UTILIZAÇÃO DOS ALGORITMOS FLOYD-WARSHALL E A* NO MAPEAMENTO DE ROTAS DE HOSPITAIS COM LEITOS DE UTI COVID-19

Wallace Duarte de Holanda
Universidade Federal Rural do Semi-Árido
BR 226, km 405 - São Geraldo, Pau dos Ferros, RN — CEP: 59900-000
wallace.holanda@alunos.ufersa.edu.br

Maria Adriana Ferreira da Silva
Universidade Federal Rural do Semi-Árido
BR 226, km 405 - São Geraldo, Pau dos Ferros, RN — CEP: 59900-000
maria.silva78326@alunos.ufersa.edu.br

RESUMO

A COVID-19 é uma doença infecciosa causada pelo novo coronavírus. Essa doença é caracterizada pela variabilidade dos sintomas e o alto índice de transmissão, culminando em uma alta procura por leitos de UTI. Devido ao número limitado de vagas, torna-se difícil identificar qual o hospital mais indicado. Dessa forma, este trabalho propõe a utilização dos algoritmos de *Floyd-Warshall* e A* no cálculo da melhor rota de transferência entre hospitais, levando em conta a melhor distância de um hospital de origem ao hospital de destino, ponderado pela quantidade de leitos vagos de UTI. Ao final, um protótipo de uma aplicação foi desenvolvido com intuito de ilustrar o processo de visualização do hospital mais indicado para a transferência de pacientes acometidos pelo COVID-19.

PALAVRAS CHAVE. Algoritmo Floyd-Warshall. Algoritmo A*. Covid 19.

Tópicos (TAG - Teoria e Algoritmos em Grafos, IC - Inteligência Computacional)

ABSTRACT

COVID-19 is an infectious disease caused by the new coronavirus. This disease is characterized by the variability of symptoms and the high rate of transmission, culminating in a high demand for ICU beds. It becomes difficult to identify the limited number of vacancies, making it difficult to identify the limited number of vacancies. Thus, this work proposes the use of *Floyd-Warshall* and A* algorithms to calculate the best transfer route between hospitals, taking into account the best distance from a origin hospital to the destination hospital, weighted by the quantity of vacant ICU beds. At the end, a prototype of an application was developed in order to illustrate the process of visualization of the most suitable hospital for transferring patients committed by COVID-19.

KEYWORDS. Floyd-Warshall Agorithm. Agorithm A*. Covid 19.

Paper topics (TAG - Theory and Algorithms in Graphs, CI - Computational Intelligence)

https://proceedings.science/p/157340?lang=pt-br

1. Introdução

A COVID-19 é uma doença causada pelo novo coronavírus, denominado *SARS-CoV-2*. De acordo com a Organização Mundial da Saúde, o primeiro caso da doença foi registrado em Wuhan, na China, em dezembro de 2019. Desde então, devido ao alto índice de transmissão, casos de indivíduos com a COVID-19 foram registrados em todos os países, culminando em uma séria pandemia [WHO, 2020].

Esse cenário caótico pode ser compreendido por meio do aspecto instável da doença. Devido a essa característica, enquanto alguns pacientes infectados não manifestam sintomas, uma outra parcela sofre significativamente pelo agravamento, acabam necessitando de cuidados especiais nas Unidades de Terapia Intensiva (UTIs) [Noronha et al., 2020].

A subnotificação de casos de COVID-19 se caracteriza como um dos principais desafios no controle da pandemia [Souza et al., 2021]. Em paralelo a este cenário, devido ao número limitado de leitos torna-se difícil saber quando um leito de UTI estará disponível e qual o hospital mais indicado para se realizar a internação, levando em conta a distância ou a quantidade de leitos vagos [Feitosa e Cabral, 2020].

Com isso, este artigo se resume em propor um algoritmo que realize a identificação do hospital mais indicado, tendo em vista a distância necessária e a quantidade de leitos disponíveis nos hospitais do Rio Grande do Norte (RN). Para tal, serão utilizados os algoritmos de *Floyd-Warshall* e A*, possibilitando a construção de uma matriz de caminho mínimo e o cálculo das trajetórias do hospital inicial até o hospital de destino.

O restante do artigo está organizado da seguinte forma: na Seção 2 são apresentados os termos envolvidos na fundamentação teórica; na Seção 3 discute-se sobre alguns trabalhos relacionados com a temática deste estudo; na Seção 4 são ilustrados os passos utilizados na modelagem do problema; na Seção 5 são destacados e discutidos os resultados alcançados; e, na Seção 6 são expostas as considerações finais.

