

Projeto de uma Rede de Baixo Custo para Monitoramento da Qualidade do Ar em Ambientes Urbanos: Arquitetura IoT e Planejamento da Implantação

Gabriel L. R. Dourado *Gustavo M. Pechta*
Henrique J. S. M. Filho *Henrique P. de Souza*
Mateus de L. Almeida *Vinicius P. M. Miguel*
Juliana F. Borin

Technical Report - IC-26-03 - Relatório Técnico
September - 2026 - Setembro

UNIVERSIDADE ESTADUAL DE CAMPINAS
INSTITUTO DE COMPUTAÇÃO

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

Projeto de uma Rede de Baixo Custo para Monitoramento da Qualidade do Ar em Ambientes Urbanos: Arquitetura IoT e Planejamento da Implantação

Gabriel Lorhan Rodrigues Dourado* Gustavo Montenari Pechta†
Henrique José Silva Moraes Filho† Henrique Parede de Souza‡
Mateus de Lima Almeida‡ Vinicius Patriarca Miranda Miguel†
Juliana Freitag Borin§

Resumo

Este estudo interdisciplinar propõe o desenvolvimento de um sistema de monitoramento da qualidade do ar de baixo custo baseado em Internet das Coisas, utilizando o município de São Paulo como estudo de caso. O projeto foi estruturado em três frentes de pesquisa: (i) a seleção dos sensores ideais para a medição das métricas almejadas pelo projeto, por meio da comparação criteriosa de diferentes opções para cada métrica considerando custo, protocolos de comunicação, vida útil, faixa e resolução da medição, tempo de resposta e condições ambientais de operação; (ii) o desenvolvimento de uma arquitetura de sistema IoT completa e seu fluxo de dados, abrangendo desde a aquisição dos dados, passando pela comunicação segura via protocolo MQTT com certificados TLS, até o processamento, armazenamento em bancos de dados especializados e orquestração por meio de contêineres Docker, garantindo a rastreabilidade, a integridade e a disponibilidade das informações coletadas; (iii) o estudo comparativo de algoritmos de otimização espacial para a definição estratégica dos pontos de instalação das estações, integrando um framework unificado que avalia diferentes métodos de otimização sob uma formulação multicritério que pondera população, vulnerabilidade social, morbidade respiratória e exposição viária, permitindo comparar cobertura, equidade espacial e custo computacional entre as abordagens.

Como resultado, foi possível: (i) obter uma seleção de sensores capazes de realizar as medições almejadas com precisão suficiente para a devida caracterização da qualidade do ar e com custo total estimado de R\$6047,28 por estação; (ii) desenvolver uma rede IoT modular e escalável, com rotinas automatizadas de *backup* e recuperação, capaz de suportar uma carga de 50 estações simultaneamente, com potencial de escalabilidade horizontal para atender a demandas futuras, conforme os resultados parciais obtidos; (iii) comparar treze algoritmos de otimização espacial distintos, demonstrando que métodos exatos (MCLP) e heurísticos (como o algoritmo guloso) são capazes de superar significativamente a cobertura populacional da atual rede de monitoramento da CETESB com um número reduzido de estações, e que a ponderação dos critérios sociais, de saúde e de exposição viária permite direcionar a expansão da rede prioritariamente às regiões mais vulneráveis e sub-monitoradas da cidade, oferecendo ao gestor público uma ferramenta flexível para equilibrar cobertura, equidade e custo computacional na tomada de decisão.

Os códigos desenvolvidos durante estes três macroestudos estão disponíveis em <https://github.com/Henrique-hpds/IoT-pollution-monitoring>.

*Instituto de Química (IQ), UNICAMP.

†Faculdade de Engenharia Elétrica e de Computação (FEEC), UNICAMP.

‡Instituto de Computação (IC), UNICAMP.

§Professora Orientadora. Instituto de Computação (IC), UNICAMP.

Sumário

1	Introdução	3
2	Seleção dos sensores ideais para o projeto	6
2.1	Metodologia	6
2.2	Comparação de modelos de sensores	8
2.3	Decisão final dos sensores	11
3	Arquitetura e Fluxo de Dados do Sistema	14
3.1	Componentes da Arquitetura	15
3.1.1	ESP32	15
3.1.2	TV Box	15
3.1.3	Servidor	16
3.2	Fluxo de Dados	16
3.2.1	Aquisição	17
3.2.2	Comunicação ESP32 - TV Box	18
3.2.3	Comunicação TV Box - Servidor	19
3.2.4	Armazenamento	24
3.2.5	Exportação	27
3.3	Gerenciamento do Sistema	29
3.3.1	Fluxo de Atualização	30
3.3.2	Backup	33
3.4	Considerações	35
4	Estudo de algoritmos para definição dos pontos de instalação	36
4.1	Dados de infraestrutura utilizados	37
4.2	Algoritmos de otimização implementados	39
4.2.1	Baselines	41
4.2.2	Modelos de otimização clássicos	42
4.2.3	Modelos multicritério	45
4.2.4	Modelos baseados em heurísticas e meta-heurísticas	48
4.3	Análise comparativa entre os algoritmos	53
4.3.1	Métricas de cobertura	53
4.3.2	Tempo de execução	58
4.3.3	Análise espacial da alocação (MCLP)	60
4.4	Limitações e trabalhos futuros	63
5	Conclusão	65
A	Apêndice	71
A.1	Responsabilidades específicas de cada integrante	71
A.2	Detalhamento da estimativa do número de dispositivos	71

1 Introdução

Tendo em vista os diversos eventos climáticos extremos ocorridos nos últimos anos, como as enchentes do Rio Grande do Sul de 2024 [27, 64, 60], os efeitos das queimadas na Amazônia de 2019 [48] e possíveis situações extremas com o próximo El Niño de 2026 [45], tornou-se evidente a necessidade de monitoramento de fatores ambientais que demonstrem o compromisso do Brasil com a vanguarda de ações climáticas e subsidiem a formulação de políticas públicas relevantes voltadas à mitigação dos impactos socioambientais associados [51, 26], tais como zonas de calor [50]; chuvas torrenciais [27, 70]; e a piora na qualidade do ar [57]. Esses fatores estão interligados, especialmente no que tange à poluição atmosférica, que, além de afetar a saúde humana, principalmente o sistema respiratório [5], contribui para a redução do pH das chuvas [65]. O excesso de óxidos — sejam de enxofre (SO e SO_2), de nitrogênio (NO e NO_2) ou de carbono (CO e CO_2) — na atmosfera não só agrava esses problemas de saúde, mas também potencializa a acidificação das chuvas, com consequências diretas para os ecossistemas e a agricultura [12].

Estudos realizados em Houston (Texas) demonstram que o monitoramento urbano em alta resolução espacial é essencial para identificar desigualdades intraurbanas na exposição a poluentes atmosféricos [25]. De forma semelhante, análises em Washington, D.C., evidenciam que a distribuição desigual da poluição do ar está diretamente associada a impactos diferenciados na saúde pública [7].

O problema do sub monitoramento é perpetuado pelo alto custo dos equipamentos de medição da qualidade do ar disponíveis atualmente no mercado. A título de exemplo, a atual cobertura das estações de monitoramento da cidade de São Paulo representou um investimento de cerca de 34 milhões de reais [8]. Além disso, essas estações apresentam baixa escalabilidade, exigem manutenção especializada, ocupam áreas consideráveis e não podem ser instaladas em qualquer local (Figura 2). Esses fatores acabam por limitar a expansão da rede de monitoramento pelo poder público municipal e dificultam a disponibilização aberta e contínua dos dados gerados.

Como consequência, formam-se lacunas espaciais significativas na coleta de informações ambientais, que afetam de maneira desigual a produção científica e a formulação de diagnósticos territoriais. Regiões com menor cobertura permanecem sub-representadas em estudos e modelagens, comprometendo assim a compreensão completa dos padrões de poluição e suas relações com fatores sociais. Desta forma, políticas públicas elaboradas utilizando a fundamentação teórica destas pesquisas acabam sendo potencialmente enviesadas, visto que são baseadas em evidências incompletas, tendendo a reproduzir as desigualdades já existentes ao direcionar recursos e intervenções de forma desproporcional.

Neste contexto, o presente estudo propõe o desenvolvimento de uma rede colaborativa de sensores para o monitoramento da qualidade do ar, com a escolha dos pontos de instalação pensada para o município de São Paulo. Embora o projeto seja fundamentado na premissa de utilização de dispositivos de baixo custo para viabilizar a ampla escalabilidade da rede, é importante ressaltar que determinados sensores — especialmente os empregados na medição de material particulado ($MP_{2,5}$) ou de gases específicos — podem demandar um investimento inicial mais elevado. Contudo, a estratégia adotada busca equilibrar o custo-benefício, priorizando a acurácia necessária para a geração de dados confiáveis, sem inviabilizar a expansão do sistema, de modo que o termo “baixo custo” refere-se ao custo médio da solução completa em comparação com estações de referência tradicionais.

Para atingir esse objetivo, o projeto foi estruturado em três frentes de estudo:

1. a seleção criteriosa dos sensores, considerando custo, protocolos de comunicação para integração nas estações, vida útil (ou seja, velocidade de decaimento ao longo do tempo), faixa e

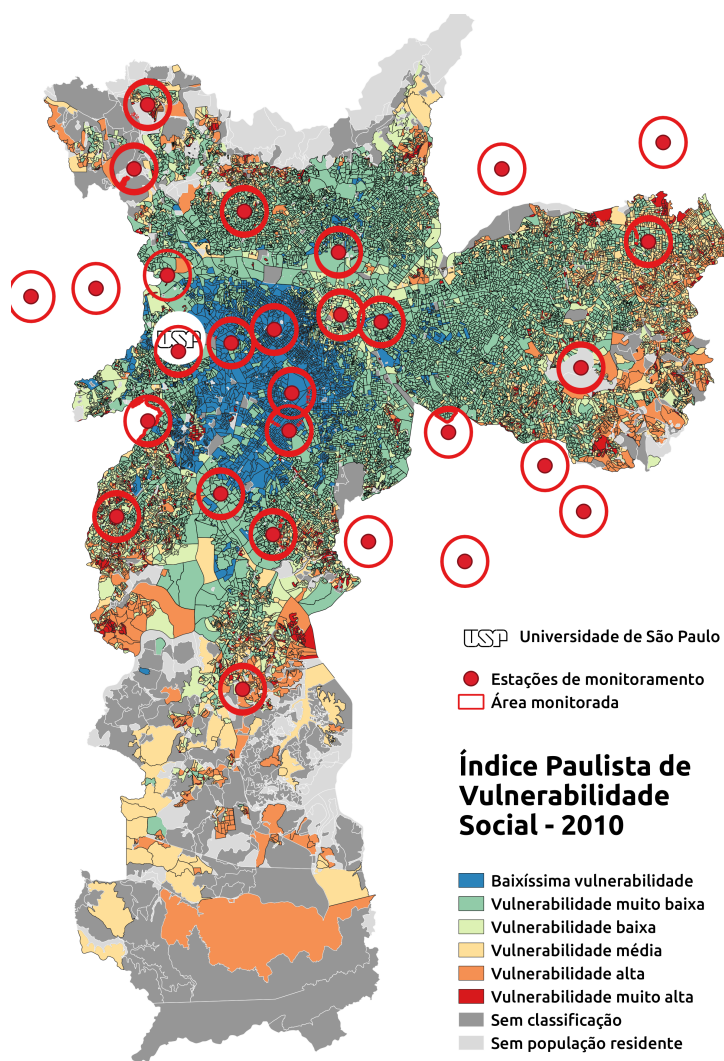


Figura 1: Distribuição do Índice Paulista de Vulnerabilidade Social (IPVS) na cidade de São Paulo e localização das estações de monitoramento da qualidade do ar. As cores indicam os níveis de vulnerabilidade social, e os círculos vermelhos representam as áreas cobertas por cada estação. Gráfico de elaboração dos próprios autores com dados do GeoSampa, do Sistema Estadual de Análise de Dados (SEADE) e da Companhia Ambiental do Estado de São Paulo (CETESB).



Figura 2: Modelo de estação de qualidade do ar - CETESB - divulgação

resolução da medição, tempo de resposta e condições ambientais de operação;

2. o desenvolvimento de um sistema completo baseado em Internet das Coisas (IoT), abrangendo desde a aquisição dos dados nas Unidades Básicas de Saúde (UBS) até a transmissão segura e o armazenamento em nuvem, incluindo as rotinas de agendamento e *backup*;
3. a criação e comparação de algoritmos de otimização para a definição da localização estratégica dos pontos de monitoramento, visando maximizar a cobertura populacional da rede, com foco nas populações socialmente vulneráveis e mais expostas à poluição atmosférica, a partir de um número limitado de dispositivos.

A Figura 3 apresenta a arquitetura geral do sistema proposto, sintetizando o fluxo completo de informações desde a infraestrutura física até a camada de aplicação. Conforme ilustrado no fluxograma, é possível estabelecer uma correlação direta entre as 3 frentes de estudo e as etapas representadas:

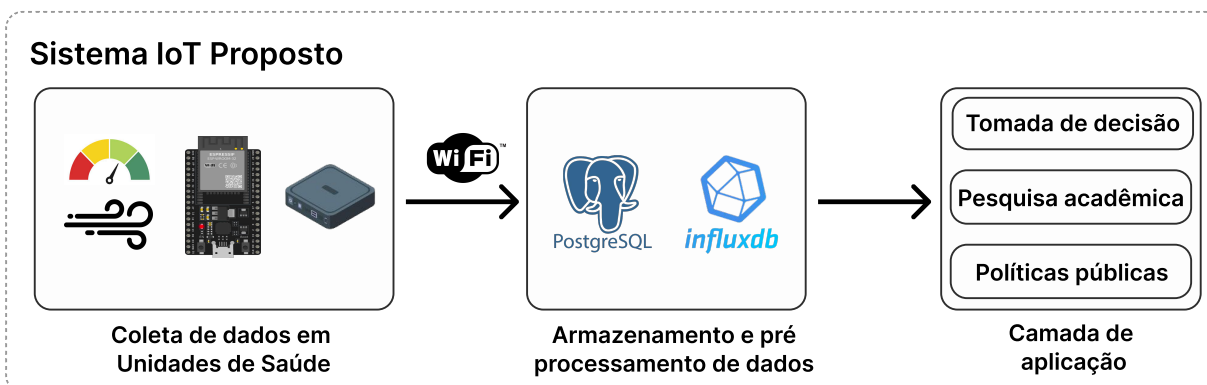


Figura 3: Fluxograma da arquitetura do sistema proposto, ilustrando o fluxo de dados.

- A **infraestrutura física para instalação** e a etapa de **coleta de dados em Unidades de Saúde** estão diretamente vinculadas ao primeiro e ao terceiro macroestudos. O primeiro

define quais sensores serão utilizados (garantindo a qualidade dos dados brutos), enquanto o terceiro determina os melhores locais para alocação desses dispositivos, otimizando a cobertura da rede.

- A etapa de **processamento de dados** e a própria construção da **camada de aplicação** são habilitadas pelo segundo macroestudo, que provê toda a infraestrutura de comunicação, orquestração de contêineres, persistência em bancos de dados (*InfluxDB* e *PostgreSQL*) e pré processamento necessários para transformar os dados brutos em informações estruturadas.
- Por fim, as camadas superiores do fluxograma — **tomada de decisão, pesquisa acadêmica e políticas públicas** — representam os objetivos finais da rede. Elas dependem da integração bem-sucedida de todos os macroestudos, uma vez que a confiabilidade dos dados (proveniente da seleção adequada dos sensores e da correta instalação) e a fluidez do processamento (garantida pelo sistema IoT) são requisitos essenciais para a geração de *insights* e suporte à gestão pública.

Dessa forma, a arquitetura apresentada na Figura 3 não apenas organiza o fluxo de dados do sistema, mas também reflete a interdisciplinaridade e a complementaridade entre os três eixos de pesquisa que norteiam este trabalho.

2 Seleção dos sensores ideais para o projeto

A seleção dos sensores constitui uma etapa crítica e estratégica do presente projeto, pois a qualidade, confiabilidade e representatividade dos dados coletados dependem diretamente do desempenho dos transdutores empregados nas estações experimentais. Além disso, o monitoramento atmosférico em uma metrópole como São Paulo impõe desafios severos aos equipamentos: variações térmicas e de umidade, elevada concentração de poluentes interferentes, chuva, radiação solar intensa, ventos e a necessidade de operação contínua e autônoma por longos períodos.

2.1 Metodologia

Inicialmente, definimos a lista de poluentes que desejamos monitorar com base no que é medido pela rede de monitoramento da CETESB: dióxido de enxofre (SO_2), óxidos de nitrogênio (NO_x), monóxido de carbono (CO), enxofre reduzido total (TRS), ozônio (O_3), BTEX (benzeno e tolueno), compostos orgânicos voláteis (VOCs), material particulado ($MP_{2,5}$ e MP_{10}), partículas totais em suspensão e aldeídos (acetaldeído e formaldeído) [11].

Vale destacar algumas definições a respeito dos poluentes listados. TRS é uma denominação utilizada para identificar os compostos sulfeto de hidrogênio (H_2S), metil mercaptana (CH_3SH), dimetil sulfeto ($(CH_3)_2S$), dimetil dissulfeto ($(CH_3)_2S_2$) [23]. No contexto de sensores industriais para medição de poluentes, é incomum encontrar sensores para medição dos outros compostos fora o H_2S , por isso buscamos sensores para detecção deste.

VOCs são uma categoria ampla da qual BTEX são um conjunto mais restrito, destacado pela sua mobilidade ambiental e toxicidade e pela sua origem associada a combustíveis, solventes e derivados de petróleo [49][20]. Dessa forma, os sensores selecionados cobrem ambas as leituras.

Material particulado e partículas totais em suspensão, conforme resolução da CONAMA que estabelece padrões nacionais de qualidade do ar [6], corresponde a partículas de material sólido ou líquido suspensas no ar, na forma de poeira, neblina, aerossol, fuligem, entre outros. Essas partículas são classificadas de acordo com seu diâmetro aerodinâmico equivalente de corte. As classificações $MP_{2,5}$ e MP_{10} , agrupadas sob a denominação de material particulado, são partículas,

respectivamente, de $2,5\ \mu\text{m}$ e de $10\ \mu\text{m}$ de diâmetro. As partículas totais em suspensão são aquelas que possuem $50\ \mu\text{m}$ de diâmetro.

Para realizar a seleção, após determinar uma lista de sensores, definimos uma série de critérios para comparação das opções elencadas.

- **Resolução:** Para realizar a devida caracterização da qualidade do ar de um ambiente urbano, precisamos de sensores capazes de gerar dados confiáveis assim como os das estações da CETESB, o que se traduz em dispositivos com limites de detecção na ordem de partes por bilhão (ppb) e seletividade química suficiente para minimizar interferências cruzadas entre gases.
- **Faixa de medição:** Caracteriza o mínimo e o máximo de concentração de gás medido pelo sensor. A amplitude dessa faixa afeta a robustez da medição contra variações da qualidade do ar ao longo do tempo.
- **Tempo de resposta (Tempo de Subida/Descida - T90):** É o intervalo de tempo necessário para que o sinal de saída do sensor atinja 90% do valor final de estabilização, após uma variação abrupta e instantânea na concentração do gás alvo. Essa característica afeta a medição de maneira similar à faixa de medição, porém em uma escala instantânea.
- **Condições Ambientais de Operação:** O microclima de São Paulo exige que os sensores operem com desempenho estável em faixas de temperatura médias de $16,4^{\circ}\text{C}$ a $26,1^{\circ}\text{C}$, com picos atípicos, e umidade relativa frequentemente entre 60% e 80% [41].
- **Integração nas estações (protocolos de comunicação):** A facilidade de interface (saídas padronizadas como UART, I2C, SPI ou sinais Leitura analógicas de 4-20 mA) e a disponibilidade de documentação técnica detalhada foram consideradas para assegurar a viabilidade da implementação eletrônica e o processamento digital dos sinais.
- **Vida útil:** Considerando a logística e os **Custos** envolvidos na manutenção de uma rede de estações espalhadas pela cidade, a reposição frequente de sensores é indesejável, por isso é ideal que os sensores apresentem a menor taxa de degradação possível.
- **Relação Custo-Benefício:** Um dos objetivos principais do projeto é propor uma alternativa econômica às estações de monitoramento da CETESB, que custam centenas de milhares de reais [8]. Dentre os materiais do projeto, os sensores representam o maior custo financeiro. Considerando um orçamento destinado na ordem de alguns milhares de reais por estação, precisamos aliar os demais critérios apresentados com o menor custo possível para realizar a decisão final dos sensores.

Para obter as informações de resolução, faixa de medição, tempo de reposta, condições ambientais de operação, protocolos de comunicação e vida útil foram consultados os datasheets dos sensores, fornecidos pelas próprias fabricantes de maneira oficial.

O custo desse tipo de sensor foi determinado como uma estimativa a partir de algumas fontes. Não é trivial encontrar o preço de mercado desses sensores pois eles são de nível profissional e não são encontrados em varejos comuns, pois não são vendidos diretamente ao público geral. O seu preço final estabelecido pelas fabricantes depende do volume do lote, do perfil do cliente (universidade, governo ou empresa privada), de contratos de fornecimento contínuo e das taxas de calibração em fábrica. Uma forma de obter uma estimativa deste preço é consultar redes de distribuidores autorizados que podem ter preços mais acessíveis ou divulgados. Para as estimativas apresentadas

no relatório, os sensores foram buscados nas plataformas DigiKey [2], Mouser Electronics [4], Direct Industry [3] e Alibaba [1], e foi realizada a média dos preços encontradas à época da pesquisa (julho de 2026).

Por fim, destacamos que uma métrica importante para a especificação das estações experimentais não será discutida nesta seção: o raio de detecção de poluentes. Após analisar uma extensa bibliografia de especificações de sensores, foi possível concluir que essa não é uma métrica intrínseca ao sensor, e que, dessa forma, deve ser determinada a partir de outros fatores fora do escopo desta seção.

2.2 Comparação de modelos de sensores

Nas tabelas 1 a 9 serão apresentadas para cada poluente a ser medido três opções de sensor para cada, organizados em tabelas listando suas características. Em cada tabela foi posto em negrito o melhor valor encontrado.

Sensores de SO ₂			
Modelo	ES4-SO2-100	SO2-B4	SO2/SF-2000-S
Fabricante	EC Sense	Alphasense	Membrapor
Faixa de medição	0 a 100 ppm	0 a 100 ppm	0 a 2000 ppm
Resolução	0,1 ppm	0,005 ppm	< 0,5 ppm
Vida útil	> 3 anos	> 3 anos	2 anos
Tempo de resposta	< 60s	< 60s	< 25s
Temperatura de operação	-40 a 55 °C	-30 a 50 °C	-20 a 50 °C
Umidade relativa de operação	15% a 95%	15% a 90%	15% a 90%
Custo	R\$349,72	R\$360,32	R\$726,46
Protocolos de comunicação	Leitura analógica	Leitura analógica	Leitura analógica

Tabela 1: Comparação dos sensores de medição de SO₂

Sensores de NO ₂			
Modelo	NO2-B43F	OX-B431	ES4-NO2-2000
Fabricante	Alphasense	Alphasense	EC Sense
Faixa de medição	0 a 20 ppm	0 a 20 ppm	0 a 2000 ppm
Resolução	0,015 ppm	0,015 ppm	< 0,1 ppm
Vida útil	> 2 anos	> 2 anos	> 3 anos
Tempo de resposta	< 80s	< 80s	< 30s
Temperatura de operação	-30 a 40 °C	-30 a 40 °C	-40 a 54 °C
Umidade relativa de operação	15% a 85%	15% a 85%	15% a 95%
Custo	R\$360,32	R\$1589,63	R\$272,04
Protocolos de comunicação	Leitura analógica	Leitura analógica	Leitura analógica

Tabela 2: Comparação dos sensores de medição de NO₂

Sensores de NO			
Modelo	S4NO-250	NO-B4	NO-AE
Fabricante	Sangbay	Alphasense	EC Sense
Faixa de medição	0 a 50 ppm	0 a 20 ppm	0 a 200 ppm
Resolução	1 ppm	0,015 ppm	< 0,1 ppm
Vida útil	2 anos	> 2 anos	> 2 anos
Tempo de resposta	< 45s	< 80s	< 30s
Temperatura de operação	-30 a 50 °C	-30 a 40 °C	-30 a 50 °C
Umidade relativa de operação	15% a 90%	15% a 85%	15% a 90%
Custo	R\$307,33	R\$588,16	R\$542,60
Protocolos de comunicação	Leitura analógica	Leitura analógica	Leitura analógica

Tabela 3: Comparação dos sensores de medição de *NO*

Sensores de CO			
Modelo	CO-B4	CO/SF-500	ES1-CO-1000
Fabricante	Alphasense	Membrapor	EC Sense
Faixa de medição	0 a 1000 ppm	0 a 500 ppm	0 a 1000 ppm
Resolução	0,004 ppm	0,2 ppm	0,1 ppm
Vida útil	> 3 anos	3 a 5 anos	5 anos
Tempo de resposta	< 30s	< 50s	< 30s
Temperatura de operação	-30 a 50 °C	-40 a 50 °C	-40 a 55 °C
Umidade relativa de operação	15% a 90%	15% a 90%	15% a 95%
Custo	R\$344,42	R\$1348,32	R\$59,77
Protocolos de comunicação	Leitura analógica	Leitura analógica	Leitura analógica

Tabela 4: Comparação dos sensores de medição de *CO*

Sensores de H ₂ S			
Modelo	H2S-B4	ES1-H2S-500	DTS-4H2S
Fabricante	Alphasense	EC Sense	DOUTESI
Faixa de medição	0 a 100 ppm	0 a 500 ppm	0 a 100 ppm
Resolução	0,004 ppm	0,01 ppm	1 ppm
Vida útil	> 2 anos	5 anos	1 ano
Tempo de resposta	< 30s	< 30s	< 30s
Temperatura de operação	-30 a 50 °C	-40 a 50 °C	-40 a 50 °C
Umidade relativa de operação	15% a 90%	15% a 95%	15% a 90%
Custo	R\$450,40	R\$219,80	R\$46,74
Protocolos de comunicação	Leitura analógica	Leitura analógica	Leitura analógica

Tabela 5: Comparação dos sensores de medição de *H₂S*

Sensores de O_3			
Modelo	O3/C-100	OX-B431	ES1-O3-5
Fabricante	Membrapor	Alphasense	EC Sense
Faixa de medição	0 a 100 ppm	0 a 20 ppm	0 a 5 ppm
Resolução	0,1 ppm	0,015 ppm	< 0,01 ppm
Vida útil	3 a 5 anos	> 2 anos	> 2 anos
Tempo de resposta	< 60s	< 80s	< 120s
Temperatura de operação	-40 a 50 °C	-30 a 40 °C	-20 a 55 °C
Umidade relativa de operação	15% a 90%	15% a 85%	15% a 95%
Custo	R\$763,02	R\$1589,63	R\$264,94
Protocolos de comunicação	Leitura analógica	Leitura analógica	Leitura analógica

Tabela 6: Comparação dos sensores de medição de O_3

Sensores de VOCs e BTEX			
Modelo	VOC-B4	PID-A15	MiniPID 2
Fabricante	Alphasense	Alphasense	ION
Faixa de medição	0 a 100 ppm	0 a 200 ppm	0 a 4000 ppm
Resolução	0,02 ppm	0,1 ppm	0,1 ppm
Vida útil	> 3 anos	5 anos	> 5 anos
Tempo de resposta	< 30s	2s	< 3s
Temperatura de operação	-30 a 50 °C	-20 a 60 °C	-40 a 65 °C
Umidade relativa de operação	15% a 90%	0% a 95%	0% a 99%
Custo	R\$169,56	R\$4541,66	R\$2956,70
Protocolos de comunicação	Leitura analógica	Leitura analógica	Leitura analógica

Tabela 7: Comparação dos sensores de medição de VOCs e BTEX

Sensores de $MP_{2,5}$, MP_{10} e partículas totais em suspensão			
Modelo	OPC-N3	HPMA115S0	SPS30
Fabricante	Alphasense	Honeywell	Sensirion
Faixa de medição	0,35 a 40 μm	2,5 a 10 μm	2,5 a 10 μm
Resolução	0,35 μm	0,35 μm	0,35 μm
Vida útil	2 anos	10 anos	10 anos
Tempo de resposta	1 a 30s (configurável)	< 6s	1 a 30s (configurável)
Temperatura de operação	-10 a 50°C	-20 a 50°C	-10 a 60°C
Umidade relativa de operação	0% a 95%	0% a 95%	0% a 95%
Custo	R\$2119,50	R\$350,25	R\$142,54
Protocolos de comunicação	SPI e USB	UART	UART e I2C

Tabela 8: Comparação dos sensores de medição de $MP_{2,5}$, MP_{10} e partículas totais em suspensão

Sensores de aldeídos			
Modelo	WZ-S	AT004-J00	CH2O/M-10
Fabricante	Dart	3ETO	Membrapor
Faixa de medição	0 a 2 ppm	0 a 20 ppm	0 a 10ppm
Resolução	0,001 ppm	0,1 ppm	0,01 ppm
Vida útil	5 anos	2 anos	5 anos
Tempo de resposta	< 40s	< 140s	< 30s
Temperatura de operação	-20 a 50°C	-20 a 50°C	-40 a 50°C
Umidade relativa de operação	10% a 90%	15% a 90%	15% a 90%
Custo	R\$64,97	R\$300,97	R\$752,43
Protocolos de comunicação	UART	Leitura analógica	Leitura analógica

Tabela 9: Comparação dos sensores de medição de aldeídos

2.3 Decisão final dos sensores

Vale destacar que as condições ambientais de operação não foram um fator relevante na decisão, pois todos os sensores considerados possuem condições que abrangem as condições climáticas de São Paulo [41].

Em relação ao critério de custo dos sensores, note que se escolhêssemos todas as opções mais caras elencadas obteríamos um custo total de R\$13706,19 por estação. Esse valor já significaria uma redução de pelo menos uma ordem de grandeza em relação às estações atuais da CETESB, portanto para o nível deste projeto de planejamento o custo não caracterizou um critério impeditivo. Ainda assim, sempre que possível foi escolhido a opção mais econômica encontrada.

Os critérios de resolução e faixa de medição são fundamentais para garantir que o sensor entregue leituras que sejam adequadas para a classificação da qualidade do ar e foram o foco principal na decisão. Exceto os sensores de material particulado, os demais realizam medições em unidades de concentração volumétrica em ppm. Entretanto, no boletim anual de qualidade do ar da CETESB as concentrações são descritas em unidades de concentração de massa $\mu\text{g}/\text{m}^3$. Dessa forma, será necessário converter as unidades dos valores coletados para esses critérios.

Para tanto, utilizamos a fórmula $\mu\text{g}/\text{m}^3 = \text{ppm} \cdot \text{Massa Molar} \cdot \frac{1000 \cdot \text{Pressão}}{R \cdot \text{Temperatura}}$, onde R é a constante universal dos gases [62]. Consideramos condições meteorológicas médias de São Paulo de 938 hPa de pressão atmosférica e 24 °C [44]. Para as massas molares, utilizamos o Livro de Química na Web do NIST [54]. Os resultados das conversões encontram-se na Tabela 10.

Utilizamos o boletim mensal de qualidade do ar da CETESB mais recente no presente momento [9] como referência para qual a faixa de medição e resolução necessária nos sensores. Os boletins classificam a concentração de poluente em faixas de qualidade do ar conforme padrões nacionais. Assim sendo, a faixa de medição do sensor precisa incluir o maior valor de faixa considerado pela CETESB e sua resolução precisa ser suficiente para detectar concentrações em todas as faixas de classificação. Os valores de referência encontram-se na Tabela 11.

