http://journal.unoeste.br/index.php/ce/index
DOI: 10.5747/ce.2021.v13.n2.e360
ISSN on-line 2178-8332

Colloquium Exactarum

Submetido: 07/03/2021 Revisado: 13/05/2021 Aceito: 24/07/2021

OTIMIZAÇÃO DE ROTAS DE AUTOMÓVEIS E ESCALAS PARA ATENDIMENTO DE HOME CARE

Optimization of car routes and scales for Home Care

Tulio Olivieri, Francisco Maracci, Daniela Russi

Universidade do Oeste Paulista
Faculdade de Informática de Presidente Prudente - FIPP
tuliolivieri@gmail.com, francisco@unoeste.br, daniela@unoeste.br

RESUMO – A frota de veículos brasileira vem crescendo ano a ano. Esse aumento no número de veículos traz consigo alguns problemas, como congestionamentos e aumento de emissões de gases causadores do efeito estufa. Em quase todos os centros urbanos estacionar fora de estabelecimentos privados é uma tarefa difícil, e o tempo gasto com o deslocamento é cada vez mais alto. Como alternativa para diminuir o número de veículos em circulação, o compartilhamento de veículos, como uma espécie de carona entre pessoas que fazem trajetos parecidos, está ganhando espaço em centros urbanos de todo o mundo. De olho nos benefícios financeiros e ambientais oferecidos por essa alternativa, algumas empresas estão começando a aderir ao seu uso, agrupando profissionais que fazem caminhos parecidos em grupos que utilizam um mesmo veículo. Este trabalho tem como objetivo buscar maneiras de otimizar a rota que esse grupo de profissionais faz em cada ciclo de trabalho, mitigando custos de manutenção veicular e reduzindo gastos com combustível, o que consequentemente diminui emissões de gases poluentes. Para os clientes, isso pode oferecer uma maior pontualidade na prestação do serviço e maior disponibilidade de tempo dos profissionais. Ao final do trabalho espera-se que seja desenvolvido um aplicativo que faça o cálculo de rotas da melhor maneira possível, possibilitando que os seus usuários tenham um melhor aproveitamento do tempo ao utilizá-lo.
Palavras-chave: Teoria dos Grafos; otimização; melhor rota; Home Care.

ABSTRACT – The Brazilian vehicle fleet has been growing year by year. This increase in the number of vehicles brings with it some problems, such as congestion and an increase in greenhouse gas emissions. In almost all urban centers, parking outside private establishments is a difficult task, and the time spent on commuting is increasingly high. As an alternative to reduce the number of vehicles in circulation, the sharing of vehicles, as a kind of ride between people who make similar journeys, is gaining ground in urban centers around the world. With an eye on the financial and environmental benefits offered by this alternative, some companies are beginning to adhere to its use, grouping professionals who follow similar paths in groups that use the same vehicle. This work aims to seek ways to optimize the route that this group of professionals takes in each work cycle, mitigating vehicle maintenance costs and reducing fuel costs, which consequently reduces pollutant gas emissions. For customers, this can offer greater punctuality in the provision of the service and greater availability of professionals' time. At the end of the work, it is expected that an application will be developed that performs the route calculation in the best possible way, allowing its users to have a better use of time when using it.
Keywords: Graph Theory; optimization; best route; Home Care.

1. INTRODUÇÃO

1.1. Contextualização e Problemática

Segundo dados do Instituto Brasileiro de Geografia e Estatística (BRASIL, 2019), no período de 2006 a 2018, a frota de veículos brasileira saltou de 45.029.257 para 100.746.553, sendo que a frota de automóveis de passeio no mesmo período saltou de 27.700.608 para 54.715.488. Junto com esse aumento do número de automóveis de passeio, temos um aumento da idade média da frota do país, que segundo dados do Sindicato Nacional da Indústria de Componentes para Veículos Automotores (2019) atingiu 9 anos e 7 meses em 2018, um aumento de 11 meses em relação a 2015.

Uma frota mais antiga significa veículos com tecnologias ultrapassadas em termos de motores, o que pode refletir diretamente na quantidade de gases poluentes emitidos pelos mesmos, uma vez que as normas de poluição que precisavam ser atendidas para homologação de um modelo poderiam ser menos rígidas anos atrás, permitindo que veículos que podem não atender as normas atuais transitem normalmente nos dias de hoje (CETESB, 2017).

Esse número crescente de veículos também reflete em maior trânsito nas cidades, uma vez que essas podem não ter sido planejadas para um grande número de automóveis, e, também, na dificuldade em estacionar nos grandes centros urbanos, já que uma grande concentração de pessoas em áreas comerciais gera uma grande demanda de vagas para veículos (ARYANDOUST; VAN VLIET; PATT, 2019).

Como resposta a essa dificuldade em relação ao trânsito, vagas de estacionamento e preocupação ambiental em diminuir a poluição proveniente do setor de transportes, surgiram iniciativas de países da União Europeia, Ásia e o Estados Unidos que incentivam o transporte coletivo, transportes alternativos como a bicicleta e o patinete e transportes compartilhados (DIJK; ORSATO; KEMP, 2013).

