Análise de Grafos para Alocação de Frequências em Redes de Telecomunicação

*Uma Comparação de Algoritmos de Coloração e seu Impacto na Minimização de Interferências

Autor(a):
1st Maíra Beatriz de Almeida Lacerda
Bacharelado em Engenharia de Computação
CEFET-MG, Campus V
Divinópolis, Brasil
maairaallacerda@gmail.com

Contribuinte Textual:
2nd Celso Vinícius Sudário Fernandes
Bacharelado em Engenharia de Computação
CEFET-MG, Campus V
Divinópolis, Brasil
celso.23@aluno.cefetmg.br

***Resumo*—A alocação eficiente de frequências em redes de telecomunicação é fundamental para garantir a qualidade do serviço e evitar interferências entre torres próximas. Neste trabalho, investigamos o desempenho de diferentes algoritmos de coloração de grafos (DSATUR, Gulosa, Backtracking e Simulated Annealing) na tarefa de distribuir cores (frequências) de forma a minimizar conflitos. Foram analisados cenários com restrições que simulam a presença de torres com frequências previamente definidas, bem como grafos com diferentes densidades para avaliar o impacto na eficiência dos métodos. Os resultados mostram que a escolha do algoritmo depende tanto do tamanho e densidade do grafo quanto da existência de restrições, evidenciando vantagens e limitações de cada abordagem.**

***Palavras Chave*—Grafos, Coloração de grafos, Alocação de frequências, DSATUR, Gulosa, Backtracking, Simulated Annealing**

I. Introducão

O crescimento exponencial do uso de dispositivos móveis e a expansão da infraestrutura de redes de telecomunicação têm aumentado a complexidade do processo de alocação de frequências. Em termos práticos, cada torre de celular ou estação de rádio deve operar em canais distintos das estações vizinhas para minimizar interferência e melhorar a qualidade do serviço. Esse problema, normalmente chamado de *Frequency Assignment Problem* (FAP), pode ser modelado como uma coloração de grafos, em que cada nó representa uma torre e as arestas indicam um potencial de interferência (ou distância insuficiente) entre torres adjacentes.

A estrutura deste artigo está organizada da seguinte forma: a Seção I apresenta a introdução ao problema, destacando sua relevância e contexto. A Seção II fornece uma contextualização teórica, abordando os principais conceitos e fundamentos necessários para a compreensão do tema. A Seção III discute os trabalhos correlatos, analisando pesquisas anteriores e suas contribuições para a área. A Seção IV descreve a metodologia adotada, detalhando os procedimentos utilizados na modelagem do problema e na execução dos experimentos. Na Seção V, são apresentados e analisados os resultados obtidos, avaliando o desempenho das abordagens implementadas. Por fim, a Seção VI traz as conclusões do trabalho, as limitações encontradas e sugestões para estudos futuros.

II. Contextualização

A alocação de frequências em redes de telecomunicações é um problema complexo que envolve múltiplos fatores, como a localização geográfica das torres, a potência de transmissão e a coexistência de diferentes tecnologias (2G, 3G, 4G, 5G). Esses fatores impactam diretamente a distribuição do espectro, tornando essencial o uso de abordagens eficientes para minimizar interferências. Nesse contexto, a teoria dos grafos é amplamente utilizada para modelar o problema, representando cada torre como um vértice e conectando aquelas que podem interferir entre si por meio de arestas. Essa modelagem permite a aplicação de algoritmos de coloração, que buscam atribuir frequências distintas a torres adjacentes, reduzindo conflitos no espectro de rádio.

A coloração de grafos é amplamente utilizada na alocação de frequências em redes de telecomunicações para minimizar interferências, atribuindo cores distintas a torres interferentes. Estudos indicam que redes móveis urbanas seguem padrões de grafos esparsos, onde a distribuição das torres e barreiras físicas influenciam na conectividade. Algoritmos heurísticos de coloração demonstram ser eficazes na redução de interferências, melhorando a qualidade do serviço sem necessidade de expansão da infraestrutura, como discutido na Seção III.

A. Impactos Reais e Considerações Práticas

Em redes de telecomunicação reais, a alocação de frequências costuma levar em conta fatores como:

(a) Potência de Transmissão: Torres com potência maior têm maior zona de interferência, podendo levar a graus efetivos mais altos no grafo.
(b) Coexistência de Tecnologias: Diferentes padrões (por exemplo, 4G, 5G, Wi-Fi) coexistem em frequência, exigindo ainda mais cuidado na separação de canais.

