DISSERTAÇÃO DE MESTRADO

Sistema para Aquisição e Análise de Dados
Provenientes de Medições de Parâmetros
de Redes Celulares

Jorge Guilherme Silva dos Santos

Brasília, Fevereiro de 2019

UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA

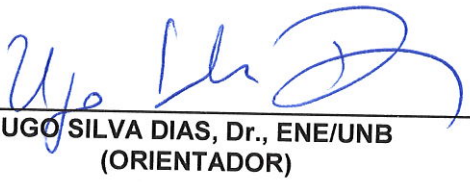
UNIVERSIDADE DE BRASÍLIA
FACULDADE DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA ELÉTRICA

SISTEMA PARA AQUISIÇÃO E ANÁLISE DE DADOS PROVENIENTES DE MEDIÇÕES DE PARÂMETROS DE REDES CELULARES

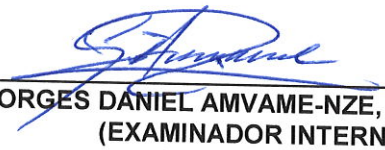
JORGE GUILHERME SILVA DOS SANTOS

DISSERTAÇÃO DE MESTRADO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE.

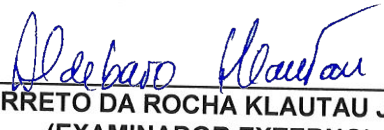
APROVADA POR:



UGO SILVA DIAS, Dr., ENE/UNB
(ORIENTADOR)



GEORGES DANIEL AMVAME-NZE, Dr., ENE/UNB
(EXAMINADOR INTERNO)



ALDEBARO BARRETO DA ROCHA KLAUTAU JÚNIOR, Dr., UFPA
(EXAMINADOR EXTERNO)

Brasília, 28 de fevereiro de 2019.

FICHA CATALOGRÁFICA

SANTOS, JORGE GUILHERME SILVA

Sistema para Aquisição e Análise de DadosProvenientes de Medições de Parâmetrosde Redes Celulares [Distrito Federal] 2019.

xvi, 70 p., 210 x 297 mm (ENE/FT/UnB, Mestre, Engenharia Elétrica, 2019).

Dissertação de Mestrado - Universidade de Brasília, Faculdade de Tecnologia.

Departamento de Engenharia Elétrica

1. Rede Celular
2. Redes
3. Sistema
4. Aplicação Móvel

I. ENE/FT/UnB

II. Título (série)

REFERÊNCIA BIBLIOGRÁFICA

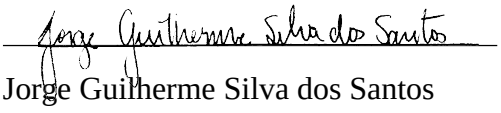
SANTOS J. G. S. (2019). *Sistema para Aquisição e Análise de DadosProvenientes de Medições de Parâmetrosde Redes Celulares*. Dissertação de Mestrado, Departamento de Engenharia Elétrica, Universidade de Brasília, Brasília, DF, 70 p.

CESSÃO DE DIREITOS

AUTOR: Jorge Guilherme Silva dos Santos

TÍTULO: Sistema para Aquisição e Análise de DadosProvenientes de Medições de Parâmetrosde Redes Celulares.

GRAU: Mestre em Engenharia Elétrica ANO: 2019



Jorge Guilherme Silva dos Santos

Depto. de Engenharia Elétrica (ENE) - FT

Universidade de Brasília (UnB)

Campus Darcy Ribeiro

CEP 70919-970 - Brasília - DF - Brasil

Agradecimentos

Ao prof. Ugo, aos colegas de projeto. Também agradeço o laboratório LATITUDE da Universidade de Brasília por fornecer a infra-estrutura para desenvolvimento e execução dos testes.

Jorge Guilherme Silva dos Santos

RESUMO

A difusão da utilização de dispositivos inteligentes e *smartphones* impõe cada vez mais importância às redes sem fio, especialmente as redes celulares. Frequentemente, usuários se mostram insatisfeitos com os serviços prestados, seja por falta de cobertura ou por desempenho insatisfatório. Contribuições surgiram nos últimos anos na forma de ferramentas de análise de redes sem fio baseadas na utilização de dispositivos dos usuários (aparelhos celulares) como forma de prover transparência e ajudar os usuários a escolher as melhores operadoras para diferentes regiões e necessidades.

Esta dissertação busca mostrar o desenvolvimento de um sistema com o objetivo de ajudar os usuários a encontrar a melhor operadora de telecomunicações em uma determinada região. Para isso, foi necessário implementar algoritmos envolvendo sistemas de coordenadas hexagonais para criar mapas de cobertura relevantes, aplicar uma técnica de agrupamento baseado na estimação de densidades para mostrar pontos críticos com potencial de baixa qualidade de serviço prestado e criar uma metodologia para classificar as operadoras com base nos dados de medição fornecidos pelos usuários.

O processamento de dados é feito de forma a facilitar a escalabilidade ao lidar com um grande volume de dados, o que permite continuação da pesquisa dentro do escopo de *BigData*. O sistema foi criado seguindo boas práticas na área de Engenharia de Software no que diz respeito à organização de sistemas em múltiplas camadas. Por fim, todos os algoritmos e técnicas são abstraídos de forma que o usuário visualize tudo facilmente na forma de um aplicativo que contém mapa de cobertura, exibição de torres com base em dados fornecidos pelas agências reguladoras e permite executar testes de taxa de transferência pontuais.

ABSTRACT

The diffusion of smart devices and *smartphones* places an increasing importance on wireless networks, especially cellular networks. Often, users are dissatisfied with the services provided, either because of coverage flaws or poor performance. Contributions have emerged in recent years in the form of wireless network analysis tools based on the utilization of users' devices (mobile devices) as a way to provide transparency and help users choose the best carriers for different regions.

This work seeks to show the development of a system aimed at helping users to find the best telecommunications operator in a given region. For this, it was necessary to implement algorithms involving hexagonal coordinate systems to create relevant coverage maps, to apply a clustering technique based on the estimation of densities to show potentially critical points with poor quality of service and to create a methodology to classify the operators based on user-supplied measurement data.

Data processing is done in a way that facilitates scalability when dealing with large amounts of data, which allows further research within the scope of *BigData*. The system was created following Software Engineering good practices with regard to the organization of systems in multiple layers. Finally, all algorithms and techniques are abstracted so that the user can access everything easily in the form of an application that contains a coverage map, display towers based on data provided by regulatory agencies, and allows to perform punctual transfer rate tests.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS E SÍMBOLOS

CDN	*Content Distribution Network*
EARFCN	*E-UTRAN Absolute RF Channel Number*
eNB	*eNodeB*
HTTP	*HyperText Transfer Protocol*
HTTPS	*HTTP Secure*
JWT	*JSON Web Token*
JSON	*Javascript Object Notation*
LTE	*Long-Term Evolution*
MNO	*Mobile Network Operator*
MVNO	*Mobile Virtual Network Operator*
PCI	*Physical Channel Identification*
RF	*Radio Frequency*
RTT	*Round Trip Time*
SPA	*Single Page Application*
TLS	*Transport Layer Security*
UARFCN	*UTRAN Absolute RF Channel Number*
UMTS	*Universal Mobile Telecommunication System*
URI	*Uniform Resource Identifier*
WLAN	*Wireless Local Area Network*

1 INTRODUÇÃO

1.1 CONTEXTUALIZAÇÃO

A importância das redes de telecomunicações no dia-a-dia das pessoas vem crescendo devido à ampla utilização de *smartphones* e o crescimento no mercado de dispositivos inteligentes que implementam funcionalidades dependentes de redes sem fios. Assim, cresce a demanda por serviços de telecomunicações de qualidade e as operadoras (MNOs - *Mobile Network Operators*) se veem diante de maior pressão por serviços melhores e maior abrangência de sua cobertura.

Atualmente, o usuário típico dos serviços das operadoras está interessado na transmissão ao vivo de vídeos (*streaming*), jogos online, comunicações multimídia em alta definição e em tempo real, além de aplicações para IoT (*Internet of Things*), formando um grande grupo de serviços que demanda disponibilidade da conexão, estabilidade, altos valores de taxas de transmissão e, em alguns casos, baixa latência.

Nesse contexto, a avaliação do desempenho e qualidade das redes celulares pode ser feita de acordo com alguns indicadores de desempenho, ou KPIs (*Key Performance Indicators*), como latência, taxas de transmissão, intensidade de sinal e até mesmo a própria demanda em termos de número de usuários [1]. Essas métricas são utilizadas pelas operadoras nos processos de otimização e planejamento de redes, assim como podem permitir proatividade na detecção de falhas nos equipamentos e de regiões com falhas de cobertura. Coletar esses dados em redes celulares pode ser feito de acordo com duas abordagens básicas:

1. Execução sob demanda de *drive-tests* usando equipamentos e mão de obra especializados;
2. Medição distribuída e possivelmente colaborativa usando dispositivos dos usuários que estão sempre conectados à Internet. Os dispositivos são responsáveis por realizar as medições, reunir os dados necessários e enviá-los aos servidores para processamento e posterior análise [2].

A primeira abordagem permite a obtenção de dados muito precisos, medidos com altos valores de taxa de amostragem. Entretanto, os equipamentos em uso para isso são caros e necessitam ser operados por mão de obra especializada, o que aumenta o custo da execução dos testes e dificulta sua execução de forma contínua. Como consequência, limita o escopo geográfico das medições.

A segunda abordagem está mais sujeita a erros de medição [1] e baixos valores de taxas de amostragem impostos por limitações no consumo de energia e duração de baterias. No entanto, possui a vantagem de ser uma alternativa de baixo custo pois utiliza a base instalada de dispositivos dos próprios usuários, o que deve aumentar o escopo geográfico das medições e permitir a obtenção de dados de forma contínua. Além disso, devido aos dispositivos utilizados serem aqueles dos usuários, as medições podem refletir diretamente a experiência dos mesmos, pois medir a partir de diferentes equipamentos, cada um com suas características a respeito de sensibilidade de recepção, antenas, gerenciamento de bateria, entre outros aspectos.

Por fim, as necessidades dos usuários parecem não ser completamente atendidas, uma vez que as operadoras de telecomunicações no Brasil lideram as estatísticas de reclamação de seus usuários, com o setor alcançando em 2017 um total de 43% de todas as reclamações de acordo com Boletim do Consumidor.gov.br [3]. Assim, há interesse dos usuários dos serviços de telecomunicação em maior transparência na abrangência e na qualidade da cobertura das operadoras disponíveis no mercado. Dessa forma, em um contexto em que todos contribuem com um sistema por meio de seus dados de medição, todos são capazes de aproveitar os dados analisados e agregados pelo sistema a fim de utilizá-los para encontrar a melhor operadora para determinada região ou condição de uso.

1.2 TRABALHOS RELACIONADOS

As pesquisas anteriores desenvolveram ferramentas de medições instaladas nos dispositivos dos usuários. Cada uma das ferramentas possui objetivos diferentes, eventualmente com alguma sobreposição entre eles. Cada uma delas foi desenvolvida e funciona de uma forma diferente, como é possível observar nas publicações referentes a cada uma delas.

Mobilyzer [4] é uma aplicação para dispositivos móveis (*smartphones*) e também uma biblioteca disponível para inclusão em outros aplicativos. Essa ferramenta foi desenvolvida com o objetivo de poder coordenar globalmente as medições por meio do monitoramento de parâmetros dos usuários e o grau de utilização das redes a cada instante. Dessa forma, não há sobreposição da execução de testes em uma mesma rede. O agendador de testes é global e os servidores usados nos testes são distribuídos ao redor do mundo em CDNs (*Content Distribution Networks*).

NetRadar [5] é uma aplicação que não está focada em redes celulares, pois está disponível para PCs e permite a execução dos testes em redes WLAN privadas. O sistema permite a execução dos testes clássicos de taxas de dados, popularmente conhecidos como "speed tests", além de coletar algumas informações técnicas como o RTT (*Round Trip Time*) entre o dispositivo e o servidor de teste assim como outras informações específicas de redes celulares como intensidade de sinal e tecnologia em uso (UMTS, EDGE, LTE, etc).

MCNet [6] é uma aplicação baseada em *crowdsourcing* capaz de coletar dados de taxas de dados e latência por meio de uma aplicação móvel que é executada em segundo plano em *smartphones* Android. Entretanto, essa ferramenta também não é focada em redes celulares, uma vez que ela exclusivamente analisa WLANs dos usuários, além de opcionalmente coletar informações sobre os APs (*Access Points*) dos mesmos. Esse tipo de obtenção de dados que depende das informações dos equipamentos nem sempre é viável em redes celulares, uma vez que as informações sobre as células são muitas vezes sensíveis e restritas ao ambiente de gerência das operadoras.

NetMap [1] é outra aplicação baseada em *crowdsourcing* que é capaz de obter medições de intensidade de sinal e RTT em redes móveis. O trabalho em questão foca na análise entre as diferenças de intensidade de sinal percebidas por vários dispositivos executando aplicativos de medições distintas como forma de validar as medições realizadas e verificar a possibilidade da utilização desses sistemas em análises dos KPIs referentes a redes celulares.

Entre os resultados apresentados, o **NetMap** mostra que há discrepâncias entre a recepção dos aparelhos, o que é esperado devido às diferenças no *hardware*, configuração das antenas e diferenças nos processos de fabricação. Entretanto, o trabalho apresenta uma comparação entre a aplicação desenvolvida e *smartphones* que executam sistemas de medição especializados que são usados pelas operadoras nos seus planejamentos e, apesar das diferenças, o valor de RMSE (*Root Mean Square Error*) obtido nas comparações ficou dentro de um intervalo de 2 a 3 dB para a tecnologia LTE, o que de acordo com o trabalho é um valor bom o bastante para obtenção de resultados consistentes, embora impossibilite a comparação direta de perda de percurso entre diferentes aparelhos.

Tanto o Mobilyzer quando o NetRadar coletam dados sobre intensidade de sinal de redes celulares. Entretanto, são focados na taxa de dados e nos testes de latência, uma vez que esses são os dados que mais impactam a experiência dos usuários. O fato dos testes serem agendados mostra o interesse na avaliação do valor máximo de taxa de dados que pode ser obtido em um determinado intervalo de tempo, o que é útil na análise de congestionamento da rede. Entretanto, nenhum desses sistemas consegue coletar taxas de transferência em *background*, o que é uma informação importante pois reflete a demanda de tráfego dos usuários ao invés de refletir diretamente a capacidade da rede.

O sistema proposto e aqui apresentado difere dos trabalhos anteriores por ser focado na experiência dos usuários finais e consumidores das operadoras de telecomunicações, além de possuir informações relevantes para auxiliar o trabalho de agências reguladoras. Não há um coordenador central de testes pelo fato de cada dispositivo funcionar de maneira independente, ou seja, sem tomar conhecimento dos demais. Assim, os dados são obtidos sempre que estão disponíveis, sem depender de um comando central. A implementação mais próxima encontrada nos trabalhos anteriores foi o NetMap, entretanto, apesar do estudo de precisão que foi apresentado, não apresentou nenhuma forma de visualização de dados focada nos usuários finais.

1.3 OBJETIVOS

A integração das redes celulares com serviços que utilizam redes IP para integração com a Internet criou a oportunidade de usar a grande base de usuários que possuem dispositivos sempre conectados para ajudar as operadoras com a implementação e planejamento de suas redes. Por esse motivo, este trabalho tem como objetivo apresentar um sistema que sirva como modelo para o desenvolvimento de aplicações móveis colaborativas que executem medições de parâmetros de redes celulares, potencialmente resultando na construção de uma base de dados centralizada de qualidade de rede.

Os usuários que contribuírem com o sistema poderão ter acesso às informações processadas na base de dados como troca pelos dados fornecidos por seus dispositivos. Isso deve criar uma forma coerente de facilitar a escolha da melhor operadora com a cobertura adequada para as regiões de interesse do usuário. No mesmo sentido, isso permite a criação de serviços de monitoramento independentes que busquem ajudar os usuários, operadoras e agências reguladores na detecção de regiões com baixa qualidade de serviço e falhas na cobertura.

1.3.1 Objetivos Específicos

Esta dissertação busca cumprir os seguintes objetivos específicos:

1. Desenvolver um sistema para medição, armazenamento e análise de dados provenientes de redes celulares, incluindo os aspectos de aquisição de dados e técnicas de agregação para visualização;
2. Propor a criação de um algoritmo para classificar as operadoras;
3. Apresentar os resultados gerados pelo sistema conforme as técnicas apresentadas;
4. Propor um formato de apresentação dos resultados focado no usuário final interessado em encontrar a melhor operadora para a região onde mora.

1.4 ESTRUTURA DO DOCUMENTO

A divisão em capítulos deste trabalho foi feita de forma a permitir um entendimento teórico prévio de todos os conceitos utilizados no desenvolvimento do sistema proposto. Assim, o Capítulo 2 trata dos conceitos gerais de redes celulares, com foco na divisão teórica da cobertura das redes em hexágonos regulares para fins de projeto e análise. Além dos conceitos básicos das células hexagonais, é apresentado o fundamento de estimação de densidades e um algoritmo de agrupamento baseado no mesmo.

A atenção é dada aos hexágonos regulares pois as técnicas e operações apresentadas e referenciadas são fundamentais para o sistema proposto, que é apresentado no Capítulo 3. Este é o capítulo que apresenta os conceitos básicos por trás do desenvolvimento de aplicações Web (em especial a arquitetura em múltiplas camadas) e o *framework* MapReduce, que é o componente central no processamento de dados e na geração dos hexágonos regulares da cobertura descritos no Capítulo 2.

As técnicas utilizadas pra cálculo dos hexágonos com MapReduce com base nos pontos de medição e a utilização desses pontos para agrupamento de pontos críticos (com baixa intensidade de sinal) geraram os resultados mostrados no Capítulo 4, que mostra todos os resultados obtidos com o desenvolvimento do sistema, incluindo a própria aplicação móvel desenvolvida, os mapas de cobertura e a metodologia criada para determinar o *ranking* das operadoras.

Finalmente, as conclusões e sugestões de trabalhos futuros com base em assuntos semelhantes encontrados na literatura são brevemente apresentados no Capítulo 5.

2 CONCEITOS DE REDES CELULARES

2.1 CONCEITOS GERAIS

Redes Celulares, como o nome sugere, são redes sem fio divididas em células para aumento de cobertura e capacidade da rede. Isso ocorre pois cada operadora em uma determinada região tem o direito de utilização de uma faixa de espectro, que é pré-determinada por alguma agência reguladora. De acordo com a tecnologia em uso, diferentes quantidades de usuários poderão ser atendidos simultaneamente dada uma determinada quantidade de frequências disponíveis.

2.1.1 Divisão em células

Por conta das limitações impostas no uso do espectro foi desenvolvido o conceito de reuso de frequência [7], que utiliza, por exemplo, hexágonos regulares para modelar a região coberta por uma célula. Assim, a divisão em células permite que as operadoras reaproveitem as frequências disponíveis em diferentes células, o que é uma forma simples de aumentar a capacidade da rede como um todo. A Seção 2.3 tratará dos conceitos matemáticos envolvidos no *grid* hexagonal como ferramenta para estimação da cobertura das operadoras.