Junto as alternativas ao transporte individual citadas acima temos também a carona entre pessoas que fazem trajetos parecidos, que podem envolver o percurso entre casa, trabalho e faculdade, por exemplo. Uma das suas vantagens é a mitigação de custos em relação a manutenção de um transporte equivalente de forma individual, de forma que os custos necessários para o transporte e manutenção do veículo são divididos entre as pessoas que o compartilham. Do ponto de vista da mobilidade urbana, isso significa que onde antes teríamos até cinco carros fazendo um trajeto parecido, agora passamos a ter apenas um, o que reflete em uma diminuição nos gases poluentes provenientes do trânsito e menos veículos nas ruas. Podemos citar como ferramentas o UBER por exemplo (FURUHATA et al., 2013)(OLSSON; MAIER; FRIMAN, 2019).

Junto com a diminuição do número de carros nas ruas, temos como outro meio para diminuir os recursos gastos no trânsito (como combustível e a manutenção de peças desgastadas de veículos) a otimização das rotas, que trazem benefícios para todos que de alguma forma dependem de transportes urbanos.

Ao otimizar as rotas diminuímos, entre outras coisas, o tempo gasto diariamente no trânsito, o que diminui os efeitos nocivos à saúde provenientes do mesmo. Pesquisas indicam que longos períodos no trânsito tendem a aumentar alguns problemas relacionados ao estresse, como ansiedade, cefaleia, nervosismo, impaciência, irritabilidade e até depressão (ESTRESSE... 2014).

A otimização de rotas é um problema comum em várias áreas da civilização moderna, como rotas de roteamento de internet e de veículos no trânsito urbano, tendo esse último ganhado certa relevância nos últimos anos. Algumas estruturas e algoritmos de busca de melhor caminho, provenientes da Teoria dos Grafos, têm sido utilizados para propor soluções eficientes para esses problemas (PAIM, 2015) (BRAGA, 2016).

Grafos são uma maneira de representação de entidades e suas relações entre si, sendo muito utilizadas para representar modelos do mundo real, como redes sociais e mapas de trânsito, por exemplo. Um dos principais assuntos da Teoria dos Grafos é o problema da busca do caminho mais curto entre vértices de um grafo, sendo esse um dos algoritmos mais estudados no ramo da Ciência da Computação devido a sua aplicabilidade em modelos de problemas reais (VINAGREIRO, GOLDMAN, s. d.) (CHEN, GOTSMAN, 2018).

Tecnologias como APIs de mapas que fornecem distâncias e rotas como o Google Maps Platform, OpenLayers e OpenStreetMaps, sendo essas duas últimas gratuitas, podem fornecer a distância entre os vários pontos de rota desejados, podendo ser utilizados para a criação de um grafo com arestas valoradas, e a partir desse grafo um algoritmo de melhor caminho pode ser aplicado para obtermos possíveis rotas.

Aplicativos de Carpool, como Waze Carpool, e aplicativos de mapas oferecem rotas com várias paradas, porém levam em consideração apenas distância e a ordem de paradas do trajeto, enquanto os profissionais ainda precisariam fazer cálculos de tempo e necessidades para obter a ordem de atendimento e inserir manualmente endereços, o que torna o processo mais demorado e trabalhoso.

Se para qualquer um que vive em grandes centros o trânsito é um problema, para quem carece de atendimento médico especializado e necessita de transporte que atenda a alguma necessidade específica de locomoção ele pode se tornar um empecilho ainda maior, o que vem dando espaço para o crescimento do atendimento de saúde domiciliar.

O chamado Home Care é um tipo de assistência médica que se encontra em expansão no Brasil e no mundo. É uma alternativa à internação hospitalar comum que diminui a estadia do paciente no hospital durante o tratamento e traz com isso a redução de custos de internação e diminuição dos riscos de complicações provenientes deste tipo de ambiente. Além disso, a casa do paciente é reconhecida como um ambiente favorável a recuperação do paciente e com capacidade de oferecer bons resultados para um tipo de atendimento relativamente novo, que pode fornecer as especialidades necessárias ao paciente sem os transtornos que envolveriam a estadia no hospital ou o deslocamento do paciente até ele quando necessário (ANDRADE AM, et. al, 2016).

Buscando um novo conceito dos atendimentos, algumas empresas que prestam serviços de Home Care estão apostando no compartilhamento de veículos entre seus profissionais que prestam atendimento a domicílio. A otimização dos veículos e rotas nos atendimentos acabam mitigando custos de manutenção da frota, gerando economia de combustível e diminuindo a emissão de gases causadores do efeito estufa.

Junto com esse novo serviço, começam a surgir alguns problemas relacionados à eficiência da rota feita entre uma residência e outra, o tempo ocioso de um ou mais especialista durante um atendimento, uma vez que a equipe nem sempre se faz necessária a todos os atendimentos estabelecidos na rota definida.

Muitas vezes os recursos gastos em uma série de atendimentos poderiam ser melhor aproveitados, com o planejamento correto. Além disso, o tempo gasto no trânsito em um trajeto sem planejamento pode acarretar em maior gasto de combustível e subaproveitamento de tempo, o que pode resultar em um menor número de atendimentos por dia e consequentemente um lucro e menor.

