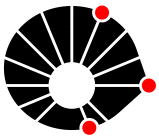

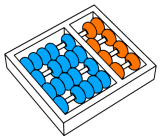

Otimização na movimentação de materiais em armazéns logísticos

João Vitor Baptista Moreira *Luiz Fernando Bittencourt*
Allan Mariano de Souza

Relatório Técnico - IC-PFG-23-21
Projeto Final de Graduação
2023 - Dezembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

The contents of this report are the sole responsibility of the authors.
O conteúdo deste relatório é de única responsabilidade dos autores.

Otimização na movimentação de materiais em armazéns logísticos

João Vitor Baptista Moreira [1], Luiz Fernando Bittencourt [2], Allan Mariano de Souza [3]

[1] Instituto de Computação Universidade Estadual de Campinas (UNICAMP), Caixa Postal 6176
13083-970 Campinas-SP, Brasil

j237833@dac.unicamp.br

[2] Instituto de Computação Universidade Estadual de Campinas (UNICAMP), Caixa Postal 6176
13083-970 Campinas-SP, Brasil

bit@ic.unicamp.br

[3] Instituto de Computação Universidade Estadual de Campinas (UNICAMP), Caixa Postal 6176
13083-970 Campinas-SP, Brasil

allanms@unicamp.br

Resumo

O número de pedidos de encomendas realizadas no Brasil por meio de e-commerces têm crescido substancialmente na última década. Nos bastidores, para garantir que um produto seja entregue de maneira rápida e confiável para o cliente, o setor de logística realiza diversas estratégias para que cada etapa da distribuição seja realizada da forma mais eficiente possível. Uma dessas estratégias é a utilização de veículos autônomos responsáveis por otimizar a movimentação de materiais dentro de armazéns. Com isso, surge espaço para diversas análises e propostas de modelagem matemática e de arquitetura do problema, a fim de otimizar a tarefa. Neste trabalho, foram apresentados modelos em grafos de armazéns logísticos com diferentes características e peculiaridades, tais como número de veículos de transporte, número de caixas, pontos de entrega, tamanho da fábrica, distância entre os pontos, massa física de uma encomenda e autonomia de bateria dos veículos. Além disso, foram exploradas diferentes técnicas de otimização para uma delegação de tarefas de movimentação de materiais dentro do armazém, levando em consideração parâmetros como complexidade computacional e disponibilidade de recursos. Ainda foi realizado um estudo sobre a utilização de sistemas distribuídos para entender as estratégias mais adequadas para cada tipo de organização.

Palavras chave: *otimização, grafos, movimentação de materiais, sistemas distribuídos*

1 INTRODUÇÃO

Atualmente, no mundo globalizado, existe uma busca incessante por bens de consumo. De acordo com Baudrillard, em sua obra “Sociedade de Consumo” [1], a sociedade atual admira sempre o novo, favorecendo assim o surgimento de novos objetos, serviços, bens e produtos.

Além disso, Zygmunt Bauman[2], analisando a sociedade de consumo da era da modernidade líquida, observou que ela é marcada pela instabilidade, liquidez e imediatismo, sendo que o motivo da pressa é a "necessidade de descartar e substituir".

Nesse contexto, surgiu na década de 1930, por Bernard London, o conceito de obsolescência programada[3]. A ideia de London era que os produtos recém lançados possuíssem uma espécie de prazo de validade, em que se tornariam obsoletos com o passar do tempo e precisariam ser constantemente substituídos, mantendo os padrões de consumo na sociedade.

Com o crescimento da tecnologia e do número de consumidores com acesso à internet no mundo globalizado, muitos negócios passaram a ser conduzidos de forma online. Chiavenato[4], apud Nascimento[5], ressalta que, para produzir produtos e serviços de qualidade e preços competitivos, a empresa deve estar em total sintonia com as informações do mercado, conhecer as necessidades dos clientes, monitorar os concorrentes, rever constantemente os processos do negócio e atualizar as tecnologias.

Com isso, os e-commerces começaram a ganhar força no mercado. Em um recorte atual, o faturamento dos e-commerces no Brasil têm crescido consideravelmente na última década[6]. A Figura 1 mostra um gráfico dos valores de receita em comércios virtuais documentados de 2011 a 2020.

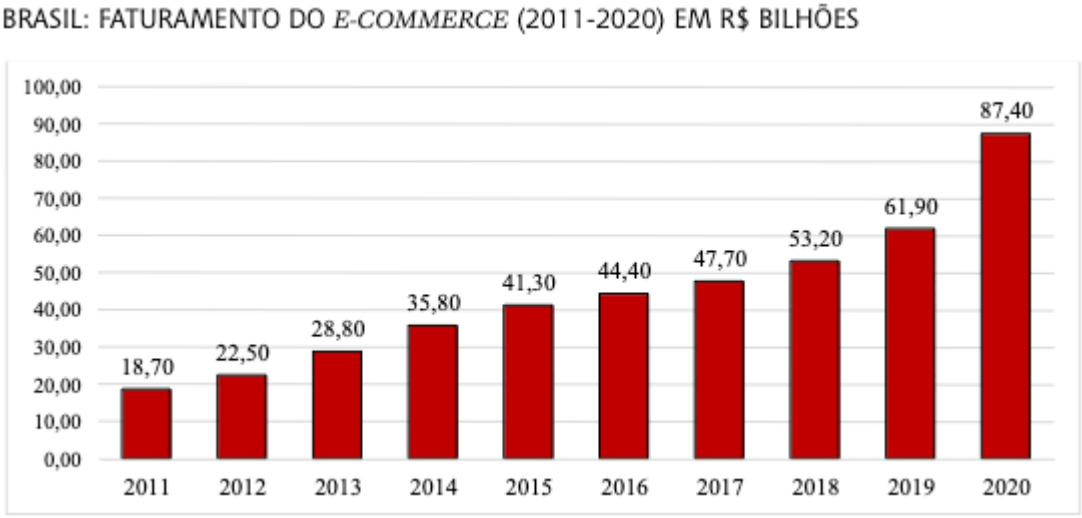

Figura 1: Faturamento do e-commerce no Brasil na última década

Esse faturamento engloba diferentes tipos de produtos, tanto físicos quanto digitais. Com relação a produtos físicos, a Figura 2 [6] mostra o crescimento do número de encomendas realizadas no Brasil na última década.