2. Fundamentação Teórica

Esta seção apresenta o problema do mapeamento de leitos de UTI, bem como uma descri ção sobre os algoritmos de *Floyd-Warshall* e A* com suas respectivas complexidades e corretudes.

2.1. O problema do mapeamento de leitos de UTI

O problema do mapeamento de leitos de UTI disponíveis, juntamente com a visualização do caminho ao hospital de destino possuem bastante relevância no cenário atual. Segundo [Cheng et al., 2020], diante do elevado número de casos graves, as estimativas indicam que a taxa de transferência para UTI de pacientes hospitalizados com COVID-19 é significativamente maior do que as taxas de transferência para UTI relatadas para outros pacientes hospitalizados.

Dessa forma, em paralelo com a baixa quantidade de leitos disponíveis se comparado com o número de pacientes infectados, torna-se necessário a existência de um mecanismo que possa melhorar a visualização dos hospitais com leitos disponíveis e identificar a melhor rota de acesso. Para isso, o GPS (Sistema de Posicionamento Global) pode ser utilizado no mapeamento dos hospitais existentes através de coordenadas. Além disso, os algoritmos *Floyd-Warshall* e A* se caracterizam como boas soluções na otimização do problema de rotas.

2.2. O Algoritmo de *Floyd-Warshall*

O algoritmo de *Floyd-Warshall* utiliza o conceito de programação dinâmica e calcula as distâncias mínimas entre todos os pares de nós de um grafo [Arenales et al., 2011]. Ele permite determinar se existe ou não caminho entre dois nós de um grafo, tendo como entrada uma matriz de adjacência que representa um grafo e calcula os caminhos mínimos entre todos os pares de nós

[Cormen et al., 2002]. O mesmo funciona com arcos com custos negativos, entretanto, não deve existir um circuito (grafo simples) com comprimentos negativos [Cormen et al., 2002].

O algoritmo considera os vértices "intermediários" de um caminho mais curto, onde um vértice intermediário de um caminho simples $p = \langle v^1, v^2, \ldots, v_l \rangle$ é qualquer vértice de p diferente de v_1 ou v_l, isto é, qualquer vértice no conjunto $\{v^2, v^3, \ldots, v_{l-1}\}$ [Cormen et al., 2002].

Seja $d_{ij}^{(k)}$, o peso de um caminho mais curto desde o vértice i até o vértice j, para o qual todos os vértices intermediários estão no conjunto $\{1, 2, \ldots, n\}$. Quando $k = 0$, um caminho desde o vértice i até o vértice j sem vértices intermediários com numeração mais alta que 0 não tem absolutamente nenhum vértice intermediário. Tal caminho tem no máximo uma aresta e então $d_{ij}^{(0)} = w_{ij}$ [Cormen et al., 2002; Arenales et al., 2011]. Uma definição recursiva é dada pela Equação 1.

$$d_{ij}^{(k)} = \begin{cases} w_{ij} & \text{se } k = 0, \\ min(d_{ij}^{(k-1)}, d_{ik}^{(k-1)} + d_{kj}^{(k-1)}) & \text{se } k \geq 1. \end{cases} \quad (1)$$

Considerando-se que para qualquer caminho todos os vértices intermediários estão no conjunto$\{1, 2, \ldots, n\}$, a matriz $D^{(n)} = (d_{ij}^{(k)})$ fornece a resposta final: $d_{ij}^{(n)} = \delta(i, j)$ para todo $i, j \in V$ [Cormen et al., 2002]. Com base nessa fórmula, é possível revolver o Problema do Caminho Mais Curto (PCMC) entre todos os pares de vértices, conceito derivado do método de programação dinâmica [Arenales et al., 2011].

O Algoritmo 1 apresenta o pseudocódigo para a resolução do PCMC que contém todos os pares de vértices em um grafo, com base nos procedimentos mencionados. O pseudocódigo é dado da seguinte forma: A entrada do algoritmo é a matriz $W = (w(i, j))$ com $n = |V|$ linhas e colunas e a saída é a matriz $D^{(n)}$.