Poluente	Massa molar (g/mol)	0,001 ppm
<i>SO₂</i>	64,064	2,403 $\mu\text{g}/\text{m}^3$
<i>NO₂</i>	46,006	1,726 $\mu\text{g}/\text{m}^3$
<i>NO</i>	30,006	1,126 $\mu\text{g}/\text{m}^3$
<i>CO</i>	28,010	1,051 $\mu\text{g}/\text{m}^3$
<i>H₂S</i> (TRS)	34,081	1,279 $\mu\text{g}/\text{m}^3$
<i>O₃</i>	47,998	1,801 $\mu\text{g}/\text{m}^3$
Benzeno	78,112	2,930 $\mu\text{g}/\text{m}^3$
Tolueno	92,138	3,457 $\mu\text{g}/\text{m}^3$
Formaldeído	30,026	1,127 $\mu\text{g}/\text{m}^3$
Acetaldeído	44,053	1,653 $\mu\text{g}/\text{m}^3$

Tabela 10: Conversão de unidades (condições: 24 °C e 930 hPa).

Poluente	Ordem de grandeza das faixas de classificação	Maior valor de faixa
<i>SO₂</i>	1 $\mu\text{g}/\text{m}^3$	> 800 $\mu\text{g}/\text{m}^3$
<i>NO₂</i>	10 $\mu\text{g}/\text{m}^3$	> 1130 $\mu\text{g}/\text{m}^3$
<i>NO</i>	10 $\mu\text{g}/\text{m}^3$	> 1130 $\mu\text{g}/\text{m}^3$
<i>CO</i>	1 ppm	> 15 ppm
<i>H₂S</i> (TRS)	10 ⁻³ ppm	> 0,2 ppm
<i>O₃</i>	10 $\mu\text{g}/\text{m}^3$	> 200 $\mu\text{g}/\text{m}^3$
Benzeno	1 $\mu\text{g}/\text{m}^3$	> 20 $\mu\text{g}/\text{m}^3$
Tolueno	1 $\mu\text{g}/\text{m}^3$	> 60 $\mu\text{g}/\text{m}^3$
Formaldeído	0,001 ppm	Não há padrão nacional
Acetaldeído	0,001 ppm	Não há padrão nacional

Tabela 11: Referência para a resolução e faixa esperadas pela CETESB em medições de concentração de poluentes.

Os demais critérios de vida útil e tempo de resposta foram utilizados como critério de desempate, quando necessário. A seguir, detalha-se a decisão final para cada poluente a ser monitorado:

- **SO₂**: pela resolução, o modelo selecionado foi o **SO2-B4 (Alphasense)**.
- **NO₂**: pela resolução, os candidatos são o NO2-B43F e o OX-B431. O **NO2-B43F (Alphasense)** foi escolhido, pois ele possui especificações iguais ao OX-B431, porém com custo menor.
- **NO**: pela resolução, foi escolhido o **NO-B4 (Alphasense)**.
- **CO**: pela resolução, foi escolhido o **CO-B4 (Alphasense)**.
- **H₂S**: pela resolução, foi escolhido o **H2S-B4 (Alphasense)**.
- **O₃**: pela resolução, foi escolhido o **OX-B431 (Alphasense)**.
- **VOCs e BTEX**: pela resolução, foi escolhido o **VOC-B4 (Alphasense)**. Note que mesmo esse sensor possuindo a melhor resolução, ela ainda não é suficiente para realizar medições capazes de serem encaixadas nas faixas de classificação. Dessa forma, o sistema seria limitado apenas a monitorar se a concentração desses compostos está acima da pior classificação.
- **Material Particulado e partículas totais em suspensão**: pela faixa de medição, foi escolhido o **OPC-N3 (Alphasense)**. Como deseja-se medir tanto material particulado quanto partículas totais em suspensão, ele é o único que possui uma faixa de medição que ultrapassa os 10 μm de diâmetro de partícula.
- **Aldeídos (Formaldeído e Acetaldeído)**: pela resolução, foi escolhido o **WZ-S (Dart)**.

Poluente Alvo	Modelo Selecionado	Fabricante	Protocolo de comunicação	Custo Unitário
SO ₂	SO2-B4	Alphasense	Leitura analógica	R\$ 360,32
NO ₂	NO2-B43F	Alphasense	Leitura analógica	R\$ 360,32
NO	NO-B4	Alphasense	Leitura analógica	R\$ 588,16
CO	CO-B4	Alphasense	Leitura analógica	R\$ 344,42
H ₂ S (TRS)	H2S-B4	Alphasense	Leitura analógica	R\$ 450,40
O ₃	OX-B431	Alphasense	Leitura analógica	R\$ 1589,63
VOCs / BTEX	VOC-B4	Alphasense	Leitura analógica	R\$ 169,56
Material Particulado	OPC-N3	Alphasense	UART e I2C	R\$ 2119,50
Aldeídos	WZ-S	Dart Sensors	UART	R\$ 64,97
Custo Total em Sensores por Estação:				R\$ 6047,28

Tabela 12: Consolidação dos sensores escolhidos para a estação experimental

O custo total estimado dos sensores foi determinado R\$6047,28 por estação. Este valor cumpre a restrição desejada, ao mesmo tempo que viabiliza uma estação tecnicamente qualificada. Como a maioria dos sensores de gases selecionados opera via leitura analógica em corrente/tensão de alta sensibilidade, a eletrônica de condicionamento de sinal da estação dedicará atenção especial à blindagem contra ruídos e ao uso de conversores Leitura analógica-digitais (ADC) externos de alta resolução acoplados ao ESP32.

3 Arquitetura e Fluxo de Dados do Sistema

A arquitetura do sistema de IoT desenvolvido é apresentada desde a estrutura existente nas estações de coleta de dados, envolvendo os sensores, o microcontrolador e a TV Box, até o armazenamento das informações no servidor.

No que se refere à infraestrutura de hardware, foram escolhidos o ESP32 e a TV Box como os principais dispositivos da arquitetura proposta. O ESP32 foi selecionado por equilibrar alto desempenho e baixo custo. Complementarmente, a TV Box atua como um nó intermediário de processamento e comunicação, sendo capaz de executar um sistema operacional Linux, o que proporciona maior robustez para a execução dos serviços responsáveis pelo gerenciamento, processamento e encaminhamento dos dados coletados. Além disso, a escolha desse dispositivo também se justifica pela ampla disponibilidade desse tipo de equipamento no mercado nacional, em decorrência das milhares de apreensões realizadas pela Receita Federal [19], tornando-o uma alternativa de baixo custo e fácil aquisição para a implementação do sistema.

Todo o sistema foi desenvolvido utilizando Docker, tendo em vista a grande flexibilidade, modularidade e escalabilidade ofertadas por essa ferramenta [69]. Essas características favorecem especialmente o desenvolvimento em ambientes com sistemas operacionais distintos (como as próprias TV Boxes), uma vez que o *Docker* cria uma camada de abstração sobre o ambiente de execução, facilitando a portabilidade entre diferentes modelos e versões de sistema operacional.

A arquitetura proposta é subdividida em três grandes fluxos (Figura 4): *Fluxo de Dados*, *Fluxo de Atualização* e *Fluxo de Backup*.

Fluxo de Dados: inicia-se com um *Docker* que faz a comunicação entre a TV Box e o ESP. Desse contêiner, os dados seguem para outro *Docker* (MQTT Sender¹), especializado no envio via protocolo MQTT para o servidor. No servidor, esses dados são consumidos e direcionados para o respectivo banco de dados. Durante esse percurso, foram implementados mecanismos de tolerância a falhas para lidar com eventuais quedas de serviços.

Fluxo de Atualização: opera sobre um servidor HTTPS contendo as versões compactadas do sistema das estações e uma lista de versões. Uma rotina na estação baixa essa lista e verifica se uma atualização é necessária. Durante o processo, logs de status são enviados via MQTT Sender. Em caso de sucesso, os contêineres *Docker* e o código presente no ESP são substituídos pelas novas versões. Em caso de falha, um mecanismo de *rollback* restaura a versão anterior à tentativa de atualização.

Fluxo de Backup: consiste em uma rotina que coleta os dados armazenados no banco de dados, salvando-os em pastas externas aos contêineres Docker. Essa abordagem visa evitar a perda de dados, garantindo a integridade das informações mesmo em cenários de falha nos volumes ou contêineres.

¹Docker responsável pelo envio de mensagens via MQTT, com funcionalidades adicionais como padronização de tópicos, gerenciamento de certificados e mecanismos de fallback em caso de falha do broker.

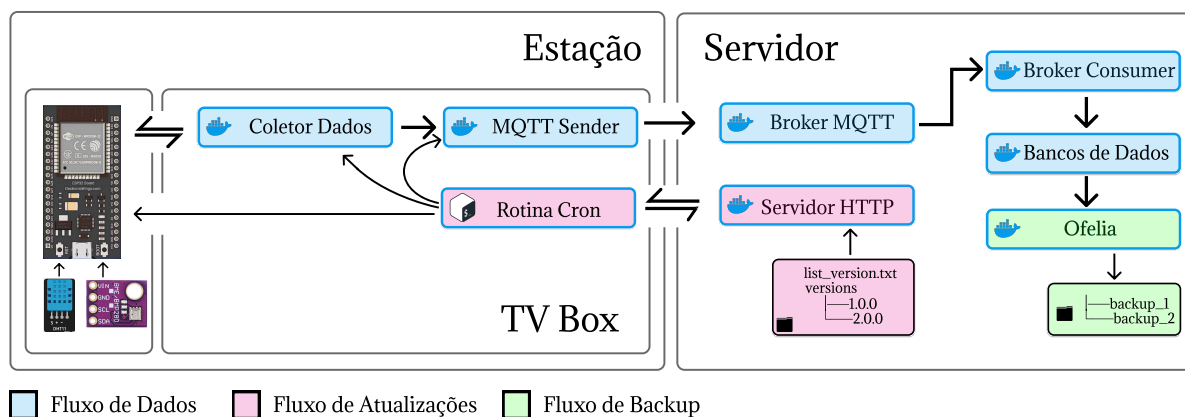


Figura 4: Esquema de geral dos fluxos

3.1 Componentes da Arquitetura

Nessa parte, serão apresentados os principais componentes que compõem a arquitetura proposta, descrevendo suas respectivas funções e a forma como se integram para viabilizar o funcionamento do sistema. Para cada componente, são discutidas suas responsabilidades dentro da arquitetura, bem como as motivações que fundamentaram sua escolha, considerando aspectos como custo, desempenho, disponibilidade, escalabilidade e compatibilidade com os requisitos da aplicação de Internet das Coisas desenvolvida.

3.1.1 ESP32

O ESP32 constitui o nó de aquisição de dados do sistema e realiza a leitura direta dos dados coletados pelos sensores. Como microcontrolador, o ESP32 possui baixa capacidade de armazenamento de dados, então sua responsabilidade é escoá-los para a TV Box. Este microcontrolador possui um processador dual-core de 32 bits com clock de até 240 MHz e 520 KB de SRAM, capazes de processar simultaneamente os dados de múltiplos sensores com baixa latência. Sua conectividade Wi-Fi (802.11 b/g/n) e Bluetooth 4.2/LE integradas eliminam a necessidade de módulos externos para comunicação com a TV Box e o sistema central. Além disso, suas múltiplas interfaces UART, I2C e SPI, além de pinos GPIO e conversores ADC garantem compatibilidade com a maioria dos sensores industriais. Seu baixo custo e vasto ecossistema de desenvolvimento viabilizam sua implantação em escala nas estações experimentais [32].

3.1.2 TV Box

A TV Box desempenha o papel de nó intermediário de processamento e comunicação entre o microcontrolador ESP32 e o servidor, agregando robustez e capacidade de armazenamento local à estação de coleta. Enquanto o ESP32 é responsável pela leitura direta dos sensores, a TV Box executa um sistema operacional Linux, sobre o qual são gerenciados *Docker* que implementam os fluxos de dados, atualização e *backup*. Essa plataforma oferece recursos computacionais e de memória substancialmente superiores aos de um microcontrolador típico, permitindo a execução de serviços complexos como o gerenciador de comunicação com o ESP, o *MQTT Sender* (com mecanismos de fallback e padronização de tópicos), rotinas de verificação e aplicação de atualizações e cópias de segurança locais. Sua capacidade de armazenamento interno viabiliza o *buffering* de

grandes volumes de dados durante falhas de conectividade, assegurando a integridade das séries temporais mesmo em cenários de instabilidade da rede. Essas funcionalidades dificilmente seriam implementadas diretamente em um ESP32, cujas limitações de memória RAM, armazenamento e poder de processamento o restringem a tarefas embarcadas mais simples. Dessa forma, a arquitetura proposta se beneficia da separação de responsabilidades: o ESP32 atua como ponte para os sensores, enquanto a TV Box provê a inteligência de borda e a resiliência necessárias a aplicações IoT críticas. A escolha específica da TV Box, por sua vez, é fundamentada na disponibilidade desse tipo de equipamento para fins acadêmicos, decorrente das milhares de apreensões realizadas pela Receita Federal [19], o que a torna uma alternativa de baixo custo para a implementação em larga escala do sistema.

3.1.3 Servidor

O servidor constitui o nó central da arquitetura, concentrando a recepção, o armazenamento e a distribuição de atualizações para todas as estações de coleta. Nele são executados três serviços principais: o *broker*, responsável por receber as mensagens de telemetria publicadas pelo *MQTT Sender* das estações e encaminhá-las para processamento; um banco de dados temporal (InfluxDB), no qual as séries de leituras dos sensores são persistidas de forma eficiente para consultas e análises posteriores; e um servidor HTTPS que hospeda as imagens compactadas do sistema e uma lista de versões, viabilizando o fluxo de atualização, os quais serão detalhados posteriormente. Adicionalmente, rotinas de *backup* periódicas extraem os dados dos volumes do banco para diretórios externos ao *Docker*, reforçando a tolerância a falhas e a integridade das informações. Para fins de desenvolvimento e teste, todo o conjunto de serviços foi implantado em uma placa *Raspberry Pi 5* executando Ubuntu 24.04.4 LTS (aarch64), mas cabe ressaltar que para fins de produção deve-se utilizar um hardware mais robusto.

3.2 Fluxo de Dados

O fluxo de dados pode ser subdividido em quatro grandes etapas: **aquisição**, a **coleta dos dados do ESP32 pela TV Box**, o **envio desses dados pela TV Box ao servidor** e o **recebimento e armazenamento dos dados no servidor**, conforme ilustrado na Figura 5. Essa divisão permite descrever o funcionamento do sistema de forma modular, uma vez que cada etapa se comunica com a seguinte por meio de uma interface bem definida. Nas subseções a seguir, cada etapa é apresentada em detalhes.

Como protocolo de comunicação entre as estações e o servidor, foi adotado o MQTT (*Message Queuing Telemetry Transport*), por se tratar de um protocolo leve baseado no modelo de publicação e assinatura (*publish/subscribe*). Esse protocolo apresenta baixo consumo de banda e bom desempenho em redes instáveis ou com recursos limitados, características que o tornam adequado para aplicações de Internet das Coisas (IoT) [56]. Além disso, o MQTT oferece diferentes níveis de qualidade de serviço (QoS), permitindo ajustar a confiabilidade da transmissão conforme a criticidade das informações enviadas [56].

O modelo de publicação e assinatura também favorece a escalabilidade da arquitetura, uma vez que desacopla produtores e consumidores de dados. Dessa forma, novos dispositivos e aplicações podem ser adicionados sem alterações significativas na estrutura do sistema. Além disso, o broker MQTT gerencia de forma eficiente múltiplas conexões simultâneas, possibilitando comunicação assíncrona com baixa latência [52].

Estudos experimentais mostram que sistemas simples implementados com o broker Mosquitto conseguem suportar, no pior caso analisado, aproximadamente 5 mil mensagens por segundo uti-

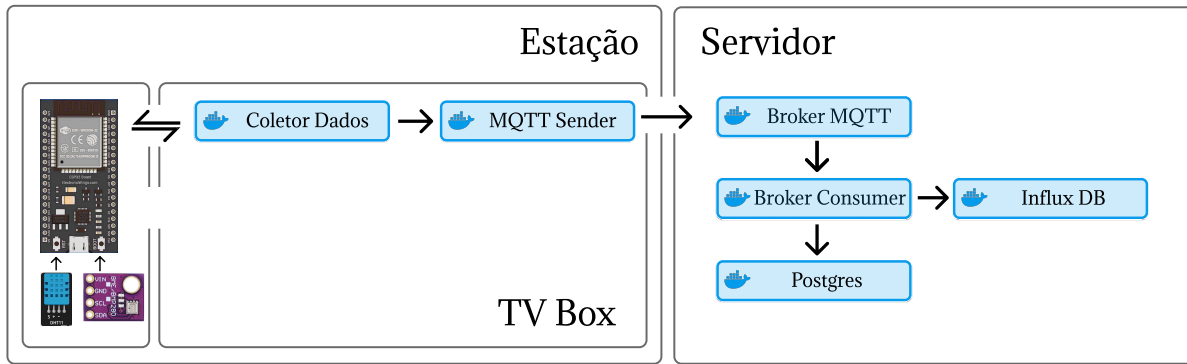


Figura 5: Esquema geral do fluxo de dados, desde os sensores até o banco de dados

lizando QoS 2 [52]. Nesse cenário, considera-se uma carga composta por 50 estações enviando mensagens em 10 tópicos distintos, o que representa um cenário de estresse do atual sistema proposto, haveria 500 mensagens por segundo. Dessa forma, esses resultados indicam que a arquitetura proposta é capaz de lidar com o pior cenário previsto para esse estudo, garantindo funcionamento adequado mesmo sob alta carga de mensagens distribuídas.

3.2.1 Aquisição

Para viabilizar a coleta e o processamento dos dados meteorológicos e ambientais, os sensores selecionados serão integrados a um microcontrolador ESP32-S3 (WROOM-1) [32]. Este modelo específico foi escolhido por possuir maior quantidade de pinos GPIO disponíveis (até 36) e múltiplos periféricos nativos altamente configuráveis, dentre eles 3 interfaces UART, 2 interfaces I2C e 4 interfaces SPI (2 reservadas internamente e 2 de propósito geral).

Sensores Digitais (I2C e UART)

- **Material Particulado (OPC-N3):** Embora o sensor suporte ambas as interfaces, foi selecionado o protocolo I2C para a sua integração. Isso permite poupar canais seriais de hardware do microcontrolador e simplifica o barramento, operando na velocidade padrão (*Standard Mode* de 100 kHz) compartilhado com outros periféricos digitais.
- **Aldeídos (WZ-S):** A comunicação será estabelecida via interface serial UART, utilizando os pinos configuráveis da UART1 do ESP32-S3. A transmissão operará com nível de 3,3 V (compatível com o microcontrolador) atendendo à taxa de transmissão padrão do fabricante de 9600 bps [21].

Sensores Analógicos e Solução para Conversão de Sinais

As estações possuirão 7 sensores com saída analógica (SO2-B4, NO2-B43F, NO-B4, CO-B4, H2S-B4, OX-B431 e VOC-B4), operando através de circuitos de condicionamento de sinal (potenciostatos/amplificadores de alta sensibilidade).

Embora o ESP32-S3 possua 16 canais de conversão analógico-digital (ADC) internos (divididos entre 2 módulos), eles não serão utilizados diretamente devido a duas limitações críticas:

1. **Não-linearidade:** O ADC interno do ESP32-S3 apresenta uma curva de resposta acentuadamente não-linear para voltagens acima de 2,75 V [31].

2. **Ruído de Radiofrequência:** Quando as funções de Wi-Fi ou Bluetooth do ESP32 estão ativas, o ruído elétrico injetado no ADC interno inviabiliza leituras estáveis de sensores de alta sensibilidade [30].
3. **Resolução limitada:** A resolução dos ADCs internos do ESP32-S3 é de 12 bits, o que para nossa aplicação que exige dados com alta resolução para medição de gases em ppm ou $\mu\text{g}/\text{m}^3$ não é suficiente.

Dessa forma, serão adotados dois chips conversores ADC externos ADS1118 [67] da Texas Instruments, conectados aos barramentos SPI de propósito geral do ESP32. Estes chips possuem 4 canais para entradas analógicas e resolução de 16 bits, adequada para leitura de dados precisos.

Estes conversores são acessíveis, sendo amplamente disponíveis em lojas comuns de eletrônica e custam apenas algumas dezenas de reais (à época da elaboração do projeto), portanto não adicionarão um custo significativo.

O ESP32 ainda terá uma interface I2C e duas interfaces UART disponíveis para expansão posterior da estação. Há ainda os múltiplos canais ADC disponíveis, caso deseje-se ler dados que não exigem alta precisão, como condições meteorológica de temperatura e umidade relativa.

Conforme a pinagem do ESP e dos módulos ADS1118, determinamos as conexões necessárias para comunicação dos sensores com o microcontrolador (Figura 6).

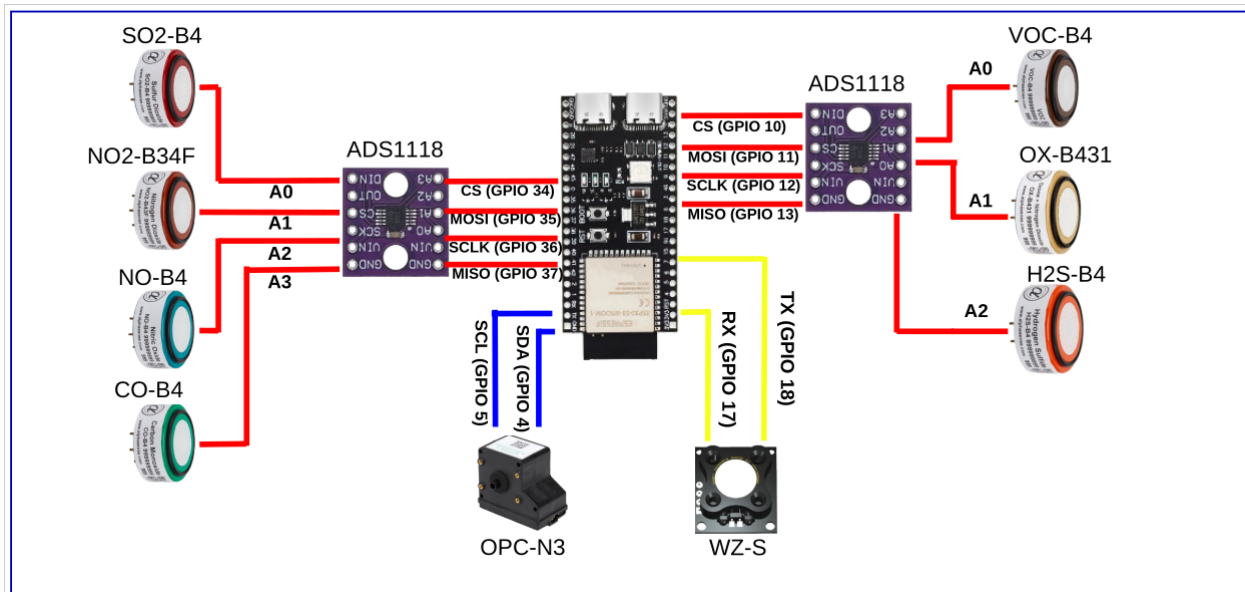


Figura 6: Diagrama da conexão entre os sensores e o microcontrolador

3.2.2 Comunicação ESP32 - TV Box

A integração entre o microcontrolador ESP32 e a TV Box foi estabelecida via interface USB, conectando a porta micro-USB do microcontrolador a uma das portas USB nativas da TV Box para viabilizar a comunicação serial. Essa conexão permitiu o fluxo bidirecional de dados, delegando à TV Box a responsabilidade pelo processo de gravação do *firmware* do ESP32.

O sistema de comunicação foi projetado seguindo o paradigma de protocolos *Controller/Target*, no qual a *TV Box* desempenha o papel de controlador e o microcontrolador atua como dispositivo alvo. Essa escolha foi motivada pelo fato de a *TV Box* concentrar os mecanismos de gerenciamento

do sistema, incluindo o envio de comandos, a coleta de dados dos sensores e a implementação de rotinas de recuperação em caso de falhas de comunicação. Assim, ao centralizar a lógica de controle na *TV Box*, torna-se possível detectar a indisponibilidade do microcontrolador e executar procedimentos automáticos para restabelecer a comunicação, como novas tentativas de conexão, reinicialização da interface serial e repetição dos comandos de aquisição de dados.

O processo de aquisição ocorre de forma cíclica. Em intervalos regulares, a *TV Box* envia pela interface serial o comando `readSensors`, solicitando ao microcontrolador uma nova leitura dos sensores conectados. Como resposta, o microcontrolador retorna uma estrutura no formato JSON contendo os valores medidos. Após o recebimento da mensagem, a *TV Box* realiza a validação do conteúdo recebido, substitui o campo de *timestamp* pelo horário atual em UTC, sincronizado com o sistema operacional da própria *TV Box*, e encaminha a mensagem para o servidor utilizando a conexão TCP previamente estabelecida.

A utilização do formato JSON proporciona flexibilidade para a adição de novos sensores e variáveis sem a necessidade de alterações significativas no protocolo de comunicação. Além disso, trata-se de um formato amplamente suportado por diferentes linguagens de programação e plataformas, facilitando a integração entre os módulos do sistema.

3.2.3 Comunicação TV Box - Servidor

Para a implementação desse sistema (apresentado na Figura 7), foi utilizado um *Docker* independente da aplicação de coleta, projetado para atuar como um serviço genérico de envio de informações ao servidor, não se restringindo, portanto, apenas a dados de sensores.

Esse contêiner expõe uma porta de comunicação TCP, utilizada pelo serviço de coleta de dados e pela rotina de atualizações para o envio de mensagens. O serviço permanece aguardando novas conexões e, sempre que um cliente estabelece comunicação, processa as mensagens recebidas durante aquela sessão. Quando o cliente encerra o envio de dados ou a conexão é finalizada, o *socket* é fechado, liberando os recursos associados e permitindo que novas conexões sejam estabelecidas normalmente. Dessa forma, múltiplos clientes podem utilizar o serviço em momentos distintos, cada um por meio de sua própria conexão TCP.

O grande diferencial desse componente reside na transparência operacional que ele confere aos outros serviços que querem se comunicar. O Coletor de Dados e a rotina de atualização não precisam incorporar lógicas complexas para lidar com instabilidades de rede, autenticação ou falhas no servidor; toda essa complexidade é encapsulada no contêiner. Essa transparência se desdobra em duas ideias fundamentais: (i) a tolerância a quedas de comunicação, baseada em armazenamento local e reenvio de mensagens, e (ii) a segurança da conexão, que abrange autenticação e criptografia dos dados trafegados.

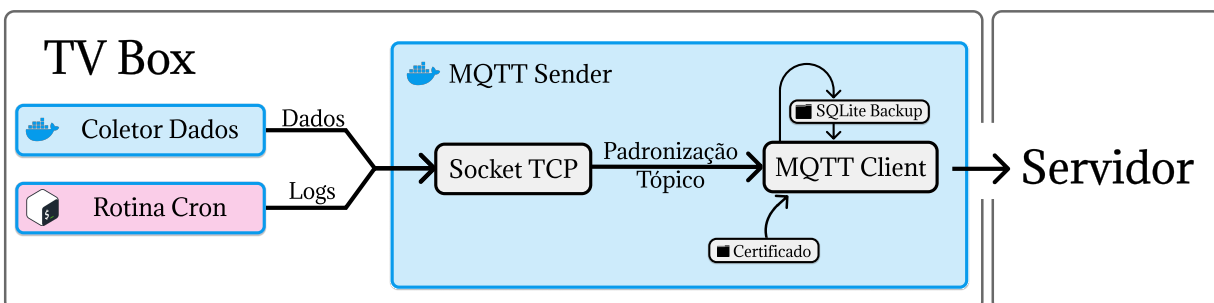


Figura 7: Esquema do MQTT Sender

Essa abordagem é particularmente relevante em ambientes de IoT, nos quais falhas de infraestrutura, como alta latência, interrupções intermitentes e exposição a vetores de ataque, são recorrentes. Ao delegar a resiliência e a segurança a uma única camada, o sistema garante a integridade, a confidencialidade e a disponibilidade das informações sem a necessidade de desenvolvimento desse mecanismo em cada um dos serviços.

Os mecanismos específicos que viabilizam esses dois pilares — o sistema de armazenamento local com reenvio (tolerância a falhas) e o esquema de autenticação e cifragem (conexão segura) — serão detalhados nas subseções seguintes.

Mecanismo de Tolerância a Falhas

Em sistemas IoT, é recorrente o enfrentamento de condições adversas de rede, como perda intermitente ou permanente de conectividade, alta latência e baixa taxa de transferência [33]. Diante desse cenário, torna-se imprescindível a adoção de mecanismos robustos que garantam a entrega confiável das mensagens provenientes dos dispositivos. Para estruturar essa solução, foram considerados três cenários principais de falha: (i) degradação da qualidade da conexão (internet lenta com perda de pacotes); (ii) interrupção total da conectividade à internet no lado do dispositivo; e (iii) indisponibilidade do servidor central (broker MQTT ou bancos de dados).

Para mitigar o primeiro cenário, optou-se pela utilização do nível de Qualidade de Serviço (QoS – *Quality of Service*) 2 do protocolo MQTT. O QoS2 garante que cada mensagem seja entregue ao *broker* uma única vez, sem perdas e nem duplicações, o que é essencial para a integridade e unicidade dos dados coletados, sem a necessidade de uma implementação explícita dessa funcionalidade. Essa escolha se deu em detrimento do QoS1 que, embora mais leve, pode ocasionar duplicidade de mensagens em caso de reenvios, exigindo mecanismos adicionais de deduplicação no servidor. Embora o QoS 2 imponha uma sobrecarga maior ao servidor devido ao *handshake* de quatro etapas (*PUBLISH* - *PUBREC* - *PUBREL* - *PUBCOMP*) [56], no contexto deste projeto, o impacto na latência é praticamente imperceptível, correspondendo a aproximadamente 0,47 ms, 0,50 ms e 0,98 ms, para QoS 0,1 e 2 respectivamente, conforme observado por [52]. Dessa forma, o ganho em confiabilidade proporcionado pelo QoS 2 justifica o pequeno aumento na latência de processamento e a consequente redução da taxa máxima de mensagens processadas por segundo.

Para os cenários de perda total de conectividade (ii) e indisponibilidade do servidor (iii), o QoS 2 isoladamente não é suficiente, uma vez que o *broker* se torna inacessível. Nesses casos, adotou-se o armazenamento local das mensagens em um banco de dados SQLite (Figura 8). A escolha pelo SQLite, em detrimento de uma fila em memória (*buffer*) ou mesmo de uma fila em disco, justifica-se por sua natureza nativa na linguagem Python (via biblioteca *sqlite3*) e pela persistência direta em arquivo, que confere atomicidade e durabilidade às operações. Essa característica garante que os dados não sejam perdidos mesmo em situações de queda abrupta de energia elétrica, conferindo resiliência ao sistema.

Além disso, o arquivo do banco de dados SQLite é mantido em um volume externo ao *Docker*. Dessa forma, mesmo que uma atualização do contêiner seja realizada enquanto ainda há dados pendentes de envio, a nova versão do sistema terá acesso imediato a esses registros e poderá retomar o processo de reenvio exatamente do ponto em que parou, desde que a nova versão espere os dados da mesma maneira que a versão anterior salva. Isso elimina a necessidade de verificações adicionais ou migrações durante o procedimento de atualização, simplificando significativamente esse processo delicado. Em um cenário extremo de perda permanente de conectividade, ainda é possível acessar fisicamente o equipamento e extrair o arquivo do SQLite, assegurando a integridade e a recuperação total dos dados coletados.

