

Fabio Kauê Araujo da Silva - 16311045

Gabriel Inumaru Esteves- 15453487

Kainã Alves Tureso - 15466391

Matheus Spinelli de Paiva - 14598682

Pedro Luís Anghievisck - 15656521

Aplicação do problema da p-mediana na distribuição de hospitais na cidade de São Carlos - SP

São Carlos - SP

24 de novembro de 2025

Sumário

Sumário	1
1 INTRODUÇÃO	2
2 OBJETIVOS	3
3 REVISÃO BIBLIOGRÁFICA	4
3.1 Grafos	4
3.1.1 Algoritmos de distância mínima	4
3.2 Problema da p-mediana	4
3.2.1 p-mediana como problema de otimização linear	5
3.3 Difusão de dados geográficos	6
4 PROCEDIMENTOS E MÉTODOS	8
4.1 Modelagem Matemática	8
4.2 Coleta e Tratamento de Dados	8
4.3 Implementação Computacional	10
4.3.1 Estratégias de Resolução	11
4.3.2 Roteiro de Execução e Dados	11
5 RESULTADOS	13
5.1 Análise e Desempenho	13
5.2 Discussão	17
6 CONCLUSÃO	18
Referências	19

1 Introdução

A fim de proporcionar um melhor atendimento à população de uma determinada cidade, em um determinado serviço, é crucial que o acesso geográfico seja, em alguma métrica, igualitário entre os indivíduos. Uma dessas formas de se medir esse acesso é através da distância entre os indivíduos e os pontos de atendimento. Assim, o problema da p -mediana (pMP) busca, dado um conjunto de pontos (indivíduos) e um número p , selecionar p pontos (locais de atendimento) de forma a minimizar a soma das distâncias entre cada ponto e o ponto mais próximo selecionado. Uma das aplicações do pMP é na localização de hospitais em uma cidade, onde o objetivo é minimizar a distância total que os habitantes da cidade precisam percorrer para chegar ao hospital mais próximo. Neste relatório, exploramos a aplicação do problema da p -mediana na distribuição de hospitais na cidade de São Carlos - SP. Utilizando dados geográficos reais, modelamos o problema como um problema de otimização linear e implementamos uma solução computacional utilizando o método Simplex primal.

2 Objetivos

O objetivo deste trabalho é comparar a disposição real de hospitais na cidade de São Carlos - SP com resultados obtidos através da aplicação do *pMP* como um problema de otimização linear, além de sugerir possíveis novas localizações ótimas (no sentido de minimizar distâncias) para a construção desses estabelecimentos na cidade.

3 Revisão Bibliográfica

3.1 Grafos

As seguintes definições são dadas em ([Aurichi, s.d.](#)):

3.1.1 Definição: Um **grafo** é o par $G = (V, E)$ em que V é um conjunto de pontos denotado por **vértices** e $E \subset V^2$ é o conjunto de **arestas**.

Informalmente, pode-se definir um grafo como um conjunto de pontos ligados de alguma forma. Um exemplo, que será usado neste estudo, é a malha viária de uma cidade, na qual as ruas representam as arestas e os pontos são as esquinas. Neste caso, cada rua tem um tamanho, o que torna o grafo **ponderado**.

3.1.2 Definição: Dizemos que um grafo $G = (V, E)$ não vazio é um **caminho** se é da forma $V = \{x_1, x_2, \dots, x_n\}$ e $E = \{(x_1, x_2), (x_2, x_3), \dots, (x_{n-1}, x_n)\}$

As duas definições seguintes serão usadas posteriormente na [Subseção 3.2.1](#) para resultados sobre a possibilidade de transformar o *pMP* em um problema linear.

3.1.3 Definição: Dizemos que um grafo $G = (V, E)$ é **conexo** se dados dois vértices $u, v \in V$, existe um caminho $A = (P, C)$ tal que $u, v \in P$.

3.1.4 Definição: Dizemos que um grafo é um **ciclo** se é um caminho e suas extremidades são iguais.

3.1.1 Algoritmos de distância mínima

Na [Seção 4.3](#), será necessário avaliar a distância mínima entre cada par de nós de um grafo ponderado com valores não negativos. Dentre as formas de computar essas distâncias, dois algoritmos principais são considerados: o algoritmo de Dijkstra e o algoritmo de Floyd-Warshall. O algoritmo de Dijkstra calcula, dado um vértice, a distância mínima entre todos os outros vértices de um grafo com complexidade (a depender da implementação) de $O((|V| + |E|) \log |V|)$, de acordo com ([Roughgarden, 2018](#), p. 106). Note que, para grafos esparsos, ou seja, $|E| = O(|V|)$, podemos aproximar a complexidade para $O(|E| \log |V|)$. Nesse caso, para calcular a distância entre todos os vértices de um grafo, temos que aplicar o algoritmo de Dijkstra $|V|$ vezes, o que resulta em uma complexidade de $O(|V||E| \log |V|) \approx O(|V|^2 \log |V|)$. O algoritmo de Floyd-Warshall calcula a distância mínima entre todos os pares de vértices de um grafo, com complexidade $O(|V|^3)$, de acordo com ([Cormen et al., 2022](#)).

