MINISTÉRIO DA EDUCAÇÃO
Universidade Federal de Ouro Preto
Instituto de Ciências Exatas e Aplicadas
Colegiado de Engenharia de Produção

# Roteamento e Alocação de Espectro em Redes Ópticas Flexíveis: integração do algoritmo genético com o método *Branch and Cut* por meio do Gurobi

**Pedro Afonso Mendes e Marinho**

João Monlevade, MG
2025

Pedro Afonso Mendes e Marinho

# Roteamento e Alocação de Espectro em Redes Ópticas Flexíveis: integração do algoritmo genético com o método *Branch and Cut* por meio do Gurobi

Trabalho de conclusão de curso apresentado ao curso de Engenharia de Produção do Instituto de Ciências Exatas e Aplicadas da Universidade Federal de Ouro Preto, como parte dos requisitos necessários para a obtenção do título de Bacharel em Engenharia de Produção.

Orientador: Prof. Dr. Thiago Augusto Oliveira Silva

João Monlevade, MG

2025

SISBIN - SISTEMA DE BIBLIOTECAS E INFORMAÇÃO

M338r Marinho, Pedro Afonso Mendes e.
Roteamento e alocação de espectro em redes ópticas flexíveis [manuscrito]: integração do algoritmo genético com o método Branch and Cut por meio do Gurobi. / Pedro Afonso Mendes e Marinho. - 2025.
40 f.: il.: color., tab.. + Equações Matemáticas.

Orientador: Prof. Dr. Thiago Augusto Oliveira Silva.
Monografia (Bacharelado). Universidade Federal de Ouro Preto. Instituto de Ciências Exatas e Aplicadas. Graduação em Engenharia de Produção .

1. Algoritmos genéticos. 2. Comunicações ópticas. 3. Fibras ópticas. 4. Otimização combinatória. 5. Roteadores (Redes de computadores). 6. Sistemas de transmissão de dados - Óptica de fibras. I. Silva, Thiago Augusto Oliveira. II. Universidade Federal de Ouro Preto. III. Título.

CDU 519.8

Bibliotecário(a) Responsável: Flavia Reis - CRB6-2431

MINISTÉRIO DA EDUCAÇÃO
UNIVERSIDADE FEDERAL DE OURO PRETO
REITORIA
INSTITUTO DE CIENCIAS EXATAS E APLICADAS
DEPARTAMENTO DE ENGENHARIA DE PRODUCAO - ICEA

# FOLHA DE APROVAÇÃO

**Pedro Afonso Mendes e Marinho**

**Roteamento e Alocação de Espectro em Redes Ópticas Flexíveis: integração do algoritmo genético com o método Branch and Cut por meio do Gurobi**

Monografia apresentada ao Curso de Graduação em Engenharia de Produção da Universidade Federal de Ouro Preto como requisito parcial para obtenção do título de Engenheiro de Produção

Aprovada em 27 de março de 2025

**Membros da banca**

Prof. Dr. Thiago Augusto de Oliveira Silva - Orientador(a) - Universidade Federal de Ouro Preto

Prof. Dr. Alexandre Xavier Martins - Universidade Federal de Ouro Preto

Prof. Dr. Samuel Martins Drei - Universidade Federal de Ouro Preto

Thiago Augusto de Oliveira Silva, orientador do trabalho, aprovou a versão final e autorizou seu depósito na Biblioteca Digital de Trabalhos de Conclusão de Curso da UFOP em 01/04/2025

Documento assinado eletronicamente por **Thiago Augusto de Oliveira Silva**, **PROFESSOR DE MAGISTERIO SUPERIOR**, em 01/04/2025, às 14:33, conforme horário oficial de Brasília, com fundamento no art. 6º, § 1º, do Decreto nº 8.539, de 8 de outubro de 2015.

A autenticidade deste documento pode ser conferida no site http://sei.ufop.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0 , informando o código verificador **0887144** e o código CRC **3375C872**.



R. Diogo de Vasconcelos, 122, - Bairro Pilar Ouro Preto/MG, CEP 35402-163
Telefone: (31)3808-0817 - www.ufop.br

*Dedico este trabalho minha mãe, meu pai e a minha irmã, que sempre me apoiaram em todas as decisões ao longo da minha vida. Obrigado por estarem sempre presentes, nos bons e maus momentos. Dedico também aos meus amigos e familiares, que sempre me motivaram e me fizeram seguir em frente.*

# Agradecimentos

Quero agradecer minha família por sempre terem me apoiado durante todas as decisões da minha vida, sempre sendo um suporte para mim. Em especial, tenho a agradecer meus pais, por terem tornado possível a minha graduação.

Também quero agradecer meu orientador, Dr. Thiago Augusto de Oliveira Silva, por ter aceitado este projeto e me orientado durante a trajetória.

Agradeço profundamente à Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES) pelo valioso apoio financeiro concedido, que possibilitou minha participação em um intercâmbio acadêmico. Essa experiência enriquecedora contribuiu significativamente para meu desenvolvimento pessoal e profissional. Sou igualmente grato à Universidade Federal de Ouro Preto (UFOP) pelas oportunidades únicas oferecidas durante minha graduação.

*“Tudo passa”*

# Resumo

Devido à crescente demanda de dados em todo o mundo, faz-se necessário uma rede de transmissão de dados eficiente. Considerando as redes de fibras ópticas, a correta definição dos caminhos e espectros utilizados se apresenta como um desafio; portanto, cria-se o problema de roteamento e alocação de espectro em redes ópticas flexíveis. Portanto, este trabalho busca trabalhar com o algoritmo genético e o método simplex e *Branch and Cut*, por meio do Gurobi, na solução de um problema do gênero, integrando os métodos e comparando os seus resultados em diferentes instâncias.

**Palavras-chaves**: Roteamento e Alocação de Espectro, Redes Ópticas, Algoritmo Genético, *Branch and Cut*.

# Abstract

Due to the growing demand for data worldwide, there is a need for an efficient data transmission network. Considering fiber optic networks, the correct definition of the paths and spectrums used presents itself as a challenge, thus creating the routing and spectrum allocation problem in flexible optical networks. Therefore, this work seeks to work with the genetic algorithm and the simplex and Branch and Cut methods, through Gurobi, in solving a problem of this kind; integrating the methods and comparing their results in different instances.

**Keywords**: Routing and Spectrum Assigment, Optical Networks, Genetic Algorithm, Branch and Cut.

# Lista de ilustrações


Figura 1 – Representação da Continuidade . . . . . 3
Figura 2 – Representação da Contiguidade . . . . . 4
Figura 3 – Representação do Espectro não sobreposto . . . . . 4
Figura 4 – Fluxograma das etapas do algoritmo genético . . . . . 6
Figura 5 – Rede óptica 6x9 . . . . . 15
Figura 6 – Representação da codificação das soluções . . . . . 16
Figura 7 – Representação de uma População . . . . . 18
Figura 8 – Representação do cruzamento em um único ponto . . . . . 20
Figura 9 – Representação do cruzamento em N pontos . . . . . 20
Figura 10 – Representação do cruzamento uniforme . . . . . 20
Figura 11 – Representação da mutação do slot . . . . . 21
Figura 12 – Representação da mutação do caminho . . . . . 21
Figura 13 – Representação da mutação do slot e do caminho . . . . . 21

# Lista de tabelas

Tabela 1 – Topologia do arquivo das redes . . . . . 14
Tabela 2 – Topologia do arquivo de demandas . . . . . 15
Tabela 3 – Resultados encontrados dos métodos . . . . . 23

# Lista de abreviaturas e siglas

**CAPES** Coordenação de Aperfeiçoamento de Pessoal de Nível Superior

**GUROBI** Gurobi

**ISIMA** Instituto Superior de Informática, Modelagem e suas Aplicações

**RSA** Routing and Spectrum Assignment

**UFOP** Universidade Federal de Ouro Preto

# Sumário


**1 INTRODUÇÃO . . . . . 1**
**1.1 Objetivo geral . . . . . 1**
1.1.1 Objetivos específicos . . . . . 1
**1.2 Contribuições . . . . . 2**
**1.3 Organização do Trabalho . . . . . 2**

**2 REVISÃO DA LITERATURA . . . . . 3**
**2.1 Roteamento e Alocação do Espectro . . . . . 3**
**2.2 Algoritmo Genético . . . . . 5**
2.2.1 **Codificação das soluções . . . . . 5**
2.2.2 **População inicial . . . . . 6**
2.2.3 **Avaliação . . . . . 7**
2.2.4 **Seleção . . . . . 7**
2.2.5 **Cruzamento . . . . . 7**
2.2.6 **Mutação . . . . . 7**
**2.3 *Branch and Cut* . . . . . 8**
**2.4 Linguagem Python . . . . . 8**
**2.5 Biblioteca NetworkX . . . . . 9**
**2.6 Pacote Gurobipy . . . . . 9**
**2.7 Modelo matemático . . . . . 10**
2.7.1 **Variáveis . . . . . 10**
2.7.2 **Funções objetivo . . . . . 10**
2.7.3 Restrições . . . . . 11

**3 METODOLOGIA . . . . . 13**
**3.1 Classificação da pesquisa . . . . . 13**
**3.2 Entrada de dados do problema . . . . . 13**
3.2.1 **Arquivo do gráfico da rede . . . . . 14**
3.2.2 **Arquivo da demanda . . . . . 14**
**3.3 Instância utilizada . . . . . 15**

**4 RESULTADOS . . . . . 16**
**4.1 Adaptação do Algoritmo Genético ao Problema . . . . . 16**
4.1.1 **Codificação das soluções . . . . . 16**
4.1.2 **População Inicial . . . . . 17**
4.1.3 **Avaliação . . . . . 18**

4.1.4 **Seleção** . . . . . 19
4.1.5 **Cruzamento** . . . . . 19
4.1.6 **Mutação** . . . . . 21
**4.2 Resolução matemática pelo GUROBI** . . . . . **22**
**4.3 Performance GUROBI e Algoritmo Genético** . . . . . **22**

**5 CONSIDERAÇÕES FINAIS** . . . . . **25**

**REFERÊNCIAS** . . . . . **26**

# 1 Introdução

O presente trabalho de conclusão de curso aborda o problema de Roteamento e Alocação do Espectro, ou, em inglês, Routing and Spectrum Assignment (RSA), em redes ópticas. A crescente demanda por transmissão de dados exige soluções eficientes para otimizar a alocação de espectro e o roteamento de conexões, tornando o RSA um problema de grande relevância para operadores de rede e pesquisadores da área, como destacado por Colares *et al.* (2022).