Assim que a conectividade é restabelecida ou o servidor retorna à operação, as mensagens

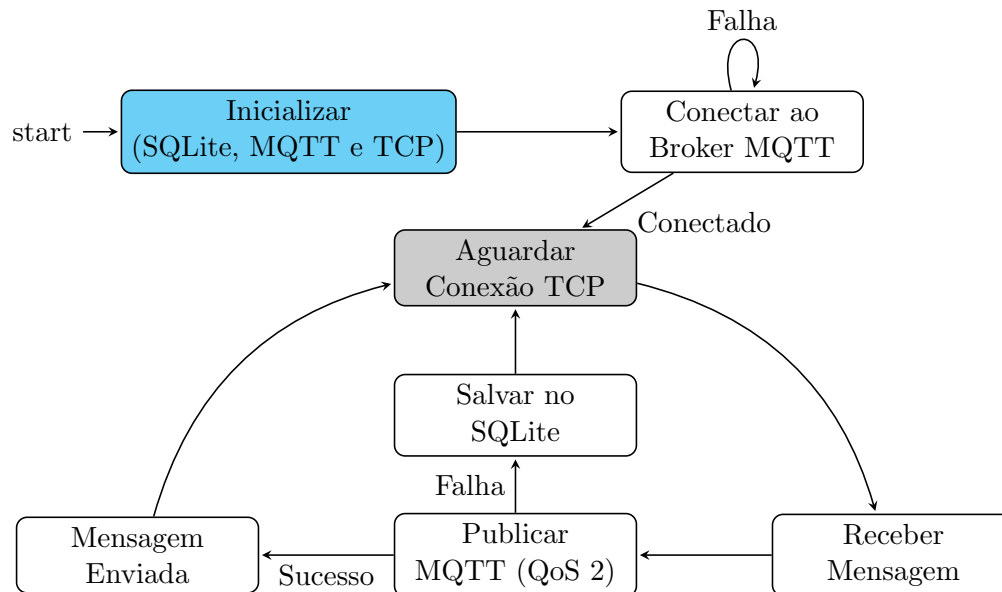


Figura 8: Máquina de estados do módulo de publicação MQTT.

armazenadas localmente são transmitidas ao destino (Figura 9). Para gerenciar esse reenvio de forma eficiente, considerando que o sistema opera sobre uma CPU com suporte a *multithreading*, implementou-se uma *thread* de *background* dedicada. Essa *thread* executa uma rotina periódica que, a cada 30 minutos, consulta o banco de dados em busca de registros pendentes; caso existam, ela dispara o processo de reenvio.

Essa abordagem mantém o fluxo principal de aquisição de dados totalmente desacoplado da lógica de tolerância de falhas. As novas mensagens geradas pelo sistema são publicadas diretamente no *broker*, sem a necessidade de enfileiramento intermediário, o que reduz a latência e a complexidade operacional do caminho crítico de coleta.

Embora os registros sejam armazenados ordenadamente no *SQLite*, o reenvio pode fazer com que mensagens antigas cheguem ao *broker* simultaneamente ou após mensagens mais recentes — especialmente em cenários de queda prolongada, resultando em uma possível entrega fora de ordem cronológica. Para resolver essa questão, cada mensagem carrega consigo um *timestamp* correspondente ao momento exato de sua coleta, tanto no banco de dados local quanto no *payload* transmitido. Dessa forma, o servidor final pode reconstruir a sequência temporal correta independentemente da ordem de chegada.

Aliado ao uso do QoS 2 do protocolo MQTT, esse conjunto de estratégias — *thread* periódica de verificação, registro temporal dos dados e garantia de entrega — assegura a tolerância a falhas mesmo diante de interrupções prolongadas da comunicação. Com isso, o mecanismo de armazenamento e reenvio torna-se completamente transparente para os demais serviços da aplicação. Dessa forma, tanto o Coletor de Dados quanto a rotina de atualização, bem como quaisquer outros serviços implementados futuramente, não precisam incorporar mecanismos próprios para garantir a entrega das mensagens. Ao centralizar essa responsabilidade no mecanismo de tolerância a falhas, a implementação de novos serviços que necessitem se comunicar com o servidor é simplificada, pois a lógica de persistência e recuperação já se encontra encapsulada e disponibilizada de forma padronizada para toda a aplicação. Consequentemente, reduz-se o acoplamento entre os módulos, favorecendo a manutenção, a extensibilidade e a evolução do sistema.

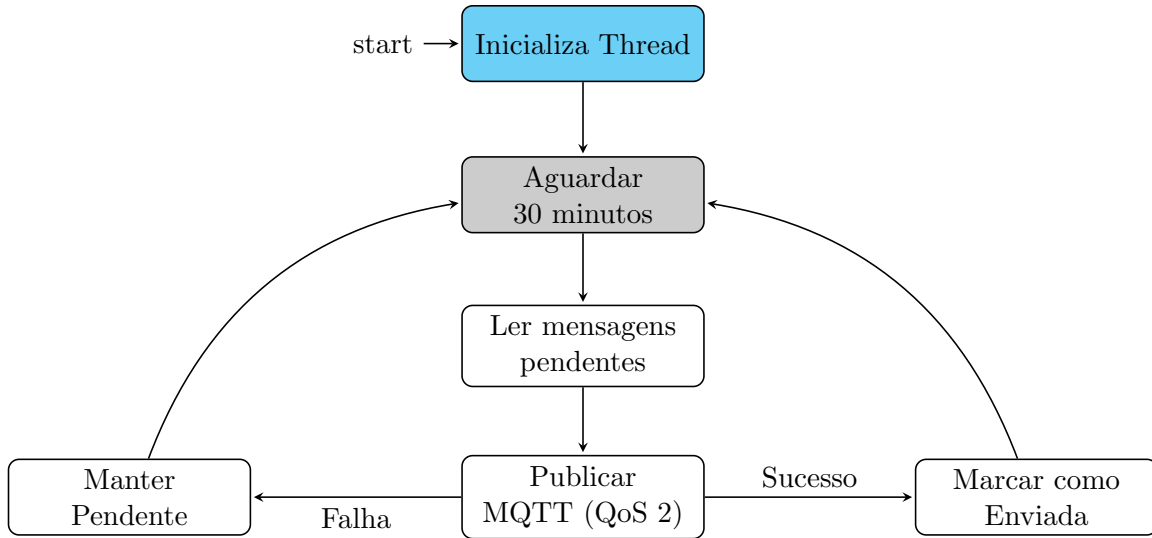


Figura 9: Máquina de estados do processo de retransmissão das mensagens pendentes.

Mecanismo para conexão segura

Para garantir a confidencialidade, integridade e autenticidade dos dados trafegados entre os dispositivos e o servidor, optou-se pelo uso do protocolo MQTT com camada de segurança TLS (*Transport Layer Security*) e autenticação baseada em certificados X.509, em detrimento do tradicional esquema de autenticação por usuário e senha. A escolha por certificados justifica-se pela maior robustez contra ataques de força bruta, pela possibilidade de revogação granular e pela facilidade de gerenciamento em larga escala por meio de uma Infraestrutura de Chave Pública (PKI) [68]. Além disso, o uso de certificados elimina a necessidade de credenciais estáticas e permite a utilização do Distinguished Name (DN) como identificador único do cliente, o que possibilita a definição de políticas de controle de acesso baseadas em ACL (*Access Control List*) [29], restringindo os níveis de acesso que cada cliente possui no *broker*, reduzindo assim a superfície de ataque em caso de comprometimento físico do nó.

Assim como o mecanismo de tolerância a falhas, esse sistema de segurança foi projetado para ser completamente transparente para os serviços que o utilizam. O certificado digital e a lógica de autenticação estão embutidos no módulo *MQTT Sender* (Figura 7), de modo que os demais componentes, como o Coletor de Dados, a rotina de atualização ou serviços futuros, não precisam implementar qualquer lógica específica de segurança, bastando utilizar a interface padronizada de envio de mensagens.

Para o desenvolvimento da arquitetura de segurança, foram considerados dois cenários principais de ameaça: (i) ataques automatizados, como *bots* que varrem a rede em busca de serviços expostos em portas padrão, na tentativa de explorar vulnerabilidades conhecidas ou realizar ataques de negação de serviço (*Denial of Service* – DoS) [66]; e (ii) comprometimento de uma das estações, com possível vazamento ou cópia indevida do certificado e da chave privada armazenados no dispositivo.

Com base nesses cenários, adotaram-se as seguintes medidas:

1. **TLS com autenticação mútua:** Tanto o *broker* quanto o cliente apresentam seus certificados durante o *handshake*, garantindo que ambas as partes sejam autenticadas antes do estabelecimento da comunicação. Para garantir o mais alto nível de segurança, recomenda-se que os certificados sejam gerados internamente em cada dispositivo ou, alternativamente,

transferidos exclusivamente por meios seguros e confinados à rede local, como por meio do *script* `deploy.sh` disponível no repositório².

2. **EKU (*Extended Key Usage*)**: Cada certificado é emitido com `extendedKeyUsage` definida como `clientAuth`, o que restringe sua aplicação unicamente à autenticação de clientes MQTT. Essa separação impede que um mesmo certificado seja empregado indevidamente para outras finalidades, como assinatura de código ou autenticação em serviços distintos.
3. **Lista de Controle de Acesso (ACL) baseada em tópicos e identidade**: O *broker* MQTT é configurado com regras de ACL que estabelecem duas restrições fundamentais:
 - Cada nó autenticado pode publicar mensagens apenas no tópico cujo prefixo contenha o nome (*Common Name* – CN) presente em seu próprio certificado, no formato `nodes/CN_DO_NÓ/[tópicos...]`. Essa restrição limita drasticamente o impacto de um possível vazamento de credenciais: se o certificado do nó denominado, por exemplo, `node_vulneravel` for comprometido, apenas os tópicos pertencentes a esse dispositivo (`/nodes/node_vulneravel/...`) poderão ser afetados. Dessa forma, o problema fica circunscrito a um único nó, isolando os demais dispositivos da rede e facilitando a contenção do incidente.
 - Apenas clientes cujo certificado possua o CN exatamente igual a `consumer` têm permissão de leitura (*subscribe*) em todos os tópicos. O *broker* mapeia o CN do certificado para o nome de usuário da conexão e, por meio da ACL, associa o usuário `consumer` à regra `topic read #`. Dessa forma, as demais estações — que possuem CNs personalizados (ex.: `node_01`) — não conseguem consumir mensagens de outros dispositivos, prevenindo ataques de espionagem interna.

O processo de geração dos certificados é automatizado por meio de *scripts* que empregam o `OpenSSL`, utilizando a curva elíptica `prime256v1` para as chaves e a extensão `subjectAltName` (SAN) preenchida com o identificador do nó (DNS name), assegurando a vinculação inequívoca entre o certificado e o dispositivo [58]. Com esse conjunto de práticas, a arquitetura alcança um nível adequado de segurança para o contexto de IoT em que se insere.

Por uma questão organizacional, os nomes contidos nos certificados seguem os nomes das máquinas (*hostname*) onde são utilizados, o que facilita significativamente a depuração e a rastreabilidade dos dispositivos em ambientes de teste e produção. Contudo, é importante ressaltar que essa escolha é apenas uma convenção: qualquer outro nome poderia ser empregado no certificado, desde que esteja em conformidade com o padrão esperado pelas regras de ACL (*Access Control List*) definidas no *broker*. Essa flexibilidade garante que o sistema possa ser adaptado a diferentes convenções de nomenclatura sem comprometer a segurança ou a funcionalidade da infraestrutura.

Portanto, garante-se, dessa maneira, um sistema robusto do ponto de vista de segurança. A escrita no *broker* só é possível mediante a apresentação de um certificado cujo nome (CN) esteja previamente aceito pelas regras de ACL, eliminando a necessidade de autenticação por senha, abordagem que, além de mais suscetível a ataques de força bruta, exige gerenciamento contínuo de credenciais. Mesmo em um cenário adverso onde um certificado de nó seja comprometido, o dano fica restrito ao tópico específico daquele dispositivo, não sendo possível contaminar o banco de dados por completo ou interferir na operação das demais estações.

Adicionalmente, a restrição imposta pelo EKU (*Extended Key Usage*) e pela ACL garante que apenas certificados do tipo *consumer* possuam permissão de leitura no *broker*. Isso impede que

²Esta prática evita que as chaves privadas trafeguem por redes públicas, reduzindo significativamente o risco de interceptação e comprometimento da identidade do dispositivo.

qualquer certificado de nó, ainda que válido, seja utilizado para consumir dados de outros dispositivos, assegurando que, em nenhum momento, um certificado diferente do MQTT Consumer consiga acessar informações sensíveis. Em nível de produção, é possível inclusive utilizar CN aleatórios nos certificados, desde que estejam em conformidade com o padrão esperado pela ACL, dificultando ainda mais a ação de um atacante que, mesmo obtendo um certificado válido, não terá conhecimento prévio dos tópicos autorizados ou das permissões concedidas. Essa camada adicional de ofuscação, aliada à autenticação mútua via TLS, confere à arquitetura um nível de segurança compatível com as exigências de ambientes IoT críticos.

3.2.4 Armazenamento

Nesta subseção, o foco recai sobre o *MQTT Consumer* (apresentado na Figura 10). Tendo em vista a discussão prévia acerca da segurança — na qual foram tratados a restrição do CN *consumer* e a gestão dos certificados —, esta seção concentra-se exclusivamente no tratamento dos dados propriamente ditos, abstendo-se dos mecanismos de comunicação e de autenticação já explorados anteriormente.

O *MQTT Consumer* é responsável, basicamente, por separar os diferentes tipos de mensagens provenientes das estações — categorizadas como **data**, **info**, **error** e **log** — e direcioná-las ao respectivo banco de dados ou, no caso dos *logs*, gerar alertas (warnings) no *Docker* para análise posterior.

O formato das mensagens recebidas segue a estrutura `nodes/<identificador_do_no>/<tipo>`, contendo um *payload* e um *timestamp*. De forma mais específica, as mensagens do tipo *data* utilizam um subtópico para indicar o sensor de origem, como **temperatura**, **umidade**, **pressao**, **co2**, etc. Nesse caso, o *payload* carrega o valor medido e o *timestamp* registra o momento exato da coleta.

Já as mensagens dos tipos *info* e *error*, no atual estágio de desenvolvimento, empregam o subtópico *update* para comunicar o *status* do processo de atualização. Aqui, o *payload* contém o conteúdo do *log* e o *timestamp* indica o instante do evento.

Por fim, o tipo *info* foi concebido para armazenar, futuramente, informações gerais sobre o estado do sistema. Até o momento, ele é utilizado apenas para que o nó informe sua inicialização e a versão do *firmware*; contudo, a arquitetura permite a inclusão de novos registros em implementações futuras, tais como *logs* de temperatura da CPU, uso de memória, capacidade de armazenamento e outras métricas de sistema.

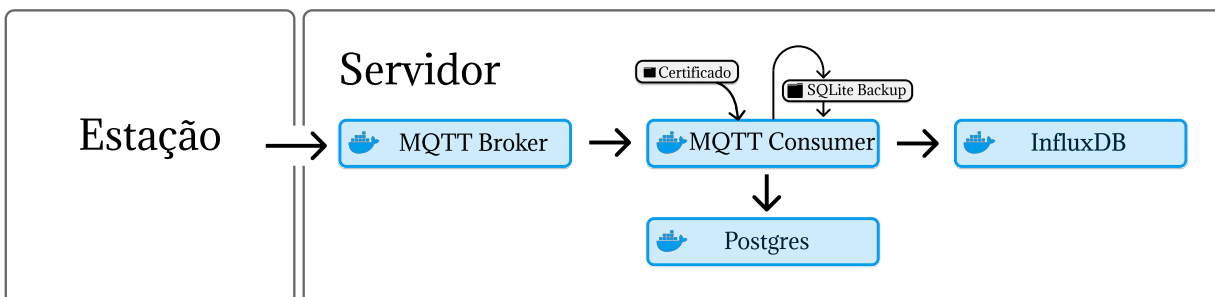


Figura 10: Esquema do Servidor MQTT

Bancos de dados

As informações desse sistema podem ser divididas em dois grandes grupos de dados: os dados provenientes dos sensores (dados temporais) e os dados mais estáticos, que possuem pouca variação ao longo do tempo — como o momento em que a versão foi instalada, a data da última atualização do sistema e a localização dos dispositivos, por exemplo. Em razão dessa distinção de características, optou-se pela utilização de dois bancos de dados distintos para lidar com essas especificidades: o *InfluxDB* e o *PostgreSQL*. Essa escolha se deu pelo fato de que cada um desses sistemas é nativamente otimizado para o respectivo perfil de dados — o primeiro é um banco de dados de séries temporais, projetado para alta taxa de ingestão e consultas temporais [40]; o segundo é um banco relacional, adequado para dados estruturados com relacionamentos e atualizações esporádicas [61]. Dessa forma, garante-se eficiência de escrita, compactação de armazenamento e desempenho de consulta para cada tipo de informação.

O fluxo de escrita dos dados temporais no *InfluxDB* inicia-se com o recebimento da mensagem proveniente do nó. Primeiramente, extrai-se o instante de coleta (*timestamp*) contido no *payload* da mensagem; caso esse campo não esteja presente, utiliza-se o momento atual do sistema como referência. Em seguida, verifica-se a existência do campo numérico (valor) no *payload*. Se ele estiver ausente ou não puder ser convertido para um número de ponto flutuante, o processo é interrompido e um alerta é gerado nos registros de *log*, evitando que dados inconsistentes sejam persistidos.

Já o fluxo de escrita dos dados estáticos no *PostgreSQL* pode se dar de algumas maneiras: (i) de maneira manual, por exemplo, para cadastrar os dispositivos no sistema com suas respectivas informações básicas, como localização, identificador único e metadados de instalação; e (ii) por meio do *MQTT Consumer*, que extrai dos tópicos de *log*, *info* e *error* o estado atual do sistema — classificando-o como estável, em atualização ou incerto — e registra os horários correspondentes a essas transições de *status*. Além disso, esse mesmo mecanismo permite armazenar informações mais elaboradas, como a associação entre um determinado dispositivo e a versão de código que ele está executando no momento.

O banco de dados encontra-se estruturado conforme o esquema apresentado na Figura 11. Esse sistema visa contemplar as informações básicas da TV Box, que estão contidas na entidade *dispositivo*; as versões dos códigos utilizados, armazenadas em *código*; e os sensores, registrados em *sensor*. Essas tabelas possuem relações entre si: o dispositivo relaciona-se ao código por meio da tabela de atualização, permitindo um histórico de versões daquele dispositivo; já o dispositivo conecta-se à tabela de sensores pela tabela de conexão, fornecendo a lista de sensores que aquele dispositivo já possuiu ou possui atualmente (o que pode ser verificado por meio da *flag ativo*). Além disso, existem tabelas de manutenção tanto para o dispositivo quanto para o sensor, a fim de registrar um histórico detalhado de intervenções e eventos.

Com o banco de dados estruturado dessa forma, é possível criar correlações básicas para identificar onde o dado presente no banco de séries temporais foi fisicamente coletado, saber qual sensor foi usado em um determinado período de tempo, qual era a versão do código em execução naquele instante ou, até mesmo, por meio dos campos *status* e *timestamp_status*, identificar o momento exato em que ocorreu um erro ou uma instabilidade no sistema. Além dessas consultas básicas, é possível realizar análises mais elaboradas, como cruzar os registros de falhas com as versões de código para identificar padrões de instabilidade recorrentes, correlacionar a vida útil dos sensores com as manutenções realizadas ao longo do tempo, ou ainda rastrear a movimentação geográfica e administrativa dos dispositivos, associando medições específicas às condições de instalação e à configuração de *firmware* vigentes em cada período.

Nesse ponto, a maior quantidade de escritas sempre ocorrerá no *InfluxDB*, em razão do elevado número de tópicos por nó — dez tópicos (conforme discutido na Seção 2) - e porque a periodicidade

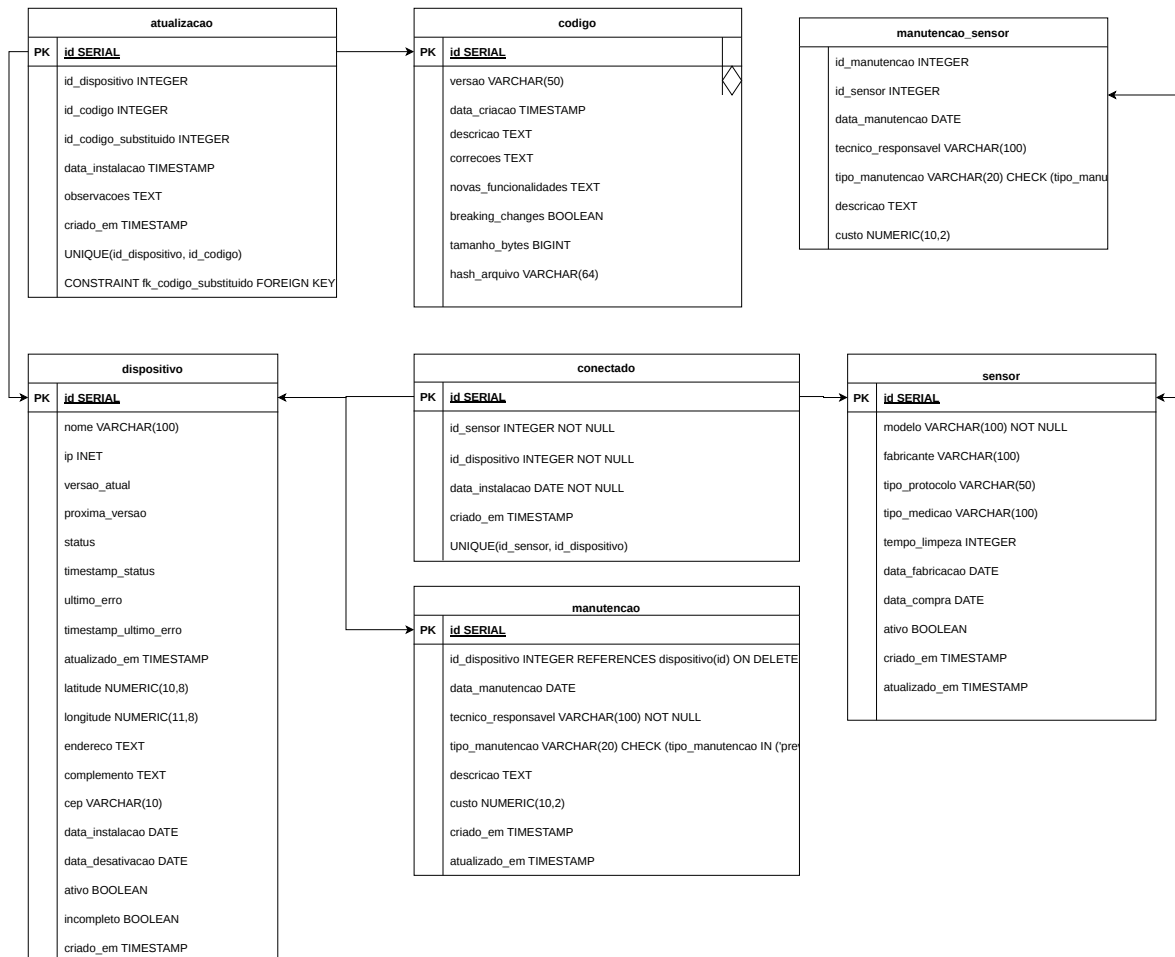


Figura 11: Esquema do banco de dados PostgreSQL

de envio dessas informações é significativamente maior do que a dos registros (*logs*) provenientes das atualizações.

Considerando a estimativa para a fase de testes em São Paulo, apresentada na Seção 4, segundo a qual 50 estações seriam suficientes para cobrir mais de 90% da população, e assumindo que todas as estações transmitam dados simultaneamente, a infraestrutura deverá suportar uma taxa de escrita de aproximadamente 500 registros por segundo, correspondentes aos dez tópicos publicados por cada uma das 50 estações.

Nesse cenário, a carga permanece muito inferior à capacidade reportada para o **InfluxDB**. Segundo o benchmark **SciTS**, o banco de dados foi capaz de sustentar aproximadamente 740 mil escritas por segundo em um teste de longa duração, alcançando picos próximos de 790 mil escritas por segundo em cenários de alta concorrência [53]. Dessa forma, seguindo os resultados apresentados por esse estudo, conclui-se que a arquitetura proposta apresenta capacidade suficiente para operar na cidade de São Paulo.

Mecanismo de Tolerância a Falhas

Como uma possível falha nesse sistema, considerou-se a situação em que os contêineres dos bancos de dados (**InfluxDB**) estejam em manutenção, passando por atualizações ou enfrentando alguma instabilidade. Para lidar com essa possibilidade, adotou-se a mesma estratégia discutida na Seção 3.2.3: caso os bancos de dados estejam indisponíveis, os dados recebidos são armazenados localmente em um banco **SQLite**, nativo do sistema e com persistência garantida em disco. Assim que o sistema retorna à operação normal, esses dados são enviados para o **InfluxDB**, garantindo a consistência eventual e a resiliência do sistema contra interrupções temporárias.

Diferentemente do que acontece no nó, o processo de salvar as mensagens no **SQLite** no lado do MQTT Consumer pode gerar um grande volume de dados, especialmente se o sistema ficar inativo por horas ou dias. Além disso, por uma questão de redundância, as mensagens salvas no **SQLite** não são apagadas quando são efetivamente enviadas aos bancos de dados definitivos, em vez disso, uma *flag* é ativada em cada registro, indicando que ele já foi processado, evitando assim o reenvio indevido de mensagens.

Entretanto, como essa estratégia pode acumular um grande volume de resíduos ao longo do tempo, o MQTT Consumer conta com uma *thread* dedicada que, de tempos em tempos, acessa o **SQLite** e apaga as mensagens mais antigas. Esse comportamento é totalmente ajustável, podendo ser configurado conforme a necessidade específica do sistema e a capacidade de armazenamento disponível, garantindo um equilíbrio entre rastreabilidade histórica e uso eficiente do espaço em disco.

3.2.5 Exportação

A etapa de exportação de dados tem como objetivo extrair e estruturar as informações ambientais armazenadas nos bancos de dados (**InfluxDB** e **PostgreSQL**) em formatos que facilitem a análise externa. O sistema permite ao usuário selecionar um conjunto de distritos municipais, um período de interesse e os sensores desejados, por exemplo, temperatura, umidade e CO_2 . A partir dessas escolhas, a rotina realiza uma série de processamentos espaciais e estatísticos antes de gerar os arquivos finais.

Inicialmente, o sistema obtém todos os dispositivos cadastrados no **PostgreSQL**, com suas respectivas coordenadas geográficas. Utilizando os *shapefiles* de distritos disponibilizados pela plataforma GeoSampa (Prefeitura de São Paulo), é calculada a cobertura de cada distrito selecionado, com base no raio de cobertura que o usuário deseja considerar. A união de todos os raios de cober-

tura é então intersectada com cada distrito, e a área de interseção é comparada com a área total do distrito, caso essa proporção for igual ou superior a um limiar definido pelo usuário, o distrito é classificado como *coberto* (azul no mapa); caso contrário, como *não coberto* (vermelho).

Em seguida, o sistema identifica quais dispositivos efetivamente contribuíram para a cobertura dos distritos classificados como cobertos. Para cada dispositivo, verifica-se se o seu raio individual cobre pelo menos 10% da área de algum distrito coberto. Aqueles que atendem a esse critério são considerados *dispositivos selecionados*. Os demais são descartados do processo de extração de séries temporais, esse escolha se dá para ter um critério de área mínima coberta, pois não é de interesse dispositivos que estejam muito distantes dos distritos de interesse.

Para cada dispositivo contribuinte, o sistema consulta o InfluxDB em busca das séries temporais dos sensores escolhidos, dentro do intervalo de datas fornecido pelo usuário. Os dados são então utilizados para gerar dois produtos:

- **Gráficos individuais:** para cada dispositivo, são plotados gráficos de linha com as medições ao longo do tempo, um para cada sensor selecionado. Os gráficos são salvos em formato PNG, organizados em subdiretórios nomeados conforme o distrito ao qual o dispositivo está associado.
- **Arquivos CSV:** os dados temporais de todos os sensores para um mesmo dispositivo são consolidados em um único arquivo CSV, com colunas para timestamp e cada sensor, facilitando a importação em ferramentas de análise estatística ou planilhas.

Por fim, o sistema gera um mapa temático em PDF que sintetiza toda a análise espacial. Nesse mapa, os distritos selecionados são exibidos com cores representando sua classificação de cobertura (azul para coberto, vermelho para não coberto). São desenhados os raios de cobertura para todos os dispositivos que contribuíram com dados para as regiões cobertas, dessa maneira permitindo uma visualização prática do que a consulta forneceu.

A Figura 12 apresenta um exemplo de saída desse mapa, ilustrando a distribuição espacial dos distritos e a contribuição individual de cada dispositivo para a cobertura. No exemplo mostrado, foram selecionados os distritos de *Socorro*, *Saúde*, *Butantã*, *Freguesia do Ó*, *Penha*, *Parelheiros*, *José Bonifácio* e *Anhanguera*. A partir dos dados coletados, um total de 13 dispositivos foram identificados como contribuintes, ou seja, seus raios cobriram pelo menos 10% da área de algum desses distritos, sendo, portanto, utilizados para a geração dos gráficos e arquivos CSV. Os demais dispositivos, embora plotados no mapa em cinza, não tiveram suas séries temporais exportadas, pois não atingiram o limiar mínimo de contribuição para a cobertura dos distritos selecionados.

Esse fluxo de exportação, embora ainda não possua uma interface gráfica interativa, já fornece um conjunto de dados estruturados e visualizações prontas que podem servir de base para outros sistemas. Um exemplo notório é o programa Qualar da Companhia Ambiental do Estado de São Paulo (CETESB), que disponibiliza um mapa interativo para navegação e seleção de estações de monitoramento da qualidade do ar [15, 16]. Nessa plataforma, ao selecionar uma estação de monitoramento, o usuário tem acesso aos dados das últimas 24 horas por meio de uma interface intuitiva e de fácil consulta pública [16].