Algoritmo 1: Pseudocódigo do Algoritimo de *Floyd-Warshall*

```
1  início
2      para i = 1,..., n faça
3          para j = 1,..., n faça
4              D^0 ← w_ij
5          fim
6      fim
7      para k = 1,..., n faça
8          para i = 1 ,..., n faça
9              para j = 1,..., n faça
10                 D_ij^(k) ← min(D_ij^(k-1), D_ik^(k-1) + D_kj^(k-1))
11             fim
12         fim
13     fim
14 fim
```

- **Complexidade e corretude do algoritmo *Floyd-Warshall***

Para o cálculo da complexidade do algoritmo, é necessário avaliar duas partes. A parte inicial do algoritmo é a inicialização, com complexidade igual a $O(n^2)$, tendo em vista a presença de dois laços aninhados. A segunda parte representa o algoritmo que cria uma matriz contendo a

menor distância entre todos os nós. Nesse momento são encontrados três laços aninhados, sendo executado n vezes. Logo, a complexidade geral é portanto, $O(n^3)$.

Para o cálculo da corretude utilizou-se a estratégia de indução matemática, na qual levou-se em consideração o conhecimento dos comprimentos dos caminhos mais curtos entre todos os pares de vértices. O vértice com o rótulo mais alto no caminho é q, para qualquer $q \leq k$.

- **Passo Base.** Para $k = 1$, o algoritmo será correto, pois temos apenas arestas diretas como caminhos. Dessa forma, será possível acessar um nó de origem e um nó de destino de maneira direta.
- **Passo Indutivo.** Seja k o caminho mais curto entre qualquer par de vértice v_k, com partida e chegada no mesmo ponto com q-caminhos, tem-se que $q \leq k - 1$. Assim, ao comparar o valor de v_k com a soma dos pontos de partida e chegada, tem-se o caminho mínimo.

2.3. O Algoritmo A*

O algoritmo A* (*A-Star*) foi descrito por [Hart et al., 1968] e trata-se de um algoritmo de busca em grafos que faz uso de uma heurística para definir a ordem em que os nós são processados, possibilitando assim, uma diminuição no tempo de processamento. Mesmo sendo definido na década de 60, atualmente existem diversas variantes do algoritmo proposto originalmente [Duchoň et al., 2014].

Caso exista apenas um caminho possível entre um dado nó de origem e um dado nó de destino, o A* irá retornar o único caminho. Entretanto, caso existam outros caminhos possíveis, o algoritmo irá retornar o de menor custo. Para isso, o A* possui um mecanismo de avaliação dos nós por meio da combinação de uma função *g(n)*, que representa o custo para se alcançar cada nó, e *h(n)*, que ilustra o menor custo de n até o destino [Edelkamp e Schrödl, 2012]. Essa relação pode ser expressa pela Equação 2:

$$f(n) = g(n) + h(n) \tag{2}$$

Desse modo *f(n)* é, portanto, o custo estimado da solução de menor custo a partir do nó n. Além disso, esta técnica requer a utilização de duas listas auxiliares, sendo elas: i) lista aberta: contém uma coleção de todos os nós gerados, funcionando como uma fila prioritária, de modo que a pesquisa pode simplesmente retirar da fila o nó de menor custo; ii) lista fechada: possui uma coleção de todos os nós já percorridos, evitando que a busca pesquise os nós repetidamente [Edelkamp e Schrödl, 2012]. O código 2 apresenta o funcionamento do algoritmo por meio de um pseudocódigo.

- **Complexidade e corretude do algoritmo A***

A complexidade do algoritmo A* para esse estudo pode ser definida como $O(nlogn)$ - (sendo n o número de vértices), tendo em vista a execução dos seus dois laços: i) o laço externo, possibilitando a verificação se o nó de destino foi alcançado; ii) laço interno, responsável por avaliar os nós adjacentes. Entretanto, a complexidade do algoritmo A*, em muitos casos, pode ser variável, dependendo da complexidade da função heurística utilizada. Neste trabalho, a função heurística foi equivalente ao cálculo da distância entre nós, logo, sendo um valor constante. Dessa forma, a complexidade final permanecerá $O(nlogn)$.

Algoritmo 2: Pseudocódigo do Algoritmo A*

```
Entrada:
Caminho - Conjunto de nós a serem pesquisados
NoInicial - Local de Partida
início
    Inicialize o Caminho partindo do NoInicial
    repita
        Escolha o nó adjacente de menor custo
        se Nó escolhido é o Nó de destino então
            Busca finalizada
        senão
            Expanda o Nó - Verifique seus vizinhos (adjacentes)
            para Nó adjacente inicial até o final faça
                se Nó adjacente está na lista fechada (foi visitado) então
                    Selecione o próximo Nó (retorne ao início do loop)
                senão
                    Calcule o custo para o Nó adjacente atual
                    se Nó atual está na lista aberta e possui menor custo então
                        Remova o Nó atual da lista aberta
                    se Nó atual não está em ambas as listas (aberta e fechada) então
                        Adicione o Nó na lista aberta
            fim
    até Nó de destino ser alcançado;
fim
```