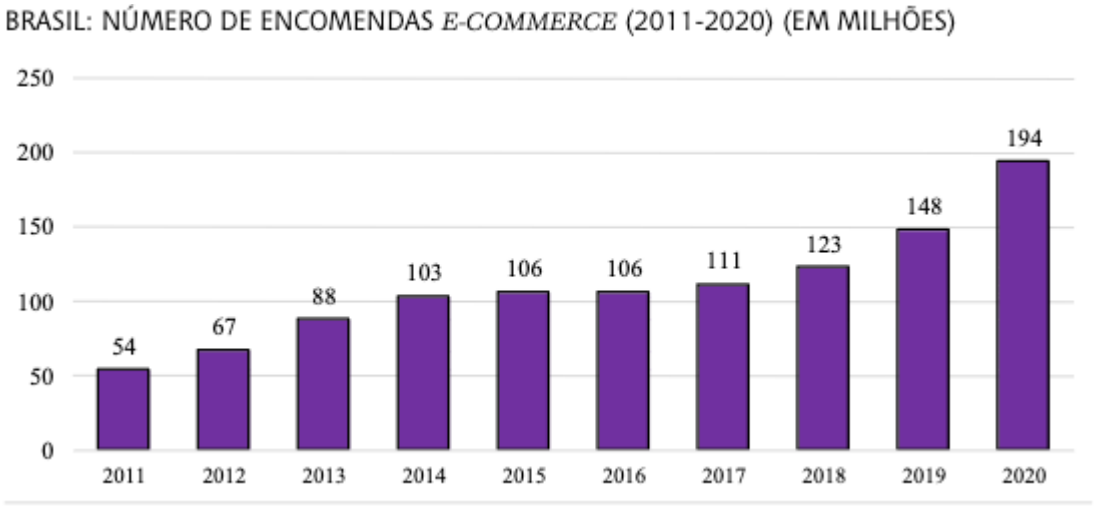

Figura 2: Número de encomendas no e-commerce na última década

Por se tratar de um mercado aquecido e em expansão, surge a necessidade das empresas de serem eficientes durante esse processo. Em alguns casos, como a Amazon, houve a instalação de mais de 100 mil robôs dentro de seus armazéns para que o transporte de carga fosse realizado dentro das fábricas[7].

Segundo Nascimento[5], além da redução de custos, existem outros objetivos não menos importantes na movimentação de materiais, como: eficiência e segurança na movimentação, transporte dos materiais em tempo útil e com a quantidade exata para o local desejado, e armazenamento de materiais otimizando a capacidade espacial fornecida pela empresa.

Nesse contexto, surge um questionamento com relação a essa eficiência. Como ter o controle de um número tão grande de robôs e como garantir que eles estão sendo utilizados da maneira mais otimizada possível?

Para tentar resolver esse desafio, primeiramente é necessário entender as variáveis que são apresentadas no contexto e como modelar isso de maneira a entender matematicamente o problema.

Modelar um cenário real não padronizado pode ser uma tarefa complicada, uma vez que as variáveis podem se modificar em cada tipo de armazém e na gestão interna de cada um. Por isso, neste trabalho, será realizada uma análise de diferentes níveis a fim de otimizar a movimentação de materiais em um armazém.

Na metodologia é realizado um estudo de modelagem do cenário, enquanto nos resultados e discussão, há uma análise da complexidade do problema, dos recursos físicos que estão disponíveis para serem utilizados, e as estratégias mais adequadas para a resolução considerando diferentes critérios.

2 OBJETIVO

O objetivo deste trabalho é modelar matematicamente o problema de movimentação de materiais em armazéns logísticos, testar algoritmos de otimização combinatória para otimizar o processo e explorar diferentes arquiteturas de comunicação e sistemas distribuídos que podem ser utilizados durante a automação das tarefas, bem como discutir as vantagens e desvantagens de cada método.

3 CONCEITOS E FERRAMENTAS

O armazém foi modelado como um grafo, uma estrutura de dados capaz de representar pontos de interesse (vértices) e as ligações entre eles (arestas), que podem possuir pesos, como distância[8].

O problema estudado é uma variação do *assignment problem*, [9] que busca encontrar a melhor atribuição de tarefas entre uma máquina e um trabalho, tal que uma máquina é responsável por um único trabalho, e um trabalho deve ser realizado por apenas uma máquina.

Na análise, foram testadas as performances de três diferentes algoritmos. O primeiro é uma delegação aleatória realizada iterativamente. Já o segundo é um algoritmo guloso, que busca sempre a melhor solução local, sem se preocupar com o global[8]. O terceiro é o algoritmo húngaro, que encontra a melhor solução possível do problema (solução ótima), mas possui uma complexidade de tempo que pode ser um empecilho em certas aplicações[9].

Finalmente, foi realizado um estudo de sistemas distribuídos, comparando computação de borda, nuvem e névoa. A borda consiste em um servidor próximo ao cliente, tendo assim uma menor latência, mas demanda uma maior manutenção e administração da organização, o que pode gerar maiores custos financeiros. Já a nuvem está mais distante, o que aumenta a latência, mas pode trazer benefícios em poder computacional e menor custo. Finalmente, a névoa é uma estratégia que consegue ser flexível em uma aplicação, definindo se uma determinada tarefa deverá ser realizada na borda ou na nuvem[10]. Esse estudo foi realizado a fim de entender as melhores estratégias para cada um dos diferentes cenários realizados no problema de movimentação de materiais em armazéns logísticos.

Para realizar os experimentos, foi utilizada a biblioteca *networkx*, que permite gerar representações visuais de redes complexas, o que facilita a análise e o estudo do problema. [11].

O código para os experimentos está disponível no Anexo 1.

4 METODOLOGIA

4.1 Modelagem do problema

Inicialmente, é importante modelar como um robô se movimenta dentro de um armazém. Para isso, podemos enxergar o mapa da fábrica como um grafo, onde os vértices são pontos de referência relevantes, como esquinas, depósitos, estações, entre outros, e as arestas são baseadas na distância real entre dois desses pontos. Com isso, o armazém pode ser representado como uma série de pontos de interesse com uma lista de adjacências

definido os vértices vizinhos. A Figura 3 mostra um exemplo de um modelo de uma fábrica com 10 nós, com o local de entrega destacado em amarelo.

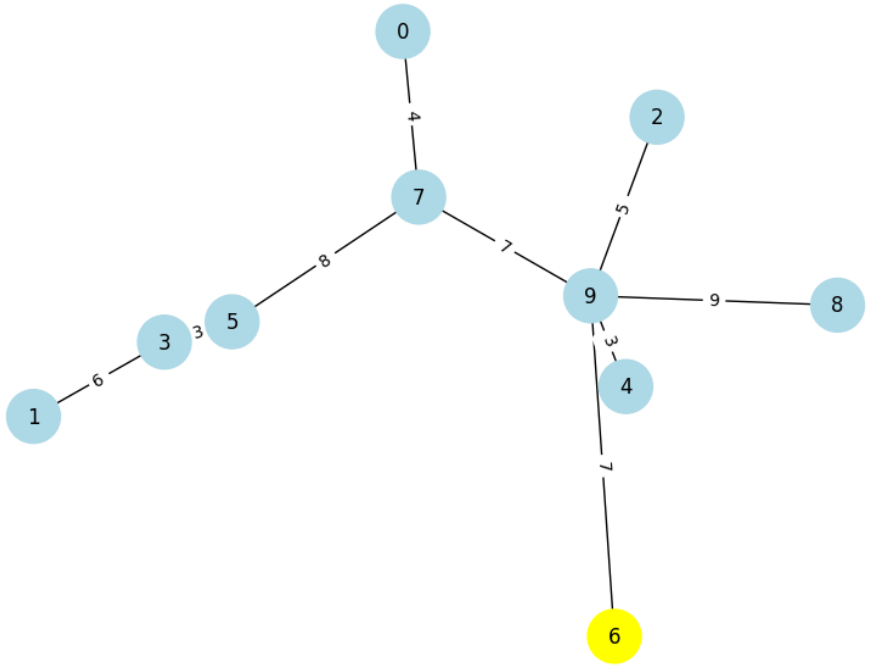

Figura 3: Modelo de um armazém de 10 nós com um ponto de entrega

Como as arestas correspondem a distâncias, é possível calcular o caminho mínimo entre um robô para qualquer ponto por meio do algoritmo de Dijkstra. O algoritmo de Dijkstra é utilizado para calcular o menor caminho entre dois pontos em grafos cujas arestas são não-negativas [8]. Nesse caso, é possível utilizar esse algoritmo para encontrar a menor distância que um robô precisa percorrer para se locomover até um determinado ponto do armazém, bem como o caminho que deverá realizar.