3.2 Problema da p-mediana

O problema da p-mediana (*pMP*) é um problema de otimização inteira em que o objetivo é encontrar a localização de p instalações em uma rede/grafo, tal que o custo

total seja reduzido (Daskin, 2013). O custo em um nó $i \in I$ é dado pelo produto da demanda em i e a distância entre o nó i e a instalação mais próxima de i . Assim, podemos formular o problema formalmente:

Seja I um conjunto de nós num grafo e $J \subseteq I$ o conjunto dos nós candidatos a receber uma instalação. **Entradas**

- h_i = demanda no nó $i \in I$
- d_{ij} = distância entre o nó de demanda $i \in I$ e o local candidato $j \in J$
- p = número de instalações a serem construídas

Variáveis de decisão

- $X_j = \begin{cases} 1, & \text{se colocarmos uma instalação no local } j \in J \\ 0, & \text{caso contrário} \end{cases}$
- $Y_{ij} = \begin{cases} 1, & \text{se o nó } i \in I \text{ é servido pela instalação no nó } j \in J \\ 0, & \text{caso contrário} \end{cases}$

Formulamos então o problema:

Problema 3.2.1

$$\text{Minimizar} \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij} \quad (3.1)$$

$$\text{Sujeito a} \sum_{j \in J} Y_{ij} = 1 \quad \forall i \in I \quad (3.2)$$

$$\sum_{j \in J} X_j = p \quad (3.3)$$

$$Y_{ij} - X_j \leq 0 \quad \forall i \in I; j \in J \quad (3.4)$$

$$X_j \in \{0, 1\} \quad \forall j \in J \quad (3.5)$$

$$Y_{ij} \in \{0, 1\} \quad \forall i \in I; j \in J \quad (3.6)$$

A função objetivo (3.1) minimiza as distâncias ponderadas pela demanda de cada nó em relação à instalação mais próxima. A restrição (3.2) diz que cada nó só pode estar designado a uma instalação e a (3.3) restringe o número de instalações a ser exatamente p . A restrição (3.4) expressa a necessidade de interligar as variáveis de localização (X_j) com as variáveis de alocação (Y_{ij}). Ela diz que se um nó i foi designado a um local j , então, necessariamente, j tem que ter uma instalação, ou seja, $X_j = 1$. As restrições (3.5) e (3.6) são as condições de integralidade.

3.2.1 p-mediana como problema de otimização linear

Note que o problema 3.2.1 é NP-difícil (Daskin, 2013, p. 241), ou seja, computá-lo é extremamente difícil. Uma das formas para tentar reduzir o tempo necessário para resolver o problema, é considerar o processo de relaxação linear, em que as restrições de integralidade são completamente retiradas ou substituídas por restrições que limitam

as variáveis a um intervalo. Com esse processo, criamos um problema linear similar, que é computável em tempo polinomial. Em particular, se os conjuntos iniciais das restrições de integralidade estão contidos nos novos intervalos de restrição, então existem mais valores possíveis no problema linear do que no problema inteiro. Logo, o problema linear nos dá uma cota para o valor ótimo do problema inteiro. Além disso, se a tradução linear tem um resultado que satisfaz as condições de integralidade antigas, então essa solução também resolve o problema inteiro. Para o *pMP*, podemos substituir as restrições de integralidade (3.5) e (3.6) por:

$$0 \leq X_j \leq 1 \quad \forall j \in J \quad (3.7)$$

$$0 \leq Y_{ij} \leq 1 \quad \forall i \in I; j \in J \quad (3.8)$$

Para resultados não inteiros, pode-se interpretar a associação das variáveis de decisão como uma medida do quanto bom é fazer uma instalação em determinado local. Com isso, temos o problema linear:

Problema 3.2.2

$$\begin{aligned} &\text{Minimizar} \sum_{i \in I} \sum_{j \in J} h_i d_{ij} Y_{ij} \\ &\text{Sujeito a} \sum_{j \in J} Y_{ij} = 1 \quad \forall i \in I \\ &\quad \sum_{j \in J} X_j = p \\ &\quad Y_{ij} - X_j \leq 0 \quad \forall i \in I; j \in J \\ &\quad 0 \leq X_j \leq 1 \quad \forall j \in J \\ &\quad 0 \leq Y_{ij} \leq 1 \quad \forall i \in I; j \in J \end{aligned}$$

Como os únicos inteiros do intervalo que restringem os Y_{ij} e X_j são o 0 e o 1, a solução do problema linear equivale à do problema inteiro quando ela é inteira. Uma condição para integralidade no *pMP* é dado em ([Baïou; Barahona, 2011](#)), no seguinte teorema:

3.2.1 Teorema: *Seja G um grafo não direcionado conexo. Então o poliedro do problema linear é integral para todo $p \in \mathbb{N}$ (os vértices do poliedro possuem todas as entradas inteiras) se e somente se G é um caminho ou um ciclo simples.*

Em geral, as redes nas quais o *pMP* é proposto são mais complexas do que caminhos ou ciclos. Logo, só é possível saber para quais p o relaxamento dará a solução ótima após computá-la.