Para o cálculo da corretude do algoritmo A*, também utilizou-se a estratégia de indução matemática, considerando o nó de partida s com destino ao nó t. Além disso, algumas notações devem ser consideradas: i) $d(u)$ é a distância de s até u; ii) $w(u, v)$ é peso da aresta u até v; iii) $h(u)$ é o valor da heurística para o nó u; iv) $d(u)'$ é o menor caminho conhecido de s até u durante a execução do A*. Quando o algoritmo expande uma aresta (u, v), ele define $d'(v) = min(d'(v), d'(u) + w(u, v))$.

Por meio do A* é possível identificar o caminho mais curto de s até t, expandindo as bordas ao longo desse caminho. Cada borda possui uma classificação de no máximo $d(t)$, devido o critério de admissibilidade. Portanto, o algoritmo sempre irá avaliar o caminho, antes de expandir qualquer borda com classificação maior que $d(t)$. Na primeira vez que A* expande uma aresta (u, t), o valor heurístico será 0 e $d'(t)$ será definido para a classificação dessa borda. Entretanto, $d'(t)$ não pode ser maior que $d(t)$, pois esse fato indicaria que o caminho mais curto não seria explorado. Para provar a afirmativa, considere os seguintes passos.

- **Passo Base.** $P = (s = v_0, v_1, \ldots, v_n = t)$ o caminho mais curto de s para t. A afirmação é que para cada $= 0, 1, \ldots, n$, devemos ter $d'(v_i) = d(v_i)$ antes que qualquer aresta com classificação maior que d (t) seja expandida. Dessa forma, o caso base será $d'(s) = d(s) = 0$.

- **Passo Indutivo.** A etapa indutiva usa o critério de admissibilidade. Se $d'(v_i - 1) = d(v_i - 1)$ então a aresta $(v_i - 1, v_i)$ deve estar na lista aberta, e a classificação desta aresta é $d(v_i - 1) +$

$w(v_i - 1, v_i) + h(v_i)$. Mas a heurística de admissibilidade garante que $h(v_i) \leq d(t) - d(v_i)$ uma vez que $d(t) - d(v_i)$ é apenas a distância restante de v_i a t.

Além disso, temos que $d(t) - d(v_i) = d(t) - d(v_i - 1) - w(v_i - 1, v_i)$. Colocando-os juntos, a classificação é no máximo $d(t)$, então A* irá expandir esta aresta, definindo $d'(v_i) = d'(vi - 1) + w(v_i - 1, vi) = d(v_i - 1) + w(v_i - 1, v_i) = d(v_i)$, ou descartando se $d'(v_i)$ for igual a $d(v_i)$.

3. Revisão da Literatura

Na literatura, diversas soluções vêm sendo desenvolvidas com o intuito de encontrar rotas com menores distâncias, saindo de um ponto inicial a um ponto de destino. Muitas dessas soluções fazem uso de algoritmos de busca de caminho mínimo, a citar: [Fernandes et al., 2020], [Erke et al., 2020], [Risald et al., 2017].

Em [Fernandes et al., 2020] é apresentado um algoritmo que calcula a distância mais próxima de um ponto de origem à uma farmácia, levando em consideração a disponibilidade do medicamento na determinada farmácia. Neste trabalho o algoritmo de Floyd-Warshall foi utilizado. Os dados da pesquisa foram coletados por meio do *Google Maps* e organizados na forma de uma lista de endereços de farmácias onde estão armazenadas as coordenadas. Além da criação do algoritmo, os autores também propuseram um protótipo para o sistema a fim de oferecer uma maior praticidade ao usuário na tomada de decisão, quando se diz respeito a busca pela farmácia mais próxima e se em tal farmácia o medicamento está disponível.

O trabalho proposto em [Erke et al., 2020] descreve um algoritmo de planejamento de caminho Veículos Terrestres Autônomos (ALVs) baseado no algoritmo A*. Entretanto, durante a aplicação do A* foram constatadas algumas deficiências, dessa forma, os autores propuseram algumas melhorias ao algoritmo, visando: i) otimizar o processo de busca; ii) reduzir o tempo de computação, e iii) aprimorar a realização das comparações entre rotas. Mediante a construção do modelo, os autores conduziram uma série de experimentos com intuito de avaliar o algoritmo, possibilitando a análise da eficácia do algoritmo proposto em comparação com o A* tradicional. Ao final foi constatado que o algoritmo proposto alcançou melhores resultados, sendo mais robusto e estável que o tradicional.