Para cada célula existe um elemento transmissor, tipicamente um rádio e um conjunto de antenas, chamado de ERB (Estação Rádio-Base) ou BS (*Base Station*). A agência reguladora dos serviços de telecomunicações no Brasil (Anatel) mantém um banco de dados público [8] que é atualizado pelas operadoras conforme ocorre o licenciamento de novas estações. No caso, cada célula deve ser cadastrada com as informações de localização (coordenadas geográficas), frequência utilizada, altura da antena, ângulo de meia potência, potência de transmissão, entre outros. Outros países mantém bases de dados semelhantes, como a base da OFCOM [9], agência reguladora do setor de telecomunicações do Reino Unido.

A estação rádio-base é responsável, dentre outros fatores, pela comunicação, via interface aérea, entre a rede fixa da operadora e os dispositivos sem fios dos usuários. As técnicas envolvidas na comunicação, como modulação, codificação, gerenciamento de múltiplos acessos, sinalização e gerenciamento de *handoffs* variam de acordo com a tecnologia empregada. Dessa forma, cada dispositivo que se conecta à rede de uma operadora celular estará associado à uma ou mais células, usando uma determinada tecnologia e faixa de frequências, informações que deverão ser usada em conjunto com medições de outros parâmetros na rede das operadoras.

2.1.2 Fenômenos de Propagação

Ao contrário dos meios de comunicação cabeados que são estacionários e previsíveis, um meio de comunicação sem fio é aleatório. As ondas de rádio sofrem influência de diferentes fenômenos de propagação como reflexão, difração e espalhamento, fazendo com que um receptor, a cada dado instante, receba diferentes versões de um mesmo sinal transmitido, o que pode provocar variações de 30 a 40 dB na potência do sinal recebido [10].

O sinal no receptor sob influência dos mecanismos de propagação pode ser dividido em três elementos básicos [10], conforme ilustra a Figura 2.1.

- Perda de Percurso (*path loss*), referente à diminuição da potência recebida em função da distância entre o transmissor e o receptor;
- Sombreamento (*shadowing*), referente ao desvanecimento de larga escala como função dos elementos presentes no meio de transmissão tais como pessoas, edifícios, veículos e vegetação;
- Multipercurso (*multipath*), referente ao desvanecimento de pequena escala que ocorre devido às diferentes versões recebidas em tempos diferentes de um mesmo sinal que é refletido e espalhado nos obstáculos presentes no meio de propagação.

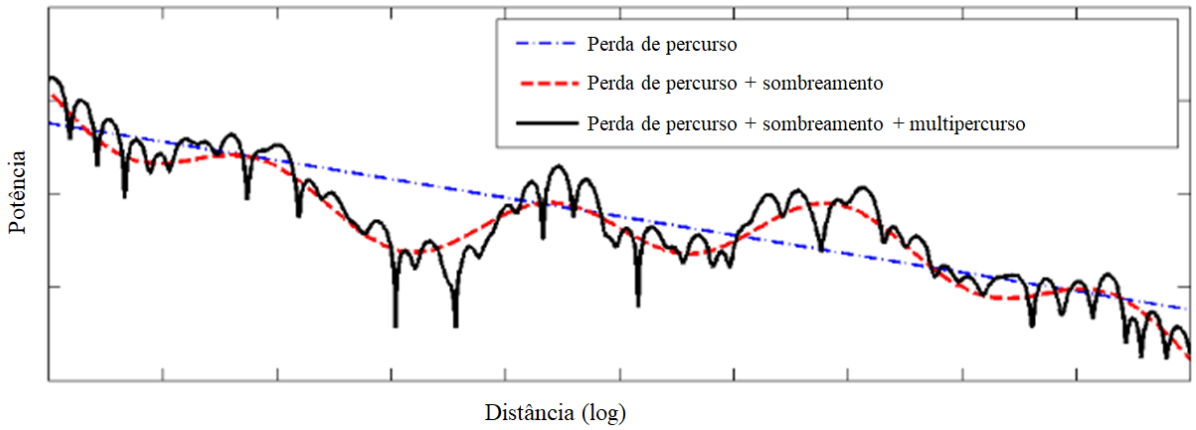


Figura 2.1: Ilustração do comportamento aleatório do sinal recebido em uma rede sem fio em função da distância entre transmissor e receptor. Adaptado de [11].

Dessa forma, não é possível determinar de forma exata qual a área de cobertura de uma determinada célula, apenas estimar a probabilidade de cobertura com base em modelos estatísticos baseados em dados empíricos, provenientes de medições nos ambientes de interesse. Assim, a utilização dos hexágonos para modelagem de células trata-se de uma aproximação, considerando um diagrama de radiação aproximadamente circular modelado como hexágono por conveniência nos cálculos. Portanto, ferramentas de medição e simulação são importantes para auxiliar a determinação correta da abrangência de cobertura real de uma célula. De fato, alguns estudos e simulações focados no desenvolvimento da quinta geração de redes celulares (5G) tratam da modelagem de redes celulares usando técnicas baseadas em geometria estocástica [12], as quais podem usar ferramentas de medição para complementar as simulações.

2.2 MEDIÇÕES EM REDES CELULARES

Dentro do contexto deste trabalho, medições em redes celulares permitem coletar dois tipos de parâmetros, o primeiro tipo refere-se à organização das redes de cada operadora e à identificação global das operadoras, o segundo tipo refere-se aos KPIs que são indicadores de desempenho e estão relacionados com o desempenho da rede. As seções a seguir tratam de uma breve descrição sobre alguns desses parâmetros.

2.2.1 Indicadores de Desempenho

Em um determinado instante de tempo, diversos dispositivos estão associados com uma célula, e cada um deles possui um valor de intensidade de sinal medida pelo aparelho, indicando a quantidade de potência que ele recebe de uma determinada célula em qualquer instante de tempo. Essa é uma das principais métricas disponíveis para análise e deve ser um dos indicadores de qualidade e abrangência da cobertura de uma operadora.

Além de intensidade de sinal, que pode ser inferida diretamente pelo hardware do dispositivo e obtida por meio do sistema operacional, há outros parâmetros úteis que podem ser inferidos indiretamente. Por exemplo, é possível acompanhar o contador de tráfego de dados durante intervalos de tempo para estimar a taxa de dados instantânea em segundo plano (*background*), medida em bits por segundo. Ou então, é possível monitorar o dispositivo do usuário para acompanhar os momentos em que ele se encontra sem conectividade com a Internet, uma métrica útil para verificar a estabilidade da conexão do aparelho em uma determinada célula.

2.2.2 Parâmetros de Organização e Identificação

Uma vez que várias células existem para cada operadora, cada uma das células possui características que podem ser medidas do ponto de vista de um dispositivo conectado e, assim, as medições podem ser identificadas como pertencentes a uma célula específica na rede de alguma operadora. Além dos parâmetros específicos de cada célula, existem diversos outros parâmetros que se relacionam com os aparelhos, com as operadoras, com as frequências disponíveis para cada operadora e com as características da rede da operadora de uma forma geral.

2.2.2.1 IMSI, MCC e MNC

Qualquer usuário conectado em uma rede celular é identificado por um código conhecido como IMSI (*International Mobile Subscriber Identity*), conforme especificado pelo 3GPP no documento *Numbering, addressing and identification* [13]. Esse número é composto por dois outros valores conhecidos como MCC (*Mobile Country Code*) e MNC (*Mobile Network Code*). A combinação desses dois número identificam de forma única uma operadora em qualquer lugar do mundo. Como exemplo, a operadora Telecom Italia (TIM) no Brasil é identificada pelo MCC 724 e MNCs 02, 03 e 04 [14].

2.2.2.2 Código identificador de tecnologia

O sistema operacional dos dispositivos identifica as tecnologias de redes celulares por um identificador numérico. Não foi encontrada especificação formal para essa numeração e, assim, cada desenvolvedora utiliza um sistema de identificação próprio. Como exemplo, dispositivos com conectividade LTE (quarta geração, ou 4G) são identificados em sistemas Android com o código 13, enquanto as tecnologias de terceira geração (3G) possuem diversos códigos diferentes associados. Esse é um valor numérico obtido pelo aparelho e entregue às aplicações pelo sistema operacional do dispositivo e sua interpretação depende da plataforma em que a aplicação é implementada. A Tabela 2.1 mostra, como exemplificação, os valores dos diferentes tipos de rede válidos para o sistema operacional Android conforme são enviados nos pontos de medição.

Tipo de Rede	**Geração**	**Código**
GPRS	2G	01
EDGE	2G	02
UMTS	3G	03
IS95 A/B	2G	04
EVDO rev. 0	3G	05
EVDO rev. A	3G	06
1xRTT	2G	07
HSDPA	3G	08
HSUPA	3G	09

Tipo de Rede	**Geração**	**Código**
HSPA	3G	10
iDEN	–	11
EVDO Rev. B	3G	12
LTE	4G	13
eHRPD	4G	14
HSPA+	3G	15
GSM	2G	16
TD_SCDMA	3G	17
IWLAN	4G	18

Tabela 2.1: Tabela de códigos de tecnologias para SO Android.

2.2.2.3 Número de canal de RF

Em uma dada tecnologia, as frequências disponíveis para utilização são identificadas por um código absoluto de canal de RF (ARFCN - *Absolute RF Channel Number*). Para tecnologia UMTS terceira geração, esse número é chamado de UARFCN (*UMTS ARFCN*) e para a tecnologia LTE de quarta geração, esse número é chamado de EARFCN (*E-UTRA ARFCN*).

A especificação do ETSI para o número EARFCN permite calcular a frequência central do canal de *downlink* em uso no instante de medição a partir do número obtido pelo dispositivo no instante da medição conforme [15]

$$f = f_0 + 0.1\,(N - N_{\text{Off}})\,, \tag{2.1}$$

em que

- f é a frequência central,
- f_0 é a frequência mais baixa do canal, proveniente da tabela fornecida pelo ETSI,
- N é o número do canal (EARFCN),
- N_{Off} é o desvio (*offset*) proveniente da tabela fornecida pelo ETSI.

Como exemplo, em um determinado instante se um dispositivo realizou uma medição que retornou o valor 1276 para o código EARFCN, a consulta à tabela da ETSI indica que o valor de f_0 correspondente é 1805 MHz e o valor de N_{Off} correspondente é 1200. Dessa forma, a equação 2.1 retorna o valor de frequência central do canal de *downlink* é 1812,6 MHz.

Por sua vez, a especificação para o número UARFCN permite calcular a frequência central do canal de *downlink* para tecnologia UMTS (terceira geração) conforme [16]

$$f = 0.2N + N_{\text{Off}}, \tag{2.2}$$

em que

- f é a frequência central,
- N é o número do canal (UARFCN),
- N_{Off} é o desvio (*offset*) proveniente da tabela fornecida pelo ETSI.

De fato, é necessário determinar todos os valores de UARFCN para a tabela do ETSI, uma vez que os valores não são fornecidos. Assim é possível encontrar o valor de N_{Off} correspondente e calcular a frequência central do canal. Por fim, para determinar o modo de operação (FDD ou TDD), é necessário consultar as tabelas do ETSI.

Novamente, como exemplo, em um determinado instante se um dispositivo realizou uma medição que retornou o valor de UARFCN 4438, a consulta à tabela (após calcular todos os valores possíveis nos intervalos) indica que o valor de N_{Off} é 0 e, portanto, a frequência central de *downlink* é 887,6 MHz.

Assim, com base nos números de canal RF medidos, é possível encontrar a frequência de operação durante uma medição realizada por um dispositivo. As informações de desempenho associadas com a frequência de utilização são úteis nas estimativas de propagação, que é influenciada diretamente pela frequência de operação.

2.2.2.4 Identificação da Célula

O código PCI (*Physical Cell Identity*) é um número utilizado para identificar unicamente uma célula em uma determinada região. Existem 504 valores únicos disponíveis, o que inevitavelmente irá levar ao reuso dos valores. Uma vez que esse valor é único para uma determinada região, a presença de valores iguais para células vizinhas ou que estejam cobrindo uma mesma área de cobertura indica um possível problema de configuração [17]. Dessa forma, a obtenção desse parâmetro durante medições pode ser usada para encontrar falhas na configuração dos equipamentos de uma determinada operadora.

2.3 GRID DE COORDENADAS HEXAGONAIS

Um sistema que coleta informações geo-referenciadas identifica cada ponto de medição no espaço por meio de um par de coordenadas em um sistema de coordenadas conveniente para a aplicação em uso. Tipicamente, devido à popularidade das aplicações comerciais de mapas e GPS, é utilizado o sistema de coordenadas cartesianas, que identifica pontos no globo por meio do par (longitude, latitude). Dessa forma, cada ponto de medição pode possuir, junto com os dados referentes às redes celulares, a informação de localização associada.

2.3.1 Mapeamento de Pontos em Hexágonos

Para fins de processamento e análise de dados, nem sempre é possível utilizar todos os pontos de medição existentes de uma vez. Frequentemente, é preciso agregar as informações coletadas por meio do agrupamento espacial dos pontos. Assim, análises posteriores podem ser feitas usando os resultados agregados que são menores em quantidade e apresentam informações resumidas. Uma forma de conveniente de realizar essa agregação de dados é aproveitar as propriedades relevantes dos hexágonos regulares, já utilizados no projeto de redes celulares, para agrupar os pontos de medição e apresentar visualmente as informações.

Para alcançar esse objetivo, é necessário utilizar alguma técnica para mapear os pontos de medição em hexágonos regulares. Há uma técnica proveniente da área de processamento de imagens usando pixeis hexagonais que permite realizar tal mapeamento por meio do grid de coordenadas hexagonais [18], também explorado indiretamente nos estudos envolvendo a divisão de redes sem fio em células [7], [10], conforme mostra a Figura 2.2.

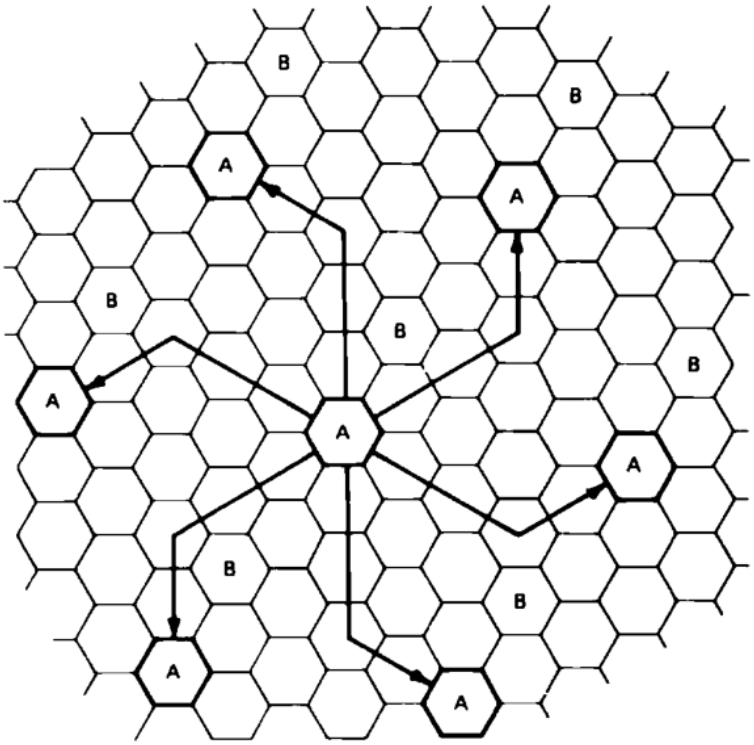


Figura 2.2: Ilustração do grid de coordenadas hexagonais do ponto de vista de reuso de frequências. Adaptado [7].

A Figura 2.2 mostra um grid de coordenadas hexagonais do ponto de vista do projeto de redes celulares com foco no reuso de frequências. Cada hexágono percorrido é identificado por um par de coordenadas (i, j), cada qual é usada para identificar o caminho percorrido em uma direção, com a segunda direção (j) rotacionada em 60 graus a partir da primeira (i). Também é importante observar que os hexágonos são identificados por valores inteiros nessas direções, o que pode ser interpretado como: "as coordenadas dos centros dos hexágonos são valores inteiros nas direções (i, j)".

A ideia básica utilizada no projeto de redes celulares pode ser adaptada para obtenção de hexágonos de tamanho arbitrário dado um ponto qualquer no espaço. Por meio da utilização das duas direções descritas na Figura 2.2 como bases de um novo sistema de coordenadas, chamado de gabarito hexagonal simétrico (*symmetrical hexagonal coordinate frame*) ${}^*R^3$ [18], ilustrado na Figura 2.3.

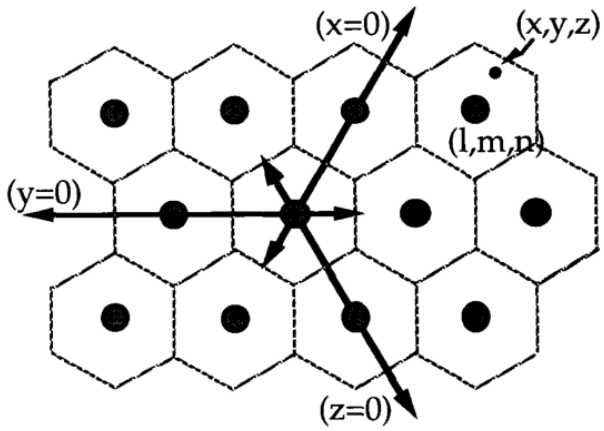


Figura 2.3: Ilustração do quadro hexagonal simétrico ${}^*R^3$. Adaptado [18].

O quadro hexagonal simétrico é utilizado por conta de um algoritmo já existente que permite o arredondamento das coordenadas, o que permite que as coordenadas dos centros dos hexágonos sejam facilmente encontradas. Assim, para mapear um ponto no espaço para as coordenadas do centro do hexágono correspondente é necessário seguir as etapas:

1. Conversão de coordenadas decimais para o quadro hexagonal simétrico (${}^*R^3$);
2. Arredondamento conforme [18];
3. Conversão contrária: de ${}^*R^3$ para coordenadas decimais.

A conversão de coordenadas é feita em duas etapas: primeiro, é feita a conversão para coordenadas oblíquas utilizando a transformação linear

$$\begin{bmatrix} q \\ r \end{bmatrix} = \begin{bmatrix} 2/3 & 0 \\ -1/3 & \sqrt{3}/3 \end{bmatrix} \begin{bmatrix} \lambda \\ \varphi \end{bmatrix} \begin{pmatrix} 1 \\ \overline{s} \end{pmatrix}, \tag{2.3}$$

em que s é o tamanho do raio (ou lado) do hexágono regular, λ é o valor da longitude, φ é o valor de latitude e (q, r) são as coordenadas no eixo de coordenadas oblíquas. Após essa conversão, as coordenadas hexagonais são obtidas com

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \\ -1 & -1 \end{bmatrix} \begin{bmatrix} q \\ r \end{bmatrix}, \tag{2.4}$$

em que z' depende das outras duas coordenadas, fazendo com que a base (x', y', z') não seja uma base linearmente independente, conforme mostra a Figura 2.3. As coordenadas (x', y', z') são então arredondadas usando o algoritmo de arredondamento no quadro hexagonal simétrico [18]. Uma vez que o arredondamento é feito, basta utilizar as inversas das transformações mostradas nas equações 2.3 e 2.4.