O sistema aqui proposto, por sua vez, embora mais simples e desprovido de uma interface gráfica para o usuário final, oferece uma camada adicional de riqueza informacional. Diferentemente da plataforma da CETESB, que disponibiliza dados padronizados e resumidos para consulta rápida, nossa ferramenta permite uma consulta mais especializada e flexível, com a geração de gráficos individuais por dispositivo e a exportação de séries temporais completas em CSV. Isso possibilita ao pesquisador realizar análises estatísticas aprofundadas, correlações com eventos de saúde pública

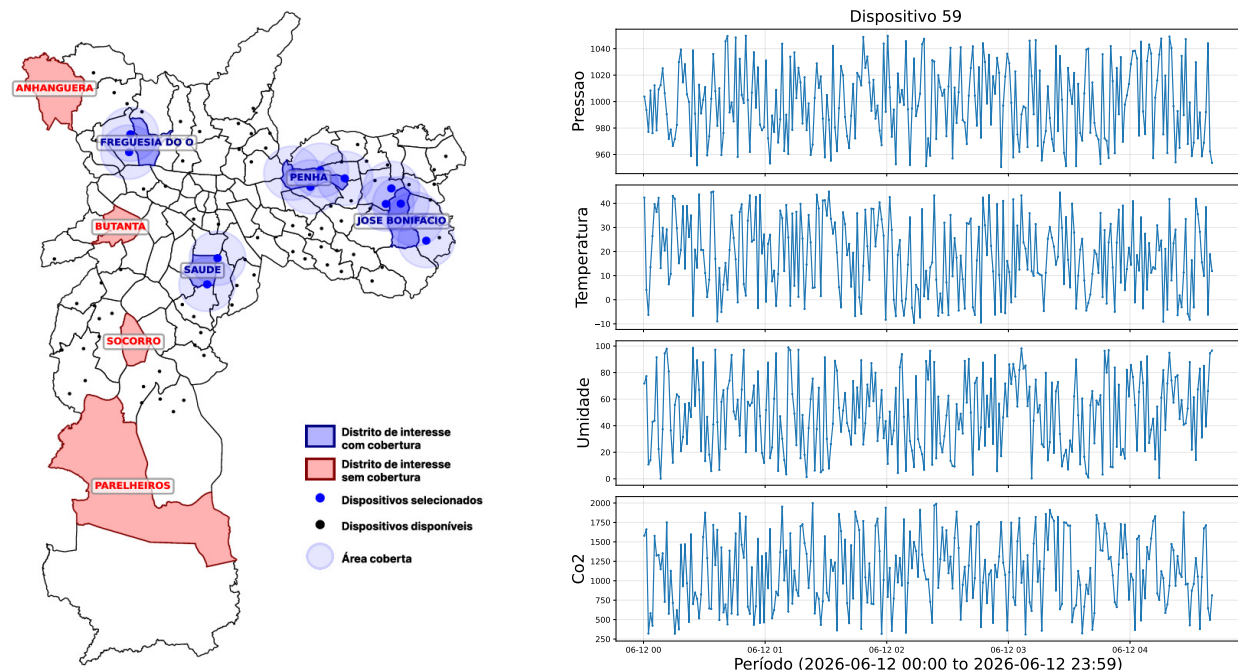


Figura 12: Interface de visualização do sistema, composta pelo mapa de cobertura dos dispositivos selecionados (à esquerda), com os distritos cobertos destacados em azul e os não cobertos em vermelho, e pelas séries temporais das variáveis monitoradas por um dos dispositivo durante o período selecionado (à direita).

e estudos de séries temporais que vão além da visualização imediata oferecida pelos sistemas de monitoramento convencionais.

Portanto, por mais que seja um sistema bruto e simples, ele já fornece o subsídio necessário para estudos ambientais, correlações com dados de saúde pública e suporte à decisão de políticas de monitoramento da qualidade do ar. Além disso, por ser mais rico para pesquisadores, permite uma consulta mais especializada e personalizada, atendendo a demandas que vão além da consulta pública oferecida por plataformas como o Qualar da CETESB.

3.3 Gerenciamento do Sistema

Para garantir a operação contínua, segura e atualizada da rede de estações de monitoramento, o sistema incorpora um módulo de gerenciamento que atua em duas frentes complementares: a atualização remota dos dispositivos de campo e *backup* periódico dos dados armazenados no servidor. Esses mecanismos foram concebidos para funcionar de forma autônoma, minimizando a necessidade de intervenção humana e assegurando que tanto as estações quanto os repositórios centrais se mantenham íntegros e em conformidade com as versões de software vigentes.

O gerenciamento das estações é efetuado por meio de um fluxo de atualização que combina um servidor HTTPS com rotinas agendadas localmente via *cron*. Esse fluxo permite que novas versões dos contêineres *Docker* e do *firmware* do ESP32 sejam distribuídas de maneira controlada, com verificação de integridade, mecanismos de reversão em caso de falha e envio de registros de *status* ao servidor, facilitando o diagnóstico remoto. Paralelamente, no lado do servidor, o fluxo de backup utiliza o agendador *Ofelia* para realizar cópias diárias dos bancos de dados de séries temporais (*InfluxDB*) e relacionais (*PostgreSQL*), armazenando-as em volumes externos aos

contêineres. Essa estratégia garante a preservação dos dados mesmo diante de falhas nos serviços ou de erros operacionais, consolidando uma base de informações confiável para análises futuras.

Nas subseções a seguir, cada um desses fluxos é detalhado, evidenciando as decisões de projeto, os mecanismos de tolerância a falhas e a integração com os demais componentes da arquitetura.

3.3.1 Fluxo de Atualização

Para a construção do mecanismo de atualização, foi utilizado um servidor HTTPS que hospeda as versões compactadas do sistema juntamente com seus respectivos *hashes* de integridade, além de uma rotina automatizada nas TV Boxes por meio do agendador *cron*. Dessa forma, manutenções corretivas e evolutivas podem ser realizadas de maneira remota, automatizada e transparente, sem a necessidade de intervenção física nos dispositivos, garantindo agilidade e escalabilidade na gestão da frota.

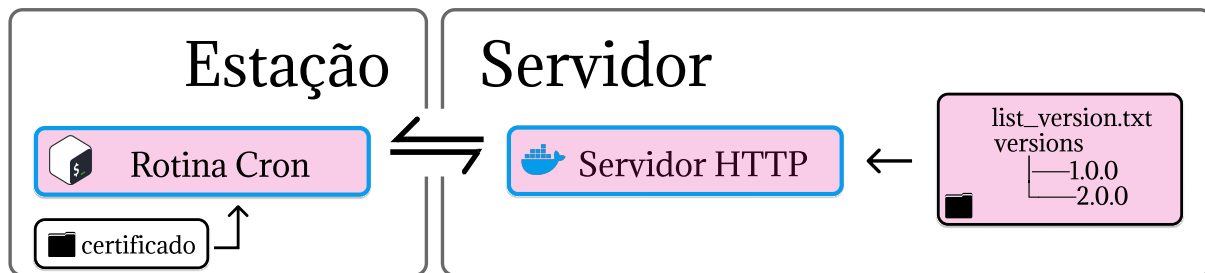


Figura 13: Esquema geral do fluxo de atualização

Atualização

O fluxo de atualização está representado na Figura 14. O sistema permanece em estado ocioso até que a rotina definida pelo *cron* seja executada. O *cron* é um serviço do sistema operacional responsável pelo agendamento da execução automática de tarefas em intervalos predefinidos [17]. A periodicidade dessa atualização pode ser ajustada conforme o intervalo desejado entre as verificações, sem a necessidade de acessar o dispositivo de forma remota ou física.

Após o disparo da rotina, o nó acessa o servidor e baixa um arquivo chamado *list_version.txt*, o qual possui o seguinte formato:

```
node01=y,v1.0
node02=n,v2.0
node03=y,v1.0
```

Nesse arquivo, o campo antes do sinal de igual corresponde ao nome do dispositivo; a letra *y* indica a necessidade de atualização, enquanto *n* indica que não é necessário; o conteúdo após a última vírgula corresponde ao nome da versão. Após verificar a necessidade de download, o nó começa a baixar um arquivo ZIP contendo o novo sistema, juntamente com o arquivo de hash correspondente. Nesse ponto, o sistema realiza até três tentativas para contornar possíveis instabilidades no servidor ou arquivos corrompidos, validando a integridade do pacote por meio da verificação do hash SHA-256.

Em caso de sucesso na validação do hash, a atualização é extraída e o *script* de instalação (`install.sh`) é executado. Esse *script* possui uma máquina de estados própria (Figura 15), que será discutida na próxima seção. Ao final da execução, a rotina de instalação retorna um código indicando sucesso (0) ou uma falha específica (códigos como 10, 20, 30, etc.), permitindo identificar com precisão a etapa em que o problema ocorreu.

Durante todo esse processo, mensagens de log são enviadas ao servidor para auxiliar na detecção e no diagnóstico de falhas, como, por exemplo: (i) atualização iniciada com sucesso; (ii) falha na validação do hash (arquivo corrompido ou incompatível); e (iii) rotina executada, porém nenhuma atualização é necessária.

Dessa maneira, o sistema permite que o gerente da rede acompanhe o estado da infraestrutura e identifique a localização dos erros. Como discutido anteriormente, esses logs, ao chegarem ao servidor, são direcionados para o banco de dados estático. Com esse histórico consolidado, é possível reportar as ocorrências à equipe de desenvolvimento, facilitando a correção de falhas e a realização de manutenções proativas.

Vale destacar que a rotina de atualização não depende de nenhum outro componente do sistema IoT para sua execução. Embora utilize o *sender* para tentar enviar os registros durante o processo, a existência ou indisponibilidade desse serviço, ou mesmo eventuais falhas no envio dos *logs*, não interrompem o fluxo principal, uma vez que a prioridade é garantir a atualização do dispositivo, e não o envio dos *logs*. Além disso, caso uma versão específica contenha erros, isso não compromete a execução das próximas versões, pois a rotina será executada novamente na próxima ativação do agendador, tentando baixar e instalar a versão mais recente disponível.

Por fim, essa rotina foi projetada para ser extensível e pode ser utilizada para atualizações mais amplas do sistema. O *script* `install.sh` pode ser modificado conforme a necessidade, aplicando mudanças no nó que vão além dos containers, como a atualização de pacotes do sistema operacional, a correção de permissões ou a alteração de configurações de rede. Assim, o mecanismo de atualização torna-se uma ferramenta centralizada e flexível para a gestão remota, garantindo que os dispositivos permaneçam atualizados, seguros e em conformidade com os requisitos operacionais ao longo do tempo.

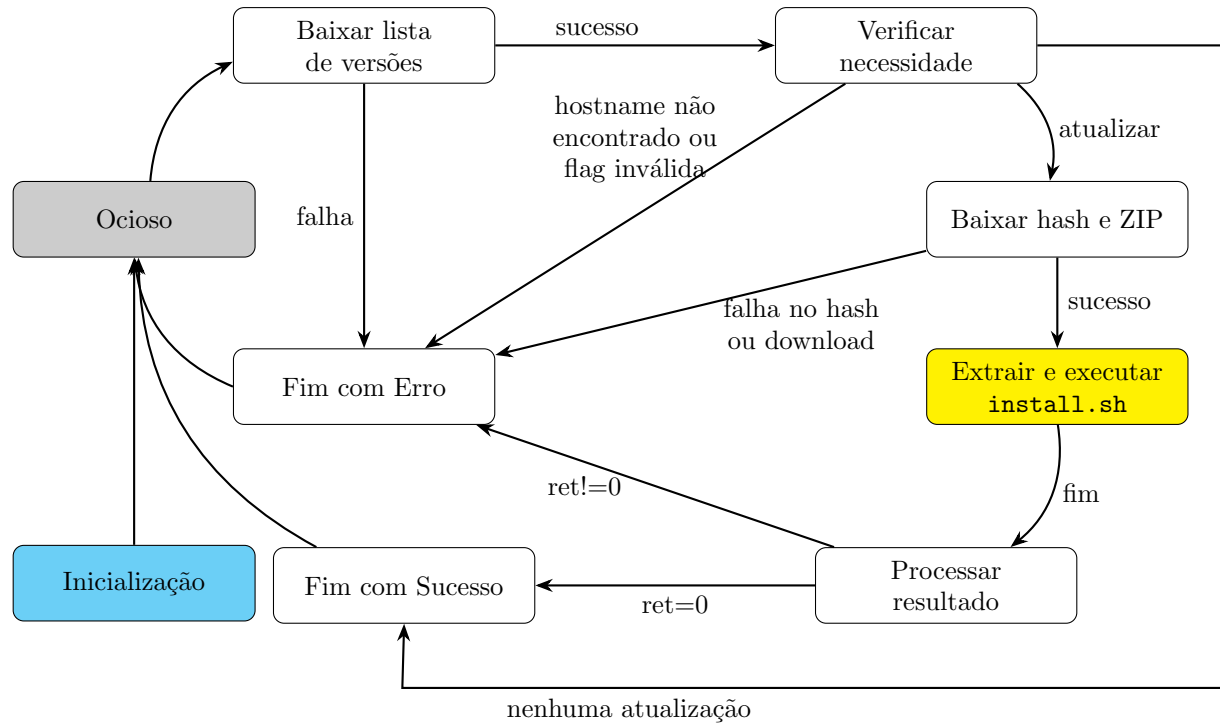


Figura 14: Máquina de estados do fluxo de verificação e atualização (update.sh)

Instalação

Essa rotina é responsável, de fato, por realizar a instalação da atualização nos dispositivos. Ela está representada de forma simplificada na Figura 14 pelo estado em amarelo, mas, na realidade, constitui uma Máquina de Estados completa por si só, a qual é apresentada em detalhes na Figura 15. O sistema foi projetado dessa maneira, contando com um *script* para gerenciar a atualização e outro para executar a instalação. Isso permite que sistemas completamente distintos sejam instalados sem problemas, pois a atualização atua como a porta de entrada para a instalação, e esse *script* pode especificar todas as medidas necessárias para o processo, sem que haja a necessidade de alterar o *script* de atualização.

Essa rotina começa com o backup dos dockers atuais, que serão usados como ponto de retorno caso o sistema apresente algum problema. Em seguida, apaga os arquivos dos dockers atuais para dar lugar aos novos e faz a cópia dos certificados utilizados para a comunicação com o servidor. Vale notar que esses certificados são gerados uma única vez e reutilizados ao longo de várias atualizações. Posteriormente, é criada a rede de comunicação entre os dockers, dando início à parte crítica do processo: a parada dos dockers do sistema atual.

Nesta etapa, com os contêineres parados e a comunicação serial USB liberada, ocorre o processo de gravação do firmware do ESP32. Para que a gravação ocorra durante a etapa de atualização do sistema, a TV Box gerencia as linhas de controle para aplicar um sinal de reset no ESP32, forçando sua transição para o modo de bootloader. Nesse estágio, o script de atualização executa a ferramenta computacional *esptool.py*, um utilitário em Python desenvolvido pela Espressif, que recebe como parâmetros as especificações de leitura, escrita e configuração do chip microcontrolador.

A partir desse ponto até o final da rotina, a *flag de Rollback* permanecerá ativa. Dessa forma, diante de qualquer falha, o sistema tentará reverter para a versão anterior (o backup). Contudo,

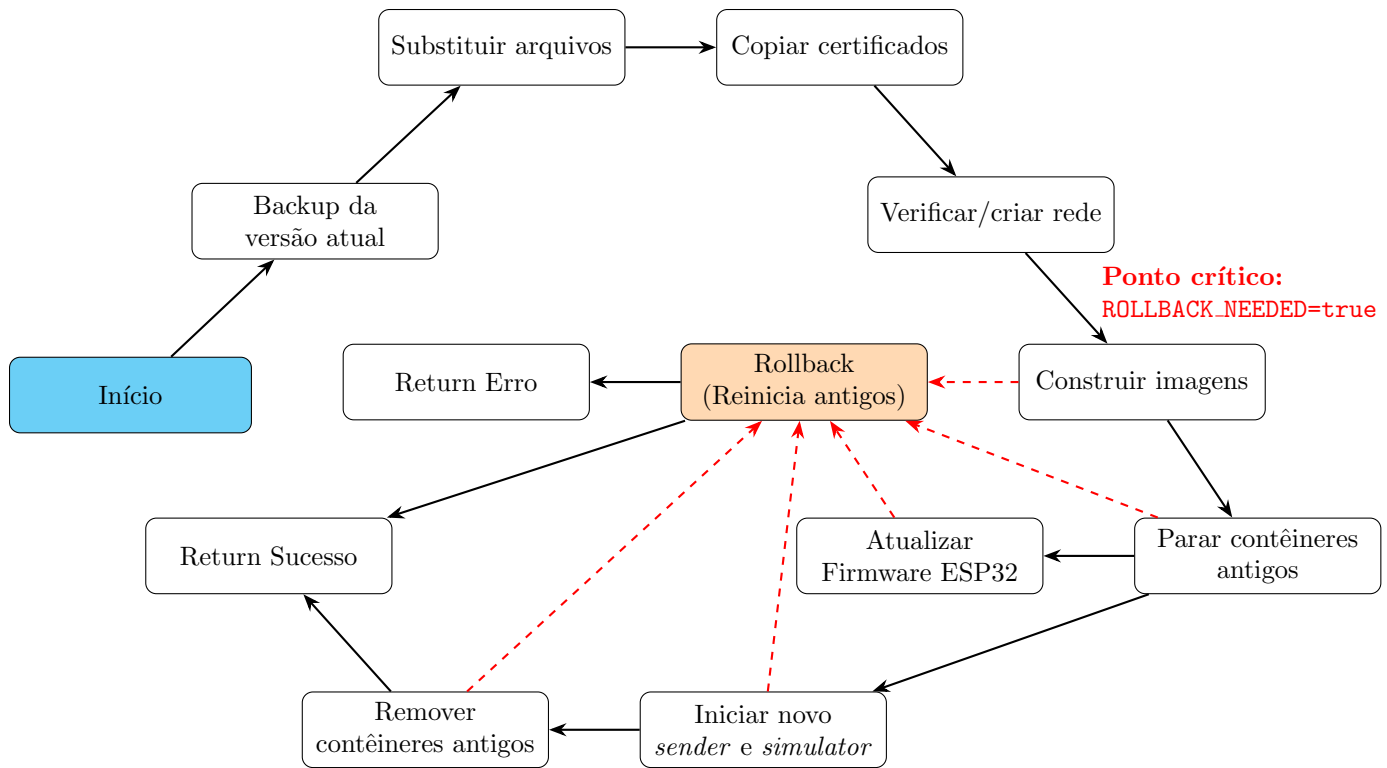


Figura 15: Máquina de estados do fluxo de instalação com mecanismo de *rollback* e retorno ao estado ocioso

dependendo do estágio em que o processo se encontrar, pode ser suficiente apenas reiniciar os dockers, uma vez que o sistema *Docker* pode ser parado e removido — operações conhecidas como *stop* e *down*, respectivamente [28]. Isso permite que, em caso de falha, o sistema retome a execução com o comando *start* ou, se necessário, tente reiniciar de fato utilizando os arquivos dos dockers.

Ao final da execução, a rotina retorna códigos numéricos que indicam o resultado da instalação, como, por exemplo: o código 10 (falha no build do *sender*), o código 30 (falha ao iniciar o novo contêiner *sender*) e o código 41 (novo contêiner *simulator* não apareceu no *docker ps*). Ademais, pode ocorrer um erro desconhecido, quando o valor capturado do retorno não corresponde a nenhum dos casos já catalogados; ainda assim, o código de erro é enviado ao servidor para tentar identificar o problema. Com base nesses códigos, a rotina consegue identificar exatamente qual etapa da atualização falhou e, assim, decidir a melhor estratégia de recuperação — seja reiniciar os contêineres, repetir uma etapa específica ou acionar o *rollback* completo para a versão anterior.

Em síntese, a rotina de instalação orquestra a atualização tanto dos serviços Docker na TV Box quanto do firmware no microcontrolador ESP32. O mecanismo de *rollback* atua como um salva vidas em que se houver falha no *build* das imagens, falta de comunicação de rede, ou problemas na gravação serial do microcontrolador, o sistema reverte para o estado anterior de forma autônoma, assegurando a operação ininterrupta do monitoramento ambiental sem a necessidade de uma intervenção manual no local da instalação.

3.3.2 Backup

O sistema discutido até o momento é suficiente para ser aplicado, mas ainda poderia haver falhas humanas durante o manuseio do sistema e gerar perda dos dados. Portando foi desenvolvido um

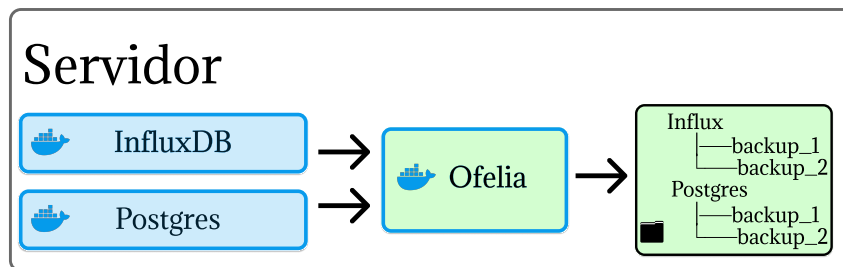


Figura 16: Esquemático do sistema de Backup

mecanismo de backup dos dados gerados afim de mitigar imprevistos futuros.

O sistema consiste em um *Docker* que executa o *Ofelia*, um agendador de tarefas moderno e de baixo consumo de recursos para ambientes containerizados. A ferramenta permite a execução periódica de comandos e *scripts* por meio de uma sintaxe semelhante à do cron, além de disponibilizar mecanismos para gerenciamento de *logs* e tratamento de falhas durante a execução das tarefas. Sua utilização mostra-se adequada para o cenário de instalação e atualização proposto, pois desacopla completamente o agendamento da lógica principal da aplicação, permite a configuração por meio de arquivos de fácil manutenção e utiliza a *Docker* API para executar comandos em contêineres existentes ou criar contêineres efêmeros sob demanda para cada tarefa, eliminando a necessidade de manter serviços cron ou dependências adicionais nos contêineres principais [18].

O contêiner *Ofelia* é responsável por duas rotinas agendadas automáticas, ambas executadas diariamente no período noturno. A primeira tarefa ocorre às 23h55 e realiza o *backup* do banco de dados *InfluxDB* (série temporal). A segunda tarefa ocorre às 23h59 e realiza o *backup* do banco de dados *PostgreSQL* (relacional). Esses horários foram estrategicamente escolhidos para o final do dia, minimizando o impacto sobre operações críticas e garantindo que os dados sejam preservados diariamente.

Do ponto de vista da infraestrutura e da segurança, a configuração dos *volumes* no arquivo *docker-compose* foi projetada para atender três ideias principais: persistência, isolamento e segurança, conforme detalhado a seguir:

- **Volumes nomeados para dados internos** – Os volumes são gerenciados diretamente pelo Docker, eles armazenam os dados permanentes de cada serviço, garantindo que as informações não sejam perdidas durante recriações, atualizações ou reinicializações dos contêineres. Essa abordagem isola o ciclo de vida dos dados do ciclo de vida dos contêineres, tornando as manutenções muito mais seguras e previsíveis.
- **Montagens para *backups*** – Os diretórios dos backups são montados diretamente do sistema hospedeiro para os contêineres. Isso centraliza todos os arquivos de *backup* em um local fixo e de fácil acesso para a administração, permitindo que operadores externos possam recuperar ou consultar esses arquivos sem precisar acessar o interior dos contêineres.
- **Montagem do socket do *Docker* no *Ofelia*** – O arquivo `/var/run/docker.sock` é montado no contêiner *Ofelia* em modo somente leitura. Essa prática permite que o agendador execute os comandos de *backup* diretamente dentro dos contêineres *Postgres* e *InfluxDB* sem a necessidade de expor portas adicionais, instalar clientes SSH ou compartilhar credenciais de rede entre os serviços. Ao restringir o acesso ao socket apenas para leitura, limita-se drasticamente o escopo de possíveis ações maliciosas, garantindo que o *Ofelia* possa orquestrar

as tarefas agendadas de forma eficiente e segura, sem comprometer a integridade do *daemon Docker* ou de outros contêineres.

3.4 Considerações

A arquitetura desenvolvida foi projetada para operar de forma robusta e confiável no município de São Paulo, garantindo alta disponibilidade, tolerância a falhas e segurança na transmissão dos dados. Os testes de estresse realizados com o *broker* Mosquitto, conforme discutido na Seção 3.2, demonstraram que a configuração atual, utilizando QoS 2 com mecanismos de armazenamento local e reenvio, é capaz de suportar a carga de 50 estações no município, mantendo a integridade e a ordem cronológica das mensagens mesmo diante de falhas intermitentes de comunicação.

Embora os resultados experimentais indiquem que a arquitetura projetada para o município de São Paulo seja capaz de operar dentro dos limites de desempenho do *broker*, ao se extrapolar a mesma abordagem para o estado de São Paulo, o aumento da escala do problema se torna evidente. Conforme detalhado a seguir, estima-se que o estado demandaria 841 dispositivos (cálculo apresentado no Apêndice A.2), ou seja, mais de três vezes o número utilizado na implementação municipal. Mantida a mesma taxa de envio de mensagens por dispositivo (10 tópicos por estação, com periodicidade de 1 segundo), o volume de tráfego gerado pelo estado poderia ultrapassar 5 mil mensagens por segundo — valor consideravelmente superior à capacidade máxima de 5 mil mensagens por segundo observada para o Mosquitto com QoS 2 [52], porém muito inferior à capacidade do InfluxDB de lidar com as escritas, que está em torno de 740 mil escritas por segundo [53], deixando evidente que o gargalo do sistema não será o banco de dados.

Essa limitação, contudo, está diretamente relacionada ao nível de qualidade de serviço empregado. Utilizando QoS 0, que elimina o mecanismo de confirmações e retransmissões e reduz significativamente a sobrecarga de processamento, o mesmo estudo reporta capacidade próxima de 18 mil mensagens por segundo [52], valor que já supera a demanda estimada para o estado. Nesse cenário, a perda eventual de algumas mensagens — característica inerente ao QoS 0 — poderia ser mitigada por mecanismos complementares implementados em camadas superiores da aplicação, tais como:

- **Deteção de lacunas temporais:** o servidor pode identificar intervalos sem recebimento de dados de um determinado nó e solicitar a retransmissão dos períodos faltantes;
- **Redundância temporal:** o nó pode ser configurado para enviar cada leitura múltiplas vezes em janelas curtas, aumentando a probabilidade de entrega sem comprometer significativamente a banda;
- **Cache local inteligente:** o SQLite já utilizado para armazenamento local pode ser estendido para guardar um histórico mais longo, permitindo recuperação sob demanda em períodos de baixa utilização da rede.

Além disso, a estimativa de 15 mil mensagens por segundo corresponde a um cenário de pico, no qual todas as estações publicam simultaneamente. Na prática, diversas estratégias podem distribuir o tráfego ao longo do tempo e evitar a saturação do *broker*:

1. **Escalonamento temporal das transmissões:** cada estação pode ser configurada com um *offset* aleatório ou determinístico de até alguns segundos, espalhando as publicações ao longo do intervalo de coleta;

2. **Balanceamento de carga:** múltiplos *brokers* MQTT podem ser utilizados em *cluster*, distribuindo as conexões e as publicações entre diferentes instâncias;

Em síntese, a arquitetura proposta demonstra ser plenamente capaz de operar no município de São Paulo com a configuração atual, apresentando robustez, segurança e tolerância a falhas em patamares adequados para uma implantação em produção. Para uma expansão em escala estadual, a infraestrutura já contempla os mecanismos fundamentais de resiliência e modularidade, sendo necessários apenas ajustes na qualidade de serviço e na distribuição dos *brokers* — mudanças que não afetam os dispositivos já instalados e que podem ser implementadas de forma progressiva e transparente. Assim sendo, o sistema não apenas atende à demanda imediata do município, mas também estabelece uma base sólida para futuras ampliações.

4 Estudo de algoritmos para definição dos pontos de instalação

Definidos os sensores ideais para o projeto (Seção 2) e a arquitetura de coleta, comunicação e armazenamento dos dados (Seção 3), resta a definição estratégica dos pontos de instalação das estações dentro de um número limitado de dispositivos. Como discutido na Seção 1, a atual rede de monitoramento da CETESB concentra-se predominantemente na porção central do município de São Paulo, deixando regiões periféricas com cobertura limitada. No entanto, ampliar essa rede de forma indiscriminada não resolveria a desigualdade de monitoramento identificada, pois, sem um critério explícito de priorização espacial, novos sensores tenderiam a reproduzir a mesma lógica de ocupação da rede existente, beneficiando regiões já relativamente bem servidas em detrimento das populações mais expostas aos efeitos da poluição atmosférica.

Trata-se, portanto, de um problema clássico de localização de instalações (*facility location*) sob restrição orçamentária, onde, dado um número p de sensores a serem instalados e um conjunto de locais candidatos, busca-se o subconjunto de pontos que maximize a cobertura da população, ponderada não apenas pelo número de habitantes, mas também por métricas como vulnerabilidade social (IPVS), taxa de internações respiratórias e exposição à poluição de origem veicular. Essa formulação permite que o gestor público ajuste a prioridade relativa entre eficiência (cobrir o maior número de pessoas) e equidade (cobrir prioritariamente quem mais precisa), em vez de impor uma única resposta correta ao problema.

Dada a ampla literatura na área de otimização, existem diversas formulações possíveis para esse problema. Sendo assim, optou-se por não adotar um único método, mas sim implementar e comparar treze algoritmos de otimização espacial dentro de um framework unificado de avaliação. Essa escolha metodológica garante que todos os métodos operem sobre exatamente a mesma instância do problema, de modo que eventuais diferenças nos resultados sejam atribuíveis unicamente ao algoritmo utilizado, e não a variações nos dados de entrada. Além de viabilizar essa comparação justa e reproduzível, a arquitetura em framework permite que novos métodos e métricas de avaliação sejam incorporados ao sistema sem alterações estruturais, favorecendo a extensibilidade do estudo.

Os métodos comparados são avaliados sob diferentes dimensões, incluindo a cobertura populacional total, a cobertura das populações socialmente mais vulneráveis, a equidade espacial da distribuição entre distritos e o custo computacional de cada abordagem. Esta escolha se deve à inexistência de um método universalmente superior, mas sim os compromissos (*trade-offs*) entre cobertura, equidade e tempo de execução, cuja ponderação cabe, em última instância, ao tomador de decisão.

Sendo assim, a Seção 4.1 apresenta os dados de infraestrutura espacial utilizados como base para o problema de localização; a Seção 4.2 detalha os treze algoritmos implementados, agrupados

em baselines, modelos clássicos, modelos multicritério, heurísticas e meta-heurísticas; a Seção 4.3 traz a análise comparativa entre esses métodos, contemplando métricas de cobertura, tempo de execução e uma análise espacial de sensibilidade da alocação; e, por fim, a Seção 4.4 discute as limitações do estudo e aponta direções para trabalhos futuros.

4.1 Dados de infraestrutura utilizados

Um princípio que norteou a seleção dos dados deste trabalho foi a utilização exclusiva de fontes públicas e padronizadas em âmbito nacional ou estadual, de modo que a metodologia possa ser transportada para qualquer outro município brasileiro com esforço mínimo de adaptação. Todos os conjuntos aqui descritos — Censo Demográfico e malhas do IBGE, Índice Paulista de Vulnerabilidade Social (IPVS) da Fundação SEADE, cadastro CNES e microdados SIH do DATASUS, malha viária do OpenStreetMap e estações da CETESB — são disponibilizados gratuitamente e seguem padrões de codificação uniformes entre municípios. A Tabela 13 resume as propriedades de cada fonte.

Tabela 13: Conjuntos de dados utilizados, suas fontes e papel no modelo.

Dado	Fonte	Ano	Granularidade	Papel no modelo
Setores censitários	IBGE (Censo) [39]	2022	polígono (27.301)	demanda ($\mathcal{A}$)
População	IBGE (Censo) [39]	2022	setor ($\approx 11,45$ mi hab.)	critério α
IPVS	SEADE [35]	2022	setor (6 grupos)	critério γ
Saúde (intern. resp.)	DATASUS [22]	2022	distrito $\rightarrow$ setor	critério β
UBS	CNES/DATASUS [14]	2026	ponto (476)	candidatos ($\mathcal{C}$)
Estações CETESB (autom.)	CETESB [10]	2026	ponto (17)	cobertura existente ($\mathcal{E}$)
Exposição (vias)	OpenStreetMap [59]	2026	setor (raio de 500 m)	critério δ

Critérios de priorização

A unidade de análise é o setor censitário (27.301 setores da cidade de São Paulo, agregados em 96 distritos administrativos). Sobre essa malha, quatro critérios foram selecionados para integrar as análises deste trabalho (Figura 17):

- **População:** número de habitantes de cada setor, segundo o Censo Demográfico 2022. Representa a demanda pelo monitoramento, onde regiões mais populosas expõem mais pessoas a eventuais níveis inadequados de qualidade do ar.
- **Vulnerabilidade social (IPVS):** o Índice Paulista de Vulnerabilidade Social classifica cada setor em um de seis grupos, do Grupo 1 (baixíssima vulnerabilidade) ao Grupo 6 (vulnerabilidade muito alta), a partir de dimensões socioeconômica (renda e escolaridade) e demográfica (composição etária e familiar da população) derivadas do Censo 2022. Sua inclusão introduz um critério de *equidade*, direcionando o monitoramento às populações socialmente mais suscetíveis a agravos ambientais.
- **Saúde (internações respiratórias):** taxa de internações hospitalares por doenças do aparelho respiratório, por mil habitantes, obtida do Sistema de Informações Hospitalares do SUS (SIH). Foram consideradas as internações classificadas no capítulo X da CID-10 (códigos J00–J99), que reúne as doenças do aparelho respiratório, como pneumonia, bronquite, asma e doença pulmonar obstrutiva crônica (DPOC). É um indicador de morbidade que capta

o efeito da qualidade do ar já materializado na saúde da população, atribuído ao local de residência do paciente.