A Figura 4 mostra um modelo do armazém com um robô (em verde) e um ponto de entrega (em amarelo).

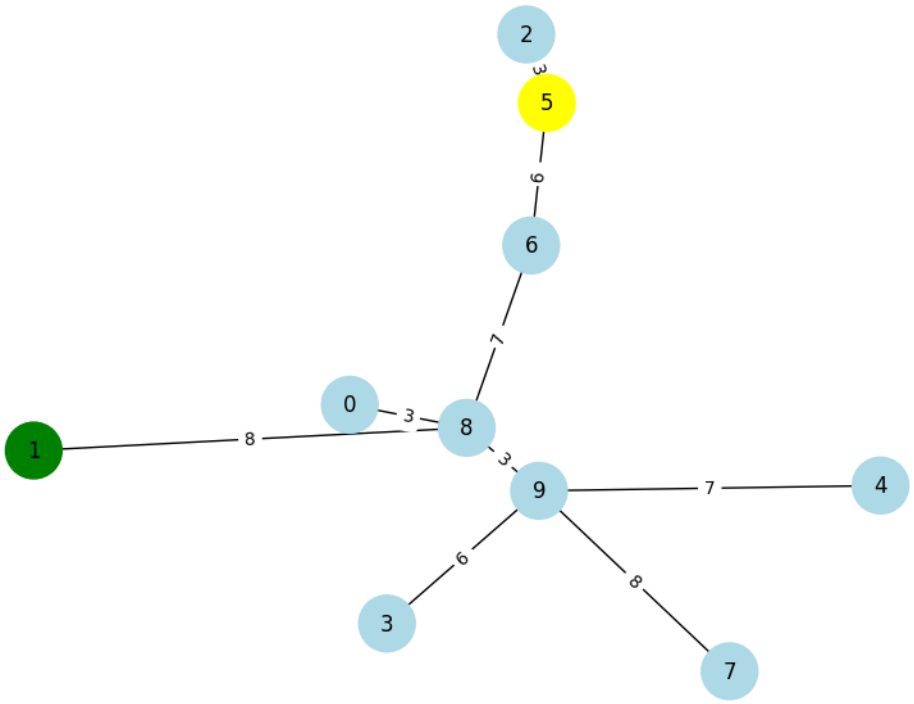

Figura 4: Modelo de um armazém com um robô e um ponto de entrega

O caminho e a distância entre os dois pontos a partir do algoritmo de Dijkstra é dado por:

$$Menor\ caminho\ entre\ os\ pontos\ 1\ e\ 5\ pelo\ algoritmo\ de\ Dijkstra:$$

$$1\ -> 8\ -> 6\ -> 5$$

$$Pesos = (C_{1,8}\ ,\ C_{8,6}\ ,\ C_{6,5})$$

$$Custo\ = \sum Pesos$$

$$Custo\ total: 21$$

No entanto, apenas com isso o problema ainda está incompleto. Um robô deve buscar uma caixa em algum local e entregar em outro. Para resolver isso, é possível fazer duas vezes o algoritmo de Dijkstra, de modo que o caminho seja o menor possível. Assim, o robô iria até a caixa pelo menor caminho e depois da caixa até a entrega pelo menor

caminho. Um modelo de armazém de 10 nós com uma caixa (vermelha), um robô (verde) e um ponto de entrega (amarela) é mostrado na Figura 5.

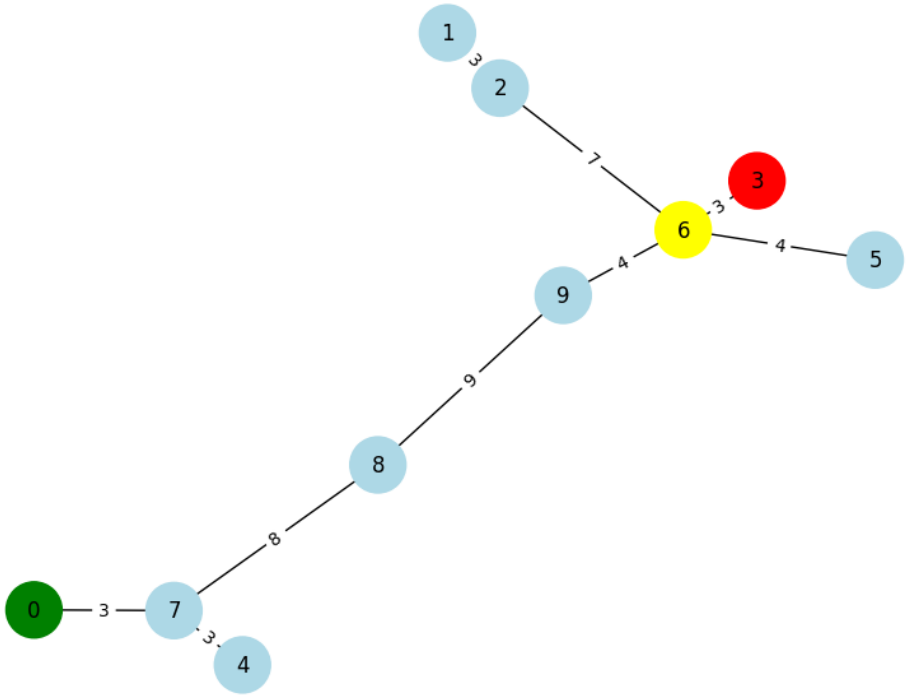

Figura 5: Modelo de um armazém com um robô, uma caixa e um ponto de entrega

Nesse caso, o robô precisa ir até a caixa pelo melhor caminho e depois levar até a entrega. Assim, é possível concatenar a solução de duas execuções do algoritmo de Dijkstra entre os pontos desejados. A solução do problema para o exemplo da Figura 5 é dada por:

$$Menor\ caminho\ entre\ os\ pontos\ 0\ e\ 3\ pelo\ algoritmo\ de\ Dijkstra$$

$$0\ -> 7\ -> 8\ -> 9\ -> 6\ -> 3$$

$$Pesos = (C_{0,7}\ ,\ C_{7,8}\ ,\ C_{8,9}\ ,\ C_{9,6}\ ,\ C_{6,3}\ )$$

$$Custo\ = \sum Pesos$$

$$custo: 27$$

$$Menor\ caminho\ entre\ os\ pontos\ 3\ e\ 6\ pelo\ algoritmo\ de\ Dijkstra$$

$$3\ \ -> \ \ 6$$

$$Pesos = (C_{3,6})$$

$$Custo\ \ = \sum Pesos$$

$$custo: 3$$

$$Custo\ total: 27\ \ +\ \ 3\ \ =\ \ 30$$

Com isso, é possível modelar a tarefa de um robô buscando uma caixa e a deixando no ponto de entrega. Essa análise corresponde ao caso base de um problema mais complexo, relacionado à otimização do custo de entrega de N robôs buscando N caixas dentro de um armazém. A Figura 6 mostra um exemplo de um armazém com 30 nós, 8 robôs e 8 caixas com um ponto de entrega.