Do ponto de vista do paciente, isso aumenta as chances de atrasos e cancelamentos. A lista de atendimento é grande entre o tempo de uma consulta e outra. O tratamento dos pacientes em estado de recuperação é mais lento e pode tornar-se insuficiente para sua recuperação.

1.2. Objetivos

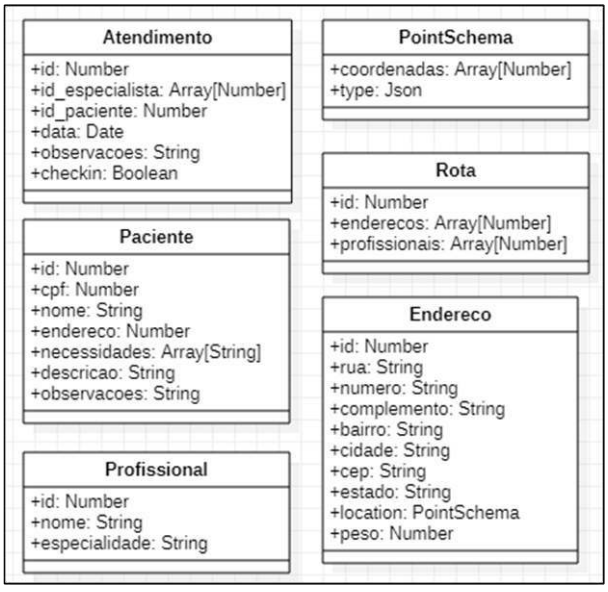
Este trabalho teve como objetivo desenvolver um aplicativo que faz uso de algoritmos de roteirização para encontrar o melhor caminho utilizadas nos atendimentos do tipo Home Care, visando agilizar o atendimento de pacientes e reduzir custos.

2. MÉTODO

2.1. Modelagens

Na figura 1 encontra-se o diagrama de classes, representando os objetos utilizados pela aplicação para a execução das funções e persistência dos dados.

Figura 1. Diagrama de classes.



Fonte: o autor.

2.2. Ferramentas Utilizadas

A aplicação foi desenvolvida no Visual Studio Code, utilizando o Yarn para gerenciar as dependências do projeto e o Android Device

Manager do Android Studio para os testes da parte mobile da aplicação.

O front-end do módulo web da aplicação foi desenvolvido em React, uma biblioteca JavaScript para criação de interfaces, enquanto o módulo mobile foi construído em React Native, um framework JavaScript para criação de interfaces para dispositivos móveis.

Já o back-end da aplicação foi desenvolvido em Node.js (um ambiente de execução orientado a eventos) e JavaScript.

As requisições feitas pelas aplicações web e mobile são gerenciadas pelo back-end através do framework Express, que encaminha as solicitações de diferentes requisições HTTP para seus devidos módulos.

Para obter os dados de latitude e longitude utilizados pelo algoritmo para calcular as rotas, foi utilizada a API Open Search, fornecida pela MapQuest, enquanto os mapas apresentados pela aplicação foram todos implementados com a biblioteca Leaflet.

2.3. Algoritmo Desenvolvido

O algoritmo desenvolvido para a resolução do problema inicial, de buscar a melhor rota a ser seguida, foi feito com base na meta-heurística GRASP (sigla em inglês para Procedimento de Busca Adaptativa Randomizada e Gulosa).

A meta-heurística GRASP é um algoritmo de busca dividido em duas fases: construção e busca local. Na primeira fase é realizada a construção de uma rota que atenda ao problema inicial, e na segunda uma busca local é feita na tentativa de melhoria da solução inicial (RESENDE; RIBEIRO, 2010).

No processo de construção da solução inicial, foi feita uma alteração na forma como os candidatos a posição na rota são escolhidos: enquanto no GRASP original um grupo tido como melhores candidatos são colocados em uma lista denominada RCL (restricted candidate list) e depois um dos elementos é sorteado para integrar a solução, no algoritmo desenvolvido os candidatos são ordenados do melhor para o pior custo, e então um dos n melhores candidatos (n definido por parâmetro) é escolhido para compor a solução. Essa alteração evita o uso de uma estrutura adicional no algoritmo que, em casos onde o número de candidatos é grande, será igualmente grande.

Já na fase da busca local foi utilizado um método estocástico para obter novos resultados como possíveis candidatos a melhor rota. O número de execuções do método estocástico é definido por um parâmetro chamado melhoria máxima.

Para atender ao problema proposto, o algoritmo utiliza não apenas a distância como critério de escolha, mas sim um conjunto de distância e atendimentos que serão realizados na residência.

Nas figuras a seguir, podemos ver as funções que compõem o algoritmo desenvolvido e a diferença entre a fase de construção original (Figura 2) e a variação desenvolvida (Figura 3):

Figura 2. Fase de construção original.

Algorithm 2.8.2: Pseudocode the `GreedyRandomized-Construction` function.