(c) Situação Dinâmica: A topologia pode mudar com a instalação de novas torres ou a remoção de outras, exigindo que o algoritmo seja capaz de atualizar a coloração sem refazer tudo do zero.

Apesar de o modelo de coloração fornecer um arcabouço teórico elegante, a implementação prática requer heurísticas adaptativas, principalmente em grandes redes com topologias em constante evolução.

B. Análise do Problema

Uma das principais dificuldades do problema é a necessidade de modelar as restrições de interferência entre torres. Essas restrições podem ser representadas como um grafo, onde cada vértice corresponde a uma torre e as arestas representam possíveis interferências. Além disso, a densidade do grafo varia de acordo com a proximidade das torres, o que afeta diretamente a escolha do algoritmo mais adequado.

A constante expansão das redes exige realocações regulares de frequências já estabelecidas, tornando essencial o uso de algoritmos flexíveis e eficientes. A escolha do algoritmo de coloração influencia diretamente a qualidade da solução, variando conforme a densidade do grafo. Em grafos densos, o DSATUR tende a ser mais eficaz, enquanto em grafos esparsos, métodos heurísticos podem oferecer soluções mais rápidas com qualidade aceitável. Além disso, restrições como torres com frequências pré-definidas aumentam a complexidade do problema, exigindo adaptações nos algoritmos para lidar com essas limitações.

Assim, esta análise visa destacar a importância de avaliar cuidadosamente as características do grafo e os requisitos do problema antes de selecionar o método de solução.

C. Algoritmos Fundamentais

1) DSATUR

O *Degree of Saturation* (DSATUR) é um algoritmo heurístico para coloração de grafos que, a cada iteração, escolhe o vértice com maior grau de saturação para colorir. O grau de saturação de um vértice v é definido como o número de cores distintas atualmente atribuídas aos seus vizinhos. Quando há empate, normalmente utiliza-se como critério de desempate o maior grau (número de vizinhos) ou algum outro critério fixo.

2) Algoritmo Guloso

O algoritmo Guloso de coloração é uma abordagem simples e eficiente para atribuir cores aos vértices de um grafo, evitando conflitos entre vértices adjacentes. Seu desempenho depende da ordem de processamento dos vértices, sendo comum ordená-los por grau decrescente para melhorar a eficiência. Embora não garanta a solução ótima, essa estratégia pode reduzir o número de cores utilizadas na coloração. Devido à sua rapidez, o algoritmo é amplamente empregado em aplicações práticas que exigem soluções ágeis.

3) Backtracking

O método de *Backtracking* busca exaustivamente a coloração mínima de um grafo, aplicando estratégias de poda para eliminar soluções inválidas e reduzir o tempo de execução. Técnicas como BTSL e BTDSATUR melhoram sua eficiência ajustando a ordem de coloração e os limites superiores com base em heurísticas. Essas otimizações tornam o algoritmo mais viável para grafos de maior dimensão, equilibrando precisão e desempenho computacional.

4) Simulated Annealing

O *Simulated Annealing* (SA) é uma meta-heurística inspirada no processo de recozimento em metalurgia. A ideia é partir de uma solução inicial (neste caso, uma coloração de grafos) e, gradualmente, fazer pequenos ajustes, aceitando até movimentos que piorem a solução de forma probabilística, para evitar ficar preso em mínimos locais. Ao longo do tempo, a probabilidade de aceitar piores soluções diminui (como uma "temperatura" que resfria), fazendo com que o processo convirja para uma solução em geral de boa qualidade.

III. Trabalhos Correlatos

Nota-se que a coloração de grafos tem sido amplamente utilizada na alocação de frequências em redes de telecomunicações, pois permite minimizar interferências entre torres de transmissão e otimizar o uso do espectro disponível. Diversos estudos têm explorado diferentes algoritmos e técnicas para resolver esse problema.

O artigo *RSOTP - Um protocolo TDMA baseado em coloração de grafos para redes de sensores sem fio* [6] propõe um protocolo TDMA baseado em coloração de grafos para redes de sensores sem fio. O trabalho aborda a necessidade de minimizar interferências entre dispositivos próximos e propõe uma abordagem que melhora a alocação de canais em redes sem fio. A pesquisa demonstra que técnicas baseadas em coloração de grafos podem reduzir significativamente a interferência e melhorar a eficiência espectral em cenários práticos.

