MEC-SETEC
INSTITUTO FEDERAL MINAS GERAIS - Campus Formiga
Curso de Ciência da Computação

OTIMIZAÇÃO DO POSICIONAMENTO DE PONTOS DE ACESSO WIRELESS

Formiga - MG
2017

MEC-SETEC
INSTITUTO FEDERAL MINAS GERAIS - Campus Formiga
Curso de Ciência da Computação

OTIMIZAÇÃO DO POSICIONAMENTO DE PONTOS DE ACESSO WIRELESS

Monografia do trabalho de conclusão de curso apresentado ao Instituto Federal Minas Gerais - Campus Formiga, como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais

Campus Formiga

Ciência da Computação

Orientador: Everthon Valadão

Formiga - MG

2017

MEC-SETEC
INSTITUTO FEDERAL MINAS GERAIS - Campus Formiga
Curso de Ciência da Computação
OTIMIZAÇÃO DO POSICIONAMENTO DE PONTOS DE ACESSO WIRELESS/
MEC-SETEC
INSTITUTO FEDERAL MINAS GERAIS - Campus Formiga
Curso de Ciência da Computação. – Formiga - MG, 2017-
91 p. : il. (algumas color.) ; 30 cm.

Orientador: Everthon Valadão

Monografia – Instituto Federal de Educação, Ciência e Tecnologia de Minas Gerais
Campus Formiga
Ciência da Computação, 2017.

1. Palavra-chave1. 2. Palavra-chave2. 2. Palavra-chave3. I. Orientador. II. Universidade xxx. III. Faculdade de xxx. IV. Título

MEC-SETEC
INSTITUTO FEDERAL MINAS GERAIS - Campus Formiga
Curso de Ciência da Computação

OTIMIZAÇÃO DO POSICIONAMENTO DE PONTOS DE ACESSO WIRELESS

Monografia do trabalho de conclusão de curso apresentado ao Instituto Federal Minas Gerais - Campus Formiga, como requisito parcial para a obtenção do título de Bacharel em Ciência da Computação.

Trabalho aprovado. Formiga - MG, 14 de novembro de 2017:

Everthon Valadão
Orientador

Diego
Convidado 1

Rafael
Convidado 2

Walace
Convidado 3

Formiga - MG
2017

Este trabalho é dedicado às crianças adultas que,
quando pequenas, sonharam em se tornar cientistas.

Agradecimentos

Agradeço a Deus, pois sem Sua ajuda, direção e o Seu agir em minha vida, não teria capacidade e o empenho para chegar até aqui; por se fazer presente até nos momentos em que os desafios foram tão grandes quanto minha vontade, por me ter dotado de saúde, sabedoria e disposição para alcançar mais uma etapa em minha vida.

Agradeço aos meus pais que, com toda humildade e simplicidade, me ensinaram a ser uma pessoa decente, a respeitar e buscar meus sonhos de forma honesta e dentro do meu tempo, mesmo que seja com muito trabalho árduo.

Agradeço ao responsável pela parte de TI do *campus*, Roger Ferreira, e ao engenheiro civil do *campus*, Alysson Geraldo, por terem gentilmente cedido as plantas-baixas.

Agradeço ao professor Everthon Valadão, a paciência e compreensão que teve comigo durante o período em que me acompanhou e que estivemos juntos realizando este trabalho. Esteve sempre presente como um grande professor e amigo, sempre se mostrando comprometido em dar o seu melhor para os alunos.

"Não vos amoldeis às estruturas deste mundo,
mas transformai-vos pela renovação da mente,
a fim de distinguir qual é a vontade de Deus:
o que é bom, o que Lhe é agradável, o que é perfeito.
(Bíblia Sagrada, Romanos 12, 2)

Resumo

A proposta deste trabalho é desenvolver um *software* para *Wireless AP Placement* que utilizará modelo(s) da propagação do sinal *Wi-Fi* de acordo com as características físicas do ambiente, aplicando o *Simulated Annealing* como metaheurística para visar ao aprimoramento da cobertura de sinal, tendo como caso de uso as dependências do IFMG campus Formiga. Tal *software* indicará um melhor posicionamento dos *access points Wi-Fi* para o ambiente analisado. O trabalho realizado foi disponibilizado de forma gratuita utilizando Licença Pública Geral GNU. Ressaltamos que ele possibilita testar disposições de APs sem o custo operacional de fisicamente movê-los, de maneira a propor uma disposição espacial dos mesmos que forneça uma maior cobertura e intensidade de sinal dentro do ambiente simulado. Para trabalhos futuros, a fim de se obter resultados que possibilitem uma melhor busca pelos pontos de acesso para dois ou mais APs, novas técnicas deverão ser utilizadas para o cálculo da função objetivo.

Palavras-chave: Wireless, Propagação, Posicionamento, Wi-Fi, Simulated Annealing, CUDA.

Abstract

This is the english abstract.

Keywords: latex. abntex. text editoration.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

IEEE	Institute of Electrical and Electronics Engineers
AP	Access Point
WLAN	Wireless Local Area Network
WiFi	Wireless Fidelity
LAN	Local Area Network
ISM	Industrial, Scientific and Medical
GHz	Gigahertz
MHz	Megahertz
RF	Radio Frequency
GNU	Gnu's Not Unix
GPL	General Public License
T-R	Transmitter-Receiver
dB	decibel
FOSS	Free Open Source Software
DXF	Drawing Exchange Format
CAD	Computer-aided Design
DWG	Drawing
2D	2 dimensões
3D	3 dimensões
EMI	Interferência Eletromagnética
CUDA	Compute Unified Device Architecture
GPU	Graphics Processing Unit
CPU	Central Processing Unit

HPC	High Performance Computing
API	Application Programming Interface
PCIe	Peripheral Component Interconnect Express
Gb/s	Gigabits por segundo
GB	Gigabits
dBm	Decibel Miliwatt
SA	Simulated Annealing

Lista de símbolos

Δ	Delta maiúsculo
μ	Mi
δ	Delta minúsculo
ξ	Ksi
σ	Sigma
α	Alpha
γ	Gamma

Sumário

1 INTRODUÇÃO

A demanda por disponibilidade e cobertura de redes locais sem fio (WLAN), seja em ambiente corporativo ou doméstico, tem crescido vertiginosamente. À medida que o uso de *Wi-Fi* se torna trivial e cotidiano, suas tecnologias são aperfeiçoadas e os preços dos equipamentos para acesso sem fio se tornam mais acessíveis.

Em um ambiente onde computadores tipicamente realizam a sua comunicação através de rede local cabeada (LAN), caso seja necessária a realização de alguma mudança física no ambiente, será inevitável o uso de arrumações, canaletas ou até obras na estrutura física do prédio. Conforme ocorrem alterações no ambiente de trabalho, variando desde a colocação dos móveis e divisórias a mudanças de sala, a limitação imposta pelos cabos se torna um problema de organização e planejamento das futuras mudanças. Quando há a necessidade de ampliar a rede para a acomodação de novos pontos de acesso para telecomunicação, a necessidade de se passarem novos cabos torna-se inconveniente. O obstáculo às alterações se torna ainda maior quando o ambiente é alguma construção em que não é viável realizar intervenções na parede (e.g., tombamento ou estética) para a redistribuição dos cabos, por vezes em edifícios, onde, sequer, houve planejamento de uma rede cabeada estruturada.

Considerando o acima exposto, é possível observar o quanto é conveniente o uso de redes sem fio (*wireless*) quando há a necessidade de ampliar a rede ou implantar, mesmo que temporariamente, o acesso à internet em alguma determinada área. Apesar da praticidade, não é viável ter uma rede *wireless* se sua cobertura de sinal não atinge boa parte da área necessária ou, então, se por mais perto que o ponto de acesso (AP) *wireless* esteja do computador, não provê uma qualidade de serviço satisfatória devido a um baixo nível de sinal ou interferências na mesma frequência do canal.

Portanto, se faz necessário um bom planejamento do posicionamento dos APs *wireless*, de maneira a oferecer uma boa cobertura de sinal onde se fizerem necessários. É importante deixar claro que o tratamento de colisão de canais como sua interferência de APs próximos não está dentre os principais objetivos dessa pesquisa e, por conseguinte, não foi implementado. A proposta deste trabalho é desenvolver um *software* para *Wireless AP Placement*, que utilizará modelo(s) da propagação do sinal Wi-Fi de acordo com as características físicas do ambiente, aplicando metaheurísticas que visem ao aprimoramento da cobertura de sinal, tendo como caso de uso as dependências do IFMG *campus* Formiga. Tal *software* indicará um melhor posicionamento dos *access points Wi-Fi* para o ambiente analisado.

1.1 Justificativa

É importante ressaltar a validade prática deste trabalho a ser desenvolvido, uma vez que uma grande dúvida dos usuários residenciais e profissionais de TI é saber qual a melhor localização para o(s) AP(s) *Wi-Fi*. Alguma vezes, essa questão até passa despercebida para o usuário (doméstico ou empresarial), resultando em locais sem cobertura de sinal. Boa parte dos ambientes corporativos e residenciais utilizam um ou mais pontos de acesso sem fio para prover acesso à rede local e dela para a internet. Entretanto, dispositivos móveis e computadores tipicamente não observam uma qualidade satisfatória de sinal do *access point wireless*. Esse problema pode ocorrer pelo fato do AP não abranger toda a área necessária ou sofrer atenuações devido às características do ambiente.

Ademais, redes Wi-Fi podem sofrer interferências externas, tais como telefones sem fio ou "babás"eletrônicas que operem na mesma faixa de frequência (banda ISM de 2.4 GHz), motores elétricos ou outras redes sem fio que estejam próximos do AP e irradiem EMI na mesma frequência do canal utilizado, seja ele na banda de 2.4 GHz (IEEE 802.11 b/g) ou 5.8 GHz (IEEE 802.11 a/ac). Por meio de uma análise do espectro de radiofrequência (RF) é possível detectar o nível dos sinais e/ou interferências em determinado local, entretanto, um problema facilmente observável que pode agravar a baixa qualidade observada no serviço da rede sem fio é a existência de pontos cegos na cobertura do sinal para determinado ambiente.

Como consequência de toda a atenuação e interferência sofrida, a rede *wireless* pode ficar inoperante ou dar a impressão de baixo desempenho. Por parte da maioria dos usuários finais, uma reclamação comum é a dificuldade de acessar determinado conteúdo e o acesso à rede parecer "lento". De um modo geral, por serem leigos no assunto, muitos usuários de rede sem fio acabam culpando o provedor de acesso à internet ou equipe de TI da instituição pela má qualidade do serviço quando, na verdade, a solução para o problema passaria por uma inspeção no espectro de sinal do(s) AP(s) *Wi-Fi* (para identificar interferências) e um melhor posicionamento deste(s) (para maximizar a cobertura do sinal e reduzir interferências entre APs).

Por fim, até onde pudemos verificar, não há disponível um *software* livre, gratuito e de código-fonte aberto para *Wireless AP Placement*, apenas soluções proprietárias, comerciais e custosas.

1.2 Objetivos

Após a contextualização do objeto de estudo deste trabalho e sua importância, sintetiza-se aqui seu objeto primário: projetar e implementar um *software* para planejamento do posicionamento dos APs (*Wireless AP Placement*), que receba como entrada

uma representação do ambiente e informações dos APs disponíveis, realize simulações de propagação dos sinais (considerando as características do ambiente) para os posicionamentos de APs propostos por meta-heurística que vise maximizar a cobertura do sinal Wi-Fi tendo, como caso de uso, as dependências do IFMG campus Formiga. São objetivos secundários e mais específicos os seguintes tópicos:

- Construir um modelo das dependências do IFMG campus Formiga, a partir da representação do ambiente fornecida ao software, e.g. planta(s)-baixa(s);
- Simular a propagação do sinal wireless de APs Wi-Fi através das dependências do campus Formiga, seguindo modelos de propagação de sinais sem fio;
- Utilizar metaheurística computacional para propor/verificar, via simulação, novas localizações para os APs Wi-Fi, visando maximizar a cobertura do sinal no campus (dentre outros fatores);
- Fornecer, ao final, uma solução proposta para o novo posicionamento dos APs Wi-Fi, se possível, auxiliada por algum método de visualização da cobertura do sinal wireless no ambiente.
- Disponibilizar o software de maneira livre e de código-fonte aberto (FOSS), distribuindo-o por meio de alguma licença que proteja a autoria do mesmo (e.g., Licença Pública Geral GNU — GPL).

2 FUNDAMENTAÇÃO TEÓRICA

No levantamento do referencial bibliográfico realizado, observamos um vasto número de trabalhos que tratam sobre a otimização do posicionamento de pontos de acesso *wireless*. Utilizaremos, como principais fundamentações teóricas deste projeto, o trabalho de Batitti, Brunato e Delai: *Optimal Wireless Access Point Placement for Location-Dependent Services* e a obra *Comunicações sem fio - Princípios e Práticas*, de Theodore S. Rappaport.

2.1 *Wireless AP Placement*

De acordo com o trabalho publicado por Battiti, Brunato e Delai (2003), "*Optimal Wireless Access Point Placement for Location-Dependent Services*", vários grupos de pesquisadores independentes têm proposto métodos para fazer estimativa a respeito da posição do usuário, com base na intensidade dos sinais de rádio recebidos de múltiplos APs (DALSSOTO, 2013; NAJNUDEL, 2004). Partindo desta distinta aplicação, os pesquisadores propõem uma nova abordagem para o AP *Placement*, pois consideram que a localização "é um importante parâmetro que pode ser usado para determinar o comportamento do sistema" (BATTITI, BRUNATO, DELAI, 2013, p. 1). Desse modo, a proposta deste TCC, que utilizará um modelo de propagação do sinal Wi-Fi em uma simulação do ambiente do IFMG *campus* Formiga é, mais uma vez, justificada pela necessidade de se ter um software livre e gratuito para *Wireless AP Placement*, que indique um melhor posicionamento dos *access points Wi-Fi*.

2.2 Tecnologias Wi-Fi (WLANs IEEE 802.11)

Quando se trata de redes *wireless*, é imprescindível dizer que, com o estabelecimento da família de padrões IEEE 802.11, se tornou possível a compatibilidade entre diferentes marcas de fabricantes de APs *wireless* e dispositivos móveis. De acordo com Franciscatti,

> As redes *wireless* utilizam freqüências de rádio para se comunicar havendo necessidade de uma padronização dos equipamentos sem fio por existir vários fabricantes. Não havia uma padronização dessa tecnologia causando, assim, a impossibilidade de comunicação de dispositivos de redes sem fio de outros fabricantes. Assim o *Institute of Electrical and Electronics [Engineers]* (IEEE) formou um grupo de trabalho com o objetivo de definir os padrões de uso em redes sem fio, denominado 802.11. (FRANCISCATTI, 2005 apud BOF, 2010, p. 14)

Desta forma, após a padronização definida pela IEEE, alguns padrões para WLAN foram estabelecidos (RIVERA, 2010; BANERJI, 2013), como:

- **IEEE 802.11a:** Definido em setembro 1999, operando na frequência de 5 GHz, uma largura de banda de 20 MHz, taxa de transmissão de até 54 Mbit/s e podendo ter um alcance de 35 metros indoor e até 5 quilômetros outdoor.
- **IEEE 802.11b:** Definido em setembro 1999, operando com uma frequência de 2.4 GHz, uma largura de banda de 22 MHz, taxa de transmissão de até 11 Mbit/s e com alcance de 35 metros indoor e 140 metros outdoor.
- **IEEE 802.11g:** Definido em junho de 2003, operando com uma frequência de 2.4 GHz, utilizando uma largura de banda de 20 MHz, com uma taxa de transmissão de até 54 Mbit/s e com um alcance de 38 metros indoor e 140 metros outdoor.
- **IEEE 802.11n:** Definido em outubro de 2009, operando com uma frequência de 2.4 GHz (802.11a) e 5 GHz (802.11g), com uma taxa de transmissão de até 72.2 Mbit/s utilizando uma largura de banda de 20 MHz e uma taxa de transmissão de até 150 Mbit/s com uma banda de 40 MHz, permitindo múltiplos fluxos espectrais (MIMO). O alcance vai de 70 metros indoor e 250 metros outdoor.
- **IEEE 802.11ac:** Definido em dezembro de 2013, operando com uma frequência de 5 GHz, utilizando uma largura de banda que vai de 20 MHz até 160 MHz, com uma taxa de transmissão de 87.6 Mbit/s até até 866.7 Mbit/s, também com uso de MIMO.

Considerando os padrões supracitados, iremos focar especificamente na penetração dos sinais *Wi-Fi* na estrutura dos edifícios (paredes, andares).

2.3 Propagação de sinais de rádio

O sinal Wi-Fi produzido pelo access point nada mais é, de um modo simplista, que uma onda de rádio que se propaga no espaço. De acordo com Rappaport,

> Os mecanismos por trás da propagação da onda eletromagnética são diversos, mas geralmente podem ser atribuídos a reflexão, difração e dispersão. (...) Devido a múltiplas reflexões de vários objetos, as ondas eletromagnéticas trafegam por diferentes caminhos de tamanhos variados. A interação entre essas ondas causa uma distorção de caminhos múltiplos em um local específico, e as intensidades das ondas diminui à medida que a distância entre transmissor e receptor aumenta. (RAPPAPORT, 2009, p. 72)

Por ser uma onda eletromagnética, a onda do sinal *Wi-Fi* como descrito por Torlak, apresenta tal comportamento, sofrendo reflexão, difração e dispersão. Outro aspecto que deve ser abordado acerca da onda do sinal *Wi-Fi* é a diferença entre modelos de propagação de larga escala (perdas) e pequena escala (atenuação). Os modelos de propagação em larga

escala são caracterizados pela intensidade do sinal para grandes distâncias de separação do transmissor e receptor (podendo variar de várias centenas ou milhares de metros), enquanto modelos de propagação em pequena escala são caracterizados pelas flutuações rápidas do sinal recebidos para distâncias muito curtas (variando de alguns comprimentos de onda ou durações de segundos) (Cf. RAPPAPORT, 2009, p. 72).

Rappaport define três modelos básicos de propagação que são usados para a previsão da intensidade do sinal recebido a determinada distância do transmissor, numa larga escala. O modelo de propagação no espaço livre (Friss) é utilizado quando transmissor e receptor possuem uma linha de visão desobstruída, ou seja, não há obstáculos entre eles que interrompam ou alterem o caminho da transmissão do sinal. O modelo de propagação no espaço livre oferece uma noção da ordem de magnitude do sinal recebido, mas é demasiado otimista pois raramente há um único caminho entre a antena transmissora e a antena receptora: em situações reais, haverá reflexão do sinal no solo. Para grandes distâncias e antenas altas, o modelo de reflexão no solo é razoavelmente preciso para prever a intensidade do sinal recebido. Esse modelo é baseado na ótica geométrica e considera o caminho direto e o caminho refletido (modelo de dois raios), que muitas vezes é no solo. E, por fim, temos o modelo de difração (por gume de faca) que torna possível propagar os sinais de rádio através de obstruções, bem como ao redor da superfície da terra, além do horizonte. Contudo, a força do campo recebido diminui rapidamente quando o receptor se aproxima do obstáculo em direção à região obstruída (sombra), porém, o campo de difração ainda existe e normalmente tem força suficiente para produzir um sinal útil (Cf. RAPPAPORT, 2009, p. 72-83).

A propagação no interior é dominada pelos mesmos mecanismos (reflexão, difração e dispersão), porém as condições são muito variáveis, podendo variar, por exemplo, até mesmo se as portas e janelas estiverem fechadas ou dependendo do local onde as antenas são montadas. Dentro do mesmo contexto, existem várias características físicas e elétricas que podem influenciar na propagação do sinal, como qual o tipo de ambiente (escritório ou casa) e de que é feita a construção (madeira, tijolo, concreto ou até ferragem). Durante a propagação, pode também ocorrer a perda do sinal entre andares de um edifício, obedecendo à lei de potência da distância, a qual leva em consideração o tipo de prédio, arredores e uma variável aleatória normal que representa o desvio padrão (Cf. RAPPAPORT, 2009, p. 104-108).

2.4 Modelos de propagação

Quando se deseja realizar um bom desempenho e planejamento da cobertura do espectro Wi-Fi, é indispensável o conhecimento do meio de transmissão e qual modelo se deve usar para obter um resultado mais realista. Em sistemas *wireless* o meio de propagação

utilizado é o canal de rádio, de forma que as características e efeitos sobre todas as informações trafegadas são de uma natureza complexa, fazendo com que medições empíricas sejam de suma importância. Com as medições é possível ver como é o comportamento de modelos em pequena e larga escala, além de ser possível determinar a variação da potência do sinal devido ao movimento de pessoas no ambiente ou atingir obstáculos fixos, como paredes, pisos, vidros, móveis, dentre outros.

Então, se faz necessário uma boa escolha de quais modelos de propagação utilizar quando se quer obter uma boa representação do espectro e que ao mesmo tempo esteja condizente com a realidade. Quanto maior for a precisão desejada, mais detalhes sobre o ambiente de propagação devem ser modelados. Nesta seção serão comentados sucintamente sete modelos que foram estudados para a realização deste trabalho.

2.4.1 Propagação no espaço livre (modelo de Friis)

O modelo *Friss free-space path loss* ou geralmente tratado como modelo de propagação no espaço livre de Friis é usado para prever a intensidade do sinal recebido quando a antena transmissora e a antena receptora possuem um caminho de linha de visão limpo, ou seja, um caminho desobstruído de qualquer objeto ou edificação (Luo, 2003). Tal modelo é em geral usado em sistemas de comunicação por satélite e em enlaces de rádio de microondas com linha de visão. Assim como os modelos outdoor, o modelo de propagação no espaço livre de Friis, pressupõe que a potência recebida diminui com uma função da distância entre a antena transmissora e a antena receptora elevada a alguma potência, ou seja, uma função da lei de potência. A potência recebida através do espaço livre pela antena receptora separada da antena transmissora por uma distância d pode ser calculada pela seguinte equação:

$$P_r = \frac{P_t G_t G_r \lambda^2}{(4\pi)^2 d^2 L} \tag{2.1}$$

sendo:

- Pt a potência transmitida;
- Pr(d) a potência recebida;
- Gt é o ganho da antena transmissora;
- Gr é o ganho da antena receptora;
- d é a distância de separação das antenas;
- L é o fator de perda (L>=1), L = 1 indica nenhuma perda no hardware do sistema

- λ é o comprimento de onda dado em metros.

