LVI SBPO SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL
DE 04 ATÉ 07 NOVEMBRO 2024
FORTALEZA | CEARÁ

Hiper-heurísticas aplicadas ao Problema do Posicionamento Automático de Pontos de Acesso de Redes Sem Fio em Ambientes Internos

Calebe Oliveira de Figueirêdo[1], **Thiago Gouveia**[1], **Andressa Theotônio Alves de Figueirêdo**[1]
[1]Instituto Federal da Paraíba, João Pessoa — PB
Autores para correspondência: calebeof14@gmail.com, thiago.gouveia.da.silva@gmail.com, andressatheotonio@gmail.com

RESUMO

O projeto de redes sem fio tornou-se crucial com a popularização de dispositivos móveis, destacando-se pela escalabilidade e facilidade de uso, além dos custos de instalação. Modelado como o Problema de Posicionamento de Pontos de Acesso, é um problema NP-Difícil, sem algoritmo que o resolva em tempo polinomial. Este trabalho propõe abordagens baseadas em hiper-heurísticas, onde cada solução é acompanhada por uma nova heurística reutilizável para novas instâncias. São utilizadas instâncias aleatórias de redes sem fio com dimensões relevantes. Comparações são feitas com abordagens de meta-heurística ILS e BRKGA. Os experimentos computacionais mostram excelentes resultados para dimensões maiores, demonstrando que esta abordagem é mais escalável e produz soluções razoáveis para o problema.

PALAVRAS CHAVE. Redes Sem Fio; Hiper-heurísticas; Problema Posicionamento Automático de Pontos de Acesso;

ABSTRACT

The design of wireless networks has become crucial with the popularization of mobile devices, standing out for scalability and ease of use, along with installation costs. Modeled as the Access Point Placement Problem, it is an NP-Hard problem, without an algorithm that solves it in polynomial time. This work proposes approaches based on hyper-heuristics, where each solution is accompanied by a new heuristic reusable for new instances. Random instances of wireless networks with relevant dimensions are used. Comparisons are made with Iterated Local Search (ILS) and Biased Random-Key Genetic Algorithm (BRKGA) meta-heuristic approaches. Computational experiments show excellent results for larger dimensions, demonstrating that this approach is more scalable and produces reasonable solutions to the problem.

KEYWORDS. Wireless Networks; Hyper-heuristics; Access Point Placement Problem.

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

1. Introdução

As redes sem fio têm se tornado cada vez mais populares nos últimos anos, muito em virtude de sua ubiquidade, flexibilidade e facilidade de uso. No entanto, a implementação de redes sem fio em ambientes internos apresenta desafios significativos, tais como a interferência de sinais, a limitação da cobertura em áreas densamente estruturadas e a necessidade de gerenciamento eficiente do espectro de frequência. Deste modo, o posicionamento de pontos de acesso (APs) se apresenta como uma tarefa importante para assegurar o funcionamento eficiente e confiável das redes sem fio. Nesse contexto, emergem problemas de otimização da classe NP-difícil, como o APPP (do inglês, *Access Point Placement Problem*, ou Problema de Posicionamento de Pontos de Acesso). Este trabalho é aborda o APPP para ambientes internos, tratando o posicionamento dos pontos de acesso para este tipo de rede, e considerando diversos tipos de AP, bem como a demanda por qualidade de serviço nos pontos de usuário, a demanda por banda passante e a proibição de cobertura em áreas externas ao ambiente.

Matematicamente, o APPP pode ser modelado como um problema de Programação Linear baseado no Problema de Localização de Facilidades. Megiddo e Tamir [1982] provam que o problema é NP-Difícil, sendo pelo menos tão difíceis quanto os problemas mais difíceis em NP, quando é proposto minimizar os custos dos APs e penalizando a cobertura de pontos proibidos da área externa. Dado que a solução exata de grandes instâncias de problemas NP-Difíceis é intratável, este trabalho propõe resolver o APPP utilizando hiper-heurísticas. O objetivo é desenvolver abordagens que possam ser usadas para gerar meta-heurísticas customizadas para o problema do posicionamento de APs, levando em conta fatores como cobertura, capacidade e interferência. Por sua vez, as hiper-heurísticas têm sido aplicadas com sucesso a diversos problemas, como otimização de rotas [Olgun et al., 2021], alocação de recursos [Zhuang e Zhou, 2020] e planejamento de manutenção [Dahal et al., 2007].

Para alcançar o objetivo proposto, serão utilizadas técnicas de geração de heurísticas baseadas em Programação Genética [Langdon e Poli, 2013], e será avaliada a eficácia de uma abordagem hiper-heurística de geração *offline* e outra *online*. A partir de uma série de heurísticas de construção e heurísticas de busca local de baixo nível, estas técnicas serão combinadas para gerar meta-heurísticas personalizadas para o problema do posicionamento de APs. Adicionalmente, o método proposto será comparado com uma abordagem baseada no *Biased Random-Key Genetic Algorithm* (BRKGA) [Gonçalves e Resende, 2010], e com o ILS de Rufino [2016]. Foram realizados testes comparativos entre estes métodos sobre o conjunto de instâncias geradas por Rufino [2016]. Os resultados são tabulados e analisados, na tentativa de esclarecer qual dos métodos obtém melhor desempenho para instâncias pequenas, médias e grandes.

O restante deste trabalho está organizado como segue: a Seção 2 aborda a literatura sobre a resolução do APPP utilizando métodos heurísticos; a Seção 3 define formalmente o APP, apresentando um modelo matemático para o problema e as heurísticas propostas para solução do mesmo; por sua vez, a Seção 4 descreve os experimentos realizados, assim como os resultados obtidos; por fim, a Seção 5 traz as considerações finais e propostas sobre trabalhos futuros.

2. Trabalhos Relacionados

Diversos trabalhos tratam sobre a resolução do APPP, mas a definição do problema apresenta alta fragmentação, por conta da sua adequação aos projetos reais, o que dificulta a estruturação das pesquisas na área. Este seção revisita a literatura sobre o APPP, abordando alguns dos trabalhos que propõem soluções para o APPP por meio de métodos heurísticos.