Outro estudo relevante é o *Spectrum graph coloring to improve Wi-Fi channel assignment in a real-world scenario via edge contraction* [7], que propõe uma abordagem para a alocação eficiente de canais *Wi-Fi* em um cenário real, utilizando a coloração de grafos combinada com a técnica de *edge contraction* (contração de arestas). O artigo explora o impacto da sobreposição de canais em redes sem fio e como a modelagem por coloração pode melhorar a alocação dinâmica de frequências. Os experimentos realizados na Universidade de Alcalá indicam que algoritmos heurísticos, como *Simulated Annealing*, podem melhorar significativamente o desempenho da rede, superando métodos tradicionais de alocação de canais.

Ambos os trabalhos demonstram a importância da teoria dos grafos na otimização da alocação de frequências, evidenciando que abordagens heurísticas e meta-heurísticas são eficazes para lidar com a complexidade do problema em redes reais. Essas pesquisas reforçam que a escolha do método de coloração depende de fatores como a densidade da rede, a necessidade de atualização dinâmica e as restrições impostas pelo ambiente de telecomunicações.

IV. METODOLOGIA

Neste trabalho, foram considerados quatro abordagens para a coloração de grafos, cada um com características distintas de complexidade, qualidade de solução e aplicabilidade ao problema de alocação de frequências em torres de transmissão.

A. Ferramentas Utilizadas

A implementação do modelo de alocação de frequências foi realizada em Python, utilizando as bibliotecas `pandas` para processamento dos dados, geopy para cálculo de distâncias geográficas e `NetworkX` para modelagem do grafo e aplicação dos algoritmos de coloração, incluindo Guloso, DSATUR, Backtracking e Simulated Annealing. A biblioteca `matplotlib` foi empregada para visualização dos grafos e dos resultados obtidos. O código-fonte foi versionado e armazenado no GitHub, a fim de garantir a reprodutibilidade dos experimentos V-A.

Os experimentos foram executados em um computador, equipado com processador AMD Ryzen 7 5700U com Radeon Graphics (1.80 GHz), 12 GB de RAM e sistema operacional *Windows 11*. Para garantir um ambiente eficiente para a execução dos algoritmos e a análise dos dados, foi utilizado o *Windows Subsystem for Linux (WSL)* em conjunto com o *Visual Studio Code*, permitindo a execução de scripts *Python* de forma otimizada.

B. Preparação de Dados

Para a construção do grafo representando a alocação de frequências em redes de telecomunicações, utilizou-se um conjunto de dados contendo informações sobre torres de transmissão na região de Divinópolis, MG. A base de dados inclui atributos como o identificador da estação, o nome da entidade responsável e as coordenadas geográficas de cada torre.

Inicialmente, os dados foram carregados a partir de um arquivo CSV e processados para conversão das coordenadas de latitude e longitude do formato DMS (graus, minutos e segundos) para o formato decimal, garantindo compatibilidade com as funções de cálculo de distância. Esse processamento foi realizado por meio da função `load_data()`, que aplica uma transformação às colunas de coordenadas.

Com os dados devidamente tratados, foi gerado um grafo em que cada nó representa uma torre e as conexões entre os nós são estabelecidas com base na distância geográfica. A criação do grafo foi realizada por meio da função `create_graph()`, que estabelece arestas entre torres cuja distância seja inferior a um limite pré-definido de 5 km, refletindo um cenário de interferência prática em redes móveis.

A estrutura resultante foi utilizada como base para a aplicação dos algoritmos de coloração de grafos, permitindo a análise da alocação eficiente de frequências e a minimização de interferências. O conjunto de dados [2] e o código-fonte utilizados estão disponíveis para reprodução e validação dos experimentos conduzidos.

C. Modelagem dos Grafos

1) DSATUR

Neste escopo o algoritmo inicia com todos os vértices sem cor atribuída. A cada iteração, seleciona-se o vértice com o maior grau de saturação, ou seja, aquele que possui mais vizinhos já coloridos com cores distintas. Esse vértice recebe a menor cor disponível que não conflite com as cores dos seus vizinhos. Após a atribuição de uma cor, os graus de saturação dos vértices afetados são atualizados. O processo é repetido até que todos os vértices tenham sido coloridos, garantindo uma coloração eficiente do grafo.