O trabalho proposto em [Risald et al., 2017], descreve um aplicativo com o objetivo de oferecer suporte para o primeiro processo de atendimento a pacientes em estado de emergência. Para isso utilizou-se o algoritmo de Dijkstra na determinação do menor tempo entre cada hospital e o algoritmo de *Floyd-warshall* no cálculo da melhor rota ao hospital mais próximo, levando em conta o estado atual do tráfego. Como resultados alcançados, além de possibilitar a identificação da melhor rota de transferência de pacientes, o modelo desenvolvido também consegue tratar casos em que fatores inesperados ocorrem, tais como: acidentes de trânsito e ruas bloqueadas. Ao final foi constatado que a utilização dos dois algoritmos impactou em um menor tempo de pesquisa na identificação das rotas e tomada de decisão.

Em uma perspectiva semelhante aos estudos citados, este trabalho visa propor a utilização dos algoritmos de *Floyd-Warshall* e A* no cálculo da melhor rota de transferência entre hospitais, levando em conta a melhor distância de um hospital de origem ao hospital de destino, ponderado pela quantidade de leitos vagos de UTI.

4. Modelagem do Problema

Aplicativos de localização como o *Google Maps*[1] podem ajudar na localização de hospitais. Entretanto, quando se trata da localização de hospitais com leitos vagos, existem poucas

[1] https://www.google.com.br/maps

ferramentas que abordem os dois parâmetros. Atualmente, o site Regula RN[2], é responsável por disponibilizar a quantidade de leitos vagos por hospitais, mas não disponibiliza uma rota que possibilite ao paciente identificar um melhor caminho para chegar até ele de forma rápida.

Com este trabalho será possível identificar hospitais com leitos vagos mais próximos, reduzindo assim, o tempo de espera por leitos UTIs, bem como evitar se deslocar a um hospital que não contenha leitos vagos. Em suma, com o auxílio dos algoritmos citados, será possível localizar os hospitais mais próximos com a maior quantidade de leitos vagos.

Os dados da pesquisa foram coletados por meio do *Google Maps* e organizados em uma lista de hospitais e suas respectivas localizações. Para a coleta dos dados de leitos de UTIs, utilizou-se o site Regula RN no mês de Junho de 2021. De acordo com o site, ao todo existem no RN 40 hospitais com leitos, no entanto, para este estudo foi utilizada uma amostra de 10 hospitais. Na Tabela 1 é descrita a localização dos hospitais selecionados para o estudo.

Tabela 1: Localização dos hospitais selecionados para o estudo

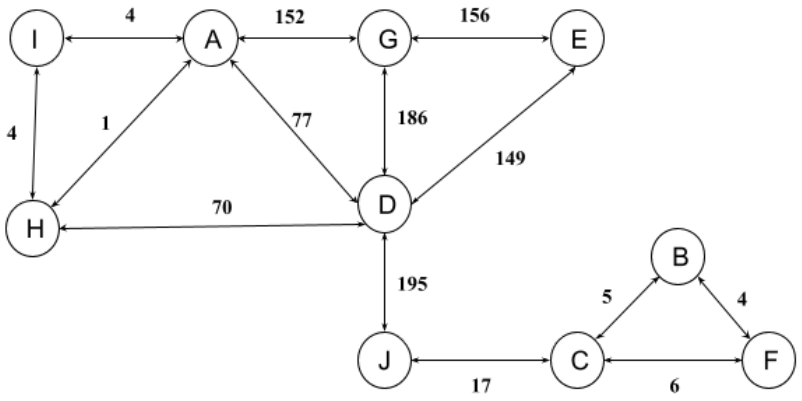
Ponto	**Localização**	**Leitos**
A	Hospital São Luiz, Mossoró - RN	0
B	Hospital Colônia Dr. João Machado, Natal - RN	10
C	Hospital Giselda Trigueiro, Natal - RN	4
D	Hospital Regional Nelson Inácio dos Santos, Assu - RN	1
E	Hospital Regional Telecila Freitas Fontes, Caicó - RN	7
F	Hospital Municipal de Natal, Natal - RN	0
G	Hospital Regional Dr. Cleodon Carlos de Andrade, Pau dos Ferros - RN	0
H	Hospital Regional Dr. Tarcísio de Vasconcelos Maia, Mossoró - RN	1
I	Hospital Rafael Fernandes, Mossoró - RN	0
J	Hospital Regional Alfredo Mesquita, Macaíba - RN	0

Fonte: Autoria Própria

A escolha dessa amostra se deu pela presença da maior quantidade de leitos de UTIs existentes no momento da coleta. A variável utilizada foi a distância, sendo representada, por exemplo, como o intervalo partindo de um nó A (início) ao nó J (destino) em quilômetros (Km). Para um melhor entendimento, na Figura 1 é apresentado o grafo do problema contendo o peso das arestas com base nas distâncias entre todos os hospitais.

