Universidade do Estado do Rio de Janeiro
Centro de Tecnologia e Ciências
Faculdade de Engenharia

Heitor Teixeira de Azambuja

Casamento de Comprimento de Trilhas de Placas de Circuito Impresso Utilizando Otimização por Colônia de Formigas

Rio de Janeiro
2025

Heitor Teixeira de Azambuja

Casamento de Comprimento de Trilhas de Placas de Circuito Impresso Utilizando Otimização por Colônia de Formigas

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientadora: Prof.ª Dr.ª Nadia Nedjah
Orientadora: Prof.ª Dr.ª Luiza de Macedo Mourelle

Rio de Janeiro
2025

CATALOGAÇÃO NA FONTE

UERJ / REDE SIRIUS / BIBLIOTECA CTC/B

A991 Azambuja, Heitor Teixeira de.
Casamento de comprimento de trilhas de placas de circuito impresso utilizando otimização por colônia de formigas / Heitor Teixeira Azambuja. – 2025.
105 f.

Orientadoras: Nadia Nedjah, Luiza de Macedo Mourelle.
Dissertação (Mestrado) - Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.

1. Engenharia eletrônica - Teses. 2. Sistemas eletrônicos - Teses. 3. Circuitos impressos - Teses. 4. Sistemas de controle inteligente - Teses. I. Nedjah, Nadia. II. Mourelle, Luiza de Macedo. III. Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia. IV. Título.

CDU 681.5

Bibliotecária: Júlia Vieira – CRB7/6022

Autorizo, apenas para fins acadêmicos e científicos, a reprodução total ou parcial desta tese, desde que citada a fonte.

Assinatura

Data

Heitor Teixeira de Azambuja

Casamento de Comprimento de Trilhas de Placas de Circuito Impresso Utilizando Otimização por Colônia de Formigas

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em: 16 de Janeiro de 2025

Banca Examinadora:

Prof.ª Dr.ª Nadia Nedjah (Orientadora)
Faculdade de Engenharia, UERJ

Prof.ª Dr.ª Luiza de Macedo Mourelle (Orientadora)
Faculdade de Engenharia, UERJ

Prof. Dr. Americo Barbosa da Cunha Junior
Instituto de Matemática e Estatística, UERJ

Prof.ª Dr.ª Mariane Rembold Petraglia
Programa de Engenharia Elétrica, COPPE, UFRJ

Rio de Janeiro
2025

AGRADECIMENTOS

Agradeço primeiramente a Deus, pela vida, pelo caminho e por tudo.

Agradeço à minha esposa Ludmila Ramos Serrano de Azambuja, pela confiança, paciência e tolerância à minha ausência nas longas noites que passei dedicado a este trabalho. Eu te amo.

Agradeço à minha filha Clarice Serrano de Azambuja, por ter me ensinado o verdadeiro significado da vida e do amor paterno. Agradeço também ao meu segundo filho, cuja vinda a este mundo é aguardada com tanto carinho e ansiedade.

Agradeço ao meu pai José Carlos Mattos de Azambuja, por ter me ensinado o valor do trabalho duro, da perseverança e da família, que Deus o tenha.

Agradeço à minha mãe Simone Gedy Teixeira de Azambuja, pelos ensinamentos de amor e força, pelos conselhos e pela confiança inabalável.

Agradeço ao meu irmão Hugo Teixeira de Azambuja, pelo exemplo inatingível de dedicação e força de vontade.

Agradeço aos meus avós, pelo carinho e pelo cuidado.

Agradeço à minha sogra Luzia Gaudêncio Ramos, pelo apoio.

Agradeço aos meus tios, tias e primos, por me mostrarem que a distância não diminui os laços familiares. Em especial, ao meu primo Augusto de Azambuja, por ser meu segundo irmão.

Agradeço aos meus amigos, todos eles, pela camaradagem e pelos trechos do caminho que seguiram ao meu lado.

Agradeço às minhas orientadoras, as professoras Nadia Nedjah e Luiza de Macedo Mourelle, pela paciência e pelos valiosos ensinamentos.