Kamenetsky e Unbehaun [2002] propuseram uma combinação de duas abordagens utilizando um algoritmo de remoções sucessivas, chamado *Pruning*, para construir uma solução inicial viável. Em seguida, melhorias são realizadas a partir desta solução utilizando a meta-heurística Simulated Annealing. A fim de avaliar o método proposto, foi desenvolvida uma função objetivo em

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

um espaço discreto, considerando a área de cobertura e qualidade do sinal obtido. Em seu trabalho, Lu et al. [2006] propuseram um método baseado na meta-heurística Taboo Search para a definição da quantidade de APs e o seu posicionamento em relação aos critérios de cobertura, interferências e qualidade de serviço. Uma cadeia de Markov é utilizada para avaliar a largura de banda em cada célula da área. Por fim, a cobertura é estimada por um simulador de propagação de rádio de muti-resolução.

Arroyo e Marques [2006] propuseram dois métodos baseados na meta-heurística GRASP (Greedy Randomized Adaptive Search Procedure) para resolver o problema de alocação de antenas de transmissão em uma região. Nesse trabalho, a função objetivo do modelo buscou maximizar a cobertura do sinal ao utilizar um número mínimo de antenas e instalando-as o mais próximo possível dos pontos de demanda, levando em consideração as restrições de alcance do sinal de transmissão e a presença de obstáculos.

Por sua vez, Vanhatupa et al. [2007] apresentam um novo Algoritmo Genético (AG) para o desenvolvimento de redes WLAN (*Wireless Local Area Network*) de acordo com os requisitos desejados, além de realizar uma estimativa da qualidade de serviço dividida entre: cobertura, custos de implantação e capacidade da rede. Assim, essas estimativas fornecem um retorno (*feedback*) para o próprio algoritmo e o construtor da rede. Outros requisitos são levados em conta neste método, como o consumo de energia, possíveis pontos de posicionamento de APs e quantidade de APs permitidos. Adicionalmente, Farkas e Huszák [2013] propuseram um novo método baseado em *Simulated Annealing*, com o objetivo de encontrar o posicionamento e o numero ideal de pontos de acesso Wi-Fi de tal forma que cada ponto na área seja coberto por pelo menos três APs.

Por fim, uma meta-heurística baseada no ILS (*Iterated Local Search*) foi proposta por Rufino [2016] para resolver instâncias sintéticas e reais de dimensões pequenas, médias e grandes. Para avaliar o método proposto, experimentos computacionais envolvendo adaptações de meta-heurísticas da literatura foram realizados e comparados com os demais resultados. Pelo nosso conhecimento, estes são os melhores resultados até esta presente data.

3. Modelagem do Problema

Esta seção apresenta a formulação matemática utilizada para definir a função objetivo que norteará os resultados de comparação com outros métodos. Além disso, as hiper-heurísticas desenvolvidas são explicadas em detalhes, bem como as instâncias de teste.

3.1. Formulação Matemática

A fim de realizar a discretização do problema, é considerada uma grade de pontos candidatos sobre a área. Dentre os pontos definidos, P é o conjunto de todos os pontos a serem cobertos, R é o conjunto de pontos onde é possível alocar um AP, T é o conjunto de diferentes tipos de APs e S é o conjunto dos pontos proibidos por serem cobertos, por exemplo por estarem em áreas externas. Definindo p_k como o custo de utilizar o roteador do tipo $k \in T$; q_s a penalidade pelo ponto proibido s estar na área de cobertura de pelo menos um roteador ativo; c_k a capacidade do roteador do tipo k e d_i a demanda por largura de banda do i-ésimo ponto. A variável de decisão r_{jk} assume o valor 1 se, e somente se, o roteador j do tipo k está sendo escolhido para ser utilizado na solução. A variável de decisão s_h assume o valor 1 se, e somente se, o ponto proibido h está inserido na área de cobertura de algum roteador utilizado. Seja $cob(r_{jk})$ uma função cujo retorno são todos os pontos inseridos na área de cobertura do roteador r_{jk}, e $cob(r_{jk})$ uma função de mesma motivação para os pontos proibidos, a modelagem do problema é apresentada nas expressões (1) a (6).

A função objetivo (1) minimiza o custo total dos roteadores em detrimento à penalização de dos pontos proibidos. As restrições (2) obrigam que todos os pontos clientes estejam ligados a um roteador. por sua vez, o conjunto de restrições (3) define que cada ponto deve ser conectado somente a um roteador ativo. As restrições (4) definem que a demanda por largura de banda dos

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

pontos conectados ao k-ésimo roteador não pode ultrapassar a sua capacidade, enquanto que as desigualdades (5) indicam se algum ponto proibido se encontra dentro da área de cobertura de algum roteador. Por fim, as restrições (6) definem o domínio das variáveis de decisão.

$$min \sum_{j \in R} \sum_{k \in T} p_k r_{jk} + \sum_{h \in S} q_h s_h \tag{1}$$

$$s.a. \sum_{j \in R} \sum_{k \in T} x_{ijk} = 1, \qquad \forall i \in P, \tag{2}$$

$$r_{jk} \geq x_{ijk}, \qquad \forall i \in cob(r_{jk}), \forall j \in R, \forall k \in T, \tag{3}$$

$$\sum_{i \in P} d_i x_{ijk} \leq c_k, \qquad \forall j \in R, \forall k \in T \tag{4}$$

$$s_h \geq r_{jk}, \qquad \forall h \in cobp(r_{jk}), \forall j \in R, \forall k \in T, \tag{5}$$

$$r_{ik}, x_{ijk}, s_h \in \{0, 1\}, \qquad \forall i \in P, \forall j \in R, \forall k \in T, \forall h \in S. \tag{6}$$

3.2. Hiper-heurística de Geração

Nesta seção será apresentada a estrutura da hiper-heurística de geração proposta. Inicialmente, apresentamos as heurísticas construtivas e heurísticas de busca local escolhidas para compor as meta-heurísticas a serem construídas. São descritas nove heurísticas, seis propostas por Rufino [2016] e três propostas neste trabalho, marcadas com '*'.