[2]O Regula RN é uma plataforma que reúne diversas informações sobre o andamento da COVID-19 nos hospitais e cidades do estado do RN - Link: https://regulacao.saude.rn.gov.br/

Figura 1: Grafo do problema.



Fonte: Autoria Própria

5. Resultados e Discussões

As subseções a seguir descrevem os resultados alcançados com a proposta de solução do problema e apresenta um protótipo com o objetivo de oferecer o serviço ao usuário de forma prática.

5.1. Resolução do Problema

Para obterem-se os resultados foram aplicadas informações a cerca dos dados coletados na Tabela 1. Dessa forma, inicialmente, o grafo elaborado (Figura 1) foi convertido em uma matriz de distâncias, de modo a possibilitar a utilização dos algoritmos. Na Tabela 2 é apresentada a matriz de distância gerada.

Tabela 2: Matriz de distâncias.

	A	B	C	D	E	F	G	H	I	J
A	0	∞	∞	77	∞	∞	152	1	4	∞
B	∞	0	5	∞	∞	4	∞	∞	∞	∞
C	∞	5	0	∞	∞	6	∞	∞	∞	17
D	77	∞	∞	0	149	∞	186	70	∞	195
E	∞	∞	∞	149	0	∞	156	∞	∞	∞
F	∞	4	6	∞	∞	0	∞	∞	∞	∞
G	152	∞	∞	186	156	∞	0	∞	∞	∞
H	1	∞	∞	70	∞	∞	∞	0	4	∞
I	4	∞	∞	∞	∞	∞	∞	4	0	∞
J	∞	∞	17	195	∞	∞	∞	∞	∞	0

Fonte: Autoria Própria

Mediante a geração da matriz de distâncias, o algoritmo de *Floyd-Warshall* foi utilizado com intuito de fornecer uma nova matriz contendo os caminhos mínimos entre os hospitais. Dessa forma, por meio da matriz gerada, dado um hospital de partida, será possível identificar o hospital de destino de menor distância. Na Figura 3 é apresentada a matriz obtida a partir do algoritmo *Floyd-Warshall*.

Com isso, a matriz obtida apenas (Figura 3) ilustra o caminho mínimo entre os hospitais a partir da distância. Entretanto, o hospital de menor distância não garante possuir leitos de UTI dis-

Tabela 3: Matriz de caminho mínimo a partir do algoritmo Floyd-Warshall.

	A	B	C	D	E	F	G	H	I	J
A	0	288	283	71	220	289	152	1	4	266
B	288	0	5	217	366	4	403	287	291	22
C	283	5	0	212	361	6	398	282	286	17
D	71	217	212	0	149	218	186	70	74	195
E	220	366	361	149	0	367	156	219	223	344
F	289	4	6	218	367	0	404	288	292	23
G	152	403	398	186	156	404	0	153	156	381
H	1	287	282	70	219	288	153	0	4	265
I	4	291	286	74	223	292	156	4	0	269
J	266	22	17	195	344	23	381	265	269	0

Fonte: Autoria Própria

poníveis naquele determinado momento. Dessa forma, faz-se necessário realizar uma ponderação levando em consideração a distância necessária e quantidade de leitos.

Para o exemplo do funcionamento pode-se supor que o paciente se encontra hospitalizado no Hospital “E” e deseja se transferir para o hospital mais indicado. Com isso, para descobrir o hospital de partida, a coluna contendo a distância dos demais hospitais até o “E” é selecionada. Além disso, a coluna contendo a quantidade de leitos por hospital é destacada, com intuito de realizar a ponderação entre o caminho mínimo.

Ao final, uma coluna contendo o resultado da ponderação é obtida, possibilitando a exclusão dos hospitais sem leitos e a visualização do hospital mais indicado para a realização da transferência (neste exemplo, o Hospital B). Na Tabela 4 são apresentados os dados analisados e a ponderação obtida para cada um dos hospitais.