Algoritmo 1: DSATUR Algorithm

Input: Grafo $G = (V, E)$
Output: Coloração dos vértices
Inicialização:
Definir todas as cores como não atribuídas.
Inicializar grau de saturação $sat(v) = 0$ para todo $v \in V$.
Escolher um vértice inicial (por exemplo, o de maior grau) e atribuir a cor 1 a ele.
Atualizar $sat(v)$ para cada v vizinho do vértice colorido.
Processo Iterativo:
while existirem vértices não coloridos **do**
    Selecionar o vértice u não colorido com maior $sat(u)$.
    if houver empate **then**
        Aplicar critério secundário (maior grau, ordem, etc.).
    end if
    Determinar a menor cor viável para u (que não conflite com vizinhos).
    Atribuir essa cor a u.
    Atualizar $sat(v)$ para cada v vizinho de u.
end while
return Coloração gerada

2) Algoritmo Guloso

O algoritmo inicia ordenando os vértices de acordo com um critério pré-definido. Para cada vértice, analisa-se as cores já utilizadas pelos seus vizinhos, garantindo que a coloração siga uma abordagem consistente. Em seguida, atribui-se ao vértice a menor cor disponível que não gere conflito com as cores dos seus vizinhos. Esse processo é repetido até que todos os vértices tenham sido coloridos.

Algoritmo 2: Greedy Coloring Algorithm

Input: Grafo $G = (V, E)$, ordem dos vértices $\{v_1, v_2, \ldots, v_n\}$
Output: Coloração dos vértices
Inicialização:
Definir todas as cores como não atribuídas.
Processo Iterativo:
for cada vértice v_i na ordem dada **do**
    Obter o conjunto de cores dos vizinhos de v_i já coloridos.
    Selecionar a menor cor c que não esteja no conjunto anterior.
    Atribuir c a v_i.
end for
return Coloração gerada

3) Backtracking

O algoritmo busca atribuir cores aos vértices de maneira sistemática, garantindo que nenhuma cor entre vizinhos entre em conflito. Caso ocorra um conflito durante o processo, a estratégia de backtracking é aplicada, permitindo que o algoritmo retroceda e tente uma cor diferente para o vértice em questão. Esse procedimento continua até que todos os vértices estejam coloridos ou que todas as opções possíveis tenham sido esgotadas.

Algoritmo 3: Backtracking Coloring Algorithm

Input: Grafo $G = (V, E)$, número máximo de cores k
Output: Coloração dos vértices ou falha se não for possível colorir com k cores
Inicialização:
Definir um vetor de cores $color[1..|V|]$ inicialmente vazio.
Processo Recursivo:
função Colore(v):
if $v > |V|$ **then**
  return true // Todos os vértices coloridos
end if
for $c = 1$ até k **do**
  if é seguro colorir v com cor c (sem conflitos) **then**
    $color[v] \leftarrow c$
    if Colore($v + 1$) = true **then**
      return true
    end if
    $color[v] \leftarrow 0$ // Reset da cor
  end if
end for
return false // Não foi possível colorir com até k cores

4) Simulated Annealing

Para aplicar o Simulated Annealing na coloração de grafos, o processo inicia-se com uma coloração inicial, que pode ser gerada de forma gulosa ou aleatória. Define-se uma temperatura inicial alta, permitindo uma maior exploração do espaço de soluções.

A cada iteração, realiza-se uma modificação na coloração, alterando a cor de um vértice ou trocando cores entre vértices. A nova solução é então avaliada: se for melhor, ou seja, apresentar menos conflitos ou utilizar menos cores, ela é imediatamente aceita. Caso contrário, pode ser aceita com uma probabilidade calculada como $p = e^{-\Delta/T}$, onde Δ representa a variação no custo da solução e T corresponde à temperatura atual. Por fim, a temperatura é reduzida gradativamente ao longo das iterações, seguindo um critério de resfriamento predefinido. O algoritmo é interrompido quando um critério de parada é atingido, como uma temperatura mínima ou um número máximo de iterações.