- **Exposição (densidade viária):** quantidade de vias de grande porte (avenidas e vias expressas) em um raio de 500 m do centroide de cada setor, obtida do OpenStreetMap. Utiliza-se como *proxy* da exposição à poluição de origem veicular, assumindo que o tráfego se concentra nessas vias. Foi escolhida por oferecer cobertura completa e alta granularidade para toda a cidade, ao contrário de medições diretas de tráfego ou de poluentes, disponíveis apenas em locais específicos.

A escolha do período de cada dado seguiu duas lógicas distintas. Os critérios de natureza social e de saúde (população, IPVS e internações respiratórias), referem-se todos ao ano de 2022, de modo a preservar a consistência temporal com o Censo Demográfico numa análise transversal. Já os dados de infraestrutura (UBS candidatas, estações da CETESB e malha viária) foram tomados em sua versão mais atual (2026), pois representam o estado presente sobre o qual a decisão de instalação incide: importa saber quais UBS estão em operação hoje, qual a configuração viária atual e quais estações de monitoramento já existem. A Figura 17 apresenta a distribuição espacial dos quatro critérios. Nota-se que vulnerabilidade e saúde concentram-se nas periferias, ao passo que a exposição viária acompanha os corredores estruturais da cidade, padrões espacialmente distintos que justificam tratá-los como critérios independentes.

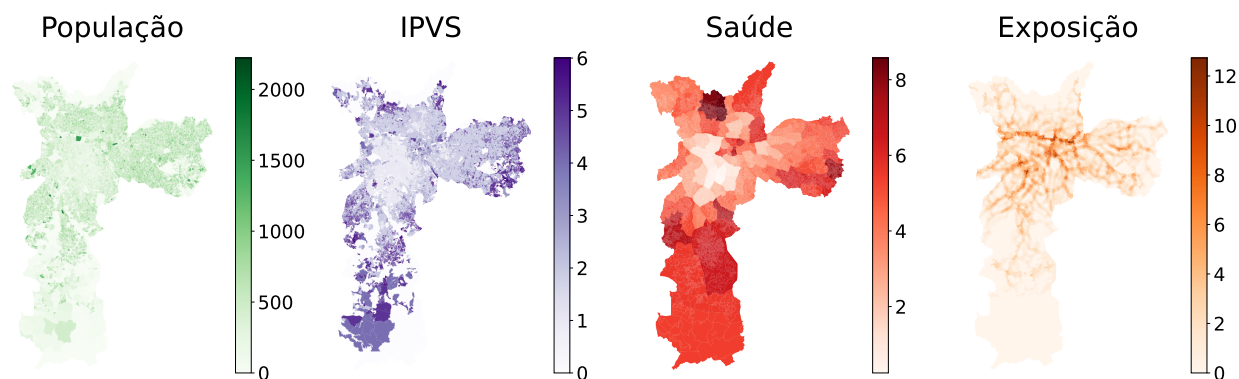


Figura 17: Distribuição espacial dos quatro critérios de priorização por setor censitário em São Paulo: população (habitantes), IPVS (grupos 1–6, valores maiores indicam maior vulnerabilidade social), taxa de internações respiratórias (por mil habitantes) e densidade viária (km de vias em raio de 500 m).

Infraestrutura espacial

Três conjuntos definem a geometria do problema de localização:

- **Setores censitários:** unidade espacial de coleta de informações utilizada pelo IBGE para a realização das pesquisas, representados por seus centroides. Neste trabalho, utilizamos os setores censitários da cidade de São Paulo como pontos de demanda de instalação das estações.
- **Unidades Básicas de Saúde (UBS):** constituem os locais candidatos à instalação de sensores: 476 unidades da rede municipal de São Paulo, extraídas do CNES (unidades básicas,

gestão municipal, ativas). A escolha das UBS como candidatas se deve ao fato dessas estruturas estarem amplamente presentes na cidade e por possuírem por padrão recursos básicos necessários para a instalação das estações, como energia e segurança, o que viabiliza a operação e manutenção dos sensores.

- **Estações automáticas da CETESB:** representam a rede de estações de monitoramento de qualidade do ar já existente na cidade da São Paulo.³

A Figura 18 sobrepõe os locais candidatos a instalação da nossa solução e a rede existente da CETESB. Enquanto as UBS cobrem o território de forma relativamente homogênea, as estações da CETESB são poucas e concentradas na porção central da cidade, deixando amplas regiões periféricas sem monitoramento, sendo exatamente a lacuna que este trabalho busca endereçar.

Infraestrutura: UBS candidatas a sensor e rede CETESB atual
São Paulo — a rede de monitoramento existente é esparsa e concentrada

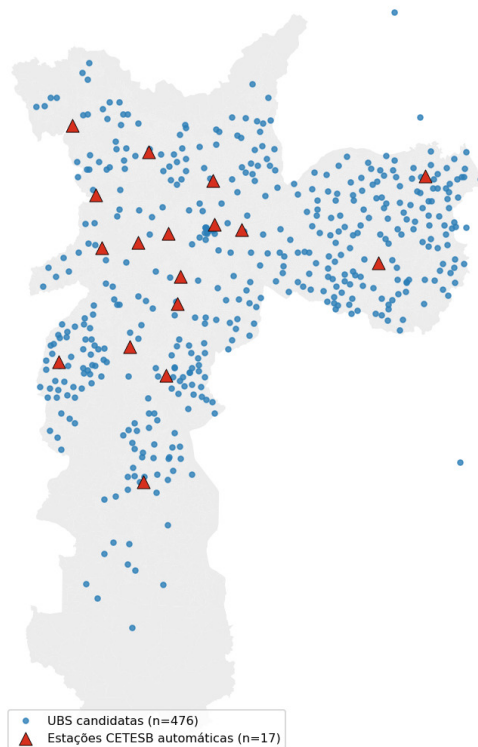


Figura 18: Locais candidatos (476 UBS) e rede de monitoramento existente (17 estações automáticas da CETESB) na cidade de São Paulo. A rede atual de estações é concentrada na região central.

4.2 Algoritmos de otimização implementados

Os algoritmos de seleção de pontos de instalação descritos a seguir não foram implementados como scripts isolados, mas sim como componentes de um framework unificado de otimização e avaliação. Além de permitir a comparação justa e reprodutível entre diferentes métodos sobre a

³Consideramos apenas as 17 estações automáticas, que realizam monitoramento contínuo (24 horas por dia) de poluentes atmosféricos, desconsiderando as estações manuais, que coletam amostras para análise laboratorial. Como o objetivo deste trabalho é ampliar a rede de monitoramento contínuo, as estações automáticas representam a cobertura já existente e são consideradas como cobertura prévia em todos os algoritmos.

mesma instância do problema, a implementação em forma de framework viabiliza a extensão do sistema para a inclusão de novos métodos e métricas de avaliação.

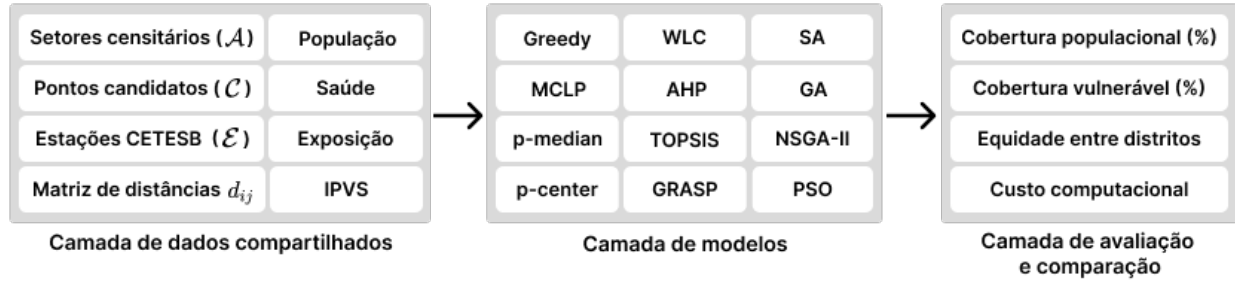


Figura 19: Arquitetura do framework de otimização e avaliação. A camada de dados compartilhados alimenta todos os métodos igualmente; cada método implementa o protocolo `Solver`; o pipeline de avaliação computa métricas independentes, gerando resultados comparáveis em formato tabular. Nem todos os critérios de cada camada estão sendo exibidos.

Formalmente, o framework organiza-se em três camadas, conforme ilustrado na Figura 19:

1. **Camada de dados compartilhados.** Os conjuntos de dados descritos na Seção 4.1 são processados pelo pipeline e incorporados a um objeto `ProblemData` que contém, além dos próprios dados mencionados, todos os elementos do problema de otimização: o conjunto de setores censitários $\mathcal{A}$, as UBSs candidatas $\mathcal{C}$, as estações CETESB existentes $\mathcal{E}$, a matriz de distâncias d_{ij} entre setores e candidatas (pré-computada e persistida em disco para evitar recálculo entre execuções), e os pesos $\alpha, \beta, \gamma, \delta$ da fórmula multicritério. Todos os métodos recebem exatamente a mesma instância de `ProblemData`, garantindo que diferenças nos resultados sejam atribuíveis apenas ao algoritmo, e não a variações nos dados de entrada.
2. **Camada de modelos.** Cada algoritmo implementa um protocolo comum (`Solver`). O contrato é uma única função:

$$\text{solve}(\text{data: ProblemData}, p: \text{int}, \text{**kwargs}) \rightarrow \text{SolveResult}, \quad (1)$$

que recebe os dados do problema e o número de sensores p , e retorna a solução encontrada — um conjunto $\mathcal{I} \subseteq \mathcal{C}$ de p UBSs selecionadas, o valor da função objetivo do método, o tempo de execução e metadados de reprodução. Este contrato permite que qualquer método possa ser registrado na configuração e executado pelo pipeline, sem alterações nas demais camadas do framework.

3. **Camada de avaliação e comparação.** Um pipeline único itera sobre o produto cartesiano de métodos e valores de p , invoca cada `Solver`, computa um conjunto fixo de métricas independentes (cobertura populacional, cobertura de vulneráveis, distância média ponderada, equidade entre distritos, custo computacional, entre outras) e persiste os resultados em disco, o que alimenta a geração automatizada de figuras e tabelas.

A Tabela 14 resume as notações matemáticas empregadas por todos os algoritmos desta seção.

O peso w_i de cada setor censitário $i \in \mathcal{A}$ é dado pela combinação linear dos quatro critérios normalizados descritos na Seção 4.1:

$$w_i = \alpha \cdot \text{pop}_i + \beta \cdot \text{saúde}_i + \gamma \cdot \text{IPVS}_i + \delta \cdot \text{exp}_i, \quad (2)$$

Tabela 14: Notações utilizadas nas formulações dos algoritmos.

Símbolo	Significado	Dimensão
$\mathcal{A}$	Conjunto de setores censitários (demanda)	$ \mathcal{A} = 27.301$
$\mathcal{C}$	Conjunto de UBSs candidatas	$ \mathcal{C} = 476$
$\mathcal{E}$	Conjunto de estações CETESB existentes	$ \mathcal{E} = 17$
$\mathcal{I}$	Conjunto de UBSs selecionadas ($\mathcal{I} \subseteq \mathcal{C}$)	$ \mathcal{I} = p$
p	Número de sensores a instalar	variável
i	Índice de setor censitário ($i \in \mathcal{A}$)	—
j	Índice de UBS candidata ($j \in \mathcal{C}$)	—
d_{ij}	Distância Haversine entre setor i e UBS j	km
R_{exist}	Raio de cobertura das estações da CETESB	3,0 km (padrão)
R_{cand}	Raio de cobertura das UBSs candidatas à instalação	2,0 km (padrão)
w_i	Peso do setor i (Equação (2))	—
$\alpha, \beta, \gamma, \delta$	Pesos dos critérios (população, saúde, IPVS, exposição)	—
a_{ij}	Indicador de cobertura: $\mathbb{1}[d_{ij} \leq R_{cand}]$	$\{0, 1\}$
c_i	Indicador de cobertura prévia pela CETESB: $\mathbb{1}[\min_{j \in \mathcal{E}} d_{ij} \leq R_{exist}]$	$\{0, 1\}$

em que pop_i , $saúde_i$, $IPVS_i$ e exp_i são, respectivamente, os valores normalizados em $[0, 1]$ dos critérios população, saúde (internações respiratórias), IPVS e exposição (densidade viária) para o setor i . A normalização é feita por *min-max*: $\tilde{x} = (x - x_{\min}) / (x_{\max} - x_{\min})$, de modo que todos os critérios contribuam em escala comparável. Os parâmetros $\alpha, \beta, \gamma, \delta \geq 0$ permitem ao tomador de decisão ajustar a importância relativa de cada dimensão; neste trabalho, adota-se $\alpha = \beta = \gamma = 1$ e $\delta = 0$ como configuração padrão, atribuindo peso igual aos três critérios sociais e de saúde.

Com o peso w_i definido, resta especificar como ele se traduz em cobertura. A definição adotada por todos os métodos é binária: um setor i é considerado coberto se estiver a uma distância de até R_{exist} de alguma estação CETESB existente, ou de até R_{cand} de alguma UBS selecionada. Formalmente, dado um conjunto de sensores $\mathcal{I} \subseteq \mathcal{C}$ (possivelmente vazio, quando apenas a rede existente é considerada), a cobertura ponderada total é:

$$f(\mathcal{I}) = \sum_{i \in \mathcal{A}} w_i \cdot \mathbb{1} \left[\min_{j \in \mathcal{E}} d_{ij} \leq R_{exist} \vee \min_{j \in \mathcal{I}} d_{ij} \leq R_{cand} \right], \quad (3)$$

em que w_i é o peso do setor i conforme a Equação (2). A função $f(\mathcal{I})$ é a função objetivo dos métodos que maximizam cobertura (MCLP, GRASP, Simulated Annealing, Genetic Algorithm, NSGA-II e PSO), enquanto p-Median e p-Center adotam funções alternativas (minimização de distância média e máxima, respectivamente). As estações CETESB existentes ($\mathcal{E}$) são consideradas como cobertura prévia em todos os métodos, de modo que a contribuição incremental de cada UBS selecionada é medida sobre a cobertura já proporcionada pela rede atual.

4.2.1 Baselines

Para estabelecer uma referência inferior de desempenho e isolar o ganho proporcionado por modelos mais sofisticados, foram implementados três algoritmos baseline. Cada um deles resolve o mesmo problema de seleção de p sensores a partir do conjunto de UBSs candidatas, respeitando o mesmo raio de cobertura R e o mesmo conjunto de pesos α (população), β (saúde), γ (IPVS) e δ (exposição) descritos anteriormente. O custo computacional desses algoritmos é propositalmente baixo, servindo como referência para a análise de custo-benefício dos métodos mais elaborados.

Estações atuais da CETESB (*cetesb_only*): Este baseline não realiza nenhuma seleção. A solução avaliada é simplesmente a rede de monitoramento já instalada pela CETESB na cidade de São Paulo, sem adicionar nenhuma UBS ao conjunto. Formalmente, $\mathcal{I} = \emptyset$, onde $\mathcal{I}$ é o conjunto de UBSs selecionadas. A cobertura obtida é unicamente a proporcionada pelas m estações existentes, ou seja:

$$\text{Cobertura}_{\text{CETESB}} = \sum_{i \in \mathcal{A}} w_i \cdot \mathbb{1} \left[\min_{j \in \mathcal{E}} d_{ij} \leq R_{\text{exist}} \right], \quad (4)$$

em que $\mathcal{A}$ é o conjunto de setores censitários, $\mathcal{E}$ o conjunto de estações CETESB, d_{ij} a distância em km entre o setor i e a estação j , R_{exist} o raio de cobertura e w_i o peso do setor i conforme a fórmula multicritério. Este baseline estabelece o piso de desempenho: qualquer solução que selecione p UBSs deve superar esta cobertura para ser considerada vantajosa.

Amostragem aleatória (*random*): Seleciona p UBSs candidatas uniformemente ao acaso, sem reposição. Dado o conjunto de n UBSs candidatas $\mathcal{C}$, cada subconjunto $\mathcal{I} \subset \mathcal{C}$ de tamanho p tem probabilidade $\binom{n}{p}^{-1}$ de ser escolhido:

$$\mathcal{I} \sim \text{Uniforme}(\{\mathcal{I}' \subseteq \mathcal{C} : |\mathcal{I}'| = p\}). \quad (5)$$

Para mitigar o efeito de sementes específicas, o framework executa múltiplas repetições com sementes distintas (padrão: 5 repetições, configurável), permitindo estimar a média e o desvio padrão da cobertura alcançável por um algoritmo puramente estocástico. Este baseline fornece um limite inferior probabilístico e é útil para detectar se um método determinístico está simplesmente fornecendo soluções ao acaso.

Seleção gulosa por cobertura (*greedy*): Implementa o algoritmo clássico de maximização submodular gulosa [55]. A cada iteração k , seleciona a UBS $j \in \mathcal{C} \setminus \mathcal{I}_{k-1}$ que maximiza o ganho marginal na cobertura ponderada:

$$j_k = \arg \max_{j \in \mathcal{C} \setminus \mathcal{I}_{k-1}} \sum_{i \in \mathcal{A}} w_i \cdot \mathbb{1} \left[\underbrace{d_{ij} \leq R_{\text{cand}}}_{j \text{ cobre o setor } i} \wedge \underbrace{\min_{j' \in \mathcal{E}} d_{ij'} > R_{\text{exist}}}_{\text{não coberto pela CETESB}} \wedge \underbrace{\min_{j' \in \mathcal{I}_{k-1}} d_{ij'} > R_{\text{cand}}}_{\text{não coberto pelas UBSs já instaladas}} \right], \quad (6)$$

ou seja, o ganho marginal é a soma dos pesos dos setores que passam a ser cobertos pela UBS j e que ainda não eram cobertos por estações CETESB ou por UBSs já selecionadas.

O algoritmo tem complexidade $\mathcal{O}(p \cdot n \cdot |\mathcal{A}|)$, sendo executado em frações de segundo mesmo para centenas de UBSs e milhares de setores. A implementação considera a cobertura inicial das estações CETESB existentes antes da primeira iteração, conforme ilustrado no Algoritmo 1.

4.2.2 Modelos de otimização clássicos

Os modelos clássicos de localização de instalações fornecem formulações matemáticas bem estabelecidas para o problema de seleção de sensores. Cada um deles define uma função objetivo distinta (cobertura, distância total ou distância máxima) e impõe restrições orçamentárias e de atendimento. A seguir, apresentamos os três modelos implementados neste trabalho: MCLP, p-Median e p-Center.

Algorithm 1 Algoritmo guloso para maximização de cobertura

```

1: procedure GREEDYSELECTION(data, p)
2:   coberto  $\leftarrow \{i : \min_j \text{dist\_existente}_{ij} \leq R_{\text{exist}}\}$ 
3:   instaladas  $\leftarrow \emptyset$ 
4:   for  $k = 1$  to  $p$  do
5:     melhorj, melhor_ganho  $\leftarrow -1, -\infty$ 
6:     for  $j \in \text{candidatas\_restantes}$  do
7:       nova_cobertura  $\leftarrow \{i : \text{dist\_matrix}_{ij} \leq R_{\text{cand}}\}$ 
8:       ganho  $\leftarrow \sum_{i \in \text{nova\_cobertura} \setminus \text{coberto}} w_i$ 
9:       if ganho > melhor_ganho then
10:        melhor_ganho, melhorj  $\leftarrow \text{ganho}, j$ 
11:      end if
12:    end for
13:    coberto  $\leftarrow \text{coberto} \cup \{i : \text{dist\_matrix}_{i, \text{melhor}_j} \leq R_{\text{cand}}\}$ 
14:    instaladas  $\leftarrow \text{instaladas} \cup \{\text{melhor}_j\}$ 
15:  end for return instaladas
16: end procedure

```

Maximal Coverage Location Problem (MCLP): O MCLP [13] é um problema de programação linear inteira mista (MILP) que maximiza a soma dos pesos dos setores cobertos, dado um orçamento de p sensores.

Para a formulação do MCLP, utilizamos as notações apresentadas na Tabela 14, além das seguintes variáveis de decisão:

- $x_j \in \{0, 1\}$: 1 se a UBS j for selecionada, 0 caso contrário;
- $y_i \in \{0, 1\}$: 1 se o setor i for coberto, 0 caso contrário.

O problema é formulado como:

$$\text{Maximizar} \quad \sum_{i \in \mathcal{A}} w_i y_i \quad (7)$$

$$\text{sujeito a} \quad y_i \leq c_i + \sum_{j \in \mathcal{C}} a_{ij} x_j, \quad \forall i \in \mathcal{A} \quad (8)$$

$$\sum_{j \in \mathcal{C}} x_j \leq p \quad (9)$$

$$x_j \in \{0, 1\}, y_i \in \{0, 1\}, \quad \forall i \in \mathcal{A}, \forall j \in \mathcal{C}, \quad (10)$$

em que $a_{ij} = \mathbb{1}[d_{ij} \leq R_{\text{cand}}]$ indica se a UBS j cobre o setor i .

A restrição (8) permite que um setor seja considerado coberto ($y_i = 1$) se ele já for atendido por uma estação CETESB ($c_i = 1$) ou se pelo menos uma UBS selecionada o cobre. A restrição (9) limita o número total de UBSs selecionadas a p . O modelo é resolvido com o solver CBC via PuLP, com *timeout* configurável (padrão: 60 s), conforme o Algoritmo 2.

***p*-Median:** O problema *p*-Median [36] busca minimizar a soma ponderada das distâncias entre cada setor e a instalação (CETESB ou UBS) mais próxima. Diferentemente do MCLP, que

Algorithm 2 Resolução do MCLP via programação linear inteira

```

1: procedure MCLP(data, p, timeout)
2:    $\mathcal{A} \leftarrow$  setores ordenados por area_index
3:    $\mathcal{C} \leftarrow$  candidatas ordenadas por cand_index
4:    $w_i \leftarrow \alpha \cdot pop_i + \beta \cdot saude_i + \gamma \cdot ipvs_i + \delta \cdot exposicao_i$ 
5:    $c_i \leftarrow \mathbb{1}[\min_j dist\_existente_{ij} \leq R_{exist}]$ 
6:    $a_{ij} \leftarrow \mathbb{1}[dist\_matrix_{ij} \leq R_{cand}]$ 
7:   Criar variáveis binárias  $x_j$  ( $j \in \mathcal{C}$ ) e  $y_i$  ( $i \in \mathcal{A}$ )
8:   objetivo  $\leftarrow \sum_i w_i y_i$ 
9:   Adicionar restrição  $y_i \leq c_i + \sum_j a_{ij} x_j, \forall i$ 
10:  Adicionar restrição  $\sum_j x_j \leq p$ 
11:  Resolver MILP com CBC (limite: timeout s)
12:  return  $\{j : x_j = 1\}$ , valor objetivo
13: end procedure

```

opera com um limiar binário de cobertura, o p-Median considera a distância real, privilegiando configurações que aproximam os sensores das regiões de maior peso. A formulação é:

$$\text{Minimizar} \quad \sum_{i \in \mathcal{A}} w_i \cdot \min \left\{ \min_{j \in \mathcal{E}} d_{ij}, \min_{j \in \mathcal{I}} d_{ij} \right\} \quad (11)$$

$$\text{sujeito a} \quad |\mathcal{I}| = p, \quad \mathcal{I} \subseteq \mathcal{C}, \quad (12)$$

em que $\mathcal{I}$ é o conjunto de UBSs selecionadas.

A implementação utiliza uma heurística construtiva gulosa seguida de busca local 1-swap. Na fase construtiva, a cada iteração seleciona-se a UBS que mais reduz a soma ponderada das distâncias mínimas. Na fase de refinamento, cada UBS selecionada é temporariamente removida e todas as candidatas restantes são testadas como substitutas; a troca é aceita se reduzir o valor objetivo em pelo menos 10^{-6} . O processo de *swap* se repete por n_{swaps} rodadas completas (padrão: 3) ou até que nenhuma troca melhore a solução. As estações CETESB existentes são incluídas como instalações fixas desde a primeira iteração. O pseudocódigo é apresentado no Algoritmo 3.

p-Center: O problema p-Center [36] adota uma perspectiva minimax: em vez de minimizar a distância média ponderada, minimiza a máxima distância entre qualquer setor e a instalação mais próxima. Este modelo é particularmente relevante para garantir equidade espacial, ou seja, que nenhuma região fique excessivamente distante de um sensor de qualidade do ar. A formulação é dada por:

$$\text{Minimizar} \quad \max_{i \in \mathcal{A}} \min \left\{ \min_{j \in \mathcal{E}} d_{ij}, \min_{j \in \mathcal{I}} d_{ij} \right\} \quad (13)$$

$$\text{sujeito a} \quad |\mathcal{I}| = p, \quad \mathcal{I} \subseteq \mathcal{C}. \quad (14)$$

A implementação segue a mesma estrutura do p-Median (construção gulosa seguida de busca local 1-swap), porém o critério de seleção na fase construtiva é o que mais reduz a distância máxima atual, em vez da soma ponderada. Para cada candidata j , calcula-se:

$$\text{melhor_max}_j = \max_i \min (min_dist_i, d_{ij}), \quad (15)$$

Algorithm 3 Heurística construtiva gulosa com busca local 1-swap para o p-Median

```

1: procedure PMEDIAN( $data, p, n_{swaps}$ )
2:    $min\_dist \leftarrow \min_j dist\_existente_{ij} \triangleright$  distância à CETESB mais próxima
3:    $instaladas \leftarrow \emptyset$ 
4:   for  $k = 1$  to  $p$  do
5:      $melhor_j, melhor\_ganho \leftarrow -1, -\infty$ 
6:     for  $j \in candidatas\_restantes$  do
7:        $nova\_min \leftarrow \min(min\_dist, dmat[:, j])$ 
8:        $ganho \leftarrow \sum_i w_i \cdot (min\_dist_i - nova\_min_i)$ 
9:       if  $ganho > melhor\_ganho$  then
10:         $melhor\_ganho, melhor_j \leftarrow ganho, j$ 
11:      end if
12:    end for
13:     $min\_dist \leftarrow \min(min\_dist, dmat[:, melhor_j])$ 
14:     $instaladas \leftarrow instaladas \cup \{melhor_j\}$ 
15:  end for
16:   $\triangleright$  Fase de busca local 1-swap
17:  for  $r = 1$  to  $n_{swaps}$  do
18:    for all  $out_j \in instaladas$  do
19:       $outros \leftarrow instaladas \setminus \{out_j\}$ 
20:       $melhor\_substituto, melhor\_obj \leftarrow \operatorname{argmin}_{in_j \notin instaladas} \sum_i w_i \cdot \min(min\_dist, dmat[:,$ 
21:         $outros \cup \{in_j\}])$ 
22:      if  $melhor\_obj < objetivo\_atual - 10^{-6}$  then
23:         $instaladas \leftarrow (instaladas \setminus \{out_j\}) \cup \{melhor\_substituto\}$ 
24:      end if
25:    end for
26:  end for
27:  return  $instaladas$ 
28: end procedure

```

e seleciona-se a UBS com o menor $melhor_max_j$. A fase de *swap* segue a mesma lógica, sempre aceitando trocas que reduzam a distância máxima. O Algoritmo 4 detalha o procedimento.

4.2.3 Modelos multicritério

Ao contrário dos métodos clássicos da Seção 4.2.2, que otimizam uma única função objetivo global, os modelos multicritério de decisão (MCDM, *Multi-Criteria Decision Making*) avaliam cada UBS candidata individualmente com base em múltiplos critérios simultaneamente, gerando um *score* composto que permite ranquear as alternativas. As p UBSs com maior pontuação são selecionadas.

A principal vantagem desta abordagem é a flexibilidade, visto que novos critérios podem ser adicionados sem modificar a estrutura do problema de otimização. Além disso, a avaliação individual permite que o tomador de decisão compreenda o desempenho de cada candidata em cada dimensão, facilitando a interpretação dos resultados. A desvantagem é que, diferentemente do MCLP, não há garantia de que a solução selecionada seja ótima do ponto de vista da cobertura global, uma vez que a seleção independente pode levar à redundância espacial (duas UBSs com alta pontuação individual podem estar próximas e cobrir os mesmos setores).

Os três modelos a seguir compartilham a mesma etapa de pré-processamento: para cada UBS

Algorithm 4 Heurística construtiva gulosa com busca local 1-swap para o p-Center

```

1: procedure PCENTER( $data, p, n_{swaps}$ )
2:    $min\_dist \leftarrow \min_j dist\_existente_{ij} \triangleright$  distância à CETESB mais próxima
3:    $instaladas \leftarrow \emptyset$ 
4:   for  $k = 1$  to  $p$  do
5:      $melhor_j, melhor\_max \leftarrow -1, \infty$ 
6:     for  $j \in candidatas\_restantes$  do
7:        $nova\_min \leftarrow \min(min\_dist, dmat[:, j])$ 
8:        $mx \leftarrow \max_i nova\_min_i$ 
9:       if  $mx < melhor\_max$  then
10:         $melhor\_max, melhor_j \leftarrow mx, j$ 
11:      end if
12:    end for
13:     $min\_dist \leftarrow \min(min\_dist, dmat[:, melhor_j])$ 
14:     $instaladas \leftarrow instaladas \cup \{melhor_j\}$ 
15:  end for
16:   $\triangleright$  Busca local 1-swap (minimizando a distância máxima)
17:  for  $r = 1$  to  $n_{swaps}$  do
18:    for all  $out_j \in instaladas$  do
19:       $outros \leftarrow instaladas \setminus \{out_j\}$ 
20:       $melhor\_substituto, melhor\_max \leftarrow \operatorname{argmin}_{in_j \notin instaladas} \max_i \min(min\_dist, dmat[:,$ 
21:         $outros \cup \{in_j\}])$ 
22:      if  $melhor\_max < max\_atual - 10^{-6}$  then
23:         $instaladas \leftarrow (instaladas \setminus \{out_j\}) \cup \{melhor\_substituto\}$ 
24:      end if
25:    end for
26:  return  $instaladas$ 
27: end procedure

```

candidata j , são calculados quatro critérios a partir da matriz de distâncias e dos dados dos setores censitários:

- c_{j1} (cobertura ponderada): fração do peso total w coberta pela UBS j dentro do raio R ;
- c_{j2} (cobertura de população vulnerável): fração da população em setores com IPVS 4 ou 5 coberta por j ;
- c_{j3} (distância média negativa): $-\frac{1}{|C_j|} \sum_{i \in C_j} d_{ij}$, em que $C_j = \{i : d_{ij} \leq R_{cand}\}$ (o sinal negativo garante que distâncias menores correspondam a valores mais altos);
- c_{j4} (cobertura de população extremamente vulnerável): fração da população em setores com IPVS 5 coberta por j .