O modelo de espaço livre de Friis é apenas uma previsão válida para uma potência recebida para uma distância d de separação entre as antenas. O campo distante que é criado entre as antenas pode ser chamado de região de *Fraunhofer*, onde essa região é definida como uma região além da distância de campo distante df, que está diretamente relacionada com a maior dimensão linear D de abertura da antena transmissora e com o comprimento de onda da portadora (antena receptora). Tal distância de *Fraunhofer* pode ser calculada pela seguinte fórmula:

$$d_f = \frac{2D^2}{\lambda} \tag{2.2}$$

O cálculo da potência recebida tem uma falha quando a distância é zero. Por este motivo, modelos de propagação em larga escala utilizam uma distância próxima, d0, com um ponto de referência de potência conhecido. A potência recebida, Pr(d), em qualquer distância que a distância d seja maior que zero, pode estar relacionada com a potência recebida no ponto de referência d0. A equação que calcula a potência recebida pode ser vista abaixo utilizando uma distância maior que d0:

$$P_r = P_r(d_0)(\frac{d_0}{d})^2 \tag{2.3}$$

O valor da distância de referência d0 em sistemas práticos em antenas de baixo ganho, entre 1 e 2 GHz, normalmente é utilizado com sendo 1 metro em ambientes internos (*indoor*) e 100 metros ou 1 quilômetro para ambientes externos (*outdoor*), de forma que o resultado obtido pelas equações anteriores são múltiplos de 10, tornando os cálculos de perda de caminho fácies em unidade de dB.

2.4.2 *Two-rays ground reflection*

O modelo *Two-rays ground reflection* é um modelo de propagação de rádio que prevê as perdas de trajetória entre uma antena transmissora e uma antena receptora. Em geral, as duas antenas têm altura diferente e raramente possuem uma linha direta. O sinal é recebido de duas formas: por LOS (linha de visão) e o por *multipath* (multicaminho) formado predominantemente por uma única onda refletida no solo.

Quando a distância entre as antenas é menor do que a altura da antena transmissora, duas ondas são adicionadas de forma positiva para gerar maior potência e, à medida que a distância aumenta, essas ondas se somam de forma construtiva e destrutiva, proporcionando regiões de exaustão e decaimento à medida que a distância aumenta além da distância crítica ou primeira zona de Fresnel, a potência cai proporcionalmente quatro vezes o

inverso da potência da distância. Essa é uma perda no caminho muito mais rápida do que é experimentada no *Friss free-space path loss model* (espaço livre de Friis).

Quando se tem valores muito altos para a distância, pode-se notar que a potência recebida e a perda no caminho se tornam independentes da frequência (RAPPAPORT, 2009). O modelo de reflexão de dois raios é uma formulação matemática de um tipo de interferência *multipath* quando a interferência é considerada como consistindo em dois caminhos: **(I)** do transmissor ao receptor diretamente, **(II)** do transmissor, refletido fora do chão, para o receptor.

A potência recebida a uma distância d do transmissor para o modelo *Two-rays ground reflection model* pode ser expressa como:

$$P_r = P_t G_t G_r \frac{h_t^2 h_r^2}{d^4} \tag{2.4}$$

sendo:

- Pr a potência recebida;
- Pt a potência transmitida;
- Gr o ganho da antena receptora;
- Gt o ganho da antena transmissora;
- ht a altura da antena transmissora;
- hr a altura da antena receptora;
- d a distância do transmissor;

A perda do caminho para o modelo *Two-rays ground reflection model* pode ser expressa em dB com a equação abaixo:

$$PL(dB) = 40logd - (10logG_t + 10logG_r + 20logh_t + 20logh_r) \tag{2.5}$$

2.4.3 *Log-distance*

O modelo *Log distance path loss* é um modelo genérico e uma extensão do modelo de espaço livre de Friis. Ele é usado para prever a perda de propagação para uma ampla gama de ambientes, enquanto que o modelo Friis é restrito ao caminho desobstruído entre o transmissor e o receptor.

O principal critério ou característica deste modelo é considerar que a perda no caminho é logaritmicamente dependente da distância. Logo a perda no caminho calculada

com uma distância d entre um transmissor e receptor (geralmente dado em quilômetros) . Na região mais distante do transmissor (onde $d \geq df$), se PL (d0) é a perda de percurso medida em dB a uma distância d0 do transmissor, então a perda do caminho (a perda na potência do sinal em dB quando se desloca de distância d0 a d) a uma distância arbitrária $d > d0$ é dada pela fórmula:

$$PL(d) = PL(d_0) + 10 \cdot n \cdot log(\frac{d_0}{d}) \tag{2.6}$$

sendo:

- d é a distância dada em quilômetros em todos os caso;
- d0 é a distância inicial de referência;
- PL(d) é a perda no caminho para a distância d;
- PL(d0) é um valor de perda de caminho para uma distância de referência;
- n é o expoente de propagação e indica a taxa na qual a perda de caminho aumenta com a distância (MATHURANATHAN, 2013).

Geralmente, para modelar ambientes reais, os efeitos *shadowing* (de sombreamento) não podem ser negligenciados. Se os efeitos de sombreamento forem deixados de lado, a perda do caminho, quando representada em um gráfico que representa a potência recebida e a distância, é simplesmente uma linha reta. Para adicionar um efeito de sombreamento e deixar o resultado final mais próximo da realizada, uma variável aleatória Gaussiana de média zero com desvio padrão σ é adicionada à equação. A perda real do caminho ainda pode variar devido a outros fatores, assim como os efeitos de reflexão, difração e dispersão. Assim, o expoente da perda de caminho (a literatura define alguns em ambientes diferentes) e o desvio padrão da variável aleatória escolhida, devem ser bem conhecidos para uma boa modelagem da perda.

2.4.4 *One-slope*

O modelo *One-slope* é classificado como sendo um modelo empírico e assume que a perda no caminho dada em dBm é linearmente na distância logarítmica da distância d entre o transmissor e receptor:

$$PL(d) = L_0 + 10 \cdot n \cdot log(d) \tag{2.7}$$

onde:

- d é a distância entre transmissor e receptor;
- PL(d) é a perda no caminho para a distância d;
- L0 é a perda no caminho calculado em uma distância de 1 metro;
- n é o expoente de perda no caminho;

Claramente, este modelo baseia-se na perda do espaço livre e visa incluir todas as perdas devido a vários mecanismos de propagação pelo caminho usando um expoente n de perda. Por ser um modelo simples, se torna muito fácil de realizar sua implementação, mas se usado de forma única pode levar a grandes erros em ambientes internos, pois é possível que um grande números de objetos interfiram nos mecanismos de propagação, ou seja, não será possível obter um resultado com uma precisão próxima da realidade.

2.4.5 *Wall and floor factor*

O modelo *Wall and floor factor* leva em conta a absorção em paredes e pavimentos e geralmente é usado em ambientes internos e considera a perda no espaço livre. Ele se resume basicamente na perda no espaço livre de um ambiente interno somado com uma perda adicional relacionada a cada piso atravessado e o número de paredes interceptadas em uma linha direta entre o transmissor e receptor.

Segundo Xie, este modelo de propagação tem um desempenho melhor que o modelo *One-slope*, uma vez que proporciona mais graus de liberdade na consideração de obstáculos (Xie, 2013). A perda no caminho utilizando o modelo de propagação *wall and floor factor* pode ser calculada pela seguinte equação:

$$PL(d) = a + 10 \cdot n \cdot log(d) \tag{2.8}$$

sendo:

- d é a distância entre transmissor e receptor;
- PL(d) é a perda no caminho para a distância d;
- L1 é a perda no caminho calculado em uma distância de 1 metro; n é o expoente de perda no caminho;
- Lf e Lw são respectivamente perdas causadas pela penetração no piso e nas paredes;
- nf e nw são respectivamente os números de pisos e de paredes.

2.4.6 *Log-normal fading*

Este modelo de propagação estatístico provê uma estimativa aproximada do que representa a perda no caminho como uma função da distância e outros parâmetros como a frequência e a altura da antena. Desta forma é possível obter uma previsão da qualidade da intensidade do sinal quando aumentamos a distância entre o transmissor e receptor. Vale ressaltar que, para se obter uma representação mais realista da perda do sinal, é necessária uma grande coleta de dados empíricos.

$$f(x;\mu;\sigma) = \frac{1}{x\sigma\sqrt{2\pi}} exp[-\frac{(ln(x)-\mu)^2}{2\sigma^2}] \tag{2.9}$$

Uma possível explicação para o motivo pelo qual este modelo utiliza uma distribuição *log-normal* é que, para cada caminho há muitos fatores que contribuem com a perda do sinal, incluindo a combinação de perda no espaço livre, difrações, reflexões, interferências de equipamentos, dentre outros motivos. Para cada uma dessas perdas, é utilizada uma variável aleatória que a representa (BUDGETS, 2013). A sua perda é dada em dB e é o somatório de todas essas perdas (também expressadas em dB). O teorema do limite central afirma que tal distribuição tenderá a uma distribuição normal.

2.4.7 *Ray-tracing fading*

Segundo Valenzuela (1993), para esse modelo de propagação é realizado o traçado de raios em todas as direções possíveis do receptor ao transmissor, utilizando os princípios da óptica geométrica. Este mesmo conceito é utilizado em sistemas de realidade virtual para que possam se tornar visíveis todos os objetos dentro de um ambiente. Nos cálculos da simulação do ambiente são levadas em consideração a reflexão, difração, espalhamento do sinal, dentre outros fenômenos físicos.

Valenzuela também afirma que o modelo de propagação *Ray Tracing* assume que todos os objetos no ambiente de propagação são objetos refletores em potencial. Para isso, considera também somente os caminhos que realmente existem entre o transmissor e receptor. A complexidade do ambiente escolhido tem um forte impacto em seu consumo de recurso computacional, uma vez que, quanto mais obstáculos forem adicionados, mais reflexões, difrações e cálculos serão feitos.

2.5 Trabalhos Relacionados

Nesta seção serão apresentados alguns softwares que estão presentes no mercado e oferecem serviços similares aos objetivos propostos nesse trabalho. Serão citadas ferramentas que realizam um trabalho semelhante ao desenvolvido aqui, como o *TamoGraph Site Survey, AirMagnet Survey, Ekahau Site Survey, D-Link Wi-Fi Planner e Xirrus Wi-FI Designer.*

Existem várias outras ferramentas que trabalham no mesmo meio, porém, os softwares apresentados a seguir possuem conhecimentos aplicados nesta pesquisa e na área da Ciência da Computação.

2.5.1 TamoGraph Site Survey

O *TamoGraph*[1] é uma ferramenta de site survey usada para a coleta, visualização e análise de dados *Wi-Fi* 802.11 com padrões a/bg/n/ac. É muito usada quando se tem a implantação e a manutenção de redes sem fio, uma vez que facilita tarefas que são demoradas e muitas das vezes, até complexas de se obter um bom resultado. O *TamoGraph* realiza como tarefas, análises contínuas e relatórios de intensidade/qualidade do sinal, ruídos e interferências, alocações de canais, taxas de dados transmitidos, dentre outros.

A ferramenta aposta que, as empresas que fizerem o seu uso, poderão reduzir drasticamente o seu tempo e custos envolvidos em implantações e manutenções de redes *wireless*, melhorar o desempenho e cobertura da rede em todos os tipos de ambiente, desde ambientes *indoor*, como prédios com escritórios, aeroportos e shoppings, até ambientes *outdoor*, como pátios, praças, campos e estacionamentos.

A empresa *TamoSoft* considera ser praticamente impossível considerar todas as variáveis que possam afetar a saúde e o desempenho da rede. Para ela, alterar as condições, até mesmo de algo aparentemente menor, como um *notebook* conectado à rede sem fio de um escritório, pode afetar gravemente o seu desempenho, que também pode ser influenciado pela ampla proliferação de redes sem fio com fatores de interferência. Por estes fatores, a ferramenta da *TamoSoft* pode ser considerada como profissional e de essencial uso para empresas, inclusive para usuários comuns.

O *TamoGraph* pode ser executado em *Microsoft Windows 7, Windows 8, Windows 8.1, Windows 10, Windows Server 2008 R2, Windows Server 2012, Windows Server 2012 R2* e versão para *MacBooks*. Possui versões em 32 e 64 bits e requer um adaptador de rede *wireless* compatível. A sua licença mais básica custa 899 dólares, aproximadamente R$2850,00 e a licença profissional custa US$1199.00, aproximadamente R$ 3802,00.

2.5.2 Netscout AirMagnet Survey

O *AirMagnet Survey*[2] é um software de pesquisa local para redes sem fio que propõe uma solução para *softwares* que realizam a análise de redes sem fio locais que necessitam projetar e planejar LANs sem fio com o padrão 802.11 a/b/g/n/ac com desempenho, segurança e conformidade. O software calcula a quantidade, a alocação e configurações ideais para a realização de uma rede local *wireless* com um bom desempenho.

[1] <http://www.tamos.com/products/wifi-site-survey/>
[2] <http://enterprise-pt.netscout.com/products/airmagnet-survey>

A *Netscout*, empresa responsável pelo *AirMagnet*, diz que o seu produto vai além do que uma simples cobertura dos sinais de radiofrequência. Ele traça o desempenho de rede real do usuário final nos termos da velocidade de conexão, taxa de transferência e estatísticas do pacote e dá, como resultado, um mapa completo do ambiente coberto pelo Wi-Fi, permitindo ao usuário implantar sua rede corretamente, já no primeiro momento, evitando custos de retrabalho e reclamações posteriores, além de ter fidelidade nos serviços prestados.

O *AirMagnet* permite que os usuários possam integrar analisadores de espectro profissionais para obter os dados do sinal *wireless* em uma única varredura, modelar cenários antes da implantação para estimar orçamentos, definir estratégias de migração para novas tecnologias, obter relatórios de pesquisa personalizados, executar inspeções internas usando dispositivos providos de tecnologia GPS, realizar inspeções de *VoiceOver* no local de implantação da rede *wireless* (para que esteja pronta para suportar serviços de voz), certificar a rede para os requisitos de aplicativos e rede dos usuários finais, com um planejamento detalhado da capacidade de usuários finais.

O *AirMagnet Survey* possui versões *Express* que oferecem uma versão mais simples de pesquisa local para padrão 802.11ac, permitindo que o usuário execute um exame básico do local de implantação da rede *wireless*, possibilitando mapear o sinal, ruído e até mesmo o desempenho de usuários. O *AirMagnet* possui também a sua versão Pro, que amplia ainda mais as capacidades oferecidas pela versão *Express*. Nela é adicionada a funcionalidade "*Planner*", pela qual é possível realizar desde a implantação de um *access point* até o orçamento dos gastos, além de suporte para a implantação de vários andares, inspeções técnicas de ambientes externos (outdoor), verificação e análise de prontidão para serviços de voz, análise do espectro de radiofrequência e mais outros recursos.

É possível executar o *software* no ambiente *Windows* em todas as versões 64 bits iguais ou superiores ao *Windows 7* e em ambientes *OSX* em que a versão é igual ou superior ao *Mac OS X v 10.5 (Leopard)*. O preço da licença para o uso do software não é informado no site da empresa. O orçamento deve ser feito pelo contato com representantes.

2.5.3 Ekahau Site Survey & Planner

A *Ekahau Wireless Design* é mais uma empresa que fornece um software para soluções sobre redes sem fio, batizado de *Ekahau Site Survey (ESS)*[3]. O *Ekahau Site Survey* propõe um design e análise experiente sobre a tecnologia Wi-Fi. A *Ekahau* define seu software como sendo um conjunto completo de ferramentas para projetar, analisar, otimizar e solucionar problemas de redes *wireless*.

O ESS não deixa de ser um instrumento para verificação e solução de uso fácil

[3] <https://www.ekahau.com/products/ekahau-site-survey/overview/>

em redes Wi-Fi. Foi desenvolvido para engenheiros e arquitetos de redes sem fio (desde sistemas integrados até administradores da área de TI). A empresa ainda garante o alto desempenho e capacidade para qualquer rede Wi-Fi com padrões 802.11ac e n. Se caso uma rede ainda não esteja em seu desempenho ótimo ou ainda não foi implantada, o ESS irá sugerir automaticamente o devido posicionamento e as configurações ideais para o *access point*.

Com a ferramenta, também é possível criar automaticamente um plano da rede Wi-Fi de vários andares com base em requisitos de desempenho e capacidades especificados. Quase que de imediato, o ESS irá identificar o número ideal de *access points*, com os melhores locais para seus posicionamentos e seus respectivos canais, simulando o comportamento de como a rede irá ser executada antes de ir para o local. Com sua funcionalidade "*3D Planner*", é considerado o espalhamento do sinal entre os andares do prédio para ajudar a minimizar a interferência dos canais.

Com o ESS é fácil realizar a análise em profundidade com mapas de calor. Nos mapas de calor é possível visualizar, por exemplo, a força do sinal, a taxa de dados, perda de pacotes, a sobreposição de canais, espalhamento do espectro, dentre outras características. É possível também realizar a análise de capacidade mais abrangentes, por exemplo, todas as descobertas podem ser compiladas em um simples relatório utilizando um sistema de relatório desenvolvido pela *Ekahau*.

O software foi projetado para ser executado em ambientes *Windows* e *MaxOS*, e é uma ferramenta caracterizada por ser, de fato, essencial para engenheiros de redes sem fio em empresas de todos os tamanhos. A sua licença *Standard* custa US$2295.00, aproximadamente R$7278,00 e vai até sua versão *Premium Pack* custando US$5649.00, aproximadamente R$17914,00; a versão *Pro Pack* com o valor de US$5995.00, aproximadamente R$19011,00s.

2.5.4 D-Link Wi-Fi Planner PRO

O *D-Link Wi-Fi Planner PRO*, como indicado pelo nome, foi desenvolvido pela famosa empresa D-link. Com essa ferramenta, é possível ter uma visão do ambiente como um todo, antes da implantação da rede Wi-Fi. Isso faz com que o planejamento, a comunicação e à boa qualidade do serviço prestado entre WLAN e clientes sejam melhores.

Para a execução do software é necessário criar uma pasta para o projeto. Logo após, o programa pede para ser carregado uma imagem que represente a planta do ambiente. A planta não tem a necessidade de ser exatamente igual, apenas é necessária uma imagem que possa ser o rascunho inicial para planta. Em seguida deve ser informada a escala, para estimar a medida da planta.

Para que a execução do programa seja possível, as zonas de coberturas e as zonas

de exclusão tem de ser definidas. O próprio usuário pode marcar os obstáculos (portas e paredes) e indicar as zonas especiais como sendo espaços fechados para salas e escritórios. Isso faz com que o WFP tenha uma simulação mais precisa e próxima da realidade. Depois de feitas as configurações, um módulo chamado *AP Placement Advisor* irá fornecer uma sugestão sobre o número de *access points* e o posicionamento necessários para eles terem uma maior cobertura do local.

A empresa não fornece mais informações sobre quais os requisitos mínimos necessários para a utilização de sua ferramenta.

3 MATERIAIS E MÉTODOS

Serão apresentados neste capítulo os materiais e métodos utilizados para o desenvolvimento desse projeto, tais como as bibliotecas do *Python*, o processo com o arquivo de entrada do *AutoCad*[1] para a representação da matriz de propagação, a paralelização do código, utilizando a GPU para um melhor desempenho da simulação, a metaheurística utilizada para a otimização do posicionamento de *access points* e o Projeto Fatorial 2K para a calibração dos parâmetros da metaheurística.

Para definir qual modelo de propagação deveria ser utilizado, foram usados dois softwares: o *R-Project*[2] e o *Maple*[3] para a realização de ajuste de curvas dos dados coletados empiricamente nos corredores do instituto.

3.1 A Metaheurística

Dada a complexidade do problema e o tamanho do ambiente simulado, para que fosse viável sugerir boas posições para a alocação do(s) *access point(s)*, foi implementado o *Simulated Annealing* como metaheurística de otimização que teve como função objetivo a busca por uma melhor cobertura do sinal wireless.

O *Simulated Annealing* é uma metaheurística para aproximar a otimização global em uma amplo espaço de busca. Este método foi proposto por Scoot Kirkpatrick em 1983 e foi utilizado para simular o processo de recozimento de metais cujo resfriamento rápido levava a produtos metaestáveis, ou seja, de maior energia interna e o esfriamento lento a produtos mais estáveis, estruturalmente fortes e de menor energia. Durante o recozimento, o material passa por vários estados possíveis com um tempo suficientemente longo para que qualquer elemento passe por todos os seus estados acessíveis.

O *Simulated Annealing* realiza o processo de otimização buscando encontrar a melhor solução viável, considerando o objetivo do problema em questão, e o conjunto de restrições para aceitação da solução proposta.

Problemas no campo das heurísticas podem ser modelados como problemas de maximização e problemas de minimização de uma função objetivo, que neste caso é obter a maior cobertura e qualidade do sinal *wireless*.

Com um problema de otimização em mãos, encontrar soluções ótimas ou aproximadas do seu ótimo para problemas NP-difíceis é um desafio nem sempre fácil de ser

[1] <https://www.autodesk.com.br/products/autocad/overview>
[2] <https://www.r-project.org/>
[3] <https://www.maplesoft.com/products/Maple/>

alcançado. O uso de heurística para auxiliar na busca por um lugar para o *access point* foi de fácil implementação e, como a maioria das heurísticas, produz boas soluções dentro de um tempo viável de acordo com os parâmetros estabelecidos.