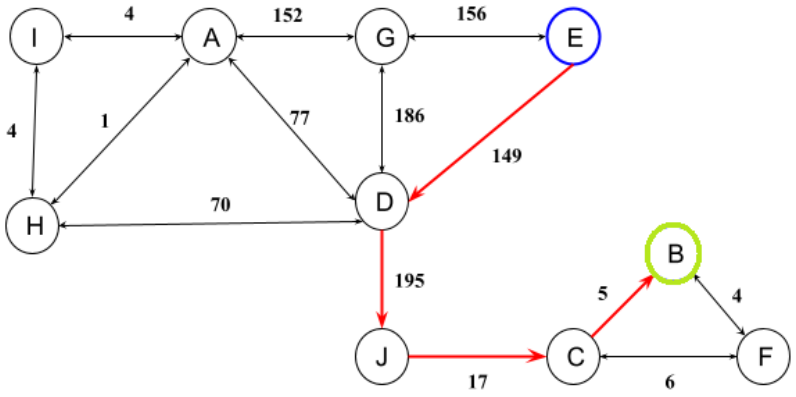
Tabela 4: Identificação do Hospital mais adequado.

Hospital	Distância ao Nó E	Leitos	Ponderação
A	220	0	-
B	366	10	**36,6**
C	361	4	65,25
D	149	1	149
E	0	7	-
F	267	0	-
G	156	0	-
H	219	1	219
I	223	0	-
J	344	0	-

Fonte: Autoria Própria

Com a descoberta do hospital de destino mais indicado, o algoritmo A* foi utilizado para ilustrar a visualização do percurso de menor custo, tendo em vista os possíveis caminhos. Assim, tem-se com o resultado que o melhor caminho a se fazer o trajeto entre o ponto “E” até o ponto “B”: E - D - J - C - B. Na Figura 2 é apresentado o caminho a ser percorrido.

Figura 2: Visualização do caminho com o algoritmo A*.



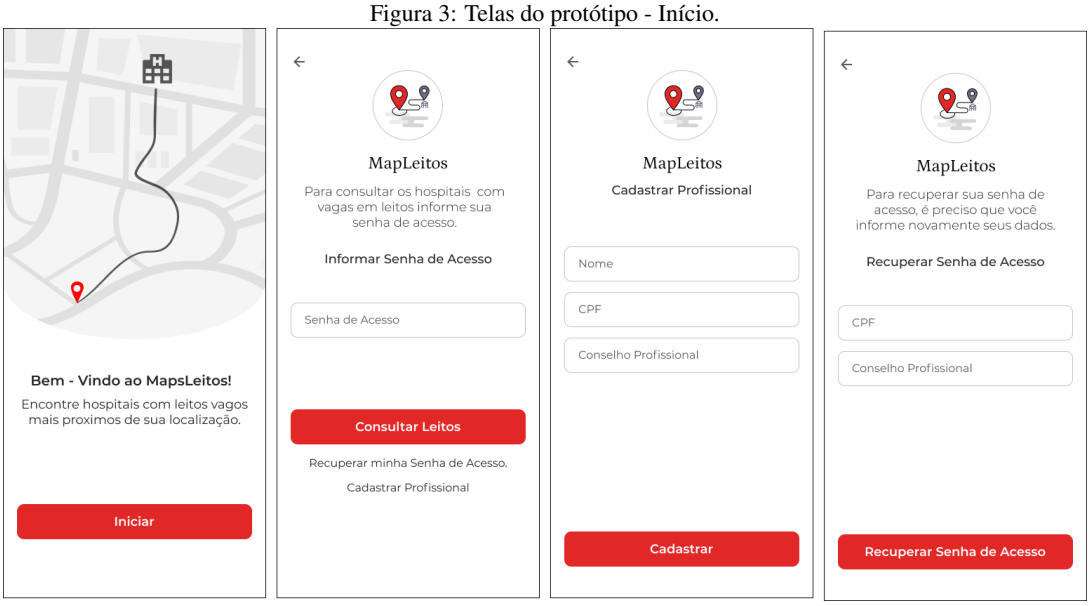
Fonte: Autoria Própria

5.2. MapLeitos: um protótipo para visualização de rotas de leitos de UTIs COVID-19

Além da implementação dos algoritmos para identificar o hospital mais indicado, também foi desenvolvido um protótipo do aplicativo ”MapLeitos”. Para melhor descrever por quem a solução será utilizada, percebeu-se a necessidade da identificação do usuário, que estará em constante interação com as funcionalidades. Nesse contexto, os usuários identificados são profissionais do âmbito da saúde, como: Médicos, Enfermeiras, Gerentes de Hospitais e outros.