Weighted Linear Combination (WLC): O WLC é o modelo multicritério mais simples, onde cada UBS recebe como pontuação a soma ponderada dos pesos dos setores que ela cobre dentro do raio R . Formalmente, o score da UBS j é dado por:

$$s_j = \sum_{i \in \mathcal{A}} w_i \cdot \mathbb{1}[d_{ij} \leq R_{cand}], \quad (16)$$

em que $w_i = \alpha \cdot \text{pop}_i + \beta \cdot \text{saúde}_i + \gamma \cdot \text{IPVS}_i + \delta \cdot \text{exposição}_i$ é o peso do setor i conforme a fórmula multicritério descrita na Seção 4.2. As p UBSs com maior s_j são selecionadas.

O WLC pode ser interpretado como uma versão gulosa *non-adaptive*: enquanto o algoritmo guloso da Seção 4.2.1 recalcula o ganho marginal a cada iteração (considerando a cobertura já estabelecida), o WLC atribui o score uma única vez com base na cobertura individual de cada UBS, ignorando redundâncias. Isso torna o algoritmo extremamente rápido ($\mathcal{O}(n \cdot |\mathcal{A}|)$ para o cálculo dos scores, mais $\mathcal{O}(n \log n)$ para a ordenação), porém sem garantia de qualidade de aproximação.

Analytic Hierarchy Process (AHP): O AHP [63] é um método estruturado para tomada de decisão multicritério que organiza o problema em uma hierarquia de critérios e alternativas. A etapa central do AHP é a construção de matrizes de comparação pareada entre os critérios, nas quais o decisor atribui julgamentos de importância relativa em uma escala de 1 a 9. O autovetor principal da matriz de comparações fornece os pesos relativos de cada critério.

Neste trabalho, os pesos α , β , γ e δ da fórmula multicritério (população, saúde, IPVS e exposição) são interpretados como os pesos AHP derivados de uma matriz de comparação pareada hipotética. Dessa forma, a implementação do AHP (Algoritmo 5) delega ao WLC o cálculo dos scores individuais, utilizando os mesmos pesos já definidos:

$$s_j^{(\text{AHP})} = \alpha \cdot \text{pop}_j^{\text{cob}} + \beta \cdot \text{saúde}_j^{\text{cob}} + \gamma \cdot \text{IPVS}_j^{\text{cob}} + \delta \cdot \text{exposição}_j^{\text{cob}} = s_j^{(\text{WLC})}, \quad (17)$$

em que $\text{pop}_j^{\text{cob}} = \sum_i \text{pop}_i \cdot \mathbb{1}[d_{ij} \leq R_{cand}]$ (e analogamente para os demais componentes). Do ponto de vista computacional, AHP e WLC produzem resultados idênticos quando alimentados com os mesmos pesos; a distinção conceitual está no processo de elicitação dos pesos: o AHP fornece uma metodologia para derivá-los a partir de julgamentos subjetivos, enquanto o WLC os toma como dados de entrada.

Algorithm 5 WLC / AHP — seleção por combinação linear ponderada

```

1: procedure WLC(data, p)
2:    $\mathcal{A} \leftarrow$  setores ordenados por area_index
3:    $\mathcal{C} \leftarrow$  candidatas ordenadas por cand_index
4:    $w_i \leftarrow \alpha \cdot \text{pop}_i + \beta \cdot \text{saude}_i + \gamma \cdot \text{ipvs}_i + \delta \cdot \text{exposicao}_i$ 
5:
6:   for  $j \in \mathcal{C}$  do
7:      $s_j \leftarrow \sum_{i \in \mathcal{A}} w_i \cdot \mathbb{1}[\text{dist\_matrix}_{ij} \leq R_{cand}]$ 
8:   end for
9:    $\mathcal{I} \leftarrow$  índices das  $p$  UBSs com maior  $s_j$ 
10:  return  $\mathcal{I}$ 
11: end procedure

```

Technique for Order Preference by Similarity to Ideal Solution (TOPSIS): O TOPSIS [38] é um método multicritério que avalia cada alternativa com base em sua distância geométrica a duas soluções de referência: a solução ideal positiva (melhor desempenho em todos os critérios) e a solução ideal negativa (pior desempenho em todos os critérios). A premissa é que a melhor

alternativa deve estar simultaneamente mais próxima do ideal positivo e mais distante do ideal negativo.

Diferentemente do WLC/AHP, que utiliza um único score linear, o TOPSIS opera sobre múltiplos critérios distintos (neste trabalho, os quatro critérios c_{j1} a c_{j4} descritos anteriormente). O procedimento é o seguinte:

1. **Normalização:** cada coluna da matriz de critérios $\mathbf{C} \in \mathbb{R}^{n \times 4}$ é normalizada pelo seu módulo Euclidiano:

$$\bar{c}_{jk} = \frac{c_{jk}}{\sqrt{\sum_{j'=1}^n c_{j'k}^2}}, \quad k = 1, \dots, 4. \quad (18)$$

2. **Ponderação:** aplicam-se pesos aos critérios normalizados. Neste trabalho, utilizam-se pesos $\boldsymbol{\pi} = [0.4; 0.3; 0.15; 0.15]$ para cobertura ponderada, cobertura de população vulnerável, distância média negativa e cobertura IPVS 5, respectivamente, de modo que a matriz ponderada é $v_{jk} = \pi_k \cdot \bar{c}_{jk}$.

3. **Soluções ideais:** identificam-se os vetores ideal positivo A^+ e ideal negativo A^- :

$$A_k^+ = \max_j v_{jk}, \quad A_k^- = \min_j v_{jk}, \quad \forall k. \quad (19)$$

4. **Distâncias:** para cada UBS j , calculam-se as distâncias Euclidianas às duas soluções ideais:

$$d_j^+ = \sqrt{\sum_{k=1}^4 (v_{jk} - A_k^+)^2}, \quad (20)$$

$$d_j^- = \sqrt{\sum_{k=1}^4 (v_{jk} - A_k^-)^2}. \quad (21)$$

5. **Score de similaridade:** a pontuação final é a similaridade relativa ao ideal negativo:

$$s_j^{(\text{TOPSIS})} = \frac{d_j^-}{d_j^+ + d_j^-} \in [0, 1], \quad (22)$$

em que valores próximos de 1 indicam maior proximidade da solução ideal.

As p UBSs com maior $s_j^{(\text{TOPSIS})}$ são selecionadas. O Algoritmo 6 resume o procedimento.

A principal vantagem do TOPSIS sobre o WLC é a capacidade de capturar trade-offs mais complexos entre os critérios por meio da geometria multidimensional. Enquanto o WLC assume que todos os critérios são comensuráveis e combináveis linearmente, o TOPSIS permite que uma UBS com desempenho moderado em todos os critérios seja preferida a outra com desempenho excelente em um critério mas ruim em outro.

4.2.4 Modelos baseados em heurísticas e meta-heurísticas

Os métodos apresentados nesta seção compartilham o mesmo objetivo do MCLP (maximizar a cobertura ponderada total), porém, diferentemente do modelo exato resolvido por programação linear inteira, eles empregam estratégias de busca aproximada para encontrar soluções de alta qualidade sem a garantia de otimalidade. A distinção entre heurísticas e meta-heurísticas é sutil.

Algorithm 6 TOPSIS para seleção de UBSs

```

1: procedure TOPSIS( $data, p, \pi$ )
2:    $\mathbf{C} \leftarrow$  matriz de critérios ( $n \times 4$ )
3:    $\bar{\mathbf{C}} \leftarrow$  normalização coluna por  $\|\mathbf{C}_{:,k}\|_2$ 
4:    $\mathbf{V} \leftarrow \bar{\mathbf{C}} \odot \pi \triangleright$  ponderação
5:    $A^+ \leftarrow \max_j \mathbf{V}_{j,:}$ ,  $A^- \leftarrow \min_j \mathbf{V}_{j,:}$ 
6:   for  $j \in \mathcal{C}$  do
7:      $d_j^+ \leftarrow \|\mathbf{V}_{j,:} - A^+\|_2$ 
8:      $d_j^- \leftarrow \|\mathbf{V}_{j,:} - A^-\|_2$ 
9:      $s_j \leftarrow d_j^- / (d_j^+ + d_j^-)$ 
10:  end for
11:   $\mathcal{I} \leftarrow$  índices das  $p$  UBSs com maior  $s_j$ 
12:  return  $\mathcal{I}$ 
13: end procedure

```

Enquanto heurísticas construtivas (como o algoritmo guloso da Seção 4.2.1) constroem uma solução passo a passo por meio de regras determinísticas e encerram-se ao final da construção, as meta-heurísticas são estratégias de alto nível que guiam a busca no espaço de soluções, podendo escapar de ótimos locais por meio de mecanismos como aleatoriedade, memória ou população de candidatos.

Todos os métodos descritos a seguir maximizam a cobertura ponderada:

$$f(\mathcal{I}) = \sum_{i \in \mathcal{A}} w_i \cdot \mathbb{1} \left[\underbrace{\min_{j \in \mathcal{E}} d_{ij} \leq R_{exist}}_{\text{coberto pela CETESB}} \vee \underbrace{\min_{j \in \mathcal{I}} d_{ij} \leq R_{cand}}_{\text{coberto por UBS selecionada}} \right], \quad (23)$$

em que $\mathcal{I} \subseteq \mathcal{C}$ é o conjunto de UBSs selecionadas, $|\mathcal{I}| \leq p$, e a cobertura prévia da CETESB ($\mathcal{E}$) é sempre considerada.

GRASP (Greedy Randomized Adaptive Search Procedure): O GRASP [34] combina uma fase de construção semi-gulosa aleatorizada com uma fase de busca local. Diferentemente do algoritmo guloso determinístico (Seção 4.2.1), que a cada iteração seleciona a UBS com maior ganho marginal, o GRASP constrói uma *Lista Restrita de Candidatos* (RCL) com as melhores $\alpha \cdot n$ candidatas (padrão: $\alpha = 0.3$) e sorteia uma delas. Esse processo é repetido por n_{iter} iterações independentes, cada uma gerando uma solução diferente.

Para cada solução construída, aplica-se uma busca local 1-swap que avalia, para cada par (UBS selecionada, UBS não selecionada), se a troca melhora a cobertura. A busca local é implementada de forma incremental, mantendo um vetor de contagem de cobertura por setor (**coverage_count**) de modo que o ganho de uma troca é calculado como:

$$\Delta = \underbrace{\sum_{i \in A_{\text{in}} \cap U_0} w_i}_{\text{ganho}} - \underbrace{\sum_{i \in A_{\text{out}} \cap U_1 \setminus A_{\text{in}}} w_i}_{\text{perda}}, \quad (24)$$

em que A_{in} e A_{out} são os conjuntos de setores cobertos pela UBS candidata a entrar e pela UBS a sair, respectivamente, U_0 são setores sem cobertura alguma e U_1 são setores cobertos por exatamente uma fonte. Uma troca é aceita se $\Delta > 0$. Ao final, retorna-se a melhor solução encontrada entre todas as iterações.

Algorithm 7 GRASP com busca local incremental

```

1: procedure GRASP( $data, p, \alpha, n_{iter}$ )
2:    $cov\_mask \leftarrow$  matriz booleana de cobertura ( $n_{\text{áreas}} \times n_{\text{cand}}$ )
3:    $exist\_mask \leftarrow$  cobertura inicial da CETESB
4:    $melhor\_sol \leftarrow \emptyset, melhor\_fit \leftarrow -\infty$ 
5:   for  $t = 1$  to  $n_{iter}$  do
6:      $coberto \leftarrow exist\_mask, instaladas \leftarrow \emptyset$ 
7:     for  $k = 1$  to  $p$  do
8:        $ganhos[j] \leftarrow \sum_i w_i \cdot (cov\_mask_{ij} \wedge \neg coberto)$ 
9:        $\theta \leftarrow \text{percentil}(ganhos, 100(1 - \alpha))$ 
10:       $RCL \leftarrow \{j : ganhos[j] \geq \theta\}$ 
11:       $j^* \leftarrow \text{sorteio}(RCL)$ 
12:       $coberto \leftarrow coberto \vee cov\_mask[:, j^*]$ 
13:       $instaladas \leftarrow instaladas \cup \{j^*\}$ 
14:    end for
15:    Aplicar busca local 1-swap incremental em  $instaladas$ 
16:    if  $f(instaladas) > melhor\_fit$  then
17:       $melhor\_sol \leftarrow instaladas, melhor\_fit \leftarrow f(instaladas)$ 
18:    end if
19:  end for
20:  return  $melhor\_sol$ 
21: end procedure

```

Simulated Annealing (SA): O *Simulated Annealing* [47] é uma meta-heurística inspirada no recozimento de metais que explora o espaço de soluções por meio de movimentos aleatórios, aceitando ocasionalmente soluções piores para escapar de ótimos locais. A aceitação de uma solução pior é governada pela temperatura T , que decai ao longo das iterações.

A implementação parte de uma solução inicial aleatória (conjunto de p UBSs escolhidas uniformemente). Em cada iteração, um vizinho é gerado trocando uma UBS selecionada por uma não selecionada (swap). Se a nova cobertura é melhor ($\Delta > 0$), a troca é aceita; caso contrário, é aceita com probabilidade $e^{\Delta/T}$. A temperatura segue um resfriamento geométrico: $T \leftarrow T \cdot \gamma$, com $\gamma = 0.95$ e $T_0 = 100$, por $n_{iter} = 500$ iterações (padrão). Ao final, retorna-se a melhor solução encontrada durante toda a busca.

$$P(\text{aceitar piora}) = \exp\left(\frac{f(\mathcal{I}') - f(\mathcal{I})}{T}\right), \quad (25)$$

em que $\mathcal{I}'$ é a solução vizinha e $\mathcal{I}$ a solução atual.

Genetic Algorithm (GA): O *Genetic Algorithm* [37] é uma meta-heurística populacional inspirada na evolução natural. Cada indivíduo da população é um cromossomo binário de comprimento $n = |\mathcal{C}|$, com exatamente p genes ativos ($1 = \text{UBS selecionada}$). A população inicial é gerada aleatoriamente, com p posições ativas sorteadas uniformemente para cada indivíduo.

A cada geração, os operadores genéticos são aplicados:

- **Seleção por torneio:** três indivíduos são sorteados; o de maior aptidão (cobertura ponderada) é selecionado como pai;

Algorithm 8 Simulated Annealing para maximização de cobertura

```

1: procedure SIMULATEDANNEALING( $data, p, T_0, \gamma, n_{iter}$ )
2:    $\mathcal{I} \leftarrow$  solução aleatória de  $p$  UBSs
3:    $melhor \leftarrow \mathcal{I}, melhor\_fit \leftarrow f(\mathcal{I})$ 
4:    $T \leftarrow T_0$ 
5:   for  $k = 1$  to  $n_{iter}$  do
6:      $\mathcal{I}' \leftarrow \text{swap}(\mathcal{I}) \triangleright$  troca uma UBS
7:      $\Delta \leftarrow f(\mathcal{I}') - f(\mathcal{I})$ 
8:     if  $\Delta > 0$  or  $\text{random}() < e^{\Delta/T}$  then
9:        $\mathcal{I} \leftarrow \mathcal{I}'$ 
10:      if  $f(\mathcal{I}) > melhor\_fit$  then
11:         $melhor \leftarrow \mathcal{I}, melhor\_fit \leftarrow f(\mathcal{I})$ 
12:      end if
13:    end if
14:     $T \leftarrow T \cdot \gamma$ 
15:  end for
16:  return  $melhor$ 
17: end procedure

```

- **Crossover uniforme com reparo:** para cada par de pais, genes concordantes são herdados diretamente; genes discordantes são sorteados, e o cromossomo resultante é reparado para garantir exatamente p genes ativos;
- **Mutação por swap:** com probabilidade $\mu = 0.1$, um gene ativo e um inativo são trocados;
- **Elitismo:** o melhor indivíduo de cada geração é copiado intacto para a geração seguinte.

A implementação utiliza população de $n_{\text{pop}} = 50$ indivíduos por $n_{\text{ger}} = 100$ gerações, com taxa de crossover de 0.9.

NSGA-II (Non-dominated Sorting Genetic Algorithm II): O NSGA-II [24] é uma extensão multi-objetivo do GA que busca simultaneamente maximizar dois objetivos conflitantes:

- f_1 : cobertura ponderada total (mesma dos demais algoritmos);
- f_2 : equidade entre distritos, definida como $1 - G$, em que G é o coeficiente de Gini das taxas de cobertura populacional por distrito administrativo.

A equidade f_2 incentiva que a cobertura seja distribuída uniformemente entre os distritos, evitando que a maximização da cobertura global concentre sensores em regiões de alto peso enquanto negligencia regiões periféricas.

O algoritmo segue a estrutura canônica do NSGA-II: ordenação por não-dominância rápida (*fast non-dominated sort*), cálculo de distância de aglomeração (*crowding distance*) para preservar diversidade na fronteira de Pareto, e seleção por torneio baseada em rank de dominância. A cada geração, a população de pais e filhos é combinada, ordenada por frente de Pareto, e os n_{pop} melhores indivíduos (considerando rank e crowding distance) são selecionados para a próxima geração.

Ao final, o ponto de joelho (*knee point*) da primeira fronteira de Pareto é selecionado como solução final. O ponto de joelho é aquele que maximiza a distância perpendicular à linha que conecta os extremos da fronteira no espaço normalizado $[0, 1]^2$, representando o melhor compromisso entre cobertura e equidade.

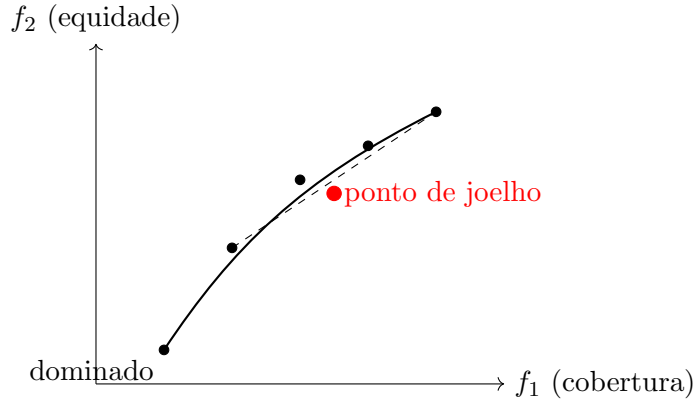


Figura 20: Representação esquemática da fronteira de Pareto entre cobertura (f_1) e equidade (f_2). O ponto de joelho (em vermelho) maximiza a distância à linha que conecta os extremos da fronteira.

Algorithm 9 NSGA-II para maximização de cobertura e equidade

```

1: procedure NSGAII( $data, p, n_{pop}, n_{ger}$ )
2:    $\mathcal{P}_0 \leftarrow$  inicializar  $n_{pop}$  cromossomos binários com  $p$  genes ativos
3:   Avaliar  $f_1$  (cobertura) e  $f_2$  (equidade) para  $\mathcal{P}_0$ 
4:   for  $t = 1$  to  $n_{ger}$  do
5:      $\mathcal{Q}_t \leftarrow \emptyset$ 
6:     for  $i = 1$  to  $n_{pop}/2$  do
7:        $p_1 \leftarrow$  torneio binário por rank de dominância em  $\mathcal{P}_{t-1}$ 
8:        $p_2 \leftarrow$  torneio binário por rank de dominância em  $\mathcal{P}_{t-1}$ 
9:        $(c_1, c_2) \leftarrow$  crossover uniforme com reparo( $p_1, p_2$ )
10:       $c_1 \leftarrow$  mutação por swap( $c_1$ ),  $c_2 \leftarrow$  mutação por swap( $c_2$ )
11:       $\mathcal{Q}_t \leftarrow \mathcal{Q}_t \cup \{c_1, c_2\}$ 
12:    end for
13:    Avaliar  $f_1, f_2$  para  $\mathcal{Q}_t$ 
14:     $\mathcal{R}_t \leftarrow \mathcal{P}_{t-1} \cup \mathcal{Q}_t$ 
15:     $\mathcal{F} \leftarrow \text{nonDominatedSort}(\mathcal{R}_t) \triangleright$  fronts  $\mathcal{F}_1, \mathcal{F}_2, \dots$ 
16:     $\mathcal{P}_t \leftarrow \emptyset, i \leftarrow 1$ 
17:    while  $|\mathcal{P}_t| + |\mathcal{F}_i| \leq n_{pop}$  do
18:       $\mathcal{P}_t \leftarrow \mathcal{P}_t \cup \mathcal{F}_i$ 
19:       $i \leftarrow i + 1$ 
20:    end while
21:     $d_c \leftarrow \text{crowdingDistance}(\mathcal{F}_i)$ 
22:     $\mathcal{F}_i \leftarrow$  ordenar  $\mathcal{F}_i$  por  $d_c$  decrescente
23:     $\mathcal{P}_t \leftarrow \mathcal{P}_t \cup \mathcal{F}_i[: n_{pop} - |\mathcal{P}_t|]$ 
24:  end for
25:   $\mathcal{F}_1 \leftarrow$  primeira fronteira de Pareto de  $\mathcal{P}_{n_{ger}}$ 
26:   $k \leftarrow \text{kneePoint}(\mathcal{F}_1)$ 
27:  return cromossomo  $k$ 
28: end procedure

```

Particle Swarm Optimization (PSO): O PSO [46] é uma meta-heurística inspirada no comportamento social de bandos, na qual cada partícula representa uma solução candidata no espaço contínuo $[0, 1]^n$. A cada iteração, as partículas movem-se no espaço guiadas por sua melhor posição individual (*pbest*) e pela melhor posição global do enxame (*gbest*):

$$\mathbf{v}_i^{(t+1)} = \omega \cdot \mathbf{v}_i^{(t)} + c_1 \cdot \mathbf{r}_1 \odot (\mathbf{p}_i - \mathbf{x}_i^{(t)}) + c_2 \cdot \mathbf{r}_2 \odot (\mathbf{g} - \mathbf{x}_i^{(t)}), \quad (26)$$

$$\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t)} + \mathbf{v}_i^{(t+1)}, \quad (27)$$

em que $\omega = 0.7$ é o peso de inércia, $c_1 = c_2 = 1.5$ são coeficientes cognitivo e social, e $\mathbf{r}_1, \mathbf{r}_2$ são vetores aleatórios uniformes em $[0, 1]$. As velocidades são limitadas a $[-0.5, 0.5]$ e as posições a $[0, 1]$.

Para converter a posição contínua $\mathbf{x}_i$ em uma seleção discreta de p UBSs, ordenam-se as n coordenadas e selecionam-se os p maiores índices. A aptidão de cada partícula é a cobertura ponderada das p UBSs selecionadas.

A população é inicializada com $n_{\text{part}} = 30$ partículas, posicionadas a partir dos scores normalizados do WLC com ruído uniforme $U(-0.3, 0.3)$, e evolui por $n_{\text{iter}} = 100$ iterações. O *warm-start* com o score WLC acelera a convergência ao posicionar as partículas próximas a regiões promissoras do espaço de busca.

4.3 Análise comparativa entre os algoritmos

Tendo apresentado os treze algoritmos de otimização espacial na Seção 4.2, esta seção volta-se à comparação sistemática entre eles. A análise é conduzida em três frentes complementares. Primeiro, a Seção 4.3.1 examina as métricas de cobertura (populacional, de populações vulneráveis e distância média ponderada) obtidas por cada método em função do número de sensores instalados, permitindo identificar quais abordagens entregam a melhor qualidade de solução. Em seguida, a Seção 4.3.2 desloca o foco da qualidade para o custo computacional, medindo o tempo de execução real de cada implementação e discutindo sua complexidade assintótica, um critério prático relevante para qualquer aplicação que precise recalculiar soluções sob demanda. Por fim, a Seção 4.3.3 aprofunda a análise ao examinar, por meio do MCLP como método de referência, como a ponderação dos critérios altera a distribuição espacial dos sensores selecionados pela cidade.

Além de permitirem a escolha dos métodos com melhor relação entre cobertura e custo computacional, essas três análises evidenciam que a escolha do algoritmo e da ponderação dos critérios reflete decisões de política pública sobre onde e como priorizar o monitoramento da qualidade do ar na cidade.

4.3.1 Métricas de cobertura

As Figuras 21 a 24 apresentam, respectivamente, a cobertura populacional total, a cobertura da população socialmente vulnerável, a distância média ponderada até o dispositivo mais próximo e o índice de Gini da distância, todas em função do número de sensores instalados ($p \in \{5, 10, 20, 50, 100\}$), para os treze algoritmos descritos na Seção 4.2 mais os dois baselines de referência (*cetesb_only* e *random*).

Cobertura populacional (Figura 21): o ponto em $p = 0$ (*cetesb_only*) fixa o piso de comparação, no qual a rede atual cobre cerca de 39% da população do município. Já em $p = 5$, todos os métodos que de fato otimizam a cobertura de forma combinatória (*greedy*, *GRASP*, *Simulated*

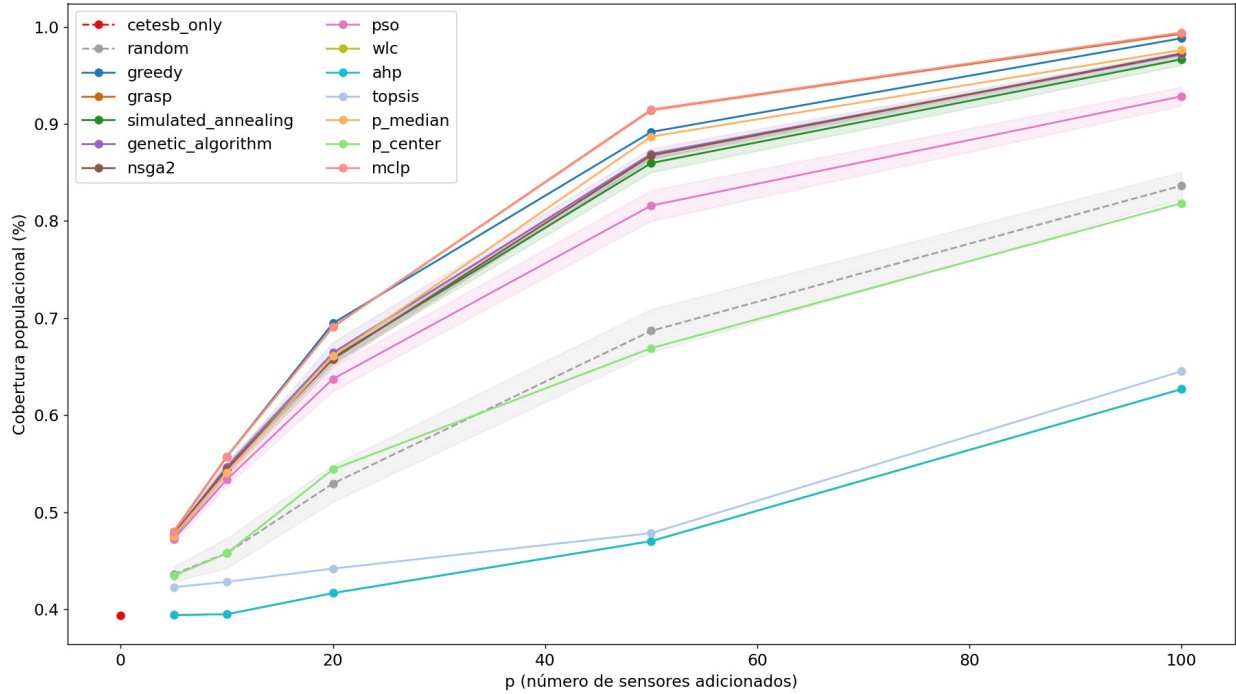


Figura 21: Cobertura populacional (%) em função de p (número de sensores adicionados), para os treze algoritmos de otimização espacial e os baselines `cetesb_only` e `random`. A faixa sombreada indica o desvio padrão entre repetições, quando aplicável.

Annealing, Genetic Algorithm, NSGA-II, p -median e MCLP) saltam para aproximadamente 48%, evidenciando que mesmo um número reduzido de sensores bem posicionados supera facilmente a rede atual. Esse grupo permanece coeso ao longo de todo o eixo de p , convergindo para 96–99% de cobertura em $p = 100$, com MCLP e GRASP consistentemente à frente. Este resultado era esperado, pois o MCLP resolve o problema de forma exata, enquanto o segundo combina uma construção semi-gulosa com busca local até a convergência.

O aspecto mais relevante desse gráfico, porém, é o comportamento dos métodos de scoring (WLC, AHP e TOPSIS). Como discutido na Seção 4.2.3, esses métodos pontuam cada UBS isoladamente, sem recalculando o ganho marginal considerando o que já foi coberto pelas UBSs anteriormente selecionadas. O efeito é sutil em p pequeno, mas se acumula: à medida que p cresce, uma fração crescente do orçamento é gasta em candidatas cujo raio de cobertura se sobrepõe ao de UBSs já escolhidas (afinal, os melhores escores tendem a estar espacialmente próximos, nos mesmos distritos densos e vulneráveis). O resultado é que TOPSIS estagna por volta de 63–65% e AHP em torno de 62–63% em $p = 100$, sendo um patamar notavelmente inferior ao do próprio `random`, e bem abaixo dos 97% do `greedy`. Isso quantifica exatamente o custo da não-adaptividade desta classe de algoritmos, que apresentam redundância espacial.

O `p_center` também fica atrás do grupo principal (67–82%), o que é coerente com sua função-objetivo que minimiza a distância máxima, não a cobertura ponderada, de modo que otimizar equidade no pior caso não é a mesma coisa que maximizar pessoas cobertas. Por fim, o PSO acompanha o grupo principal com uma defasagem pequena e estável (≈ 3 –4 pontos percentuais abaixo do `greedy` em todo o eixo), indicando que a conversão de posição contínua para seleção discreta (top- p por coordenada) introduz alguma perda de qualidade em relação aos métodos combinatórios

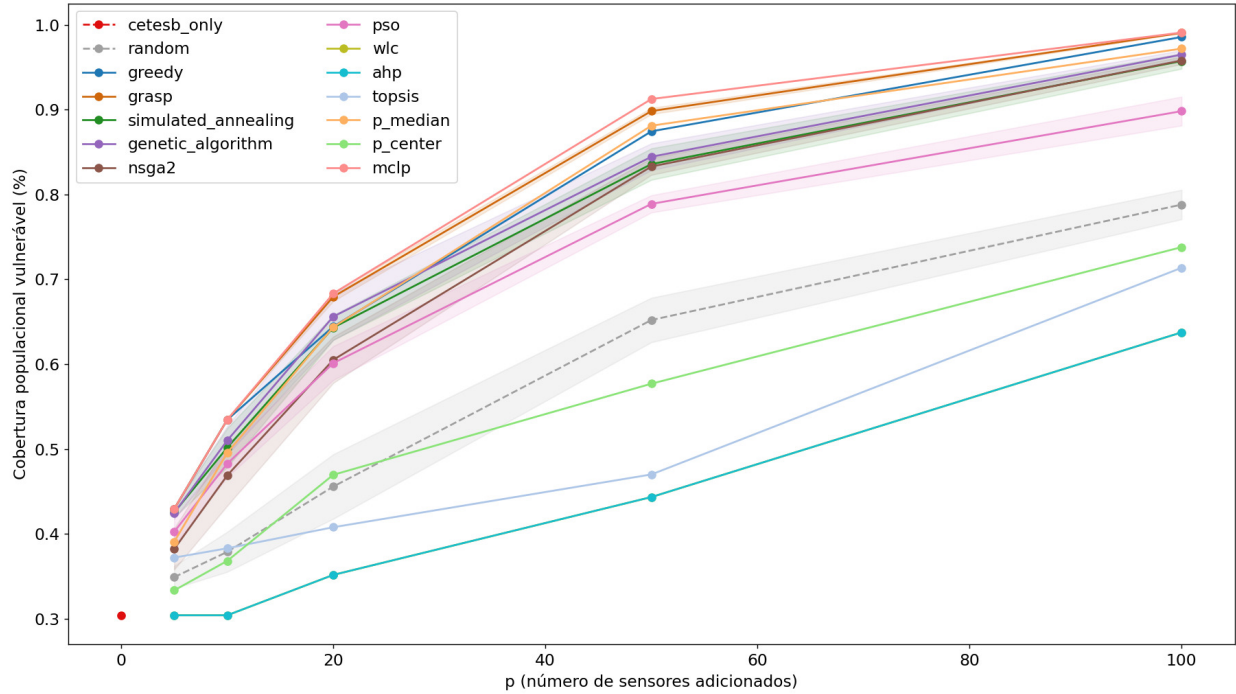


Figura 22: Cobertura da população socialmente vulnerável (%), definida como a fração da população residente em setores censitários de IPVS 4 ou 5 atendida por algum dispositivo, em função de p , para os treze algoritmos e os baselines `cetesb_only` e `random`.

diretos, mesmo com o warm-start a partir do score do WLC.