Com isso, é preciso criar uma forma de visualização de dados formada por vários hexágonos regulares sobrepostos sobre um mapa utilizando coordenadas decimais. Isso permite utilizar os hexágonos para formar um mapa de cobertura, cujo nível de detalhamento e precisão varia com o tamanho escolhido para os mesmos.

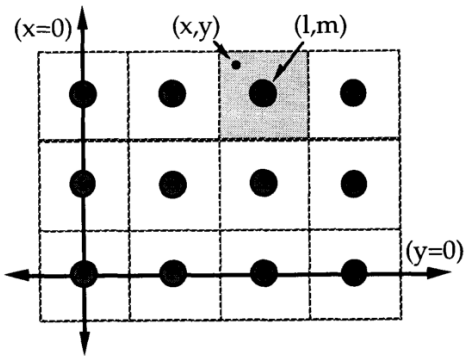
2.3.2 Arredondamento em coordenadas hexagonais

O algoritmo de arredondamento proposto em [18] utiliza a definição de um *domínio de atração*. Um domínio de atração de um ponto é uma região no sistema de coordenadas sobre dentro da qual quaisquer pontos são arredondados para as coordenadas de seu centro, conforme mostra a região sombreada na Figura 2.4a. Para regiões retangulares, a área de atração do ponto representado por (l, m) é o conjunto de todos os pontos (x, y) que respeitam a relação

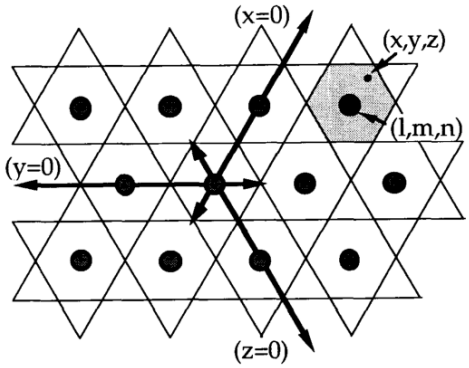
$$\begin{aligned} |x - l| &\leq 0.5, \\ |y - m| &\leq 0.5. \end{aligned} \tag{2.5}$$

É possível observar na equação 2.5 que o ponto será arredondado para as coordenadas (l, m) se a distância entre cada uma de suas coordenadas e as respectivas coordenadas do ponto central forem menores que 0.5. De fato, o arredondamento de qualquer número consiste em escolher o número que está menos de 0.5 unidades distante do mesmo, e isso é naturalmente estendido para as coordenadas retangulares analisando as componentes do vetor individualmente.

Entretanto, no gabarito hexagonal simétrico, a definição desse domínio de atração não é clara, e o arredondamento não pode ser feito componente a componente uma vez que os domínios de atração possuem formato retangular. Conforme mostra a Figura 2.4b, os domínios de atração são hexágonos como desejado, contudo, existem regiões triangulares não cobertas pelos domínios de atração.



(a) Ilustração do domínio de atração em coordenadas retangulares [18].



(b) Ilustração do domínio de atração em no gabarito simétrico de coordenadas hexagonais [18].

Figura 2.4: Ilustração dos domínios de atração.

O algoritmo proposto em [18] para resolver o arredondamento em coordenadas hexagonais é descrito na Figura 2.5.

2.3.3 Distorção dos hexágonos em diferentes latitudes

A projeção utilizada nas aplicações comerciais de mapas precisa representar o formato aproximadamente esférico da Terra em um plano retangular, o que causa distorções para os elementos que estão mais próximos dos polos. A cobertura de hexágonos gerada para essas regiões deve seguir a mesma distorção, para manter coerência com a visualização dos locais no mapa. Além disso, também há redução do raio da circunferência conforme os pontos se aproximam dos polos, causando a diminuição da distâncias (em metros) entre as linhas de longitude.

```
hex_round(x,y,z):
  l = round(x)
  m = round(y)
  n = round(z)
  sum = l+m+n
  if sum == 0:
    return
  if sum > 0:
    xr = x-l
    yr = y-m
    zr = z-n
  else:
    xr = l-x
    yr = m-y
    zr = n-z

  smallest = min(xr,yr,zr)
  if smallest == xr:
    l = -(m+n)
  elif smallest == yr:
    m = -(l+n)
  else:
    n = -(l+m)

  return (l,m,n)
```

Figura 2.5: Algoritmo de arredondamento em ${}^*R^3$ [18].

A coerência na visualização é alcançada simplesmente por meio da utilização das coordenadas decimais (latitude e longitude) conforme especificação EPSG:4326 [19], inclusive para a especificação do tamanho do hexágono na equação 2.3. Ao utilizar as coordenadas decimais no lugar de distâncias (em metros, por exemplo), os hexágonos poderão ser encaixados naturalmente sobre os mapas, com a desvantagem de não ser possível comparar os tamanhos (em metros) para latitudes muito distintas.

A distância entre dois pontos no mapa pode ser determinada considerando-os como dois pontos de uma esfera, se a Terra for aproximada como uma esfera perfeita de raio igual a 6371 quilômetros. A diferença entre a distância provocada pela proximidade com os polos pode ser observada aplicando, para diferentes pares de coordenadas, a fórmula haversine

$$d = 2r \arcsin\left(\sqrt{\text{hav}\left(\frac{\varphi_2 - \varphi_1}{2}\right) + \cos(\varphi_1)\cos(\varphi_2)\,\text{hav}\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right), \tag{2.6}$$

em que r é o raio médio da terra, $\text{hav}(\theta) = \sin^2(\theta/2)$ é a função haversine, φ_1 e λ_1 são as coordenadas (longitude e latitude, respectivamente) do primeiro ponto e φ_2 e λ_2 são as coordenadas do segundo ponto, todas em radianos.

Para fins de ilustração, ao aplicar a fórmula acima para os pares de coordenadas $(\lambda_1, \varphi_1) = (0, 0)$ e $(\lambda_2, \varphi_2) = (0.1\pi/180, 0)$, representando uma distância no sentido leste-oeste de 0,1 graus decimais, é obtida a distância aproximada de 11,12 km. Para verificar a diferença provocada pela altitude, a mesma distância em graus decimais pode ser obtida utilizando os pares de coordenadas $(\lambda_1, \varphi_1) = (0, 60\pi/180)$ e $(\lambda_2, \varphi_2) = (0.1\pi/180, 60\pi/180)$, o que resulta, pela fórmula, em uma distância aproximada de 5,56 km. Essa relação também pode ser visualizada graficamente na Figura 2.6. Assim, é importante destacar o cuidado que é necessário ao se comparar os hexágonos calculados para os pontos de medição em lugares distantes.

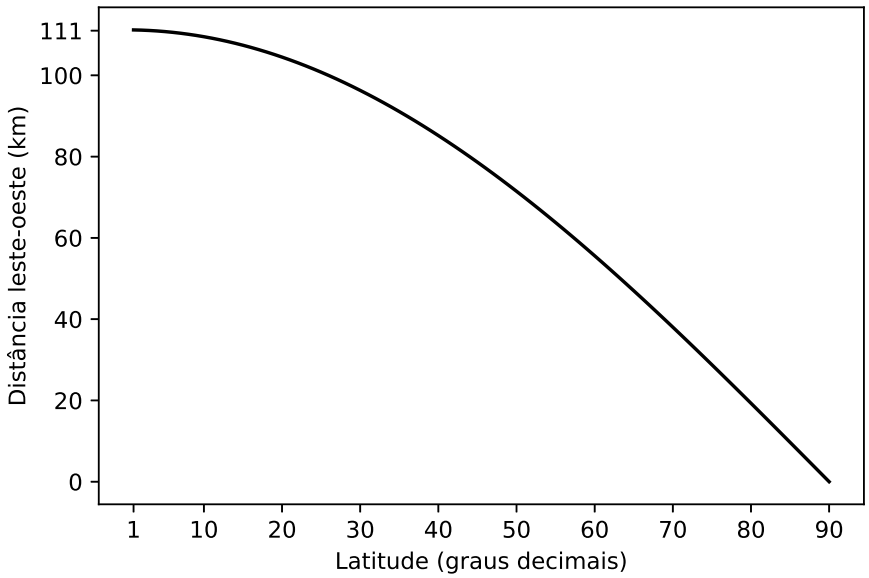


Figura 2.6: Distância haversine correspondente para 1 grau decimal em função da latitude.

2.4 ESTIMATIVA DE DENSIDADES

A estimativa de densidades abrange o conjunto de técnicas estatísticas (recentemente utilizadas dentro do contexto de aprendizado de máquina) que buscam encontrar as densidades de probabilidade da ocorrência de eventos de interesse que possuem natureza de ocorrência aleatória. Assume-se que há uma amostra $\mathcal{X} = \{\mathbf{x}^t\}_{t=1}^N$, em que cada x^t é uma amostra (ou observação) d-dimensional entre as N observações do conjunto de dados (*dataset*) $\mathcal{X}$ que são amostradas independentemente de uma densidade de probabilidade desconhecida $p(\cdot)$. As técnicas de estimativa de densidade tem como objetivo encontrar o melhor estimador da densidade de probabilidade $\hat{p}(\cdot)$. A Figura 2.7 ilustra um exemplo de um conjunto de dados em uma dimensão amostrado aleatoriamente e sua respectiva curva de densidade de probabilidade estimada.

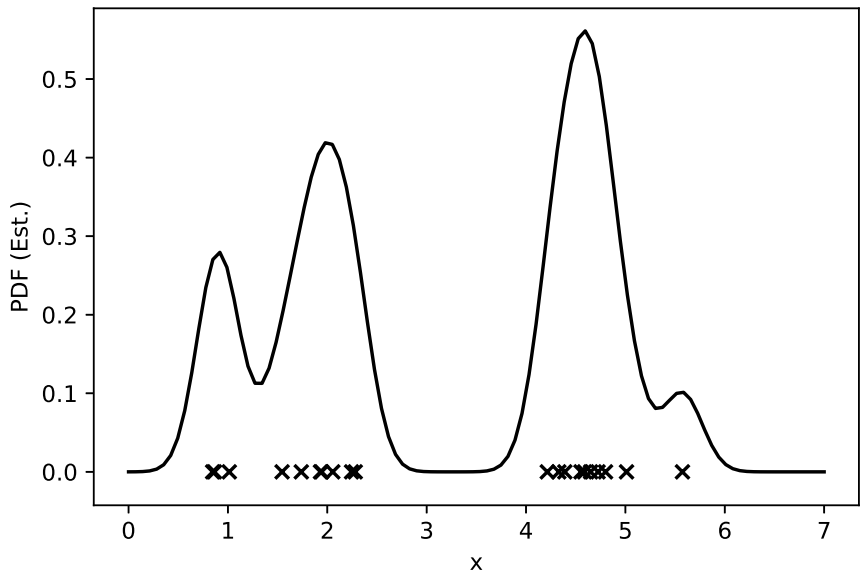


Figura 2.7: Exemplo de estimativa de densidade em uma dimensão. Kernel: Gaussiano com $h = 0.2$.

Em geral, os métodos para encontrar as densidades de probabilidade podem ser divididos em dois grandes tipos [20]:

- **Paramétrico**, em que é assumido previamente que os dados obedecem à alguma função distribuição de probabilidade, o que faz com que os métodos possuam como objetivo encontrar os parâmetros do modelo assumido. Nesse caso, normalmente é possível encontrar uma solução analítica prévia para os parâmetros a partir da minimização de alguma função de erro com base em operações algébricas aplicadas ao conjunto de dados $\mathcal{X}$. Para essa abordagem, imputa-se um modelo prévio para $\hat{p}(\cdot)$ que acredita-se ser o modelo que dá origem à aleatoriedade dos dados (como uma densidade gaussiana, por exemplo), o que apresenta a desvantagem de enviesar o modelo.

- **Não-Paramétrico**, em que não há formato previamente assumido para a distribuição dos dados, utilizando quando não se pode fazer nenhuma estimativa prévia a respeito dos mesmos. Para esse caso, normalmente há alguma manipulação matemática associada a algum método iterativo, mas não há solução algébrica fechada para os parâmetros de interesse. Por não possuir parâmetros prévios provenientes de um modelo previamente imputado, não há enviesamento, o que permite que o modelo treinado seja capaz de assumir qualquer forma, útil para dados que são muito complexos, apesar da desvantagem de aumentar a complexidade computacional e aumentar os tempos de processamento. É especialmente útil para os casos em que não se tem informações prévias sobre a densidade de probabilidade dos dados.

2.4.1 Estimador Kernel

O estimador kernel utiliza uma função kernel $K(\cdot)$ que determina o formato da influência de cada instância dentro de uma janela de influência de tamanho h. Dessa forma, o estimador da densidade é definido por [20]

$$\hat{p}(\mathbf{x}) = \frac{1}{Nh^d} \sum_{t=1}^{N} K\left(\frac{\mathbf{x} - \mathbf{x}^t}{h}\right). \tag{2.7}$$

Observa-se na Equação 2.7 que cada instância $\mathbf{x}^t$ possui uma região de influência de tamanho h que diminui conforme a distância entre $\mathbf{x}$ e $\mathbf{x^t}$ aumenta. Diversas funções kernel podem ser utilizadas, de acordo com o formato desejado para a influência de cada amostra do *dataset*. É possível, por exemplo, fazer a influência de cada amostra decair exponencialmente com a distância ao se utilizar um kernel gaussiano, ou ainda fazer a influencia de cada amostra ser constante dentro de um raio, o que é alcançado ao se utilizar um kernel plano (*flat kernel*). Em uma situação em que muitas amostras estão concentradas, há maior superposição das regiões de influência, o que provoca um aumento da densidade na região, conforme ilustra a Figura 2.7 para o caso unidimensional.

Para dados geo-referenciados, as amostras podem ser bi-dimensionais, com cada *feature* representando um elemento do par de coordenadas decimais que localizam o ponto no mapa. Dessa forma, é possível realizar uma estimativa de densidade que identifique a maior concentração da ocorrência de um determinado fenômeno no mapa. Calcular e representar os pontos mais influentes de uma densidade de probabilidade cabe a um algoritmo de busca dos máximos locais da função estimada.

Para que uma função kernel possa ser utilizada, ela deve satisfazer a condição

$$\int_{\Re^d} K(\mathbf{x}) d\mathbf{x} = 1.$$

Entre as diversas possibilidades, está o kernel gaussiano, cuja generalização para o caso d-dimensional é

$$K(\mathbf{x}) = \left(\frac{1}{\sqrt{2\pi}}\right)^d \exp\left[-\frac{||\mathbf{x}||^2}{2}\right], \tag{2.8}$$

entretanto, seu uso não é recomendado para espaços em dimensões muito altas por conta do emprego da distância euclidiana ($||\cdot||$), que não é clara ou intuitiva em dimensões altas. Para evitar a chamada maldição da dimensionalidade, é possível utilizar um kernel gaussiano que leva em consideração a correlação entre as amostras e a distância de Mahalanobis

$$K(\mathbf{x}) = \frac{1}{(2\pi)^{d/2}|\mathbf{S}|^{1/2}} \exp\left[-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{S}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right], \tag{2.9}$$

em que d é a dimensão dos dados (número de *features*), $\mathbf{S}$ e $\boldsymbol{\mu}$ são a matriz de covariância e o vetor de médias do *dataset* $\mathcal{X}$, respectivamente e $(\mathbf{x}-\boldsymbol{\mu})^T\mathbf{S}^{-1}(\mathbf{x}-\boldsymbol{\mu})$ é a distância de Mahalanobis.

Um kernel mais simples é o kernel plano (*flat kernel*), que foi o kernel adotado neste trabalho. Apesar de empregar a distância euclidiana, os problemas apresentados no trabalho são bi-dimensionais, o que dispensa a preocupação com as distâncias em dimensões altas e, portanto, dispensa o uso da distância de Mahalanobis. Esse kernel é unitário dentro de uma esfera de raio h e assim, é definido

$$K(\mathbf{x}) = \begin{cases} 1 & \text{se } ||\mathbf{x}|| \leq h \\ 0 & \text{se } ||\mathbf{x}|| > h \end{cases} \tag{2.10}$$

2.4.2 Algoritmo MeanShift

Dado um conjunto de dados $\mathcal{X}$, haverá uma estimativa de densidade de probabilidade associada que possuirá pontos de máximo locais. O algoritmo *MeanShift* é um algoritmo iterativo utilizado para encontrar grupos (*clusters*) de pontos por meio desses pontos de máximo, sem estimar a densidade completa [21].

O algoritmo se baseia na estimativa do gradiente da densidade estimada na equação 2.7. Em conjunto com a determinação de um segundo kernel $G(\mathbf{x}) = -K'(\mathbf{x})$, em que $K(\cdot)$ é chamado de sombra de $G(\cdot)$ e assumindo que a derivada $K'(\mathbf{x})$ existe, o cálculo do gradiente resulta na criação do vetor *mean shift* definido por

$$\mathbf{m}_{h,G}(\mathbf{x}) = \frac{\sum_{i=1}^{N} G\left(\frac{\mathbf{x}-\mathbf{x_i}}{h}\right)\mathbf{x}_i}{\sum_{i=1}^{N} G\left(\frac{\mathbf{x}-\mathbf{x_i}}{h}\right)}, \tag{2.11}$$

que é um vetor que aponta na direção em que a densidade de probabilidade é mais alta, ou, de maneira equivalente, em que há maior aumento na concentração dos pontos, ou ainda em que o gradiente da densidade é o maior possível. Nesse sentido, pode-se dizer que o algoritmo é um método de gradiente ascendente.

Para empregar o *flat kernel* definido na equação 2.10, basta considerá-lo como o kernel $G(\mathbf{x})$. Isso pode ser observado na equação 2.11 que define o vetor *mean shift*: o vetor $\mathbf{m}_{h,G}(\mathbf{x})$ é especificado apenas em função de $G(\cdot)$.

O funcionamento do algoritmo pode ser resumido brevemente em:

1. Um candidato à centroide de *cluster* é escolhido. Esse é o valor **x** na equação 2.11;
2. A partir desse candidato, o vetor *mean shift* $\mathbf{m}_{h,G}(\mathbf{x})$ é calculado conforme a equação 2.11;
3. O valor de $\mathbf{m}_{h,G}(\mathbf{x})$ é somado a **x**;
4. O processo é repetido até a convergência do algoritmo, em que o valor **x** resultante encontrado é o centroide do *cluster* de interesse.

Para encontrar todos os centroides de *cluster*, o algoritmo descrito precisa ser executado para vários candidatos. A escolha dos mesmos depende de implementação: podem ser escolhidos candidatos próximos às instâncias de $\mathcal{X}$ ou cada uma das instâncias pode ser escolhida como candidato, como ocorre em [22]. Ao final do processo deve haver um processo de poda (*pruning*) para remover possíveis centroides redundantes.

