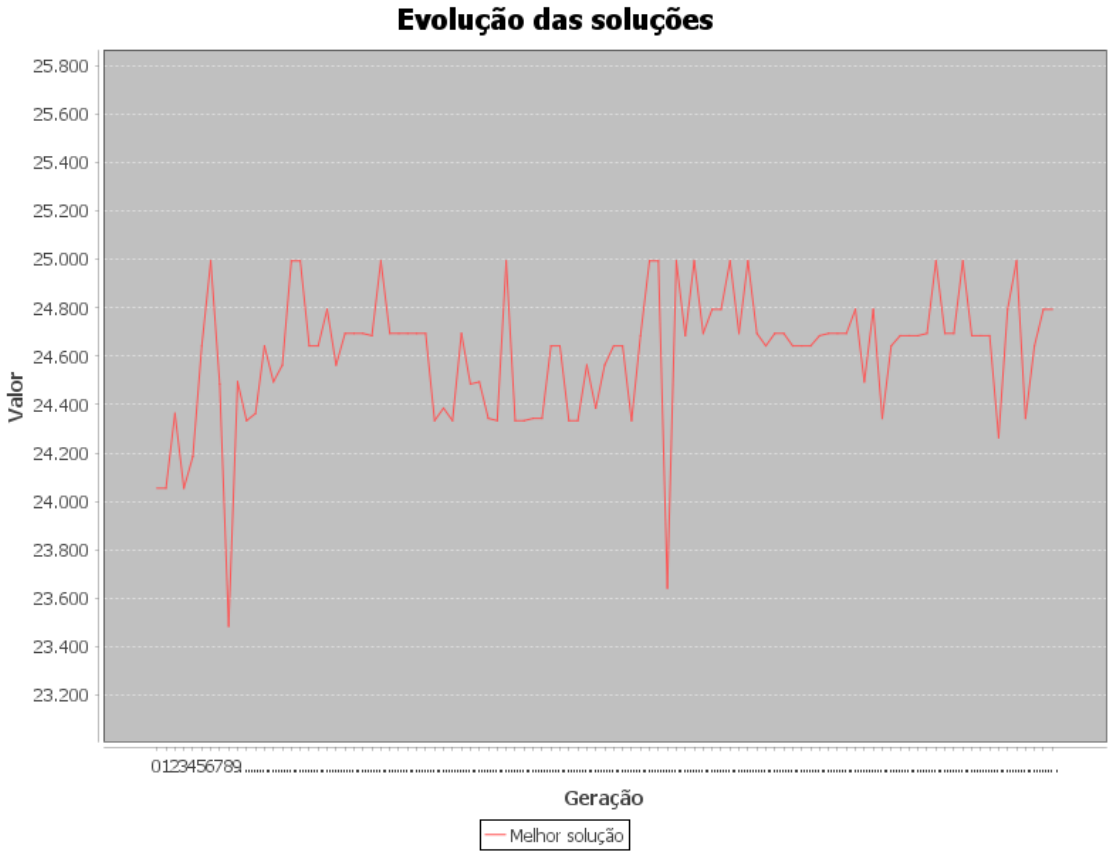

Figura 11: Representação gráfica da terceira rodada.

.

Geração: 0	**Geração: 4**	**Geração: 8**	**Geração: 12**
Valor: 20613.56	Valor: 24484.900000000005	Valor: 23993.560000000005	Valor: 24993.550000000003
Espaço: 2.964489	Espaço: 2.7558890000000003	Espaço: 2.543289	Espaço: 2.9172890000000002
Geração: 1	**Geração: 5**	**Geração: 9**	
Valor: 24642.660000000003	Valor: 24694.260000000002	Valor: 24064.370000000003	
Espaço: 2.682289	Espaço: 2.885389	Espaço: 2.711989	
Geração: 2	**Geração: 6**	**Geração: 10**	
Valor: 24694.260000000002	Valor: 23904.100000000002	Valor: 23684.900000000005	
Espaço: 2.885389	Espaço: 2.585489	Espaço: 2.500889	
Geração: 3	**Geração: 7**	**Geração: 11**	
Valor: 24694.260000000002	Valor: 24494.270000000004	Valor: 24793.560000000005	
Espaço: 2.885389	Espaço: 2.766389	Espaço: 2.7982890000000005	

Figura 12: Representação da quarta rodada chegando no máximo fitness na geração 12.