Na Figura 3 são apresentadas as telas iniciais do protótipo, destacando a visualização acerca do acesso ao sistema, o cadastro do profissional de saúde, e a possibilidade de recuperação de senha. Na Figura 4 são apresentadas as telas de ativação da localização, visualização das rotas e a tela de visualização do menu do usuário.

Figura 3: Telas do protótipo - Início.



Fonte: Autoria Própria

Figura 4: Telas do protótipo - Visualização de rotas e menu do usuário.



Fonte: Autoria Própria

6. Considerações Finais

Devido ao alto índice de transmissão da COVID-19, inúmeras pessoas acabam necessitando de cuidados especiais nas UTIs. Em meio a essa alta necessidade de internação, tem-se uma baixa quantidade de leitos disponíveis em hospitais, o que torna difícil identificar o hospital mais indicado para realizar a internação, levando em conta a distância e a quantidade de leitos disponíveis. Isso evidencia a importância de desenvolver soluções que auxiliem na tomada de decisões mediante esse cenário de escolher o hospital mais próximo que contenha leitos vagos.

Como resultados deste trabalho, foi possível desenvolver uma solução apoiada nos algoritmos de *Floyd-Warshall* e A* que possibilita a identificação do hospital de destino mais adequado, tendo em vista o número de leitos e a distância necessária de locomoção. Como trabalhos futuros, pretende-se desenvolver a partir do protótipo elaborado, uma aplicação móvel que possibilite a identificação de maneira simples e visual do hospital mais indicado para a transferência de pacientes com a COVID-19.

Referências

Arenales, M., Armentano, V., Morabito, R., e Yanasse, H. (2011). *Pesquisa Operacional.Rio de Janeiro: Editora Campus*.

Cheng, F., Joshi, H., Tandon, P., Freeman, R., Reich, D., Mazumdar, M., Kohli-Seth, R., Levin, M., Timsina, P., e Kia, A. (2020). Using machine learning to predict icu transfer in hospitalized covid-19 patients. *Journal of Clinical Medicine*.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., e Stein, C. (2002). *Algoritmos: teoria e prática*.

Duchoň, F., Babinec, A., Kajan, M., Beňo, P., Florek, M., Fico, T., e Jurišica, L. (2014). Path planning with modified a star algorithm for a mobile robot. *Modelling of Mechanical and Mechatronic Systems*, p. 59–69.

Edelkamp, S. e Schrödl, S. (2012). Basic search algorithms. In *Heuristic Search*, p. 47–87. Elsevier.

Erke, S., Bin, D., Yiming, N., Qi, Z., Liang, X., e Dawei, Z. (2020). An improved a-star based path planning algorithm for autonomous land vehicles. *International Journal of Advanced Robotic Systems*, 17(5).

Feitosa, C. e Cabral, A. C. (2020). Localização de postos para testagem de covid-19. *Anais do LII Simpósio Brasileiro de Pesquisa Operacional*.

Fernandes, V. M. B., Rocha, R. S., Freitas, T. C. M., Liberalino, C. H. P., e Júnior, F. C. L. (2020). Algoritmo de floyd-warshall utilizado no mapeamento das rotas de farmácias. *Anais do LII Simpósio Brasileiro de Pesquisa Operacional*.

Hart, P., Nilsson, N., e Raphael, B. (1968). A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, p. 100–10.

Noronha, K., Guedes, G., Turra, C., Andrade, M., Botega, L., Nogueira, D., Calazans, J., Carvalho, L., Servo, L., e Ferreira, M. (2020). Pandemia por covid-19 no brasil: análise da demanda e da oferta de leitos hospitalares e equipamentos de ventilação assistida segundo diferentes cenários. *Cadernos de Saúde Pública*, 36(6).

Risald, Mirino, A. E., e Suyoto (2017). Best routes selection using dijkstra and floyd-warshall algorithm. In *2017 11th International Conference on Information Communication Technology and System (ICTS)*, p. 155–158.

Souza, G. F. G., Aguilar, S., Baião, F. A., e Maçaira, P. M. (2021). Estudo de similaridade da evolução da covid-19 nos estados brasileiros. *Anais do LIII Simpósio Brasileiro de Pesquisa Operacional*.

WHO (2020). Coronavirus. https://www.who.int/health-topics/coronavirus#tab=tab_1.