2.5 CONCLUSÃO

Este capítulo apresentou conceitos básicos necessários para o entendimento das técnicas aplicadas no sistema. O grid de coordenadas hexagonais será utilizado para criar um mapa de cobertura e o algoritmo de arredondamento é utilizado para criar um hexágono regular com base em qualquer ponto em um espaço de coordenadas arbitrário.

O próximo capítulo tratará dos conceitos básicos envolvidos no desenvolvimento de um sistema Web e do processamento de dados em larga escala utilizando o *framework* MapReduce, que é utilizado para executar o algoritmo de arredondamento. Com a combinação dos conceitos apresentados nos dois capítulos será possível entender como o sistema proposto faz o processamento e a apresentação dos dados de medição provenientes dos dispositivos dos usuários.

A estimação de densidades pode ser aplicada a várias métricas. Especificamente, os próximos capítulos apresentarão a estimação da densidade dos pontos com baixa intensidade de sinal em que as componentes dos vetores processados com o MeanShift serão as coordenadas decimais referentes à localização dos pontos no planeta.

3 SISTEMA PROPOSTO

3.1 CONCEITOS BÁSICOS DE APLICAÇÕES WEB

Com a popularização da Internet no início dos anos 2000, a Web se tornou a principal plataforma para interação entre pessoas, seja para troca de informações, lazer, ou outras demandas como compras e operações bancárias. Essa demanda criou diversas oportunidades de negócios que levaram ao surgimento de produtos pensados para a Web, ou seja, que interagiam com os usuários por meio dos navegadores de seus computadores conectados à Internet. As aplicações Web, como são chamadas, são plataformas ou sistemas que processam e armazenam dados em servidores remotos, enquanto o computador do usuário era responsável pela visualização dos dados e preenchimento de formulários.

3.1.1 HTTP e REST

A Web e todas as suas aplicações e sistemas utilizam o popular protocolo HTTP [23] para transferência de dados na Internet. Esse protocolo se caracteriza pela utilização da arquitetura cliente-servidor, em que os clientes (originalmente os navegadores instalados nos computadores dos usuários) são responsáveis por solicitar os recursos fornecidos pelos servidores que, por sua vez, respondem às solicitações dos clientes. As páginas Web eram os primeiros recursos disponíveis na Web, e o protocolo HTTP era utilizado para baixar as páginas que eram visualizadas por meio dos navegadores ou, no máximo, enviar formulários e arquivos para armazenamento remoto.

Os sistemas existentes na Internet hoje não se resumem simplesmente a envio de dados e visualização de páginas. Os sistemas são complexos, permitindo que os usuários interajam de forma dinâmica com os mesmos, editem configurações e personalizem a sua experiência. Além disso, a recente popularização dos *smartphones* e outros dispositivos inteligentes trouxe a difusão dos aplicativos que, instalados nos dispositivos dos usuários, continuam a utilizar o protocolo HTTP, contudo, com objetivos distintos do que os que estavam inicialmente previstos na especificação do protocolo.

O REST [24] surgiu como um tipo de arquitetura para sistemas distribuídos que busca padronizar a forma como ocorre a transferência de dados entre os elementos que constituem um sistema distribuído na Web. Entre os diversos princípios propostos nessa arquitetura, estão a obrigatoriedade da comunicação seguir o modelo *cliente-servidor*, do servidor não guardar estado dos seus clientes, ou seja, os dados trocados devem conter todas as informações necessárias para seu processamento, e a padronização da interface de comunicação, o que desacopla os serviços fornecidos da sua implementação, permitindo que diferentes entidades possam se comunicar desde que sigam a mesma interface padronizada pelos projetistas do sistema.

O REST possui grande importância no cenário atual da Internet, devido à difusão dos *smartphones* e outros dispositivos inteligentes. Apesar de não ser o único tipo de arquitetura possível, muitos sistemas são construídos seguindo as especificações do REST, o que ajuda o desenvolvimento de aplicações Web capazes de atender diversos dispositivos diferentes, além dos tradicionais navegadores nos computadores dos usuários. Assim, possui fundamental importância no funcionamento de muitos aplicativos para *smartphones* existentes no mercado.

3.1.2 Tipos de aplicações Web

Nos primórdios da Web, as páginas eram estáticas e os navegadores recebiam seus conteúdos em HTML para renderização e visualização pelos usuários. Para criar sistemas interativos e personalizar a experiência dos usuários, surgiram as páginas dinâmicas, que mudam o seu conteúdo de acordo com o estado do sistema e do usuário. Isso permitiu o surgimento de aplicações com formulários dinâmicos, sistemas corporativos de gestão de pessoas, sistemas de gestão financeira, etc, todos utilizados a partir de navegadores por meio da Internet.

Tradicionalmente, as páginas dinâmicas são montadas nos servidores Web e retornadas aos navegadores para visualização. O emprego de modelos arquiteturais como o REST e a necessidade de criações de aplicações Web rápidas que se comportem como aplicações nativas levou ao surgimento de *single page applications* (SPAs), ou “aplicações de uma página”, em tradução livre. Essas aplicações caracterizam-se por baixar todo o conteúdo e código necessário para a estrutura da página no primeiro carregamento, com os conteúdos baixados e renderizados dinamicamente conforma e utilização. Para os usuários, as páginas carregam mais rápido e a experiência de uso se torna semelhante à experiência percebida com aplicações instaladas localmente em seus computadores.

Devido à natureza do funcionamento das SPAs, os servidores fornecem interfaces para troca de dados por meio do protocolo HTTP, usadas pelos navegadores para obtenção e visualização dinâmica dos conteúdos. Essas interfaces podem ser padronizadas conforme o modelo arquitetural REST, mencionado na seção anterior, permitindo que aplicativos instalados em *smartphones* de usuários (e outros dispositivos inteligentes) sejam capazes de utilizar as mesmas interfaces, o que diminui o trabalho envolvido no desenvolvimento dos sistemas.

Dessa forma, os servidores encarregados de processamento e armazenamento de dados são capazes de atender, usando as mesmas tecnologias, tanto usuários de navegadores e SPAs quanto usuários de *smartphones* utilizando aplicativos. Portanto, esse modelo de arquitetura é importante para a construções de aplicações Web atualmente, e justifica o crescimento e a difusão de SPAs e aplicativos em dispositivos móveis.

3.1.3 Aplicações em Múltiplas Camadas

Para simplificar o desenvolvimento de sistemas, é comum que desenvolvedores dividam seus projetos em camadas. Essas camadas podem ser fisicamente separadas, como no caso de um *software* no cliente e um *software* no servidor, ou podem ser camadas em execução em um mesmo computador. No que diz respeito a aplicações Web, é comum fazer a divisão em três camadas [25] conforme ilustra a Figura 3.1:

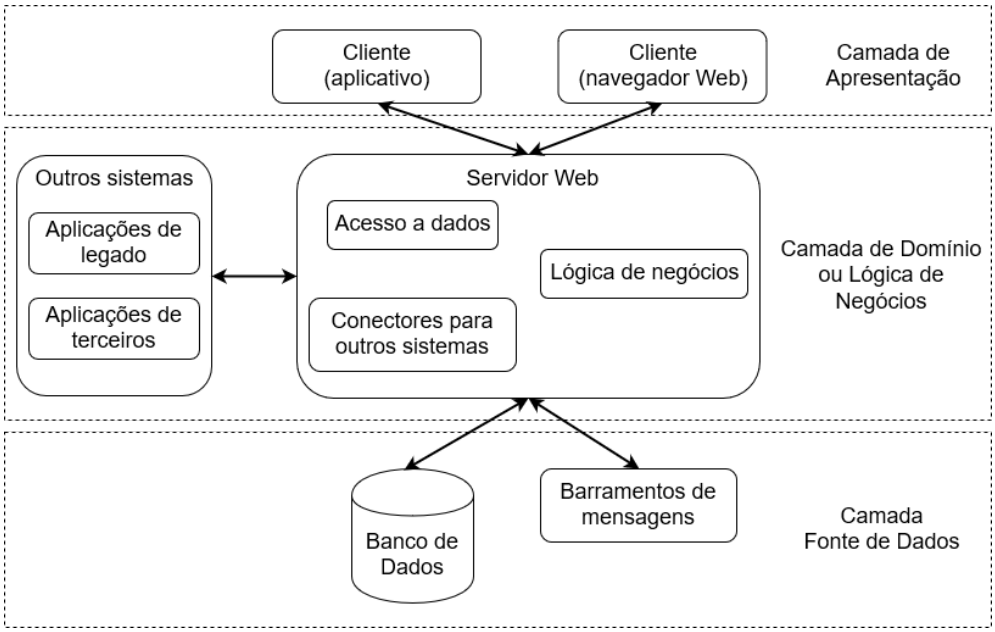


Figura 3.1: Aplicações Web em Múltiplas Camadas (Arquitetura *Multitier*).

- **Apresentação**: responsável pela interação entre o *software* e o usuário. No caso de aplicações Web, é normalmente uma página no navegador do usuário que traz a execução de alguns *scripts* necessários à montagem e manipulação da interface do usuário.

- **Domínio ou Lógica de Negócios**: Envolve todos os cálculos e procedimentos específicos ao domínio de negócios ao qual o sistema pertence. Isso inclui a validação dos dados recebidos dos clientes (usuários), o processamento dos dados para exibição e quais sistemas da camada de dados utilizar de acordo com as ações do usuário.

- **Fonte de Dados**: responsável pela comunicação com outros sistemas como bancos de dados, barramentos de mensagens, outros sistemas. Normalmente o elemento mais importante dessa camada é um banco de dados, responsável por persistir grandes quantidades de dados.

Como é comum em sistemas divididos em camadas, cada uma das camadas possui seu conjunto de responsabilidades e fornece um serviço para a camada subsequente. Isso permite que, desde que definidas interfaces padronizadas entre elas, as camadas possam ser modificadas individualmente com pouca interferência no sistema como um todo. É possível modificar o sistema gerenciador de banco de dados na camada de fonte de dados sem grandes modificações na camada de domínio e, mais ainda, idealmente essas modificações serão transparentes para os usuários interagindo com a camada de apresentação.

Além disso, a divisão em camadas facilita a documentação do sistema e os estudos envolvendo os elementos de cada camada, além de permitir maior flexibilidade no lançamento de atualizações. Uma atualização na camada de domínio referente à lógica de negócios não exige necessariamente que todos os clientes sejam atualizados ou ainda que as versões do sistema nas máquinas dos clientes sejam sincronizadas para o correto funcionamento do sistema. A desvantagem dessa abordagem são os tempos de resposta mais elevados em comparação com o que seria obtido caso a lógica de negócios fosse executada localmente

A arquitetura REST apresentada na sessão anterior pode ser utilizada nesse contexto como a arquitetura utilizada para comunicação entre as camadas de apresentação e domínio. Dessa forma, é possível montar múltiplas camadas de apresentação, como uma página Web e um aplicativo em um *smartphone*, por exemplo, sem modificar as demais camadas, que são executadas normalmente em servidores remotos na Internet.

3.2 ARQUITETURA DO SISTEMA PROPOSTO

Para atender aos objetivos apresentados na Seção 1.3, foi desenvolvido um sistema conforme a arquitetura de aplicações em múltiplas camadas que será apresentada nesse capítulo. Trata-se de uma aplicação Web com o objetivo de fornecer uma API REST utilizada pela aplicação instalada nos dispositivos dos usuários (inicialmente *smartphones* Android). Essa API tem objetivos que podem ser classificados em dois grandes grupos:

- Permitir o envio de informações sobre medições dos usuários a partir dos seus *smartphones* aos servidores remotos para processamento;
- Permitir aos usuários finais a visualização de dados processados e agregados pelos servidores;
- Permitir aos administradores do sistema e pesquisadores a visualização de dados específicos que não estão disponíveis para os usuários finais.

A Figura 3.2 apresenta todos os componentes existentes e sua organização nas camadas propostas. Os elementos presentes em *Scripts* e *Servidor Web (app)* são todos de autoria própria e foram divididos de acordo com a sua importância no sistema e complexidade de implementação. Os scripts são executados de tempos em tempos e realizam operações em lotes enquanto a aplicação em si (no servidor Web) é a entidade responsável por atender às solicitações HTTP dos clientes e, portanto, está continuamente em execução. As próximas seções farão uma breve descrição das camadas e seus respectivos elementos.

3.2.1 Camada de Apresentação

A camada de apresentação inclui todos os elementos que estão em contato direto com os usuários do sistema. O contato dos usuários é resumido a dois elementos principais:

- **Aplicação** (atualmente para *Android*) responsável por medir os parâmetros de rede e exibir aos usuários o ranking de operadoras e o mapa de cobertura. Aplicativo mede continuamente, em segundo plano, os dados que podem ser coletados a respeito das operadoras de redes celulares, incluindo parâmetros de performance (intensidade de sinal e taxa de dados) e outros parâmetros auxiliares como os códigos de operadora, de país, código EARFCN para determinação do canal em uso, entre outros;

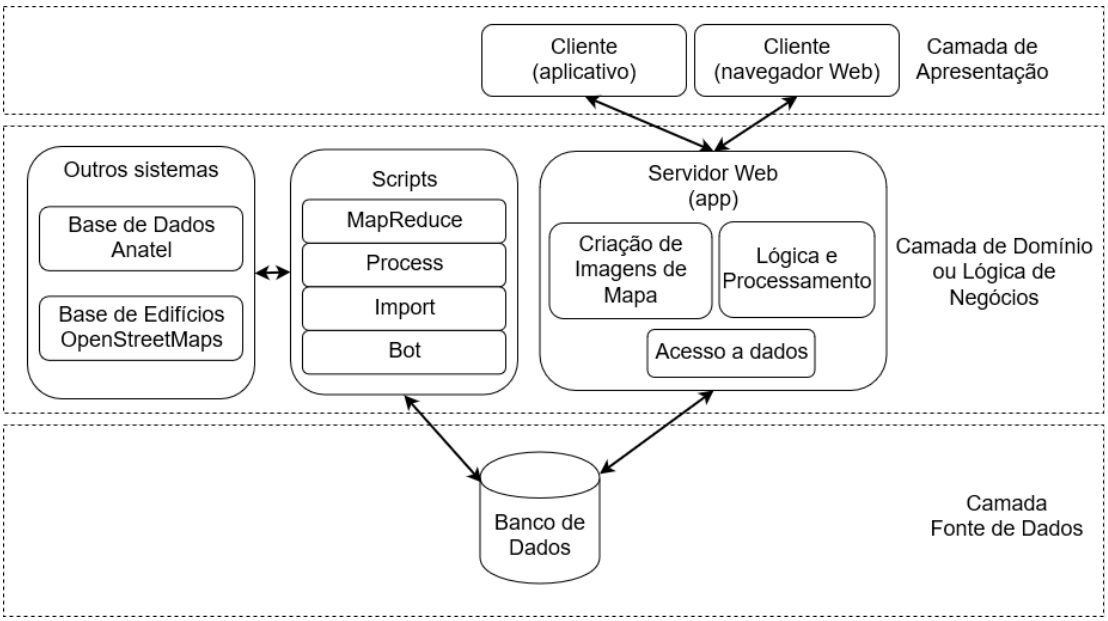


Figura 3.2: Arquitetura do sistema proposto e sua respectiva divisão em camadas.

- **Sistema Web** preparado para visualização nos navegadores chamado de **Painel de Controle**. Esse painel é utilizado apenas pelos administradores do sistema e fornece, além do mapa de cobertura, dados detalhados dos pontos de medição individuais, com suporte a filtros por operadora, tecnologia e dispositivo. Além de visualização de dados, o painel de controle também permite o gerenciamento das permissões dos usuários que podem acessar os dados administrativos.

A comunicação entre a camada de apresentação e a camada inferior (de domínio ou lógica de negócios) é feita por meio de uma API REST desenvolvida para a aplicação. Uma série de *endpoints* foram criados para permitir a gravação de medições de dados no banco de dados e a obtenção de informações agregadas para visualização dos usuários.

3.2.2 Camada de Domínio

Nessa camada foram posicionados os elementos responsáveis pelo processamento dos dados. Aqui, estão presentes todos os scripts de processamento de dados, os scripts de importação de dados de fontes externas, o servidor Web que faz consultas ao banco de dados e atende às requisições dos usuários.

3.2.2.1 Servidor Web

O elemento *Servidor Web* no diagrama é responsável por atender às requisições HTTP dos clientes e, portanto, é o responsável por filtrar as requisições, realizar o tratamento dos dados de entrada, autenticar os usuários, fazer transformações nos dados (quando aplicável) e entregar as imagens de mapa para visualização. Esse elemento está constantemente em conexão com o banco de dados, uma vez que suas operações se resumem a ler e escrever do mesmo, realizando as demais operações sobre os dados quando aplicável. Em termos gerais, as operações podem ser resumidas em dois grandes grupos:

- A gravação de pontos de medição enviadas pelos dispositivos dos usuários Nessa operação, é feita a validação dos dados com um esquema padrão para evitar inválidos enviados fora dos intervalos aceitáveis ou a existência de pontos sem os dados essenciais. Essa validação é feita utilizando a especificação de um esquema (*schema*) de dados que funciona como um gabarito para os dados enviados pelos dispositivos dos usuários;
- A obtenção e visualização de dados previamente agregados e processados, ou a obtenção de dados crus do banco de dados, normalmente sob solicitação dos administradores e pesquisadores. Apesar do servidor Web receber todos os dados, eles são armazenados e o processamento é responsabilidade dos scripts, que devolvem os dados agregados para o banco de dados para que a aplicação Web seja capaz de retornar informações relevantes para seus usuários.

Do ponto de vista de implementação, esse elemento é um servidor Web escrito utilizando a linguagem de programação Python [26] com o *framework* Tornado [27] para o tratamento assíncrono de requisições HTTP. Para autenticação dos usuários, uma vez que a aplicação Web utiliza REST como interface com os clientes, é necessário que as requisições não utilizem informações de estado, ou seja, todas as requisições e respostas associadas deverão possuir todas as informações necessárias para seu entendimento para ambas as partes da comunicação. Para alcançar esse objetivo, a autenticação dos usuários é feita por meio do JWT (*JSON Web Tokens*) [28].

O JWT permite que o sistema conheça o usuário e identifique as suas permissões por meio do *token* fornecido no cabeçalho de todas as requisições, a cada qual o servidor é responsável por decodificar e validar o *token*. Por ser assinado com um segredo conhecido apenas pelos servidores de aplicação e autenticação, um usuário malicioso não pode manipular o *token* para tentar ganhar acessos administrativos ao sistema, pois isso invalida a sua assinatura. Há a possibilidade de interceptação da comunicação para sequestro do *token* e, por isso, recomenda-se o uso de HTTPS, que emprega uma camada de criptografia usando TLS, impedindo que terceiros visualizem o conteúdo dos cabeçalhos das mensagens trocadas pela Internet.