A seguir está disponível o gráfico da solução:

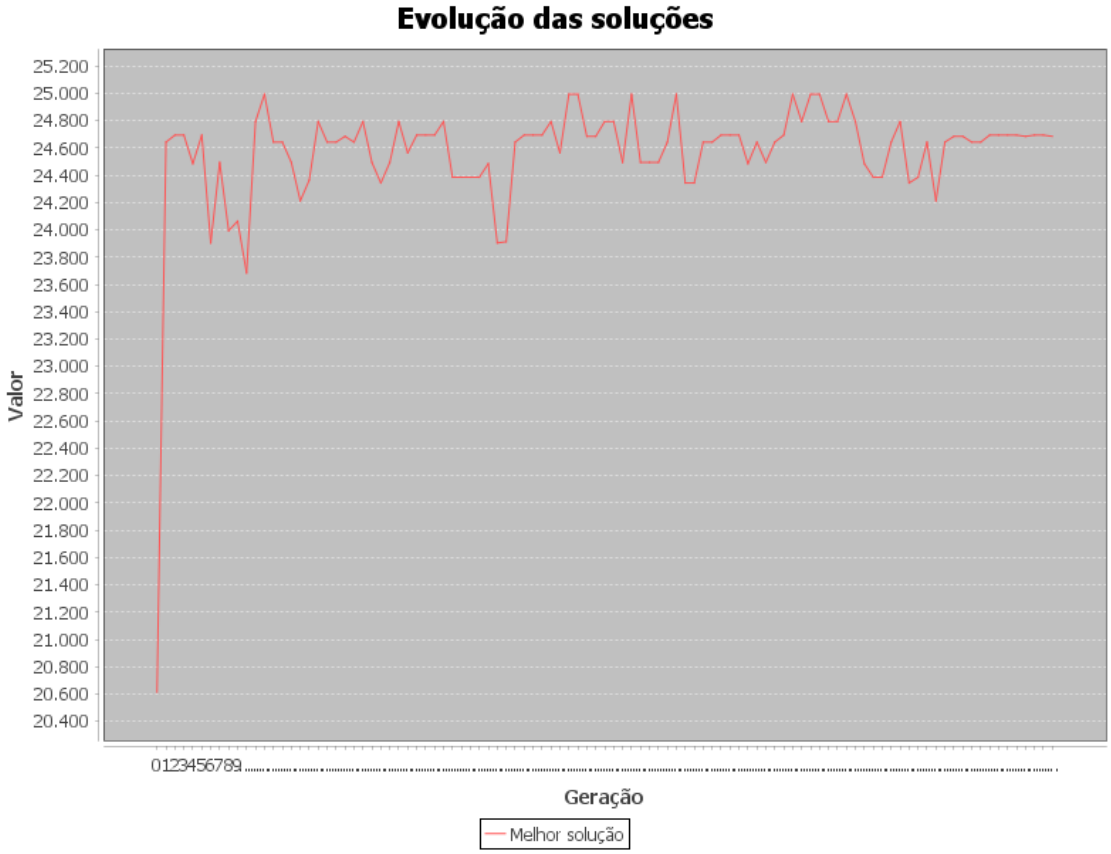

Figura 13: Representação gráfica da quarta rodada.

.

Geração: 0	**Geração: 5**	**Geração: 10**	**Geração: 15**
Valor: 24064.370000000003	Valor: 24034.710000000003	Valor: 24642.660000000003	Valor: 24993.550000000003
Espaço: 2.711989	Espaço: 2.607989	Espaço: 2.682289	Espaço: 2.9172890000000002
Geração: 1	**Geração: 6**	**Geração: 11**	
Valor: 23064.47	Valor: 24264.36	Valor: 24212.760000000002	
Espaço: 1.960989	Espaço: 2.8309889999999998	Espaço: 2.6278889999999997	
Geração: 2	**Geração: 7**	**Geração: 12**	
Valor: 24034.710000000003	Valor: 24363.660000000003	Valor: 24343.370000000003	
Espaço: 2.607989	Espaço: 2.7438890000000002	Espaço: 2.6503889999999997	
Geração: 3	**Geração: 8**	**Geração: 13**	
Valor: 24343.370000000003	Valor: 24385.600000000002	Valor: 24494.270000000004	
Espaço: 2.6503889999999997	Espaço: 2.842989	Espaço: 2.766389	
Geração: 4	**Geração: 9**	**Geração: 14**	
Valor: 24684.890000000003	Valor: 24563.65	Valor: 24343.370000000003	
Espaço: 2.874889	Espaço: 2.862889	Espaço: 2.6503889999999997	

Figura 14: Representação da quinta rodada chegando no máximo fitness na geração 15.

A seguir está disponível o gráfico da solução:

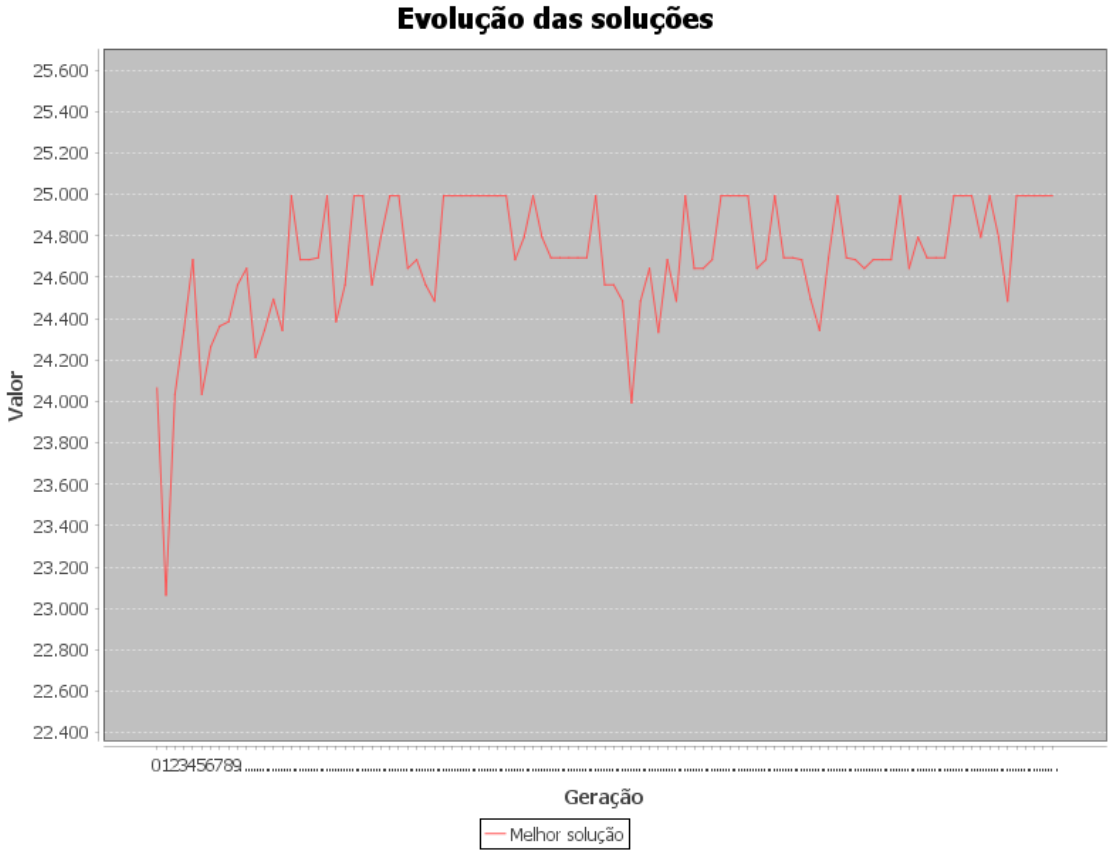

Figura 15: Representação gráfica da quinta rodada.

3 CONSIDERAÇÕES FINAIS

Os algoritmos genéticos buscam soluções de forma análoga ao processo de evolução dos seres vivos e podem ser utilizados para resolver um grande número de problemas de otimização complexos. Eles são apropriados quando há muitas variáveis envolvidas e um espaço de soluções de dimensão elevada. Embora sejam mais lentos, o tempo de execução torna-se um problema cada vez mais secundário devido à rápida evolução dos sistemas computacionais. Os problemas de transporte representam uma faixa de questões muito importante nos dias atuais. Procurar a melhor forma de distribuir recursos de transporte de bens, serviços e pessoas é uma parte primordial no planejamento das organizações, seja em qual nível for.

Uma vez que a complexidade de tais problemas consome uma grande quantidade de recursos computacionais e, na maioria dos casos, os mesmos não podem ser resolvidos usando ferramentas determinísticas que forneçam resultados exatos, a busca por outras formas de resolução é de suma importância. Como tais, os *softwares* que implementam a computação evolutiva, mais especificamente, os algoritmos genéticos, possuem grande vantagem nessa área, uma vez que trabalham com avaliações de soluções ótimas através de procedimentos de análise baseados em dados estatísticos e probabilidades. Partindo de um conjunto conhecido onde situam-se as melhores respostas, podem selecionar a que melhor se adapta ao contexto da questão.

Para um trabalho futuro, é importante destacar uma maior variabilidade nas entradas de dados e nos modos de seleções cromossonicas. Dessa forma, podemos encontrar uma solução ótima em menos tempo e assim agilizar o processo de preenchimento do caminhão baú. Tudo isso atrelado a experiência do carregador e no desempenho do software.

4 REFERENCIAS

CARVALHO, Marco Antonio Moreira de. SANTOS, André Gustavo dos. Algoritmo genético aplicado à seleção de colunas no problema de alocação de tripulações. Disponível em <http://www.bibl.ita.br/xiiencita/Comp-01.pdf>.

H., Alexandre. Java Algoritmo Genético – Problema da Mochila (Knapsack Problem). Tecnoblog, 2016. Disponível em: < http://alexandrehenrique.esy.es/programacao/algoritmo-genetico-knapsack-problem/>.

MARTINS, G. A. Estatística geral e aplicada. 3. Ed. São Paulo: Atlas, 2005.

MITCHELL, T.M. Machine Learning. McGraw-Hill Science Engineering, 1997.

https://www.pngwing.com/pt/free-png-dzoxi. Acessado em 16 de junho de 2021.

https://docs.ufpr.br/~volmir/PO_II_12_TSP.pdf. Acessado em 15 de junho de 2021.

https://repositorio.ufms.br/bitstream/123456789/2970/1/Bianca%20de%20Almeida%20Dantas.pdf. Acessado em 9 de maio de 2021.

https://www.devmedia.com.br/algoritmos-geneticos-em-java-java-magazine-82/17702. Acessado em 4 de maio de 2021.

https://www.docsity.com/pt/artigo-algoritmos-geneticos-e-problemas-de-transporte-2/4741655/. Acessado em 25 de abril de 2021.