3.2.1. Heurísticas de Construção

GeraçãoGulosa(): Cada roteador é posicionado para adicionar o máximo de pontos clientes dentro de seu raio de cobertura, sendo os empates resolvidos de maneira equiprovável. Esta heurística desconsidera as restrições de capacidade, tendendo a gerar soluções inviáveis, que são fortemente penalizadas, resultando em um alto valor de função objetivo. Observa-se que os roteadores com maior raio são privilegiados.

GeraçãoAleatória(): Enquanto existirem pontos descobertos, são posicionados roteadores com tipos diferentes em locais aleatórios. Todas as escolhas são equiprováveis. Esta heurística complementa a *GeraçãoGulosa()* uma vez que constroi soluções com tipos diversos de roteadores. Mais uma vez, as soluções inviáveis são fortemente penalizadas.

3.2.2. Heurísticas de Busca Local

AllChangeLink(): Verifica-se, para todos os pontos clientes da solução se o roteador ao qual está conectado viola a restrição de capacidade de banda. Neste caso, será realizada uma tentativa de conectar este ponto a outro roteador que ainda não tenha atingido sua capacidade máxima.

AllDryRouter(): Para cada AP que esteja acima do limite de banda passante, distribuem-se todos os pontos conectados para outros roteadores ativos e que ainda não estão na sua capacidade máxima de banda.

BestAddRouter(): Verifica-se todos os pontos não cobertos na solução. Adiciona-se um novo AP que cobre a maior quantidade de pontos não conectados, respeitando as restrições de capacidade e de qualidade de serviço.

AllBestRadiusType(): Analisa todos os roteadores da solução. Se um roteador possuir algum ponto cliente conectado a uma demanda por qualidade de sinal acima ou abaixo da necessária, este roteador será substituído por outro com um raio de cobertura menor ou maior, se disponível, com o objetivo de suprir a demanda por qualidade de sinal do ponto cliente com eficiência.

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

BestLink(): Todos os pontos clientes não conectados na solução são verificados. Se um ponto cliente estiver dentro do raio de cobertura de um roteador com banda suficiente, os dois serão conectados.

****AllBestCapacityType()***: Analisa-se todos os roteadores da solução com o objetivo de adicionar o tipo de roteador com melhor capacidade para os pontos clientes. Se ainda houver a possibilidade de trocar um tipo de roteador com maior ou menor capacidade em relação ao roteador existente, este é adicionado.

****AllClosestMoveRouter()***: Procura-se mover os roteadores da solução para os pontos mais próximos possíveis que não sejam proibidos ou possuam outro roteador. Se existir mais de um ponto possível para a locomoção do roteador, mova-o para o ponto que cobrirá mais pontos.

3.3. Hiper-heurística *Online*

Dadas as heurísticas de construção e de busca local selecionadas, esta seção descreve a estrutura proposta para a hiper-heurística *online*, assim como os operadores de Crossover e Mutação. A hiper-heurística de construção proposta realiza buscas no espaço das heurísticas de maneira *online*, ou seja, a aprendizagem da hiper-heurística ocorre com o *feedback* em tempo de execução do programa. Além disso, a estrutura é baseada em um AG, como se descreve no Algoritmo 1.

Algorithm 1: Hiper-heurística de construção

```
1  Procedimento Hiper-heurística():
2      t ← 0;
3      P(t) ← população inicial com 14 soluções;
4      Avalie P(t) de acordo com o fitness;
5      while tempo limite não for excedido do
6          t ← t + 1;
7          Gere P(t) a partir de P(t − 1), utilizando crossover e mutação;
8          Avalie P(t) de acordo com o fitness;
9          Defina a população sobrevivente a partir das melhores soluções;
10     return P(t);
```

Cada indivíduo da população é uma meta-heurística, representada pelo vetor binário $S = (s_1, s_2, \cdots, s_9)$. Os 2 primeiros *bits* representam as heurísticas de construção utilizadas, enquanto os demais são as heurísticas de busca local. Deste modo, para cada *bit* s_i ativado, a i-ésima heurística é utilizada para construir a meta-heurística em questão. A Figura 1 ilustra a composição de bits de uma solução. Neste exemplo, a segunda heurística de construção foi escolhida, bem como a segunda, terceira, quinta e sétima heurísticas de busca local. A esta solução, também será associado o valor da função objetivo, como descrita na Equação (1).

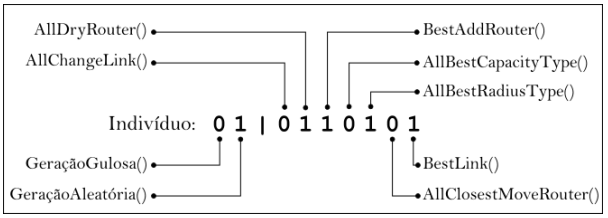


Figura 1: Sequência de *bits* de uma solução

Durante a fase de criação da população inicial são geradas exatamente 14 meta-heurísticas. São utilizadas as duas heurísticas de construção e as sete de busca local, sendo cada meta-heurística

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

composta por exatamente uma heurística de construção e uma de busca local, totalizando 14 combinações diferentes. Enquanto o tempo limite não é alcançado, o algoritmo gera novas populações de meta-heurísticas. São selecionados aleatoriamente sete pares, e é criado, para cada par, mais um par de soluções, utilizando o operador de *Crossover*. Estas soluções passam por um processo de mutação, e 50% das soluções são descartadas, sendo estas as que obtiveram os piores valores de função objetivo.