Input: α
Output: $S_{candidate}$
$S_{candidate} \leftarrow \emptyset$;
while $S_{candidate} \neq$ ProblemSize **do**
  $Feature_{costs} \leftarrow \emptyset$;
  for $Feature_i \notin S_{candidate}$ **do**
    $Feature_{costs} \leftarrow$ `CostOfAddingFeatureToSolution`$(S_{candidate}, Feature_i)$;
  end
  RCL $\leftarrow \emptyset$;
  $Fcost_{min} \leftarrow$ `MinCost`$(Feature_{costs})$;
  $Fcost_{max} \leftarrow$ `MaxCost`$(Feature_{costs})$;
  for $Fi cost \in Feature_{costs}$ **do**
    if $Fi cost \leq Fcost_{min} + \alpha \cdot (Fcost_{max} - Fcost_{min})$ **then**
      RCL $\leftarrow Feature_i$;
    end
  end
  $S_{candidate} \leftarrow$ `SelectRandomFeature`(RCL);
end
return $S_{candidate}$;

Fonte: (Brownlee, 2012).

Figura 3. Fase de construção desenvolvida.

```
function greedyRandomizedConstructionVariant()
{
 1   entrada locais;
 2   entrada inicio;
 3   entrada n;
 4   saida solucao = [];
 5   var candidatos = locais.copiar();
 6   var custos = [];
 7   candidatos.ordena();
 8   var bestCusto = calculaCusto(inicio, candidatos[0]);
 9   custos.push(bestCusto);
10   para(i = 2; até i = candidatos.length)
11        custos.push(calculaCusto(inicio, candidatos[0]));
12    enquanto(candidatos.length > 1)
13    {
14        var pos;
15        se(candidatos.length > 2)
16            pos = Math.floor((Math.random() * n));
17        senão
18            pos = Math.floor((Math.random() * candidatos.length));
19        solucao.push(candidatos[pos]);
20        candidatos.remover(pos);
21        candidatos.ordena();
22        bestCusto = calculaCusto(solucao[solucao.length - 1], candidatos[0]);
23        para(i = 2; até i = candidatos.length)
24            custos.push(calculaCusto(solucao[solucao.length - 1], candidatos[0]));
25    }
26    solucao.push(candidatos[0]);
}
```

Fonte: o autor.

Figura 4. Busca Local

```
procedure local(P,N(P),s)
1     for s not locally optimal →
2           Find a better solution t ∈ N(s);
3           Let s = t;
4     rof;
5     return(s as local optimal for P)
end local;
```

Fonte: (Feo; Resende, 1995).

Figura 5. Método estocástico

```
def stochastic_two_opt(permutation)
  perm = Array.new(permutation)
  c1, c2 = rand(perm.size), rand(perm.size)
  exclude = [c1]
  exclude << ((c1==0) ? perm.size-1 : c1-1)
  exclude << ((c1==perm.size-1) ? 0 : c1+1)
  c2 = rand(perm.size) while exclude.include?(c2)
  c1, c2 = c2, c1 if c2 < c1
  perm[c1...c2] = perm[c1...c2].reverse
  return perm
end
```

Fonte: (Brownlee, 2012).

Figura 6. Função que faz a chamada dos algoritmos de construção e busca local.

```
function greedyRandomizedAdaptiveSearchProcedure()
{
 1    entrada locais;
 2    entrada inicio;
 3    entrada maximo_iteracoes;
 4    entrada melhoria_maxima;
 5    saida melhor;
 6
 7    melhor = [];
 8    var candidato;
 9    var menorCusto = 0;
10
11    para(i = 1; até i = maximo_iteracoe)
12    {
13        candidato = greedyRandomizedConstructionVariant(locais, inicio);
14        candidato = buscaLocal(candidato, inicio, melhoria_maxima);
15
16        se(melhor.tamanho == 0 || custo(candidato, inicio) < menorCusto)
17        {
18            melhor = candidato;
19            menorCusto = custo(candidato, inicio);
20        }
21    }
}
```

Fonte: o autor.

2.4. Metodologia dos Testes

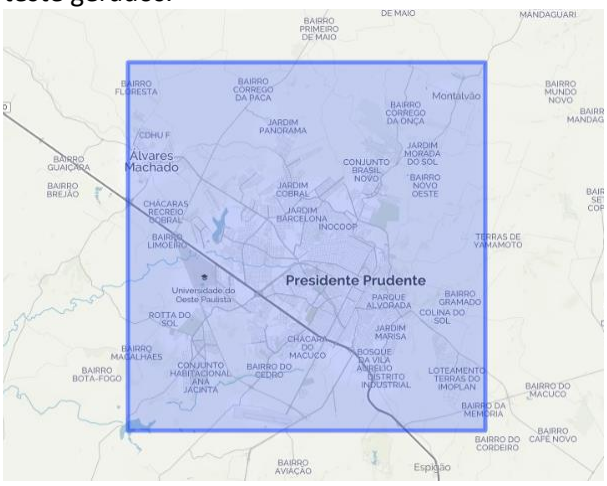
Para testar o algoritmo desenvolvido, primeiro foram sorteados locais dentro do retângulo que compreende Presidente Prudente, porém como o formato do município é irregular, muitos deles eram em locais pouco acessíveis. Para contornar isso, foram selecionados apenas os endereços que também pertenciam ao polígono da área urbana da cidade, como vemos abaixo:

Quadro 1. Coordenadas das diagonais do retângulo.

Diagonal	Latitude	Longitude
1	-22.18042	-51.481822
2	-22.038849	-51.335039

Fonte: o autor.

Figura 7. Retângulo que compreende os casos de teste gerados.