Cobertura da população vulnerável (Figura 22): o padrão geral repete o da Figura 21. Este comportamento é esperado, já que, como os pesos $\alpha = \beta = \gamma = 1$ tratam população, saúde e IPVS com a mesma importância, maximizar a cobertura ponderada tende a beneficiar proporcionalmente a população vulnerável. Ainda assim, a defasagem dos métodos mais fracos se acentua. Em $p = 100$, por exemplo, o `p_center` cai para 74% (vulnerável) contra 82% (total), sugerindo que, ao ignorar completamente os pesos multicritério e otimizar apenas a distância máxima geométrica, o `p-center` tende a “gastar” sensores em regiões espacialmente extremas que nem sempre coincidem com as mais vulneráveis, ou seja, otimizar equidade geográfica não é equivalente a otimizar equidade social.

Distância média ponderada (Figura 22): este gráfico inverte a leitura anterior (menor é melhor) e permite avaliar os métodos sob uma métrica contínua, não apenas binária de cobertura. Como esperado pela própria formulação do problema, o `p-median` é o método com menor distância média em praticamente todos os valores de p , o que é natural pois sua função-objetivo é exatamente essa. O grupo formado por `greedy`, `GRASP`, `MCLP`, `Simulated Annealing`, `Genetic Algorithm` e `NSGA-II` acompanha de perto o `p-median`, indicando que otimizar cobertura ponderada binária (dentro do raio R) tem como efeito colateral positivo a redução da distância média, indicando, portanto, que os dois objetivos estão fortemente correlacionados nesta instância do problema.

Já `AHP` e `TOPSIS` permanecem com a maior distância média em todo o eixo (caindo de ≈ 3.6 km em $p = 5$ para apenas ≈ 2.2 – 2.4 km em $p = 100$, contra ≈ 1.1 – 1.3 km dos métodos combinatórios),

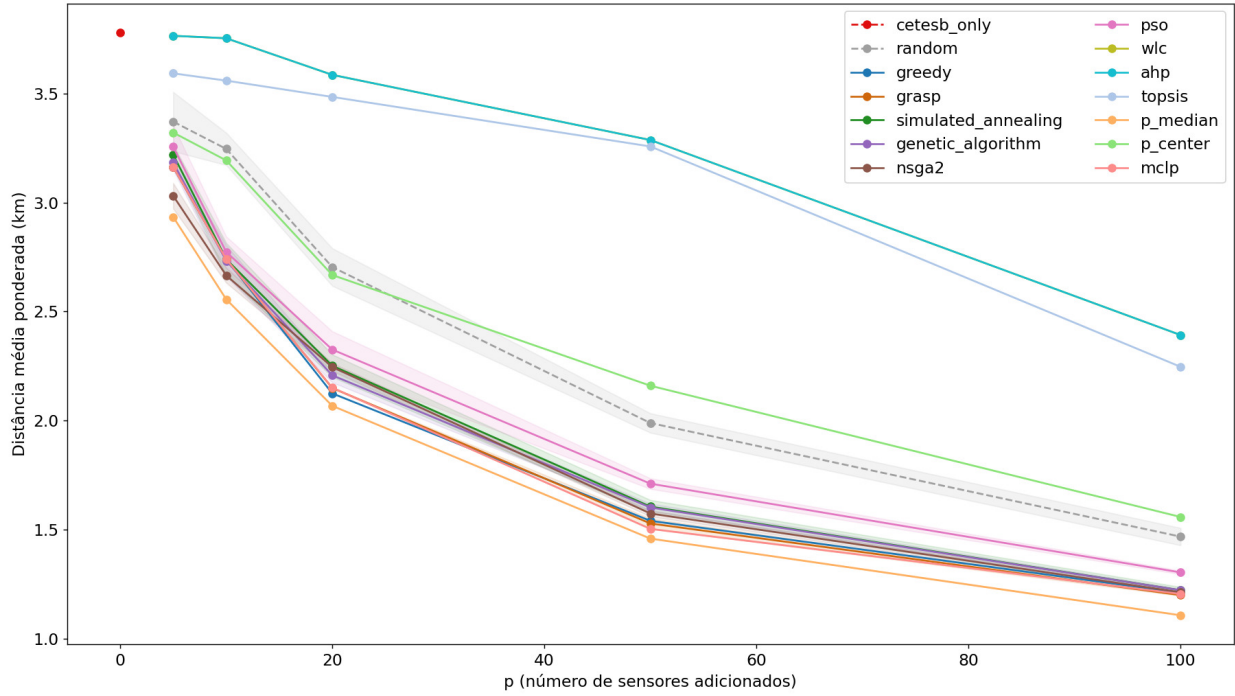


Figura 23: Distância média ponderada (km) entre os setores censitários e o dispositivo de monitoramento mais próximo (CETESB ou UBS selecionada), em função de p , para os treze algoritmos e os baselines `cetesb_only` e `random`.

reforçando a mesma conclusão da Figura 21 de que a ausência de recálculo de ganho marginal faz esses métodos escolherem UBSs redundantes entre si, deixando setores inteiros sem nenhuma instalação próxima. O `p_center`, por sua vez, ocupa uma posição intermediária, sendo um resultado ilustrativo do trade-off da sua formulação minimax que, ao concentrar esforço em reduzir o pior caso, não necessariamente minimiza a distância típica, deixando uma parcela grande de setores em distâncias medianas em vez de poucas bem servidas.

Equidade espacial - índice de Gini da distância (Figura 24) : os métodos de cobertura ponderada (`greedy`, `GRASP`, `MCLP`, `p-median`, `Simulated Annealing`, `Genetic Algorithm` e `NSGA-II`) reduzem o índice de Gini à medida que p cresce, convergindo para ≈ 0.25 – 0.27 em $p = 100$ (abaixo até do valor da rede atual da CETESB, 0.29), enquanto `AHP` e `TOPSIS` aumentam o Gini continuamente, chegando a 0.40–0.42, sendo o pior resultado entre todos os métodos testados.

A explicação está na natureza binária da cobertura combinada com o mecanismo de seleção de cada família de métodos. Nos métodos adaptativos, cada nova UBS é escolhida para cobrir setores ainda não cobertos; conforme p cresce, uma fração cada vez maior da população passa a ter distância pequena ($\leq R$) ao sensor mais próximo, homogeneizando a distribuição de distâncias e reduzindo o Gini. Já nos métodos de scoring, como discutido acima, os sensores adicionais tendem a se sobrepor aos já instalados, sendo que uma fração dos setores fica com distâncias muito pequenas (os poucos “polos” bem servidos), enquanto uma fração crescente permanece com distâncias grandes e praticamente inalteradas, configurando assim uma distribuição cada vez mais desigual, o que o índice de Gini captura corretamente.

Particularmente, o caso do `p_center` revela que, apesar de sua formulação minimax ser nor-

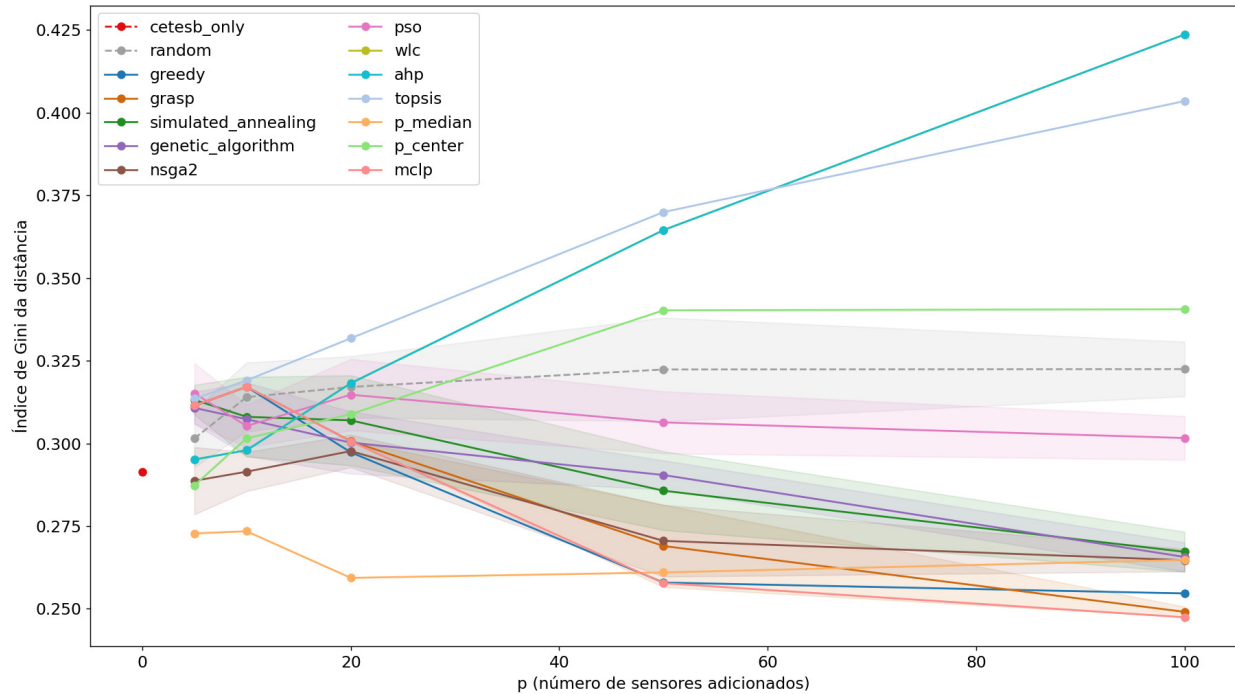


Figura 24: Índice de Gini da distância média ponderada por distrito administrativo, em função de p , para os treze algoritmos e os baselines `cetesb_only` e `random`. Valores mais altos indicam maior desigualdade espacial na distribuição da cobertura entre distritos.

malmente associada a soluções equitativas, seu Gini de distância se mantém elevado e crescente ($\approx 0,34$), ficando mais próximo do comportamento de AHP/TOPSIS do que do grupo de cobertura. Isso ocorre porque o p -center otimiza exclusivamente a distância máxima sem qualquer preocupação com a distribuição do restante das distâncias. Uma explicação possível para este fenômeno é que, ao reduzir a distância do setor mais isolado, o algoritmo deixe a maior parte dos demais setores em distâncias medianas relativamente dispersas, em vez de aproximá-los do raio de cobertura R como fazem os métodos baseados em cobertura binária. Em outras palavras, minimizar o pior caso não é equivalente a minimizar a desigualdade da distribuição como um todo — uma distinção conceitual importante entre “equidade no extremo” (p -center) e “equidade agregada” (Gini).

Por fim, o `random` mantém o Gini praticamente estável (≈ 0.30 – 0.32) em todo o eixo de p , indicando que sem nenhum critério de seleção espacial, a distribuição de distâncias resultante de uma amostra aleatória tende a manter sua forma estatística independentemente de quantos pontos adicionais forem sorteados, servindo como uma espécie de “nível de base” de desigualdade contra o qual os demais métodos podem ser comparados.

Síntese: As quatro métricas, lidas em conjunto, convergem para uma mesma conclusão qualitativa: o fator determinante do desempenho não é a sofisticação do algoritmo em si, mas sim se o método recalcula o ganho marginal a cada instalação (`greedy`, `GRASP`, `Simulated Annealing`, `Genetic Algorithm`, `NSGA-II`, p -median, `MCLP`) ou se pontua as candidatas de forma independente e estática (`WLC`, `AHP`, `TOPSIS`). O primeiro grupo entrega cobertura populacional e de vulneráveis muito próximas do ótimo (`MCLP`), menor distância média e maior equidade espacial conforme p cresce; o segundo grupo estagna em patamares comparáveis a uma seleção aleatória de

UBSs, tanto em cobertura quanto, de forma mais acentuada, em equidade. O *p_center* ocupa uma posição peculiar, superando a seleção aleatória em cobertura e distância média, mas não entrega necessariamente maior equidade agregada, evidenciando que seu critério minimax responde a uma noção diferente (e mais restrita) de equidade do que a capturada pelo índice de Gini.

4.3.2 Tempo de execução

Além da qualidade da cobertura, o custo computacional de cada implementação é um critério prático relevante, sobretudo se o objetivo consiste em obter uma ferramenta interativa capaz de recalculando a solução sob demanda. Esta seção consolida a complexidade assintótica dos catorze métodos apresentados e mede o tempo de execução real de cada um sobre a base de dados completa da cidade de São Paulo, com cinco repetições por combinação de método e p .

Metodologia experimental: Os tempos de cada método foram obtidos executando o framework sobre a mesma base de dados (Tabela 14) utilizada nas métricas de cobertura. Considerando a presença de algoritmos não determinísticos no framework, escolhemos por reportar a média e o desvio padrão dos tempos de execução para 5 *runs* distintas e diferentes números p de candidatos a instalação ($p \in \{5, 10, 20, 50, 100\}$), totalizando 330 tarefas em subprocessos independentes.

Para todos os testes desta seção, utilizamos um computador equipado com 16GB de RAM e uma CPU AMD Ryzen 5 3500X com seis *cores*. Os métodos determinísticos também rodaram uma vez por semente, o que não altera o resultado mas permite estimar a variabilidade do tempo de parede devida a fatores externos, como a contenção entre processos. As 330 tarefas completaram sem erros em aproximadamente 2 horas e 21 minutos de tempo total de parede.

Resultados empíricos: A Tabela 15 apresenta o tempo de execução médio (segundos), com desvio padrão entre parênteses, para os catorze métodos em função de p ., enquanto a Figura 25 mostra o mesmo em escala logarítmica, necessária dada a amplitude de quase cinco ordens de grandeza.

Método	$p = 5$	$p = 10$	$p = 20$	$p = 50$	$p = 100$
random	0,0011 (0,0017)	0,0002 (0,0000)	0,0002 (0,0000)	0,0002 (0,0000)	0,0002 (0,0000)
wlc	0,09 (0,00)	0,09 (0,00)	0,09 (0,01)	0,09 (0,01)	0,09 (0,00)
topsis	0,12 (0,02)	0,12 (0,01)	0,11 (0,00)	0,12 (0,00)	0,12 (0,00)
ahp	0,14 (0,02)	0,11 (0,01)	0,11 (0,02)	0,10 (0,00)	0,11 (0,01)
greedy	0,32 (0,02)	0,60 (0,01)	1,12 (0,04)	2,67 (0,05)	5,00 (0,10)
p_center	0,87 (0,00)	1,69 (0,01)	3,96 (0,36)	8,51 (0,32)	16,95 (0,35)
simulated_annealing	0,92 (0,03)	1,67 (0,03)	3,22 (0,10)	7,86 (0,41)	15,12 (0,25)
p_median [†]	4,62 (1,16)	7,20 (1,76)	21,36 (8,07)	57,15 (4,41)	275,88 (147,32)
mclp	5,41 (0,40)	7,17 (0,61)	12,94 (0,77)	31,19 (1,24)	35,20 (0,88)
pso	7,25 (0,39)	13,62 (0,39)	26,87 (0,97)	63,69 (3,05)	128,73 (11,71)
genetic_algorithm	8,81 (0,14)	16,75 (0,11)	32,97 (1,04)	74,28 (2,10)	136,39 (1,13)
grasp	72,80 (8,19)	269,94 (14,27)	765,00 (69,18)	2058,90 (105,97)	4457,59 (369,84)
nsga2	103,45 (1,66)	121,92 (2,47)	155,97 (1,18)	248,11 (5,62)	437,38 (34,99)

Tabela 15: Tempo de execução em segundos, média (desvio padrão) sobre 5 sementes. [†]variância elevada por contenção de recursos, discutida no texto.

Ordenando pelo tempo médio em $p = 5$, o espectro cobre cerca de cinco ordens de grandeza: de 0,001 s (random) a 103,4 s (NSGA-II). O padrão previsto pela Tabela 16 se confirma na Figura 25, em que os métodos WLC, AHP e TOPSIS permanecem praticamente constantes com p ; greedy, *p_center*, Simulated Annealing, PSO e Genetic Algorithm crescem de forma limpa e próxima do

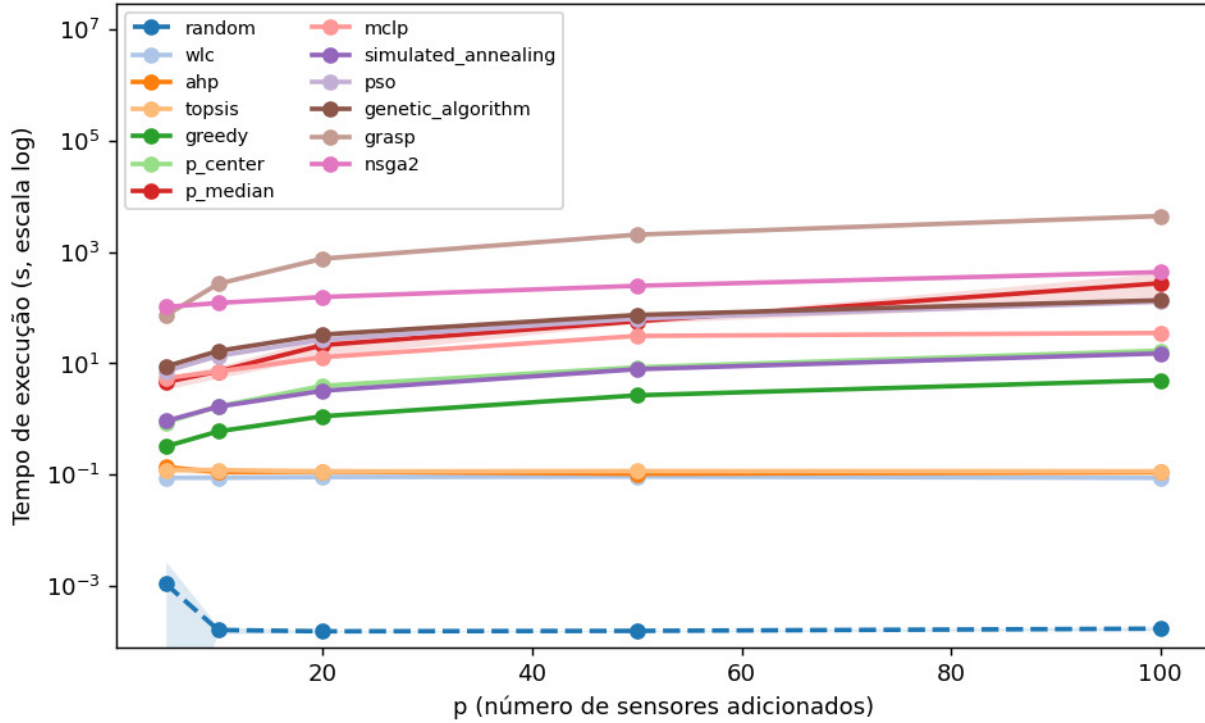


Figura 25: Tempo médio de execução em função de p para os catorze métodos medidos, com banda de ± 1 desvio padrão sobre 5 sementes (escala logarítmica no eixo do tempo). O método `cetesb_only` foi omitido por ter tempo próximo de zero, indefinido em escala log.

linear (o greedy cresce 15,5 vezes e o `p_center` cresce 19,5 vezes quando p cresce 20 vezes), com desvio padrão pequeno em relação à média.

O GRASP é o caso mais superlinear: cresce 61 vezes quando p cresce 20 vezes, chegando a 4 458 segundos (quase 1h14m) em $p = 100$, sinal de que o número de passes de busca local cresce com p , além do nominal $\mathcal{O}(\text{iter} \cdot p \cdot n \cdot |\mathcal{A}|)$.

O NSGA-II é sublinear em p : cresce só 4,2 vezes de $p = 5$ a $p = 100$, contra 15,5 vezes do GA (mesma estrutura populacional). Isso confirma que seu custo é dominado por um termo quase independente de p : o laço não vetorizado do objetivo de equidade f_2 sobre as 27 301 áreas, que já custa 103,4 s em $p = 5$.

O MCLP resolveu todos os pontos de p com status ótimo, sem nenhum atingir *timeout* de execução fornecido ao *solver*. O tempo cresce suavemente de 5,4 segundos ($p = 5$) a 35,2 segundos ($p = 100$), bem abaixo do *timeout* de 60s. O raio maior da CETESB ($R_{\text{exist}} = 3$ km) provavelmente já cobre boa parte dos setores antes de qualquer UBS ser escolhida, reduzindo as decisões binárias relevantes; ainda assim, o MCLP continua NP-difícil no pior caso, e esse resultado não deve ser generalizado para outras configurações de maior valor de p .

O `p_median` ilustra um efeito de contenção: apesar de ter complexidade idêntica à do `p_center`, seu desvio padrão relativo é muito maior (53% da média em $p = 100$, contra 2% do `p_center`). Como os dois algoritmos compartilham a mesma estrutura, a diferença vem do agendamento das tarefas: o `p_median` às vezes concorre pelos mesmos núcleos com tarefas muito mais pesadas (GRASP, NSGA-II).

Complexidade assintótica: A Tabela 16 resume a complexidade de tempo de cada implementação, obtida por inspeção direta do código. Hiperparâmetros (iterações, tamanho de população etc.) são tratados como constantes, independentes de $|\mathcal{A}|, n, p$. Ter dois raios de cobertura em vez de um não altera a ordem de complexidade de nenhum método.

Apenas os métodos de *scoring* (WLC, AHP, TOPSIS) são assintoticamente independentes de p , pois pontuam cada candidata isoladamente sem recalculer ganho marginal, ao custo de não evitar redundância de cobertura (Seção 4.2.3). Já o greedy e as variantes de p-mediana e p-centro pagam um fator adicional de p por reavaliarem o ganho marginal a cada instalação.

Custo-benefício e recomendação: Entre os métodos que de fato otimizam a cobertura de forma combinatória, o greedy (Algoritmo 1, Seção 4.2.1) segue sendo o de melhor custo-benefício: complexidade mínima $\mathcal{O}(p \cdot n \cdot |\mathcal{A}|)$ entre os métodos com garantia $(1 - 1/e)$, crescimento limpo com p , e de 2 a 3 ordens de grandeza mais rápido que GRASP ou NSGA-II em p grande (5,00s contra 437,4s e 4 457,6s em $p = 100$, Tabela 15). Também inspira a fase construtiva do GRASP (Algoritmo 7). Para $p \gtrsim 50$, é o método recomendado quando tempo de resposta importa mais que a otimalidade estrita do MCLP.

4.3.3 Análise espacial da alocação (MCLP)

Além da comparação quantitativa entre os métodos, é instrutivo examinar como a escolha dos pesos na fórmula multicritério influencia a distribuição espacial dos sensores na cidade. Para isso, utilizou-se o MCLP como método de referência, por ser o único que fornece a solução exata (ótimo global) do problema de cobertura, e realizou-se uma análise de sensibilidade sistemática sobre a ponderação dos critérios. A ideia é simples: se o modelo aloca os sensores em lugares diferentes quando o gestor muda os pesos, então os critérios estão de fato capturando dimensões distintas do problema. Se, por outro lado, a solução permanece a mesma independentemente dos pesos, os critérios são redundantes.

Para operacionalizar essa análise, cada um dos quatro critérios (população, saúde, IPVS e exposição) foi progressivamente priorizado. Para um critério c , atribuiu-se-lhe peso relativo λ e distribuiu-se o peso restante $(1 - \lambda)$ igualmente entre os três demais, de modo que cada um dos outros recebe $(1 - \lambda)/3$. Variou-se $\lambda \in \{0,4; 0,6; 0,8; 1,0\}$, abrangendo desde uma priorização moderada (40% do peso total, contra 20% para cada um dos demais) até uma priorização exclusiva ($\lambda = 1,0$, em que apenas aquele critério é considerado). O MCLP foi resolvido para $P = 20$ sensores em cada uma das 16 configurações resultantes (4×4). O número de sensores foi fixado porque seu efeito sobre a cobertura já é analisado na Seção 4.3.1; aqui, o interesse é isolar o efeito da ponderação, mantendo o orçamento de sensores constante. O padrão observado mantém-se qualitativamente estável ao longo de diferentes valores de P .

A Figura 26 apresenta os mapas resultantes dessa análise, e a Tabela 17 resume as métricas de cobertura nos casos extremos ($\lambda = 1,0$).

Critério de população. Quando a população é o critério dominante ($\lambda = 1,0$), o modelo busca cobrir o maior número absoluto de habitantes. A cobertura populacional atinge 68,1%, o valor mais alto entre todas as configurações testadas. Esse resultado é esperado, uma vez que o modelo está otimizando exatamente essa dimensão. No entanto, ao maximizar a cobertura populacional bruta, o modelo tende a privilegiar os setores mais densamente povoados, que em São Paulo concentram-se em faixas de renda intermediária. Isso se traduz em um IPVS médio coberto de apenas 2,84, o menor entre os quatro critérios, indicando que a solução voltada à população pura beneficia proporcionalmente menos as comunidades mais vulneráveis. A saúde média coberta também cai

Complexidade de tempo assintótica por implementação		
Método	Complexidade	Observação
cetesb_only	$\Theta(1)$	Não instala nada, apenas avalia a rede existente.
random	$\Theta(n)$	Amostragem sem reposição.
wlc / ahp	$\mathcal{O}(n \cdot \mathcal{A})$	Um único produto matricial. Não depende de p , pois apenas ordena e recorta o top- p , conforme já discutido na Seção 4.2.3.
topsis	$\mathcal{O}(n \cdot \mathcal{A})$	Construção da matriz de critérios domina o custo. Normalização e distâncias custam apenas $\mathcal{O}(n)$.
greedy	$\mathcal{O}(p \cdot n \cdot \mathcal{A})$	Conforme Seção 4.2.1 e Algoritmo 1: p passos, cada um varrendo as n candidatas restantes com ganho marginal $\mathcal{O}(\mathcal{A})$.
p_median / p_center	$\mathcal{O}(p \cdot n \cdot \mathcal{A})$	Construção gulosa (Algoritmos 3, 4) igual à anterior, seguida de busca local 1-swap com n_{swaps} passes (constante, padrão 3) de mesmo custo.
mclp	NP-difícil, pior caso $\mathcal{O}(2^{ \mathcal{A} +n})$	<i>Maximum Coverage</i> exato via <i>branch-and-bound</i> (CBC, Algoritmo 2). O tempo prático depende fortemente da densidade de cobertura da instância, inclusive dos raios R_{cand} e R_{exist} usados, e é limitado por <i>timeout</i> .
grasp	$\mathcal{O}(\text{iter} \cdot p \cdot n \cdot \mathcal{A})$ nominal	Construção semi-gulosa vetorizada (Algoritmo 7). A busca local por <i>swap</i> incremental custa $\mathcal{O}(\mathcal{A})$ por candidato avaliado, mas o número de passes até convergência cresce empiricamente com p (discussão adiante).
simulated_annealing	$\mathcal{O}(\text{max_iter} \cdot p \cdot \mathcal{A})$	Um único vizinho por iteração (Algoritmo 8). O <i>fitness</i> é recomputado do zero, custando $\mathcal{O}(p \cdot \mathcal{A})$.
genetic_algorithm	$\mathcal{O}(\text{ger} \cdot \text{pop} \cdot p \cdot \mathcal{A})$	Avaliação de <i>fitness</i> por cromossomo a cada geração.
nsga2	$\mathcal{O}(\text{ger} \cdot \text{pop} \cdot \mathcal{A})$	O custo do objetivo de equidade f_2 (Seção 4.2.4) domina: laço não vetorizado sobre as $ \mathcal{A} $ áreas por indivíduo, para o agrupamento por distrito usado no coeficiente de Gini. Esse custo é quase independente de p .
pso	$\mathcal{O}(\text{iter} \cdot \text{part} \cdot (p \cdot \mathcal{A} + n \log n))$	Ordenação do vetor de posição ($\mathcal{O}(n \log n)$) mais avaliação de cobertura por partícula a cada iteração.

Tabela 16: Complexidade de tempo assintótica por implementação.

para 4,56, contra 4,78 quando a saúde é o critério dominante. Em outras palavras, maximizar a quantidade de pessoas cobertas não equivale a maximizar a proteção das populações que mais sofrem com a poluição atmosférica.

Critério de saúde (internações respiratórias). Ao priorizar a saúde ($\lambda = 1,0$), o modelo direciona os sensores para os distritos com maior taxa de internação por doenças respiratórias. A saúde média coberta sobe de 4,73 ($\lambda = 0,4$) para 4,78 ($\lambda = 1,0$), e um resultado notável é que o IPVS médio coberto também se eleva, passando de 2,94 para 2,97. Esse comportamento revela uma **aliança** entre os critérios de saúde e de vulnerabilidade social: as internações por doenças respiratórias concentram-se nas mesmas periferias socialmente vulneráveis da cidade, de modo que priorizar um desses critérios beneficia automaticamente o outro. Essa correlação é consistente com a literatura epidemiológica, que documenta maior prevalência de doenças respiratórias em populações de baixa renda, expostas a piores condições de moradia e com menor acesso a serviços de saúde. Do ponto de vista da alocação, isso significa que o gestor não precisa escolher entre proteger a saúde e proteger os vulneráveis: os dois objetivos apontam na mesma direção espacial.

Critério de vulnerabilidade social (IPVS). A configuração que prioriza exclusivamente o IPVS ($\lambda = 1,0$) produz o IPVS médio coberto mais alto de todo o sweep: 3,05, contra uma média municipal de 2,46. Em contrapartida, a cobertura populacional é a segunda menor (65,8%), refletindo o clássico compromisso entre eficiência e equidade: ao direcionar sensores para regiões mais vulneráveis, que tendem a ser menos densamente povoadas, o modelo sacrifica parte da cobertura total em favor de uma distribuição mais justa.

Espacialmente, essa configuração concentra os 20 sensores nas periferias da cidade. Na zona leste, aparecem distritos como Cidade Tiradentes, Guaianases, Itaim Paulista, Itaquera, José Bonifácio, Sapopemba, Ermelino Matarazzo e São Rafael. Na zona sul, o modelo seleciona UBS em Grajaú, Cidade Dutra, Jardim Ângela, Jardim São Luís, Pedreira, Sacomã, Jabaquara e Campo Limpo. Na zona norte, os sensores vão para Cachoeirinha, Jaraguá e Jaçanã. O IPVS médio dos distritos selecionados é de 3,31, significativamente acima da média da cidade (2,46). A presença de distritos como Jardim Ângela e Grajaú, dois dos mais vulneráveis do município, ilustra como o modelo identifica corretamente as regiões de maior necessidade. Esse padrão espacial contrasta diretamente com a rede atual da CETESB, que é concentrada na porção central da cidade (Figura 18): os novos sensores preenchem precisamente a lacuna de monitoramento das periferias.