Sejam $A = (a_1, a_2, \cdots, a_9)$ e $B = (b_1, b_2, \cdots, b_9)$ o par de soluções sobre as quais será realizado o *Crossover*, e $C = (c_1, c_2, \cdots, c_9)$ o seu resultado, cada bit c_i é escolhido de maneira equiprovável entre a_i e b_i. São geradas duas soluções para cada operação de *Crossover*. Após a realização de um *Crossover*, cada meta-heurística gerada passa por um procedimento de mutação. Aqui, cada *bit* da solução passa tem uma chance de 2% de ser substituído pelo seu valor oposto, permitindo criar uma maior diversidade de soluções e não prender-se em ótimos locais.

3.4. Hiper-heurística *Offline*

Além de uma abordagem *online*, foi elaborada uma hiper-heurística de construção para realizar buscas no espaço das heurísticas de maneira *offline*, possibilitando a realização de treinamentos em um conjunto sintético de instâncias de treino com objetivo de usar o *feedback* destas execuções para extrair informações relevantes para a resolução do problema nas instâncias da literatura. A criação das instâncias de treino foi baseada na geração de plantas baixas em um espaço bidimensional. A divisão do espaço em cômodos, simulando paredes, foi efetuada usando a técnica de *Binary Space Partition* (BSP), definida primeiramente por Fuchs et al. [1980].

A técnica de BSP se baseia na divisão recursiva de um espaço em subconjuntos menores, através de hiperplanos. Este processo resulta em uma árvore BSP, comumente chamada de *BSP-Tree*, onde cada nó representa uma região do espaço e cada folha corresponde a uma subdivisão final do espaço inicial. Nesta representação de *BSP-Tree*, cada divisão do espaço resulta em uma nova partição da planta baixa, de forma que os polígonos internos são salas e possuem paredes que são obstáculos servindo como anteparos. Dada inicialmente uma região do espaço bidimensional, o método de geração de instâncias proposto seleciona aleatoriamente um corte horizontal ou vertical para particionar esta região em duas.

A Figura 2 exibe um exemplo de execução do BSP para geração de uma planta baixa usada para criar instâncias do APPP. A Figura 2a) exibe a planta baixa com a sequência de cortes que originou as salas ABCDE e F. Por sua vez, a Figura 2b) exibe a árvore binária utilizada pelo BSP para gerar esta instância. Dada a área inicial, foi escolhido aleatoriamente o corte vertical (que poderia ser horizontal, ou tocar outro ponto) que separa as regiões AB e CDEF. O método é chamado recursivamente para as duas novas regiões criadas. Este procedimento termina quando o número máximo de cortes ou o tamanho mínimo de área for atingido.

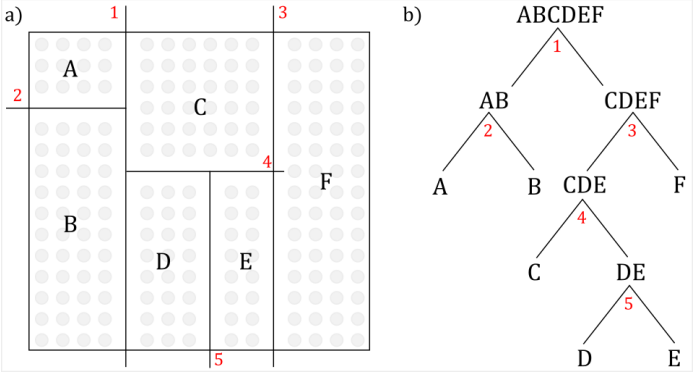


Figura 2: Exemplo de Execução do BSP

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

Foram gerados 9 grupos de instâncias com tamanhos variando entre 8×14 e 100×100, e para cada grupo foram geradas 10 instâncias diferentes. A hiper-heurística *offline* recebe estas instâncias como entrada para treinamento. Para cada instância de treino, são executadas todas as $9! =$ permutações de possíveis meta-heurísticas. Após esta etapa de treinamento, é analisada a frequência dos bits que foram mais utilizados entre todas as soluções para a construção de uma nova meta-heurística, considerada a melhor para execução.

3.5. *Biased Random-Key Genetic Algorithm*

Biased Random-Key Genetic Algorithm (BRKGA) é uma técnica genérica de Algoritmos Genéticos projetada para problemas de otimização combinatória proposta por Gonçalves e Resende [2010]. O BRKGA representa indivíduos da população do AG como vetores de números. Estes vetores devem ser decodificados em soluções viáveis para o problema de otimização que se deseja resolver. Uma das principais características do BRKGA é a introdução de um viés na geração de novos indivíduos. Em vez de selecionar pais de forma completamente aleatória, o BRKGA escolhe uma fração da nova população a partir dos melhores indivíduos da geração anterior, garantindo uma pressão seletiva que acelera a convergência para soluções ótimas.

A sequência de passos do BRKGA inclui: *(i) Inicialização:* geração de uma população inicial de vetores de números reais; *(ii) Decodificação:* conversão destes vetores em soluções concretas para o problema; *(iii) Avaliação:* cálculo da qualidade de cada solução; *(iv) Seleção e Reprodução:* formação de uma nova geração a partir de indivíduos selecionados com base em sua aptidão, com introdução de viés para favorecer os melhores; e *(v) Mutação:* pequenas alterações aleatórias nos vetores para manter a diversidade genética.

Para a implementação deste trabalho, foi adotada como base a interface de programação de aplicação (API) proposta por Toso e Resende [2014]. Esta API explora a característica genérica da abordagem BRKGA, delegando ao usuário apenas a responsabilidade pelos requisitos específicos do problema, permitindo uma fácil adaptação e aplicação do algoritmo a diferentes contextos e tipos de problemas de otimização combinatória.

Para a aplicação do BRKGA no APPP, dado o vetor $v = (v_1, v_2, \cdots, v_n - 1, v_n)$, cada par de valores consecutivos é decodificado como uma adição de um roteador em um ponto p do tipo t. Desta forma, esta etapa se assemelha à heurística de construção ***GeraçãoAleatória***, na medida que constrói uma solução baseada em adição de roteadores aleatoriamente. Na etapa de Avaliação, o custo total é calculado de acordo com a função objetivo (Equação 1) para a solução em questão, considerando os diversos fatores de penalização. Em seguida, as etapas de seleção, reprodução e mutação são realizadas conforme o modelo básico do BRKGA.