No que diz respeito à escalabilidade, uma vez que se trata de uma API REST, nenhuma informação de sessão é necessária para comunicação e assim é possível aumentar a disponibilidade e capacidade do sistema por meio da execução da aplicação Web em diferentes máquinas ao mesmo tempo. Para isso, basta que a infraestrutura possua um balanceador de carga para requisições HTTP (um proxy reverso entre os usuários e os servidores de aplicação, que pode ser um servidor HTTP, um *appliance* de hardware dedicado ou um serviço de CDN contratado, por exemplo) que permita a distribuição das requisições nas múltiplas instâncias que executam em múltiplos servidores diferentes. Dessa forma, é possível multiplicar o número de requisições HTTP simultâneas atendidas a cada momento, tornando o sistema adequado para utilização em nuvens públicas com escalabilidade automática ou ainda em sistemas baseados em *containers* com criação e destruição automática.

As requisições HTTP que resultam em consultas custosas para o banco de dados possuem seu resultado armazenado em cache. O cache é feito no Redis [29], um banco de dados do tipo *chave-valor* que armazena seus dados em memória com uma data de validade específica. Após a validade do cache, o Redis cuida da exclusão automática dos dados.

Assim, a cada nova requisição, a aplicação faz uma consulta para verificar se há cache disponível e, se os dados não estiverem mais presentes, uma nova consulta ao banco de dados é feita e os resultados salvos em memória para acelerar consultas futuras. Isso é necessário devido ao alto volume de dados armazenados. Consultas ao banco de dados são custosas para uma coleção que possui potencialmente milhões de pontos de medição e, por isso, a utilização de recursos diminui com a utilização do cache, o que possibilita o atendimento de um maior número de usuários simultâneos.

3.2.2.2 Servidor de Imagens de Mapa

A geração de imagens de mapa existe para permitir que os clientes possuam uma maneira eficiente e leve de visualizar dados sobre um mapa. Quando um dispositivo cliente (como um navegador ou a aplicação Android) precisa visualizar uma grande quantidade de elementos no mapa, ele precisa manter todos os objetos desenhados em memória, o que prejudica o desempenho da aplicação e afeta a experiência dos usuários.

Uma forma de resolver isso é fornecer os dados por meio de imagens de mapa. Essas imagens são geradas por um serviço pertencente à aplicação Web e entregues para os dispositivos para que eles façam a sobreposição das imagens sobre os mapas, dessa forma, ao invés de possuir milhares de elementos simultâneos na tela, eles possuirão uma pequena quantidade de imagens de mapa contendo os desenhos dos elementos de interesse.

3.2.2.3 Scripts

Os scripts são rotinas escritas separadas da aplicação principal por serem chamadas em intervalos regulares de tempo, ou sob demanda. Essas rotinas tratam do processamento dos dados recebidos pelos dispositivos para que sejam retiradas informações úteis para os usuários. Isso é feito pois o processamento envolve grande volume de dados e os cálculos sob demanda afetariam a experiência de uso por meio de altos tempos de acesso devido às complexas consultas feitas ao banco de dados. Os resultados são gravados no banco de dados para utilização futura, uma vez que o servidor Web pode consumir esses dados para retorná-los ao usuário em forma de visualizações úteis. Existem quatro scripts no sistema proposto, conforme mostra a Figura 3.2:

- O **MapReduce** é responsável por determinar como o banco de dados irá agregar os dados utilizando o modelo MapReduce. Aqui, pode-se dizer que há uma quebra do modelo em camadas pois o script em si não é responsável pelo processamento, ele apenas comanda o que será executado no *MapReduce* integrado ao servidor de banco de dados. Uma vez que o processamento foi concluído, os dados agregados são salvos no banco de dados para consultas futuras.

- O **Process** é responsável por calcular os centroides dos *clusters* de pontos medidos com baixa intensidade de sinal utilizando o algoritmo *MeanShifts* apresentado na Seção 2.4.2 utilizando a implementação em [22].

- O **Import** é um conjunto de scripts responsáveis por importar dados de fontes externas. Por enquanto, o sistema importa as bases de ERBs provenientes da Anatel [8], utilizado nas visualizações dos mapas, e uma base aberta de edifícios disponibilizada pelo *OpenStreetMaps* [30] contendo polígonos prontos para uso futuro.

- O **Bot** é uma aplicação separada utilizada para monitoramento que notifica a equipe do projeto a cada 30 segundos informando os pontos de medição inseridos no intervalo. Para isso, ele utiliza a API do aplicativo de mensagens *Telegram*.

3.2.3 Camada de Fonte de Dados

A camada de fonte de dados é formada por um único banco de dados, onde são armazenados tanto os pontos de medição quanto os dados processados. O banco de dados escolhido para o sistema foi o MongoDB [31], um banco de dados não-relacional orientado a documentos que armazena os dados em formato JSON.

Quando um ponto de medição é recebido pela API, ele já vem no formato JSON. Então, uma série de validações são feitas nos campos para verificar que os dados não estão fora de intervalo ou com formato incorreto. Após essas verificações, o documento em JSON é passado diretamente para o banco de dados, onde é armazenado e indexado para consultas futuras.

O banco de dados separa os documentos em coleções, listadas abaixo:

- **Pontos**: Todos os pontos de medição são salvos nessa coleção;

- **Hexágonos**: Armazena os hexágonos resultantes da operação de MapReduce;

- **Ranking**: Armazena os dados referentes ao ranking das operadoras;

- **Clusters**: Armazena os clusters de pontos com baixa intensidade de sinal, chamados também de **pontos críticos**, calculados utilizando o algoritmo *MeanShift*;

- **Usuários**: Armazena os usuários registrados no sistema, seus respectivos emails de contato e permissões no sistema.

3.3 O MODELO MAPREDUCE

O *MapReduce* é um modelo de programação que permite o desenvolvimento de programas para processamento de grandes quantidades de dados em grandes *clusters* de processamento de maneira simplificada por meio da abstração dos procedimentos da divisão de processamento entre os computadores, tratamento de erros, controle do *cluster* e distribuição dos dados a serem processados. Apesar de originalmente desenvolvida pelo Google [32], implementações livres e de código aberto surgiram ao longo dos anos [33], o que contribuiu para a popularização do modelo e sua integração com diversos outros sistemas.

O modelo permite que os desenvolvedores das aplicações se concentrem em escrever códigos para processamento dos dados desejados, sem se preocupar com os detalhes do processamento paralelo. Contudo, é possível empregá-lo mesmo quando o processamento é feito em apenas uma máquina, seja por motivos de possibilidade de expansão futura (de uma máquina para um *cluster* de máquinas) seja pela facilidade de implementação. Neste trabalho, embora o processamento seja feito em uma única máquina, o modelo foi adotado pela facilidade fornecida na abstração do processamento dos dados. Quando necessário escalar o processamento, os códigos já foram escritos conforme a especificação do *framework* e será necessário fazer pequenas adaptações.

3.3.1 Formato dos Dados

Na especificação do modelo, os dados são trabalhados no formato chave-valor (*key-value*). O desenvolvedor é responsável por escrever duas funções: o *Map* e o *Reduce*. A rotina de *Map* recebe pares chave-valor na sua entrada e produz uma série de pares intermediários. Uma vez que diversos valores possuirão uma mesma chave, é necessário agrupar e ordenar as mesmas, o que resulta em novos pares chave-valor em que a chave é uma das chaves intermediárias e o valor é uma lista de valores emitidos pela rotina de *Map*. Após esse procedimento, o *Reduce* recebe como entrada os dados agrupados e é responsável por reduzir os dados para um conjunto ainda menor de chaves.

Sejam k_1 e k_2 dois tipos de chaves e v_1, v_2 dois tipos de valores, os tipos de dados associados às operações de *Map* e *Reduce* são [32]

$$\begin{array}{lllll} \text{map}: & (k_1, v_1) & \rightarrow & \text{list}(k_2, v_2) \\ \text{reduce}: & (k_2, \text{list}(v_2)) & \rightarrow & \text{list}(v_2) \end{array} . \tag{3.1}$$

A equação 3.1 mostra que a rotina de *Map* é responsável por gerar uma lista de pares chave-valor e também que entre a execução das duas rotinas ocorre um agrupamento que transforma $\text{list}(k_2, v_2)$ em $(k_2, \text{list}(v_2))$. Por fim, destaca-se que a rotina de *reduce* retorna apenas uma lista de valores, provavelmente muito menor do que o conjunto de dados fornecido no formato (k_1, v_1).

3.3.2 Descrição da Execução

O funcionamento do modelo originalmente proposto consiste em criar um processo mestre (*Master*) e uma série de processos executores (*workers*). O processo *master* coordena quais são as funções executadas pelos *workers* que, por sua vez, podem executar e máquinas separadas e podem receber para processamento tanto as funções de *Map*, quanto as funções de *Reduce*, de acordo com a disponibilidade dos demais *workers*. Com base nessa organização, a descrição da execução do *MapReduce* segue:

1. É feito o particionamento dos dados de entrada em M pedaços. Cada um deles é enviado para um processo executor diferente que executará a rotina de *Map* na partição correspondente. O resultado da execução da rotina de *Map* irá gerar resultados intermediários, que são armazenados em um *buffer* em memória;

2. Periodicamente, os resultados em *buffer* são salvos para o disco local, no quel é feito um novo particionamento: O espaço de resultados intermediários é dividido em R pedaços, em que cada um é enviado para um processo executor diferente para execução da rotina de *Reduce*;

3. Os processos responsáveis pelas rotinas de *Reduce* são notificados pelo processo mestre e, então, fazem a leitura do disco dos processos que executaram a rotina de *Map*, onde estão os dados previamente particionados. Então, os dados são ordenados e agrupados, conforme especificado anteriormente, para que possam ser enviados para execução a rotina de *Reduce*;

4. O processo mestre comanda a execução da rotina *Reduce* nos processos executores;

5. Após a execução completa do *Reduce* em todos os processos executores, as saídas são salvas e o programa é finalizado.

É possível destacar da execução acima a utilização dos termos *processos* tanto para o mestre quanto para os executores. É importante destacar que os processos podem estar em execução em máquinas diferentes, para aumentar a capacidade de processamento, mas que processos diferentes podem ser executados em uma única máquina com diversos núcleos de processamento. Para esse último caso, o processamento de grandes quantidades de dados não será tão eficiente, devido à menor capacidade de processamento total, entretanto, o modelo continua funcionando e os desenvolvedores podem optar por utilizá-lo pelas facilidades oferecidas pelas abstrações do mesmo.

A Figura 3.3 ilustra um exemplo de rotinas *Map* e *Reduce*. Nesse exemplo, o objetivo é simplesmente contar o número de ocorrências das classes e agrupá-las de forma que as classes B e C sejam classificadas como “2” e a classe A seja classificada como “1”.

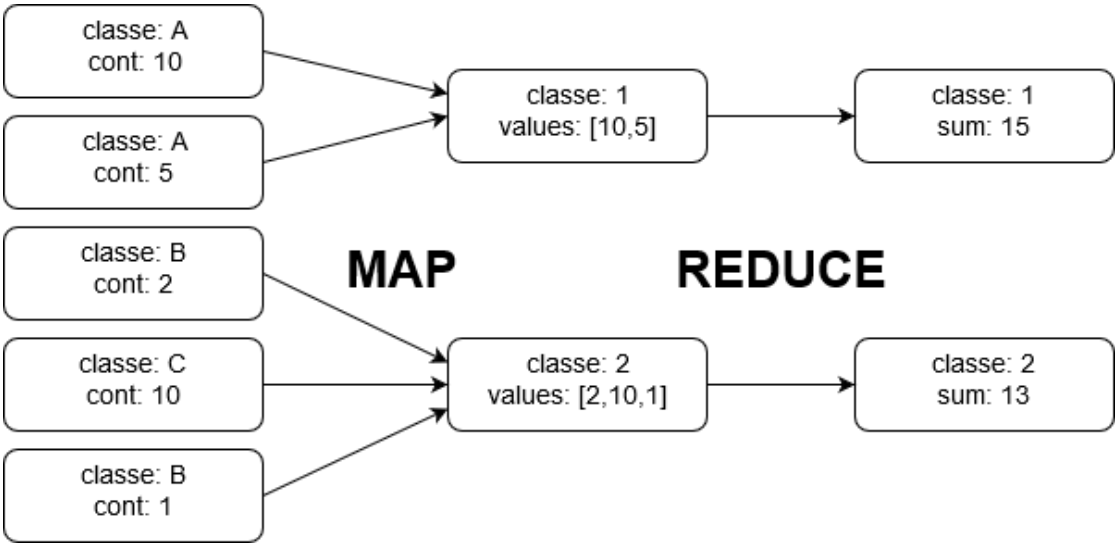


Figura 3.3: Exemplo de MapReduce aplicado a dados fictícios.

Na Figura 3.3 observa-se a relação apresentada na equação 3.1, ou seja, a operação de *Map* mapeia as classes para as novas classes e repassa os valores presentes na variável *cont*. O agrupamento de chaves iguais está implícito, assim como na equação, pois para a rotina de *Reduce* já é possível observar a lista de valores agrupados para cada valor de classe. Nesse exemplo, a variável *classe* é do tipo k_1, a variável *cont* (contagem) é do tipo v_2, a variável *classe* intermediária é do tipo k_2 e a variável *values* é do tipo list(v_2). Por fim, após a operação de *Reduce*, os dados foram simplesmente somados, ou seja, obteve-se ao final do processo uma redução de todos os dados de classe e suas respectivas contagens aos valores das novas classes com as contagens somadas.

É possível observar que é possível empregar diversas operações matemáticas na fase de *Map*, caso necessário para o processamento dos dados. Como exemplo, é possível supor que a identificação da classe seja um valor numérico que precisa ser mapeado por meio de uma operação matemática, como uma mudança de coordenadas, isso pode ser feito facilmente em paralelo por meio do emprego do modelo de *MapReduce*, o que distribui a carga dos cálculos realizados entre os nós de processamento, o que por sua vez aumenta a capacidade total de processamento do sistema. Além disso, é possível implementar uma forma de tratamento dos dados de forma a fazer com que a rotina *Map* emita para a fase seguinte somente dados que respeitarem algum filtro pré-determinado, o qe pode ser usado como uma forma de excluir dados inválidos.

O pseudo-código na Figura 3.4 ilustra como seria a rotina de *Map* escrita para o exemplo da Figura 3.3. É possível observar a complexidade adicional para criar o código no modelo do MapReduce. No entanto, as vantagens compensam a dificuldade adicional, nesse caso, por permitir que o processamento ocorra de forma paralela em vários computadores. Além disso, a tarefa de executar uma soma que foi escolhida como exemplo é simples e disfarça a simplicidade obtida com o modelo para cálculos mais complexos.

```
1 map (key, value):
2   class = value.classe
3   cont = value.cont
4   if class == 'A':
5     emit(1, cont)
6   else:
7     emit(2, cont)
```

(a) Rotina *Map*.

```
1 reduce (key, values):
2   sum = 0
3   for value in values:
4     sum += value
5   return sum
```

(b) Rotina *Reduce*.

Figura 3.4: Pseudocódigo de exemplo do MapReduce da Figura 3.3.

3.4 PROPOSTA DE APLICAÇÃO DO MAPREDUCE AOS DADOS COLETADOS

A Seção 3.2.3 apresentou a camada de apresentação do sistema desenvolvido e foi mencionada a presença de um Mapa de Cobertura. Esse mapa foi criado com o objetivo de facilitar a visualização das informações agregadas pelos usuários, com o objetivo de ajudá-los a encontrar o melhor par operadora e tecnologia para uma determinada região de interesse. Existem algumas técnicas para visualização de dados sobrepostos sobre um mapa.

O mapa de calor permite que os dados sejam visualizados sem a utilização de figuras geométricas específicas por meio da criação de gradientes cujas cores variam com a métrica de interesse. Essa abordagem é bastante difundida e existem diversas implementações prontas que facilitam o processo de implementação, contudo, ela comete erros na generalização dos dados para diferentes níveis de zoom no mapa, fazendo com que regiões aparentem ter cobertura sem necessariamente possuírem tal cobertura ou ainda sem possuírem medições válidas.

Além do mapa de calor, é possível utilizar figuras geométricas para delimitar regiões e atribuir uma métrica de interesse a cada uma delas. Foi decidido que o sistema utilizaria hexágonos regulares devido à duas características principais que eles apresentam:

1. Permitem se encaixar perfeitamente sobre a superfície de visualização ao mesmo tempo em que se aproximam de círculos, que podem apresentar sobreposição ou espaços em branco no mapa;

2. Não possuem a desvantagem de retângulos em que os vértices são muito mais distantes das arestas, o que faz com que os dados próximos aos vértices sejam potencialmente diferentes em relação aos dados do centro;

3. Fazem referência ao projeto de redes celulares utilizando hexágonos, o que indica que parte da literatura e operações matemáticas já está difundida e bem estabelecida;

4. Permitem a determinação dos polígonos a partir de qualquer ponto arbitrário no mapa, conforme apresentado na Seção 2.3. Essa característica é fundamental pois permitiu o processamento e agregação de dados em larga escala utilizando o modelo de programação que será apresentada nas próximas sessões.

Por simplicidade e agilidade de implementação, foi adotado a funcionalidade de *MapReduce* integrada ao banco de dados utilizado no projeto. Uma vez que a lógica se adapta ao modelo e não muda de implementação para implementação, os códigos podem ser portados para outras plataformas, como o *MapReduce* do Hadoop, sem grandes dificuldades além da mudança de linguagem de programação. Essa abordagem também possui a vantagem de que o processamento é feito próximo aos dados, diminuindo a penalização de desempenho que ocorre ao movimentar grandes volumes de dados pela rede.

As próximas sessões apresentarão o formato dos dados presentes nas medições enviadas pelos dispositivos e, em seguida, apresentarão a lógica das rotinas de *Map*, *Reduce* e *Finalize* criadas para utilização do modelo de processamento *MapReduce*.

3.4.1 Descrição dos dados obtidos nas medições

Para entender as chaves e os valores que surgirão no detalhamento da implementação dos modelos para processamento, é necessária uma breve descrição sobre cada um dos campos. Cada um dos campos mencionados é um parâmetro presente em cada medição feita por um *smartphone*, conforme mostra a Tabela 3.1.

Campo	**Tipo**	**Descrição**
op	Texto	Nome da operadora
tech	Número	Número identificador da tecnologia
sstr	Número	Intensidade de Sinal (Potência) em dBm
mobileTX	Número	Taxa de upload medida em segundo plano
mobileRX	Número	Taxa de download medida em segundo plano
pci	Número	Código PCI retornado pela rede
lnglat	Número[2]	Coordenadas obtidas com o GPS
acc	Número	Precisão do GPS (em metros)
speed	Número	Velocidade de deslocamento conforme o GPS

Tabela 3.1: Tabela com uma amostra dos parâmetros medidos pelo sistema.

Nessa tabela, os campos *mobileTX* e *mobileRX* podem precisar de um melhor detalhamento: Esses são os dados calculados para a medição em segundo plano da taxa de dados instantânea dos aparelhos. A cada medição, é calculada a diferença do tempo entre o instante atual e o instante da medição anterior e a diferença entre o contador de dados do sistema operacional para os dois instantes. Assim, é criada a estimativa de taxa de dados em segundo plano, chamada de *mobileTX* para o contador de dados transmitidos e *mobileRX* para o contador de dados recebidos.