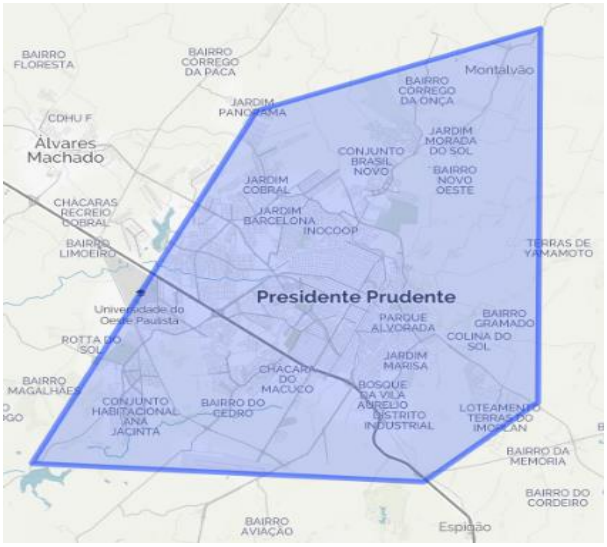
Fonte: o autor.

Quadro 2. Coordenadas dos vértices do polígono.

Vértice	Latitude	Longitude
1	-22.18042	-51.368734
2	-22.174706	-51.481822
3	-22.064525	-51.417007
4	-22.038849	-51.335039
5	-22.155711	-51.33562

Fonte: o autor.

Figura 8. Polígono que compreende os casos de teste escolhidos.



Fonte: o autor.

Ao todo foram sorteados 18 conjuntos de latitude e longitude para comporem os casos de teste.

Após escolher os locais, foram sorteados quais deles iriam para cada um dos 3 casos de teste. O primeiro caso possui 3 endereços, o segundo 5, e o último 10, chegando aos seguintes conjuntos:

Quadro 3. Caso de teste 1.

Locais	Latitude	Longitude
1	-22.1313577	-51.4477181
2	-22.1471772	-51.3758653
3	-22.1186717	-51.378204

Fonte: o autor.

Quadro 4. Caso de teste 2.

Locais	Latitude	Longitude
1	-22.1370775	-51.4186016
2	-22.1384986	-51.3922516
3	-22.1465409	-51.3832212
4	-22.136123	-51.3774918
5	-22.110272	-51.3850406

Fonte: o autor.

Quadro 5. Caso de teste 3.

Locais	Latitude	Longitude
1	-22.1259367	-51.3931548
2	-22.112124	-51.400825
3	-22.1212771	-51.4307221
4	-22.1118842	-51.4297077
5	-22.1125459	-51.4425976
6	-22.084368	-51.4351297
7	-22.087532	-51.384055
8	-22.0754558	-51.3625407
9	-22.098447	-51.3978063
10	-22.1301472	-51.4086177

Fonte: o autor.

Para analisar o desempenho e comparar os resultados obtidos pelo algoritmo desenvolvido, foram usados outros algoritmos para comparação, sendo eles: Inserção do Mais Barato e Stochastic Hill Climbing.

O algoritmo de inserção do mais barato consiste em inserir na solução, a cada iteração, um local que não foi visitado e que possua o menor custo de todos os componentes ainda disponíveis (Silva et al, 2013).

Figura 9. Algoritmo de Inserção do Mais Barato.

```
1 HEURÍSTICA inserção do mais barato
2 INÍCIO
3       ENTRADA: Rota;
4       SAÍDA : Rotaminimizada;
5       Ponto_verificar = Rota[1];
6       Rotaminimizada = Rota[1];
7       Menor =1000000;
8       PARA Rota[K] não pertencente a Rotaminizada FAÇA
9           SE distância(Ponto_verificar, Rota[K]) < Maior ENTÃO
10              Menor = distância(Ponto_verificar, Rota[K]);
11              Ponto_Menor = Rota[K];
12          FIM-SE;
13      FIM-PARA;
14      Rotaminimizada = Ponto_Menor;
15      Rotaminimizada = Rota[1];
16      Tamanho_Percorrer = Tamanho(Rota)- 2;
17      ENQUANTO Tamanho_Percorrer > 0 FAÇA
18          Menor = 1000000;
19          Tamanho = 10000000;
20          PARA Rota[K] e não pertencente a Rotaminimizada FAÇA
21              PARA todo Rotaminimizada[I] FAÇA
22                Tamanho2 = distância(Rotaminimizada[I], Rota[k]) + distância(Rota[k],
Rotaminimizada[I+1]) - distância(Rotaminimizada[I], Rotaminimizada[I+1]);
23                SE Tamanho2 < Tamanho ENTÃO
24                   Tamanho = Tamanho2;
25                   Ponto_Inserir = Rotaminimizada[I];
26                   Ponto_Menor = Rota[K];
27                FIM-SE;
28              FIM-PARA;
29          FIM-PARA;
30          InserirPontoApos(Ponto_Inserir,Rotaminimizada,Ponto_Menor);
31          Tamanho_Percorrer = Tamanho_Percorrer - 1;
32      FIM-ENQUANTO;
33 FIM.
```

Fonte: (Silva et al, 2013).