3.6. *Benchmark*

Como *benchmark*, foram utilizadas instâncias criadas por Rufino [2016], baseadas na discretização das plantas baixas em uma grade de pontos. Cada um destes pontos é posicionado horizontal e verticalmente (90º entre si) a exatamente um metro de distância. A Figura 3a) exemplifica a discretização ao ilustrar possíveis APs em uma planta baixa. O modelo de propagação utilizado foi baseado nos estudos de Nakprasit e Phongcharoenpanich [2018] e Androne e Palade [2010], que abordam o comportamento do sinal das redes Wi-Fi na frequência de 2,4 GHz. A Tabela 1 descreve os valores de atenuação utilizados nestas instâncias.

Cada instância deve ser resolvida utilizando tipos de APs com três características específicas: o raio de cobertura, o custo e a capacidade. Todas as instâncias devem utilizar roteadores dos seguintes tipos: A=(150R$, 100m, 150Mbps), representando um AP de entrada; B=(200R$, 100m, 300Mbps), representando um AP de médio porte; e B=(300R$,400m, 400Mbps), representando um AP de alta qualidade. Além disso, cada ponto é fornecido com as informações sobre sua posição na grade (x e y), demanda por qualidade (baixa, média ou alta) e a demanda por capacidade. Os

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

Tabela 1: Modelo de atenuação para anteparos [Rufino, 2016]

Tipo de Anteparo	Espessura (cm)	Atenuação (dB)
Divisória de Madeira	4	2
Divisória de Vidro	3	3
Placa de gesso (Drywall)	5 a 10	4
Parede de Tijolo	15	6
Parede de Concreto	15 a 20	9
Parede de Concreto	25 a 30	12

anteparos são descritos pelas coordenadas iniciais e finais na grade e pelo fator de redução do sinal. A Figura 3b) ilustra uma instância de exemplo. Neste caso, há paredes como anteparos. Os pontos vermelhos representam o requerimento de sinal com alta capacidade de rede (+++). Os pontos em laranja representam a necessidade obrigatória de sinais de qualidade alta (++). Os pontos em azul representam requerimentos de cobertura com qualidade mínima (+).

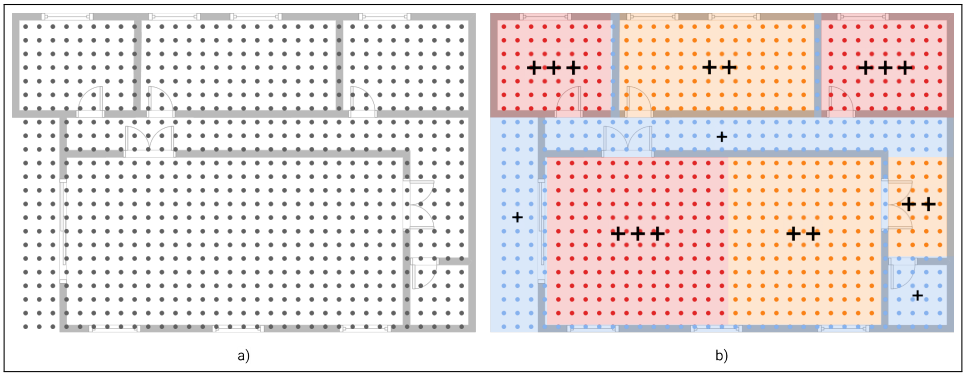


Figura 3: Exemplo de uma planta discretizada e a diferença entre a instância de teste e seus pontos

O *benchmark* possui 9 grupos de instâncias aleatórias, cada um contendo 10 casos de teste, em ordem crescente de dificuldade. Cada grupo possui o mesmo conjunto de pontos e de anteparos, sendo o tipo de anteparo selecionado ao acaso, bem como a demanda por qualidade e por largura de banda de cada ponto. Os grupos 8x14 e 100x100 são os menores e maiores, respectivamente.

4. Experimentos Computacionais

Esta seção apresenta os experimentos computacionais e os resultados obtidos. As hiper-heurísticas *offline* e *online* desenvolvidas, bem como a meta-heurística BRKGA aplicada ao APPP, são comparadas com os resultados da a meta-heurística *ILS-RVND* [Rufino, 2016].

Todos os métodos deste trabalho foram desenvolvidos em linguagem C++, utilizando o compilador g++. Os experimentos foram executados em um computador com processador Intel Core i7 (64 bits), com CPU contendo *cores* de 2,6 GHz, além de memória RAM DDR4 de 24 GB, 2667 MHz, e executando o sistema operacional Ubuntu 23.04. Os conjuntos de testes utilizados podem ser organizados em 9 grupos, divididos pelas suas dimensões: 8×14, 13×13, 11×23, 25×25, 30×30, 50×25, 50×50, 100×50 e 100×100. Cada conjunto de teste possui 10 instâncias distintas, indexados a partir de 0. Para fins de simplificação, denominamos as hiper-heurísticas desenvolvida neste trabalho de *HH-off* e *HH-on* (siglas para hiper-heurísticas *offline* e *online*, respectivamente), e a meta-heurística *ILS-RVND* de apenas ILS. Em conjunto com o BRKGA, todos os métodos foram submetidos a 10 execuções para cada instância, com um tempo limite de 10 segundos para cada execução.

Em todas as tabelas, a coluna **I** denomina cada instância de entrada. Para cada algoritmo: **ILS**, **BRKGA**, **HH-on** e **HH-off**, temos as colunas **avg** e **best**, que reportam a média do resultado da função objetivo para as 10 execuções de cada instância, e a melhor solução encontrada entre as

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

execuções, respectivamente. Adicionalmente, a coluna **best-sequence** denota a sequência (descrita na Figura 1) encontrada para a melhor função objetivo obtida para a **HH-on**. Por motivos de espaço neste trabalho, os resultados das instâncias de dimensões 8x14, 13x13, 11x23, 25x25 e 30x30 serão apresentados apenas textualmente.