Os campos de contagem (*countMobileRX*, *countMobileTX* e *countSstr*) são utilizados para inicializar os contadores de pontos que possuem medições válidas de *mobileRX*, *mobileTX* e *sstr_mw*, respectivamente. Isso é necessário pois poderão haver pontos em que os campos são iguais a zero (por não haver tráfego na rede de dados), mas há um valor de potência válido e, assim, é possível acompanhar as contagens individualmente para calcular corretamente as médias nas rotinas posteriores.

O campo *size* é utilizado para diferenciar os hexágonos caso seja necessário gerar hexágonos de múltiplos tamanhos. Assim, é possível criar um mapa de cobertura com hexágonos de tamanho variável. Apesar da possibilidade, essa funcionalidade ainda não foi implementada.

3.4.2 Map

A rotina de *Map* que foi implementada atua sobre a coleção de pontos, ou seja, percorre todos os pontos de medição existentes no banco de dados, e possui cinco funções básicas:

1. **Mapear** os pontos de medição para as coordenadas do centro dos hexágonos regulares e mapeamento desses centros para as coordenadas completas do hexágono;
2. **Mapear** a tecnologia em cada ponto de medição (Tabela 2.1) para uma sequência de caracteres que identifica a tecnologia (“2G”, “3G” ou “4G”);
3. **Mapear** o nome da operadora para um nome de operadora padronizado, uma vez que as operadoras podem possuir nomes diferentes em localizações diferentes, de acordo com a configuração da rede. Como exemplo, a operadora **TIM** pode se apresentar como “TIM 61”, “TIMBRASIL”, “TIM | TIMBRASIL”, entre outros, e deve ser identificada no sistema somente como “TIM”;
4. **Converter** o valor de medição de dBm para milliWatts, uma vez que todos os cálculos precisam ser feitos em escala linear para montagem do mapa de cobertura;
5. **Filtrar** pontos de medição relevantes para cálculos dos hexágonos, ou seja, pontos com dados inválidos ou fora de formato são ignorados. Além disso, são escolhidos os pontos de medição em que o dispositivo do usuário estava de fato associado à célula no momento da medição, ou seja, dados de medição referente às células vizinhas não são considerados.

Os processos de filtragem são definidos por estruturas do tipo *if .. else* dentro da rotina, fazendo com que somente os pontos de interesse sejam emitidos adiante para os processos responsáveis por executar a rotina *Reduce*. O mapeamento de tecnologias consiste em uma varredura em uma estrutura fixa com os dados presentes na Tabela 2.1, o mapeamento dos centros dos hexágonos é feito utilizando o algoritmo e operações algébricas apresentadas na Seção 2.3 e, por fim, a conversão de dBm para milliWatts é feita utilizando

$$P_{\text{mw}} = 10^{P_{\text{dbm}}/10}, \tag{3.2}$$

em que P_{dbm} é a potência medida em dBm e P_{mw} é a potência calculada em milliWatts que será passada adiante para a rotina *Reduce* continuar com os cálculos.

```
key = (
  op,
  tech,
  center,
  coordinates,
  size,
  center_grid
)
```

(a) Definição da chave k_2.

```
value = (
  sstr_mw,
  mobileTX,
  mobileRX,
  countMobileRX,
  countMobileTX,
  countSstr
)
```

(b) Definição do valor v_2.

Figura 3.5: Definição das chaves e valores k_2 e v_2 para as rotina de *MapReduce*.

Na nomenclatura da equação 3.1, o sistema define a chave k_2 com a identificação do hexágono gerado, conforme mostra a Figura 3.5a, e a chave v_2 com os valores correspondentes ao hexágono identificado por k_2, conforme mostra a Figura 3.5b. Um exemplo dos resultados gerados pela rotina *Map* para um ponto de medição pode ser visualizado na Figura 3.6. Nesse exemplo é possível observar as coordenadas do hexágono determinadas com base no ponto de medição dado como entrada da rotina, a transformação no nome da operadora, a transformação do número da tecnologia no código da tecnologia e, por fim, a inicialização dos contadores.

```
{
  "sstr"; -100,
  "mobileRX": 10000,
  "mobileTX": 1000,
  "op": "TIM 61",
  "tech": 13,
  "lnglat": [-47.8699387,-15.76702]
}
```

(a) Dados de entrada.

```
{
  key: {
    "size": 0.000561444,
    "center": [-47.86955760600001, -15.766810513099815],
    "op": "TIM",
    "tech": "4G",
    "coordinates": [
      [-47.87011905000001,-15.766810513099815],
      [-47.86983832800001,-15.766324288333012],
      [-47.86927688400001,-15.766324288333012],
      [-47.86899616200001,-15.766810513099815],
      [-47.86927688400001,-15.767296737866618],
      [-47.86983832800001,-15.767296737866618],
      [-47.87011905000001,-15.766810513099815]
    ],
    "center_grid": [-56841,44634]
  },
  value: {
    "sstr_mw": 0.0000000001,
    "mobileTX": 10000,
    "mobileRX": 1000,
    "countMobileRX": 1,
    "countMobileTX": 1,
    "countSstr": 1
  }
}
```

(b) Dados de saída.

Figura 3.6: Exemplo de dados de entrada e saída da rotina *Map*.

3.4.3 Reduce

A rotina *Reduce* é chamada após a execução de uma rotina de *Map* ou após uma execução prévia da mesma rotina *Reduce*. Por isso, os dados de entrada e saída dessa rotina deverão possuir o mesmo formato dos dados de saída da rotina *Map* apresentados na Figura 3.5. A especificação da rotina depende dos resultados desejados ao final do processamento. No caso, o objetivo é encontrar as médias de intensidade de sinal e taxas de dados para cada hexágono processado.

Então, o *Reduce* possui a função de acumular todos os contadores e demais métricas. Para cada chave k_2, a rotina pode ser chamada várias vezes e, portanto, o código deve ser construído de forma a permitir o acúmulo das métricas e contadores, no qual os dados presentes em todos os valores v_2 recebidos são somados e retornados. Assim, a construção da rotina se resume a uma estrutura *for* .. que itera sobre list(v_2) (equação 3.1) e realiza a soma das métricas de interesse.

```
{
  "key": { ... },
  "values": [
      {
        "sstr_mw": 0.0000000001,
        "mobileRX": 10000,
        "mobileRX": 1000,
        "countMobileRX": 100,
        "countMobileTX": 135,
        "countSstr": 311
      },
      {
        "sstr_mw": 0.0000000002,
        "mobileRX": 20000,
        "mobileRX": 500,
        "countMobileRX": 244,
        "countMobileTX": 141,
        "countSstr": 132
      },
      // abaixo: proveniente diretamente de um Map
      {
        "sstr_mw": 0.00000000015,
        "mobileRX": 10000,
        "mobileRX": 200,
        "countMobileRX": 1,
        "countMobileTX": 1,
        "countSstr": 1
      }
  ]
}
```

(a) Dados de entrada.

```
{
  "sstr_mw": 0.00000000045,
  "mobileRX": 40000,
  "mobileRX": 1700,
  "countMobileRX": 345,
  "countMobileTX": 277,
  "countSstr": 444
}
```

(b) Dados de saída.

Figura 3.7: Exemplo de dados de entrada e saída da rotina *Reduce*.

A Figura 3.7 ilustra um exemplo da entrada e saída de dados de uma rotina *Reduce*. Nesse caso, a chave foi omitida por organização e espaço e entre os valores emitidos, o terceiro elemento foi emitido diretamente a partir de uma rotina *Map*, equanto os dois primeiros foram emitidos a partir de outro *Reduce*. É possível observar que os dados de entrada e saída possuem o meso formato e que o exemplo é um caso especial em que a saída é do tipo list(v_2) contendo apenas um elemento.

Vale ressaltar ainda que não necessariamente os dados do tipo list(v_2) possuirão dados provenientes de operações de *Reduce* e *Map* em um mesmo objeto, entretanto, eles foram apresentados assim na Figura para ilustrar a preparação para que os dados possam ser provenientes de qualquer uma das duas rotinas.

3.4.4 Finalize

A última rotina é uma rotina de finalização que é chamada imediatamente dos dados serem salvos no banco de dados. Ela é usada para fazer pequenas adaptações nos dados finais, finalizando os cálculos desejados. No sistema proposto, uma vez que o interesse está no cálculo das médias, essa rotina se encarrega de dividir os valores acumulados de *mobileRX*, *mobileTX* e *sstr_mw* pelos seus respectivos contadores *countMobileRX*, *countMobileTX* e *countSstr*. Assim, são obtidos os valores médios para as taxas de dados instantâneas e para a intensidade de sinal, que por sua vez ainda é convertida para dBm em uma última etapa usando o inverso da operação na equação 3.2.

A Figura 3.8 mostra o formato dos dados de saída que serão salvos no banco de dados com a chave do tipo k_2.

```
{
  "count": 444,
  "countMobileRX": 345,
  "countMobileTX": 277,
  "avgSstr": 10*log10(0.00000000045 / 244), // dBm
  "mobileTX": 1700 / 277, // bps
  "mobileRX": 40000 / 345 // bps
}
```

Figura 3.8: Exemplo de dados de saída da rotina *Finalize*.

Ao final do processo, os dados serão inseridos no banco de dados usando a chave do tipo k_2 emitida pela função *Map* e os dados apresentados na Figura 3.8 como seus respectivos valores.

3.5 GERAÇÃO DE IMAGENS DE MAPA

Durante o desenvolvimento do sistema, foram testadas duas formas de montagem do mapa de cobertura para visualização. A primeira forma consiste em ter as figuras (polígonos e pontos) desenhados pelo *frontend* sobre o mapa, assim, seria a responsabilidade do navegador Web (no caso do painel de controle) ou do aplicativo Android de criar os objetos e desenhá-los sobre o mapa.

Entretanto, devido ao tamanho dos hexágonos, há um grande número de elementos presente no mapa, o que prejudica o desempenho devido à necessidade de manter muitos objetos em memória ao mesmo tempo e de criar constantemente novos objetos de acordo com a movimentação do mapa. Para isso, foi adotada a abordagem de criar um servidor de imagens de mapa integrado ao sistema que desenha os elementos em imagens no backend, o que diminuiu o processamento necessário nos dispositivos e melhorou a experiência de uso. A sessão a seguir descreverá o funcionamento desse serviço, explorando os detalhes de geração das imagens.

As aplicações comerciais de mapas mais comuns dividem as imagens exibidas nos mapas em quadrados denominados *tiles*, cada qual é identificado em uma projeção específica que varia de aplicação para aplicação. Entre as projeções, uma das mais comuns é o EPSG:3857 [34], que identifica os *tiles* por meio de um sistema de coordenadas (x, y, z), em que z representa o nível de zoom aplicado à visualização do mapa.

O servidor de imagens de mapa é um servidor responsável por receber requisições HTTP com as coordenadas (x, y, z) e retornar a imagem do *tile* correspondente a essas coordenadas. Essa imagem será sobreposta ao mapa pelos clientes, criando uma camada de visualização. Diversas camadas podem existir simultaneamente, e qualquer figura arbitrária pode ser desenhada nos *tiles* de cada camada. Assim, ao invés dos clientes terem a responsabilidade de desenhar centenas de elementos no mapa, sua responsabilidade se resume a posicionar no máximo algumas dezenas de imagens sobre o mapa.

Cada *tile* é uma região quadrada, cujos limites em coordenadas decimais variam com o nível de zoom aplicado, conforme mostra a Figura 3.9. A cada visualização de mapa por cliente, diversas requisições são feitas e, por isso, essa operação pode exigir grandes quantidades de processamento. Por isso, para aumentar a eficiência do sistema, as imagens são colocadas em *cache* no Redis com tempo de vida de 1 hora.

(a) Zoom = 14. (b) Zoom = 15.

Figura 3.9: Visualização das imagens de mapa para a métrica de intensidade de sinal em diferentes níveis de zoom.

Ao receber a requisição, a aplicação utiliza a informação das coordenadas (x, y, z) no formato de uma URI personalizada (*operadora/tecnologia/métrica/x/y/z*) para verificar se o *tile* requisitado está ou não em cache. Caso esteja em cache, ele retorna o mesmo para o cliente e, caso contrário:

1. A aplicação utiliza a informação das coordenadas (x, y, z) para calcular os limites (em termos de coordenadas decimais). Como pode ser visto na Figura 3.9, diferentes níveis de zoom resultam em regiões de tamanhos diferentes;

2. É montada uma região quadrada que delimita os pontos de interesse e, com isso, a aplicação faz uma consulta no banco de dados filtrando os hexágonos que respeitam o filtro de operadora e tecnologia fornecido e que possuem coordenadas (longitude, latitude) dentro da região quadrada calculada anteriormente;

3. Os tiles são desenhados utilizando uma biblioteca [35], que desenha os hexágonos nas posições corretas, atribui uma cor de acordo com o valor da métrica de interesse e gera um arquivo de imagem binário;

4. A aplicação responde à requisição HTTP com o arquivo de imagem gerado

3.6 CONCLUSÃO

Este capítulo apresentou a fundamentação teórica necessária para o desenvolvimento de certos tipos de aplicação Web e como essa arquitetura foi utilizada como modelo para a criação do sistema proposto. Além disso, o MapReduce foi descrito em detalhes que foram utilizados para explicar a implementação do mesmo no contexto dos pontos de medição coletados.

É importante destacar que a adoção de um framework como esse permite que o processamento dos dados possa ser feito em qualquer sistema compatível com pequenas adaptações no formato dos dados. Ainda faz-se necessária a mudança do código para diferentes linguagens de programação, mas as ideias básicas e a organização dos dados permanece inalterada. Assim é possível, por exemplo, implementar o processamento dos dados em um *cluster* envolvendo dezenas de nós de processamento conforme as necessidades de escalabilidade do sistema.

O trabalho consistiu na escrita de todas essas rotinas para o processamento dos pontos de medição coletados pelo sistema. O MapReduce é utilizado em um único nó de processamento e é capaz de calcular todos os hexágonos com base em cerca de 10 milhões de pontos de medição espalhados pelo mundo em aproximadamente 30 minutos. Também é possível expandir facilmente o processamento para qualquer métrica de interesse com poucas alterações nos códigos. Os resultados referentes a esse processamento serão apresentados no Capítulo 4 em conjunto com os resultados dos demais conceitos apresentados em capítulos anteriores.

4 RESULTADOS

4.1 MAPA DE COBERTURA

4.1.1 Média de intensidade de sinal

A Figura 3.9 mostrou os *tiles* gerados para a métrica de intensidade de sinal. As cores adotadas para o hexágono seguem o valor da média de intensidade de sinal medida para os pontos circunscritos pelo mesmo que é determinada utilizando o *MapReduce*. O valor dos intervalos é configurável e seus valores atuais são:

- Ruim, cor vermelha, potência média abaixo de -100 dBm;
- Médio, cor amarela, potência média entre -100 e -80 dBm;
- Bom, cor verde, potência média acima de -80 dBm.

Em conjunto com o processamento da média, é feito o processamento de outras métricas, entre elas a média de taxas de dados, a contagem de pontos de medição e o código identificador de célula (PCI) mais frequente. Todas elas estão disponíveis para visualização por meio de hexágonos nos mapas de cobertura. As métricas serão descritas brevemente nas próximas sessões.

4.1.2 Média de taxa de dados

A média de taxa de dados, descrita como *mobileRX* na Tabela 3.1, indica a média dos valores de taxas de dados instantâneas medidas em segundo plano para todos os pontos de medição circunscritos no hexágono. É uma média calculada usando o MapReduce conforme a Seção 3.3.

A Figura 4.1 mostra dois *tiles* para essa métrica para uma determinada operadora com filtro de tecnologia 4G em dos níveis de zoom diferentes. Poucos hexágonos são visualizados devido à menor quantidade de pontos disponíveis com essa métrica.

Os intervalos para as cores adotadas com essa métrica são configuráveis e atualmente estão definidos como:

- Ruim, cor vermelha, média abaixo de 100 Kbps;

(a) Zoom = 14. (b) Zoom = 15.

Figura 4.1: Visualização das imagens de mapa para a métrica de média de taxa de dados instantânea em diferentes níveis de zoom.

- Médio, cor amarela, média entre 100 e 200 Kbps;
- Bom, cor verde, média acima de 200 Kbps.

4.1.3 Contagem de pontos de medição

A contagem de medições indica quantos pontos de medição com valor de intensidade de sinal válido existem dentro de um determinado hexágono. Essa métrica pode ser utilizada para inferir a densidade de usuários do sistema em diferentes locais, ou ainda a demanda de cada operadora em termos de número de usuários, considerando um volume grande o bastante de medições.

A Figura 4.2 mostra dois *tiles* para essa métrica para uma determinada operadora com filtro de tecnologia 4G em dois níveis de zoom diferentes. Devido à baixa quantidade de usuários do sistema até o momento, poucos pontos de medição estão disponíveis na maior parte das regiões.

Os intervalos para as cores adotadas com essa métrica são configuráveis e atualmente estão definidos como:

- Ruim, cor vermelha, contagem abaixo de 50 pontos;
- Médio, cor amarela, contagem entre 51 e 100 pontos;
- Bom, cor verde, contagem acima de 101 pontos.

(a) Zoom = 14. (b) Zoom = 15.

Figura 4.2: Visualização das imagens de mapa para a métrica de contagem de pontos em diferentes níveis de zoom.

4.1.4 Código PCI mais frequente

O código PCI é um código para redes 4G (LTE) que identifica fisicamente uma célula na rede da operadora e, devido à limitação de valores possíveis, deve ser reutilizado. A métrica se baseia na informação de código PCI mais frequente dentro de um hexágono para colorir o mesmo. Ao contrário das métricas anteriores, as cores não são quantificadores de qualidade, mas sim uma simples identificação. Como exemplo, se um hexágono conter um ponto de medição com código PCI 110 e dez pontos de medição com código PCI 215, ele será rotulado com a cor correspondente ao código 215. Essa métrica pode ser usada para identificar potenciais problemas de configuração nos equipamentos a respeito da distribuição dos códigos PCI entre as células.

As cores foram escolhidas por meio da divisão dos valores inteiros possíveis para identificação das mesmas (cerca de 65 mil valores, pois as imagens geradas possuem 16 bits de profundidade de cores) em 512 regiões igualmente espaçadas, pois esse é o número de valores que o código PCI pode assumir. Os códigos das divisas das regiões são os códigos de cores escolhidos e, assim, existem 512 cores diferentes que podem ser utilizadas.

Cores semelhantes indicam códigos PCI próximos. Cores iguais indicam o mesmo código reutilizado. Conforme descrito no Capítulo 2, o código PCI não pode ser reutilizado em duas células geograficamente próximas. Se isso acontecer, há um potencial erro de configuração. Assim, esse mapa de cobertura pode auxiliar na detecção dessas falhas de configuração ou ainda ajudar a perceber células com cobertura maior do que o planejado.

(a) Zoom = 14. (b) Zoom = 15.

Figura 4.3: Visualização das imagens de mapa para a métrica de código PCI mais frequente em diferentes níveis de zoom.

4.1.5 Pontos

Além das figuras hexagonais apresentadas, o sistema também gera imagens de mapa contendo pontos. Os pontos são identificados por uma imagem identificando as coordenadas que se deseja marcar. Esses desenhos são arbitrários, configuráveis e podem ser modificados a qualquer momento. Atualmente, dois tipos de pontos são utilizados:

- As ERBs, importadas da base de dados da Anatel
- Os pontos críticos, calculados com o algoritmo MeanShift.