3.3 Difusão de dados geográficos

Em geral, dados populacionais coletados em pesquisas são compilados em parcelas do território. Nas pesquisas do IBGE por exemplo, os setores censitários são a menor unidade territorial de coleta e divulgação de dados. Logo, esses setores representam uma parte da população que mora naquela região, mas não indicam a localização exata

de cada indivíduo. Para o *pMP*, é interessante que os dados sejam mais diluídos, ou seja, que cada indivíduo tenha uma localização específica. Assim, é possível calcular distâncias mais precisas entre os indivíduos e os pontos de instalação. Uma forma de fazer isso é através da difusão dos dados geográficos, que consiste em distribuir os indivíduos de um setor censitário para pontos nele ou próximos a ele. Uma das formas de se fazer isso, que será usada posteriormente, é utilizar uma discretização da equação do calor, dada por:

$$\frac{\partial u}{\partial t} = -\eta \nabla^2 u \quad (3.9)$$

Em que $u(x, t)$ é a densidade de indivíduos no ponto x no tempo t e η é uma constante de difusão. A equação do calor modela o processo de difusão de calor em um meio, mas pode ser adaptada para modelar a difusão de indivíduos em um espaço geográfico. Com essa equação, utilizamos o método das diferenças finitas para obter:

$$\frac{u_{n+1} - u_n}{\Delta t} = -\eta \nabla^2 u_n \quad (3.10)$$

Logo, temos que:

$$u_{n+1} = u_n - \eta \Delta t \nabla^2 u_n \quad (3.11)$$

Para grafos, o operador laplaciano ∇^2 é substituído pelo laplaciano discreto, que é dado por $L = D - A$, em que D é a matriz diagonal dos graus dos vértices e A é a matriz de adjacência do grafo. Além disso, u_k , neste caso, representa um vetor com a quantidade de entrada igual ao número de nós. Assim, a equação discreta da difusão em grafos é dada por:

$$u_{n+1} = u_n - \eta \Delta t L u_n \quad (3.12)$$

Tomando $\alpha = \eta \Delta t$, considerando u_n^k a k-ésima entrada do vetor u_n e L_k a k-ésima linha da matriz laplaciana, temos:

$$u_{n+1}^k = u_n^k - \alpha L_k^\top u_n \quad (3.13)$$

Com algumas manipulações simples, chegamos em:

$$u_{n+1}^k = (1 - \alpha |N(k)|) u_n^k + A_k^\top u_n \quad (3.14)$$

Em que $N(k)$ é o conjunto de vizinhos do nó k . Note que, nessa forma, a quantidade de indivíduos em um nó é atualizada com base na quantidade de vizinhos de um determinado nó. Dessa forma, é mais conveniente utilizar a Laplaciana de *random-walk*, dada por $L_{rw} = I - D^{-1}A$ ([Lu, 2014](#)). Com essa matriz, a equação de difusão fica:

$$u_{n+1}^k = (1 - \alpha) u_n^k + \alpha (D^{-1} A)_k^\top u_n \quad (3.15)$$

Com isso, a dissipação em cada nó é sempre uma porcentagem constante da quantidade de indivíduos nele, o que é mais intuitivo e prático. Chamaremos de fator de retenção o valor $1 - \alpha$.

4 Procedimentos e métodos

4.1 Modelagem Matemática

Modelamos o problema da localização de hospitais na cidade de São Carlos - SP como um pMP relaxado linearmente, conforme pode ser visto em [3.2.2](#). Para isso, foi considerado o grafo da malha viária da cidade, onde os nós representam as interseções entre as ruas e as arestas representam os trechos de rua entre essas interseções. Para cada nó foi atribuído um valor de população aproximado, que representa a demanda por serviços hospitalares naquele ponto. As distâncias entre os nós foram calculadas com base no comprimento dos trechos de rua que os conectam. Para os locais candidatos à instalação foram considerados apenas os nós que possuem uma demanda maior que p_0 , com o intuito de reduzir o custo computacional do problema. Na notação do pMP anteriormente definida, temos que:

- I = conjunto de todos os nós do grafo (interseções da malha viária)
- h_i = população aproximada no nó $i \in I$
- $J = \{i \in I : h_i \geq p_0\}$
- d_{ij} = distância entre o nó de demanda $i \in I$ e o local candidato $j \in J$.
- p = número de hospitais a serem localizados

4.2 Coleta e Tratamento de Dados

O grafo da cidade de São Carlos - SP foi obtido através do OpenStreetMap (OSM) utilizando a biblioteca OSMnx ([Boeing, 2017](#)) (ver [Figura 1](#) e [Figura 2](#)). A população aproximada em cada nó foi estimada com base nos dados do censo demográfico do IBGE de 2022 ([Instituto Brasileiro de Geografia e Estatística IBGE, 2022](#)), estimados nos setores censitários ([Figura 3](#)), em que foi utilizado o método de difusão de dados geográficos descrito na [Seção 3.2](#). Em suma, a população de cada setor foi concentrada em um único nó representativo do setor, e então a difusão foi aplicada para distribuir essa população pelos nós vizinhos na malha viária. Esse processo foi repetido por três iterações, com fator de retenção de 0.4, resultando em uma distribuição mais realista da população ao longo da malha viária da cidade.