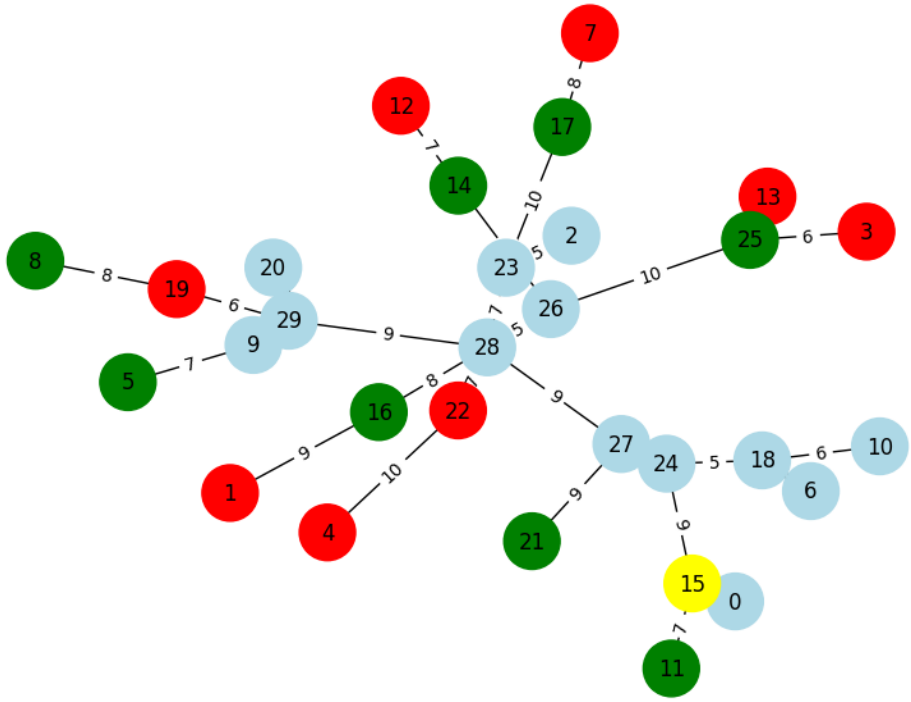

Figura 6: Modelo de armazém com 30 nós, 8 robôs e 8 caixas a serem transportadas

4.2 Construção do problema de atribuição

Nessa etapa, é preciso encontrar uma solução otimizada para escolher qual robô deverá pegar qual caixa e levar até o ponto de entrega, de maneira em que a soma total dos tempos seja a menor possível.

Uma maneira de solucionar esse problema é mapeando as distâncias que cada robô precisa percorrer para buscar cada uma das caixas e encontrar a melhor atribuição de tarefas para cada um, o que seria uma transposição do desafio para um *assignment problem*[9]. Dessa forma, é possível analisar três sub-problemas que compõem a solução final.

1. Calcular a distância entre o ponto de entrega e cada caixa
2. Calcular a distância entre cada caixa e cada robô
3. Encontrar a melhor atribuição de tarefas para cada robô

O sub-problema 1 pode ser resolvido pelo algoritmo de Dijkstra. Com ele, é possível guardar em um vetor a distância de cada caixa até o ponto de entrega. Assim, ao atribuir uma caixa a um robô, o custo dessa operação é somado ao custo de deixar a caixa no ponto de entrega. Além disso, o caminho mínimo de adjacências também pode ser guardado em um vetor para que o percurso seja mapeado posteriormente.

Já o subproblema 2 consiste em rodar o algoritmo de Dijkstra em cada nó correspondente a um robô, de modo a encontrar a menor distância entre ele e qualquer caixa presente no armazém. Com isso, é possível somar o custo encontrado em uma atribuição a uma caixa com o custo correspondente à entrega dela, que foi armazenado no sub-problema 1.

Um pseudocódigo para os subproblemas 1 e 2 pode ser visto a seguir:

$matrixCosts = []$

$deliveryCosts = Dijkstra(delivery)$

$for\ robot\ in\ robots$:

    $robotCosts = []$

    $for\ box\ in\ boxes$:

        $robotBoxCost = Dijkstra(robot, box)$

        $robotBoxDeliveryCost = robotBoxCost + deliveryCosts[box]$

        $robotCosts.push(robotBoxDeliveryCost)$

    $endfor$

    $matrixCosts.push(robotCosts)$

$endfor$

Ao final dessa etapa, é gerada uma matriz com o custo que cada robô possui para entregar cada caixa. A Tabela 1 mostra o custo de cada robô para pegar e entregar cada caixa no exemplo apresentado na Figura 6.

	5	8	11	14	16	17	21	25
1	74	78	83	69	47	72	73	70
3	82	86	91	67	71	80	81	48
4	74	78	83	69	63	72	73	70
7	90	94	99	85	79	54	89	86
12	82	86	91	49	71	80	81	68
13	78	82	87	63	67	76	77	44
19	52	44	79	65	59	68	69	66
22	54	58	63	49	43	52	53	50

Tabela 1: Custo de entrega de uma caixa por um robô

Uma vez que a tabela de custos foi gerada, o terceiro passo é encontrar a melhor delegação de tarefas possível de modo a minimizar o custo total. Esse problema de

delegação pode ser resolvido pelo método húngaro[9]. Esse algoritmo analisa diferentes permutações da matriz de custo e encontra a melhor combinação de atribuições para que o custo total seja minimizado. O algoritmo ainda respeita as restrições do problema, de modo que um robô seja responsável por apenas uma caixa e cada caixa seja transportada por apenas um robô.

5 RESULTADOS E DISCUSSÃO

5.1 Análise de cenários e variáveis

A solução encontrada pelo algoritmo húngaro no problema da Figura 6, que encontra o menor custo possível para o problema de atribuição gerado na Tabela 1, é apresentada na Tabela 2.

	5	8	11	14	16	17	21	25
1	74	78	83	69	47	72	73	70
3	82	86	91	67	71	80	81	48
4	74	78	83	69	63	72	73	70
7	90	94	99	85	79	54	89	86
12	82	86	91	49	71	80	81	68
13	78	82	87	63	67	76	77	44
19	52	44	79	65	59	68	69	66
22	54	58	63	49	43	52	53	50
Custo Total = 456								

Tabela 2: Melhor atribuição de modo a gerar custo mínimo pelo método húngaro

Com isso, é possível escolher cada tarefa de maneira otimizada para que cada robô faça a movimentação de um material dentro do armazém com o menor custo total possível.

Apesar desse modelo se mostrar interessante para a delegação das tarefas, o problema não para por aí. Por se tratar de um cenário não ideal, é preciso considerar que existem outras variáveis que interferem no desempenho dos carrinhos.