Algoritmo 4: Simulated Annealing for Graph Coloring

Input: Grafo $G = (V, E)$, solução inicial S, temperatura inicial T_0, taxa de resfriamento α, número de iterações $maxIter$
Output: Melhor solução encontrada S^*
Inicialização:
Definir $S^* \leftarrow S$ // Melhor solução encontrada
Processo Iterativo:
for $iter = 1$ até $maxIter$ **do**
  Gerar nova solução S' a partir de S (alterando a cor de um vértice ou trocando cores de dois vértices)
  Calcular o custo (número de conflitos, cores usadas, etc.) de S e S'
  $\Delta \leftarrow$ custo(S') $-$ custo(S)
  if $\Delta < 0$ **then**
    $S \leftarrow S'$ // Aceita melhoria
    if custo(S') $<$ custo(S^*) **then**
      $S^* \leftarrow S'$ // Atualiza melhor solução global
    end if
  else
    Aceitar S' com probabilidade $p = e^{-\Delta/T}$
    if aceito **then**
      $S \leftarrow S'$
    end if
  end if
  Atualizar temperatura $T \leftarrow \alpha \cdot T$
end for
return S^* // Melhor solução encontrada

D. Configuração do Cenário

Para avaliar o desempenho dos algoritmos, conduziram-se experimentos em grafos esparsos e densos. Também foram inclusos cenários com e sem restrições de cores pré-definidas. Nesse contexto, os grafos foram gerados com número de nós variando de 50 a 1000, ajustando-se a probabilidade de cada aresta existir para controlar a densidade (utilizou-se um modelo *Erdős-Rényi* simples como referência).

1) Cenários com Restrições

Além disso, foram realizadas simulações considerando cenários em que um subconjunto de nós (variando entre 5% e 15% do total) tem suas cores pré-fixadas. Essa situação corresponde, por exemplo, a torres que já possuem frequências estabelecidas por motivos contratuais, licenças específicas ou limitações de hardware, não podendo ser alteradas livremente durante a otimização.

Em contrapartida, quando parte dos nós já está colorida, o problema torna-se essencialmente o de "estender" uma coloração parcial (ou seja, respeitar as cores pré-definidas e colorir os demais nós sem gerar conflitos).

E. Métricas de Avaliação

Foram avaliados o número de cores utilizadas, o tempo de execução do algoritmo e a qualidade da solução, considerando tanto o número de conflitos restantes (caso permitidos na meta-heurística) quanto a proximidade do resultado obtido em relação ao ótimo em instâncias menores.

V. Resultados e Análise

Os experimentos foram conduzidos seguindo a metodologia descrita na Seção IV, utilizando grafos representando redes de telecomunicações com diferentes densidades e restrições de frequência. A partir desse cenário, foram analisados o desempenho dos algoritmos de coloração em termos de número de cores utilizadas, tempo de execução e impacto das restrições na alocação do espectro. Os resultados apresentados a seguir refletem a eficácia de cada abordagem na minimização de interferências e na adaptação às condições estabelecidas.

A. Reprodução

Com o objetivo de garantir a transparência e a reprodutibilidade dos resultados apresentados neste estudo, todo o código utilizado para a modelagem e execução dos experimentos está disponível em um repositório público. O diretório contém scripts implementados em Python, permitindo que outros pesquisadores possam validar, reproduzir ou expandir a pesquisa. O código-fonte, bem como a documentação detalhada da pesquisa, pode ser acessado para futuras replicações e aprimoramentos, promovendo a continuidade dos estudos na área de otimização de espectro em redes de telecomunicações. A URL do repositório Git é fornecida na Seção de Referências [1] desse artigo

B. Resultados Experimentais

1) Grafos Esparsos

A Figura 1 ilustra resultados obtidos em um cenário de 200 nós, onde cada aresta tem probabilidade de existência de 0,05 (ou seja, 5% de chance), caracterizando um grafo bem esparso.

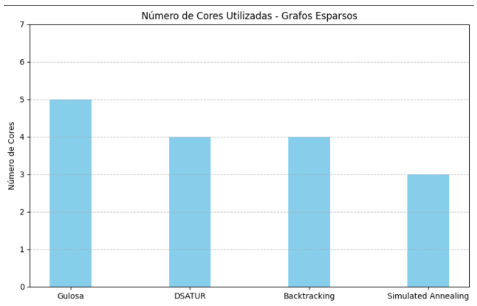

Figura 1. Grafico dos valores do grafo esparso.

Observa-se que o Simulated Annealing encontrou soluções com apenas 3 cores, sugerindo que as modificações dinâmicas e aceitação probabilística de soluções intermediárias ajudaram a evitar alocações excessivas. DSATUR e Backtracking ficaram com 4 cores, o que ainda é um bom desempenho, enquanto o Gulosa usou 5 cores, sendo menos eficiente.