Diversos estudos têm buscado estratégias eficazes para resolver o problema RSA, combinando técnicas heurísticas e métodos exatos. Entre as abordagens existentes, algoritmos genéticos têm-se mostrado promissores na obtenção de soluções iniciais de qualidade, como demonstrado por Daouadi (2022), que podem ser refinadas posteriormente por métodos de otimização matemática.

Portanto, este trabalho propõe o desenvolvimento de um algoritmo híbrido, que utiliza um algoritmo genético para gerar soluções iniciais próximas da ótima e, em seguida, emprega o solver Gurobi (GUROBI) para refinar essas soluções e, sempre que possível, encontrar a solução ótima do problema. A escolha dessa abordagem baseia-se em estudos prévios e na experiência adquirida durante um intercâmbio acadêmico no Instituto Superior de Informática, Modelagem e suas Aplicações (ISIMA), em Clermont-Ferrand, na França.

É importante ressaltar que esse intercâmbio foi viabilizado pelo auxílio da CAPES. Durante este período, foi desenvolvida uma versão inicial do algoritmo genético voltado para o problema RSA que, neste caso, irá fornecer uma solução inicial para o problema em questão.

Ao longo deste estudo, serão apresentadas as principais metodologias aplicadas ao problema, bem como os resultados obtidos com a implementação da abordagem híbrida proposta. A expectativa é que a combinação do algoritmo genético com o solver GUROBI contribua para uma solução eficiente e viável para o RSA, auxiliando na gestão de redes ópticas de forma mais otimizada e avançando o campo de pesquisa para o problema.

## 1.1 Objetivo geral

O objetivo geral deste trabalho é encontrar a solução ótima para o problema RSA em diferentes instâncias, utilizando um algoritmo genético para gerar uma solução inicial e, posteriormente, o solver GUROBI para refiná-la quando necessário.

### 1.1.1 Objetivos específicos

Dessa forma, os objetivos específicos são:

- Revisar estudos relacionados ao assunto;
- Instalar pacotes e bibliotecas para o desenvolvimento dos códigos ;
- Aprimorar o método gerador de uma solução inicial por meio do Algoritmo Genético;
- Desenvolver um código em Python em que o método *Branch and Cut* se aplique;
- Integrar o GUROBI neste código para posterior comparação de resultados;
- Configurar o resultado do algoritmo genético como *upper bound* do método *Branch and Cut* no novo código;
- Analisar a performance dos métodos a fim de comparação.

## 1.2 Contribuições

Este trabalho trará contribuições no campo teórico ao investigar a integração de diferentes métodos para a resolução do problema RSA. A pesquisa proporcionará uma análise sobre como a combinação de algoritmos heurísticos e métodos exatos pode influenciar a obtenção de soluções mais eficientes e próximas da ótima. Além disso, serão discutidos os impactos dessa integração na redução de custos computacionais e na escalabilidade das soluções para instâncias maiores do problema. Espera-se que os resultados obtidos forneçam uma base teórica sólida para futuras pesquisas e aprimoramentos no campo da otimização aplicada a redes ópticas.

## 1.3 Organização do Trabalho

A organização deste trabalho está feita de tal forma que facilite a compreensão do problema em si, assim como as medidas que foram propostas com o decorrer do conteúdo reportado. Dentro do Capítulo 1 pode-se ter a noção do que de fato será desenvolvido, do que se trata o problema, alguns conceitos iniciais e o que se espera ao fim do trabalho.

No Capítulo 2 os conceitos e definições serão melhor explicados de acordo com uma revisão bibliográfica de todo o conteúdo. Dentro do Capítulo 3 estão descritas as características desta pesquisa, assim como uma explicação dos dados utilizados.

Com o Capítulo 4 é possível ver em maior detalhe o que de fato foi desenvolvido e quais os resultados foram encontrados. Além de poder perceber como a integração dos métodos se deu.

Já no Capítulo 5 tem-se a análise geral de todo o trabalho, considerando os resultados finais e, desta forma, como eles podem ajudar e impactar em futuros trabalhos semelhantes.

# 2 Revisão da Literatura

## 2.1 Roteamento e Alocação do Espectro

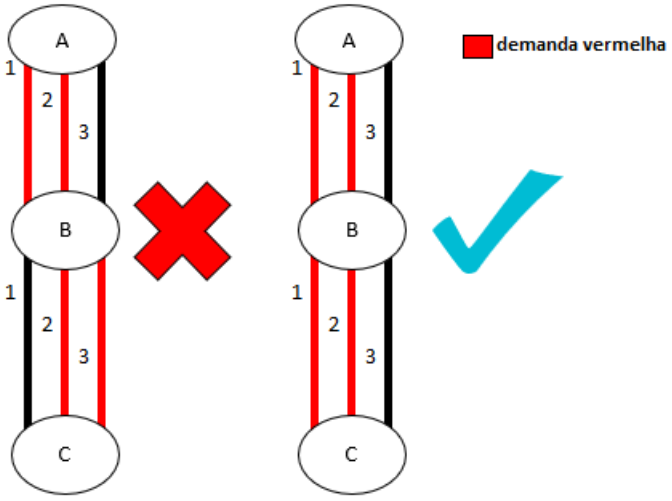
Choudhury *et al.* (2020) definem o problema RSA quando é necessário encontrar uma rota para cada solicitação de tráfego e alocar um número necessário de slots espectrais, para que o uso do espectro seja o melhor possível em uma rede. Assim, é possível dizer que o RSA contém duas partes a resolver: o roteamento entre os pontos de origem e destino e a alocação do espectro das fibras da rede, conforme Araujo, Subramanian e Eduardo (2017).

Considerando esta característica, o RSA é considerado um problema NP-difícil, ou seja, a complexidade de resolução aumenta exponencialmente com o aumento do problema Cunha, Bonasser e Abrahão (2002). Como o objetivo é a melhor alocação possível do espectro entre todos os caminhos disponíveis para uma determinada demanda, não existe um único método conhecido que seja capaz de resolvê-lo sempre de forma eficiente.

Além da complexidade do problema, podem existir vários objetivos a serem encontrados. Pode-se encontrar o caminho mais curto entre o ponto de origem e o destino, bem como priorizar a melhor alocação de espectro da rede. Independentemente da escolha da função objetivo, Kerivin *et al.* (2021) dizem que certas regras devem ser sempre respeitadas. Essas regras são:

- **Continuidade do espectro:** se os slots de solicitação escolhidos forem aceitáveis, deverão utilizar os mesmos intervalos ao longo de todo o caminho, entre todos os links da rede óptica do ponto de origem ao ponto de destino. Como mostrado na imagem abaixo:

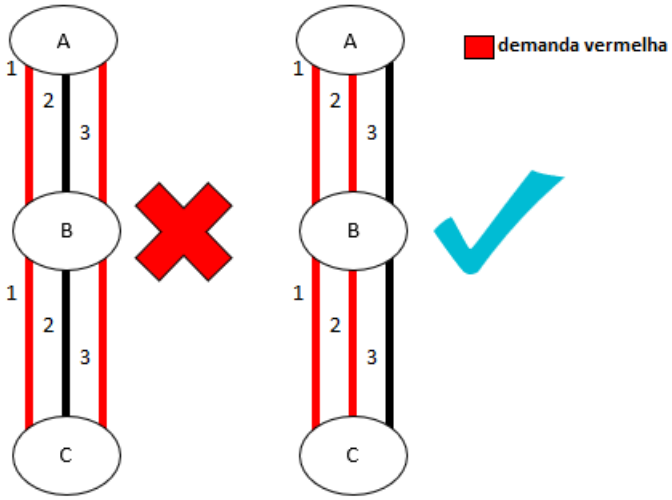
Figura 1 – Representação da Continuidade




Fonte: Elaborado pelo autor

- **Contiguidade do espectro:** o número de slots de cada solicitação deve estar próximo um do outro, ou seja, os slots de cada uma das solicitações devem ser adjacentes na fibra. Como mostrado na imagem abaixo:

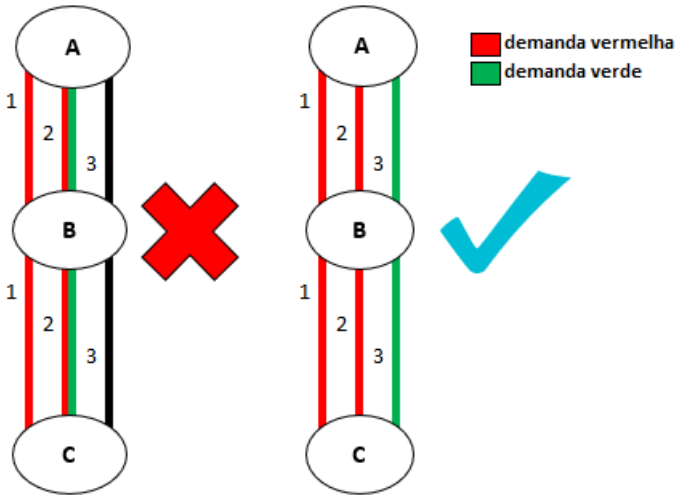
Figura 2 – Representação da Contiguidade




Fonte: Elaborado pelo autor

- **Espectro não sobreposto:** em cada link da rede óptica deve ser designado um slot específico para apenas uma solicitação, no máximo. Como mostrado na imagem abaixo:

Figura 3 – Representação do Espectro não sobreposto




Fonte: Elaborado pelo autor

## 2.2 Algoritmo Genético

Segundo Golfeto, Moretti e Neto (2007), a ideia de um algoritmo genético surgiu na segunda metade do século e ganhou notoriedade a partir de alguns autores. Trata-se de uma meta-heurística que é totalmente baseada na teoria evolucionária de Darwin, de acordo com Camargo e Toledo (2008). Esta teoria propõe que a recombinação de indivíduos resultantes de mutações e cruzamentos genéticos possibilita a obtenção de uma população mais adaptada ao ambiente, conforme a teoria da seleção natural. No caso das redes ópticas, isto aplica-se de tal forma que cada solução representa um indivíduo e é capaz de gerar uma outra a partir de uma evolução, entende-se operações que geram outras soluções.

Então, cada solução na população representa uma possível solução para o problema estudado. Uma vez que existam diversas soluções possíveis, as operações de mutação e cruzamento serão feitas até que algum critério de parada seja alcançado.

Como resultado, reflete o processo de seleção natural onde os indivíduos mais adaptados sobreviverão e, consequentemente, também se reproduzirão. Portanto, os indivíduos mais adaptados serão os responsáveis por fazer a espécie evoluir, em analogia, a evolução das soluções fornecerá a solução ótima ou a solução mais próxima possível disso, como mencionado por Tanweer *et al.* (2020).

Em resumo, a ideia é ter soluções possíveis com esses cruzamentos e mutações das soluções que sejam melhores que as soluções iniciais. A imagem abaixo demonstra a ideia do algoritmo genético e quais são as etapas que compõem o algoritmo.

### 2.2.1 Codificação das soluções

A ideia do Algoritmo Genético é usar os mesmos conceitos da genética, onde cada indivíduo representa uma possível solução para o problema. As soluções de cada um são codificadas na forma de cromossomos, e esses cromossomos são constituídos de genes. Essa etapa, segundo Katoch, Chauhan e Kumar (2020), exerce um papel de extrema importância para o algoritmo, uma vez que garante que toda a população seja composta de cromossomos com as mesmas características, o que possibilita realizar as operações necessárias.

De maneira geral, os cromossomos são apresentados em uma lista com um tamanho igual ao total de demandas. A configuração do conteúdo do vetor foi feita de acordo com a realidade do problema e também para facilitar a leitura de dados e o desempenho do código.

Figura 4 – Fluxograma das etapas do algoritmo genético


Fonte: Elaborado pelo autor

### 2.2.2 População inicial

Para que seja possível implementar o algoritmo genético, é necessário ter uma população inicial; esta servirá como ponto de partida para todas as próximas etapas Rodrigues *et al.* (2004). A primeira etapa para gerar essa população é ter o grafo e a lista de demandas nas quais ela será gerada, como mostrado na seção 4.3.

Uma vez dado o grafo, a próxima etapa é encontrar os k caminhos mais curtos para cada uma das demandas do problema. O número k de caminhos pode ser definido livremente, mas deve ter caminhos suficientes para obter uma resposta possível no caso de superação da capacidade do enlace entre dois nós.

Com os k caminhos mais curtos encontrados para cada demanda, é necessário selecionar o método que será usado para definir a alocação do espectro e o roteamento de cada uma das demandas. Após isso, caso a alocação correta para todas as demandas tenha sido possível, haverá uma população inicial.

### 2.2.3 Avaliação

Uma vez que exista uma população, faz-se necessário avaliá-la, isto é, garantir que ela atende aos critérios que definem o problema. Desta maneira, esta etapa possui o objetivo de dizer se as soluções são aceitáveis para o problema, considerando o objetivo também definido Dianati, Song e Treiber (2020).

Ao avaliá-las, será possível não apenas dizer sobre sua possibilidade, mas também sobre o valor que ela representa de acordo com a função objetivo que foi selecionada.

### 2.2.4 Seleção

Como a população pode crescer muito, é necessário fazer uma seleção de tamanho reduzido para facilitar as operações subsequentes, melhorar o desempenho do código, o tempo de processamento e chegar cada vez mais perto da melhor solução possível que o método pode entregar Kumar *et al.* (2010).

A seleção é a etapa que antecede a mutação e o cruzamento. Esta função selecionará várias soluções possíveis, ou seja, cromossomos, da população total. Os resultados dessa seleção serão as soluções que sofrerão alterações nas etapas seguintes.

### 2.2.5 Cruzamento

O cruzamento realiza modificações entre os cromossomos enquanto mantém características cruciais de seus pais.

Portanto,a ideia é realizar o cruzamento com dois pais, ou seja, duas soluções que já são possíveis. Logo, segundo Mathew (2012), é por isso que os filhos terão características apenas dos dois pais, uma vez que a operação é sempre feita em pares.

### 2.2.6 Mutação

A mutação é uma parte importante do processo de busca pela solução ótima, pois realiza pequenas modificações nos cromossomos, permitindo a preservação da maioria de suas características. Apesar da existência de diferentes métodos, todos têm a mesma ideia principal.

Tal ideia está baseada no conceito explicado por Han e Xiao (2022), onde diz que o objetivo é selecionar um indivíduo da população e realizar mutações em um ou mais genes do cromossomo para gerar melhores indivíduos.

## 2.3 *Branch and Cut*

Muitos desafios de otimização combinatória podem ser representados matematicamente como problemas de programação linear inteira mista. O algoritmo *Branch and Cut*, uma abordagem híbrida que combina os métodos *Branch and Bound* e geração de planos de corte, é uma ferramenta poderosa para solucionar esses problemas. Por ter uma aplicação ampla, Balcan *et al.* (2022) diz que é um algoritmo poderoso e espinha dorsal de diversos solvers de programação inteira modernos.

Nessa técnica, o problema original é decomposto em subproblemas menores, formando uma árvore de busca Li *et al.* (2023). A cada nível da árvore, planos de corte são adicionados para aperfeiçoar a estimativa do valor ótimo da solução, reduzindo assim o número de nós a serem explorados e acelerando a convergência do algoritmo.

Diante da versatilidade deste método, é possível encontrar inúmeras aplicações para a resolução de problemas. Brucka, Coutinho e Munari (2024) propõem a formulação de um algoritmo *Branch and Cut* para resolver o problema de equilíbrio de um sistema de compartilhamento de bicicletas. Já Lam, Stuckey e Harabor (2024), propõem uma adaptação desse método para resolver o problema de *Multi-Agent Pickup and Delivery*, que é, em resumo, a abstração de controlar robôs em um galpão logístico automatizado.

Este trabalho, no entanto, busca utilizar também o *Branch and Cut* para solucionar um problema do tipo RSA, assim como no trabalho apresentado por Bianchetti e Marenco (2023), em que tal método foi proposto para a resolução do problema.

## 2.4 Linguagem Python

A linguagem escolhida foi Python, uma vez que a mesma possui uma sintaxe clara, concisa e simples, porém, além da simplicidade, é muito poderosa, permitindo a realização de grandes projetos, segundo Menezes (2010). O uso dessa linguagem permitirá um desenvolvimento de código mais rápido e compreensível. Esta linguagem é de primeira linha, orientada a objetos, dinâmica e fortemente tipada, interpretada e interativa.

Python também se apresenta como uma linguagem de alto nível, com simplicidade e clareza, conforme Yuill e Halpin (2006). Por exemplo, em outras linguagens de programação, utiliza-se muita marcação, como ponto (.) ou ponto e vírgula (;), no final de cada linha, além de marcadores de início e fim de bloco, como chaves ( ) ou palavras-chave específicas. Esses marcadores tornam os programas um pouco mais difíceis de ler e, felizmente, não são usados em Python.

Além disso, permite adicionar diversas bibliotecas e módulos ao código Borges (2014). Esta facilidade permitirá melhor processamento e manipulação de dados e melhor utilização de funções adicionais. E ainda, devido à sua ampla adesão em todo o mundo, existe vasto leque de suportes na Internet que proporcionam uma ajuda adicional muito interessante.