Abaixo apresentamos o pseudocódigo da metaheurística com funções genéricas que foram implementadas neste trabalho:

[inserir pseudo código]

Estes são os identificadores utilizados:

- S0: Configuração Inicial (Entrada);
- Si: Configuração da Iteração i;
- S: Configuração Final;
- T0: Temperatura Inicial;
- Ti: Temperatura na Iteração i;
- M: Número máximo de iterações (Entrada);
- P: Número máximo de Perturbações por iteração (Entrada);
- L: Número máximo de sucessos por iteração (Entrada);
- α: Fator de redução da temperatura (Entrada);
- f(Si): Valor da função objetivo correspondente á configuração Si;
- nSucesso: Contador de sucesso em uma iteração;
- i e j: Variáveis de controle de Loops.

Além dos indicadores acima, consideremos as seguintes funções:

- Perturba(S): Função que realiza uma perturbação na Solução S;
- Randomiza(): Função que gera um número aleatório no intervalo [0,1];
- TempInicial(): Função que calcula a temperatura inicial;

Mais adiante, será descrito como a metaheurística foi adaptada para o problema de otimização para alocação de *access point* nos ambientes do Instituto. Além disso, também serão apresentados técnicas e resultados de sua utilização.

3.2 Linguagem Python

A linguagem que foi utilizada para implementar todo o trabalho foi o Python. A linguagem Python é uma poderosa linguagem de programação e de fácil aprendizado. Surgiu no final dos anos 80 e foi criada por Guido Van Rossum. Na época, Guido trabalhava no Centro de Matemática e Ciência da Computação de Amsterdã, Holanda, no desenvolvimento de outras linguagens, quando percebeu que o desenvolvimento de utilitários para o sistema operacional Amoeba (projeto em que também trabalhava na época) utilizando a linguagem C estava tomando muito tempo e fazê-los em *Shell Scrip*t não era viável. Precisava de algo que completasse o que cada linguagem deixava a desejar. Esses foram o principais motivos que fizeram com que o desenvolvimento do Python realmente tivesse início no ano de 1989 e nos primeiros meses de 1990 fosse a linguagem mais utilizada no departamento de Computação de Amsterdã e hoje por muitos desenvolvedores de software.

O python é uma linguagem interpretada. Isso significa que seu código é executado por um interpretador, e não compilado para linguagem de máquina para depois ser executada para um sistema e arquitetura específica, como acontece em algumas linguagens, como por exemplo a linguagem C.

Não obstante, o Python não trabalha com tipagem de objetos, o que permite, no geral, um ótimo desempenho. Alguns processamentos que realizam a demanda de mais recursos, como o processamento de imagens, são feitos por bibliotecas que geralmente são escritas em C ou C++, inclusive com possíveis trechos em *assemble*, quando o alto desempenho é necessário. Sendo assim, tais processamentos não fazem uso de tantos recursos num *script* escrito em Python. Em outra linguagem compilada seriam exigidos mais recursos.

Mesmo que o Python faça uso bem definido do tipo de dados que está manipulando, ele trabalha com tipagem dinâmica. Isso implica que uma variável possuirá características de um tipo específico de sua declaração, até ser declarada novamente. Pode-se ver então que, com o uso da tipagem dinâmica, são geradas flexibilidade e simplicidade no código de funções e classes do Python, reduzindo significativamente a quantidade de parâmetros em uma função.

A separação de blocos no Python não é feita com colchetes, chaves ou *begins* e *ends*. Toda a separação dos blocos é feita por tabulações. Isso força sempre o programador a manter o código mais organizado, além de reduzir consideravelmente o tamanho do código. Caso o programador que escreveu o código não mantenha a indentação, um erro de indentação é mostrado e ele não é executado.

Além dos tipos primitivos como *int*, *float* e *boolean*, também estão presentes no Python tipos especiais como listas, sendo listas bem definidas entre colchetes e elementos

mutáveis separados por vírgulas; tuplas sendo agora elementos imutáveis definidos entre parênteses, separados por vírgula e dicionários definidos entre chaves com tipos imutáveis variados. O conceito de Orientação a Objetos também está presente no Python, no qual todas as variáveis são definidas como objeto, até mesmo os tipos primitivos. Todos esses tipos especiais enriquecem ainda mais o poder que a linguagem tem. O que em outra linguagem levaria tempo e linhas de código para ser feito, no Python é feito com simplicidade.

3.3 Bibliotecas utilizadas

Uma das vantagens de se usar o Python, é seu vasto número de bibliotecas embutidas e bibliotecas externas, as quais dão diversas facilidades ao resolver problemas simples do dia a dia até problemas mais complexos. Para o desenvolvimento do projeto foram utilizadas várias bibliotecas que estão embutidas no Python, que vieram complementar outras bibliotecas externas essenciais para o resultado esperado fosse obtido.

De início, foi utilizada a biblioteca *ezdxf*[4]. Tal biblioteca é um pacote do Python criado para manipular arquivos DXFs exportados do independentemente de sua versão. Com ela, é possível criar, abrir, salvar e modificar arquivos com a extensão *.dxf* sem perder qualquer informação, podendo depois, ser utilizada em outros programas que utilizam a mesma extensão de arquivo. Tal biblioteca está presente desde o Python 2.7.

Outra biblioteca utilizada no projeto é a PyGame[5]. Seu desenvolvimento teve início no ano 2000 por Pete Shinners, sendo um código livre e é principalmente usada em aplicações que utilizam recursos multimídia e/ou programação gráfica, como em desenvolvimento de jogos multiplataforma (independentemente do sistema operacional).

3.4 Programação Paralela

A computação paralela realizada por placas de vídeo é o uso de uma GPU juntamente com uma CPU para acelerar aplicações que necessitam de um alto recurso computacional. Desde 2007, quando a NVIDIA criou o conceito de 'computação acelerada', o uso da GPU vem potencializando data centers de eficiência energética em laboratórios governamentais, universidades, corporações e empresas de médio e grande portes em todo o mundo. Elas desempenham um papel fundamental na aceleração de aplicações em plataformas que variam de inteligência artificial até carros, drones e robôs.

A seguir serão descritos algum conceitos e ferramentas que foram utilizadas para o desenvolvimento deste trabalho. Todas tiveram um papel essencial para que os objetivos

[4] <https://pypi.python.org/pypi/ezdxf>
[5] <https://www.pygame.org/>

propostos fossem alcançados.

3.4.1 CUDA

CUDA[6] é um acrônimo de *Compute Unified Device Architecture*. É uma plataforma de computação paralela e um modelo de programação inventados pela NVIDIA. Tal plataforma faz com que as aplicações tenham aumentos significativos de desempenho computacional ao aproveitar a potência e os recursos da GPU[7]. Hoje há milhões de GPUs habilitadas para que trabalhem juntamente com CUDA. Muitos cientistas e pesquisadores vêm descobrindo inúmeras possibilidades para o uso da computação com GPU CUDA. Dentre as possíveis aplicações estão: a identificação de placas ocultas em artérias em pacientes com problemas de ataque cardíaco, análise do fluxo de tráfego aéreo junto ao *National Airspace System* (Sistema de Espaço Aéreo Nacional), aumento de desempenho para a simulação de visualização de moléculas.

Há algum tempo era complicado escrever códigos para que fossem executados utilizando a GPU. Todo o assunto sobre programação paralela em GPU era focado em processamento de imagens; hoje esse conceito já não é o mesmo. A placa de vídeo faz mais do que realizar o processamento de imagens, ela lida com um teraflop de desempenho de ponto flutuante e processa tarefas que vão de problemas de finanças à medicina.

Para que seja possível escrever um código que busque aproveitar o desempenho oferecido pelo *hardware* juntamente com a CUDA, será necessário utilizar o CUDA Toolkit, que oferece um ambiente de desenvolvimento abrangente para desenvolvedores C, C#, C++, Fortran, R e Python. O CUDA Toolkit inclui compilador, bibliotecas de matemática e ferramentas para depuração e otimização do desempenho do código.

Hoje a NVIDIA oferece tudo gratuitamente. Tanto a NVIDIA quando a CUDA só tendem a crescer, à medida que cada vez mais empresas fornecem ferramentas, serviços e soluções. Além disso, áreas como a ciência e a medicina podem crescer ainda mais com suas pesquisas.

3.4.2 CUDA Tool Kit

O CUDA Toolkit[8] nada mais é que um kit de ferramenta disponibilizado pela NVIDIA. O kit fornece um ambiente de desenvolvimento para a criação de aplicações a serem otimizadas em uma GPU. Com o kit de ferramentas é possível otimizar e implementar códigos escritos utilizando CUDA em sistemas que aceitam a paralelização com GPU, *workstations*, centro de dados empresariais, plataforma baseadas em nuvem e supercomputadores HPC. No pacote vem incluso bibliotecas otimizadas para GPU,

[6] <http://www.nvidia.com.br/object/cuda_home_new_br.html>
[7] <http://www.nvidia.com.br/object/what-is-gpu-computing-br.html>
[8] <https://developer.nvidia.com/cuda-toolkit>

ferramentas para depuração e otimização, um compilador C/C++ e uma biblioteca de tempo para ser usada nas aplicações que serão desenvolvidas.

As bibliotecas otimizadas possibilita a otimização de vários tipos de aplicação, como por exemplo, aplicações que realizam cálculos matemáticos utilizando conceitos de álgebra linear, processamento de imagem e de vídeo, inteligência artificial e até análise de grafos. O que torna possível desenvolver aplicações personalizadas para diferentes linguagens é o uso da API. A API fornece um controle mais seguro e objetivo, especialmente sobre o contexto ou módulo que a aplicação está sendo desenvolvida. Uma chamada feita diretamente ao *kernel* é muito mais complexa de ser implementada, pois a configuração de execução e os parâmetros do *kernel* devem ser especificados com chamadas de funções muito bem explícitas. Outra grande vantagem da API é que seu uso é independente da linguagem.

A versão utilizada neste trabalho foi a CUDA Tool Kit 9.0 e foi o que tornou possível a otimização do algoritmo na sua versão final. O CUDA Tool Kit foi instalado juntamente com os *drivers* e é facilmente encontrado no site de desenvolvedores da NVIDIA.

3.4.3 Numba

O Numba[9] é um compilador para *arrays* e funções numéricas em Python que proporciona a aceleração de módulos do código, escritos diretamente na linguagem Python. O Numba realiza a geração de código de máquina otimizado a partir do código escrito em Python puro, utilizando a infra-estrutura do compilador LLVM (instalado no momento da instalação do CUDA). Com algumas anotações simples nos cabeçalhos dos métodos, o código escrito utilizando vetores e matrizes pode ser otimizado em *just-in-time* com um desempenho semelhante ao C, C++ e Fortran, sem ter que alterar a linguagem de programação ou interpretador do Python.

Os principais pontos positivos da utilização do Numba são:

- geração de código *on-the-fly* (geração de código em tempo de execução)
- geração de código nativo para a CPU e hardware GPU
- integração com a pilha de bibliotecas científicas do Python, graças ao Numpy

3.4.4 JIT (*just-in-time*)

O JIT é uma anotação da biblioteca Numba utilizada como *@jit* acima do nome do método escrito em Python. O código do método contido abaixo da *flag* será compilado em tempo de execução, utilizando o conceito *just-in-time*. O JIT possui a seguinte assinatura: *@jit(signature, nopython, nogil, cache, forceobj, locals)*. Os parâmetros não são obrigatórios,

[9] <http://numba.pydata.org/>

o que faz com que o programador deixa a cargo do compilador escolher as melhores configurações para a compilação. Uma explicação simples dos parâmetros podem ser vistos abaixo.

- ***signature*** é a assinatura do método pelo qual será gerado o código de máquina. A assinatura pode ser uma assinatura simples, sendo somente o tipo de retorno ou pode ser uma lista de assinaturas contendo cada tipo de dados dos parâmetros do método em questão. O tipo de retorno pode ser omitido, sendo mostrado apenas o tipo dos parâmetros e pode ser utilizado o tipo definido pelo Numba, por exemplo: *(numba.int32, numba.double)*. Caso queira deixar bem definido o tipo retorno, pode ser usado da seguinte forma: *numba.void (numba.int32, numba.double)*. Outra forma de definir é usando o nome simples do tipo dos dados, assim: *void(int32, double)*.

- ***nopython*** é um valor booleano quando seu valor for *true*; força o método a ser compilado no modo *"nopython"*. Neste modo de compilação, o Numba gera código que não acessa a API Python C. Este modo de compilação produz o código de desempenho mais alto, mas requer que os tipos nativos de todos os valores no método possam ser inferidos. Caso contrário, o *@jit* retornará automaticamente um erro se não puder ser usado.

- ***forceobj*** é um valor booleano quando seu valor for *true*. Força o método a ser compilado no modo objeto. Como o modo objeto é mais lento que o modo *nopython*, isso é principalmente útil para fins de teste. Esta *flag* faz com que o modo de compilação Numba que gera o código que manipula todos os valores como objetos Python e usa a API Python C para executar todas as operações nesses objetos. O código compilado no modo objeto geralmente executará não mais rápido do que o código interpretado pelo Python, a menos que o compilador Numba possa aproveitar o mesmo código já compilado.

- ***nogil*** é um valor booleano que, quando seu valor for *true*, tenta liberar o bloqueio de interpretação global dentro do método compilada. O GIL (*global interpreter lock*) só será liberado se o Numba puder compilar a função no modo *nopython*, caso contrário, um aviso com erro de compilação será exibido.

- ***cache***, é um valor booleano que, quando seu valor for *true*, habilita um cache baseado em arquivo para encurtar os tempos de compilação quando a função já foi compilada em uma invocação anterior. O *cache* é mantido no subdiretório *__pycache__* do diretório que contém o arquivo de origem. Nem todas as funções podem ser armazenadas em cache, uma vez que algumas funcionalidades não podem ser sempre persistentes no disco. Quando um método não pode ser armazenado em cache, um aviso é emitido informando.

- ***locals***, é um dicionário que pode ser usado para forçar os tipos de dados Numba de variáveis locais particulares, por exemplo, se for necessário forçar o uso de variáveis de ponto flutuante com uma precisão específica. A documentação do Numba recomenda deixar o compilador definir os próprios tipos de variáveis.

Para a versão final do algoritmo deste trabalho, foi utilizada a anotação *@jit*, porém deixado com que o compilador escolhesse quais dados e forma de compilação usar. O conhecimento e estudo de cada parâmetro foi essencial para seguir com o desenvolvimento. Caso o programador decida definir qual a precisão utilizar e otimizar ainda mais o desempenho do código, é essencial um conhecimento mais aprofundado do compilador *just-in-time*.

3.4.5 Anaconda Navigator

A Anaconda[10] é uma plataforma bastante popular de dados científicos para Python e é utilizada por 4.5 milhões de usuários em todo o mundo. Está sempre liderando projetos de código aberto como Numpy e SciPy, que atualmente formam a base de muitas aplicações no meio científico. A Anaconda também pode ser utilizada e integrada com a linguagem utilizando o RStudio[11].

A plataforma pode ser obtida através do site oficial. Após a instalação, se o usuário achar preferível usar uma interface gráfica, deve-se usar o Navigator; caso prefira usar o próprio terminal, basta usar o conda. É possível alternar entre eles (qualquer modificação feita em uma biblioteca será refletida no outro). O Anaconda acaba sendo a interface gráfica para o conda. É possível instalar, remover ou atualizar qualquer pacote científico da Anaconda com apenas alguns cliques, utilizando o Navigator ou apenas um único comando do conda no terminal.

A versão do conda utilizada neste trabalho foi a versão 4.3.30 e a versão utilizada na interface gráfica, Anaconda Navigator, foi a 1.6.9.

[10] <https://www.anaconda.com/>
[11] <https://www.rstudio.com/>

4 PROJETO E DESENVOLVIMENTO

Foram conduzidos estudos bibliográficos para compreensão dos modelos físicos de propagação de sinais *wireless* em pequena e grande escala, levando em conta as perdas de sinal e sua atenuação (ALMERS et al., 2007). Então foi feito um levantamento do estado da arte sobre simulação da propagação de sinais, tal como, por exemplo, a modelagem da propagação de sinais de rádio através de *Ray Tracing*[1] (YUN e ISKANDER, 2015), visando à definição da técnica a ser utilizada para a simulação da propagação dos sinais *wireless* (LENTZ, 2013; NAJNUDEL, 2004; SANDEEP et al., 2008). Uma vez definido o mecanismo de simulação, foi definida qual técnica de otimização seria aplicada para o problema da maximização da cobertura de sinal do(s) AP(s) *Wi-Fi*, bem como definida a função objetivo para avaliar as soluções propostas. Por fim, foi utilizado um método de visualização da intensidade do sinal *wireless* (RENSBURG, 2005; SULAIMAN, 2012), por meio de um mapa de calor (*heat map*).

4.1 Representação do ambiente

O modelo bidimensional do ambiente utilizado como caso de estudo foi o bloco A, onde atualmente os professores e alunos têm reclamado de baixa cobertura de sinal Wi-Fi. Especificamente, os ambientes da sala dos professores e da sala de estudos tem apresentado baixa cobertura de sinal Wi-Fi, com frequentes desconexões dos usuários que ali estejam. Tipicamente, os pavimentos de edifícios e residências são representados através de softwares CAD, tais como o AutoCAD[2]. Entretanto, softwares CAD tipicamente utilizam um formato proprietário (DWG no caso do AutoCAD) e como este trabalho se propõe a seguir a filosofia de software livre e de código-fonte aberto (FOSS[3]), optamos por utilizar um formato de entrada interoperável chamado DXF[4] (*Drawing Interchange Format*), que pode ser facilmente exportado a partir dos *softwares* CAD popularmente utilizados.

4.1.1 Arquivo DXF

O software desenvolvido neste trabalho deve receber como arquivo de entrada uma representação bidimensional do ambiente a ser simulado, esperando como tal um arquivo com extensão DXF. Arquivos DXF são comuns na exportação de desenhos e projetos de

[1] <https://en.wikipedia.org/wiki/Ray_tracing_(graphics)>
[2] <https://www.autodesk.com.br/products/autocad/overview>
[3] <https://www.gnu.org/philosophy/floss-and-foss.html>
[4] <https://www.loc.gov/preservation/digital/formats/fdd/fdd000446.shtml>

peças para serem utilizadas em softwares de CAM e máquinas de corte automáticas. São populares por promoverem a troca aberta destes arquivos entre programas desenvolvidos por diferentes empresas[5]. Para interpretar o arquivo de entrada DXF, foi implementado um procedimento escrito na linguagem Python utilizando alguns recursos da biblioteca *ezdxf*, descrita na seção de materiais. Inicialmente, o procedimento realiza a leitura do arquivo DXF fornecido.

A partir do conteúdo do DXF, é obtido o *modelspace* no qual é possível buscar os valores (x,y) extremos do modelo 2D. Isto se faz necessário para que tais extremos sejam subtraídos no momento da leitura das paredes do edifício, a fim de evitar desperdício de processamento com as margens da planta-baixa e para que o desenho do ambiente possa se enquadrar perfeitamente na tela de visualização do mesmo. Tanto na busca dos valores limítrofes de x e y, quanto na leitura das linhas do modelo DXF, é feita uma busca em seu *modelspace* utilizando um laço de repetição. Na busca, é verificado se o tipo de dados no *modelspace* é do tipo LINE (linha) e se a camada percorrida é do tipo ARQ (arquitetura). Durante a verificação de linhas na camada ARQ, são adicionados a uma lista de paredes as coordenadas do ponto inicial (x1,x1) e ponto final (x2,y2) de cada linha representada no arquivo DXF, pois cada linha na camada ARQ do modelo representa uma parede. Observe que cada coordenada (x,y) deve ser devidamente multiplicada pela escala definida pelo usuário, para que seja mantido a proporção entre o tamanho da matriz de simulação e o ambiente real simulado. Ao final do processamento do arquivo DXF, os valores máximos (e mínimos) das coordenadas (x,y), bem como cada tupla de coordenadas das paredes é armazenada em memória pelo software, para que sejam utilizados pela heurística.

4.1.2 Escala e precisão

Observe que, a partir dos valores (x,y) limítrofes obtém-se as dimensões do modelo bidimensional e, desde que esteja disponível a informação sobre o tamanho real do edifício, pode-se calcular a escala da planta-baixa para o tamanho real do edifício. De posse das dimensões da planta-baixa e do edifício, pode-se representá-lo computacionalmente como um arranjo bidimensional, ou seja, uma matriz de tamanho Comprimento X Largura, podendo ser utilizada uma escala 1:1 do ambiente simulado para o ambiente real. Neste caso, uma célula da matriz representaria 1 m^2 no mundo real.