Em termos de tempo de execução, o Guloso foi muito rápido, enquanto o Backtracking foi consideravelmente mais lento (apesar de ainda viável nesta instância de 200 nós). O DSATUR também apresentou velocidade satisfatória, pois sua heurística de saturação não demanda busca exaustiva.

2) Grafos Densos

Para grafos densos, fixou-se novamente 200 nós, mas a probabilidade de existência de cada aresta foi de 0,5, resultando em um número muito maior de arestas. A Figura 2 mostra resultados ilustrativos:

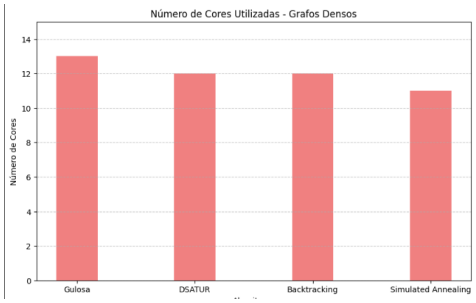

Figura 2. Grafico de valores do grafo denso

No cenário denso, há muito mais conflitos potenciais. O Gulosa, seguindo sua estratégia simples, chegou a 13 cores. DSATUR e Backtracking ficaram em 12 cores, enquanto o Simulated Annealing conseguiu chegar a 11, uma diferença pequena, mas significativa em termos de uso real de frequências.

3) Análise Comparativa e Visual

A Tabela I apresenta uma comparação entre os métodos testados, considerando o número de cores utilizadas e o tempo de execução. Esses dados permitem avaliar o desempenho de cada abordagem em termos de otimização da alocação de frequências em redes de telecomunicações. Os resultados obtidos serão analisados a seguir, destacando as vantagens e limitações de cada algoritmo no contexto estudado.

Tabela I
TEMPO DE EXECUÇÃO

Algoritmo	Número de Cores	Tempo de Execução (ms)
Guloso	80	1,40
DSATUR	78	5,70
Backtracking	80	12,80

Em primeira análise, o algoritmo **DSATUR** demonstrou eficiência ao lidar com nós de cor fixa, pois o cálculo do grau de saturação já incorpora essas restrições naturalmente. Isso permite que, ao selecionar o próximo vértice a ser colorido, o impacto dos vizinhos com cores pré-definidas seja considerado, reduzindo a probabilidade de conflitos e mantendo o número de cores adicionais baixo. Na prática, como ilustrado na Figura 3, observou-se que a presença de nós com cores fixas teve pouco ou nenhum impacto no total final de cores utilizadas, quando comparado ao cenário sem restrições.

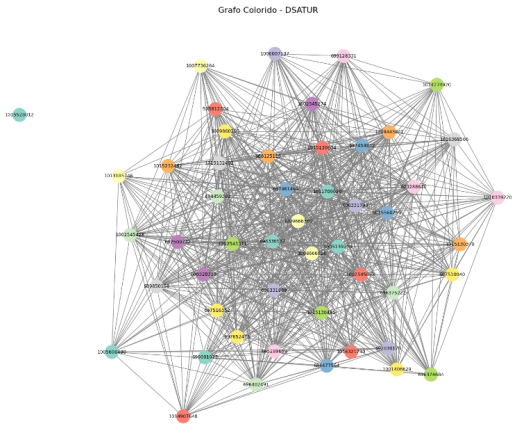

Figura 3. Exemplo de grafo denso colorido pelo Algoritmo DSATUR.

Por outro lado, o algoritmo Guloso segue uma ordem fixa para colorir os nós, sem reavaliar decisões anteriores, a menos que heurísticas de reordenamento sejam aplicadas. Quando há nós com cores pré-definidas, o método pode enfrentar conflitos que exigem o uso de cores adicionais nos nós subsequentes, já que a ordem de coloração pode não ser a mais eficiente para acomodar essas restrições. Na prática, como ilustrado na Figura 4, observa-se que, em regiões densas do grafo, onde há uma grande quantidade de conexões, o número de cores necessárias pode aumentar devido a decisões locais subótimas. No entanto, apesar dessa limitação, o algoritmo apresenta um tempo de execução significativamente menor em comparação com abordagens mais sofisticadas, sendo aproximadamente 75,44% mais rápido que o DSATUR, conforme os dados apresentados na Tabela I.