Por esse motivo, nas simulações realizadas, foram adicionados dois atributos que modelam o comportamento dos robôs e das caixas, influenciando no custo das arestas. A primeira característica é que cada robô possui uma autonomia máxima de bateria quando está se movimentando sem nenhuma caixa. Dessa maneira, ele não é capaz de percorrer um caminho até uma caixa caso o custo desse percurso seja maior do que sua autonomia.

Já o segundo atributo é intrínseco a cada caixa. Cada uma delas possui massa, que por sua vez influencia na autonomia de um robô que a carrega. Em outras palavras, robôs que carregam caixas mais pesadas possuem uma tendência maior a realizar mais esforço nos motores e, consequentemente, ter uma autonomia de bateria reduzida.

Para completar o modelo de simulação, os robôs podem recarregar suas baterias no ponto de entrega, de modo que, uma vez cheias, podem se locomover livremente para realizar sua tarefa.

Com isso, as arestas não possuem mais um peso fixo, mas são dependentes de uma função que envolve a autonomia de bateria do robô, o peso da caixa e a distância total que deve ser percorrida durante a movimentação do material.

Finalmente, a simulação foi modelada de modo em que, caso um robô não tenha autonomia suficiente para pegar uma caixa e entregá-la no ponto de entrega, ele precisa necessariamente ir diretamente ao ponto para recarregar antes de fazer qualquer movimentação de carga.

Um exemplo da tabela de custos da simulação da Figura 6 é mostrado na Tabela 3.

	5	8	11	14	16	17	21	25
1	116	78	83	69	47	72	97	112
3	124	86	91	67	71	80	105	120
4	116	78	83	69	63	72	97	112
7	132	94	99	85	79	54	113	128
12	124	86	91	49	71	80	105	120
13	120	82	87	63	67	76	101	116
19	52	44	79	65	59	68	93	108
22	54	58	63	49	43	52	53	92
Custo Total = 540								

Tabela 3: Melhor atribuição de custo mínimo com arestas de peso variável

Note que, após adicionar a restrição, houve mudanças nos custos de alguns robôs para movimentar caixas. Isso ocorreu pois aquele veículo não possuía autonomia suficiente para transportar a carga, e precisou ir para o ponto de entrega recarregar antes de buscar de fato a caixa. Esse caminho extra gera um aumento do custo total da tarefa.

5.2 Análise de complexidade

Por se tratar de um problema que exige decisões constantes dos caminhos realizados dentro do armazém, é interessante realizar uma breve análise da complexidade dos algoritmos utilizados, bem como pensar em alternativas e arquiteturas adequadas para uma maior otimização dos recursos no sistema.

O algoritmo de Dijkstra possui uma complexidade de tempo de O(V+E), em que V corresponde ao número de nós e E o número de arestas, que no pior caso pode ser descrito como $O(V^2)$[8]. No problema abordado, ele é executado uma vez para medir a distância entre as caixas e o ponto de entrega, e em seguida *r* vezes, sendo *r* o número de robôs, para medir o custo entre cada robô e cada caixa. Assim, no pior caso, podemos considerar que *r* corresponde a metade dos nós, V/2. Com isso, a complexidade total de se executar esse algoritmo é de $O(V^3)$.

Já com relação ao algoritmo húngaro, a complexidade para gerar a matriz de permutação e encontrar a melhor atribuição de tarefas para minimização de custo, também é de $O(V^3)$[9]. Com isso, a complexidade total de calcular os custos e tomar as decisões de atribuição correspondem a $O(V^3)$.

Essa complexidade pode se tornar bastante custosa à medida que o número de nós do armazém aumenta. Por esse motivo, é preciso pensar em estratégias e arquiteturas para minimizar o processamento e manter o funcionamento da resolução do problema.

5.3 Análise de recursos

Uma maneira de fazer isso é utilizando sistemas distribuídos para dividir a responsabilidade de processamento. Uma proposta para esse problema é distribuir o processamento entre os robôs, de modo a aproveitar o processamento já existente dentro deles para realizar algumas operações e dividir o custo computacional entre eles. A Figura 7 mostra um esquema da distribuição do processamento.

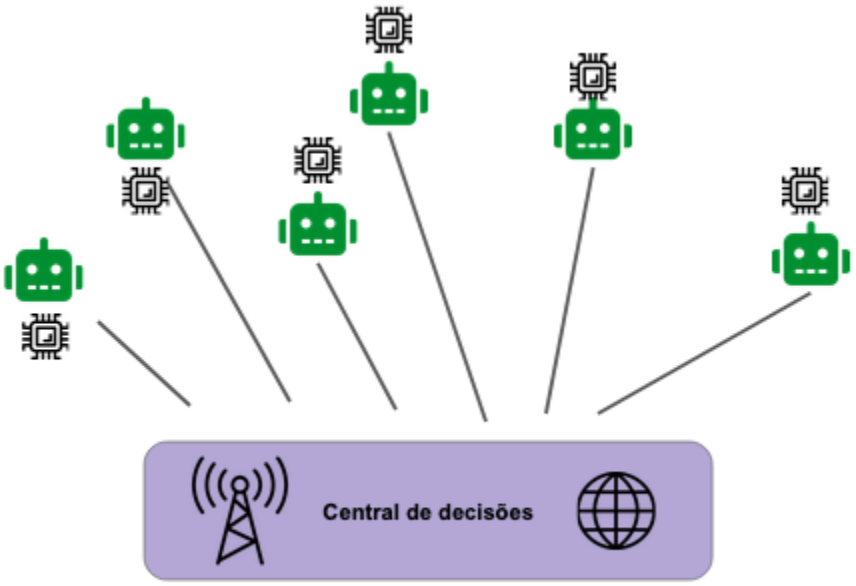

Figura 7: Distribuição de processamento entre robôs e uma central de decisões

Dessa maneira, o custo computacional será dividido entre os robôs e não será de total responsabilidade da central de decisões. Assim, o cálculo do algoritmo de Dijkstra pode ser realizado por cada robô, calculando o custo total da entrega de cada uma das caixas. Esse vetor de custos seria então retornado à central, que por sua vez seria responsável por realizar a atribuição. Com isso, o custo computacional de montar a matriz de distâncias passa a ser $O(V^2)$ para cada carrinho, e isso é concatenado na central em $O(V)$. Assim, a complexidade da etapa de montagem da matriz cai de $O(V^3)$ para $O(V^2)$ com a utilização de sistemas distribuídos.

Com relação à arquitetura desse sistema, poderia ser interessante adotar um modelo de *publish-subscribe*, em que cada robô se inscreve em um tópico relacionado ao mapa, que é fornecido pela central, e ao caminho que deverá seguir. Além disso, o robô deve fornecer à central o seu vetor de custo para a entrega de cada caixa e atualizações do mapa, seja ao chegar no seu destino ou até mesmo alertando obstruções presentes no seu percurso. A Figura 8 mostra um esquema de informações trocadas entre a central e um robô.