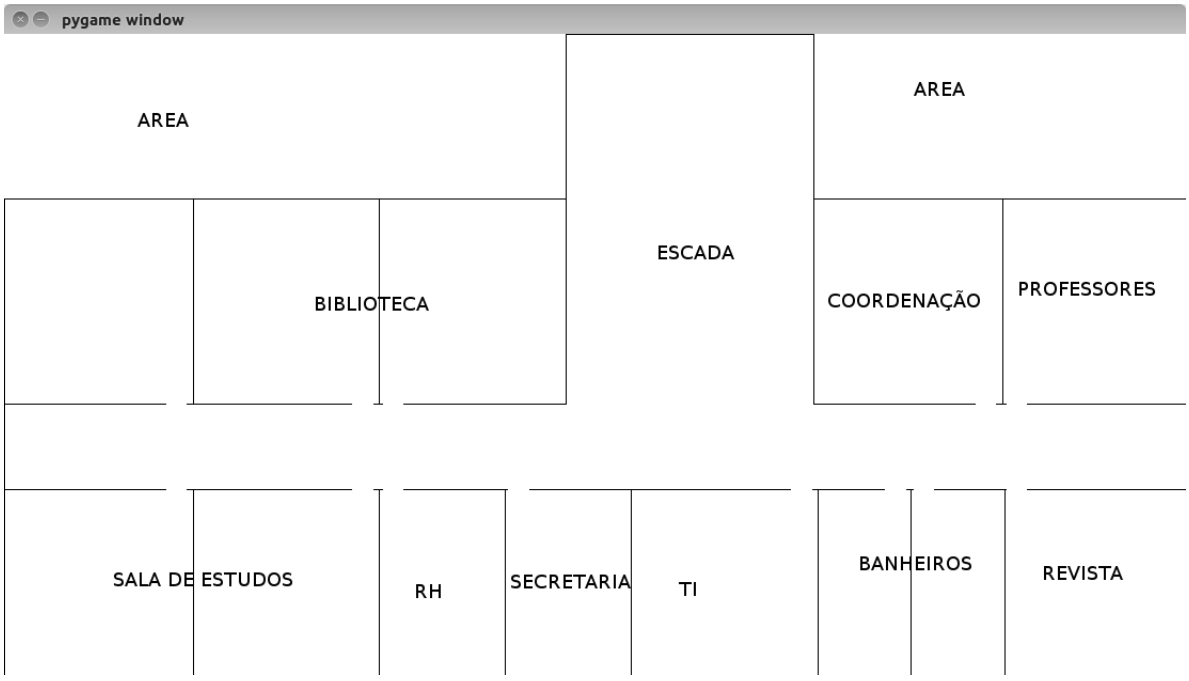
Entretanto, vale ressaltar que o software produzido é capaz de trabalhar com diferentes níveis de precisão na simulação de propagação de sinais, onde caso o usuário queira uma precisão de 0,5 m^2 por célula da matriz, bastaria informar tal configuração que o software passaria a utilizar um fator de escala de 2.0x a dimensão real do edifício (pois precisão = 1/escala). Assim, o software pode trabalhar com a precisão desejada

[5] Disponível em <https://www.otimizenesting.com.br/single-post/2016/09/29/Arquivo-DXF>. Acesso em 29 out. 2017.

pelo usuário, podendo inclusive realizar uma simulação de propagação de ondas onde cada célula da matriz poderia ter a dimensão de um comprimento de onda ($\lambda = c/f$). Por exemplo, caso o usuário deseje a precisão de 1 comprimento de onda para simular a propagação de Wi-Fi na frequência de 2,484 GHz, cada célula representa[6] uma área de 12,3 cm^2. Entretanto, vale ressaltar que quanto maior a precisão da simulação, maior serão as dimensões da matriz que representa o ambiente e, portanto, mais demorada será a simulação computacional.

As plantas-baixas representando os ambientes simuladores neste trabalho foram gentilmente cedidas pela equipe de TI do campus Formiga (bloco A) e com o setor de engenharia civil do campus Formiga (no caso dos blocos B e C). A figura abaixo ilustra o ambiente do bloco A do campus Formiga, tendo sido carregado automaticamente a partir do arquivo DXF. O texto descrevendo no que consiste cada sala do ambiente foi adicionado posteriormente à captura da imagem, para melhor compreensão e localização do leitor.

Figura 1 – representação do ambiente a partir do arquivo DXF contendo a planta-baixa



Fonte: Elaborado pelo autor

Para a visualização dos resultados dos testes e uma melhor interpretação dos resultados, foi utilizado o PyGame para prover a visualização gráfica da matriz contendo os resultados dos cálculos de intensidade de sinal em cada ponto do modelo simulado, conforme proposto pelo modelo de propagação utilizado. Após a representação da matriz de propagação no PyGame, podem ser desenhadas também as paredes do ambiente simulado

[6] $2.998 \times 10^8 m/s \div 2.437$ GHz <https://www.wolframalpha.com/input/?i=c+%2F+2.437+GHz>

de acordo com a lista de linhas obtidas no processamento do arquivo DXF. A maneira como a representação visual do ambiente simulado foi realizada será tratada em detalhes em seção posterior.

4.2 Propagação dos sinais

Conforme assunto abordado na seção onde foram falados sobre os modelos de propagação, se faz necessário uma boa escolha de qual modelo de propagação de sinais utilizar, para obter uma boa representação da intensidade dos sinais para aquela banda do espectro eletromagnético e, assim, simular algo mais próximo da realidade. Quanto maior for a precisão desejada para o valor da intensidade de sinal, devido a perdas e sua atenuação, mais detalhes sobre o ambiente de propagação devem ser modelados (frequências, distâncias, paredes, pisos, materiais, etc).

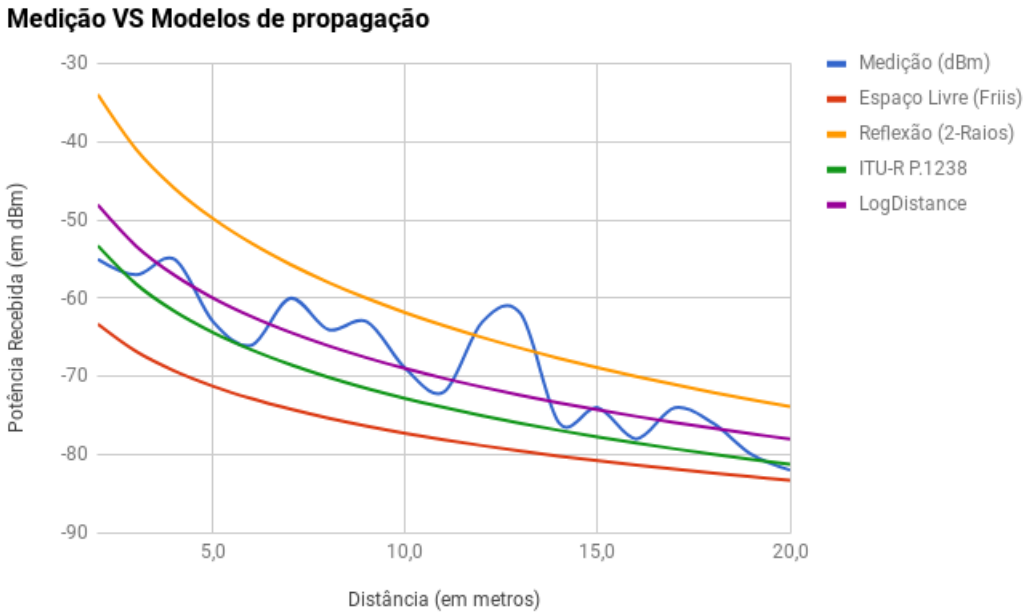
4.2.1 Definição do modelo de propagação para sinais Wi-Fi

Foram conduzidos nas dependências do campus Formiga experimentos para medição do decaimento da intensidade do sinal *Wi-Fi*. Foram realizadas medições *indoor* nos corredores do campus. Para tal, foi posicionado no corredor um *access point* Cisco WAP200 (IEEE 802.11b/g) e configurado para utilizar um canal *Wi-Fi* que correspondesse a uma frequência que, naquele momento, estivesse livre de interferência de outros APs naquela região do campus. A potência máxima de transmissão que o AP suporta é -14 dBm (0,0398 mW), mas ele foi configurado para utilizar apenas 25% dessa potência. Tal configuração visou apenas reduzir a distância a ser percorrida na condução do teste, uma vez que as paredes e pisos absorvem mais ou menos o sinal de acordo com a frequência dele e não de sua intensidade, portanto é foi mantido o comportamento esperado na propagação do sinal. O *access point WAP200* utilizado como transmissor possui antenas com 2 dBi de ganho e a interface Wi-Fi do notebook utilizado como receptor nos testes possui ganho de 1 dBi. Foi utilizado o canal 11 (portanto com frequência de 2,462 GHz e λ de 12,18 cm), tendo sido realizadas medições após o campo distante da antena (campo de *Fraunhofer* = 148 cm).

As medições foram coletadas utilizando o comando *iwlist* da biblioteca *iw-utils*, que realiza a varredura (*scanning*) da faixa de espectro correspondente ao Wi-Fi e registra os valores de intensidade de sinal recebidos bem como o endereço MAC do(s) AP(s). A figura XXX ilustra quão bem cada modelo de propagação considerado previu o valor da intensidade de sinal de acordo com com o aumento da distância. Pela figura, observe que o modelo de reflexão de dois raios, por definição e de acordo com a altura das antenas (aproximadamente 90 cm, considerando também o suporte utilizado), por definição seria apropriado somente se as distâncias entre transmissor e receptor fossem superiores a 83,59

m, faixa demasiado próxima do limite de alcance de um enlace *Wi-Fi* típico. Também, observe que com o modelo de propagação no espaço livre (modelo de Friss), as estimativas foram abaixo do valor real observado, por tal modelo não considerar os fenômenos de reflexão, difração e dispersão que tipicamente ocorrem na propagação de ondas em um ambiente geometricamente complexo (paredes, piso, teto, portas, janelas, mobília, etc).

Figura 2 – Medição da intensidade de sinal vs. modelos de sua propagação



Fonte: Elaborado pelo autor

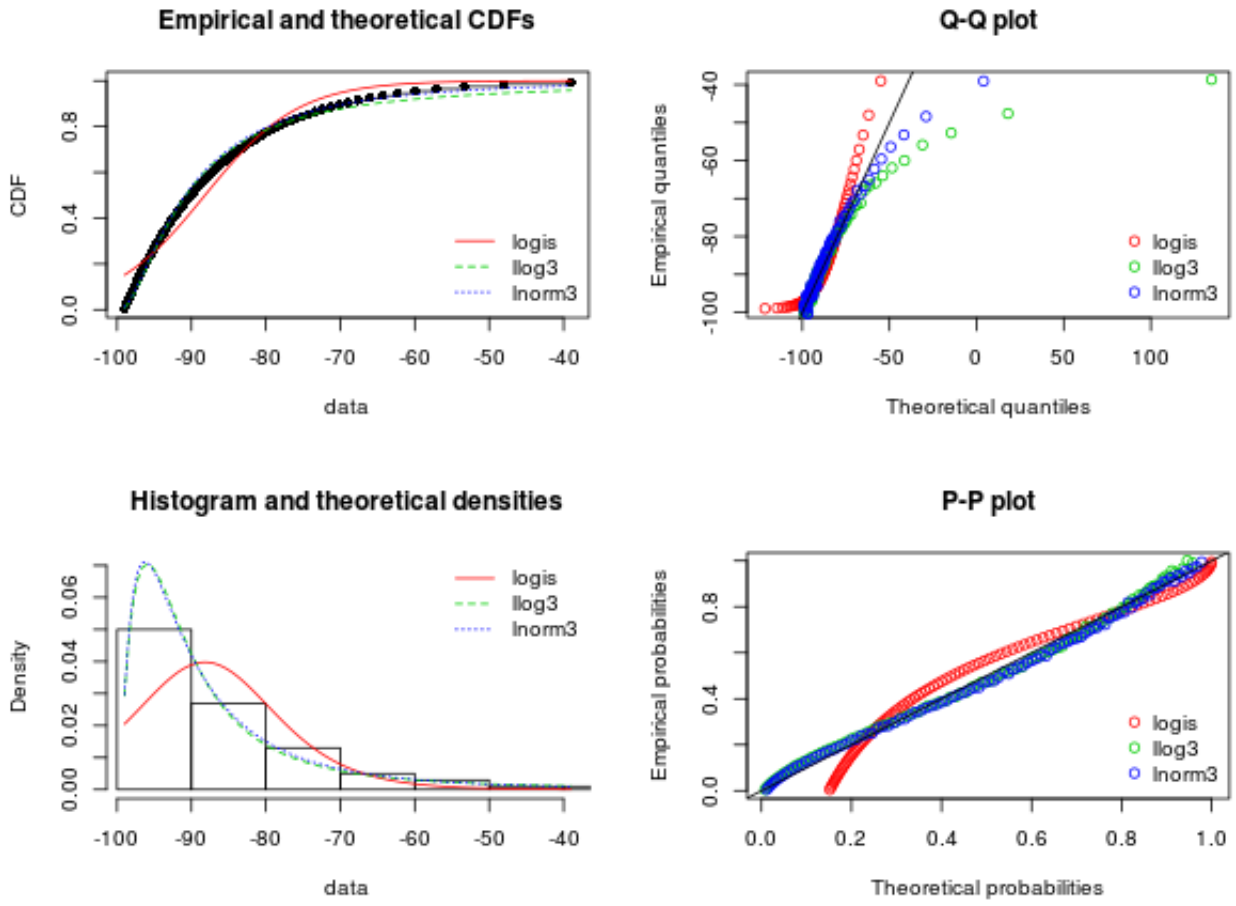
Pela Figura 2 é possível observar que, dos modelos de propagação analisados, o *LogDistance Path Loss* apresentou um bom casamento entre os valores de suas estimativas e os valores reais mensurados, sendo portanto considerado como o mais promissor. Entretanto, observe que a figura acima ilustra apenas uma das várias medições realizadas, ilustrando portanto o decaimento exponencial da intensidade do sinal *Wi-Fi* ao longo de um corredor do bloco B. Neste caso, o modelo *LogDistance* foi calibrado para uma distância de referência (d0) de 10 metros com a respectiva perda (PL0) mensurada em -69 dBm, com expoente de perda de caminho manualmente ajustado como $\gamma = 3$ (*path loss exponent*).

4.2.2 Ajuste do modelo de propagação

Não podemos basear nossa simulação em parâmetros de um modelo de configuração calibrado para medições coletadas em *apenas um corredor de um pavimento de um prédio do campus Formiga*. Se faz necessário calibrar o modelo de propagação para que que seja mais abrangente para o *campus*, devendo ser calibrado de acordo com múltiplas medições realizadas em diversas salas, corredores e áreas internas de nosso caso de estudo.

Para tal, as informações de distância vs intensidade das várias medições conduzidas foram utilizadas como entrada para um ajuste de curvas estatístico, técnica de regressão que foi conduzida no *software R-Project*[7]. Inicialmente, conduzimos regressões com um ajuste de curvas estatístico utilizando os pacotes *fitdistrplus*[8] e *FAdist*[9] do software *R-Project*.

Figura 3 – Ajuste de curvas da 2P-Logística, 3P-LogNormal e 3P-LogLogística



Fonte: Elaborado pelo autor

Pela comparação do ajuste de curvas ilustrado na Figura 3, observamos que a distribuição Logística com apenas 2 parâmetros (logis) não consegue modelar bem o decaimento do nível de sinal observado na medições do RSSI no campus Formiga. Por outro lado, as distribuições 3P-LogNormal (lnorm3) e 3P-LogLogistica (llog3) se aproximaram bem do comportamento esperado para o decaimento da intensidade do sinal, estando tal fato em consonância com o exposto nas seções 2.4.6 (Log-normal fading) e 2.4.3 (Log-distance).

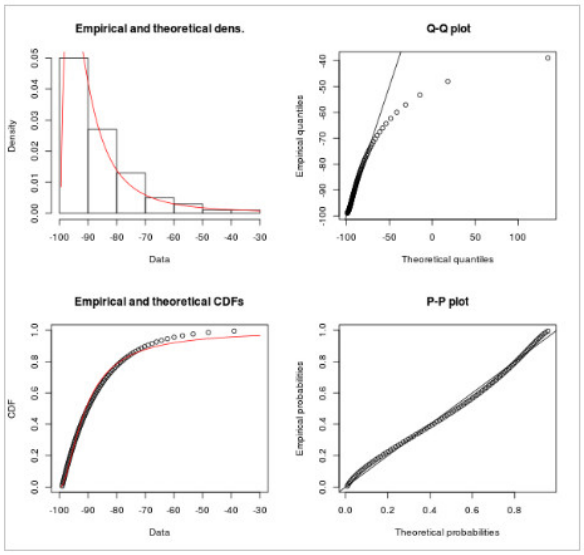
[7] <https://www.r-project.org/>
[8] <https://cran.r-project.org/web/packages/fitdistrplus/index.html>
[9] <https://cran.r-project.org/web/packages/FAdist/index.html>

A Figura 4 e a Figura 5 apresentam em detalhes a qualidade do ajuste de curvas estatístico, obtido através das regressões realizadas. Ao comparar o GOF (Goodeness-of-fit) entre duas ou mais distribuições, pode-se considerar os critérios de qualidade do ajuste AIC (Akaike's Information Criterion) e BIC (Bayesian Information Criterion).

Figura 4 – Qualidade do ajuste (GOF) da distribuição 3P-LogLogística

3P-LogLogística

shape	scale	thres
0.6238972	2.1584165	-99.4197743



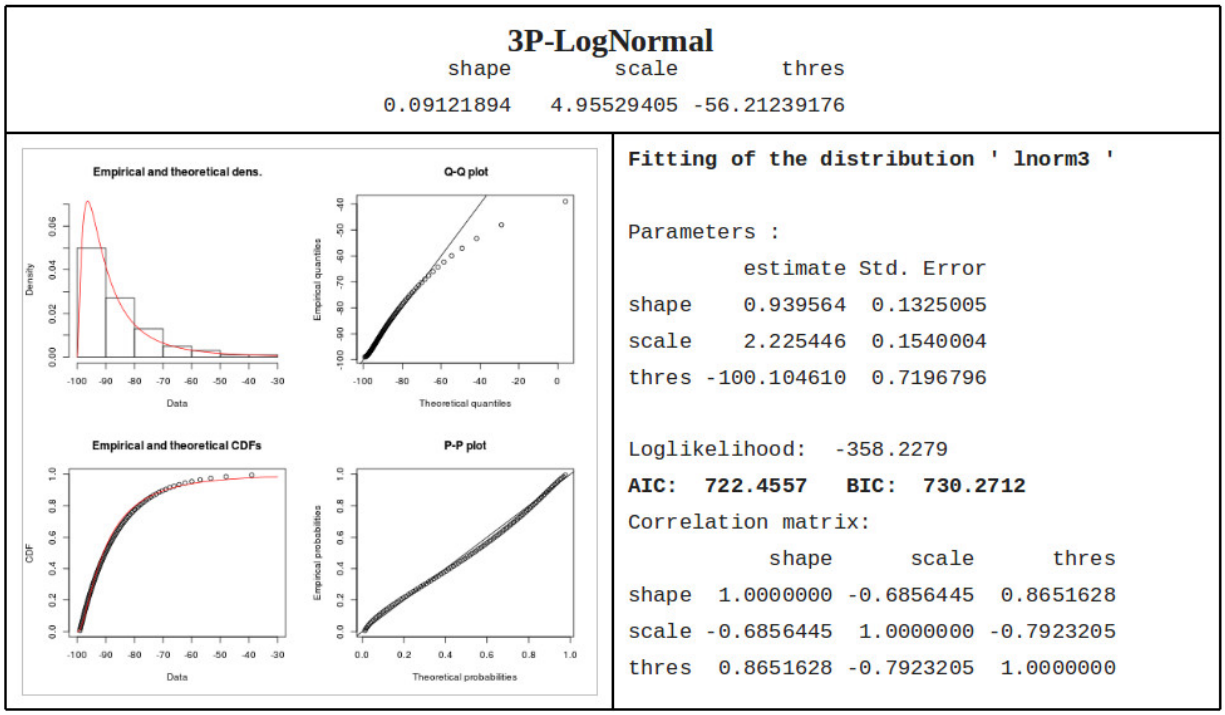

```
Fitting of the distribution ' llog3 '

Parameters :
        estimate Std. Error
shape   0.6230023 0.08748687
scale   2.1586816 0.13736264
thres -99.4227152 0.51870476

Loglikelihood:  -360.3786
AIC:  726.7572   BIC:  734.5728
Correlation matrix:
           shape      scale      thres
shape  1.0000000 -0.5133994  0.8078526
scale -0.5133994  1.0000000 -0.6065548
thres  0.8078526 -0.6065548  1.0000000
```

Fonte: Elaborado pelo autor

Considerando o melhor ajuste das distribuições LogNormal e LogLogistica, ambas com 3 parâmetros, passamos a considerar uma regressão estatística com distribuições estatísticas mais complexas, de 4 ou mais parâmetros, mas mantendo o comportamento logarítmico. Uma regressão **NP-Logística** foi conduzida com a utilização do pacote *nplr*, também no software *R-Project*.

Para obtermos um melhor resultado, considerando o bom desempenho das distribuições LogNormal e LogLogística, instruímos o modelo de regressão Logístico a utilizar uma base logarítmica para os valores do eixo x, ou seja, a distância ao AP deve ser interpretada como $log_{10}(x)$. Em uma livre interpretação, isso corresponderia a instruir a regressão logística a utilizar modelos NP-LogLogísticos. Assim, buscamos partir dos bons resultados obtidos na análise prévia, mas tentando melhorá-los com a inclusão de um quarto ou quinto parâmetros de ajuste da distribuição. Obtivemos um melhor fitness com um modelo Logístico de 4 parâmetros (*4-Parameter Logistic* ou simplesmente 4PL), conforme resultados apresentados a seguir. Conforme ilustra a Figura 6 e a Figura 7, o

Figura 5 – Qualidade do ajuste (GOF) da distribuição 3P-LogNormal



Fonte: Elaborado pelo autor

melhor ajuste de curva (GOF) foi obtido com uma distribuição 4P-Logística e com a distância fornecida em em escala logarítmica, como $log_{10}(x)$ metros.