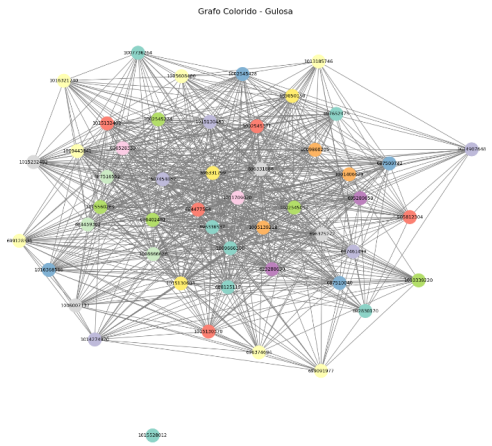

Figura 4. Exemplo de grafo denso colorido pelo Algoritmo Gulosa.

Além disso, o algoritmo *Backtracking*, apesar de encontrar a coloração ótima, tem seu tempo de execução impactado pela presença de nós com cores pré-fixadas, pois precisa respeitar essas restrições e propagar seus efeitos ao longo da busca. Em alguns casos, a limitação de cores pode reduzir o espaço de busca, restringindo opções e agilizando certas decisões. No entanto, se muitos nós restritos estiverem em regiões críticas do grafo, o número de retrocessos aumenta significativamente. Na prática, como ilustrado na Figura 5, observa-se que o alto número de conexões entre os vértices torna o processo de coloração mais custoso, exigindo mais iterações para garantir que vértices adjacentes recebam cores distintas. Devido ao seu alto custo computacional, conforme indicado na Tabela I, o Backtracking teve um tempo de execução 128,07% maior que o DSATUR. Desse modo, esse método pode ser inviável para instâncias de grande porte, sendo mais adequado como referência para avaliar soluções em cenários menores.

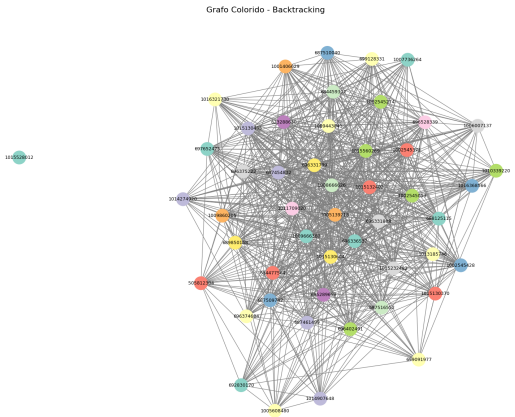

Figura 5. Exemplo de grafo denso colorido pelo Algoritmo Backtracking.

O algoritmo **Simulated Annealing (SA)**, por ser uma meta-heurística flexível, incorpora naturalmente restrições de nós com cores pré-fixadas, tratando-os como elementos imutáveis no espaço de solução. Durante as iterações, ao modificar a coloração de outros nós ou introduzir perturbações, esses vértices restritos são preservados, permitindo que o algoritmo ajuste a coloração dos demais para minimizar conflitos ou reduzir o número de cores. Além disso, seu mecanismo probabilístico de aceitação de soluções piores no início da execução ajuda a evitar impasses, permitindo que o SA se adapte bem às restrições impostas e explore soluções de melhor qualidade ao longo do tempo. Na Figura 6, observa-se que o SA conseguiu uma distribuição mais equilibrada das cores em comparação com abordagens mais rígidas, minimizando áreas com alta concentração da mesma cor. Isso indica que o algoritmo foi capaz de evitar agrupamentos excessivos de vértices da mesma classe cromática, favorecendo uma solução mais distribuída e eficiente.

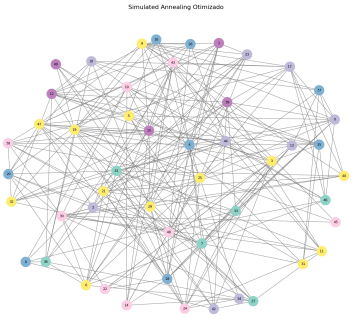

Figura 6. Exemplo de grafo denso colorido pelo Algoritmo Simulated Annealing.

4) Discussão e Análise

Primeiramente, nota-se que o percentual de nós restritos influencia diretamente a complexidade do problema. Quando apenas 5% dos nós estão pré-coloridos, o impacto na coloração final é pequeno, mas com 15% ou mais, a dificuldade para ajustar as cores aumenta. Ainda assim, DSATUR e Simulated Annealing mantiveram um desempenho relativamente bom, enquanto o Guloso, dependendo da ordem dos nós, pode exigir mais cores, e o Backtracking se torna inviável devido ao tempo de execução elevado.