## 2.5 Biblioteca NetworkX

Dada a escolha da linguagem Python, a implementação foi possível graças à utilização da biblioteca *NetworkX* em Python. De acordo com a documentação oficial, disponível aqui, esta biblioteca possui código-fonte aberto para análises de redes e fornece ferramentas para criação, manipulação e análise de redes complexas, como a rede no problema RSA. Portanto, segundo Martins (2023), essa biblioteca permite maior controle sobre a topologia da rede como um todo e facilita suas operações.

*NetworkX* oferece uma ampla gama de algoritmos gráficos para análise e manipulação de redes. É possível realizar operações como encontrar caminhos mais curtos Silva (2024), calcular centralidade de nós, detectar comunidades, calcular componentes fortemente conectados, calcular árvores geradoras mínimas e diversas outras.

A criação dos gráficos utilizados no trabalho foi possível graças a diversas funções da biblioteca *NetworkX* que permitiram criar gráficos direcionados com nós e arestas específicas do nosso problema, além de permitir sua utilização nas funções do algoritmo genético.

## 2.6 Pacote Gurobipy

O pacote Gurobipy, uma interface Python para o solver de otimização GUROBI, é uma ferramenta poderosa para modelar e resolver uma vasta gama de problemas de otimização linear, não linear, inteira e mista, de acordo com informações oficiais, disponíveis aqui. Sua integração com Python, uma linguagem de programação popular e versátil, torna-o acessível a uma ampla comunidade de pesquisadores, engenheiros e cientistas de dados. Um exemplo de sua utilização está na operação cooperativa otimizada de sistema de energia fonte-rede-carga-armazenamento Qi *et al.* (2023), em que o solver GUROBI foi utilizado para resolver o problema.

Com este pacote, a usabilidade dos recursos do GUROBI é facilitada, uma vez que apresenta de forma simples todas as possibilidades. Da mesma forma, flexibiliza a utilização, já que pode ser usado juntamente com outros pacotes, como NumPy e pandas, para realizar um tratamento de dados mais eficiente.

## 2.7 Modelo matemático

Considerando a natureza do trabalho, entende-se que será necessário utilizar um modelo matemático como base para o GUROBI e o método *Branch and Cut*. Por conta disso, a abordagem proposta por Silva (2023) será usada como o modelo para o trabalho. A estruturação do problema se encontra nas linhas abaixo.

Um grafo direcionado $G'$ é construído a partir do grafo da rede $G$ criando um arco de entrada $(u,v)$ e um arco de saída $(v,u)$ para cada aresta $uv$ em $E(G)$. Para cada arco $a \in A(G')$, uma fibra é associada. Vamos denotar $f_a$ como a fibra associada a cada arco individual (a aresta original $e \in E$).

Uma vez apresentada a estruturação dos dados, serão apresentadas abaixo apenas as partes que compõem o modelo em si, sendo elas: as variáveis, funções objetivo e restrições.

### 2.7.1 Variáveis

A variável abaixo está presente para todas as funções objetivo do problema:

$f_{k,a,s} \in \{0,1\}$ se a demanda $k \in K$ é roteada através do arco $a \in A(G')$ utilizando o slot $s \in [\bar{s}]$.

As variáveis abaixo podem ser adicionadas a depender da função objetivo escolhida:

$p_e \in \mathbb{R}$ é a posição do maior slice de frequência utilizada na fibra $e \in E$.

$p \in \mathbb{R}$ é a posição do maior slice de frequência utilizada no geral.

### 2.7.2 Funções objetivo

As funções objetivo abaixo são as que foram trabalhadas durante o trabalho.

1. Minimizar a soma sobre a posição do último slice de frequência usado nas fibras para todas as demandas:

$$\min \sum_{e \in E} p_e$$

2. Minimizar o número de fibras usadas por todas as demandas:

$$\min \sum_{k \in K} \sum_{a \in A(G')} f_{k,a,s}$$

3. Minimizar o total de slots usados por todas as demandas:

$$\min \sum_{k \in K} \sum_{a \in A(G')} w_k \cdot f_{k,a,s}$$

4. Minimizar o comprimento total das rotas usadas por todas as demandas:

$$\min \sum_{k \in K} \sum_{a \in A(G')} l_a \cdot f_{k,a,s}$$

5. Minimizar a posição do maior slice de frequência usado no geral:

$$\min p$$

### 2.7.3 Restrições

**1. Restrição de Grau:** Para cada demanda $k$ e cada nó $v$, no máximo um slot é usado em um dos arcos que saem de cada nó.

$$\sum_{a \in \delta^+(v)} \sum_{s \in [\vec{s}]} f_{k,a,s} \leq 1 \quad \forall k \in K, \forall v \in V$$

**2. Restrição de Origem:** Para cada demanda $k$, apenas um slot é usado em um arco que sai do nó de origem $o_k$.

$$\sum_{a \in \delta^+(o_k)} \sum_{s \in [\vec{s}]} f_{k,a,s} - \sum_{a \in \delta^-(o_k)} \sum_{s \in [\vec{s}]} f_{k,a,s} = 1 \quad \forall k \in K$$

**3. Restrição de Destino:** Para cada demanda $k$, apenas um slot é usado em um arco que entra no nó de destino $d_k$.

$$\sum_{a \in \delta^+(d_k)} \sum_{s \in [\vec{s}]} f_{k,a,s} - \sum_{a \in \delta^-(d_k)} \sum_{s \in [\vec{s}]} f_{k,a,s} = -1 \quad \forall k \in K$$

**4. Conservação do fluxo:** Para cada demanda $k$, o número de slots usados e os arcos que entram em um nó $v \neq o_k \neq d_k$ é igual ao número de slots usados e os arcos que saem dele.

$$\sum_{a \in \delta^+(v)} f_{k,a,s} - \sum_{a \in \delta^-(v)} f_{k,a,s} = 0 \quad \forall k \in K, \forall v \in V(G) \setminus \{o_k, d_k\}, \forall s \in [\vec{s}]$$

**5. Restrição de Comprimento Máximo:** O comprimento do caminho de cada demanda deve respeitar seu alcance máximo:

$$\sum_{a \in A(G')} \sum_{s \in [\vec{s}]} l_a \cdot f_{k,a,s} \leq L_k \quad \forall k \in K$$

**6. Restrição de Não Sobreposição:** Um slice de frequência em uma fibra pode suportar no máximo uma demanda:

$$\sum_{k \in K} \sum_{\substack{a \in A(G'): \\ s \leq s' \leq s + w_k - 1 \\ f_a = e}} f_{k,as'} \leq 1 \quad \forall e \in E, \forall s \in [\vec{s}]$$

As restrições abaixo são necessárias apenas se algumas funções objetivo forem escolhidas. A primeira é necessária se a função 1 for escolhida e, a segunda, se a função 5 for escolhida.

**7. Maior slice de frequência por fibra.** A posição do maior slice de frequência usado em uma fibra é maior do que todas as posições de slice de frequência usadas nela:

$$\sum_{a \in A(G')} (f_a = e) \cdot s_a \cdot f_{k,as} \leq p_e \quad \forall k \in K, \forall e \in E$$

**8. Maior slice de frequência no geral.** A posição do maior slice de frequência usado no geral é maior do que todas as posições de slice de frequência usadas (pode ser reforçada):

$$\sum_{a \in \delta^+(v)} s_a \cdot f_{k,as} \leq p \quad \forall k \in K$$

# 3 Metodologia

Nesta etapa, serão explicados os pontos que compõem a metodologia da pesquisa. Desta forma, haverá uma breve explicação quanto à sua classificação e um panorama geral dos dados que serão utilizados.

## 3.1 Classificação da pesquisa

De acordo com Nascimento (2019) pode-se classificar esta pesquisa como básica, uma vez que tem o objetivo de gerar novas possibilidades para a resolução de problemas similares. Nesta mesma linha, Gomes e Gomes (2019) também diz que a pesquisa básica faz evoluir a fronteira do conhecimento, isto é, expande os conceitos que envolvem aquele assunto.

Quanto à abordagem, pode-se dizer que esta se caracteriza como uma pesquisa quantitativa, já que utiliza dados e modelos para compreender e aplicar os conceitos relacionados. Portanto, seguindo os apontamentos de Fontelles *et al.* (2009), onde diz que seus resultados são passíveis de generalização, isto é, utilização futura para outros problemas similares, tal abordagem é considerada. Ainda em complemento, esse tipo de pesquisa também diz que há complemento de novas ideias e teorias que contribuam para o avanço da área Berto e Nakano (1998).

Em relação ao método, segundo Nakano *et al.* (2012) a pesquisa se classifica como "modelagem". Essa classificação se dá em razão da utilização de técnicas matemáticas para descrever ao menos uma parte do problema.

Por fim, em relação ao objetivo, pode-se dizer que é exploratória, já que busca trazer mais e novas informações ao problema, contribuindo assim para a descoberta de soluções mais eficientes. Esse tipo de pesquisa se caracteriza, portanto, pelo levantamento bibliográfico sobre o assunto e consequente aplicação dos conceitos ao problema em si, como realizado por Gerhardt e Silveira (2009).

## 3.2 Entrada de dados do problema

Para viabilizar a utilização do algoritmo, é necessária a leitura de todos os dados que serão utilizados. São dois arquivos que serão lidos: o primeiro refere-se à rede óptica, e o segundo refere-se à lista de todas as solicitações(demandas) desta rede. A lista de requisições representa o problema que o código precisa resolver e tem diferentes tamanhos, porém o código é capaz de encontrar soluções possíveis para qualquer conjunto de requisições.