Já o algoritmo Stochastic Hill Climbing é um método de otimização local, sendo uma extensão de algoritmos determinísticos de escalada, como o Simple Hill Climb e Steepest-Ascent Hill Climbing (BROWNLEE, 2012).

Este procedimento consiste em gerar vizinhanças aleatoriamente como candidatas a solução, para evitar o problema das técnicas de escalada determinísticas que ficam presas em mínimos locais por causa da sua natureza gananciosa (BROWNLEE, 2012).

Figura 10. Pseudocódigo do algoritmo Stochastic Hill Climbing.

Algorithm 2.4.1: Pseudocode for Stochastic Hill Climbing.

```
Input: Iter_max, ProblemSize
Output: Current
Current ← RandomSolution(ProblemSize);
foreach iter_i ∈ Iter_max do
    Candidate ← RandomNeighbor(Current);
    if Cost(Candidate) ≥ Cost(Current) then
        Current ← Candidate;
    end
end
return Current;
```

Fonte: (Brownlee, 2012).

Para o cálculo de desempenho dos algoritmos, cada caso de teste foi executado 3 vezes, sendo o tempo de execução considerado a média do tempo obtido nas 3 execuções.

Nos casos em que o valor do parâmetro que define o máximo de iterações não foi suficiente para que o resultado obtido pelo GRASP fosse ótimo nas 3 execuções, o resultado considerado é a média das 3 execuções.

3. RESULTADOS

3.1. Aplicações Desenvolvidas

Ao final do desenvolvimento foram construídas duas aplicações: uma aplicação web para uso em navegadores de internet e uma aplicação para dispositivos móveis.

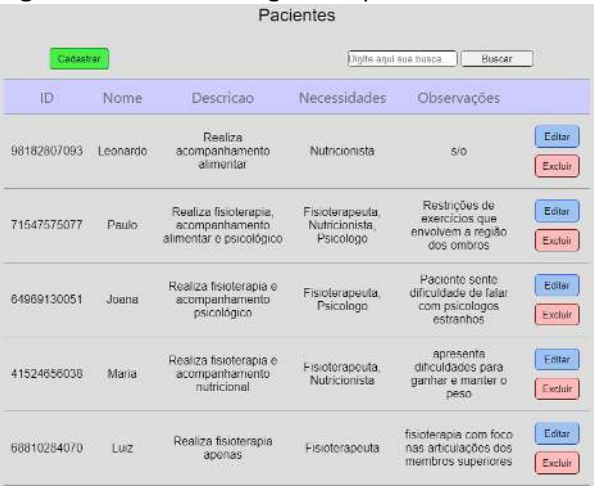
A aplicação web faz, além dos cadastros de clientes e profissionais, a criação das rotas a partir da escolha dos endereços a serem visitados.

Figura 11. Tela de listagem de profissionais



Fonte: o autor.

Figura 12. Tela de listagem de pacientes.



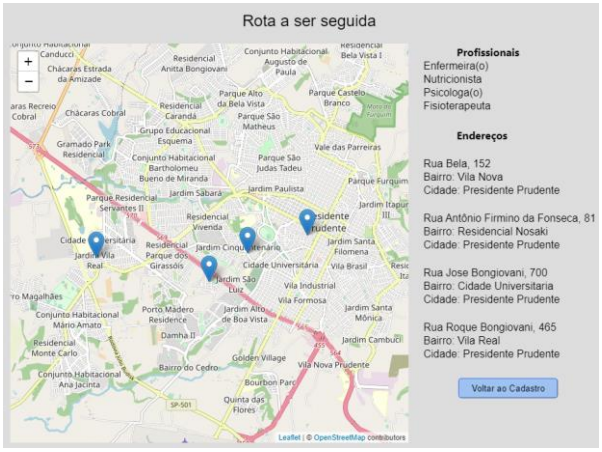
Fonte: o autor.

Figura 13. Tela de escolha dos pacientes a serem atendidos.



Fonte: o autor.

Figura 14. Tela onde são exibidos os locais que serão visitados.



Fonte: o autor.

Após escolher os clientes que serão visitados, o algoritmo faz os cálculos e retorna a ordem a ser seguida, com os locais marcados no mapa e os endereços listados. Além disso, são listados os profissionais que devem ir para os atendimentos.

A aplicação web também permite fazer e acompanhar os check-ins dos atendimentos, possibilitando o acompanhamento dos locais já visitados de uma rota e qual será o próximo ponto de parada da equipe.

Figura 15. Tela de acompanhamento dos atendimentos.



Fonte: o autor.

Na aplicação para dispositivos móveis, é possível apenas fazer o check-in ao chegar em um ponto da rota e também realizar a consulta das rotas geradas.

Figura 16. Tela inicial do aplicativo para dispositivos móveis.



Fonte: o autor.

O mapa mostra apenas os pontos a serem visitados, com a ordem de visita e o endereço completo sendo exibidos ao clicar nos pontos.

Figura 17. Mapa com os locais a serem visitados.



Fonte: o autor.

Figura 18. Tela de check-in da equipe em deslocamento.



Fonte: o autor.