Nas instâncias de dimensão 8x14, o *ILS*, *HH-on* e a *HH-off* ficaram aproximadamente empatadas em todos os testes, obtendo os melhores resultados. Já o BRKGA ficou próximo dos melhores resultados apenas 4 vezes nas 10 instâncias de teste. Para as instâncias de 13x13, o desempenho entre o *ILS* e o *HH-on* se mantém, com os melhores resultados empatados. Porém, o *HH-off* e, principalmente, o BRKGA ficaram mais distantes dos melhores resultados encontrados. No caso do BRKGA, a diferença entre resultados chegou até 355 vezes. Nas instâncias 11x23, a *HH-off* conquistou os melhores resultados em 8 casos de teste, enquanto o *ILS* e a *HH-on* tiveram melhores resultados em 2 desses casos. Além disso, o BRKGA teve o pior desempenho entre todos as abordagens selecionadas. Com os testes das instâncias de dimensão 25x25, a *HH-on* teve os melhores resultados em todos os testes e o *ILS* empatou em 8 deles. A *HH-off* ficou próxima dos melhores resultados encontrados e BRKGA seguiu com padrões de discrepância altos. Já nas instâncias 30x30, o *ILS* teve melhores resultados em 9 casos de teste, a *HH-on* venceu em 1 caso e empatou com o *ILS* em mais um. Além disso, a *HH-off* e o BRKGA seguiram o comportamento usual até o momento, com a *HH-off* se aproximando dos melhores valores e o BRKGA alcançando valores até 487 vezes maiores.

Já nas instâncias de 50x25, como pode ser visto na Tabela 2, o *HH-on* consegue superar o *ILS* por 4 vezes, mas possui valores de função objetivo piores por 5 momentos. E como foi visto anteriormente, a *HH-off* segue aproximando-se dos valores vencedores.

Tabela 2: Execução de testes com instâncias de dimensões 50x25

	ILS		BRKGA		HH-on			HH-off	
I	**avg**	**best**	**avg**	**best**	**avg**	**best**	**best-sequence**	**avg**	**best**
0	2300	2300	1790300	1790300	**2290**	**2250**	0 1 | 0 0 0 0 0 0 0 0	9435	2400
1	2700	2700	220300	220300	**2620**	**2600**	0 1 | 0 0 0 0 0 0 0 0	9700	2800
2	3690	3600	600300	600300	**3655**	3600	0 1 | 0 0 0 0 1 0 0	7775	3700
3	4070	4000	2190300	2190300	**4060**	4000	0 1 | 0 0 1 0 0 0 0	6190	4000
4	4980	4900	2160300	2160300	4980	**4850**	0 1 | 0 0 0 0 0 0 0	14040	5050
5	**5185**	**5100**	230300	230300	5215	5150	0 1 | 0 0 1 0 0 0 0	12320	5350
6	**5985**	**5800**	1500300	1500300	6025	5900	0 1 | 0 0 1 0 0 0 0	14050	6150
7	**6605**	**6550**	510300	510300	6650	6650	0 1 | 0 0 1 0 0 0 1	12730	6600
8	**7765**	**7600**	1430300	1430300	7890	7750	0 1 | 0 0 0 0 1 0 0	12910	7800
9	**8725**	**8600**	1840300	1840300	8865	8800	0 1 | 0 0 0 0 0 0 1	17795	9000

Ao analisar as Tabelas 3 e 4, pode-se verificar que o *HH-on* consegue superar o método *ILS* em todos os melhores resultados de função objetivo e com uma diferença bem mais expressiva, chegando a ultrapassar os grandes valores obtidos pelo *BRKGA* em alguns dos testes. Adicionalmente, o *HH-on* apenas possui média menor de função objetivo em 2 instâncias de 50x50. Além disso, o *HH-off* seguiu apresentando os mesmos comportamentos dos testes anteriores. Por fim, a Tabela 5 deixa ainda mais clara a vantagem do HH-on em relação aos demais métodos, uma vez que este método alcançou os melhores resultados, tanto na coluna best quanto na avg, para todas as 10 instâncias deste grupo.

Por conseguinte, podemos observar que o *HH-on* tende a encontrar resultados melhores para instâncias de dimensões maiores. No conjunto de instâncias construídas por Rufino [2016], o *HH-on* conseguiu encontrar resultados melhores no mesmo tempo limite mais vezes, principalmente impulsionado pelas instâncias de maior escala. O *ILS*, mesmo sendo uma meta-heurística interessante para o problema, parece não conseguir explorar outros ótimos locais rapidamente. Isto

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

Tabela 3: Execução de testes com instâncias de dimensões 50x50

	ILS		BRKGA		HH-on			HH-off	
I	**avg**	**best**	**avg**	**best**	**avg**	**best**	**best-sequence**	**avg**	**best**
0	4950	4850	4850331	4850331	**4550**	**4300**	0 1 | 1 1 0 0 0 0 1	15095	5200
1	6185	6150	4420324	4420324	**5810**	**5450**	0 1 | 0 1 0 0 0 0 0	13320	6350
2	7265	7150	1360349	1360349	**6890**	**6700**	0 1 | 0 0 1 0 1 0 1	18365	7200
3	8338	8240	10130200	10130200	**8173,7**	**7847**	0 1 | 0 0 1 0 0 0 0	20404,5	8350
4	9650	9600	2920308	2920308	**9190**	**9000**	0 1 | 0 0 0 0 0 0 0	32590	9850
5	**11146,4**	11050	10990200	10990200	11170	**10750**	0 1 | 0 0 1 0 0 0 1	33134,3	20850
6	12605	11450	10230321	10230321	**11215**	**10900**	0 1 | 0 0 0 0 1 0 0	34580	21850
7	13262,7	12979	8960300	8960300	**12799,9**	**12250**	0 1 | 1 1 1 0 0 0 0	42178,6	23100
8	**16905**	16550	8130330	8130330	17030	**16500**	0 1 | 0 0 1 0 0 0 0	45735	36650
9	17435	17150	2770307	2770307	**17380**	**17000**	1 0 | 1 1 1 0 1 0 0	31465	17600