Essa visualizações foram criadas como uma forma de facilitar a correlação entre dados, como a presença ou ausência de pontos críticos em função da presença próxima de uma ERB, por exemplo, ou ainda as cores que identificam os códigos PCI em função das ERBs próximas.

A Figura 4.4 é uma captura de tela de parte da visualização do mapa contendo três camadas de *tiles*. A camada de ERBs possui um marcador com desenho de uma pequena torre nos pontos que foram importados da base de dados da Anatel, a camada de pontos críticos possui um desenho de ponto de exclamação para cada ponto crítico calculado com o algoritmo *MeanShit* e a camada de média de intensidade de sinal com os hexágonos e suas respectivas cores associadas aos valores médios correspondentes.

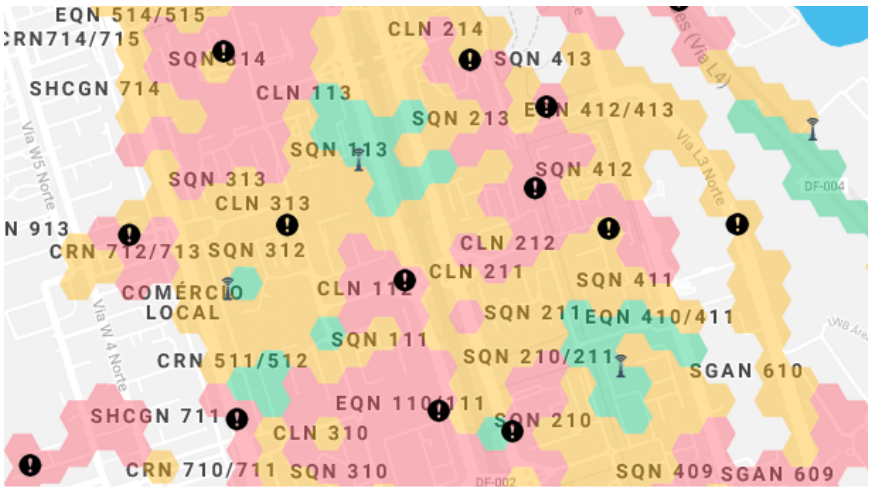


Figura 4.4: Exibição do mapa de cobertura mostrando a métrica de intensidade de sinal, os pontos críticos e as ERBs importadas da base da Anatel simultaneamente.

4.2 AGRUPAMENTO POR ESTIMAÇÃO DE DENSIDADE

Os capítulos anteriores apresentaram como o modelo de MapReduce pode ser utilizado no cálculo de métricas de médias de intensidade de sinal e taxas de dados para pontos delimitados dentro hexágonos que podem ser utilizados no mapa de cobertura. Uma região que possua muitos hexágonos próximos com valores baixos para as suas médias pode ser uma região com problemas de acesso, no entanto, um valor médio elevado para as métricas pode esconder problemas.

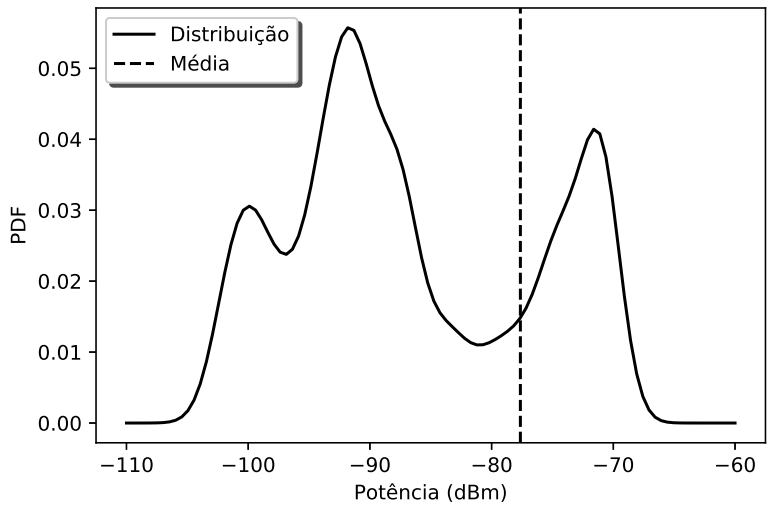


Figura 4.5: Distribuição de potência utilizando pontos de medição de uma região arbitrária.

A Figura 4.5 apresenta a distribuição de probabilidade estimada com base em cerca de 6800 pontos de medição de uma única operadora feitos com tecnologia 4G e em uma região com valores altos de média de intensidade de sinal. A distribuição foi estimada utilizando um kernel Gaussiano com $h = 1.5$ e o valor médio indicado pela linha vertical tracejada é de cerca de -77,6 dBm.

O alto valor de potência encontrado disfarça o fato de muitos pontos estarem concentrados ao redor de -90 dBm e -100 dBm. Ou seja, um alto valor de média de intensidade de sinal não implica necessariamente que não hajam problemas de cobertura causadas por baixa intensidade de sinal recebida pelos dispositivos.

4.2.1 Filtragem de pontos

Antes de entrar nos detalhes da implementação do algoritmo *MeanShift*, é preciso abordar alguns aspectos sobre os parâmetros de cada ponto de medição e como eles foram filtrados antes de sua passagem para o algoritmo. Primeiramente destaca-se que os pontos utilizados são filtrados de acordo com as seguintes condições:

- Pontos de medição efetivamente associados a uma célula, isto é, pontos de medição que mediram informação de potência de uma célula vizinha não são considerados;
- Pontos medidos usando tecnologia 4G (LTE);
- Pontos que mediram potência recebida abaixo de um limiar, atualmente configurado para -110 dBm.

Atualmente, há maior interesse na análise dos dados em tecnologia 4G LTE, por isso o segundo filtro acima existe. Entretanto, não há impedimento em executar o algoritmo em uma segunda rodada para as tecnologias de terceira geração (3G), por exemplo.

Para acelerar a execução do algoritmo, sua execução é limitada a uma região de cada vez. Define-se como uma região um hexágono regular de tamanho configurável (atualmente 0,1 coordenadas decimais de raio) que é calculado junto com os demais hexágonos do mapa de cobertura com as técnicas exploradas na Seção 2.3 e utilizando o *MapReduce* conforme a Seção 3.2.3. Esse grande hexágono não está visível nos mapas de cobertura e é utilizado apenas para delimitar a execução do algoritmo de agrupamento, o que evita que o algoritmo tenha que lidar com uma grande quantidade de pontos simultaneamente.

4.2.2 Detecção de Pontos Críticos com MeanShift

O algoritmo *MeanShift* é uma forma de utilizar estimação não-paramétrica de densidades para encontrar centroides de *clusters* por meio de um algoritmo iterativo de busca de máximos locais baseado em gradiente ascendente.

O sistema proposto busca encontrar *clusters* de pontos com baixa intensidade de sinal, Assim, o algoritmo é executado para todos as medições existentes que mediram uma potência igual ou inferior a um limiar configurável. É possível assim encontrar regiões no planeta que representam o máximo local da densidade desses pontos, efetivamente fazendo com que os pontos sejam os centroides dos agrupamentos de pontos com baixa intensidade de sinal. Esses pontos são os pontos críticos mostrados na Figura 4.4 e o limiar de potência está atualmente configurado com o valor de -110 dBm.

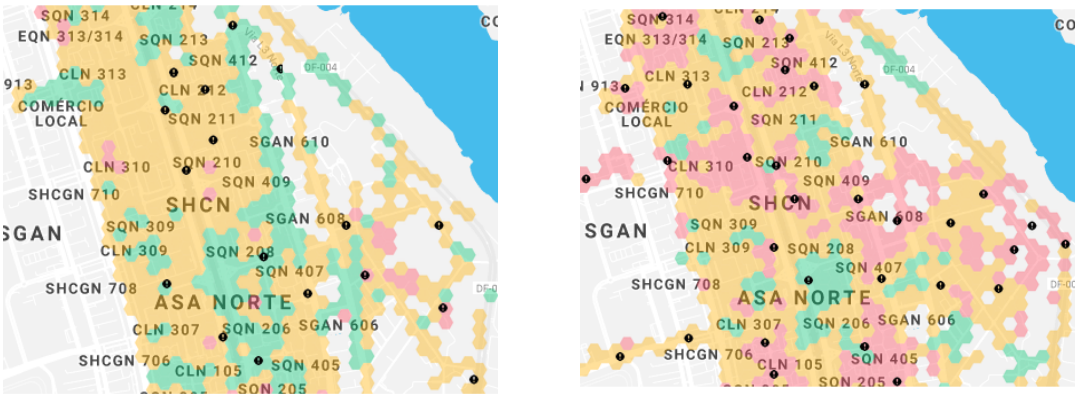
A implementação do MeanShift que foi adotada [22] utiliza um kernel plano conforme a equação 2.10 e permite a especificação de alguns parâmetros, entre eles:

1. Tamanho da região de influência de cada ponto utilizado pelo algoritmo, chamado de largura de banda (*bandwidth*) ou simplesmente *bw*;

2. *Flag bin_seeding* que indica que ao invés de utilizar todos os pontos como pontos iniciais do algoritmo permite que os pontos sejam discretizados em regiões (*bins*) de tamanho fixo igual ao parâmetro *bw*. Essa opção foi marcada como verdadeira pois acelera a execução do algoritmo uma vez que existem menos pontos de partida para a execução do algoritmo;

3. *Flag cluster_all* que indica que todos os pontos precisam ser agrupados, mesmo aqueles que estão muito distantes dos centroides dos grupos. Essa opção foi marcada como false, pois, dessa forma, em uma situação em que exista apenas um ponto isolado em uma região, ele será considerado *outlier* e não será vinculado a nenhum grupo.

A escolha do parâmetro *bw* é importante pois permite resumir (agrupar) corretamente as informações visualizadas no mapa. Como exemplo, um valor de *bw* muito pequeno faz com que muitos centroides de cluster existam próximos um do outro, o que não é o resultado desejado uma vez que o objetivo do agrupamento é visualizar poucos pontos que sejam representantes dos demais pontos próximos com características semelhantes. Por outro lado, um valor de *bw* muito alto faz com que muitos centroides sejam descartados e os agrupamentos sejam feitos com distâncias muito elevadas, o que diminui o significado atribuído a cada ponto crítico.



(a) Operadora T.

(b) Operadora V.

Figura 4.6: Visualização no mapa dos pontos críticos calculados com o *MeanShift*.

O ajuste do parâmetro foi feito visualmente e o valor de *bw*, que é configurável, foi adotado como quatro vezes o valor do raio do hexágono. A Figura 4.6 mostra um pedaço do mapa de cobertura mostrando os seus pontos críticos. Mais detalhes sobre a geração do mapa de cobertura e o significado das cores dos hexágonos são apresentados no Capítulo 4.1.

Destaca-se da Figura 4.6 a presença de pontos críticos em regiões cujos hexágonos indicam uma boa média de potência medida pelos dispositivos, representados pelos hexágonos de cor verde. Esse é o mesmo fenômeno que foi observado no gráfico de estimação de densidade da Figura 4.5, ou seja, uma média elevada esconde o fato de existirem pontos medidos com baixos valores de intensidade de sinal.

4.3 IMPLEMENTAÇÃO DO RANKING DAS OPERADORAS

Além da visualização dos hexágonos no mapa de cobertura e dos pontos críticos calculados pelo sistema, o usuário pode precisar de mais informações para que seja capaz de escolher a melhor operadora para as suas regiões de interesse. Com essa possível necessidade em mente, foi montado um *ranking* de operadoras que busca classificar, em ordem, as operadoras e, com isso, criar uma visualização em forma de gráficos que permite que os usuários visualizem claramente as informações de interesse.

O ranking é feito com base em uma composição das métricas de intensidade de sinal (parâmetro *sstr_mw* da Tabela 3.1), taxa de transmissão instantânea de recepção em segundo plano (parâmetro *mobileRX* da Tabela 3.1), e a taxa de transmissão de recepção média durante a execução de um teste artificial de taxa de dados, conhecido popularmente como *SpeedTest*.

Os capítulos anteriores mostraram que os valores médios de quaisquer parâmetros disfarçam potenciais problemas de cobertura que as operadoras venham a possuir. Para isso, foi decidido que seria utilizada a distribuição dos parâmetros coletados de alguma forma, para garantir que a frequência relativa dos intervalos de valores medidos seja considerada.

4.3.1 Metodologia

Foi definido um filtro para os dados que seriam utilizados para o cálculo do ranking. Primeiramente, todos os hexágonos existentes previamente calculados para o Brasil foram levantados e, para cada um deles, até três pontos de medição foram aleatoriamente amostrados. Além do filtro por localização, são considerados também apenas pontos de medição referentes à tecnologia 3G ou 4G, no entanto, os dados são analisados individualmente. Por fim, são filtrados os pontos cujo valor de *mobileRX* (vide Tabela 3.1) são inferiores a 50 Kbps, o que é suficiente uma vez que esse parâmetro se refere a medições em segundo plano.

Os valores de intensidade de sinal, taxa de dados instantânea em segundo plano e taxa de dados instantânea em execução de teste artificial (*SpeedTest*) são colocados em *arrays* de dados separados e, portanto, são analisados individualmente. Então, para cada uma das métricas coletadas, é calculada a sua CDF (distribuição de densidade acumulada). A Figura 4.7 ilustra as CDFs para a intensidade de sinal, definida por $F_{P_i}(P = \text{sstr})$ para cada operadora i.

A partir dessa CDF, os pontos mais próximos das frequências relativas 0.4 e 0.8 são somados. Para o caso da intensidade de sinal, o valor é previamente convertido de dBm para milliWatt. Ao final desse processo, há valores acumulados para cada uma das métricas, separados por operadora. Por fim, ocorre o processamento do ranking:

1. O maior valor entre todos os valores salvos é salvo para ser utilizado como referência;
2. É feita então uma interpolação linear que posiciona o valor acumulado para cada operadora entre uma escala de 0 a 100;

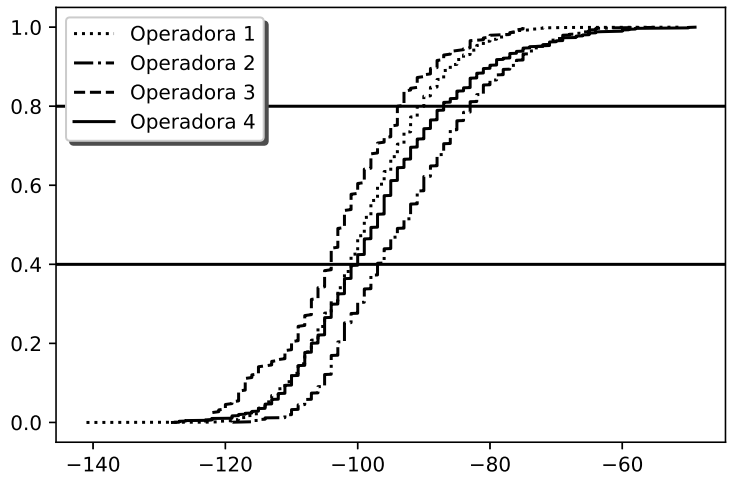


Figura 4.7: CDF da distribuição dos valores de potência para diferentes operadoras. Linhas horizontais marcam os pontos de interesse.

3. É atribuído um peso (configurável) ao valor interpolado para cada uma das métricas de forma que maior pontuação possível seja igual a 1. Isso permite variar a importância das métricas no cálculo do ranking.

As operações descritas nos passos acima podem ser resumidas em forma de uma equação. Seja s_i a pontuação da operadora i, **P** o vetor contendo todos os valores de potência acumuladas para cada operadora, em que cada valor p_i é a potência em milliWatts acumulada para a operadora i. Analogamente, o vetor **Q** contem todos os valores de taxas de dados em segundo plano, com cada elemento q_i contendo o valor acumulado para a operadora i, o vetor **R** contem todos os valores de taxas de dados medidas em testes artificiais, com cada elemento r_i contendo o valor acumulado para a operadora i.

Por fim, sejam w_p o peso (ou importância) aplicado ao parâmetro de potência, w_q a importância aplicada ao parâmetro de taxa de dados em segundo plano e w_r o peso aplicado à taxa de dados em testes artificiais. A fórmula que descreve o valor de s_i com base nos parâmetros analisados é

$$s_i = 100 \left(w_p \frac{p_i}{\max(\mathbf{P})} + w_q \frac{q_i}{\max(\mathbf{Q})} + w_r \frac{r_i}{\max(\mathbf{R})} \right), \tag{4.1}$$

em que os valores são definidos em termos das CDFs inversas dos dados medidos correspondentes a cada operadora i

$$
\begin{aligned}
p_i = &\quad F_{P_i}^{-1}(0.4) + F_{P_i}^{-1}(0.8), \\
q_i = &\quad F_{Q_i}^{-1}(0.4) + F_{Q_i}^{-1}(0.8), \\
r_i = &\quad F_{R_i}^{-1}(0.4) + F_{R_i}^{-1}(0.8),
\end{aligned}
\tag{4.2}
$$

em que os valores de probabilidade 0.4 e 0.8 são parâmetros de configuração do sistema e foram escolhidos arbitrariamente caso seja necessário utilizar outros valores na determinação do ranking.

Os resultados do ranking conforme a metodologia apresentada anteriormente são mostrados na Figura 4.8. Esse ranking foi feito levando em consideração os pontos de medição de todo o Brasil. No entanto, há a possibilidade de restringir o cálculo do ranking em regiões hexagonais genéricas maiores ou ainda por município, caso seja possível obter dados de fontes abertas sobre a delimitação dos mesmos para que sejam importados no sistema.

(a) Tecnologia: 3G. (b) Tecnologia: 4G.

Figura 4.8: Visualização do ranking de operadoras calculado pelo sistema.

Destaca-se da Figura 4.8 que as operadoras podem apresentar resultados muito diferentes ao comparar as tecnologias 3G e 4G. O resultado do ranking é uma visualização de alto nível que leva em consideração a distribuição dos pontos de medição dentro da região de análise. A metodologia mostrou que as operadoras possuem apresentar bons resultados de taxa de dados e intensidade de sinal para se posicionarem de maneira favorável no ranking.

Por se tratar de um método que combina diversos parâmetros medidos, há maior probabilidade de uma operadora se posicionar nas primeiras posições se sua rede for capaz de manter tanto boa intensidade de sinal quanto bons valores de taxas de dados.

4.4 IMPORTAÇÃO DE DADOS E MONITORAMENTO

Devido à complexidade do sistema e à grande quantidade de módulos diferentes envolvendo a captação de dados e o processamento, foi necessário criar uma estratégia de monitorar o recebimento dos dados de medição enviados pelos dispositivos.

Dessa forma, é possível detectar problemas nos servidores, caso os mesmos parem de receber dados por qualquer motivo, além de observar a quantidade de dados enviada pelos dispositivos, o que permite o monitoramento sem acesso aos logs da aplicação. Assim, o desenvolvimento do aplicativo é facilitado, já que é possível verificar rapidamente o efeito das modificações nos códigos. As seções a seguir descrevem e exemplificam o funcionamento do sistema de monitoramento.