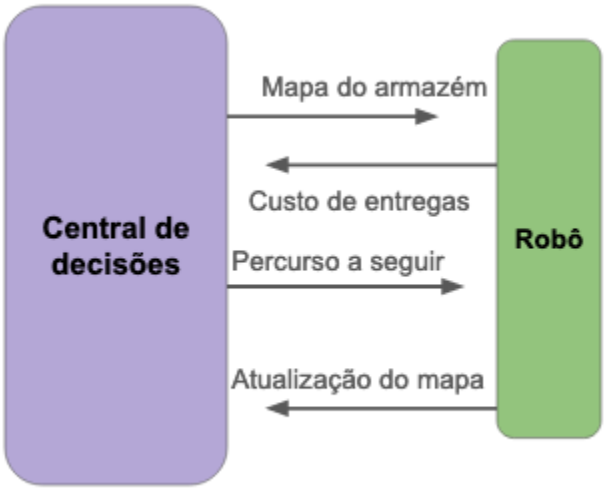

Figura 8: Arquitetura *publish-subscribe* do sistema de comunicação entre robôs e a central

Apesar da complexidade ter sido reduzida nessa etapa, a central de processamento ainda precisa executar o algoritmo húngaro em uma complexidade $O(V^3)$. Nesse caso, por se tratar de uma central, espera-se que ela possua um poder de processamento maior do que

um sistema embarcado como o robô, e poderia ser capaz de realizar essa tarefa em tempo hábil, atuando como um servidor de borda. Isso garantiria uma latência de comunicação menor com os veículos e um poder computacional dependente da necessidade de velocidade na tomada de decisões, bem como o tamanho do armazém[10].

Uma alternativa é mover a central de decisão da borda para a nuvem. Isso permitiria um processamento de dados maior, que pode ser recomendado em armazéns de grande porte, com uma frota de veículos muito grande e um fluxo de movimentação de materiais acima da média. Em contrapartida, mover a decisão para a nuvem pode acarretar em uma latência maior de comunicação, que é algo que deve ser considerado conforme as necessidades e peculiaridades da empresa[10].

Caso o armazém possua um comportamento variável ao longo do ano, como um aumento do número de entregas durante os feriados e uma menor demanda durante o restante do ano, a empresa poderia ainda implementar uma estrutura de computação em névoa. Dessa maneira, a responsabilidade de processamento pode ser alocada para a borda ou para a nuvem, conforme o aumento do número de nós no modelo em grafo[10].

Em um cenário em que a empresa queira realizar um corte de gastos em poder computacional e precise de uma maneira de reduzir a complexidade da atribuição das tarefas, é possível implementar algumas estratégias de aproximação da solução ótima.

5.4 Estratégias de aproximação

Uma estratégia que pode ser utilizada é a geração de n atribuições aleatórias e o cálculo do custo de cada uma delas, selecionando aquela que gerou o menor custo ao final da análise. Para grafos com um número reduzido de nós, é possível encontrar a solução ótima em poucas iterações, já que as atribuições aleatórias eventualmente vão chegar próximo do valor mínimo. A Figura 9 mostra o custo encontrado iterativamente em atribuições aleatórias em um grafo de 30 nós com 8 tarefas.

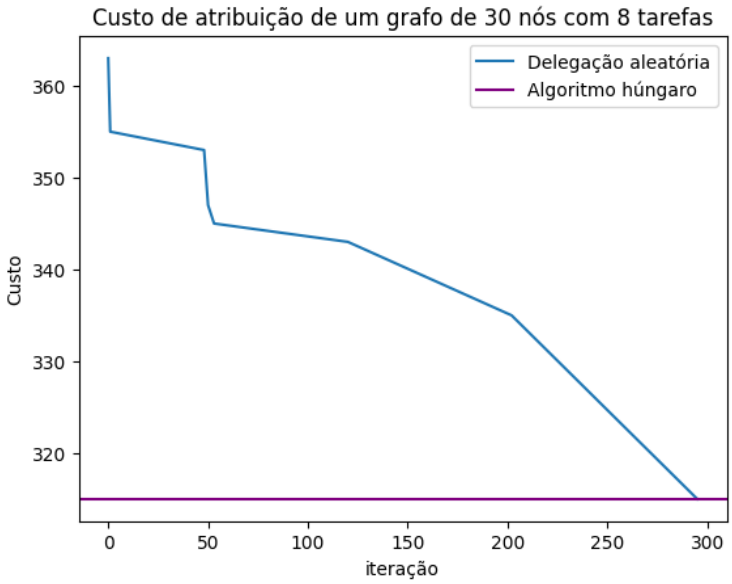

Figura 9: Custo total a partir de atribuições aleatórias

No caso de grafos mais complexos, buscar a solução ótima aleatoriamente pode ser uma estratégia muito arriscada, já que existem muitas possibilidades de atribuição e encontrar a de menor custo rapidamente depende de sorte, além de não ser possível garantir que é de fato a melhor atribuição.

Por essa razão, pode ser interessante utilizar uma estratégia gulosa para resolver o problema. Essa estratégia consiste em, dada a matriz de custos, selecionar iterativamente para um robô, a tarefa ainda não delegada que produz o menor custo para esse veículo. Com isso, a ideia é sempre buscar a melhor solução do sub-problema de um veículo buscando uma caixa, sem se preocupar com a solução global. Isso reduziria a complexidade de tempo de $O(V^3)$ do algoritmo húngaro para $O(V^2)$ do algoritmo guloso, mas em contrapartida a solução não é necessariamente a melhor possível.

As Figuras 10, 11, 12, 13, 14 e 15 mostram grafos de diferentes tamanhos e as soluções encontradas pelo algoritmo húngaro, pela delegação aleatória e pela estratégia gulosa.