]

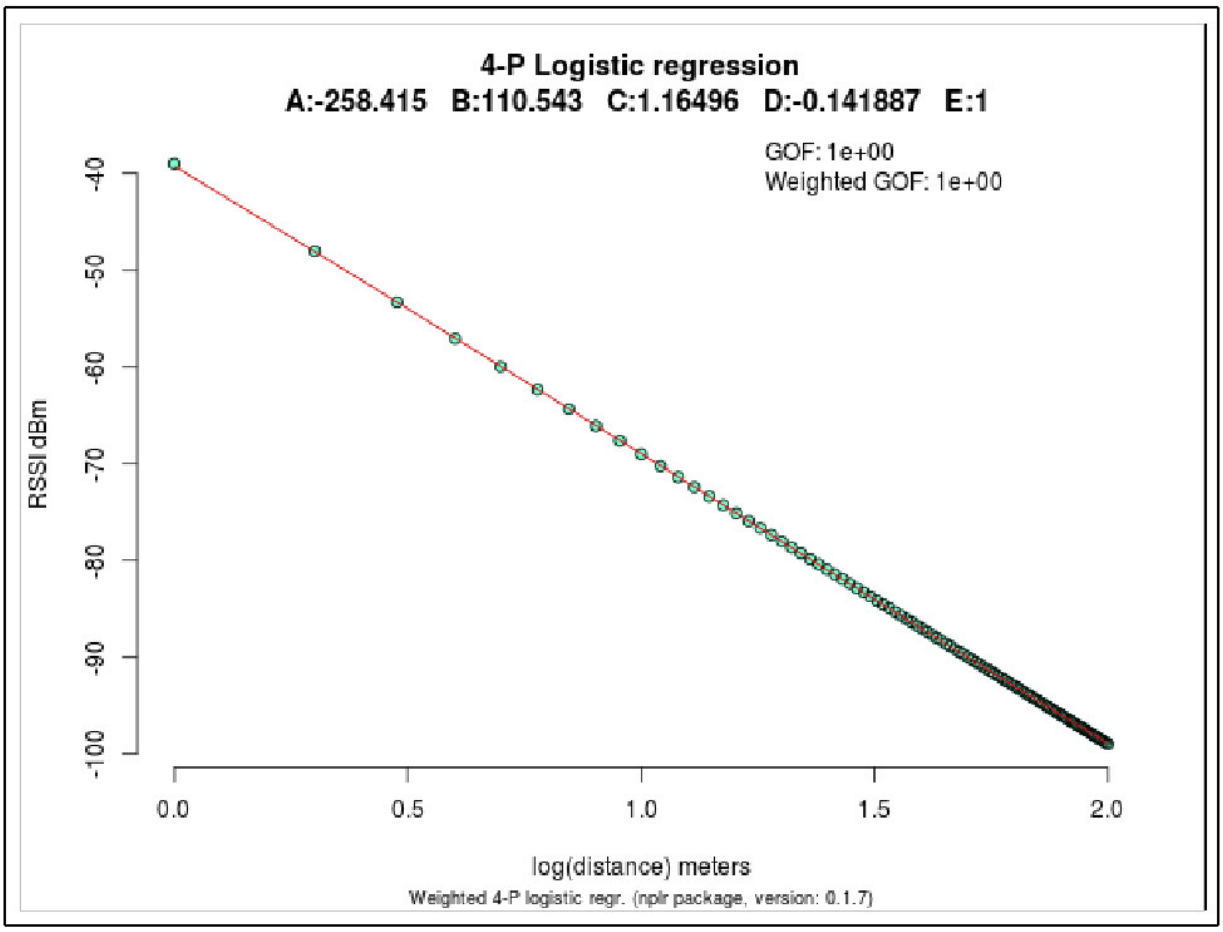
Enfim, a partir dos parâmetros da 4P-Logística (4PL ou logis4) obtidos pela regressão, podemos configurar as constantes de sua equação e, a partir dela, obtiver a estimativa de decaimento do nível de sinal com o aumento da distância do AP, calibradas para a realidade do campus Formiga. A equação característica de uma NP-Logística de cinco parâmetros é:

$$f(x; A, B, C, D, E) = D + (A - D)/((1 + (x/C)^B)^E) \tag{4.1}$$

, na qual os parâmetros A, B, C, D e E são obtidos ao aplicar-se a regressão logística para os valores de intensidade de sinal (em dBm ou mW). Observe que, fixando "E=1"obtemos a equação da 4PL (*4-Parameter Logistic*) e se fixarmos ambos "D=0"e "E=1", obtemos a equação da 3PL (*3-Parameter Logistic*). Os cinco parâmetros da NP-Log tem o seguinte significado:

- A = Mínima assintótica. Em medições da propagação de sinais de rádio, considere-o

Figura 6 – Previsão de RSSI da 4P-LogLogística para distâncias em $log_{10}(x)$ metros



Fonte: Elaborado pelo autor

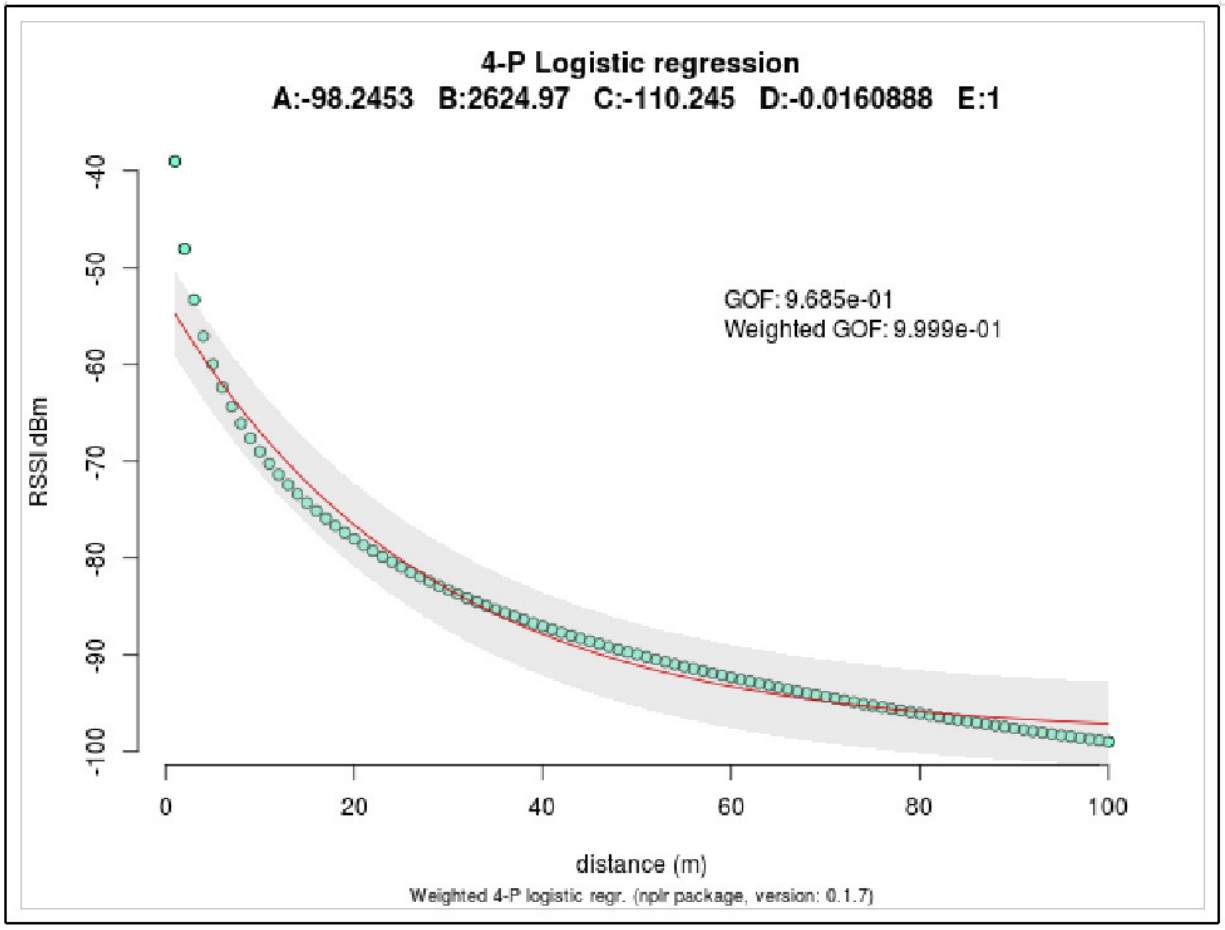
como o valor do sinal para uma distância de referência. Em propagação de sinais de rádio, tipicamente é o valor do sinal a 10 metros do transmissor. No caso do *Wi-Fi*, será um valor de referência aferido após o campo distante da antena (distância de *Fraunhoffer*), ou seja, intensidade do sinal aferida a 2 metros (2,4 GHz) ou 4 metros (5,6 GHz).

- B = Inclinação da curva, que pode ser positiva ou negativa. No caso de propagação de sinais de rádio, para obtermos um decaimento do sinal provavelmente B será positivo.

- C = Ponto de inflexão, definido como o ponto onde a curvatura muda de direção ou sinal. C é a concentração onde y = (D-A) / 2. No caso do Wi-Fi, provavelmente será o valor do sinal aferido em torno 10 a 20 metros.

- D = Máxima assintótica. Em medições da propagação de sinais de rádio, considere-o como o valor do sinal para uma distância muito grande. No caso do *Wi-Fi*, pode-se

Figura 7 – Previsão de RSSI da 4P-Logística para distâncias em metros



Fonte: Elaborado pelo autor

utilizar um valor de referência aferido a 100 metros (limita da WLAN).

- E = Fator de assimetria. Para E=1, temos uma curva simétrica em torno do ponto de inflexão e, portanto, temos uma equação logística de quatro parâmetros.

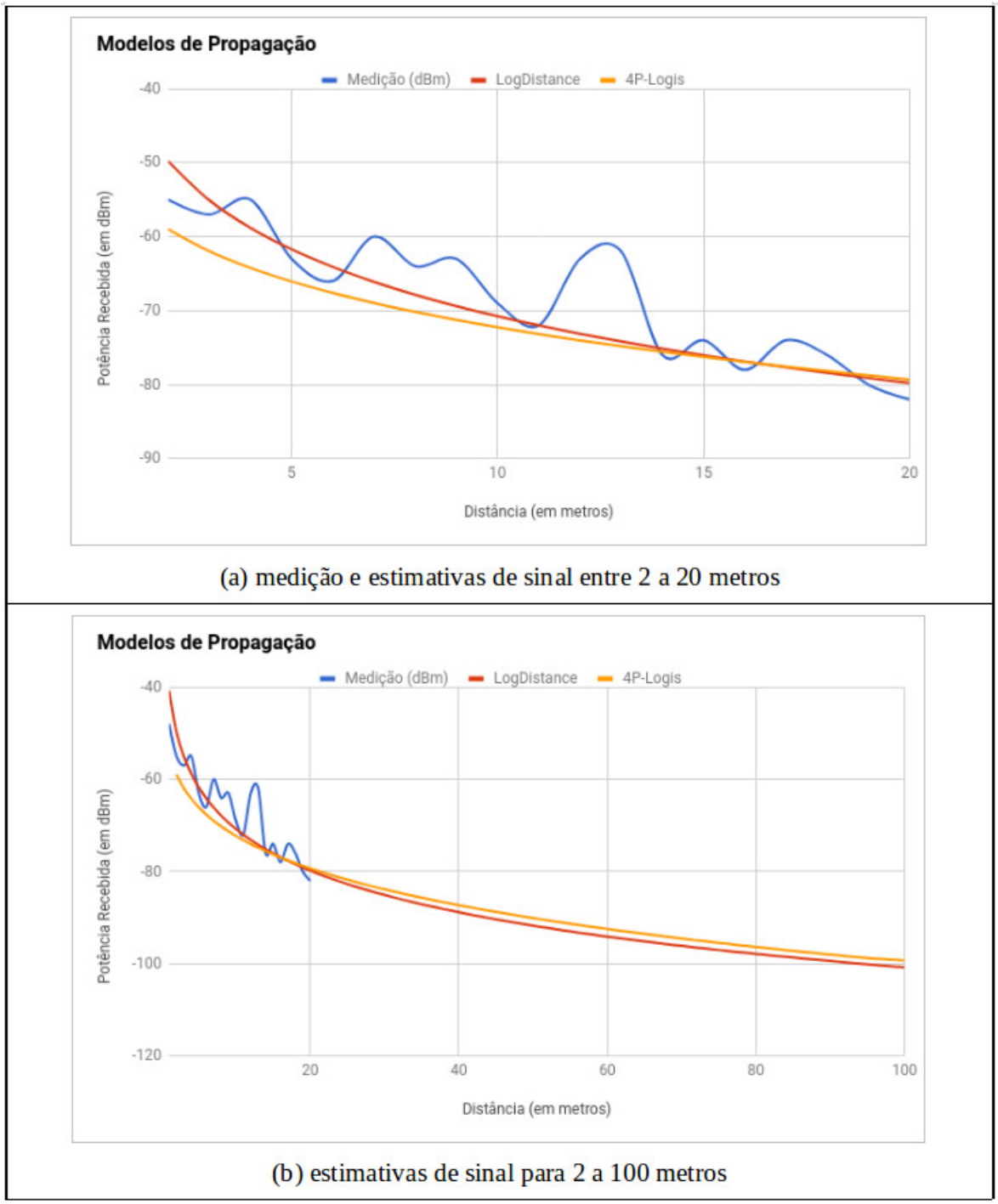
Para utilizar tal distribuição estatística como modelo de propagação, deve-se implementar no software sua **função de densidade de probabilidade** (PDF) para que possa retornar qual seria o valor da intensidade do sinal recebido (RSSI) em determinada distância do AP Wi-Fi. Durante a implementação, vale observar que ambas as distribuições 4P-Log e LogDistance devem ser utilizadas como um modelo de **perda de sinal**, ou seja, o valor da estimativa obtido (PL) deve ser subtraído da potência de transmissão (Pt) do AP para obter a previsão da potência recebida (Pr):

$$Pr(x) = Pt - PL(x) \tag{4.2}$$

, onde Pr é a potência recebida à distância x do transmissor; Pt é a potência do sinal do transmissor (ex.: -20 dBm) e PL é a perda do sinal ao longo do caminho entre o transmissor e receptor (calculada com a equação da 4PL ou da *LogDistance*). Inicialmente, implementamos tais fórmulas em uma planilha para visualizarmos graficamente uma comparação das medições de RSSI realizadas no campus com a expectativa de decaimento proporcionado pelos modelos LogDistance e 4P-Logístico. A Figura 6 ilustra o decaimento se sinal previsto por cada um dos modelos acima citados em relação aos valores reais de medição coletados.

Já na Figura 8 (b), é apresentada a extrapolação que os modelos de propagação de fornecem para distâncias de até 100 metros (limite de alcance de um sinal Wi-Fi indoor. Comparando as Figura 8 (a) e (b), é interessante notar que os dois modelos diferiram mais para curtas distâncias, escala onde acontece muita variação do nível de sinal devido aos efeitos de múltiplas reflexões nas paredes, difração nas portas, dispersão e absorção pelos materiais. Isto está plenamente de acordo com a abordagem do assunto pela literatura, no que tange a modelos de propagação de pequena e larga escala (RAPPAPORT, 2009).

Figura 8 – comparação dos modelos de propagação LogDistance e NP-Logístico



(a) medição e estimativas de sinal entre 2 a 20 metros

(b) estimativas de sinal para 2 a 100 metros

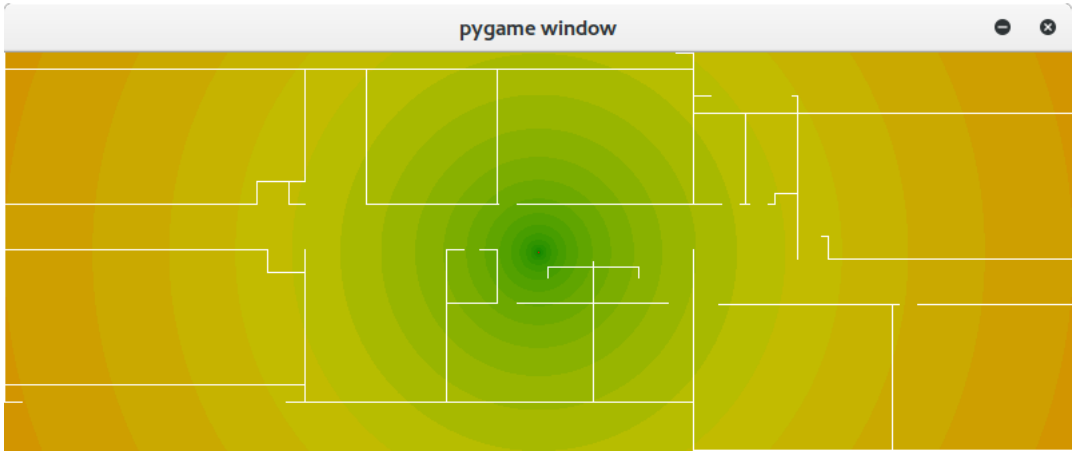
Fonte: Elaborado pelo autor

4.2.3 Simulação da propagação de sinais no ambiente

Como dito anteriormente, para a realização dos cálculos, o ambiente foi representado como uma matriz bidimensional, representando em escala o ambiente a ser simulado. Cada posição [x,y] da matriz é um ponto candidato a hospedar o *access point* (AP). Uma vez determinada a proposta de nova localização do AP, para cada posição da matriz é necessário realizar o cálculo do modelo de propagação. Na configuração do software produzido, pode-se informar a posição atual do AP no ambiente real, ou deixar que o software inicie a exploração a partir de uma posição aleatória. Para preencher os valores da matriz representando a propagação do sinal de microondas, deve-se realizar o cálculo determinado pelo modelo de propagação, conforme tratado na seção anterior. Tipicamente, como os modelos de propagação fazem uso da distância entre o transmissor e receptor, esta deve ser calculada utilizando as coordenadas do *access point* na simulação e do ponto para o qual deseja-se calcular a estimativa da intensidade do sinal ali recebido (RSSI).

A Figura 9 exibe uma representação gráfica da matriz com os valores de intensidade de sinal conforme o decaimento exponencial da intensidade do sinal recebido (RSSI), com o aumento da distância de cada ponto da matriz com o AP Wi-Fi (ponto central).

Figura 9 – decaimento RSSI de Wi-Fi de acordo com a distância ao AP

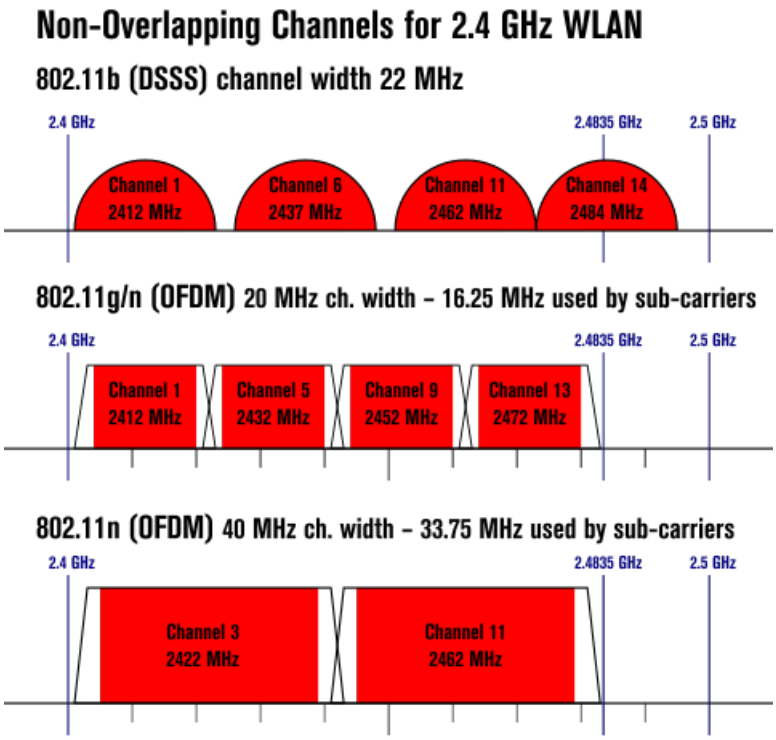


Fonte: Elaborado pelo autor

Neste ponto, vale notar que o escopo deste trabalho não considera interferências entre canais Wi-Fi de vários APs, de maneira que para cada AP simulado é gerada uma matriz numérica individual, que receberá as estimativas de valores para a intensidade de sinal em decibel miliwatt (dBm). O propósito deste trabalho é sugerir um melhor posicionamento dos APs e a decisão de não terem sido modeladas interferências entre APs diferentes é baseada no fato de que a tecnologia *Wi-Fi* possibilita coexistência de alguns APs em um mesmo ambiente, desde que o canal central de cada AP esteja suficientemente espaçado dos demais. Por exemplo, nos padrões IEEE 802.11b/g/n a largura de banda é

de 20 MHz e portanto podem coexistir até quatro APs em um mesmo ambiente, conforme ilustra a Figura 10.

Figura 10 – canais não sobrepostos para WLANs de 2,4 GHz



Fonte: <https://en.wikipedia.org/wiki/File:NonOverlappingChannels2.4GHzWLAN-en.svg>

4.2.4 Atenuação do sinal ao atravessar paredes

Para se ter um resultado que fosse condizente com a realidade e fosse possível ver quanto o sinal ficava ruim quando se propagava através de cômodos, foi adicionada uma técnica para absorção do sinal em paredes. Todas as linhas (que representam as paredes) coletadas antes de iniciar a simulação são essenciais no momento de se guardar o valor da intensidade do sinal em um respectivo ponto (ponto que representa uma posição válida na planta baixa).

O resultado do modelo de propagação para cada ponto da matriz foi subtraído por um valor que é obtido pelo número de paredes que o sinal ultrapassou multiplicado por um valor em dBm que representa a energia absorvida por parede. A literatura sugere que esse valor pode variar entre 8 e 15 dBm para paredes de concreto. O valor definido para esse trabalho foi 8 dBm, mas pode ser facilmente modificado para que possa se ter resultados de simulações onde o decaimento por absorção em paredes seja maior ou menor.

Para se saber quantas paredes o sinal sofre atenuação, foram utilizadas técnicas de geometria analítica. No momento do cálculo da absorção das paredes para um ponto [x, y] qualquer da matriz, a lista de linhas representando as paredes é percorrida. A técnica utilizada foi a intersecção de retas num plano 2D. Tal técnica foi fácil de se utilizar, pois é simples de se ver que há um segmento de reta entre o *access point* e o ponto que está sendo calculado a intensidade do sinal. O outro segmento de reta está na lista de linhas obtidas do arquivo *.dxf*. Como já foi dito anteriormente, para cada linha é guardada uma lista com as coordenadas do ponto inicial e ponto final de cada reta. A partir disso se torna também muito fácil obter o outro segmento de reta.

Com a fácil obtenção dos dois segmentos de retas, pôde se aplicar tranquilamente o conceito de interseções de retas e definir se ambas possuem um ponto em comum. Esta definição é dada por retas concorrentes. Duas retas são concorrentes se possuírem um ponto em comum, ou seja, a intersecção das duas retas é o ponto em comum.