Todos os dados utilizados para fazer os testes durante o trabalho podem ser encontrados no GITHUB criado especificamente para este projeto. Portanto, tanto as instâncias do gráfico quanto as instâncias de solicitação podem ser recuperadas lá.

### 3.2.1 Arquivo do gráfico da rede

O arquivo que representa o gráfico contém colunas que formam a seguinte estrutura:

1. Índice
2. Nó de origem do link
3. Nó de destino do link
4. O comprimento entre origem e destino
5. A capacidade de espectro deste link

Tabela 1 – Topologia do arquivo das redes

| **Índice** | **Origem** | **Destino** | **Tamanho** | **Slices** |
|---|---|---|---|---|
| **1** | 1 | 2 | 94 | 80 |
| **2** | 1 | 3 | 11 | 80 |
| **3** | 2 | 3 | 34 | 80 |
| **4** | 2 | 4 | 82 | 80 |
| **5** | 3 | 4 | 103 | 80 |
| **6** | 3 | 5 | 47 | 80 |
| **7** | 4 | 5 | 86 | 80 |
| **8** | 4 | 6 | 72 | 80 |
| **9** | 5 | 6 | 77 | 80 |

### 3.2.2 Arquivo da demanda

O arquivo das demandas contém as colunas que formam a seguinte estrutura:

1. Índice
2. Nó de origem da demanda
3. Nó de destino da demanda
4. Número necessário de slots no espectro da demanda
5. Distância máxima que esta solicitação pode percorrer na rede

Tabela 2 – Topologia do arquivo de demandas

| Índice | Origem | Destino | Slots | Distância Máxima |
|---|---|---|---|---|
| **1** | 1 | 2 | 3 | 2000 |
| **2** | 1 | 3 | 3 | 2000 |
| **3** | 1 | 4 | 2 | 2000 |
| **4** | 1 | 5 | 2 | 2000 |
| **5** | 1 | 6 | 3 | 2000 |
| **6** | 2 | 1 | 2 | 2000 |
| **...** | ... | ... | ... | ... |
| **30** | 6 | 5 | 3 | 2000 |

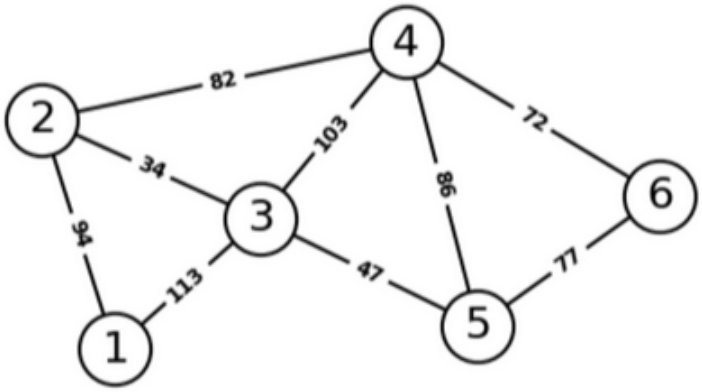
## 3.3 Instância utilizada

Durante o trabalho, todas as análises foram feitas exclusivamente em uma rede. A utilização desta rede permite uma maior abrangência dos testes, uma vez que ela possui poucos fatores limitantes e complicantes.

Esta escolha foi feita para facilitar o teste das ideias iniciais, já que, como dito anteriormente, possui um baixo nível de complexidade. Desta forma, ela é caracterizada por possuir apenas seis nós, que podem ser interpretados como cidades, e 9 ligações entre eles, que podem ser interpretados como os cabos que as ligam. Estas ligações podem ser chamadas também de fibras. As distâncias entre esses nós também estão presentes, pois são cruciais na busca pela solução ótima, a depender do objetivo escolhido. Tal rede foi denominada 6x9.

Assim, a imagem abaixo mostra o gráfico da rede 6x9, com os nós 1, 2, 3, 4, 5 e 6, as conexões existentes entre os nós e as distâncias entre eles.

Figura 5 – Rede óptica 6x9




Fonte: Elaborado pelo autor

# 4 Resultados

Primeiramente, é importante mencionar que os resultados foram obtidos utilizando uma máquina Windows 64 bits, com 20 GB de RAM e processador Intel Core i5 octa-core de 2,40 GHz. Os testes foram realizados para diferentes objetivos, no algoritmo genético, no Gurobi e no método integrado de ambos.

## 4.1 Adaptação do Algoritmo Genético ao Problema

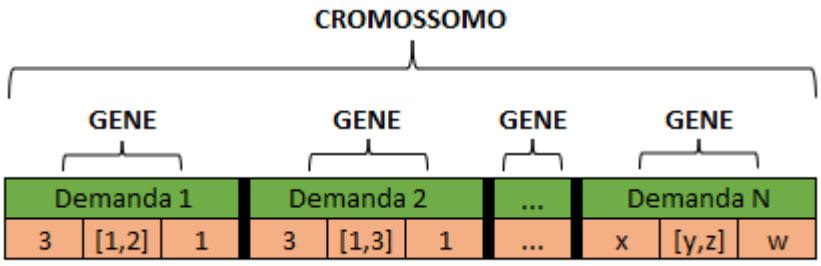
### 4.1.1 Codificação das soluções

Como mencionado, a população é composta por vários cromossomos e cada um desses cromossomos representa uma possível solução para o problema. Cada cromossomo é composto de genes, e cada um desses genes representa uma demanda do problema. O comprimento do cromossomo é igual ao número de demandas, e o comprimento do gene é sempre o mesmo. Este comprimento do gene é definido como três, e esses três elementos são os seguintes:

- 1º: O número de slots necessários para a demanda
- 2º: O caminho escolhido para a demanda
- 3º: O primeiro slot de frequência que será utilizado para a demanda

Por essa razão, o exemplo da estrutura de um cromossomo e de um gene do projeto se assemelha à imagem abaixo:

Figura 6 – Representação da codificação das soluções




Fonte: Elaborado pelo autor

### 4.1.2 População Inicial

Como explicado anteriormente, faz-se necessário a criação de uma população inicial para dar início ao algoritmo. Neste trabalho, quatro maneiras diferentes foram propostas, e cada uma tem suas características específicas. Os métodos e suas características são:

- **nonerandom:** o método utiliza o caminho mais curto para verificar se há slots disponíveis para todos os links que percorrem o caminho. Se houver slots disponíveis, a demanda é atribuída a esses slots e as fibras desse caminho; caso contrário, o próximo caminho mais curto é escolhido e a verificação começa novamente. Se o método percorrer todos os caminhos e todos os slots e não encontrar uma atribuição possível, pode-se dizer que a geração da população inicial usando esse método é impossível.

- **pathrandom:** o método escolhe um caminho aleatoriamente para verificar se há slots disponíveis para todos os links que percorrem o caminho. Se houver slots disponíveis, a demanda é atribuída a esses slots e a este caminho. Caso contrário, o caminho é alterado aleatoriamente novamente e a verificação do slot começa desde o início. Se o método percorrer todos os caminhos e todos os slots e não encontrar uma atribuição possível, é possível afirmar que a geração da população inicial usando este método é impossível.

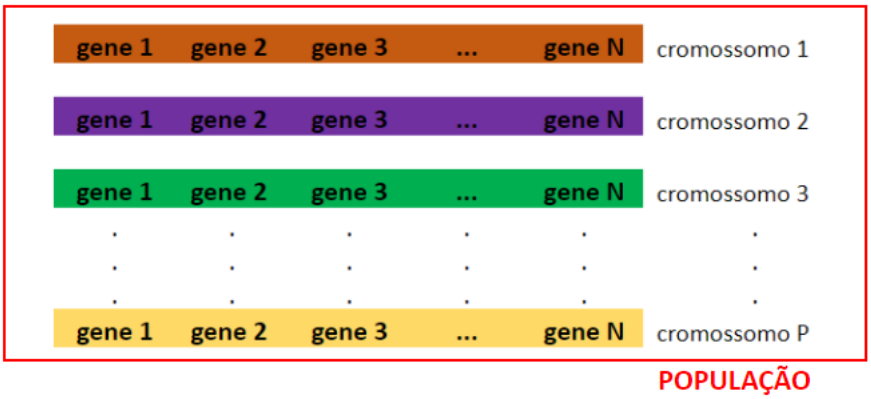
- **slotrandom:** o método utiliza o caminho mais curto e um slot que é escolhido aleatoriamente. Se os slots de todos os links do caminho não estiverem ocupados, a demanda é atribuída a esses slots e as fibras desse caminho. Caso contrário, um novo slot é escolhido aleatoriamente e testado. Se todos os slots deste caminho forem testados e não for possível uma atribuição, o próximo caminho mais curto é escolhido. Se o método percorrer todos os caminhos e todos os slots e não encontrar uma atribuição possível, é possível afirmar que a geração da população inicial usando este método é impossível.

- **bothrandom:** o método escolhe o caminho e o slot aleatoriamente. Se os slots de todos os links do caminho não estiverem ocupados, a demanda é atribuída a esses slots e a esse caminho. Caso contrário, um novo slot é escolhido aleatoriamente e testado. Se todos os slots deste caminho forem testados e não for possível uma atribuição, outro caminho é escolhido aleatoriamente e a verificação do slot começa novamente aleatoriamente. Se o método percorrer todos os caminhos e todos os slots e não encontrar uma atribuição possível, é possível afirmar que a geração da população inicial usando este método é impossível.