Malha Viária de São Carlos



Figura 1 – Grafo da malha viária de São Carlos (Autoria própria)



Figura 2 – Grafo da malha viária de São Carlos, detalhe para o centro urbano (Autoria própria)

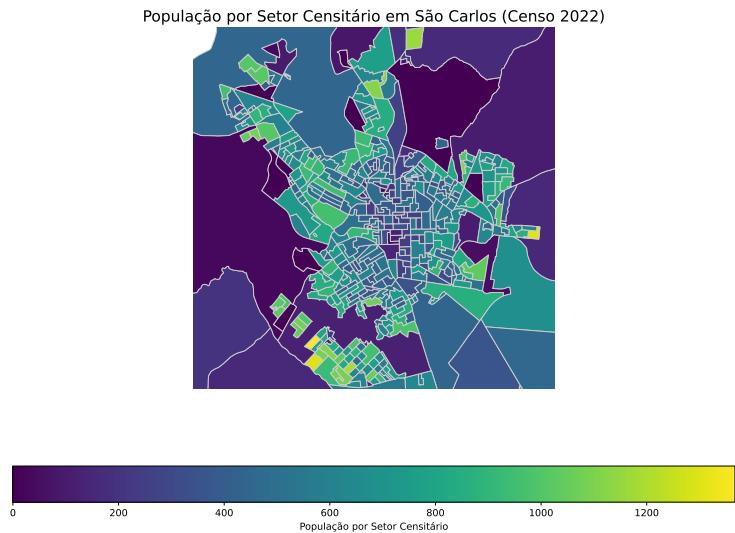


Figura 3 – Setores censitários de São Carlos (Autoria própria)

4.3 Implementação Computacional

A solução proposta foi implementada na linguagem de programação Python, com auxílio das LLMs Gemini ([Google LLC, 2025](#)) e ChatGPT ([OpenAI, 2025](#)), utilizando um conjunto de bibliotecas especializadas em análise geoespacial e otimização matemática. O fluxo de trabalho foi dividido em cinco módulos sequenciais, permitindo a modularização do tratamento de dados e da resolução do problema.

Para a manipulação e visualização de dados geográficos, utilizou-se as bibliotecas OSMnx e GeoPandas, que permitem a extração da malha viária diretamente da plataforma OpenStreetMap e a manipulação de arquivos *shapefile*. Para a construção da matriz de distâncias e operações em grafos, utiliza-se a biblioteca NetworkX, aplicando o algoritmo de Dijkstra para o cálculo de caminhos mínimos; pontua-se a tentativa de utilização do algoritmo de Floyd-Warshall, mas o mesmo mostrou-se muito ineficaz para este problema.

A resolução do problema de otimização linear (conforme modelado na [Subseção 3.2.1](#)) foi realizada por meio da biblioteca PuLP, com o método Simplex Dual que foi escolhido por apresentar desempenho superior ao Simplex Primal no nosso modelo, que possui muitas restrições (uma para cada par de candidato). Como mostrado por ([Bixby, 2002](#)), implementações modernas do Simplex Dual tendem a superar o Primal em problemas grandes, altamente restritos e derivados de relaxações lineares, exatamente como o pMP. Na prática, o Simplex Primal mostrou-se muito mais lento, enquanto o Simplex Dual obteve soluções em tempo hábil.

4.3.1 Estratégias de Resolução

Foram desenvolvidas duas abordagens para a resolução computacional do problema da p-mediana:

- **Abordagem Completa com Filtro de Candidatos (5_final.py):** Nesta abordagem, todos os nós da malha viária atuam como pontos de demanda. No entanto, para reduzir o espaço de busca das variáveis de decisão X_j (locais de instalação), aplica-se um filtro onde apenas nós com uma população mínima preestabelecida (ex: 200 habitantes) são considerados candidatos a receber um hospital.
- **Abordagem Simplificada (5.1_final_simp.py):** Visando eficiência computacional, esta estratégia reduz o grafo original apenas aos nós que possuem população estritamente positiva antes mesmo do processo de difusão. Estes nós atuam simultaneamente como pontos de demanda e únicos locais candidatos. Isso reduz drasticamente a dimensão da matriz de distâncias e o número de restrições do modelo linear.

4.3.2 Roteiro de Execução e Dados

Para a reprodução dos resultados e garantia de eficiência no teste de múltiplos cenários, o fluxo de execução foi dividido em cinco etapas sequenciais:

1. **Extração do Grafo (1_grafo.py):** Executar este script para baixar a malha viária de São Carlos via OSMnx, projetá-la para coordenadas métricas (UTM) e salvá-la em formato GraphML.
2. **Integração Censitária (2_densidade.py):**
 - **Pré-requisito:** É necessário realizar o download dos dados do Censo 2022 (Malha de Setores Censitários em .shp e Agregados por Setores em .csv) nos seguintes sites do IBGE: <https://www.ibge.gov.br/estatisticas/downloads-estatisticas.html> e https://geoftp.ibge.gov.br/organizacao_d_o_territorio/malhas_territoriais/malhas_de_setores_censitarios_divisoes_intramunicipais/censo_2022/setores/shp/UF.
 - O script cruza os dados geométricos com a tabela de população e projeta os centroides para a malha viária.
3. **Difusão Populacional (3_difusao.py):** Aplica o algoritmo de suavização como descrito na [Seção 3.3](#) para distribuir a população dos nós para seus vizinhos. Com isso, obtemos a população suavizada no grafo, como pode ser visto na [Figura 4](#), além dos pontos em que $p \geq 200$, ou seja, os pontos passíveis de construir um hospital.