Ao analisar o tempo de processamento, observa-se que o Guloso foi a opção mais rápida, sendo 75,44% mais eficiente que o DSATUR e 89,23% mais rápido que o Backtracking, embora com uma coloração menos otimizada. Esse desempenho ocorre porque o algoritmo Guloso atribui cores de forma sequencial, sem reconsiderar decisões anteriores, tornando sua execução extremamente rápida, mas potencialmente ineficiente em relação ao número de cores utilizadas.

Por outro lado, o Backtracking, apesar de garantir uma solução favorável, apresentou um tempo 128,07% superior ao DSATUR, devido ao seu caráter exaustivo, explorando todas as possibilidades de coloração para encontrar a melhor solução. Esse método sofre grande impacto em grafos densos, pois o número de combinações possíveis cresce exponencialmente com a quantidade de vértices e arestas, resultando em um tempo de execução significativamente maior.

O DSATUR apresentou um bom equilíbrio entre eficiência e qualidade, pois ao priorizar vértices com maior grau de saturação, consegue distribuir as cores de forma mais estratégica, reduzindo conflitos sem comprometer tanto o tempo de execução. Já o Simulated Annealing, por ser uma heurística probabilística, se mostrou uma alternativa promissora ao permitir ajustes graduais na coloração, escapando de soluções locais ruins sem precisar explorar todas as combinações possíveis, como no Backtracking. Assim, a escolha do algoritmo mais adequado depende do contexto: enquanto métodos heurísticos oferecem um bom compromisso entre tempo e qualidade para grafos maiores, abordagens exatas como o Backtracking são mais indicadas para instâncias menores, onde encontrar a solução ótima é uma prioridade.

VI. Conclusão

A coloração de grafos aplicada à alocação de frequências mostrou-se uma abordagem eficiente para minimizar interferências em redes de telecomunicações. Os resultados obtidos evidenciaram que a escolha do algoritmo impacta diretamente a qualidade da solução e o tempo de execução. Enquanto métodos heurísticos como DSATUR e Simulated Annealing foram capazes de encontrar boas soluções em tempo razoável, abordagens exatas, como o Backtracking, apresentaram um custo computacional muito elevado para grafos maiores. Esses achados reforçam a necessidade de equilíbrio entre precisão e eficiência ao lidar com cenários reais, onde a escalabilidade e a adaptabilidade são fatores críticos para o sucesso da alocação de espectro.

Embora os resultados sejam promissores, algumas limitações devem ser consideradas. Em grafos muito grandes, o Backtracking apresenta um tempo de execução elevado, e até mesmo heurísticas como DSATUR podem exigir ajustes ou paralelização para manter um bom desempenho. Além disso, as simulações foram feitas em um cenário estático, sem considerar a adição ou remoção de torres ao longo do tempo. Como trabalhos futuros, sugere-se o desenvolvimento de meta-heurísticas híbridas, combinando a rapidez de métodos como DSATUR com refinamentos avançados, como Tabu Search. Outra possibilidade é incorporar fatores realistas, como potência de sinal e atenuação, aproximando o modelo das condições reais. Por fim, a criação de estratégias de coloração dinâmica pode tornar a alocação de frequências mais eficiente em redes em constante evolução.

Referências

[1] LACERDA, Maíra Beatriz de Almeida. Repositório do trabalho final sobre grafos. 2025. Disponível em: https://github.com/mairaallacerda/AEDSII-Trabalho-Final-Grafos.git.

[2] LACERDA, Maíra Beatriz de Almeida. Dataset sobre alocação de frequências em redes de telecomunicações. 2025. Disponível em: https://zenodo.org/records/14835048.

[3] Bondy, J. A., & Murty, U. S. R. (2008). *Graph theory*. Springer.

[4] Brélaz, D. (1979). New methods to color the vertices of a graph. *Communications of the ACM*, 22(4), 251-256.

[5] Kirkpatrick, S., Gelatt, C. D., & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science*, 220(4598), 671-680.

[6] Santos, B. R. S. (2020). RSOTP - Um protocolo TDMA baseado em coloração de grafos para redes de sensores sem fio.

[7] Orden, D., Marsa-Maestre, I., Gimenez-Guzman, J. M., De La Hoz, E., & Álvarez-Suárez, A. (2019). Spectrum graph coloring to improve Wi-Fi channel assignment in a real-world scenario via edge contraction. *Discrete Applied Mathematics*, 263, 234–243.