Para utilizar qualquer uma dessas metodologias, é necessário informar o tamanho da população que deseja, a lista de demandas a serem resolvidas e, por fim, a rede na qual se trabalhará. Assim, a função retornará uma população viável ou indicará que sua geração não foi possível.

Dada a definição de cada conceito relacionado à população inicial, a imagem abaixo demonstra os conteúdos de uma população inicial e como ela pode ser. Também mostra que o número de demandas é igual a N, que o tamanho da população é igual a P, e que todos os cromossomos têm o mesmo número de genes, já que o número de demandas é sempre o mesmo.

Figura 7 – Representação de uma População




Fonte: Elaborado pelo autor

### 4.1.3 Avaliação

Esta função é responsável por avaliar, isto é, calcular o valor de acordo com a função objetivo. Após a inicialização da população e após cada iteração, os indivíduos são avaliados com base nos parâmetros da função objetivo para identificar a melhor solução viável.

Cinco maneiras diferentes de avaliar uma solução foram propostas, e essas diferentes maneiras referem-se às diferentes funções objetivos. Cada uma retornará um número como resultado, que será único para cada objetivo escolhido. No entanto, uma vez escolhido o objetivo, todas as soluções terão resultados similares, isto é, de mesma grandeza de valor.

Desse modo, temos que as cinco funções objetivo são:

- **fiberslastfrequency:** Este objetivo visa minimizar a soma na última posição da faixa de frequência utilizada pela fibra. Assim, o resultado será a soma total do último slot utilizado por cada fibra da rede.
- **numberofhops:** Este objetivo visa minimizar o número de saltos, ou seja, as fibras utilizadas por todas as demandas. Assim, o resultado será o total de slots requisitados multiplicado pelo número de fibras utilizadas por cada demanda.
- **totalofslotsused:** Este objetivo visa minimizar o total de slots utilizados. Assim, o resultado será a soma total dos slots requisitados multiplicada pelo número de saltos utilizados por cada demanda.

- **totallength:** Este objetivo visa minimizar o comprimento total dos caminhos. Assim, o resultado será a soma total do comprimento dos caminhos escolhidos para cada demanda.
- **highestfrequencyslice:** Este objetivo visa minimizar a posição da faixa de frequência mais alta utilizada por todas as demandas. Assim, o resultado será simplesmente a posição da faixa de frequência mais alta utilizada, após comparação entre todas as demandas.

É importante ressaltar que a função de avaliação não tem como objetivo encontrar a melhor solução para nenhum dos objetivos. Ela apenas calcula os resultados e, depois de seguir todas as etapas, o algoritmo genético fornecerá a melhor solução encontrada, podendo ela ser ótima ou não.

### 4.1.4 Seleção

Três métodos foram propostos para esta etapa do algoritmo. O parâmetro 'n', que representa o tamanho da seleção, é necessário para todos os métodos. Também é importante ressaltar que os resultados usados nesta função são os resultados fornecidos pela função de avaliação, logo antes desta etapa.

Os métodos e suas características são:

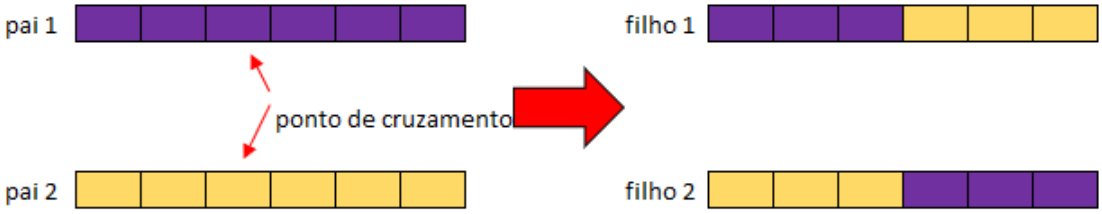
- **torneio:** esse método realizará um torneio, e as melhores n soluções serão selecionadas por ela. Por exemplo, se o tamanho da seleção for igual a 50, as 50 melhores soluções da população serão selecionadas.
- **aleatória:** esse método fará a seleção de forma aleatória, e serão selecionadas n soluções aleatórias da população. Por exemplo, se o tamanho da seleção for igual a 50, 50 soluções aleatórias da população serão selecionadas.
- **melhor:** esse método encontrará a melhor solução na população e, em seguida, ela será copiada n vezes. Por exemplo, se o tamanho da seleção for igual a 50, a melhor solução da população será selecionada 50 vezes.

### 4.1.5 Cruzamento

Três métodos diferentes também foram propostos para o cruzamento. Os métodos e suas características podem ser descritas da seguinte forma:

- **Cruzamento em um único ponto:** Um ponto é escolhido para realizar o cruzamento e, em seguida, a troca entre os cromossomos é feita. É importante destacar que o ponto escolhido deve ser o mesmo para os dois cromossomos, para que as características sejam mantidas.

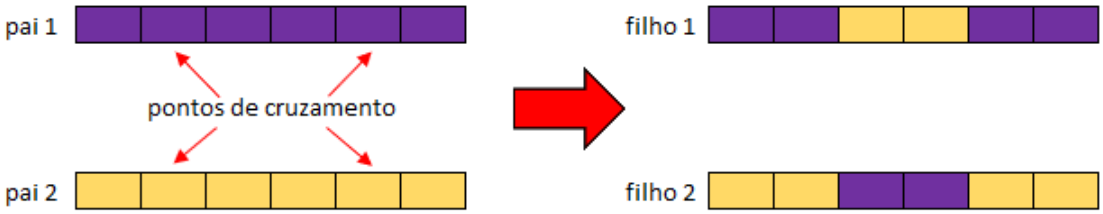
Figura 8 – Representação do cruzamento em um único ponto




Fonte: Elaborado pelo autor

- **Cruzamento em N pontos:** Vários pontos são escolhidos para realizar o cruzamento e, em seguida, a troca entre os cromossomos é feita, seguindo a mesma regra utilizada no cruzamento de ponto único.

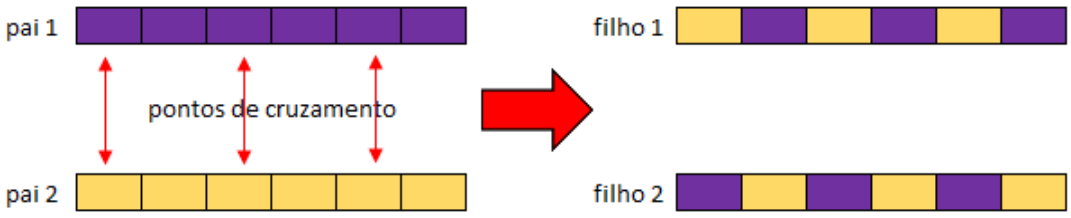
Figura 9 – Representação do cruzamento em N pontos




Fonte: Elaborado pelo autor

- **Cruzamento uniforme:** cada gene de um cromossomo é copiado para o outro até que não haja mais possibilidade de mudança, ou seja, cada par de cromossomos é trocado.

Figura 10 – Representação do cruzamento uniforme




Fonte: Elaborado pelo autor

Os cruzamentos são sempre realizados, mas os filhos gerados só são adicionados à população total se atenderem todas as restrições do problema. Como há várias restrições, adicionar novas soluções não é um processo simples.

Portanto, os filhos são testados separadamente e, se forem considerados soluções viáveis, são adicionados à população total. É importante destacar que, como são testados separadamente, um filho pode ser adicionado e o outro não.

### 4.1.6 Mutação

Para mutação, os métodos propostos foram:

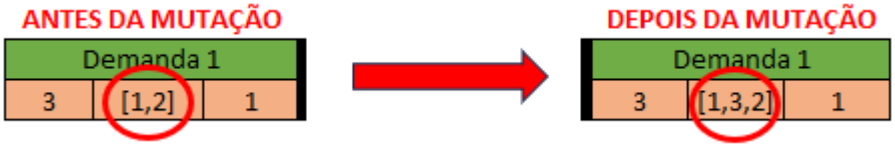
- ***slot:*** realizará uma modificação aleatória nos slots ocupados de uma demanda.

Figura 11 – Representação da mutação do slot


Fonte: Elaborado pelo autor

- ***path:*** realizará uma modificação aleatória no caminho utilizado por uma demanda.

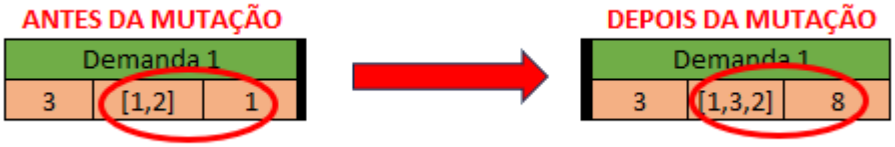
Figura 12 – Representação da mutação do caminho




Fonte: Elaborado pelo autor

- ***both:*** realizará uma modificação aleatória tanto no caminho utilizado quanto nos slots ocupados por uma demanda.

Figura 13 – Representação da mutação do slot e do caminho




Fonte: Elaborado pelo autor

A mutação é sempre realizada, mas o resultado da mutação é adicionado à população total se, e somente se, respeitar todas as restrições do problema. Se a mutação resultar em uma solução viável, ela será incluída na população total; caso contrário, será descartada sem alterar a população.