Comparativo da Densidade Populacional de São Carlos (populacoes_suavizadas.pkl)

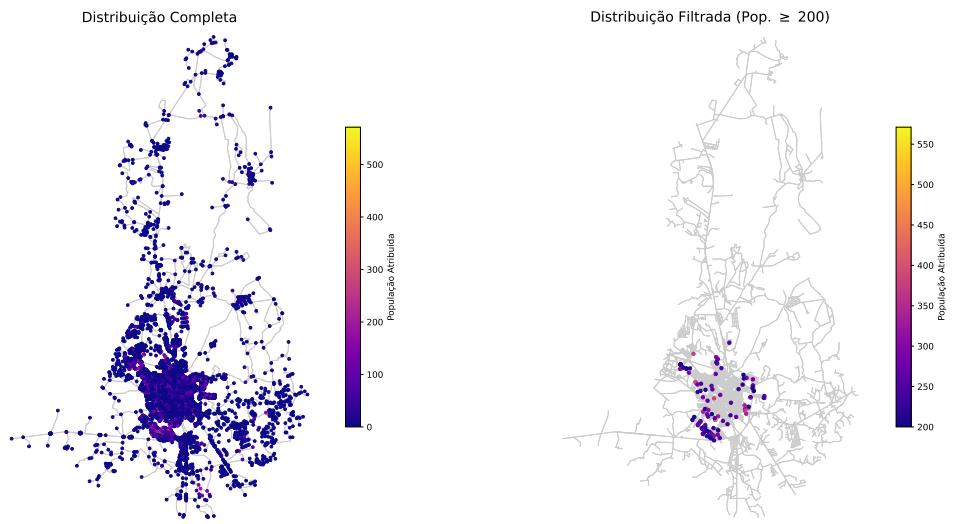


Figura 4 – População suavizada na malha viária de São Carlos (Autoria própria)

4. **Pré-cálculo de Distâncias (4_distancias.py):** Dada a complexidade computacional do cálculo de caminhos mínimos em grafos urbanos, este script foi criado para executar o algoritmo de Dijkstra para todos os pares de nós ($O(V \cdot E \log V)$) uma única vez. O resultado é uma matriz de distâncias persistida em disco (arquivo .pkl), permitindo que as otimizações subsequentes sejam executadas rapidamente sem recalcular rotas, também vale ressaltar que nosso grafo foi transformado em um grafo não-direcionado, simplificando muito o cálculo de caminhos entre dois pontos.
5. **Otimização (5_final.py ou 5.1_final_simp.py):** Carrega o grafo, a população e a matriz de distâncias pré-calculada. Resolve o problema linear via Simplex Dual e exporta os mapas e tabelas de resultados, permitindo testar diferentes valores de p (número de hospitais).

Para a execução completa do fluxo, é recomendável utilizar uma máquina com pelo menos 8 GB de RAM.

5 Resultados

5.1 Análise e Desempenho

Com fins de realizar uma boa análise, executamos o código `5_final.py` variando o número de hospitais desejados, com o intuito de ver as semelhanças e diferenças entre cada caso. Para analisar o quanto bom é cada caso, usaremos o seguinte critério:

- **Acerto:** Quando o local proposto estiver suficientemente próximo a um hospital real;
- **Semi-Acerto:** Quando o local proposto está na mesma região de um hospital mas não está proximo o suficiente para ser considerado um acerto;
- **Erro:** Quando não há nenhum hospital proximo o suficiente. Note que casos assim indicam uma sugestão de construção futura de hospitais, visto que teoricamente há necessidade de um hospital nessa região;

Com o intuito de permitir a classificação de nossos resultados, o seguinte mapeamento dos hospitais de São Carlos foi usado para as seguintes análises:

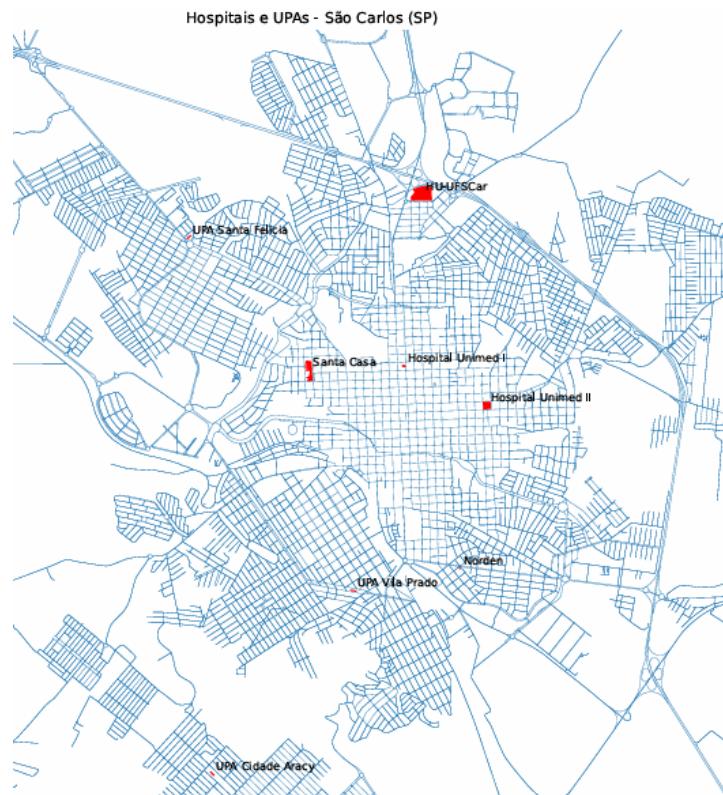


Figura 5 – Mapa dos Hospitais de São Carlos (Autoria própria)

Portanto, temos então os 4 casos, sendo estes:

1. **Caso $p = 5$:** Para acharmos os locais ideias de 5 hospitais, foi necessário 2388.86 segundos, sendo o segundo caso mais rápido. Aqui obtivemos 40% de acertos e 60% de semiacertos, sendo que todos os acertos foram nas zonas mais densas da cidade e os semiacertos concentrados no norte, note também que como temos um número muito pequeno de hospitais desejados, a região central da cidade, que é uma zona comercial, acabou ficando sem hospitais, devido a sua baixa densidade populacional e o fato dos outros hospitais estarem distribuídos de forma que a distância do centro até o hospital mais próximo não seja muito grande.

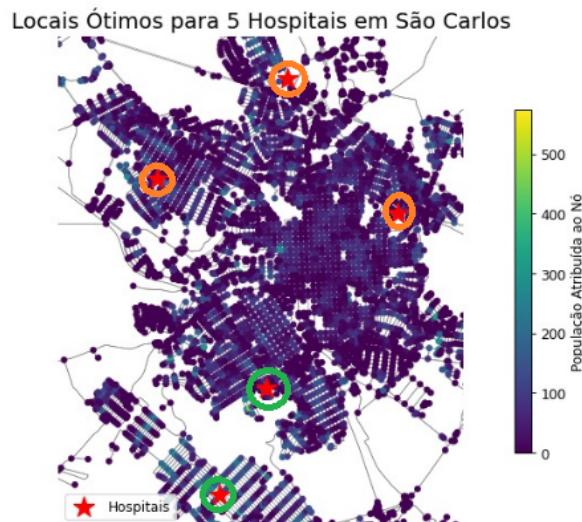


Figura 6 – Resultado gerado pelo arquivo `5_final.py` para $p = 5$

2. **Caso $p = 7$:** Nesse caso, tivemos uma performance surpreendentemente boa, sendo necessário apenas 1384.23 segundos. Temos como hipótese que, devido à distribuição dos pontos candidatos visto em [Figura 4](#) é provável que teremos 7 núcleos, o que reduz em muito o número de iterações do método. Obtivemos 3 acertos, 2 erros e 2 semiacertos. Note que, do caso 5 para o caso 7 ganhamos um acerto a mais na região centro-sul da cidade, o que reforça cada vez o impacto da distribuição demográfica da cidade.

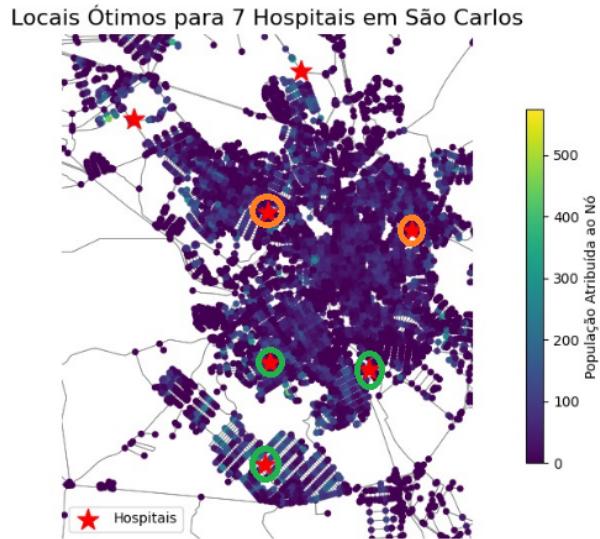


Figura 7 – Resultado gerado pelo arquivo `5_final.py` para $p = 7$

3. **Caso $p = 8$:** São Carlos atualmente tem 8 hospitais, então com o intuito de descobrir o quanto efetivo é nosso modelo, foi necessário 8836.57 segundos. Como é possível ver na imagem abaixo, obtivemos 50% de acertos, 12.5% de semiacertos e 37.5% de erros, sendo este um bom resultado, tendo em vista as diversas adaptações necessárias para a simplificação do problema que deve acabar interferindo no resultado final. Sendo uma dessas adaptações, mostrada na [Figura 4](#), os locais candidatos a serem hospitais estão quase que concentrados na região sudoeste da cidade, perto do Aracy, onde conseguimos ver uma grande taxa de acerto na maioria dos casos analisados, o que sugere que de fato, muitos hospitais em São Carlos foram localizados com a intenção de reduzir distâncias entre o povo e os hospitais. Porém, devido a essa concentração demográfica, o nosso modelo acaba ficando um pouco mais falho quanto mais perto do centro e norte da cidade estamos, o que acaba sendo resolvido em parte ao aumentar o número de hospitais, pois uma vez que a necessidade da densa região do sudeste é saciada, a atenção acaba se voltando justamente ao noroeste e centro-oeste de São Carlos, as regiões com menos taxa de acertos.