Tabela 4: Execução de testes com instâncias de dimensões 100x50

	ILS		BRKGA		HH-on			HH-off	
I	**avg**	**best**	**avg**	**best**	**avg**	**best**	**best-sequence**	**avg**	**best**
0	26985,4	10266	15010300	15010300	**10331,7**	**9993**	0 1 | 0 0 1 0 1 0 0	39173,7	19929
1	12643,7	12415	4980364	4980364	**12548,9**	**12189**	0 1 | 0 0 1 0 0 0 0	18414,5	12310
2	57995,1	15500	16620367	16620367	**15263,3**	**14737**	0 1 | 0 0 1 0 0 0 0	53954,7	24873
3	51970	18880	20200371	20200371	**18968,3**	**18409**	0 1 | 0 1 0 0 0 0 0	47379,1	28143
4	44730	20100	11650354	11650354	**19775**	**19000**	0 1 | 0 0 0 0 0 0 0	33550	29400
5	115255,9	22450	16670300	16670300	**22056,5**	**21400**	0 1 | 0 0 0 0 0 0 0	57570,5	41684
6	131958,4	25400	14310300	14310300	**25074,8**	**23723**	0 1 | 0 1 0 0 0 0 0	55428,6	34974
7	164387	31422	18850361	18850361	**27734,5**	**26850**	0 1 | 1 1 0 0 0 0 0	43975	36800
8	239045,3	37400	21200300	21200300	**35120,2**	**34350**	0 1 | 0 1 1 0 0 0 0	85067,3	64200
9	454971,5	37731	21190300	21190300	**37322,8**	**36700**	0 1 | 1 1 1 0 0 0 0	71963,9	46438

se percebe ao verificar a magnitude alta das colunas **ILS-avg** em relação a **HH-on-avg**, chegando a serem quase 10 vezes de diferença em certos casos. O **HH-on-avg** possui valores próximos de **HH-on-best** também nas instâncias, indicando que o método de solução *HH-on* conseguiu apresentar robustez.

Outro ponto a ressaltar são as escolhas de heurísticas de baixo nível pela *HH-on*. Por diversas vezes na solução, as melhores execuções não precisaram ter mais que 4 heurísticas de busca local. Na maioria das vezes, precisaram apenas de 1 ou 2 destas heurísticas. Um ponto a se considerar é que apenas uma instância, entre as Tabelas 3 e 4, possui a heurística de construção gulosa. Todas as demais tiveram a heurística de construção aleatória como parte da solução final, indicando que as heurísticas performaram melhor ao precisarem reajustar as posições de roteadores ou pontos de acesso. Contudo, este comportamento se inverte nos resultados da Tabela 5.

Tabela 5: Execução de testes com instâncias de dimensões 100x100

	ILS		BRKGA		HH-on			HH-off	
I	**avg**	**best**	**avg**	**best**	**avg**	**best**	**best-sequence**	**avg**	**best**
0	40768502.8	29270699	24950499	24950499	**20149**	**19834**	1 1 | 0 0 0 0 0 0 1	58291,3	30808
1	41170694.3	21991592	17170645	17170645	16689132,2	**23734**	1 0 | 1 0 0 0 0 0 0	**51009**	24134
2	53204670.0	47520817	44730442	44730442	**29774**	**29484**	1 0 | 0 0 0 0 0 1 0	429555521,6	39934
3	49635720.7	36711217	37980430	37980430	**32944**	**32684**	1 0 | 0 0 0 0 0 0 0	429545356,6	53234
4	52938632.8	47640799	32930448	32930448	**38474**	**38134**	1 0 | 0 0 0 0 1 1 0	214828042,2	48674
5	56033458.3	47020661	29130300	29130300	**43984**	**43734**	1 1 | 0 0 0 0 0 0 1	115904,5	73882
6	58082761.8	52181074	28180586	28180586	4203024	**47034**	1 0 | 0 1 0 0 0 0 0	**91748,6**	67484
7	57622475.8	51380651	16000398	16000398	**52584**	**51984**	1 0 | 0 0 0 1 1 0 0	214833615,5	73116
8	72629523.6	69550778	42540452	42540452	**66714**	**65984**	1 1 | 0 1 0 0 0 0 0	119663,2	66926
9	71399614.1	66950919	19130563	19130563	6286054	**69484**	1 0 | 0 1 0 0 0 0 0	214865285,2	109934

É importante ressaltar, contudo, que as grandes diferenças entre os valores encontrados pe-

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

las heurísticas se deve ao forte fator de penalização adotado para soluções inviáveis. Isto demonstra o grau de dificuldade para lidar com o APPP, uma vez que os métodos apresentam dificuldades até para conseguir soluções viáveis em instâncias de grandes dimensões. Neste sentido, acreditamos que seja necessário criar novas heurísticas de construção que sejam capazes de gerar, de início, soluções viáveis.

5. Considerações Finais

O objetivo deste trabalho foi propor soluções ao APPP utilizando hiper-heurísticas *online* e *offline* de construção de heurísticas, além de uma aplicação da meta-heurística BRKGA, a fim de comparar com a literatura e expandir as abordagens de resolução deste problema. Para construir a hiper-heurística, cinco heurísticas de busca local desenvolvidas por Rufino [2016] foram utilizadas como heurísticas de baixo nível para fazerem parte da estrutura. Duas outras heurísticas de refinamento são propostas neste trabalho, totalizando sete. A heurística de construção gulosa proposta por Rufino [2016] também foi utilizada, e uma heurística construtiva aleatória foi acrescentada.