Com mais um tratamento do espectro do sinal a ser feito, surgiu mais um problema. Com a simulação da colisão do sinal nas paredes, o tempo de execução do algoritmo extrapolou o esperado.

4.3 Visualização dos dados

A princípio, foi implementada a etapa que trata a representação do ambiente, ao utilizar a matriz de propagação para a geração do modelo 2D correspondente à planta-baixa fornecida (KASE et al., 2005; THAKUR et al., 2009). Tal modelo pode representar algumas características físicas do ambiente, tais como paredes e janelas (MARSCHALLINGER, 1996), no qual o sinal pode atenuar em um maior ou menor grau sua intensidade de propagação.

Com o preenchimento da matriz de propagação de acordo com a posição do *access point* e com o respectivo valor da intensidade do sinal, se tornou difícil a visualização de quanto o sinal ficava ruim quanto maior era a distância e mais paredes eram atravessadas. Tendo então uma matriz com valores em dBm, ela já é suficiente para a realização dos cálculos com a função objetivo, mas para a visualização a olho nú, não é possível distinguir, dentre uma grande quantidade de números numa matriz, o que poderia ser considerado um valor bom ou ruim.

Com a vasta quantidade de ferramentas e bibliotecas disponibilizadas pela linguagem Python, foi necessário utilizar algum método para transformar a matriz de valores em dBms, algo um tanto quanto visível e de fácil interpretação. Diante disso, o PyGame foi utilizado na interpretação da matriz resultante. Com a obtenção dos valores máximos e mínimos dentro da matriz de propagação, é fácil definir uma faixa de valores para serem uma escala de referência para a exposição dos resultados.

Foi implementado um método que explorasse os recursos disponibilizados pelo PyGame e pela matriz resultante. Tal método, que pode ser visto abaixo, percorre a matriz obtendo o valor calculado em dBm da posição [x, y]. Em seguida, a função *get_color_of_interval* retorna à respectiva cor, de acordo com o valor informado e os valores máximos e mínimos da matriz. O método *draw_point* fica responsável por colorir o ponto de acordo com o valor da matriz em sua posição. Ao término da impressão do espectro na janela do PyGame, os pontos onde estão localizados os *access points* são também coloridos com a cor vermelha. Por fim, o método atualizada a janela, fazendo com que a impressão de todo o espectro e posições dos APs seja instantânea.

```
def print_pygame(matrix_results, access_points, DISPLAYSURF):
    matrix_max_value = matrix_results.max()
    matrix_min_value = -100
    # Le os valores da matriz que contem valores
    # calculados e colore
    for x in range(WIDTH):
        for y in range(HEIGHT):
            color = get_color_of_interval(matrix_results[x][y],
            matrix_max_value, matrix_min_value)
            draw_point(DISPLAYSURF, color, x, y)
    # Pinta de vermelho a posicao dos Access Points
    for ap in access_points:
        draw_point(DISPLAYSURF, RED, ap[0], ap[1])
    # Atualiza a janela do PyGame para que exiba a imagem
    pygame.display.update()
```

O método que obtém uma cor numa determinada faixa de valores, *get_color_of_interval*, como foi mostrada sua utilização anteriormente, recebe como parâmetro os valores máximos e mínimos do intervalos, juntamente com o valor no qual se quer obter a respectiva cor. Neste método, foi utilizada uma constante *SENSITIVITY*, que representa a sensibilidade máximo definida nos APs. O valor definido durante a implementação e teste variou entre -75 e -100 dBm, o que significa que, se o valor informado for menor que a sensibilidade do AP, é considerado um ponto que pode ser entendido como "cego", uma vez que não há sinal suficiente para ter uma boa qualidade de serviço, na janela do PyGame, aparecerá um ponto na cor preta.

Ainda no método que obtém uma cor num intervalo numérico, de acordo com os valores máximo e mínimos e o valor de x, é adquirido um valor em porcentagem que é diretamente correspondente ao valor de x no intervalo. A fórmula utilizada no método que retorna essa porcentagem é a seguinte:

```
def get_percentage_of_range(min, max, x):
```

```
return ((x - min) / (max - min)) * 100
```

Com a porcentagem do respectivo valor calculado, precisa-se então adquirir a cor desejada. Tal cor pode ser obtida com o método *get_value_in_list*, o qual recebe a porcentagem e uma lista, sendo ela genérica, mas neste caso, utilizada para guardar a faixa de cores, que segue uma escala gradiente. Neste método a porcentagem então é convertida de volta em um valor que representará a posição na lista de cores. A posição é obtida pela seguinte fórmula:

```
def get_value_in_list(percent, list):
    position = (percent / 100) * len(list)
    if position < 1:
        position = 1
    elif position >= len(list):
        position = len(list)
    return hex_to_rgb(list[int(position - 1)])
```

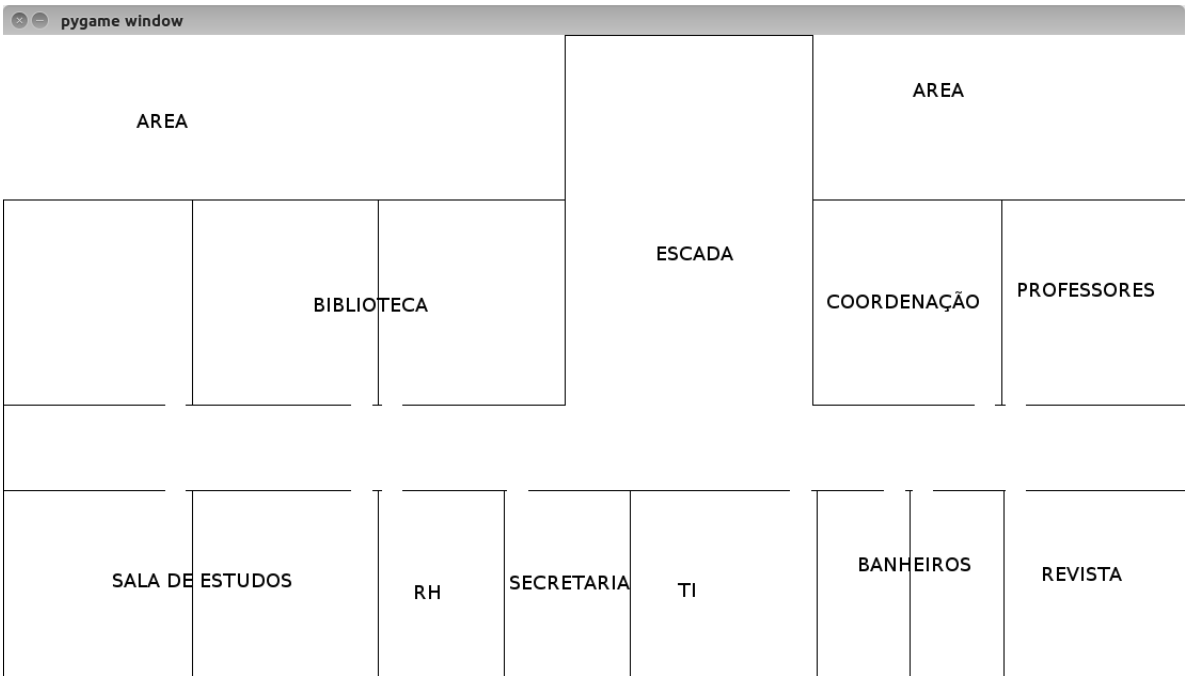
Com a posição bem definida, pode-se ter a cor, cor que está com o valor em hexadecimal. O PyGame utiliza valores de cor em RGB para colorir os pontos definidos. Desse modo, é necessário converter a cor que foi obtida da lista para o padrão do PyGame.

A imagem abaixo é uma representação gráfica do bloco A[10] utilizando a biblioteca PyGame. Nela é destacado o que cada cômodo representa atualmente.

Como o método que realiza a simulação e o preenchimento da matriz retorna à mesma, e tal método é chamado no *Simulated Annealing* toda vez que é necessário avaliar uma solução, o trajeto do APs ao procurar o melhor ponto dentro da planta pode ser desenhado utilizando o PyGame. Com a coleta de dados no final, essa funcionalidade foi descontinuada. No entanto, para que o resultado da simulação possa ser visualizado em real time, basta chamar o método *print_pygame* a cada vez que uma avaliação for realizada (ela é aceita pelo SA).

[10] A modelagem de ambiente utilizada para as simulações apresenta as mesmas divisões de cômodos do arquivo AutoCAD disponibilizado para o este trabalho.

Figura 11 – representação do ambiente a partir do arquivo DXF contendo a planta-baixa



Fonte: Elaborado pelo autor

4.4 Heurística de otimização

Para o cumprimento dos objetivos e um bom resultado, foi implementada a metaheurística, que irá explorar o espaço de soluções de posicionamento dos APs no modelo espacial do ambiente, visando à maximização da cobertura do sinal (dentre outras possíveis métricas). Conforme exposto na seção de materiais e métodos, foi utilizada a metaheurística proposta por Scoot Kirkpatrick. O *Simulated Annealing* foi utilizado para a realização do processo de otimização, buscando encontrar a melhor solução viável, considerando o objetivo do problema em questão e o conjunto de restrições para aceitação da solução proposta.

4.4.1 Implementação do *Simulated Annealing*

Inicialmente a metaheurística buscava o melhor ponto apenas para um AP. Com o cumprimento dos objetivos, se tornou possível a busca e otimização de mais de um AP. O SA inicia com os valores predefinidos no momento da sua chamada. Para o método é informado: o número de APs a serem utilizados no ambiente, o número máximo de iterações, o número máximo de perturbações por cada iteração, o número máximo de sucessos por iteração e o *alpha* como o fator de redução da temperatura.

Com a estratégia de perturbar mais de um AP por vez, o SA inicia alocando todos os APs na posição central da planta baixa. A primeira solução é avaliada (APs no centro) para que então seja realizada a busca e novas perturbações sejam feitas. O SA fica em um *loop* principal, no qual se verifica se foram atendidas todas as condições de término do algoritmo (se extrapolou o máximo de iterações ou se nenhum ponto com ótimo local foi encontrado). Outro *loop* interno realiza a perturbação dos APs em forma de lista circular. Com os valores de *fSi* (resultado da avaliação da função objetivo na *i-ésima* iteração do SA) e *fS* (melhor resultado da avaliação da função objetivo iteração do SA) é possível determinar o valor de *deltaFi* (diferença da avaliação da função objetiva atual com a avaliação da função objetiva ótima encontrada até o momento). Caso o valor de *deltaFi* seja menor ou igual a zero, há uma redução de energia, a qual implica que a nova solução é melhor que a anterior. Então é aceito a solução e *fSi* passa a ser a nova solução corrente. A aceitação desse tipo de solução é mais provável em altas temperaturas e bastante improvável em temperaturas reduzidas. Para reproduzir essas características, geralmente usa-se, para calcular a probabilidade de se aceitar a nova solução, uma função conhecida por fator de *Boltzmann*, que em sua fórmula o valor de T é a temperatura atual e que regula a probabilidade de soluções com pior custo. Tal fórmula pode ser vista abaixo:

$$e^{(-\Delta/T)} \tag{4.3}$$

Com a aceitação, é atualizada a lista das melhores posições para a alocação

dos APs e também a variável *fS* com o valor de melhor função objetivo, o número de sucessos representando a vizinhança, que também é fator de parada do SA, e o número de perturbações por iteração é incrementado. O método é finalizado quando a temperatura chega a um valor próximo de zero e nenhuma solução que piore o valor da melhor solução seja mais aceita, ou seja, quando o sistema estiver estável.

Por fim, a melhor solução encontrada pela metaheurística é retornada, indicando a proposta do novo posicionamento dos APs e a cobertura do sinal *Wi-Fi* alcançada, caso tal solução fosse implantada no ambiente real. Além disso, é fornecida uma visualização gráfica do resultado a ser obtido com a solução proposta utilizando o PyGame.

4.4.2 Calibração dos parâmetros do Simulated Annealing

Definir quais os parâmetros utilizar na metaheurística não é uma tarefa fácil, então para isso, qualquer ferramenta que auxiliasse neste processo seria útil. Para a escolha de quais parâmetros utilizar para os valores foi empregado um Projeto Fatorial 2K com foco no ajuste do *Simulated Annealing*.

Com a aplicação do planejamento de experimentos, é possível definir uma sequência de coletas de dados experimentais a fim de atingir um objetivo. Dentre os métodos de planejamento experimental disponíveis na literatura, o planejamento fatorial é o mais indicado quando se deseja estudar os efeitos de duas ou mais variáveis de influência, sendo que em cada tentativa ou réplica, todas as combinações possíveis dos níveis de cada variável são investigadas (BARROS NETO et al., 1996). O caso mais simples de planejamento fatorial descrito na literatura é aquele em que cada fator k está presente em apenas dois níveis (o experimento aplicado neste trabalho), ou seja, em um experimento com k fatores (ou variáveis) e dois níveis.

Os valores dos parâmetros do SA foram definidos quando ainda estava sendo utilizado apenas um AP. Para o teste, foram considerados os valores da quantidade de vizinhos, temperatura inicial, o fator de resfriamento e a quantidade máxima de perturbações. O valor do máximo de iterações foi definidos em 50 e a posição do AP foi definida como sendo [0, 0]. Foi desenvolvido um *script* em Python que automatizasse o processo. Como são quatro parâmetros que se deseja saber os valores e são utilizados dois níveis de profundidade, foi utilizado o conceito de tabela verdade, contendo 2^4 linhas fazendo com que todas as possibilidades de parâmetros sejam testadas.

Com a coleta das informações obtidas pela saída do *script* em Python, os valores foram adicionados em uma planilha do Excel cedida pelo orientador. A tabela realiza todos os cálculos necessários para saber qual parâmetro é mais propício a se utilizar. A tabela abaixo apresenta a configuração da primeira iteração do planejamento 2K com os fatores e níveis bem definidos.

Tabela 1 – Fatores para parâmetro para o *Simulated Annealing* na primeira iteração

		FATORES				**FATORES**				**FIXO**
k=	4	A (n de vizinhos)		B (temperatura inicial)		C (fator resfriamento)		D (perturbações)		Iteracoes
n=	Niveis	1	20	1	100	1	75%	1	10	1000
		2	40	2	200	2	95%	2	20	1000

O resultado dado pela planilha é o seguinte:

Tabela 2 – Candidatos a parâmetros na primeira iteração

	CANDIDATOS A PARÂMETRO			
Frações	vizinhos	temperatura	resfriamento	pertubações
Função Objetivo	1,4%	1,0%	1,2%	1,0%
Tempo	0,9%	0,8%	1,2%	1,0%
MÉDIA	1,2%	0,9%	1,2%	1,0%

Com o resultado acima, é possível notar que o melhor candidato é o número de vizinhos e o resfriamento. Ainda que o valor de ambos tenha sido o mesmo, o número de vizinhos acaba sendo melhor, pois consumiu uma fração menor do tempo total. O número de vizinhos foi aumentado para ter um resultado melhor ainda; os valores da temperatura inicial também foram aumentados e o fator de resfriamento foi decrementado junto com o número de perturbações. Modificando os parâmetros, se torna propício a um novo melhor conjunto de valores do SA. Os novos valores da segunda iteração do planejamento fatorial 2K podem ser vistos abaixo:

Tabela 3 – Fatores para parâmetro para o *Simulated Annealing* na segunda iteração

		FATORES				**FATORES**				**FIXO**
k=	4	A (n de vizinhos)		B (temperatura inicial)		C (fator resfriamento)		D (perturbações)		Iterações
n=	Niveis	1	40	1	150	1	75%	1	10	1000
		2	80	2	300	2	85%	2	15	1000

O resultado dado pela configuração acima é a seguinte:

Tabela 4 – Candidatos a parâmetros na segunda iteração

	CANDIDATOS A PARÂMETRO			
Frações	vizinhos	temperatura	resfriamento	pertubações
Função Objetivo	1,4%	1,0%	1,3%	1,0%
Tempo	1,2%	0,9%	1,2%	1,1%
MÉDIA	1,3%	0,9%	1,2%	1,0%

Como na iteração anterior, o melhor candidato a manter o seu valor é o número de vizinhos. Com a análise feita, o número de vizinhos no segundo nível e a temperatura

inicial também obtiveram um bom resultado no segundo nível. O valor do resfriamento no segundo nível manteve sua média, enquanto o número de perturbações em ambos os níveis, teve um resultado inferior ao anterior, sendo necessário modificá-lo na próxima iteração. Abaixo é possível ver como ficaram definidos os parâmetros para a terceira iteração e último teste.

Tabela 5 – Fatores para parâmetro para o *Simulated Annealing* na terceira iteração

		FATORES				**FATORES**				**FIXO**
k=	4	A (n de vizinhos)		B (temperatura inicial)		C (fator resfriamento)		D (perturbações)		Iteracoes
n=	Niveis	1	80	1	300	1	80%	1	5	1000
		2	60	2	600	2	85%	2	10	1000

De acordo com a configuração acima são dados os candidatos a parâmetros:

Tabela 6 – Candidatos a parâmetros na terceira iteração

	CANDIDATOS A PARÂMETRO			
Frações	vizinhos	temperatura	resfriamento	pertubações
Função Objetivo	1,4%	1,0%	1,3%	1,0%
Tempo	1,2%	0,9%	1,1%	1,1%
MÉDIA	1,3%	0,9%	1,2%	1,0%

Os resultados apresentados acima passaram por três refinamentos. Assim, como dito no início desta seção, o teste foi realizado com o intuito de aprimorar os parâmetros do *Simulated Annealing* com apenas um AP. A estratégia para a versão final e com múltiplos *access points* é definir também como um fator (variável), o valor do raio de perturbação e fazer com que ele seja uma porcentagem da largura da matriz de propagação, tornando-o dinâmico de acordo com o tamanho da planta. A tabela abaixo mostra de uma forma melhor como ficaram os parâmetros utilizados na metaheurística:

A resolução foi dada em metros e transformada de uma forma compatível com o processamento e a duração esperada para analisar cada solução. Os parâmetros definidos para a alocação de vários APs foi um compilado dos parâmetros utilizados no *Simulated Annealing* com um AP juntamente com testes feitos repetidas vezes.

Tabela 7 – Parâmetros definitivos utilizados na metaheurística

Raio de perturbação	**Número máximo de iterações**	**Número máximo de perturbações**	**Número máximo de vizinhos**	**Decaimento da temperatura (Alpha)**	**Temperatura inicial**
WIDTH * 0.025 (2.5% da largura da matriz)	600 * nº de APs	5	80	85%	300 * 2 (600)

4.5 Avaliação da solução

Nesta seção apresentaremos como foram implementadas as técnicas utilizadas para avaliações com um ou mais *access points* bem como o aperfeiçoamento da função objetivo.

4.5.1 Avaliação da solução com um AP

Para a implementação do SA, se faz importante a decisão de uma boa função objetivo. A função objetivo está diretamente ligada à eficácia com que o *Simulated Annealing* irá buscar seu ótimo global. Inicialmente a função objetivo foi implementada de forma bem prática: como a matriz gerada continha em cada posição o resultado da potência do sinal, conforme ele se propagava pelo ambiente, para representar uma indicação numérica da cobertura do sinal *Wi-Fi* foi feita a soma de todos os valores da matriz. Num primeiro momento, tentamos somar os valores da potência do sinal em mW mas em função do decaimento exponencial do sinal com o distanciamento do transmissor, a precisão de um número de ponto flutuante chegava rapidamente ao limite de seus 32 bits. Tão pouco adiantou aumentar a precisão para 64 bits ou mais. Curiosamente, uma função objetivo baseada na soma dos valores em miliwatts dos sinais simulados era demasiado custosa de se calcular e trouxe pouco benefício quanto à exploração do espaço de soluções pelo *Simulated Annealing*.

Considerando o acima exposto, decidimos por manter os valores da matriz resultante em dBm, uma escala logarítmica que representa a ordem de magnitude da intensidade do sinal. Assim, apesar de não ser conceitualmente correto somar valores de potência em dBm, nosso objetivo não era ter o valor exato em mW da soma de toda a energia irradiada no ambiente simulado, mas sim ter uma noção da ordem de magnitude do quanto de sinal útil aquela simulação do ambiente apresentava. Observe que as interfaces de rede sem fio tipicamente apresentam sensibilidade mínima em torno de -85 dBm a -100 dBm (em torno de $3,162 \times 10^{-12}W$ a no mínimo $1 \times 10^{-13}W$), assim sendo, quanto mais "positivo"fosse o valor da função objetivo avaliada, maior seria o montante da potência dos sinais que cobriam o ambiente simulado. Levando em considerações tais fatos, experimentamos com diversas maneiras de sumarizar a matriz em um só valor para ser avaliado como função objetivo e curiosamente, aquele que apresentou o melhor custo-benefício foi justamente a soma simples de todos os valores da matriz, mesmo que as informações nela armazenada estivessem em dBm. Numa inspeção visual da solução proposta pelo *Simulated Annealing*, com a colorização dos valores de dBm em um gradiente de cores de acordo com o posicionamento do *Access Point*, podemos observar que a metaheurística conduzia as soluções candidatas para a região central do ambiente 2D, como seria desejado.

4.5.2 Aperfeiçoamento da função objetivo

Assim que foi implementada uma primeira versão das etapas de entrada e processamento do arquivo *.dxf*, foram conduzidos testes no simulador e na otimização, de maneira a checar a proximidade de dados reais coletados com a simulação e o quanto a otimização conseguiria melhorar a solução, direcionada por meio de sua função objetivo. Tal continuidade na condução dos testes permitiram alterações e aperfeiçoamentos na função objetivo da metaheurística de otimização e na simulação das características do ambiente e da propagação dos raios (sinais de RF). Foram comparados os resultados simulados com resultados reais coletados nas dependências do campus.