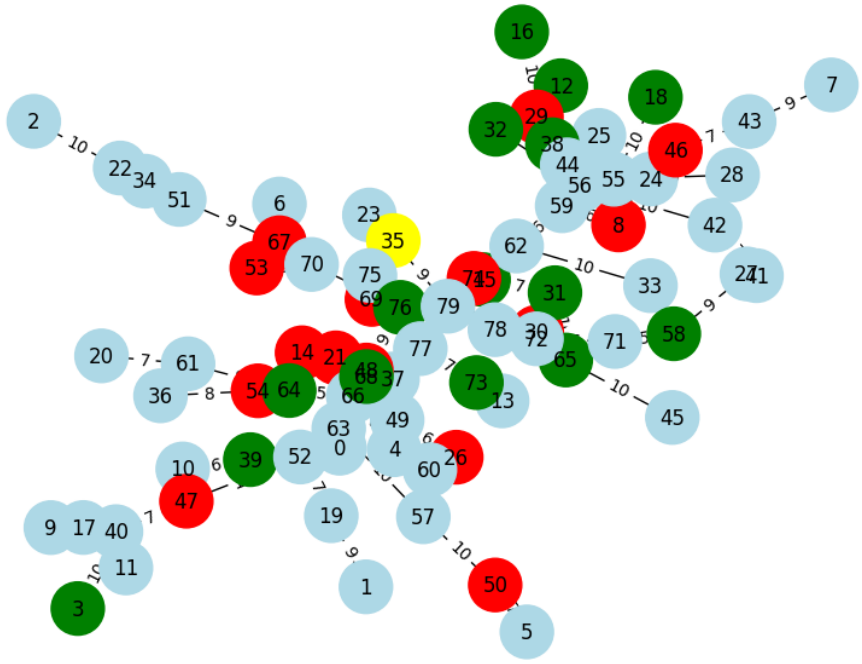

Figura 10: Modelo de armazém com 80 nós e 15 atribuições

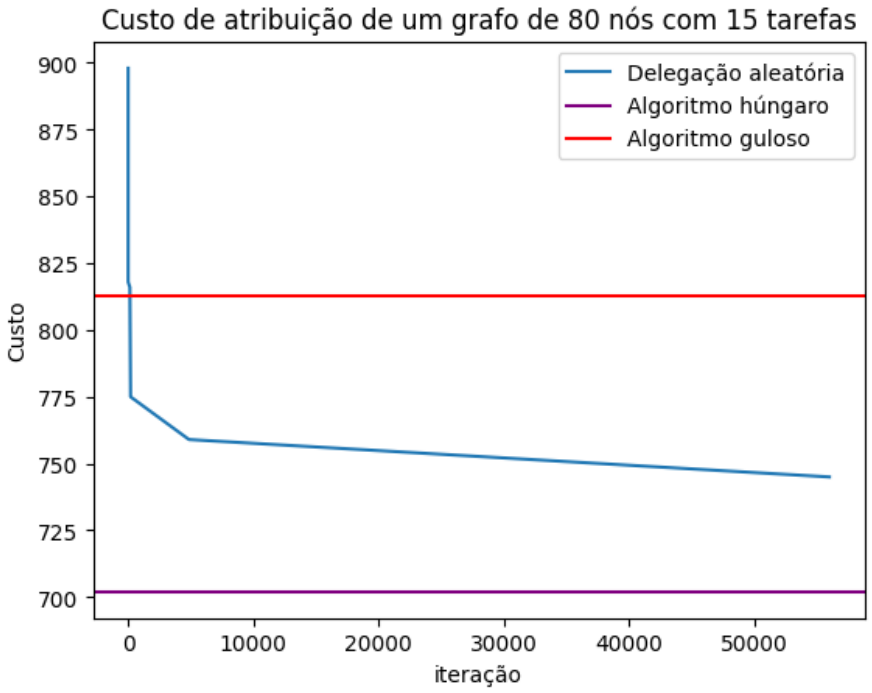

Figura 11: Desempenho de algoritmos no grafo de 80 nós e 15 atribuições

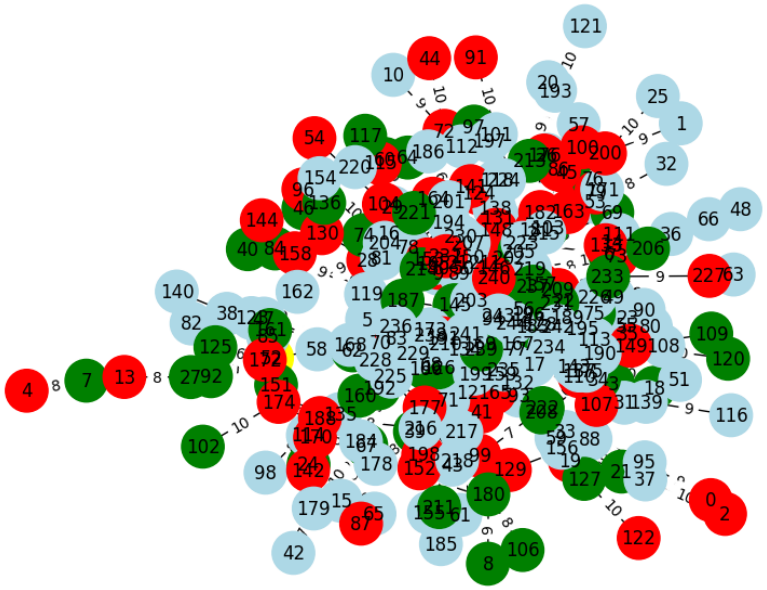

Figura 12: Modelo de armazém com 245 nós e 60 atribuições

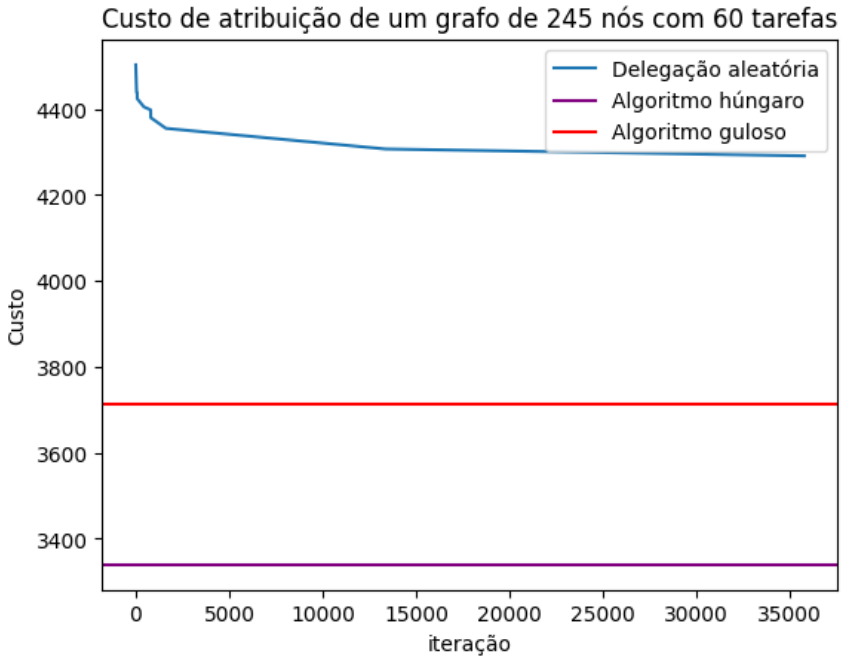

Figura 13: Desempenho de algoritmos no grafo de 245 nós e 60 atribuições

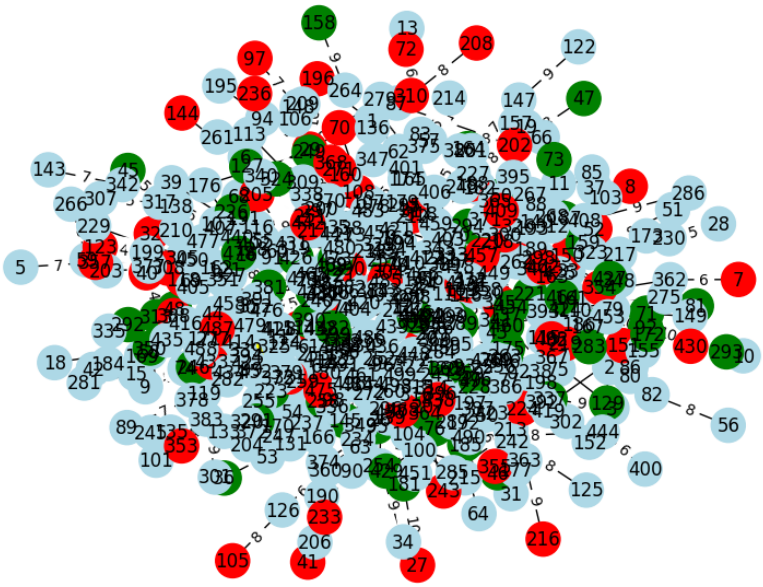

Figura 14: Modelo de armazém com 500 nós e 70 atribuições

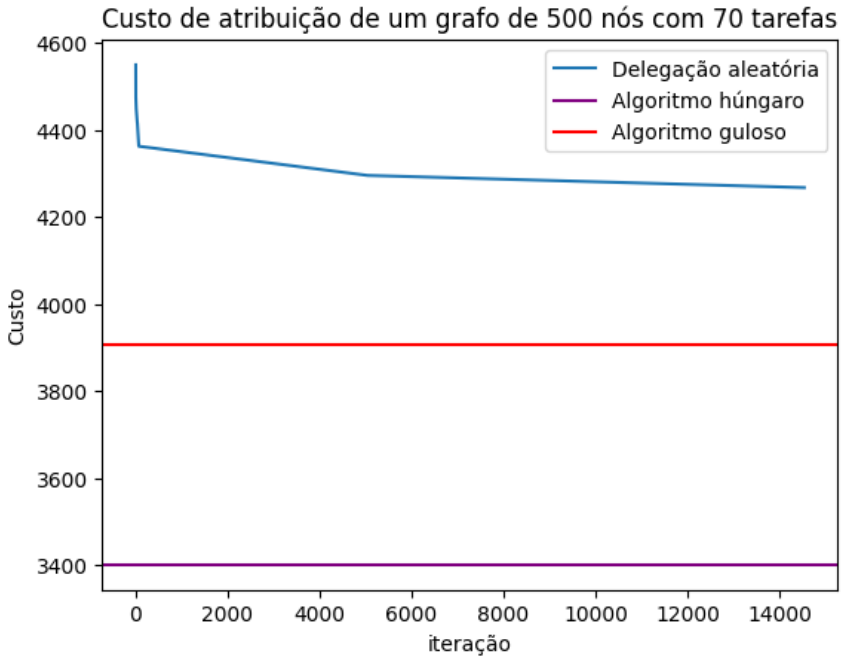

Figura 15: Desempenho de algoritmos no grafo de 500 nós e 70 atribuições

Como foi possível observar, em grafos de tamanho menor, a atribuição aleatória se mostrou uma opção interessante, superando inclusive a estratégia gulosa em desempenho, e tendo uma complexidade linear.

No entanto, nos gráficos mais complexos, a estratégia gulosa se mostrou mais eficiente, encontrando uma solução relativamente próxima da ideal, tudo isso por um custo computacional mediano.

Já o algoritmo húngaro encontra sempre a melhor solução para o problema, mas fazendo isso em uma complexidade elevada, o que pode ser problemático em modelos de grafos com uma alta quantidade de nós.

6 CONCLUSÕES

O problema de otimização de custos na movimentação de materiais em um armazém logístico se trata de uma tarefa complexa e específica de cada organização. Como foi possível observar ao longo dos experimentos, existem diversas variáveis que podem ser analisadas no problema de modo a modelar de maneira mais fiel possível o caso real.

Descrever matematicamente um fenômeno físico é algo interpretativo e adaptável, à medida que pode ser estudado de formas diferentes conforme as necessidades de uma organização ou empresa.

Como foi analisado, o problema pode ser modelado como um grafo com pesos variáveis em relação à autonomia de bateria dos veículos e a massa de cada material que será transportado, em um cenário em que há uma estação de recarga de bateria e um destino final de entrega das caixas. Além disso, diferentes maneiras de lidar com a complexidade de processamento foram exploradas, cada uma delas com suas peculiaridades, vantagens e desvantagens.

O uso de sistemas distribuídos se mostrou uma estratégia interessante para dividir o custo computacional, uma vez que aproveita o poder de processamento dos veículos para contribuir na montagem da matriz de custos.

A arquitetura de *publish-subscribe* apresenta vantagens no processo de comunicação e transmissão dos dados, já que não necessita de uma troca constante de

informações, mas sim quando eventos são notificados, que alteram o comportamento do sistema.