Os experimentos computacionais demonstraram excelentes resultados para instâncias de dimensões 50×50, 100×50 e 100×100, as maiores da base de dados consultada. Dentre as 30 instâncias destes dois grupos, a hiper-heurística *online* obteve resultados melhores em todas, ao comparar com a meta-heurística *ILS-RVND* [Rufino, 2016]. Entretanto, para instâncias de dimensões menores, as hiper-heurísticas não obtiveram resultados melhores que o *ILS-RVND* com facilidade, levando à possibilidade de melhorias. Como o tempo limite da execução das instâncias foi configurado como 10 segundos, isto pode ter afetado a comparação entre a meta-heurística e a hiper-heurística de construção *online*, visto que esta última necessita executar mais passos para conseguir resultados razoáveis.

Pontos a se destacar são as possibilidades de melhorias ao método BRKGA, como a utilização de mais heurísticas nas construções de respostas, já que este teve a maior utilização na escolha de roteadores do que na escolha de heurísticas. A hiper-heurística *offline* ainda pode conseguir resultados melhores com formas de treinamento mais robustas, bem como a introdução de variações de heurísticas aleatórias de primeira melhora, proporcionando maiores possibilidades de entendimento do problema. De forma geral, o método proposto cumpriu seu objetivo, de maneira a ter conseguido resultados animadores. Dito isto, diversas melhorias ainda podem ser realizadas para a estrutura exposta em trabalhos futuros, expandindo nosso conhecimento sobre o APPP ao modificar as abordagens do problema. Ainda como propostas de trabalhos futuros, destacamos que a proposição de novas heurísticas de baixo nível, quer seja construtivas ou de busca local, apresentam grande potencial de melhoria para os métodos *online* e *offline*. Adicionalmente, também é possível inserir as heurísticas de perturbação propostas por Rufino [2016] em ambos *frameworks*.

Os autores agradecem à FAPESQ-PB pela contribuição para este artigo através do financiamento EDITAL Nº 010/2021-FAPESQ/PB-MCTIC/CNPq.

Referências

Androne, C. e Palade, T. (2010). Radio coverage and performance analysis for local area networks. In *2010 9th International Symposium on Electronics and Telecommunications*. IEEE. URL http://dx.doi.org/10.1109/isetc.2010.5679347.

Arroyo, J. E. C. e Marques, T. B. (2006). Heurísticas grasp aplicado ao problema de alocação de antenas de transmissão. *XXXVIII Simpósio Brasileiro de Pesquisa Operacional*.

Dahal, K. P., Aldridge, C. J., e Galloway, S. J. (2007). Evolutionary hybrid approaches for generation scheduling in power systems. *European Journal of Operational Research*, 177(3): 2050–2068. ISSN 0377-2217. URL http://dx.doi.org/10.1016/j.ejor.2005.12.018.

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br

Farkas, K. e Huszák, A. (2013). Optimization of wi-fi access point placement for indoor localization. *Journal IIT (Informatics & IT Today)*.

Fuchs, H., Kedem, Z. M., e Naylor, B. F. (1980). On visible surface generation by a priori tree structures. In *Proceedings of the 7th annual conference on Computer graphics and interactive techniques - SIGGRAPH '80*, SIGGRAPH '80. ACM Press. URL http://dx.doi.org/10.1145/800250.807481.

Gonçalves, J. F. e Resende, M. G. C. (2010). Biased random-key genetic algorithms for combinatorial optimization. *Journal of Heuristics*, 17(5):487–525. ISSN 1572-9397. URL http://dx.doi.org/10.1007/s10732-010-9143-1.

Kamenetsky, M. e Unbehaun, M. (2002). Coverage planning for outdoor wireless lan systems. In *2002 International Zurich Seminar on Broadband Communications Access - Transmission - Networking (Cat. No.02TH8599)*, IZSBC-02. IEEE. URL http://dx.doi.org/10.1109/izsbc.2002.991793.

Langdon, W. B. e Poli, R. (2013). *Foundations of genetic programming*. Springer Science & Business Media.

Lu, J.-L., Jaffres-Runser, K., Gorce, J.-M., e Valois, F. (2006). Indoor wlan planning with a qos constraint based on a markovian performance evaluation model. In *2006 IEEE International Conference on Wireless and Mobile Computing, Networking and Communications*. IEEE. URL http://dx.doi.org/10.1109/wimob.2006.1696372.

Megiddo, N. e Tamir, A. (1982). On the complexity of locating linear facilities in the plane. *Operations Research Letters*, 1(5):194–197. ISSN 0167-6377. URL http://dx.doi.org/10.1016/0167-6377(82)90039-6.

Nakprasit, K. e Phongcharoenpanich, C. (2018). Investigation of received signal strength of ieee 802.11n wlan in coverage of nakhon ratchasima rajabhat university. In *2018 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. IEEE. URL http://dx.doi.org/10.1109/ecticon.2018.8620018.

Olgun, B., Koç, u., e Altıparmak, F. (2021). A hyper heuristic for the green vehicle routing problem with simultaneous pickup and delivery. *Computers & Industrial Engineering*, 153:107010. ISSN 0360-8352. URL http://dx.doi.org/10.1016/j.cie.2020.107010.

Rufino, T. G., A.; Silva (2016). Meta-heurística ils para o problema de posicionamento automático de pontos de acesso em ambientes internos. *XLVIII Simpósio Brasileiro de Pesquisa Operacional*.

Toso, R. e Resende, M. (2014). A c++application programming interface for biased random-key genetic algorithms. *Optimization Methods and Software*, 30(1):81–93. ISSN 1029-4937.

Vanhatupa, T., Hannikainen, M., e Hamalainen, T. D. (2007). Genetic algorithm to optimize node placement and configuration for wlan planning. In *2007 4th International Symposium on Wireless Communication Systems*. IEEE. URL http://dx.doi.org/10.1109/iswcs.2007.4392413.

Zhuang, Y. e Zhou, H. (2020). A hyper-heuristic resource allocation algorithm for fog computing. In *Proceedings of the 2020 the 4th International Conference on Innovation in Artificial Intelligence*, ICIAI 2020. ACM. URL http://dx.doi.org/10.1145/3390557.3394321.

DOI: 10.59254/sbpo-2024-193362
https://proceedings.science/p/193362?lang=pt-br