4.5.3 Avaliação da solução para dois ou mais APs

Como desde o princípio, a implementação da função objetivo para apenas um AP visava maximizar a cobertura do sinal WiFi para o ambiente. Isso não foi diferente para a simulação de mais um AP. Foram utilizadas técnicas de manipulação de grandezas em dBm para obter o resultado esperado. Como dito no parágrafo anterior, a manipulação com a soma dos dados em dBm estava errada. Como uma proposta de solução para esse problema, antes de realizar a soma dos valores da matriz, a potência do sinal em questão era convertido de dBm para mw, fazendo com que a operação ficasse correta e uma verificação da potência recebida com a sensibilidade do access point também foi feita para penalizar zonas em que o sinal era tão fraco que não se tornava uma opção válida. Com a conversão dos valores matriz e a verificação da qualidade do sinal adicionada, o tempo de execução do algoritmo aumentou consideravelmente, fazendo com que outra alternativa fosse buscada.

Diante de uma opção que demandava mais tempo para que fosse processada, outra estratégia para a função objetivo foi implementada. Quando se tem mais um AP para buscar os seus respectivos pontos ótimos, somar suas matrizes de resultados também não é algo que dê um resultado correto. Foi, então, implementado um método que realizava a sobreposição das matrizes de resultado da propagação. Para o SA saber se a escolha era ruim ou boa, precisava transformar essa escolha em um número. Nesse sentido, de acordo com as n matrizes de resultado de n access points, foi necessário realizar suas sobreposições.

Tal método recebe como parâmetro uma lista (ou array, ou vetor, em Python são o mesmo tipo de dados) contendo a matriz resultado da simulação de todos os APs e uma variável size que corresponde à quantidade de APs no ambiente. Posteriormente é feito um laço de repetição percorrendo a lista de matrizes e guardado as matrizes mínimas e máximas utilizando respectivamente os métodos minimum e maximum da biblioteca Numpy. No final da iteração é feita então a razão dasdas matrizes mínimas e máximas, fazendo com que a operação de subtração (e sobreposição) dos valores resultantes em dBm

fosse correta. O método que realiza essa operação pode ser visto abaixo.

```
def sobrepoe_solucoes_DIV_dBm(propagation_array, size):
    if size == 1:
        return propagation_array[0]
    matrixMin = propagation_array[0]
    matrixMax = propagation_array[0]
    for i in range(1, size):
        matrixMin = np.minimum(propagation_array[i], matrixMin)
        matrixMax = np.maximum(propagation_array[i], matrixMax)
    # pois ao subtrair dBm, deve ser o maior/menor
    sub = np.divide(matrixMax, matrixMin)
    return sub
```

Com as matrizes sobrepostas, o método acima retorna à matriz resultante. O método que realiza a avaliação da lista de APs fará uso da matriz para obter um valor a se comparar no processo de busca de um ponto melhor no Simulated Annealing. Os cálculos para a penalização de áreas onde o sinal tem uma qualidade de inferir a sensibilidade do equipamento não foram implementados nessa estratégia.

4.6 Paralelização em GPU

A decisão de fazer uso da GPU para ajudar no desempenho foi eficiente. A transição de um código sequencial para um código paralelizado foi feita aos poucos. No início, foi importado da biblioteca numba o módulo *jit* e adicionado sob todos os cabeçalhos dos métodos a anotação *@jit*. Apenas com isso, o código será compilado em código de máquina no momento que for executado (*just-in-time*). Abaixo é possível ver um exemplo de um método que será compilado para código de máquina utilizando a anotação *@jit*:

```
@jit
def calc_distance(x1, y1, x2, y2):
    return sqrt(pow((x1 - x2), 2.0) + pow((y1 - y2), 2.0)) * precisao
```

Após utilizar as anotações nos métodos, foi escolhido o método que demandava mais recursos e demandava todo o tempo. Com uma rápida olhada no código, foi possível ver que o método era o que realizava a avaliação da solução em ponto dentro do *Simulated Annealing*. Neste momento, o método calcula a intensidade do sinal para cada ponto da matriz e, para cada ponto, calcula a absorção das paredes, percorrendo uma lista com todas as paredes do ambiente. Quanto maior o detalhamento do ambiente, mais linhas o ambiente terá e maior será a lista de paredes a se verificar absorções.

Posteriormente à decisão do método a deixar a cargo da GPU realizar os cálculos, todo o código foi transcrito de forma a dividir todo o trabalho entre os núcleos de CUDA disponíveis. No código abaixo é possível ver, no método que realiza a propagação do sinal, que uma matriz utilizando a biblioteca numpy é criada, as dimensões dos *blocks* e *grids* (os valores foram escolhidos empiricamente, dependem do hardware utilizado e do problema a ser trabalhado). Logo após, a matriz é enviada para a GPU para ser compartilhada entre *blocks* e *threads*. O método que realiza o cálculo da intensidade do sinal é chamado de acordo com os *blocks* e *grids* definidos. Então, a matriz que foi enviada para a GPU é trazida de volta e retornada para continuar com os cálculos.

```
@jit
def simula_propagacao_gpu(pointX, pointY):
    g_matrix = np.zeros(shape=(WIDTH, HEIGHT), dtype=np.float32)
    blockDim = (48, 8)
    gridDim = (32, 16)
    d_matrix = cuda.to_device(g_matrix)
    simulate_kernel[gridDim, blockDim](pointX, pointY,
                                       d_matrix, floor_plan)
    d_matrix.to_host()
    return g_matrix
```

Na Figura 12 é possível ver facilmente como é a divisão dos *blocks*, *grids* e *threads*.

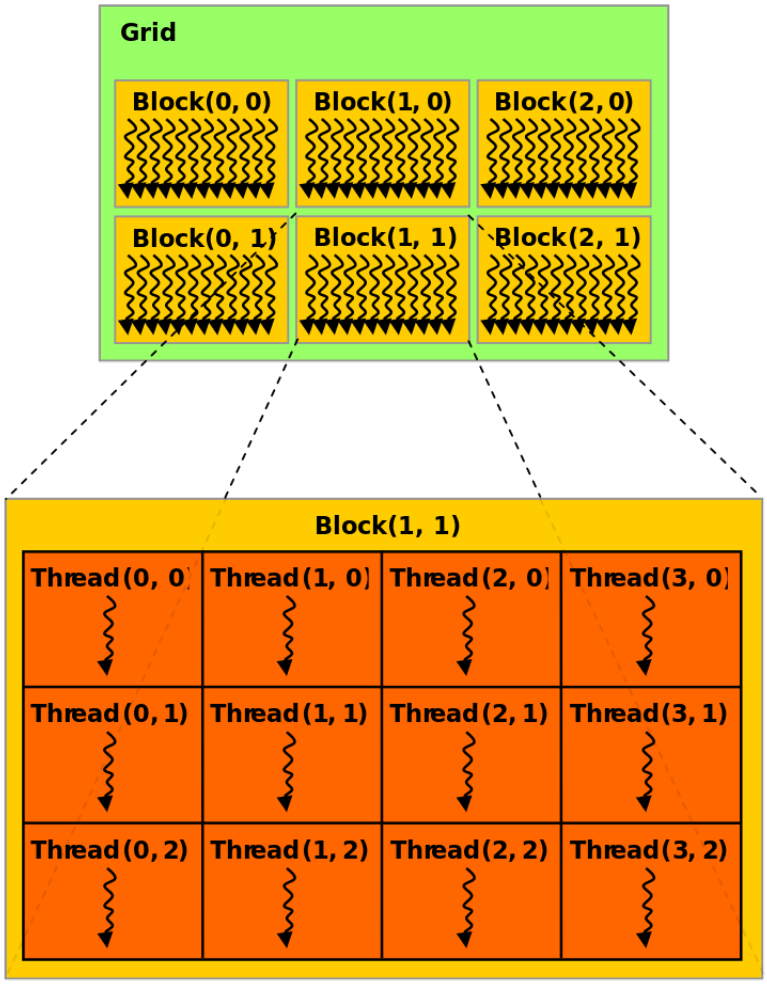
O equipamento utilizado para o desenvolvimento utiliza uma placa de vídeo NVIDIA modelo GeForce GT 740M[11] com 2GB de memória dedicada; possui 384 núcleos CUDA, com um clock básico de 993 MHz, 1300 milhões de transistores e com vazão do barramento PCIe de até 80 Gb/s.

O método "*simulate_kernel_gpu*"foi escolhido para trabalhar com a matriz de propagação. Nele, a matriz é percorrida de acordo com os valores dos blocks e grids informados no método anterior. Para cada célula da matriz é executado o método "*propagation_model_gpu*", o qual, segundo o próprio nome já diz, realiza o cálculo do modelo de propagação utilizando a GPU e está diretamente ligado ao "*propagation_model*"que obtém valores do NP-Log e os miliwatts a serem subtraídos do cálculo da perda por paredes. O código referente ao "*simulate_kernel_gpu*"pode ser visto abaixo.

```
@cuda.jit
def simulate_kernel(apX, apY, matrix_results, floor_plan):
    startX, startY = cuda.grid(2)
    gridX = cuda.gridDim.x * cuda.blockDim.x
    gridY = cuda.gridDim.y * cuda.blockDim.y
```

[11] Disponível em <https://www.geforce.com/hardware/notebook-gpus/geforce-gt-740m>. Acesso em: 29 out. 2017.

Figura 12 – Divisão interna da GPU dos blocks, grids e threads



Fonte: <https://en.wikipedia.org/wiki/Thread_block>

```
for x in range(startX, WIDTH, gridX):
    for y in range(startY, HEIGHT, gridY):
        matrix_results[x][y] = propagation_model_gpu(x, y,
                                                     apX, apY, floor_plan)
```

O cálculo do valor do sinal naquele ponto é $O(k)$ pois consiste em um operação envolvendo multiplicação, divisão, exponenciação/logaritmo, portanto, assintoticamente é: $O(1)$. Em cada simulação deve-se calcular o valor para a propagação em cada ponto de uma matriz NxM. Considerando que uma das dimensões é maior que a outra, podemos dizer que o custo é NxN: $O(n^2)$ e $O(1 \times n^2) = O(n^2)$ para aplicar a operação em todos os pontos da matriz. Mas, entre cada ponto da matriz (PM) e o AP, podem haver k paredes, que deve-se verificar se há ou não interseção entre a reta AP-PM e a reta formada pela

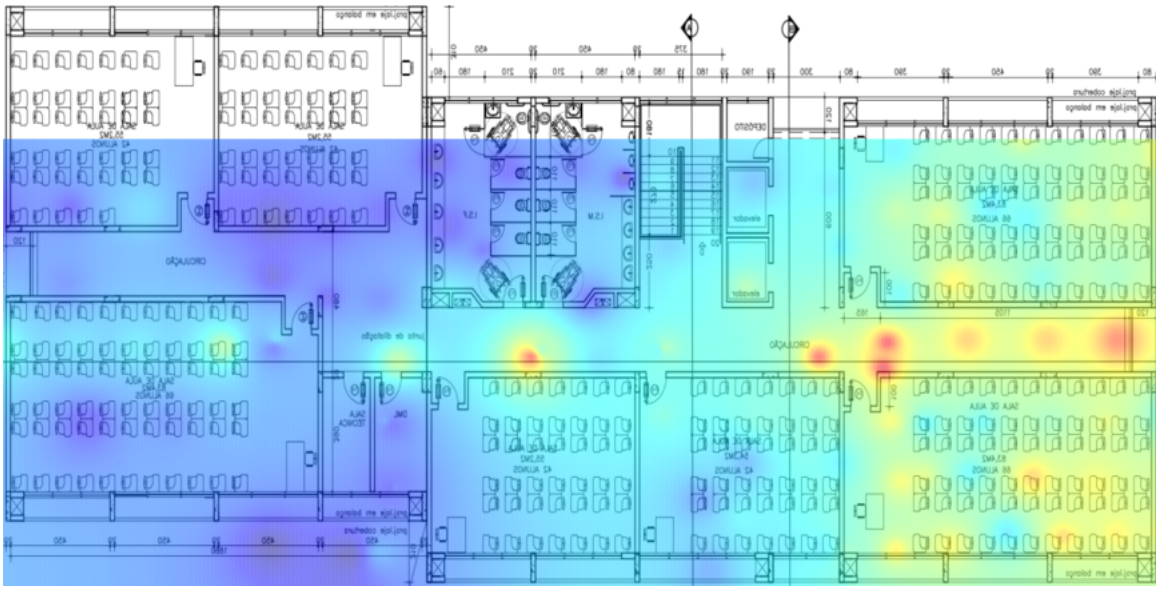
parede. Tal operação, utilizando aquela fórmula de geometria analítica, custa $O(1)$ para cada parede e $k \times O(1)$ para verificar cada parede. Como k é suficientemente grande em relação a n, podemos considerar o custo de verificar se há interseção com paredes como O(n). Para fazer essa verificação de cada parede para cada ponto da Matriz, temos então: $O(n) \times O(n^2) = O(n^3)$. Quanto ao *Simulated Annealing* (SA), a cada rodada ele aleatoriza uma solução vizinha (v) e avalia-a (fo), e como o SA tem uma quantidade finita e decremental de iterações em função do fator de resfriamento, a quantidade de vizinhos explorados é em torno de $v = log(n)$. Assim, a complexidade do SA é em torno de $O(fo * v)$, ou seja: $O(fo \times logn)$. Como a função objetivo (fo) consome $O(n^3)$, teríamos: $O(n^3 * logn)$. É importante notar que a anotação deste método é diferente das demais que utilizam o *@jit* para gerar código de máquina. A anotação *@jit* é o caminho geral do compilador, que pode ser orientado opcionalmente para um dispositivo CUDA. A anotação *@cuda.jit* é efetivamente o dialeto de núcleo de mais baixo nível do CUDA para Python que a empresa *Continuous Analytics* desenvolveu. Assim é possível receber suporte para variáveis embutidas do CUDA como o *threadIdx* e especificadores de espaço de memória compartilhada. Caso seja necessário escrever um *kernel* CUDA em Python e compilá-lo e executá-lo (como ocorreu nesse caso), deve-se usar a anotação *@cuda.jit*. Caso contrário, se for necessário apenas acelerar algum método existente, deve-se usar somente *@jit* com uma anotação alvo do CUDA.

4.7 Calibração dos modelos

Durante e após a implementação dos modelos de perda e atenuação do sinal *wireless* e modelagem de propagação dos sinais pelo ambiente, foram comparados os resultados da simulação com o resultado real esperado. Para tal, pode-se utilizar o atual posicionamento dos APs Wi-Fi no campus e realizar experimentos de *Wireless Site Survey*, ou seja, medições reais da intensidade de sinal em determinados pontos do ambiente analisado.

Tais resultados foram então comparados com os resultados previstos pela simulação (através da visualização gráfica da propagação no ambiente), para que os modelos de propagação e características do modelo espacial pudessem ser calibrados, num processo de melhoria contínua visando ao aumento da precisão da solução a ser apresentada. É válido reiterar que os parâmetros do *Simulated Annealing* e da função *NP-Log* foram calibrados para melhor representar o ambiente do IFMG *campus* Formiga, de acordo com seus materiais de construção, espessura de suas paredes, pisos e teto.

Figura 13 – Captura realizada no Bloco C, para um AP "18:8B:9D:69:E8:B2"posicionado em (x=660,y=260) e irradiando no Canal 11 (2.462 GHz)



Fonte: Elaboração do autor

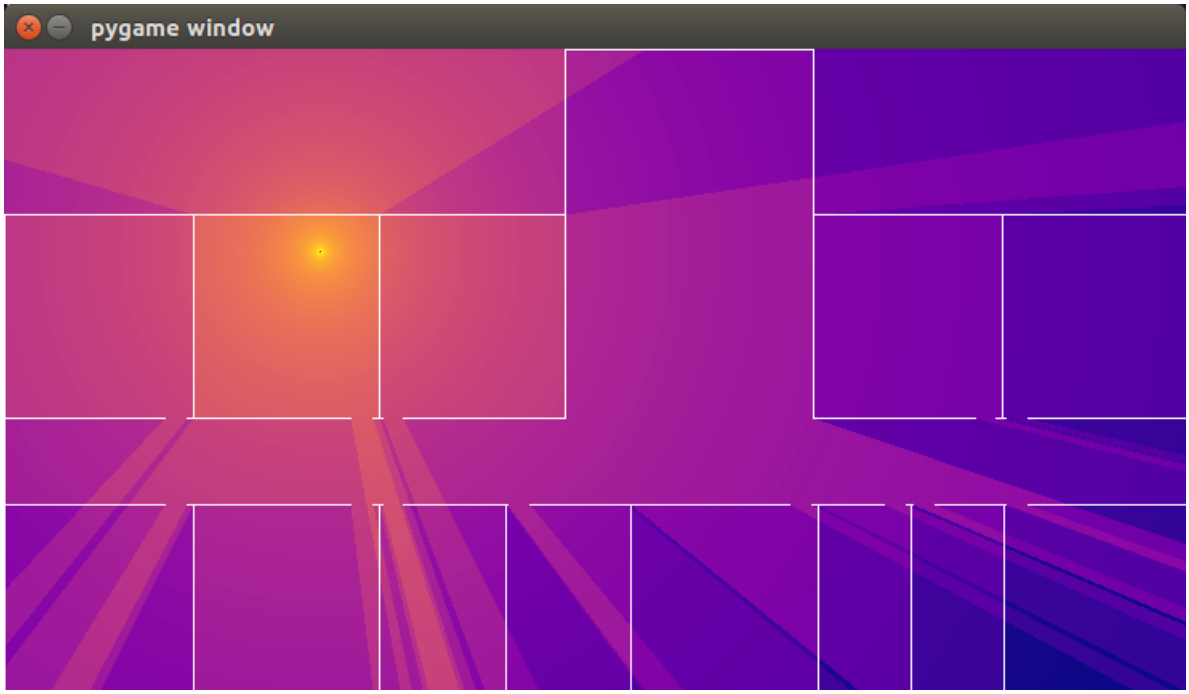
5 RESULTADOS E ANÁLISE

Neste capítulo serão mostrados os resultados obtidos a partir da implementação de toda a fundamentação teórica e a explicação dos procedimentos desenvolvidos. Será feita também uma análise de custo computacional e um ajuste de curvas utilizando os softwares *cProfile* e o *R-Project*, respectivamente.

5.1 Simulação da propagação de sinais *wireless*

As primeiras simulações gráficas realizadas tiveram como objetivo apenas exibir o comportamento do espectro do sinal *WiFi*, realizando o tratamento das paredes e, assim, demonstrar graficamente o seu valor dentro da matriz de propagação. A Figura 14 mostra a simulação da propagação de sinais de *microondas* na planta baixa do bloco A do IFMG *campus* Formiga. Para a obtenção de tal resultado da simulação e dos próximos resultados que serão mostrados neste capítulo, foi utilizado o código em Python que faz uso da GPU.

Figura 14 – simulação da propagação de sinais de microondas no edifício utilizando versão inicial do algoritmo.

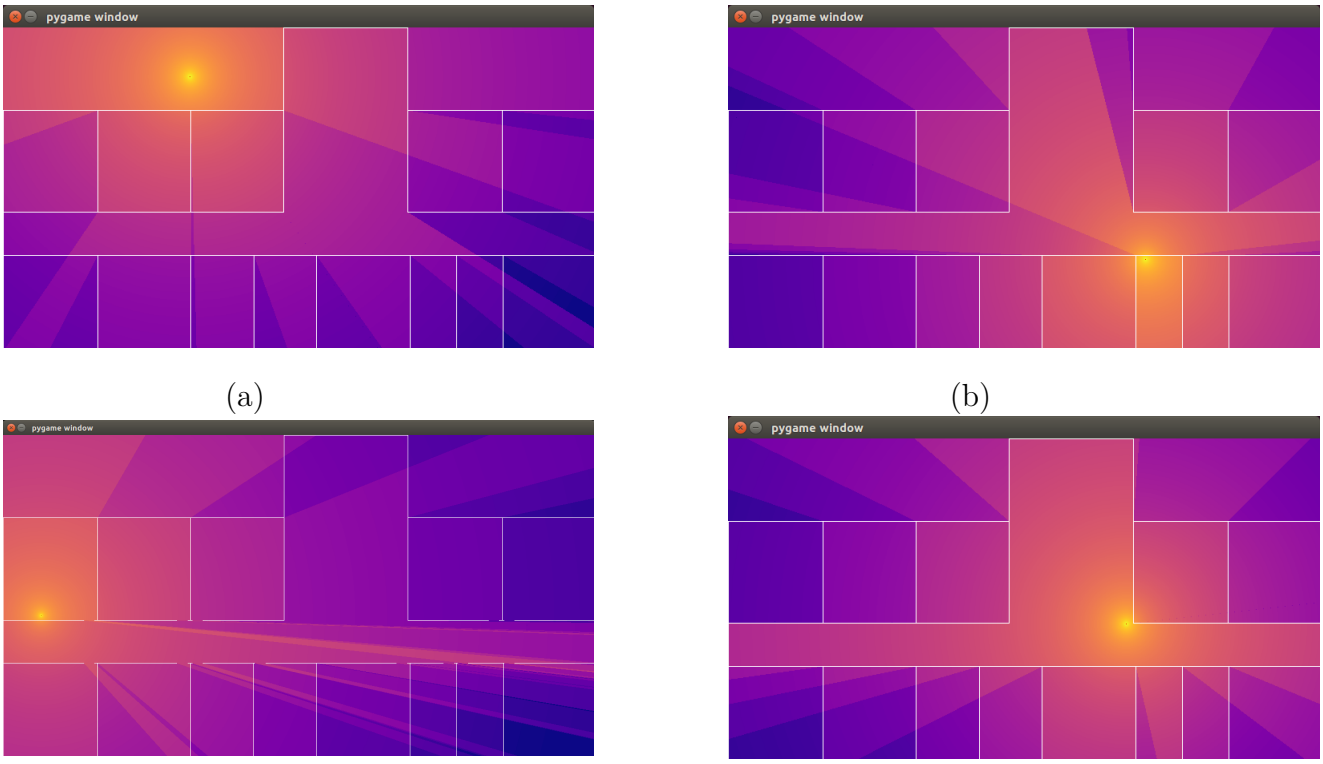


Fonte: Elaboração do autor

A seguir temos quatro imagens, as quais a, b e d não possuem as portas ou janelas implementadas, ou seja, os cômodos são tratados como ambientes fechados. Já a imagem c,

por sua vez, possui o resultado gráfico da simulação e tem aberturas nas paredes, simulando as portas. Observe que quanto maior é o número de paredes, com a atenuação do sinal, sombreamentos são criados.