A princípio, é possível pensar em um sistema de monitoramento que notifica sempre que um novo lote de pontos é recebido. Entretanto, a qualquer instante de tempo, o sistema recebe uma quantidade aleatória de pontos de medição. Da mesma forma, é possível que o sistema passe períodos sem receber nenhum ponto de medição. Assim, essa não é uma maneira escalável devido ao número de notificações potencialmente elevado.

Assim, foi criado um sistema que utiliza uma fila para notificações. Essa fila contem o resumo dos lotes de medições recebidas. Essa fila pode crescer livremente por períodos de 30 segundos. Ao final do período, o sistema de monitoramento agrupa as informações por dispositivo e tecnologia, envia as informações ao responsáveis e limpa a fila.

Em caso de problemas, a fila não é limpa e os dados ficam armazenados até a próxima rodada de envios. Atualmente, a fila é implementada em memória usando o Redis, o que permite que o processo de monitoramento seja isolado do processo principal da aplicação, pois as duas entidades podem utilizar esse meio como um mecanismo de troca de dados.

A Figura 4.9 mostra algumas mensagens enviadas pelo sistema de monitoramento. Nessa Figura, é possível observar a informação referente à tecnologia dos lotes de pontos recebidos (13, indicando 4G), o dispositivo que enviou as medições (“Mi A1”) e a indicação *cellRegistered*, que se refere à quantidade de pontos de medição referentes a momentos em que o dispositivo estava, de fato, associado a uma célula. Por fim, cada mensagem exibida se refere a uma rodada de notificações feita pelo sistema de monitoramento.

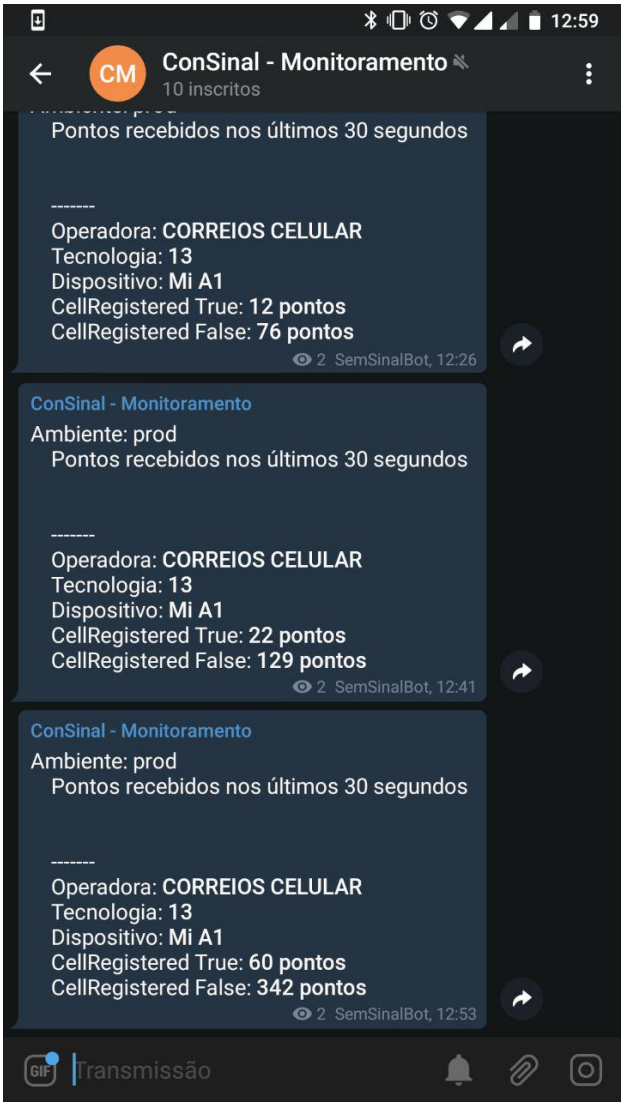


Figura 4.9: Captura de tela mostrando as mensagens enviadas pelo sistema de monitoramento.

A Figura 3.2 mostrou que há interfaces entre o sistema proposto e outros sistemas, como a base de dados de ERBs licenciadas da Anatel e e base de edifícios aberta do *OpenStreetMaps*. A primeira base é utilizada para permitir a visualização das localizações das torres em conjunto com o mapa de cobertura e as outras visualizações disponíveis. A segunda está reservada para usos futuros e poderá, por exemplo, ajudar a determinar quando uma medição foi feita em ambiente interno ou externo.

Os dados são obtidos manualmente a partir do sistema de licenciamento da Anatel [8]. Ou seja, apenas dados referentes ao Brasil são levados em conta por enquanto, apesar de ser possível expandir para outros países utilizando os sistemas das suas respectivas agências reguladoras, como a OFCOM no Reino Unido [9], por exemplo.

Cada arquivo obtido da Anatel refere-se às ERBs cadastradas para uma unidade da federação. Assim, foi criado um script de importação que lê cada um desses arquivos, faz as adaptações necessárias para o formato de dados adequado e os salva no banco de dados. Assim, o banco de dados do sistema possui uma coleção que armazena todas as estações licenciadas pela Anatel, permitindo a consulta por filtros de localização, assim como é feito com os hexágonos e pontos individuais de medição.

Por fim, a importação da base de endereços e formas do *OpenStreetMaps* proveniente de [30] é um pouco mais complicada devido ao formato dos dados, que traz as informações de coordenadas no formato *shapefile* criado pelo ESRI. Assim, é necessário utilizar uma biblioteca para interpretação desses dados para extrair as coordenadas e adaptá-las ao formato dos dados utilizado no banco de dados. Uma vez que os dados são adaptados, os dados são salvos em uma coleção de endereços, onde ficam presentes as informações abertas dos limites de edifícios do Brasil inteiro.

4.5 CONCLUSÃO

A geração de imagens de mapa utilizando um módulo do servidor de aplicação fornece uma interface simplificada e padronizada para qualquer tipo de cliente exibir as informações que foram calculadas nos servidores. Por ser adotada a projeção que é considerada padrão de mercado, qualquer cliente pode utilizar facilmente a API fornecida para as imagens de mapa, mostrando simultaneamente todas as camadas de interesse do usuário.

A implementação também foi padronizada, fazendo com que seja possível montar um mapa de cobertura com hexágonos utilizando qualquer métrica coletada, assim como exibir qualquer tipo de pontos no mapa desde que estejam presentes nos bancos de dados. Esse formato de visualização permite uma melhor experiência de uso devido à rapidez no carregamento das imagens, o que permite que o mapa seja utilizado livremente sem interrupções e que o usuário possa facilmente exibir ou ocultar as camadas de visualização disponíveis.

A visualização criada com os pontos críticos permite obter informações adicionais sobre problemas de cobertura que não seriam facilmente visualizados apenas com os hexágonos e suas respectivas médias de intensidade de sinal, conforme ilustrado na Figura 4.6. Essa informação pode ser útil para pesquisadores interessados em analisar o algoritmo nessas condições, para as operadoras interessadas em detectar potenciais problemas de cobertura e para os usuários, que poderão analisar as informações para ajudar a encontrar a melhor operadora para suas regiões de interesse.

O nível de granularidade dos centroides dos clusters pode ser configurado pelos administradores do sistema através do parâmetro *bw*. Contudo, sua configuração atual permite visualizar uma boa quantidade de grupos sem exibir muita informação redundante no mapa de cobertura.

5 CONCLUSÃO

O sistema apresentado é capaz de coletar métricas de desempenhos sobre redes celulares ao redor do mundo, independente de tecnologia, operadora ou dispositivo em uso. Dessa forma, qualquer pessoa que queira contribuir com o sistema pode instalar em seu dispositivo e deixar que as medições sejam feitas, o que enriquece o banco de dados permitindo a criação de mapas de cobertura mais completos. Assim, todos os usuários se beneficiam das medições dos demais.

Um usuário interessado em escolher a melhor operadora de redes celulares para a sua região pode utilizar o aplicativo e visualizar, através do mapa de cobertura, as regiões em que o sinal das operadoras é mais forte ou mais fraco. Assim, a operadora com as melhores médias de intensidade de sinal podem ser escolhidas, de acordo com o interesse do usuário. Contudo, ainda não há suporte à delimitação de regiões por cidades ou municípios, uma possibilidade que depende da existência de fontes de dados abertas e confiáveis e do emprego de algoritmos eficientes de delimitação geográfica.

Também foi visto que a média de intensidade de sinal disfarça potenciais problemas de cobertura, uma vez que uma média alta não significa que a região está isenta de pontos com baixa intensidade de sinal e/ou outros problemas de cobertura. Dessa forma, o algortimo *MeanShift* se mostrou adequado na identificação de pontos críticos, pois é capaz de encontrar locais em que a densidade de pontos com baixa intensidade de sinal são pontos de máximos locais. Essa é uma funcionalidade que não existe nas outras ferramentas pesquisadas e nem mesmo na ferramenta comercial mais comum [36] e é uma ferramenta poderosa na identificação de pontos em que há possibilidade de melhorar a experiência dos usuários.

Por fim, foi mostrado que a flexibilidade do modelo permite a obtenção de médias para **qualquer métrica de interesse**, não somente média de intensidade de sinal ou taxa de dados. Isso permite expandir o funcionamento do sistema para analisar diversos parâmetros não limitadas somente a redes celulares.

5.1 SUGESTÕES DE TRABALHOS FUTUROS

Foi mostrado que o sistema é capaz de coletar informações geo-referenciadas envolvendo métricas de redes celulares. Entretanto, o emprego das técnicas descritas permite que o sistema se estenda para quaisquer métricas de interesse, até mesmo fora do escopo de redes móveis. É possível, por exemplo, obter informações de sensores espalhados em diversos locais, situação em que apenas o tipo do valor medido é diferente, mas todas as técnicas de agregação permanecem aproximadamente as mesmas.

O sistema possui potencial para armazenamento e processamento de grandes volumes de dados. De fato, o uso do modelo de *MapReduce* é um indicativo de que é necessário empregar técnicas de **Big Data** para que o sistema continue escalável e possa atender mais usuários simultâneos. Assim, é possível empregar técnicas no escopo de *Mobile Big Data* especificamente para redes móveis [37].

Espera-se que este trabalho possa ser útil como base para coleta e análise de dados provenientes de usuários interessados. Como exemplo, um estudante interessado em analisar a intensidade de sinal em uma região para realização de um trabalho poderia utilizar o sistema como ferramenta. Também espera-se que seja possível criar sistemas semelhantes com base nas técnicas apresentadas e incentivar o surgimento de novas formas de processamento de dados utilizando BigData e aprendizado de máquina que utilizem os dados coletados por meio dos equipamentos dos usuários interessados.

Também é possível criar uma modificação no aplicativo para aumentar a taxa de amostragem das medições em troca da diminuição da autonomia de bateria. Com isso, um pesquisador interessado pode utilizar o aplicativo para realizar medições controladas em uma região de interesse para, por exemplo, obter informações úteis sobre modelagem de canal sem fio ou informações sobre a propagação do sinal.

Por fim, o tratamento genérico aos dados coletados permite que futuras gerações de redes celulares possam ser medidas e analisadas sem grandes mudanças no sistema. Assim, um possível trabalho futuro é ampliar a funcionalidade do sistema para coleta e análise de parâmetros de redes 5G, o que também pode ser útil durante as fases preliminares de projeto e implementação.

5.2 PUBLICAÇÕES ASSOCIADAS

As seguintes publicações estão associadas à esta dissertação:

- Artigo de Iniciação Científica premiado com Menção Honrosa apresentado no 23º Congresso de Iniciação Científica da Universidade de Brasília: “Desenvolvimento de plataforma computacional para análise de qualidade de redes celulares” [38];
- Artigo de Iniciação Científica que recebeu prêmio de melhor artigo de Iniciação Científica apresentado no XXXV Simpósio Brasileiro de Telecomunicações e Processamento de Sinais: “Protótipo de Aplicação para Análise de Intensidade de Sinais de Redes Celulares” [39];
- Artigo Completo apresentado no XXXVI Simpósio Brasileiro de Telecomunicações e Processamento de Sinais: “Sistema Colaborativo para Medição e Análise de Redes Celulares Baseado em Aplicativo Móvel” [40].

REFERÊNCIAS

[1] M. Lauridsen, I. Rodriguez, L. M. Mikkelsen, L. C. Gimenez, and P. Mogensen, "Verification of 3G and 4G received power measurements in a crowdsourcing Android app," in *Wireless Communications and Networking Conference (WCNC), 2016 IEEE*. IEEE, 2016, pp. 1–6.

[2] U. Goel, M. P. Wittie, K. C. Claffy, and A. Le, "Survey of end-to-end mobile network measurement testbeds, tools, and services," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 105–123, 2016.

[3] "Boletim 2017 Consumidor.gov.br," https://www.consumidor.gov.br/pages/publicacao/externo/, 2017, acessado em 10/09/2018.

[4] A. Nikravesh, H. Yao, S. Xu, D. Choffnes, and Z. M. Mao, "Mobilyzer: An open platform for controllable mobile network measurements," in *Proceedings of the 13th Annual International Conference on Mobile Systems, Applications, and Services*. ACM, 2015, pp. 389–404.

[5] S. Sonntag, J. Manner, and L. Schulte, "Netradar: Measuring the wireless world," in *Modeling & Optimization in Mobile, Ad Hoc & Wireless Networks (WiOpt), 2013 11th International Symposium on*. IEEE, 2013, pp. 29–34.

[6] S. Rosen, S.-j. Lee, J. Lee, P. Congdon, Z. M. Mao, and K. Burden, "Mcnet: Crowdsourcing wireless performance measurements through the eyes of mobile devices," *IEEE Communications Magazine*, vol. 52, no. 10, pp. 86–91, 2014.

[7] V. H. Mac Donald, "Advanced mobile phone service: The cellular concept," *The Bell system technical Journal*, vol. 58, no. 1, pp. 15–41, 1979.

[8] "Anatel: Estações licenciadas," https://sistemas.anatel.gov.br/se/public/view/b/licenciamento.php, 2018, acessado em 10/09/2018.

[9] "OFCOM sitefinder database (archive)," http://webarchive.nationalarchives.gov.uk/tna/20150106113122/http://stakeholders.ofcom.org.uk/sitefinder/sitefinder-dataset/, 2012, acessado em 10/09/2018.

[10] T. S. Rappaport *et al.*, *Wireless communications: principles and practice*. prentice hall PTR New Jersey, 1996, vol. 2.

[11] “ETSI technical report tr 103 116.” https://www.etsi.org/deliver/etsi_tr/103100_103199/103116/01.01.01_60/tr_103116v010101p.pdf, 2012, acessado em 24/09/2018.

[12] W. Lu and M. Di Renzo, “Stochastic geometry modeling of cellular networks: Analysis, simulation and experimental validation,” in *Proceedings of the 18th ACM International Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems*. ACM, 2015, pp. 179–188.

[13] “3GPP specification 23.003,” https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=729, 1999, acessado em 10/09/2018.

[14] “Evolution to LTE report,” http://www.gsacom.com/news/gsa_365.php, 2012, acessado em 10/09/2018.

[15] “ETSI technical specification TS 136 106,” https://www.etsi.org/deliver/etsi_ts/136100_136199/136106/12.01.00_60/ts_136106v120100p.pdf, 2015, acessado em 24/09/2018.

[16] “ETSI technical specification TS125 106,” https://www.etsi.org/deliver/etsi_ts/125100_125199/125106/10.02.00_60/ts_125106v100200p.pdf, 2012, acessado em 24/09/2018.

[17] “ETSI technical specification TR 136 902,” https://www.etsi.org/deliver/etsi_tr/136900_136999/136902/09.03.01_60/tr_136902v090301p.pdf, 2011, acessado em 24/09/2018.

[18] I. Her, “Geometric transformations on the hexagonal grid,” *IEEE Transactions on Image Processing*, vol. 4, no. 9, pp. 1213–1222, 1995.

[19] “WGS84 - world geodetic system 1984, used in gps,” https://epsg.io/4326, acessado em 04/10/2018.

[20] E. Alpaydin, *Introduction to machine learning*. MIT press, 2009.

[21] D. Comaniciu and P. Meer, “Mean shift: A robust approach toward feature space analysis,” *IEEE Transactions on pattern analysis and machine intelligence*, vol. 24, no. 5, pp. 603–619, 2002.

[22] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[23] T. Berners-Lee, R. Fielding, and H. Frystyk, "RFC 1945: Hypertext transfer protocol—http/1.0, may 1996," *Status: INFORMATIONAL*, vol. 61, 2005.

[24] R. T. Fielding and R. N. Taylor, *Architectural styles and the design of network-based software architectures*. University of California, Irvine Doctoral dissertation, 2000, vol. 7.

[25] M. Fowler, *Patterns of enterprise application architecture*. Addison-Wesley Longman Publishing Co., Inc., 2002.

[26] G. Van Rossum and F. L. Drake, *Python language reference manual*. Network Theory United Kingdom, 2003.

[27] "Tornado web framework," https://www.tornadoweb.org/en/stable/, 2018, acessado em 02/10/2018.

[28] M. Jones, J. Bradley, and N. Sakimura, "RFC 7519: Json web token (jwt)," *IETF, May*, 2015.

[29] "Redis," https://redis.io/, 2018, acessado em 02/10/2018.

[30] "Geofabrik: Openstreetmap data," https://download.geofabrik.de/south-america/brazil.html, 2018, acessado em 02/10/2018.

[31] "MongoDB," https://www.mongodb.com/, 2018, acessado em 02/10/2018.

[32] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[33] "Hadoop," http://hadoop.apache.org/, 2009, acessado em 01/10/2018.

[34] "EPSG:3857 - SR-ORG:6864," http://spatialreference.org/ref/sr-org/epsg3857/, acessado em 04/10/2018.

[35] "Mapnik," https://mapnik.org/, 2018, acessado em 04/10/2018.

[36] "Opensignal," https://opensignal.com/, accessed: 2018-04-08.

[37] M. S. Parwez, D. B. Rawat, and M. Garuba, "Big data analytics for user-activity analysis and user-anomaly detection in mobile wireless network," *IEEE Transactions on Industrial Informatics*, vol. 13, no. 4, pp. 2058–2065, 2017.

[38] J. G. Santos, Valle, and U. Dias, “Desenvolvimento de plataforma computacional para análise de qualidade de redes celulares,” *23º Congresso de Iniciação Científica da UnB*, vol. 1, no. 1, 2017.

[39] J. G. Santos, E. C. R. Costa, M. A. Rocha, V. Carazza, and U. Dias, “Protótipo de aplicação para análise de intensidade de sinais de redes celulares,” *XXXV SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES E PROCESSAMENTO DE SINAIS*, vol. 1, no. 1, pp. 762–763, 2017.

[40] J. G. Santos, L. M. Valle, E. C. R. Costa, and U. Dias, “Sistema colaborativo para medição e análise de redes celulares baseado em aplicativo móvel,” *XXXV SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES E PROCESSAMENTO DE SINAIS*, vol. 1, no. 1, pp. 1024–1028, 2018.