Critério de exposição viária. A exposição é o único critério que se comporta como **antagonista** dos demais. Ao priorizá-la ($\lambda = 1,0$), a exposição média coberta sobe de 1,97 para 2,23, indicando que o modelo de fato encontra regiões com maior densidade de vias de tráfego intenso. Porém, simultaneamente, o IPVS médio coberto cai para 2,67, o menor valor entre todas as configurações, e a cobertura populacional recua para 64,0%, também a menor. A saúde média coberta sofre a queda mais acentuada, de 4,70 ($\lambda = 0,4$) para 4,27 ($\lambda = 1,0$), uma redução de quase 10% que evidencia o desalinhamento espacial entre as fontes de poluição e as populações mais afetadas por ela.

Ao examinar os distritos selecionados quando a exposição é o critério exclusivo, observa-se um deslocamento marcante em relação às configurações sociais. Na zona leste, distritos como Tatuapé, Penha, Ponte Rasa e São Lucas substituem os mais periféricos Cidade Tiradentes, Guaianases e Itaim Paulista que apareciam na configuração do IPVS. Na zona sul, Cursino e Ipiranga tomam o lugar de Grajaú, Cidade Dutra e Jardim Ângela. Na zona norte, Pirituba e Vila Maria substituem Cachoeirinha. O IPVS médio dos distritos selecionados nessa configuração é de 2,74, bem

mais próximo da média municipal (2,46) do que os 3,31 observados na configuração do IPVS. Isso confirma que, ao priorizar a exposição, os sensores migram para regiões de classe média, onde os grandes corredores viários (marginais, avenidas radiais, vias expressas) concentram o tráfego.

Essa inversão espacial não é um defeito do modelo, mas sim a consequência de uma distinção conceitual fundamental entre os critérios. Os critérios sociais e de saúde miram o *receptor* da poluição, ou seja, as populações que sofrem seus efeitos na forma de doenças respiratórias e vulnerabilidade social. Já a exposição viária mira a *fonte* da poluição, isto é, o tráfego veicular, que é a principal origem dos poluentes atmosféricos no ambiente urbano e que se concentra nos corredores estruturais da cidade. Os critérios são, portanto, não redundantes: monitorar onde a poluição é gerada não é a mesma coisa que monitorar onde ela causa mais dano.

Configuração equilibrada e implicações para o gestor. A configuração equilibrada, em que os quatro critérios recebem peso igual ($\alpha = \beta = \gamma = \delta = 1$), produz uma solução que equilibra as duas perspectivas. Com $P = 20$, essa configuração cobre 67,2% da população, com IPVS médio de 2,96 e exposição média de 1,97. Os distritos selecionados nessa configuração distribuem-se por todas as zonas periféricas: na zona leste, aparecem Cidade Tiradentes, Guaianases, Itaim Paulista, Itaquera, Ermelino Matarazzo, Sapopemba, São Rafael, Artur Alvim e Vila Formosa; na zona sul, Grajaú, Cidade Dutra, Jardim São Luís, Pedreira, Jabaquara, Campo Limpo e Sacomã; na zona norte, Cachoeirinha, Jaraguá, Jaçanã e Vila Medeiros. Trata-se de 20 distritos distintos, com IPVS médio de 3,17, indicando que mesmo com a inclusão do critério de exposição, a solução equilibrada mantém um viés em favor das populações vulneráveis.

O valor dessa análise de sensibilidade reside em tornar explícito o espaço de decisão disponível ao gestor. A formulação multicritério não impõe uma resposta única; ao contrário, ela permite navegar entre configurações que privilegiam a cobertura das fontes de poluição, a proteção das populações mais vulneráveis, ou um compromisso entre ambos. Os pesos são o mecanismo pelo qual essa escolha é traduzida em uma solução espacial concreta, e a Figura 26 oferece ao tomador de decisão uma visualização direta de como cada ajuste de peso se materializa no mapa da cidade.

Critério priorizado	Pop. coberta (%)	IPVS médio	Saúde média	Exp. média
População	68,1	2,84	4,56	2,00
Saúde	66,2	2,97	4,78	1,95
IPVS	65,8	3,05	4,73	1,93
Exposição	64,0	2,67	4,27	2,23
Equilibrado ($\lambda = 0,25$)	67,2	2,96	4,72	1,97

Tabela 17: Métricas de cobertura nos extremos do *sweep* de pesos ($P = 20$, $\lambda = 1,0$). Cada linha corresponde ao critério exclusivamente priorizado. As médias são ponderadas pela população dos setores cobertos.

4.4 Limitações e trabalhos futuros

Os resultados apresentados devem ser interpretados à luz de algumas limitações, que também delineiam direções para trabalhos futuros.

Grão ecológico do critério de saúde. A taxa de internações respiratórias é calculada no nível de distrito administrativo e atribuída igualmente a todos os setores que o compõem, pois o

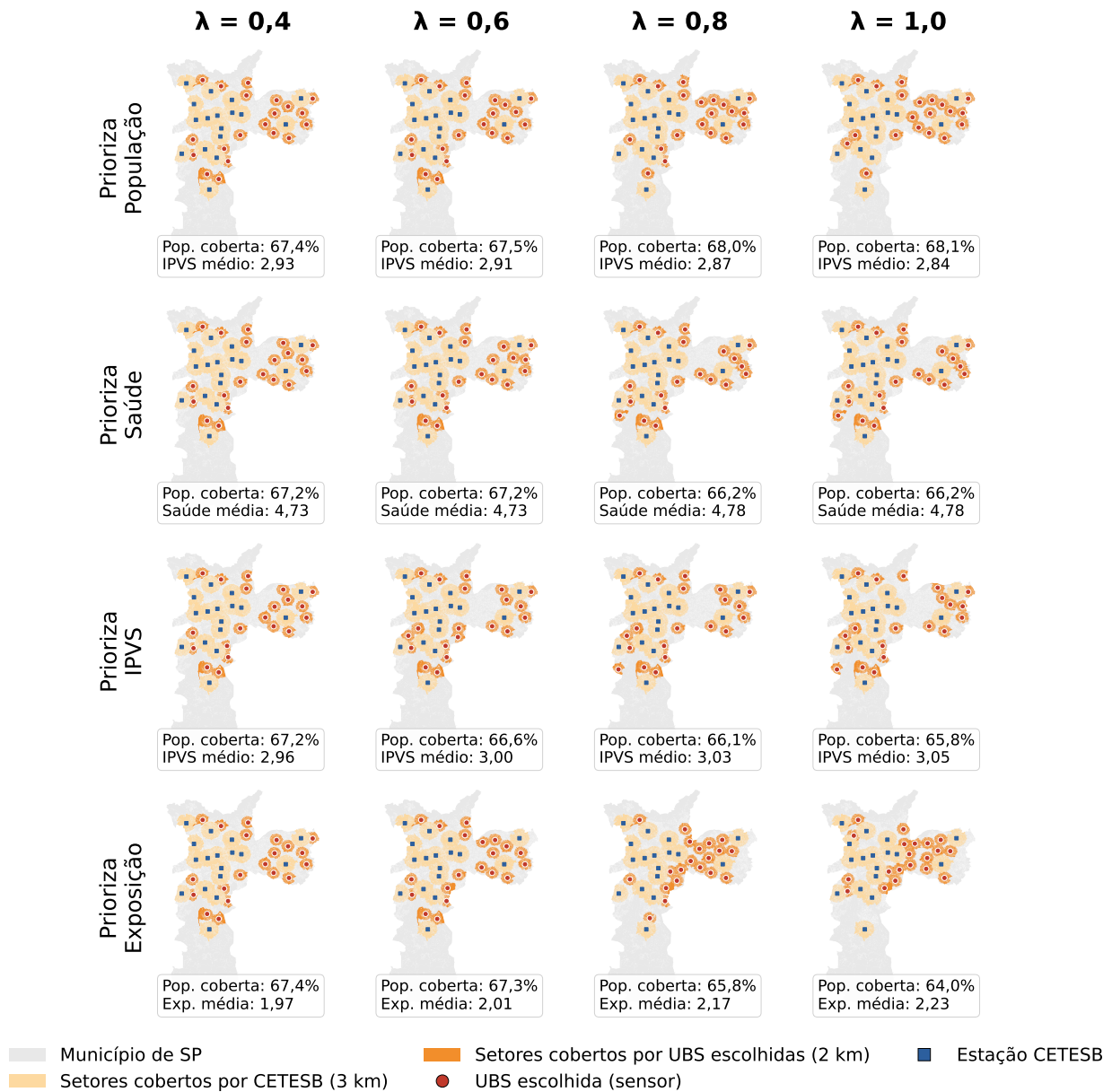


Figura 26: Análise de sensibilidade da alocação do MCLP à ponderação dos critérios, com $P = 20$ sensores e raio de cobertura de 2 km para as UBS candidatas e 3 km para as estações da CETESB. Em cada linha, um critério é progressivamente priorizado: atribui-se-lhe peso relativo λ (colunas) e distribui-se o peso restante $(1 - \lambda)$ igualmente entre os três demais. Em vermelho, as UBS selecionadas como pontos de instalação; em laranja, os setores cobertos pelos sensores instalados; em laranja-claro, os setores já cobertos pela rede CETESB existente. As anotações indicam a fração da população coberta e a média ponderada (pela população) do critério priorizado nos setores cobertos.

microdado do SIH permite geolocalizar o paciente apenas até esse nível. Trata-se de uma agregação ecológica: a variação intradistrital da morbidade é perdida, e a taxa de um setor não reflete necessariamente a sua realidade local. Optou-se deliberadamente por esse grão porque, no nível de setor, o baixo número de casos tornaria a taxa dominada por ruído estatístico. Uma fonte de saúde com resolução intraurbana mais fina — por exemplo, via geocodificação do endereço completo do paciente — permitiria refinar esse critério em trabalhos futuros.

Exposição como *proxy*. A densidade viária foi utilizada como um *proxy* da exposição à poluição de origem veicular por constituir uma variável amplamente disponível, de fácil obtenção e passível de reprodução em diferentes municípios. Embora não represente diretamente a intensidade do tráfego não incorporando informações como volume de veículos, variações temporais do fluxo ou processos de dispersão atmosférica, sua utilização permite construir um indicador consistente mesmo em contextos nos quais dados detalhados de contagem volumétrica de tráfego não estão disponíveis de forma pública, padronizada ou com cobertura espacial adequada. Assim, a opção pela densidade viária representa um compromisso entre realismo e aplicabilidade, favorecendo a replicabilidade da metodologia. Quando disponíveis, dados de volume de tráfego ou modelos de dispersão de poluentes podem ser incorporados para refinar o critério de exposição.

Cobertura binária. O modelo adota uma noção binária de cobertura: um setor é considerado coberto se houver um sensor dentro do raio R , e não coberto caso contrário. Não há cobertura fracionária nem decaimento da qualidade da medição com a distância, o que simplifica a realidade física da dispersão de poluentes. Uma formulação com cobertura gradual (por exemplo, com peso decrescente na distância) é uma extensão natural.

5 Conclusão

O trabalho desenvolvido teve como objetivo a pesquisa e o desenvolvimento de um sistema de monitoramento da qualidade do ar, utilizando a cidade de São Paulo como um estudo de caso.

Com a conclusão das três frentes de pesquisa propostas, foi possível desenvolver uma solução integrada capaz de coletar métricas de qualidade do ar consistentes com os padrões da rede de monitoramento estabelecida pela CETESB, atendendo a uma entrega de dados mesmo diante de falhas de conectividade e por meio dos algoritmos de escolha de locais ideais, é possível cobrir mais de 90% da população paulistana com um número viável de apenas 50 dispositivos, assim melhorando a granularidade e a representatividade dos dados.

Além disso, o desenvolvimento baseado no microcontrolador ESP32 confere grande flexibilidade ao projeto, permitindo a integração de novos sensores sem a necessidade de reestruturação da arquitetura existente. O mecanismo de coleta de dados pela TV Box opera de forma independente da quantidade de sensores acoplados, assim como o módulo de envio desses dados ao servidor, o que assegura a modularidade e a fácil manutenção do sistema.

Do ponto de vista da comunicação IoT, a arquitetura desenvolvida na Seção 2 demonstrou-se robusta, com capacidade comprovada para operar na cidade de São Paulo sem riscos de sobrecarga. Conforme discutido, ajustes pontuais no mecanismo de envio permitem estender a solução para todo o estado de São Paulo ou para a região metropolitana, mantendo a confiabilidade dos dados. Dessa maneira, o sistema se consolida como uma base sólida para implementação na maioria dos centros urbanos brasileiros, tendo em vista a densidade populacional e a extensão territorial típicas de grandes municípios.

Por fim, a metodologia de otimização espacial desenvolvida na Seção 4 fecha o ciclo proposto pelo estudo ao integrar o desenvolvimento técnico das Seções 2 e 3 na decisão estratégica sobre onde instalar as estações. A análise comparativa demonstrou que algoritmos como o MCLP e o Greedy são capazes de superar substancialmente a cobertura da rede atual da CETESB com um número reduzido de dispositivos, e que a ponderação dos critérios sociais, de saúde e de exposição permite direcionar essa expansão prioritariamente às populações mais vulneráveis e sub-monitoradas da cidade, contribuindo para a mitigação da desigualdade de monitoramento identificada.

Desta forma, as três frentes de pesquisa se complementam ao formarem, em conjunto, a base teórica completa para a proposta de rede de monitoramento, abrangendo sensores confiáveis, uma infraestrutura de dados robusta e uma estratégia de alocação fundamentada em equidade.

Portanto, a conclusão deste estudo é que o projeto atingiu maturidade teórica suficiente para avançar à transição do ambiente controlado para um espaço de teste. O próximo passo imediato consiste na construção de uma solução piloto (demo) em campo, com a instalação de um número reduzido de estações (entre 5 e 10 unidades) em pontos espaciais estrategicamente selecionados. Esse protótipo permitirá:

- i. validar a acurácia dos sensores de baixo custo em confronto direto com as estações de referência da CETESB ao longo de diferentes condições climáticas;
- ii. testar a resiliência física do hardware (TV Box + ESP32) diante de variações de energia, temperatura e intempéries;
- iii. verificar, na prática, a eficácia dos mecanismos de tolerância a falhas e atualização remota em cenários reais de rede instável;
- iv. e, por fim, ajustar os pesos dos algoritmos de otimização com base em dados empíricos de poluição efetivamente coletados, refinando as recomendações para a expansão em larga escala.

Somente com a execução bem-sucedida dessa fase de demonstração e validação prática será possível consolidar esta solução não apenas como um exercício acadêmico, mas como uma ferramenta operacional eficaz, escalável e pronta para subsidiar a gestão pública e a justiça climática no município de São Paulo e em outras cidades brasileiras.

Referências

- [1] Alibaba - plataforma de negócios b2b, 2026. Acesso em: 6 jul. 2026. URL: <https://www.alibaba.com/>.
- [2] Digikey - electronic parts and components, 2026. Acesso em: 6 jul. 2026. URL: <https://www.digikey.com/>.
- [3] Directindustry - b2b marketplace, 2026. Acesso em: 6 jul. 2026. URL: <https://www.directindustry.com/>.
- [4] Mouser electronics, 2026. Acesso em: 6 jul. 2026. URL: <https://br.mouser.com/>.
- [5] Marcos Abdo Arbex, Ubiratan de Paula Santos, Lourdes Conceição Martins, Paulo Hilário Nascimento Saldiva, Luiz Alberto Amador Pereira, and Alfésio Luis Ferreira Braga. A poluição do ar e o sistema respiratório. *Jornal Brasileiro de Pneumologia*, 38:643–655, 2012.

- [6] Brasil. Ministério do Meio Ambiente e Mudança do Clima. Conselho Nacional do Meio Ambiente - CONAMA. *Resolução nº 506, de 5 de julho de 2024*, edição 130, seção 1 edition, 2024.
- [7] Maria D. Castillo, Patrick L. Kinney, Veronica Southerland, C. Anneta Arno, Kelly Crawford, Aaron van Donkelaar, Melanie Hammer, Randall V. Martin, and Susan C. Anenberg. Estimating intra-urban inequities in $pm_{2.5}$ -attributable health impacts: A case study for washington, dc. *GeoHealth*, 5(11):e2021GH000431, 2021.
- [8] CETESB. Pregão cetesb, 2020. Acesso em: 9 out. 2025. URL: <https://sistemasinter.cetesb.sp.gov.br/servicos/INTERDESLICITACOES/Editais/EDITAL-24-2020-308-C.PDF>.
- [9] CETESB. Boletim mensal da qualidade do ar para o estado de são paulo. Technical report, Secretaria de Meio Ambiente, Infraestrutura e Logística, Governo do Estado de São Paulo, São Paulo, jan. 2026.
- [10] CETESB. Dados abertos – qualidade ambiental, 2026. Acesso em: 5 jul. 2026.
- [11] Cetesb. Redes de monitoramento cetesb, 2026. Acesso em: 30 jun. 2026. URL: https://www.cetesb.sp.gov.br/cetesb/qualidade_ambiental/ar/informacoes_basicas/redes_de_monitoramento.
- [12] Sourangsu Chowdhury, Sagnik Dey, and et al. Air pollution and health in south asia: A comprehensive review. *The Lancet Planetary Health*, 5(12):e846–e859, December 2021.
- [13] Richard Church and Charles ReVelle. The maximal covering location problem. In *Papers of the regional science association*, volume 32, pages 101–118. Springer-Verlag Berlin/Heidelberg, 1974.
- [14] CNES/DATASUS. Base de dados do cadastro nacional de estabelecimentos de saúde, 2026. Acesso em: 5 jul. 2026.
- [15] Companhia Ambiental do Estado de São Paulo (CETESB). Programa qualar. <https://www.cetesb.sp.gov.br/cetesb>, 2025. Acesso em: 30 jun. 2026.
- [16] Companhia Ambiental do Estado de São Paulo (CETESB). Qualidade do ar – mapa interativo. <https://arcgis.cetesb.sp.gov.br/portal/apps/experiencebuilder/experience/?id=13582bb92885456fb9f10ee84ad1507e&page=Classifica%C3%A7%C3%A3o-Atual>, 2025. Acesso em: 30 jun. 2026.
- [17] The Cron Project. *cron(8) – daemon to execute scheduled commands*, 2025. Acesso em: 30 jun. 2026.
- [18] Marc Cuadros. Ofelia – docker job scheduler. <https://hub.docker.com/r/mcuadros/ofelia>, 2025. Acesso em: 1 jul. 2026.
- [19] Gustavo P. C. P. da Luz, Gabriel Massuyoshi Sato, Luis Fernando Gomez Gonzalez, and Juliana Freitag Borin. Repurposing of tv boxes for a circular economy in smart cities applications. *Scientific Reports*, 15(1):22638, 2025.
- [20] Keller Dantara. Análise de btex e vocs: Entenda as diferenças e aplicações, 2026. Acesso em: 6 jul. 2026. URL: <https://www.polarisanalises.com.br/post/analise-btex-vocs-diferencas-aplicacoes>.

- [21] Dart Sensors. *Operation Manual for WZ-S HCHO Module*. ProSense Technologies Co., Ltd., 2026. Disponível em: <http://www.dart-sensors.com>. Acesso em: 7 de jul. 2026.
- [22] DATASUS. Transferência de arquivos, 2026. Acesso em: 5 jul. 2026.
- [23] Afonso Moraes de Moura, Klabin Riocell, Celso Edmundo Bochetti Foelkel, and Sonia Maria Bitencourt Frizzo. Análise de compostos reduzidos de enxofre: desenvolvimento de metodologia baseada em conjunto de elementos filtrantes seletivos. *Congresso Internacional de Celulose e Papel*, page 4, 2000.
- [24] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, 2002.
- [25] Mary Angelique G. Demetillo, Aracely Navarro, Katherine K. Knowles, Kimberly P. Fields, Jeffrey A. Geddes, Caroline R. Nowlan, Scott J. Janz, Laura M. Judd, Jassim Al-Saadi, Kang Sun, Brian C. McDonald, Glenn S. Diskin, and Sally E. Pusede. Observing nitrogen dioxide air pollution inequality using high-spatial-resolution remote sensing measurements in houston, texas. *Environmental Science & Technology*, 54(16):9882–9895, 2020.
- [26] Departamento de Economia e Estatística (DEE-SPGG). Impactos socioeconômicos dos eventos climáticos extremos de 2024 no rio grande do sul: Uma análise após um ano do desastre. Relatório, Secretaria de Planejamento, Governança e Gestão. Governo do Rio Grande do Sul, Porto Alegre, RS, 04 2025.
- [27] Luana Pavan Detoni, Carolina Rezende Faccin, Rogério Leandro Lima da Silveira, Geisa Zanini Rorato, and Brenda Eckel Machado. Eventos climáticos extremos e seus impactos socioespaciais em cidades pequenas do rio grande do sul-brasil. *Revista Brasileira de Gestão e Desenvolvimento Regional*, 21(1), 2025.
- [28] Docker Inc. docker compose. <https://docs.docker.com/reference/cli/docker/compose/>, 2026. Acesso em: 30 jun. 2026.
- [29] Eclipse Foundation. mosquitto.conf – the configuration file for mosquitto. <https://manpages.debian.org/unstable/mosquitto/mosquitto.conf.5.en.html>. Acesso em: 30 de junho de 2026.
- [30] Espressif Systems. *Analog to Digital Converter (ADC) — ADC Limitations*. Espressif Systems, esp-idf programming guide edition, 2024. Disponível em: <https://docs.espressif.com/projects/esp-idf/en/latest/esp32s3/api-reference/peripherals/adc.html#adc-limitations>. Acessado em: 7 de jul. 2026.
- [31] Espressif Systems. Comparing ADC performance of Espressif SoCs, 2025.
- [32] Espressif Systems, Shanghai, China. *ESP32-S3-WROOM-1 & ESP32-S3-WROOM-1U Datasheet: 2.4 GHz Wi-Fi (802.11b/g/n) and Bluetooth 5 module*, version 1.8 edition, 2026.
- [33] Laith Farhan, Rupak Kharel, Omprakash Kaiwartya, Marcela Quiroz-Castellanos, Ali Alissa, and Mohamed Abdulsalam. A concise review on internet of things (iot) -problems, challenges and opportunities. In *2018 11th International Symposium on Communication Systems, Networks & Digital Signal Processing (CSNDSP)*, pages 1–6, 2018.
- [34] Thomas A Feo and Mauricio GC Resende. Greedy randomized adaptive search procedures. *Journal of global optimization*, 6(2):109–133, 1995.

- [35] Fundação SEADE. Índice paulista de vulnerabilidade social (ipvs) – bases cartográficas, 2022. Acesso em: 5 jul. 2026.
- [36] S Louis Hakimi. Optimum locations of switching centers and the absolute centers and medians of a graph. *Operations research*, 12(3):450–459, 1964.
- [37] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.
- [38] Ching-Lai Hwang and Kwangsun Yoon. *Multiple attribute decision making: methods and applications a state-of-the-art survey*. Springer Science & Business Media, 2012.
- [39] IBGE. Censo demográfico 2022, 2022. Acesso em: 5 jul. 2026.
- [40] InfluxData. Influxdb documentation, 2025. Acesso em: 30 de junho de 2026.
- [41] Inmet. Normais climatológicas do brasil, 2026. Acesso em: 06 jul. 2026 URL: <https://portal.inmet.gov.br/normais>.
- [42] Instituto Brasileiro de Geografia e Estatística. São paulo - panorama estadual. <https://cidades.ibge.gov.br/brasil/sp/panorama>, 2026. Acesso em: 30 jun. 2026.
- [43] Instituto Brasileiro de Geografia e Estatística. São paulo - panorama municipal. <https://www.ibge.gov.br/cidades-e-estados/sp/sao-paulo.html>, 2026. Acesso em: 30 jun. 2026.
- [44] Instituto de Astronomia, Geofísica e Ciências Atmosféricas - IAG/USP. Gráficos do clima em tempo real, 2026.
- [45] Instituto Nacional de Meteorologia (INMET). El niño 2026: saiba detalhes sobre o monitoramento, previsões e os possíveis impactos do fenômeno no brasil, 6 2026. Boletim n.º 1 divulgado em parceria com INPE, ANA, CEMADEN, SGB e SEDEC.
- [46] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, volume 4, pages 1942–1948 vol.4, 1995.
- [47] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.
- [48] MDCR Lemes, Michelle Simões Reboita, and Bruno César Capucin. Impactos das queimadas na amazônia no tempo em são paulo na tarde do dia 19 de agosto de 2019. *Revista Brasileira de Geografia Física*, 13(3):983–993, 2020.
- [49] Livia Lopes. Análise de vocs e btex: quando investigar compostos orgânicos voláteis, 2025. Acesso em: 6 jul. 2026. URL: <https://www.lab2bio.com.br/post/an%C3%A1lise-de-vocs-e-btex-quando-investigar-compostos-org%C3%A2nicos-vol%C3%A1teis>.
- [50] G. Melis et al. Urban heat island effect and social vulnerability in turin. *Science of the Total Environment*, 903:163300, 2023.
- [51] Bruno Milanez and Igor Ferraz Fonseca. Justiça climática e eventos climáticos extremos: uma análise da percepção social no brasil. *Revista Terceiro Incluído*, 1(2):82–100, 2011.

- [52] Biswajeeban Mishra, Biswaranjan Mishra, and Attila Kertesz. Stress-testing mqtt brokers: A comparative analysis of performance measurements. *Energies*, 14(18), 2021.
- [53] Jalal Mostafa, Sara Wehbi, Suren Chilingaryan, and Andreas Kopmann. Scits: A benchmark for time-series databases in scientific experiments and industrial internet of things. In *Proceedings of the 34th International Conference on Scientific and Statistical Database Management, SSDBM '22*, New York, NY, USA, 2022. Association for Computing Machinery.
- [54] National Institute of Standards and Technology - NIST. Nist chemistry webbook, 2026.
- [55] George L Nemhauser, Laurence A Wolsey, and Marshall L Fisher. An analysis of approximations for maximizing submodular set functions—i. *Mathematical programming*, 14(1):265–294, 1978.
- [56] OASIS. Mqtt version 5.0. <https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html>, 2019.
- [57] Gibson Oworo Ofremu, Babatunde Yusuf Raimi, Samuel Omokhaye Yusuf, Beatrice Akorfa Dziwornu, Somtochukwu Godfrey Nnabuife, Adaeze Mary Eze, and Chisom Assumpta Nnaji-for. Exploring the relationship between climate change, air pollutants and human health: Impacts, adaptation, and mitigation strategies. *Geography and Environment*, 1:100074, 2024.
- [58] OpenSSL Software Foundation. *OpenSSL Documentation: x509v3-config*, 2024. Acesso em: 12 jul. 2026.
- [59] OpenStreetMap contributors. Openstreetmap, 2026. Acesso em: 5 jul. 2026.
- [60] Carlos Pereira and Ana Santos. *Mudanças Climáticas e Eventos Extremos no Sul do Brasil*, pages 102–120. Editora UFGRS, Porto Alegre, RS, 2022. Relaciona o aumento de enchentes com mudanças climáticas e defende melhor monitoramento.
- [61] PostgreSQL Global Development Group. Postgresql documentation, 2025. Acesso em: 30 de junho de 2026.
- [62] PPM Calculation. Ppm to $\mu\text{g}/\text{m}^3$ converter, 2026.
- [63] Thomas L Saaty. The analytic hierarchy process, 1980.
- [64] João Silva and Maria Oliveira. Análise do sistema de alerta precoce em bacias hidrográficas do rio grande do sul. *Revista Brasileira de Recursos Hídricos*, 28:e45, 2023. Destaca a importância de sistemas de alerta baseados em dados para prevenir desastres.
- [65] Anita Singh and Madhoolika Agrawal. Acid rain and its ecological consequences. *Journal of environmental biology*, 29(1):15, 2007.
- [66] William Stallings. *Cryptography and Network Security: Principles and Practice*. Pearson, 8th edition, 2019.
- [67] Texas Instruments Incorporated. *ADS1118 Ultrasmall, Low-Power, SPI™-Compatible, 16-Bit Analog-to-Digital Converter with Internal Reference and Temperature Sensor*, 2019. Data Sheet SBAS457F, revisado Out. 2019.
- [68] ThingsBoard. X.509 certificate authentication. <https://thingsboard.io/docs/mqtt-broker/pe/security/authentication/x509/>. Acesso em: 30 de junho de 2026.

- [69] Xili Wan, Xinjie Guan, Tianjing Wang, Guangwei Bai, and Baek-Yong Choi. Application deployment using microservice and docker containers: Framework and optimization. *Journal of Network and Computer Applications*, 119:97–109, 2018.
- [70] Shuxia Wang, Liping Zhang, Gangsheng Wang, Dunxian She, Qin Zhang, Jun Xia, and Yanjun Zhang. More intense and longer torrential rain and flood events during the recent past decade in eurasia. *Water Resources Research*, 59(8):e2022WR033314, 2023.

A Apêndice

A.1 Responsabilidades específicas de cada integrante

- **Henrique Filho:** Pesquisa e seleção dos sensores para o projeto (Seção 2) e camada de Aquisição (Seção 3.2.1) e Comunicação ESP32 - TV Box (Seção 3.2.2).
- **Gustavo:** Seleção dos sensores ideais para o projeto (Seção 2) e camada de Aquisição (Seção 3.2.1) e Comunicação ESP32 - TV Box (Seção 3.2.2).
- **Vinicius:** Arquitetura e Fluxo de Dados do Sistema (Seção 3).
- **Gabriel:** Pontos de instalação (Seção 4), camada de dados compartilhados, análise espacial da alocação (Seção 4.3.3), dashboard.
- **Henrique Parede:** Pontos de instalação (Seção 4), camada de modelos (exceto MCLP), camada de avaliação e comparação: cobertura populacional, cobertura populacional vulnerável, distância média e Gini.
- **Mateus:** Pontos de instalação (Seção 4), MCLP, camada de avaliação e comparação: análises assintótica e de memória.

A.2 Detalhamento da estimativa do número de dispositivos

Esta subseção apresenta o cálculo utilizado para estimar a quantidade total de dispositivos necessária para o estado de São Paulo, valor apresentado na Seção 3.2.

A estimativa parte do município de São Paulo que, segundo a análise apresentada na Seção 4, demandaria 50 pontos de instalação para cobrir um valor superior a 90% da população. Considerando uma população de aproximadamente 11,9 milhões de habitantes [43], obtém-se uma razão de aproximadamente 240 mil habitantes por ponto de instalação. Para uma análise de pior caso ainda, será considerado 120 mil habitantes por ponto de estação, assim dobrando a quantidade de estações necessários.

Assumindo que essa mesma razão seja aplicável aos demais 644 municípios do estado, cuja população total é de aproximadamente 32,5 milhões de habitantes (desconsiderando a capital) [42], estima-se a necessidade de 791 dispositivos para o interior paulista. A distribuição desses municípios e a respectiva quantidade de sensores, conforme a faixa populacional, está detalhada na tabela 18.

Conforme demonstrado, a soma dos sensores para o interior totaliza 791 unidades. Somando-se esses dispositivos aos 50 previstos para o município de São Paulo, obtém-se uma estimativa total de 841 dispositivos para todo o estado de São Paulo.

Faixa populacional	Quantidade de cidades	Sensores por cidade	Total de sensores
0 – 119.999	579	1	579
120.000 – 239.999	31	2	62
240.000 – 359.999	16	3	48
360.000 – 479.999	9	4	36
480.000 – 599.999	1	5	5
600.000 – 719.999	2	6	12
720.000 – 839.999	4	7	28
1.080.000 – 1.199.999	1	10	10
1.200.000 – 1.319.999	1	11	11
Total	644	–	791

Tabela 18: Distribuição de municípios (desconsiderando a cidade de São Paulo) e sensores por faixa populacional