Figura 15 – simulação da propagação inicial de sinais de microondas utilizando versão inicial do algoritmo.



Fonte: Elaboração do autor

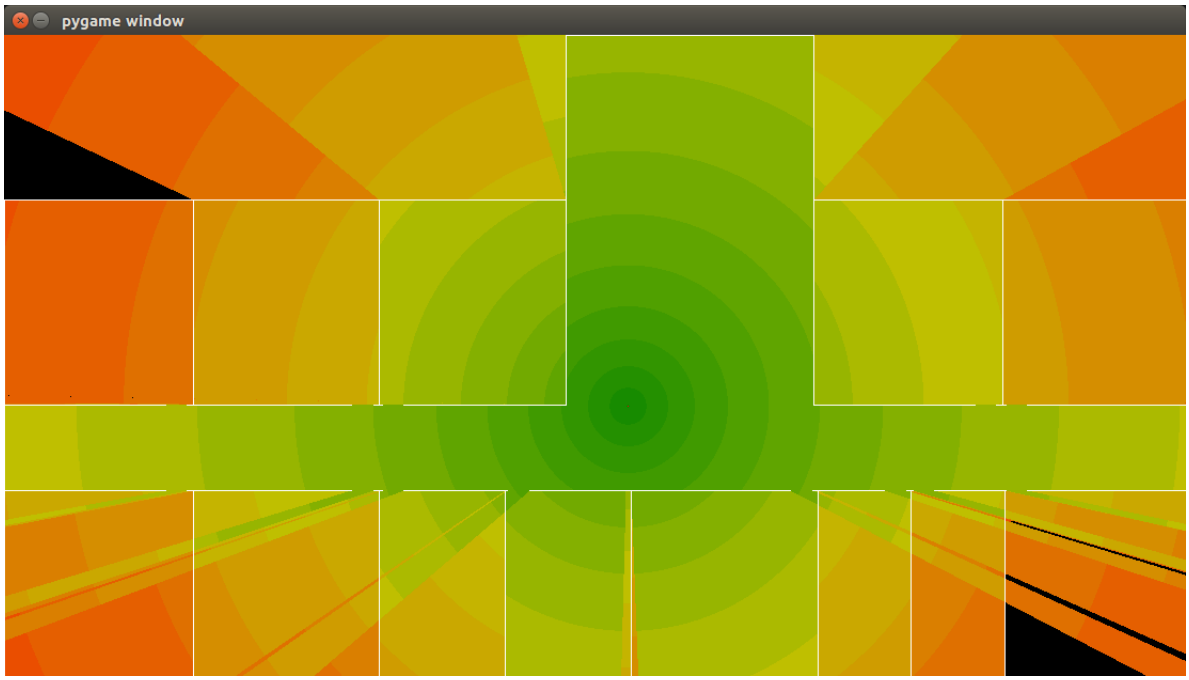
Neste sentido, é importante ressaltar que a figuras acima são resultados dos primeiros testes executados. A escala de cores utilizada foi a plasma, variando das cores laranja até o azul escuro. Na próxima seção mostraremos os resultados de testes efetuados utilizando ainda um *access point* com uma nova escala de cores.

5.2 Wi-Fi Placement para 1 AP

A figura abaixo, se comparada com as figuras do tópico anterior, possuem cores mais condizentes com os resultados obtidos, além de uma restrição na parte gráfica, destacada com a cor preta nas áreas onde o sinal é menor que a sensibilidade do AP.

É possível notar que o SA definiu como um caso ótimo, nesta situação, um ponto próximo ao meio da planta. Com a posição escolhida, a cobertura do sinal se propaga em áreas próximas à biblioteca e sala de estudo, que contém o maior fluxo de pessoas

Figura 16 – Simulação da propagação de sinais de microondas no bloco A utilizando 1 AP, sua absorção e limiar de sensibilidade.



Fonte: Elaboração do autor

neste bloco que utilizam o acesso à internet. Na figura, o único ponto que deixa a desejar corresponde à sala de estudos.

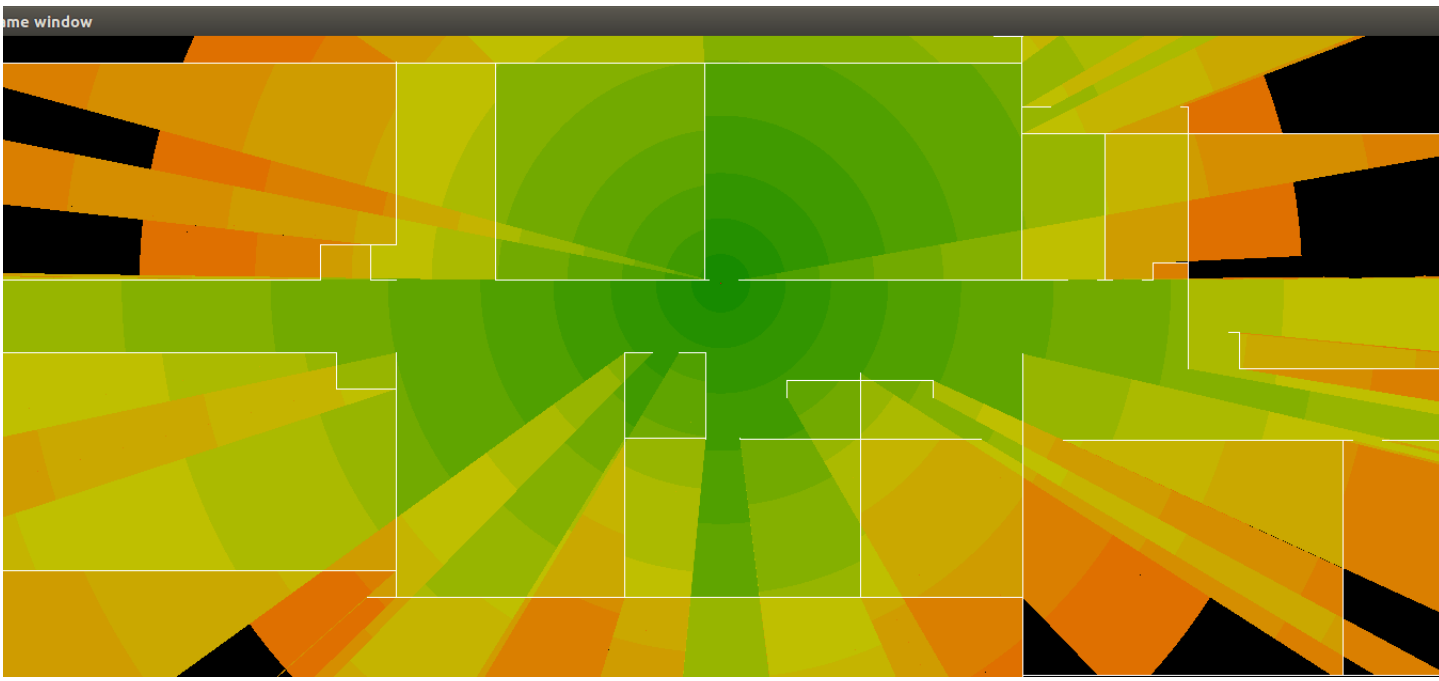
As plantas dos pisos 1, 2 e 3 do bloco C foram fornecidas e convertidas para o formato *.dxf* ao final da implementação. Tais arquivos foram simulados e obtidas suas representações gráficas. A imagem abaixo mostra a simulação da planta baixa no bloco C utilizando 1 AP.

Como pode ser visto na imagem, quando utilizada a busca pelo ponto ótimo com um *access point*, ele tendeu a permanecer ao centro da planta. Com o resultado dado para esta simulação com um AP, os cômodos mais ao centro são bem atendidos com o sinal *Wi-Fi*, mas ao contrário dos mais afastados, não irão usufruir de uma boa qualidade de sinal e esse é retirado da apresentação gráfica e dos cálculos da avaliação da função objetivo. O resultado toma essa característica por não realizar uma verificação das áreas mais importantes. Caso tivesse essa funcionalidade implementada, a metaheurística poderia fazer com que o AP cobrisse as áreas mais críticas.

5.3 Wi-Fi Placement para 2 ou mais APs

Feita a propagação do sinal *wireless* para um AP nos bloco A, foi validada a implementação da simulação com dois *access points*. A imagem abaixo mostra como foi o

Figura 17 – simulação da propagação de sinais de microondas no bloco C utilizando 1 AP.



Fonte: Elaboração do autor

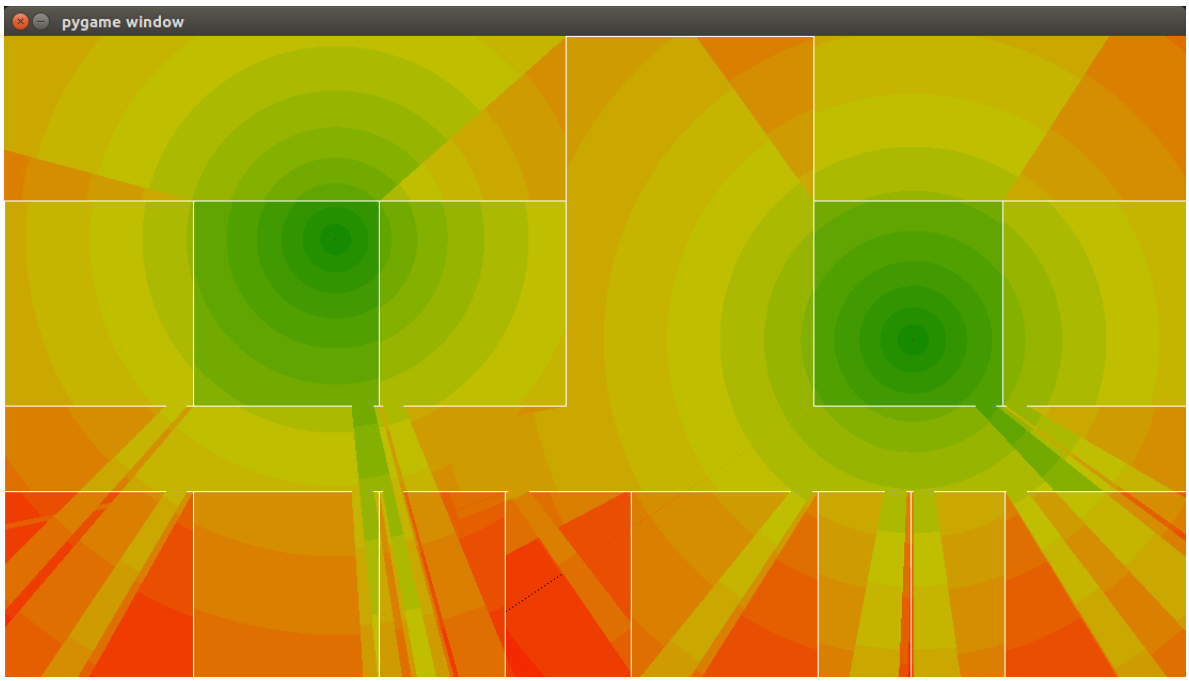
resultado da simulação do *Wi-Fi* no bloco A.

Como pode ser visto, a heurística pode então dar, como o resultado, dois pontos que visualmente aparentam ser muito bons. Na imagem acima pode ser visto que os dois APs foram alocados com uma distância de folga, fazendo com que possam aproveitar a propagação do espectro no ambiente. Com a escolha sugerida pelo SA, pode-se também notar que as salas da parte inferior não estão recebendo uma qualidade de sinal como na sala em que os APs estão instalados, mas mesmo assim, toda a área está recebendo sinal, não havendo zonas onde a potência recebida é inferior à sensibilidade do equipamento.

O mesmo teste foi realizado com a planta baixa do bloco C. Com a imagem abaixo, pode ser visto quais são os pontos sugeridos pelo SA na busca para a alocação dos mesmos.

Com o resultado dado pelo SA no bloco C, pode-se notar que foi possível cobrir praticamente toda a área. Tiveram algumas áreas com pontos cegos porém, podem ser consideradas de certa forma irrelevantes quando o objetivo é obter a maior cobertura do sinal. Um resultado ainda melhor pode ser encontrado se for feito um maior refinamento utilizando o planejamento fatorial 2K.

Figura 18 – simulação da propagação de sinais de microondas no bloco A utilizando 2 APs.



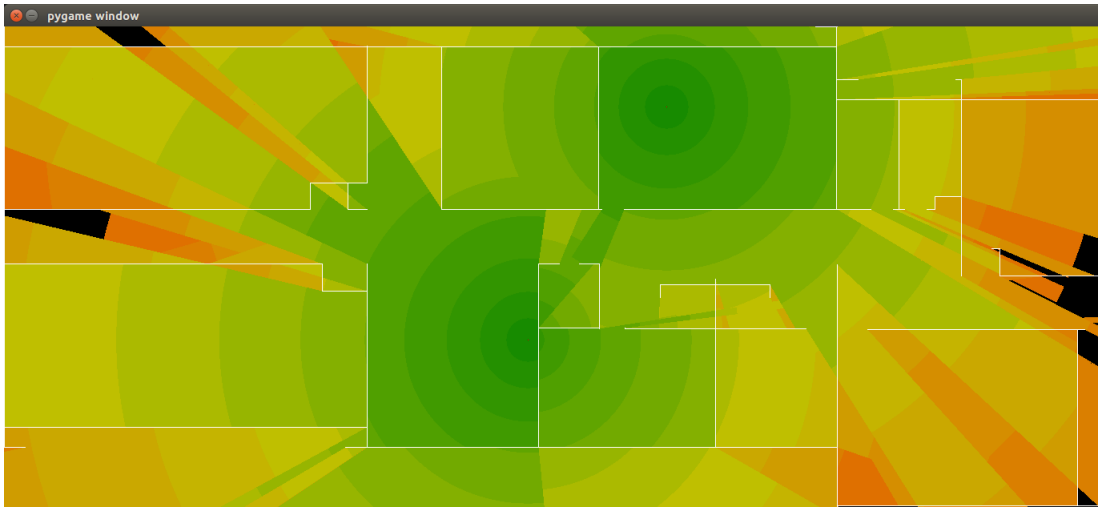
Fonte: Elaboração do autor

5.4 Análise da utilização de recursos

A fim de buscar um melhor visualização do fluxo de execução do algoritmo, foi utilizada uma ferramenta de profile, chamada *cProfile*, para geração do arquivo com extensão *.cprof* e o software *pyprof2calltree* para realização desta análise. Com tais ferramentas é possível obter análises da porcentagem de tempo gasto em cada método do algoritmo, de acordo com o tempo total. Como é possível ver abaixo, foi gerada uma imagem de saída utilizando o software *pyprof2calltree* com o fluxo de execução do método *Run*, que é responsável desde os cálculos de proporção da matriz/planta, leitura de arquivos de entrada, até a etapa final, com a exibição do resultado em forma gráfica, utilizando o PyGame.

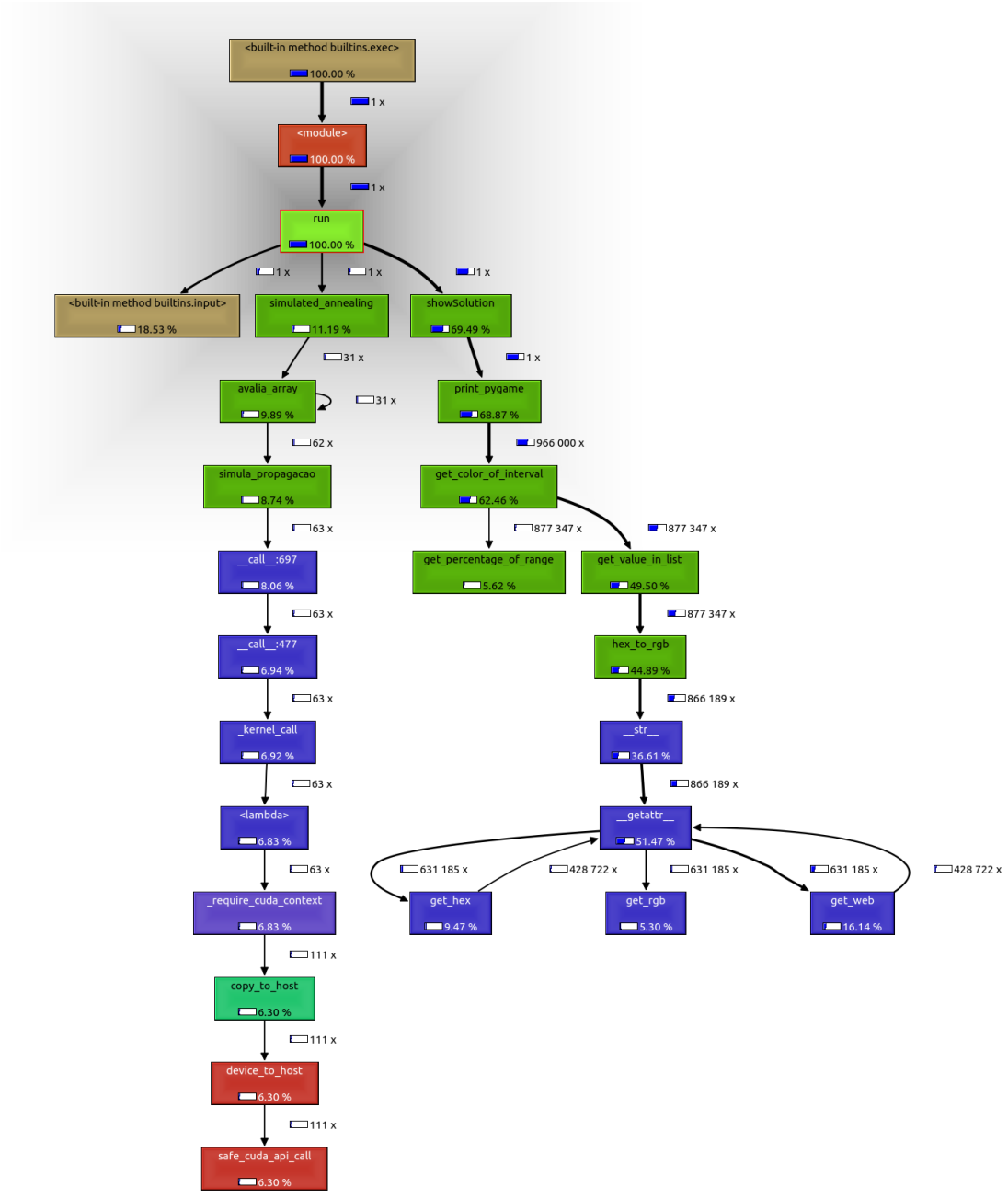
Assim, cada retângulo do grafo acima representa um método implementado, sendo eles referentes aos métodos implementados utilizando Python e aos métodos de bibliotecas utilizadas no desenvolvimento da aplicação. Também pode ser visto que cada aresta no grafo possui uma barra de porcentagem representando visualmente o seu tempo gasto na aplicação, juntamente com o número de vezes que o método foi chamado. Para realizar a interpretação do arquivo de saída, basta utilizar o comando *pyprof2calltree -k -i PlacementAPs.cprof* no terminal com o arquivo gerado ao fim da simulação, nesse caso, *PlacementAPs.cprof*.

Figura 19 – simulação da propagação de sinais de microondas no bloco C utilizando 2 APs.



Fonte: Elaboração do autor

Figura 20 – grafo do ciclo de execução do algoritmo



Fonte: Elaboração do autor

6 CONSIDERAÇÕES FINAIS

O *software* desenvolvido neste trabalho é capaz de (i) receber como entrada uma representação espacial 2D do ambiente (planta-baixa de um piso do edifício), podendo também ser informada a posição inicial dos APs como parâmetros adicionais para o algoritmo; (ii) realizar a simulação da propagação dos sinais de microondas dos APs Wi-Fi pelas dependências do edifício; (iii) aplicar a metaheurística computacional para explorar o espaço de soluções do posicionamento de APs; (iv) fornecer como saída a proposta de novo(s) posicionamento(s) dos APs visando ampliar a cobertura do sinal *Wi-Fi* para aquele ambiente.

O trabalho realizado foi disponibilizado de forma gratuita utilizando Licença Pública Geral GNU — GPL no site *github.com*. Até onde pudemos verificar, não há disponível um software livre, gratuito e de código-fonte aberto para *Wireless AP Placement*, apenas soluções proprietárias, comerciais e custosas. Para reforçar a relevância deste trabalho, ressaltamos que ele possibilita testar disposições de APs sem o custo operacional de fisicamente movê-los, de maneira a propor uma disposição espacial dos mesmos que forneça uma maior cobertura e intensidade de sinal dentro do ambiente simulado.

Para trabalhos futuros, é necessária a realização de tratamento da interferência do canal, além de uma maior coleta de dados para obtenção de parâmetros mais precisos o modelo de regressão logístico *(NP-Log)*. Também, sugere-se que seja utilizado mais de um modelo de propagação, complementando-se, visto que aparentemente um único modelo não realiza estimativas tão boas tanto para distâncias perto e longe. Portanto, sugere-se utilizar dois modelos de propagação, um para perto (1-10 m) e outro para mais longe (10-100 m), numa abordagem híbrida. Uma outra abordagem poderia ser a aplicação de técnicas que envolvem simulação via *Ray-Tracing*, podendo ser incluída em pesquisas posteriores. Ainda, caso haja demanda, sugere-se a realização de simulação de propagação de sinais em ambiente 3D, com um maior aproveitamento dos recursos disponibilizados pela biblioteca CUDA, objetivando uma maior precisão da obtenção dos pontos cegos dentro de cômodos. Para se obter resultados que possibilitem uma melhor busca pelos pontos de acesso para dois ou mais APs, novas técnicas deverão ser utilizadas para o cálculo da função objetivo.

Referências