Além disso, diferentes maneiras podem ser utilizadas para otimizar o processamento de dados a fim de encontrar a atribuição de tarefas no grafo. A utilização de computação em borda, nuvem ou névoa são decisões que precisam ser tomadas de acordo com as necessidades da organização.

Já a função de decisão e delegação pode ser realizada de diferentes maneiras. Uma delegação aleatória se mostrou interessante para os casos em que o número de nós é reduzido, além de possuir uma complexidade linear. A atribuição gulosa, por sua vez, possui um comportamento equilibrado para grafos maiores, já que permite ganhos em complexidade e chega relativamente mais próxima da solução ótima. Finalmente, o algoritmo húngaro encontra a melhor solução possível, mas sua utilização deve ser ponderada com o alto custo de processamento que ele demanda.

Em conclusão, diversas estratégias podem ser utilizadas para extrair o máximo de eficiência na movimentação de materiais em armazéns logísticos, e cabe a cada organização uma ponderação de quais características são mais adequadas para as peculiaridades de sua aplicação.

7 REFERÊNCIAS BIBLIOGRÁFICAS

1 - Baudrillard, Jean. "A sociedade de consumo." (1995).

2 - BAUMAN, Zygmunt. A cultura do lixo. Em: Vidas Desperdiçadas. Rio de Janeiro: Jorge Zahar Editor, 2004, pág.117-164.

3 - London, Bernard. "Ending the depression through planned obsolescence." *Revue du MAUSS* 44.2 (2014): 47-50.

4 - Chiavenato, Idalberto. Os novos paradigmas: Como as mudanças estão mexendo com as empresas. São Paulo: Manole Ltda, 2008

5 - Nascimento, Elaine, et al. "TECNOLOGIA NA MOVIMENTAÇÃO DE MATERIAIS." *Diálogos Interdisciplinares* 8.5 (2019): 132-143.

6 - de Melo Cruz, Wander Luis. "Crescimento do e-commerce no Brasil: desenvolvimento, serviços logísticos e o impulso da pandemia de Covid-19." *GeoTextos* (2021).

7 - Brown Aaron. “Rise of the machines? Amazon's army of more than 100,000 warehouse robots still can't replace humans because they lack 'common sense'. DailyMail (2018)” Disponível em: <https://www.dailymail.co.uk/sciencetech/article-5808319/Amazon-100-000-warehouse-robots-company-insists-replace-humans.html>

8 - Cormen, Thomas H.; Leiserson, Charles E.; Rivest, Ronald L.; Stein, Clifford (2001). *Introduction to Algorithms* (Second ed.). MIT Press and McGraw–Hill. pp. 595–601. ISBN 0-262-03293-7.

9 - Harold W. Kuhn, "The Hungarian Method for the assignment problem", *Naval Research Logistics Quarterly*, 2: 83–97, 1955. Kuhn's original publication.

10 - Van Steen, Maarten, and Andrew S. Tanenbaum. *Distributed systems*. Leiden, The Netherlands: Maarten van Steen, 2017.

11 - Networkx Library. https://networkx.org/

ANEXO

Repositório no github com o código utilizado para a realização dos experimentos:

https://github.com/jvbm07/warehouse-graph-analysis