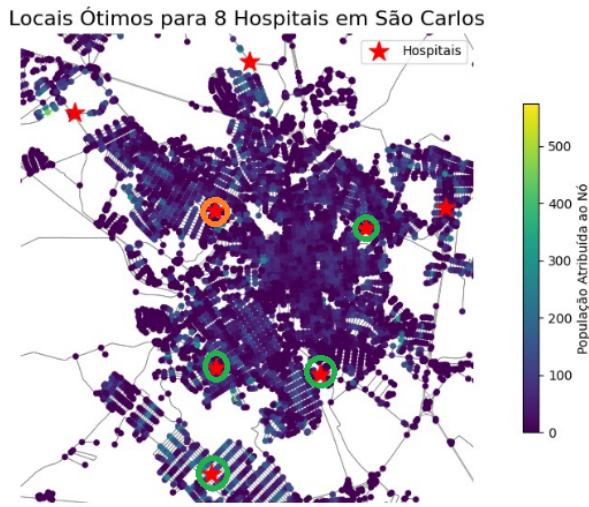


Figura 8 – Resultado gerado pelo arquivo 5_final.py para $p = 8$

4. Caso $p = 9$: Nossa caso com maior número de acertos, foi necessário 3958.15 segundos para o código terminar de rodar e é notável como o aumento do número de hospitais desejados influencia na visibilidade do centro da cidade. Também é interessante, novamente notar que a região perto do Aracy manteve, em todas as iterações realizadas, pelo menos 2 acertos, o que evidencia novamente a alta concentração populacional dessa área.

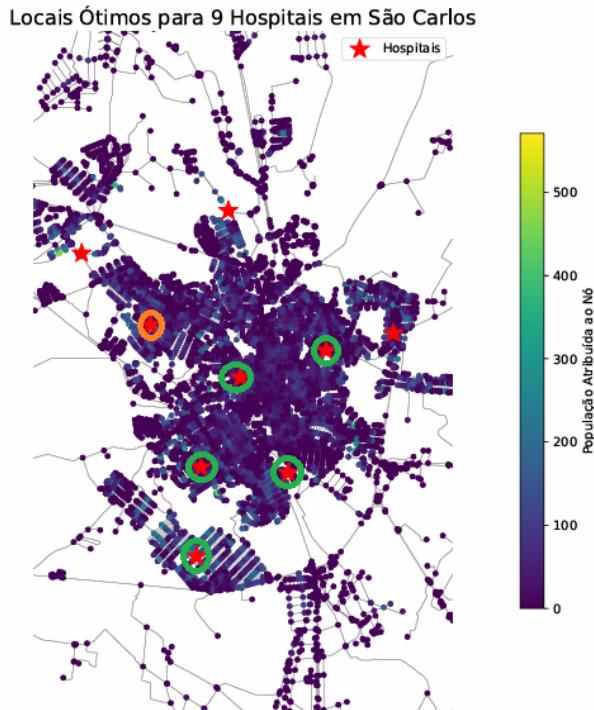


Figura 9 – Resultado gerado pelo arquivo 5_final.py para $p = 9$

Por fim é possível afirmar que esta solução, apesar de não ter uma precisão perfeita produz resultados muito bons e serve também para indicar quais áreas dentro da cidade de São Carlos demandam por hospitais. Seguindo a tendência desse resultados,

pode-se ver que as áreas mais ao norte da cidade não são bem tão bem representadas com relação à presença de hospitais, o que sugere que a construção de novos hospitais nessa região poderia ser benéfica para a população local.

É evidente que dado um p pequeno, acabaremos com resultados menos precisos porém que reforçam a importância de um hospital na região, por isso que no caso de $p = 5$ visto na [Figura 6](#) não há nenhum Erro, mas sim acertos e semiacertos, o que mostra que nossa solução está condizente com a realidade além de se manter consistente para diferentes valores de p .

5.2 Discussão

É importante ressaltar que, apesar dos bons resultados obtidos, o modelo apresenta algumas limitações que podem impactar a precisão das soluções propostas. A simplificação do grafo para um modelo não direcionado, embora tenha facilitado o cálculo das distâncias, pode ter introduzido imprecisões, especialmente em áreas onde o tráfego é unidirecional ou onde existem barreiras físicas que afetam o acesso aos hospitais. Além disso, a escolha do limiar populacional p_0 para definir os nós candidatos a receber hospitais pode ter excluído locais que, apesar de terem uma população menor, são estrategicamente importantes devido à sua localização geográfica ou acessibilidade.