3.2. Resultados

Abaixo vemos os resultados obtidos pelos algoritmos GRASP (com o número máximo de iterações em 50 e 200, para observação de como esse parâmetro afeta o tempo de execução e os resultados obtidos pelo algoritmo), Inserção do Mais Barato e Stochastic Hill Climbing (SHC). Quanto menor os valores das colunas resultado (soma de distância e peso) e tempo, melhor.

Quadro 6. Resultados obtidos no caso de teste 1.

Algoritmo	Resultado	Tempo (ms)
GRASP (50)	11.3941932	196
GRASP(200)	11.3941932	2471
IMB	15.8452484	1
SHC	16.1363943	5

Fonte: o autor.

Quadro 7. Resultados obtidos no caso de teste 2.

Algoritmo	Resultado	Tempo (ms)
GRASP (50)	10.5270753	277
GRASP(200)	10.2992531	4000
IMB	10.7686414	1
SHC	10.6772709	6

Fonte: o autor.

Quadro 8. Resultados obtidos no caso de teste 3.

Algoritmo	Resultado	Tempo (ms)
GRASP (50)	24.5505143	466
GRASP(200)	25.8538292	7510
IMB	29.1383392	3
SHC	27.3672825	16

Fonte: o autor.

4. DISCUSSÃO

Analisando o resultado, vemos que o algoritmo desenvolvido obteve os melhores resultados em todos os casos de teste, porém com um tempo de execução relativamente maior. Quanto maior o número de endereços que compõe a rota e quanto mais iterações são executadas, mais o tempo de execução aumenta.

Ao observar apenas os resultados do GRASP, vemos que quanto mais endereços na rota, mais iterações são necessárias para se obter um resultado próximo ou igual ao ótimo, com um grande aumento do tempo de execução.

Uma forma de se obter valores para o parâmetro máximo de iterações de forma que sempre seja obtido um resultado ótimo ou próximo dele, seria utilizar aprendizado de máquina para a partir de um banco de treinamento obter um conjunto de valores para serem usados como parâmetro em tempo de execução, de acordo com a quantidade de endereços na rota.

Em casos onde o tempo de execução precisa ser baixo, o GRASP com 50 iterações se mostrou melhor que os demais algoritmos e obteve um tempo de execução ainda abaixo de 1 segundo. No caso de teste 3, chegou a ficar até mesmo na frente do GRASP executado com 200 iterações, o que mostra na prática a característica randômica do algoritmo, definida pela escolha aleatória de um dos membros da RCL atual para compor a solução a cada iteração.

5. CONSIDERAÇÕES FINAIS

Considerando um caso real de roteirização para atendimentos de uma equipe de Home Care, as rotas normalmente possuem poucos endereços, o que coloca o GRASP como uma ótima alternativa para a resolução do problema, com bons tempos de execução.

Em casos com rotas maiores, ele ainda apresenta bons resultados, mas o tempo de execução deve ser considerado, podendo ser um empecilho para aplicações que dependem do resultado em tempos baixos e usam muitas iterações.

Futuramente, modelos de aprendizagem de máquina podem ser implementados para realizar o refinamento dos parâmetros usados pelo algoritmo, tornando-o ainda mais eficiente já que evita a execução de iterações desnecessárias, diminuindo o tempo de processamento nos casos onde os parâmetros definidos estão acima do necessário para obter uma solução próxima ou igual a ótima.

AGRADECIMENTOS

A Deus, que me deu a existência, saúde e ânimo para realizar este trabalho.

A meus pais e amigos, que durante a realização não apenas deste trabalho, mas de todo o curso, me deram foças e motivação para continuar, e sempre compreenderam a minha ausência nas diversas ocasiões em que não compareci.

Aos orientadores deste trabalho, que sempre buscaram ajudar, orientar e corrigir o projeto desenvolvido, contribuindo para um melhor desempenho pessoal e um melhor resultado do trabalho.

Aos professores e demais profissionais da Universidade, os quais dividi anos da minha vida e aprendi importantes lições não apenas

acadêmicas, mas sim verdadeiras lições para a vida.

REFERÊNCIAS

ANDRADE, AM; SILVA, KL; SEIXAS, CT; BRAGA, PP. **Nursing practice in home care: an integrative literature review**. Rev Bras Enferm [Internet]. 2017;70(1):199-208. Disponível em: <http://www.scielo.br/pdf/reben/v70n1/0034-7167-reben-70-01-0210.pdf>. Acesso em: 17 set. 2019. DOI: http://dx.doi.org/10.1590/0034-7167-2016-0214.

ARYANDOUST, Arsam; VAN VLIET, Oscar; PATT, Anthony. City-scale car traffic and parking density maps from Uber Movement travel time data, **Scientific Data**, [*s. L.*], p.158. 21 ago. 2019. Disponível em: <https://www.nature.com/articles/s41597-019-0159-6>. Acesso em: 05 set. 2019. PMID: 31434904. DOI: 10.1038/s41597-019-0159-6.

BRAGA, Marcus de Lima. **Análise e Planejamento de Rotas em Redes Viárias.** 2016. 126 p. Tese (Doutorado) - Curso de Engenharia Elétrica, Coppe, Ufrj, Rio de Janeiro, 2016. Disponível em: <https://www.gta.ufrj.br/ftp/gta/TechReports/Marcus16.pdf>. Acesso em: 20 fev. 2020.