A mutação segue o mesmo princípio da operação de cruzamento, mas como as mudanças são simples, é mais provável que o resultado da mutação seja adicionado a população.

## 4.2 Resolução matemática pelo GUROBI

Para fins de comparação dos resultados, foi-se necessário também criar o código baseado no modelo descrito na seção 2.7. Para isso, foram utilizadas as funções e as funcionalidades que o GUROBI proporciona para implementar de fato todas as linhas do modelo. Não foram necessárias adaptações quanto à utilização, apenas interpretação do problema em conjunto com a utilização do pacote.

## 4.3 Performance GUROBI e Algoritmo Genético

Dado que temos as duas maneiras de solucionar o problema, pode-se finalmente comparar a eficiência de ambos. A integração entre os métodos cria, por si só, uma nova abordagem para resolver o problema. Logo, testes em relação ao tempo de processamento e à solução encontrada foram realizados para poder então comparar os resultados.

Os testes foram definidos de forma idêntica para os métodos, isto é, as mesmas funções objetivo e lista de demandas serão processadas. Por exemplo, para uma lista de 30 demandas, na rede 6x9, foi testada a função objetivo *"total length"* em todos os três métodos . A partir disso, encontraram-se os resultados referentes à função objetivo e o tempo de processamento.

Para o Algoritmo Genético, o método de geração da população inicial utilizado foi o *"none random"*, que se mostrou o mais eficaz durante os testes. Os outros parâmetros do algoritmo serão: seleção aleatória, cruzamento uniforme e mutação do caminho e slot.

Já em relação à utilização do GUROBI, não existem parâmetros a serem definidos previamente. Dado o modelo matemático, basta processar os dados junto ao modelo para obter as respostas necessárias.

No método integrado, faz-se necessário uma adaptação. Funciona da seguinte forma: o resultado do algoritmo genético servirá como entrada para o solver GUROBI. Portanto, o valor da função fornecido pelo algoritmo genético vai gerar um limite na busca de soluções, reduzindo assim a busca pela solução ótima. Em suma, o valor fornecido pelo algoritmo genético será o *upper bound* no GUROBI.

Há ainda um ponto em comum para todos estes métodos. Tendo em mente a complexidade do problema, um limite de tempo de 3600 segundos foi estabelecido para o tempo de processamento. Desta forma, há ocorrências em que a solução ótima não foi encontrada devido a tal limitação.

Considerando isso, a tabela abaixo representa os valores encontrados para todos os métodos, diferentes funções objetivos e vários sets de demandas. Todas as informações estão agregadas apenas na tabela abaixo.

Tabela 3 – Resultados encontrados dos métodos

| Set de demandas | Métodos utilizados | total_of_ slots_used | | total_ length | | number_ of_hops | | highest_ frequency_slice | | fibers_last_ frequency | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | result. | tempo | result. | tempo | result. | tempo | result. | tempo | result. | tempo |
| **30 demandas** | **Alg. Genético** | 93 | 175ms | 3280 | 145ms | 44 | 101ms | 19 | 116ms | 110 | 446ms |
| | **Gurobi** | 93 | 1.27s | 3280 | 1.27s | 44 | 1.27s | 10 | 4.07s | 69 | 3600s |
| | **Integrado** | 93 | 1.31s | 3280 | 1.4s | 44 | 1.35s | 10 | 3.28s | 68 | 3600s |
| **50 demandas** | **Alg. Genético** | 171 | 168ms | 6231 | 238ms | 77 | 188ms | 35 | 136ms | 253 | 571ms |
| | **Gurobi** | 171 | 2.4s | 6231 | 2.54s | 77 | 2.34s | 23 | 29.9s | 156 | 3600s |
| | **Integrado** | 171 | 2.55s | 6231 | 2.65s | 77 | 2.55s | 23 | 12.5s | 153 | 3600s |
| **70 demandas** | **Alg. Genético** | 228 | 214ms | 8405 | 231ms | 107 | 229ms | 45 | 223ms | 279 | 736ms |
| | **Gurobi** | 228 | 3.62s | 8405 | 3.36s | 107 | 3.22s | 29 | 55.8s | 207 | 3600s |
| | **Integrado** | 228 | 4.21s | 8405 | 4.05s | 107 | 4.05s | 29 | 40.5s | 206 | 3600s |
| **80 demandas** | **Alg. Genético** | 254 | 251ms | 9337 | 250ms | 119 | 270ms | 50 | 389ms | 295 | 868ms |
| | **Gurobi** | 254 | 3.88s | 9337 | 4.2s | 119 | 3.82s | 34 | 57.8s | 275 | 3600s |
| | **Integrado** | 254 | 4.16s | 9337 | 4.49s | 119 | 4.58s | 34 | 30s | 275 | 3600s |
| **100 demandas** | **Alg. Genético** | 361 | 300ms | 11771 | 348ms | 151 | 290ms | 72 | 321ms | 426 | 1.24s |
| | **Gurobi** | 361 | 4.92s | 11771 | 5.06s | 151 | 4.95s | 49 | 147s | 337 | 3600s |
| | **Integrado** | 361 | 5.85s | 1171 | 5.93s | 151 | 8.1s | 49 | 168s | 337 | 3600s |

Fonte: Elaborado pelo autor

Na tabela acima, os valores de resultados em verde representam o valor ótimo para o problema em questão. Já os valores em vermelho mostram o pior tempo de processamento entre os métodos para o problema em questão. Importante ressaltar que existe a comparação do tempo apenas quando a solução ótima foi encontrada.

Assim sendo, nota-se que quando o algoritmo genético é capaz de encontrar a solução ótima, ele é mais rápido que o GUROBI e o método integrado. Desta forma, temos que o algoritmo genético se mostrou mais eficiente para as seguintes funções objetivo: *total of slots used, total length* e *number of hops*. A diferença no tempo de processamento é da grandeza de 10 vezes; isto significa que o algoritmo genético é capaz de encontrar a solução ótima 10 vezes mais rápido que o GUROBI para estas três funções objetivo.

Contudo, o algoritmo genético não se mostrou tão eficiente no que diz respeito às outras funções objetivo, estas sendo: *highest frequency slice* e *fibers last frequency*. Para estas, o algoritmo não foi capaz de encontrar a solução ótima. Portanto, o modelo matemático implementado no GUROBI foi mais eficiente. Quando não encontrou a solução ótima, forneceu uma solução melhor.

Com isso em mente, destaca-se uma diferença entre estas duas funções. Para *highest frequency slice*, apenas a implementação no GUROBI foi capaz de encontrar a solução ótima. Porém, o método integrado se mostrou mais eficiente. Dado um limite para o problema, limite esse fornecido pelo algoritmo genético, encontrou-se a solução ótima mais rapidamente do que usando o modelo matemático sem nenhuma restrição extra. Portanto, a integração neste caso foi benéfica para a resolução do problema.

No que tange ao objetivo *fibers last frequency*, nota-se que é um problema mais complexo, dado o alcance do limite máximo de processamento. Tanto para o método integrado quanto para o modelo matemático, o limite de 3600 segundos foi atingido antes de encontrar a solução ótima.

# 5 Considerações Finais

Este trabalho buscou abordar o problema RSA em redes ópticas, utilizando uma abordagem híbrida que integra Algoritmos Genéticos e o solver Gurobi. A proposta teve como objetivo oferecer uma solução eficiente para um problema classificado como NP-difícil, no qual a complexidade cresce exponencialmente com o tamanho da instância analisada.

A implementação do Algoritmo Genético possibilitou a obtenção de soluções iniciais de qualidade superior às geradas aleatoriamente, proporcionando um ponto de partida mais eficiente para a resolução do problema. Posteriormente, o método *Branch and Cut*, por meio do solver Gurobi, refinou essas soluções, buscando a otimização completa do problema sempre que viável.

Os resultados obtidos demonstraram que a combinação dos dois métodos pode ser uma estratégia promissora para lidar com o RSA, equilibrando a exploração do espaço de soluções com a busca pela melhor resposta possível dentro das restrições do problema. A integração entre heurísticas evolucionárias e métodos exatos possibilitou reduzir o tempo de processamento sem comprometer a qualidade das soluções encontradas em alguns casos.

Além disso, este estudo contribui para o entendimento da aplicabilidade de diferentes abordagens computacionais no contexto de otimização de redes ópticas. A análise comparativa entre os métodos implementados evidenciou os benefícios e desafios inerentes a cada um deles, oferecendo *insights* para futuras pesquisas na área.

Dessa forma, este trabalho demonstra a viabilidade do desenvolvimento de técnicas híbridas para a resolução de problemas complexos, em particular, a aplicação combinada de Algoritmos Genéticos e programação matemática no aprimoramento do desempenho de redes ópticas. Além disso, a metodologia proposta abre caminho para novos estudos que possam expandir e aprimorar essa abordagem.

Por fim, o trabalho abre um leque de opções no que tange à exploração de diferentes estratégias híbridas para a resolução do problema. Ainda dentro do contexto dos Algoritmos Genéticos, a investigação de outras heurísticas pode contribuir para a otimização do processo e gerar ainda mais informações para o problema RSA

# Referências


ARAUJO, C. M. d. O.; SUBRAMANIAN, A.; EDUARDO da F. I. Um modelo de programac¸ ~ao linear inteira para o problema rsa em redes opticas el´asticas. **SBPO**, 2017. Disponível em: <http://ws2.din.uem.br/~ademir/sbpo/sbpo2017/pdf/169510.pdf>.