Note que, também, os dados do IBGE mostram a população residente, mas não consideram fatores como o fluxo diário de pessoas que podem necessitar de serviços hospitalares, como trabalhadores e estudantes que se deslocam para a cidade. Esses fatores podem influenciar significativamente a demanda por serviços de saúde em determinadas áreas, como no centro, que não apresenta quase nenhum residente, mas é uma das áreas com mais pessoas diariamente. Além disso, o modelo não leva em conta a capacidade dos hospitais existentes, o que pode ser um fator crucial na determinação da necessidade de novas instalações. Hospitais com alta demanda podem justificar a construção de novos hospitais nas proximidades, mesmo que a população local não seja tão alta.

O resultado mais expressivo do modelo foi observado no caso com $p = 9$, que corresponde a um número maior de hospitais do que o atualmente existente na cidade. Nesse cenário, o modelo conseguiu identificar locais que não apenas minimizam as distâncias para a população, mas também sugerem áreas que poderiam se beneficiar de novas instalações hospitalares. Isso indica que o modelo pode ser uma ferramenta útil para planejamento urbano e alocação de recursos de saúde, especialmente em cidades em crescimento ou com distribuição populacional desigual.

6 Conclusão

Em suma, foi possível reformular o pMP, um problema de otimização combinatória, para um problema de otimização linear utilizando o método da relaxação. Os dados foram retirados do OpenStreetMap e do IBGE, possibilitando a modelagem do problema na escala municipal. Com isso, foi aplicado o Simplex Dual para a resolução, implementado na biblioteca PuLP do Python.

A relaxação para o problema da p-mediana se mostrou como uma boa alternativa, considerando a alocação de hospitais em São Carlos, para reduzir a complexidade do problema. Porém, ainda foi necessário lidar com algo computacionalmente pesado, considerando que havia uma grande quantidade de vértices no grafo da cidade e diversos parâmetros e variáveis no problema relaxado.

Algumas simplificações ,como tornar o grafo da cidade para não direcionado e reduzir a quantidade de nós candidatos para receber um hospital, foram feitas para ser possível computar soluções em tempo hábil para computadores domésticos e usando a linguagem Python. Apesar disso, os resultados obtidos sobre a localização de hospitais apresentam grande semelhanças com as localizações reais dessas construções na cidade.

Referências

AURICHI, Leandro. **Grafos**. [S. l.: s. n.].

<https://sites.icmc.usp.br/aurichi/wikimat/doku.php?id=grafos:grafos>.

Acesso em: 15 nov. 2025.

BAÏOU, Mourad; BARAHONA, Francisco. On the linear relaxation of the p-median problem. **Discrete Optimization**, v. 8, n. 2, p. 344–375, 2011. ISSN 1572-5286. DOI:

<https://doi.org/10.1016/j.disopt.2010.12.002>. Disponível em:

<https://www.sciencedirect.com/science/article/pii/S1572528610000733> .

BIXBY, Robert E. Solving real-world linear programs: A decade and more of progress. **Operations Research**, INFORMS, v. 50, n. 1, p. 3–15, 2002. DOI:

[10.1287/opre.50.1.3.17780](https://doi.org/10.1287/opre.50.1.3.17780).

BOEING, Geoff. OSMnx: New methods for acquiring, constructing, analyzing, and visualizing complex street networks. **Computers, Environment and Urban Systems**, v. 65, p. 126–139, 2017. DOI: [10.1016/j.compenvurbsys.2017.05.004](https://doi.org/10.1016/j.compenvurbsys.2017.05.004).

CORMEN, Thomas H. *et al.* **Introduction to Algorithms**. 4. ed. Cambridge, MA: MIT Press, 2022. ISBN 9780262046305.

DASKIN, Mark S. **Network and Discrete Location: Models, Algorithms, and Applications**. 2nd. Hoboken, NJ: John Wiley & Sons, 2013. p. 544. ISBN 9780470905364.

GOOGLE LLC. **Gemini**. [S. l.: s. n.], 2025. <https://gemini.google.com>. Acesso em: 15 nov. 2025.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATÍSTICA IBGE. **Malhas de setores censitários (2022) por Unidade da Federação**. Acesso em: 18 nov. 2025. 2022. Disponível em: https://geoftp.ibge.gov.br/organizacao_do_territorio/malhas_territoriais/malhas_de_setores_censitarios__divisoes_intramunicipais/censo_2022/setores/shp/UF .

LU, Linyuan. **Laplacian and RandomWalks on Graphs**. [S. l.: s. n.], 2014.

https://people.math.sc.edu/lu/talks/nankai_2014/spec_nankai_2.pdf. Palestra em Nankai, acesso em: 17 nov. 2025.

OPENAI. **ChatGPT**. [S. l.: s. n.], 2025. <https://chat.openai.com>. Acesso em: 15 nov. 2025.

ROUGHGARDEN, Tim. **Algorithms Illuminated (Part 2): Graph Algorithms and Data Structures**. [S. l.]: Soundlikeyourself Publishing, 2018.