BROWNLEE, Jason. **Clever Algorithms**: Nature-Inspired Programming Recipes. [S. l.: s. n.], Jun. 2012. Disponível em: https://github.com/clever-algorithms/CleverAlgorithms. Acesso em: 15 dez. 2020.

CHEN, Renjie; GOTSMAN, Craig. **Efficient Fastest-Path Computations in Road Maps**. 2018. Disponível em: <https://arxiv.org/ftp/arxiv/papers/1810/1810.01776.pdf>. Acesso em: 05 set. 2019.

BRASIL. INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA. (Org.). **Frota de Veículos:** Série histórica. [*s. L.*]: [*s. N.*], 2019. Disponível em: <https://cidades.ibge.gov.br/brasil/pesquisa/22/28120?tipo=grafico&indicador=28122>. Acesso em: 05 set. 2019.

DIJK, Marc; ORSATO, Renato; KEMP, Rene. The emergence of an electric mobility trajectory. **Energy Policy**. [s. L.], p. 135-145. 31 jan. 2013. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0301421512003242?via%3Dihub>. Acesso em: 05 set. 2019. https://doi.org/10.1016/j.enpol.2012.04.024

ESTRESSE no trânsito afeta desde o coração até o sistema reprodutivo. 2014. Disponível em: <https://saudebusiness.com/voce-informa/estresse-no-transito-afeta-desde-o-coracao-ate-o-sistema-reprodutivo/>. Acesso em: 20 fev. 2020.

FEO, Thomas; RESENDE, Mauricio G. C. Greedy Randomized Adaptive Search Procedures. **Journal of Global Optimization**, [S. l.], v. 6, p. 109-133, Mar. 1995. DOI 10.1007/BF01096763. Disponível em: https://www.researchgate.net/publication/225237245_Greedy_Randomized_Adaptive_Search_Procedures. Acesso em: 15 dez. 2020. https://doi.org/10.1007/BF01096763

FRIED, Harold O.; LOVEL, C. A. Knox; SCHMIDT, Shelton S.. **The measurement of productive efficiency and productivity growth.** Nova Iorque: Oxford University Press, 2008. https://doi.org/10.1093/acprof:oso/9780195183528.003.0001

FURUHATA, Masabumi et al. Ridesharing: The state-of-the-art and future directions. **Sciencedirect.** [s. L.], p. 28-46. nov. 2013. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0191261513001483?via%3Dihub>. Acesso em: 19 set. 2019. DOI: https://doi.org/10.1016/j.trb.2013.08.012.

OLSSON, Lars E.; MAIER, Raphaela; FRIMAN, Margareta. Why Do They Ride with Others? Meta-Analysis of Factors Influencing Travelers to Carpool. **Sustainability.** Karlstad. 24 abr. 2019. Disponível em: <https://www.mdpi.com/2071-1050/11/8/2414/htm>. Acesso em: 19 set. 2019. DOI: https://doi.org/10.3390/su11082414.

PAIM, Rodrigo Rodrigues. **Estudo de Algoritmos para Redes de Transporte.** 2015. 53 f. TCC (Graduação) - Curso de Engenharia de Computação e Informação, Escola Politécnica, UFRJ, Rio de Janeiro, 2015. Disponível em: <http://monografias.poli.ufrj.br/monografias/monopoli10012695.pdf>. Acesso em: 20 fev. 2020.

SÃO PAULO. Antônio de Castro Bruni [*et al.*]. Companhia Ambiental do Estado de São Paulo.

PCPV: Plano de Controle de Poluição Veicular 2017- 2019. São Paulo: [s. N.], 2017. 61 p. Disponível em: <https://cetesb.sp.gov.br/veicular/wp-content/uploads/sites/6/2018/01/PCPV-2017-2019.pdf>. Acesso em: 05 set. 2019.

SILVA, Gabriel Altafinineves da; SILVA, Francisco Assis da; RUSSI, Daniela Tereza Ascencio; PAZOTI, Mário Augusto; SISCOUTTO, Robson Augusto. ALGORITMOS HEURÍSTICOS CONSTRUTIVOS APLICADOS AO PROBLEMA DO CAIXEIRO VIAJANTE PARA A DEFINIÇÃO DE ROTAS OTIMIZADAS. **Colloquium Exactarum.** Presidente Prudente, p. 30-46. dez. 2013. Disponível em: http://journal.unoeste.br/index.php/ce/article/view/939/995. Acesso em: 15 dez. 2020. DOI: https://doi.org/10.5747/ce.2013.v05.n2.e058.

SINDICATO NACIONAL DA INDÚSTRIA DE COMPONENTES PARA VEÍCULOS AUTOMOTORES (Org.). **Relatório da Frota Circulante.** São Paulo: [*s. N.*], 2019. 13 p. Disponível em: <https://www.sindipecas.org.br/sindinews/Economia/2019/RelatorioFrotaCirculante_Maio_2019.pdf>. Acesso em: 05 set. 2019.

VINAGREIRO, Marcelo; GOLDMAN, Alfredo. **Um sistema distribuído para busca de caminhos em grafos dinâmicos**. Disponível em: <https://www.ime.usp.br/~gold/publications/pdf/wscad06.pdf>. Acesso em: 05 set. 2019.