BALCAN, M.-F.; PRASAD, S.; SANDHOLM, T.; VITERCIK, E. Improved sample complexity bounds for branch-and-cut. **Cornell University**, 2022. Disponível em: <https://arxiv.org/pdf/2111.11207>.

BERTO, R.; NAKANO, D. Metodologia da pesquisa e a engenharia de produÇÃo. **ABEPRO**, 1998. Disponível em: <https://abepro.org.br/biblioteca/enegep1998_art174.pdf>.

BIANCHETTI, M.; MARENCO, J. A branch-and-cut algorithm for the routing and spectrum allocation problem. **Elsevier**, 2023. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0166218X23002366?casa_token=kWb9zLrKjlYAAAAA:8zZRveyBBdfF3FJArPxWGTPSLidNej8Ck47tZuw8ar_pdUphRuPbzxNa9IXiT_TcDwPmNCwxHgQ>.

BORGES, L. E. **Python para desenvolvedores: aborda Python 3.3.** [S.l.]: Novatec Editora, 2014.

BRUCKA, B. P.; COUTINHO, W. P.; MUNARI, P. The robust bike sharing rebalancing problem: Formulations and branch-and-cut algorithm. **Optimization Online**, 2024. Disponível em: <https://optimization-online.org/wp-content/uploads/2023/10/BruckEtAl2023-OO3.pdf>.

CAMARGO, V. C. B.; TOLEDO, F. M. B. Planejamento da produÇÃo em fundiÇÕes - um algoritmo genÉtico. **SBPO**, 2008. Disponível em: <http://ws2.din.uem.br/~ademir/sbpo/sbpo2008/pdf/arq0044.pdf>.

CHOUDHURY, P. D.; RAKESH, R. P.; CHATTERJEE, B. C.; OKI, E. Performance of routing and spectrum allocation approaches for multicast traffic in elastic optical networks. **Elsevier**, 2020. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1068520020302376>.

COLARES, R.; KERIVIN, H.; WAGLER ANNEGRET, P. A.; FLEURY, A.; MELLO, C. H. P.; NAKANO, D. N. An extended formulation for the constrained routing and spectrum assignment problem in elastic optical networks. **Open Proceedings**, 2022. Disponível em: <https://openproceedings.org/2022/conf/alioeuro/ALIOEURO_2021_paper_5.pdf>. Acesso em: 9 set. 2024.

CUNHA, C. B. da; BONASSER, U. d. O.; ABRAHãO, F. T. M. Experimentos computacionais com heurÍsticas de melhorias para o problema do caixeiro viajante. **ongresso da Anpet – Associação Nacional de Pesquisa e Ensino em Transportes**, 2002. Disponível em: <https://www.researchgate.net/profile/Claudio-Cunha-3/publication/228434832_Experimentos_computacionais_com_heuristicas_de_melhorias_para_o_problema_do_caixeiro_viajante/links/54803ccb0cf2ccc7f8bb2c18/Experimentos-computacionais-com-heuristicas-de-melhorias-para-o-problema-do-caixeiro-viajante.pdf>.

DAOUADI, M. I. A genetic algorithm for the routing and spectrum assignment problem. 2022.

DIANATI, M.; SONG, I.; TREIBER, M. An introduction to genetic algorithms and evolution strategies. **University of Waterloo,**, 2020.

FONTELLES, M.; SIMÕES, M.; FARIAS, S.; FONTELLES, R. Metodologia da pesquisa cientÍfica: Diretrizes para a elaboracao de um protocolo de pesquisa. **USP**, 2009. Disponível em: <https://edisciplinas.usp.br/pluginfile.php/3049277/mod_resource/content/1DIRETRIZES%20PARA%20A%20ELABORA%C3%87%C3%83O%20DE%20UM%20PROJ%20PESQUISA.pdf>. Acesso em: 9 set. 2024.

GERHARDT, T. E.; SILVEIRA, D. T. Métodos de pesquisa. **Universidade Federal do Rio Grande do Sul**, 2009. Disponível em: <https://lume.ufrgs.br/bitstream/handle/10183/213838/000728731.pdf>. Acesso em: 9 set. 2024.

GOLFETO, R. R.; MORETTI, A. C.; NETO, L. L. d. S. Algoritmo genÉtico simbiÓtico aplicado ao problema de corte unidimensional. **SBPO**, 2007. Disponível em: <http://www.din.uem.br/~ademir/sbpo/sbpo2007/pdf/arq0132.pdf>.

GOMES, A.; GOMES, C. Classificação dos tipos de pesquisa em informática na educação. **Research Gate**, 2019. Disponível em: <https://www.researchgate.net/profile/Alex-Gomes-11/publication/333603263_Classificacao_dos_Tipos_de_Pesquisa_em_Informatica_na_Educacao/links/5cf650ff4585153c3db1ea61/Classificacao-dos-Tipos-de-Pesquisa-em-Informatica-na-Educacao.pdf>.

HAN, S.; XIAO, L. An improved adaptive genetic algorithm. **SHS Web of Conferences**, 2022. Disponível em: <https://www.shs-conferences.org/articles/shsconf/pdf/2022/10/shsconf_iteme2022_01044.pdf>.

KATOCH, S.; CHAUHAN, S.; KUMAR, V. A review on genetic algorithm: past, present, and future. **Multimed Tools Appl 80**, 2020. Disponível em: <https://link.springer.com/article/10.1007/s11042-020-10139-6>.

KERIVIN, H.; CHOUMAN, H.; GRAVEY, A.; GRAVEY, P.; HADHBI, Y.; MORVAN, M.; WAGLER, A. K. Impact of rsa optimization objectives on optical network state. **HAL SCIENCE**, 2021. Disponível em: <https://uca.hal.science/hal-03155966>.

KUMAR, M.; HUSIAN, M.; UPRETI, N.; GUPTA, D. Genetic algorithm: Review and application. **International Journal of Information Technology and Knowledge Management**, 2010.

LAM, E.; STUCKEY, P. J.; HARABOR, D. Optimal multi-agent pickup and delivery using branch-and-cut-and-price. **Monash University**, 2024. Disponível em: <https://ed-lam.com/papers/bcpmapd2024.pdf>.

LI, S.; OUYANG, W.; PAULUS, M. B.; WU, C. Learning to configure separators in branch-and-cut. **37th Conference on Neural Information Processing Systems (NeurIPS 2023)**, 2023.

MARTINS, V. O. Optimization of synchronization in multilayer networks. **UNICAMP**, 2023. Disponível em: <https://repositorio.unicamp.br/acervo/detalhe/1274021>.

MATHEW, T. V. Genetic algorithm. **Indian Institute of Technology Bombay,**, 2012. Disponível em: <http://datajobstest.com/data-science-repo/Genetic-Algorithm-Guide-[Tom-Mathew].pdf>. Acesso em: 10 set. 2024.

MENEZES, N. N. C. **Introdução à Programação com Python**. [S.l.]: Novatec Editora, 2010.

NAKANO, D.; FLEURY, A.; MELLO, C.; LIMA, E. P. de; TURRION, J. B.; HO, L. L.; MORABITO, R.; MARTINS, R. A.; SOUSA, R.; COSTA, S. E. G. da; PUREZA, V. **METODOLOGIA DE PESQUISA EM ENGENHARIA DE PRODUÇÃO E GESTÃO DE OPERAÇÕES**. [S.l.]: Elsevier, 2012.

NASCIMENTO, F. P. do. Classificação da pesquisa. natureza, método ou abordagem metodológica, objetivos e procedimentos. 2019. Disponível em: <https://www.franciscopaulo.com.br/arquivos/Classificando%20a%20Pesquisa.pdf>. Acesso em: 9 set. 2024.

QI, X.; LIU, L.; ZHANG, Z.; PAN, Z.; WU, W. Study on the cooperative optimized operation of power system source-grid-load-storage based on gurobi mathematical programming. **IEEE**, 2023.

RODRIGUES, F. L.; LEITE, H. G.; SANTOS, H. d. N.; SOUZA, A. L. d.; SILVA, G. F. d. MetaheurÍstica algoritmo genÉtico para soluÇÃo de problemas de planejamento florestal com restriÇÕes de integridade. **Scielo Brasiç**, 2004. Disponível em: <https://www.scielo.br/j/rarv/a/GjchDFjXJj87v3bmMgVZdNC/?lang=pt#>. Acesso em: 10 set. 2024.

SILVA, L. S. Uso de grafos de visibilidade para tomada de decisões de investimentos na bolsa de valores brasileira. **Projeto final do curso - Universidade Federal do Rio Grande do Su**, 2024.

SILVA, P. H. F. da. Updates on the routing and spectrum assignment problem. **University Clermont Auvergne**, 2023.

TANWEER, A.; SHAMIMUL, Q.; AMIT, D.; MOHAMED, B. Genetic algorithm: Reviews, implementations, and applications. **International Journal of Engineering Pedagogy (iJEP)**, 2020. Disponível em: <https://arxiv.org/pdf/2007.12673>.

YUILL, S.; HALPIN, H. Python. **Free Software Foundation**, 2006. Disponível em: <https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=1f2ee3831eebfc97bfafd514ca2abb7e2c5c86bb>. Acesso em: 9 set. 2024.