Sem vocês eu não estaria aqui. Eu amo todos vocês.

Não tendo uma idéia do futuro, também não temos uma idéia de hoje, porque o hoje, para o homem de ação, não é senão um prólogo do futuro.

Fernando Pessoa

RESUMO

AZAMBUJA, Heitor T. de. *Casamento de Comprimento de Trilhas de Placas de Circuito Impresso Utilizando Otimização por Colônia de Formigas*. 2025. 107f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2025.

O avanço da engenharia eletrônica tem impulsionado a redução do tamanho dos componentes eletrônicos e o aumento das frequências de operação. Consequentemente, as tolerâncias e limitações eletromagnéticas associadas à geometria dos circuitos tornam-se cada vez mais restritivas, gerando novos desafios no projeto de sistemas eletrônicos. Ferramentas para roteamento automático de placas de circuito impresso (PCIs) e circuitos integrados são essenciais para auxiliar engenheiros e projetistas, otimizando tempo e recursos. Para enfrentar essas dificuldades, estratégias baseadas em inteligência computacional têm sido amplamente empregadas para obter soluções de roteamento viáveis de forma eficiente. Entre essas estratégias, algoritmos multiagentes destacam-se por sua relevância e versatilidade. Neste trabalho, apresentamos uma variação do algoritmo tradicional de Otimização por Colônia de Formigas (ACO), modificada para realizar o roteamento em PCIs com foco no casamento de comprimentos das trilhas do circuito impresso. A área da PCI é dividida em uma grade que representa o espaço de busca do algoritmo. O roteamento de cada trilha é conduzido por uma colônia de formigas individual, executada em um processo independente. As formigas de cada colônia percorrem o caminho entre a posição inicial e a posição final de uma trilha, buscando identificar a rota que melhor atende a uma função de otimização multiobjetivo. Essa função minimiza simultaneamente três objetivos, que são, o comprimento total das trilhas, o número de cruzamentos entre trilhas e a diferença de comprimento entre elas. Em outras palavras, as formigas buscam encontrar a rota mais curta possível que mantenha o mesmo comprimento das demais trilhas, sem que haja cruzamentos. Para avaliar esses objetivos, foram aplicadas múltiplas estratégias heurísticas. O algoritmo proposto foi implementado e testado em sete cenários fundamentais de validação. As heurísticas desenvolvidas foram analisadas e comparadas estatisticamente com base nos resultados médios de centenas de execuções. Além disso, o programa foi aplicado em três cenários provenientes de trabalhos relacionados. Os resultados obtidos demonstram que a abordagem proposta é viável e eficaz para o roteamento de trilhas em PCIs, atendendo aos requisitos de casamento de comprimento e contribuindo para a solução dos desafios de projeto.

Palavras-chave: Roteamento Automático. Placas de Circuito Impresso. Otimização por Colônia de Formigas. Algoritmos Multiagentes.

ABSTRACT

AZAMBUJA, Heitor T. de. *Length Matching of Printed Circuit Board Traces Using Ant Colony Optimization*. 2025. 107f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2025.

The advancement of electronic engineering has driven the miniaturization of electronic components and the increase in operating frequencies. Consequently, the electromagnetic tolerances and limitations associated with circuit geometry have become increasingly restrictive, creating new challenges in electronic system design. Automatic routing tools for Printed Circuit Boards (PCBs) and integrated circuits are essential to assist engineers and designers, optimizing time and resources. To address these challenges, strategies based on computational intelligence have been widely employed to obtain viable routing solutions efficiently. Among these strategies, multi-agent algorithms stand out for their relevance and versatility. In this work, we present a variation of the traditional Ant Colony Optimization (ACO) algorithm, modified to perform PCB routing with a focus on length matching of the circuit's traces. The PCB area is divided into a grid that represents the algorithm's search space. The routing of each trace is conducted by an individual ant colony, executed as an independent process. The ants of each colony traverse the path between the initial and final positions of a trace, seeking to identify the route that best satisfies a multi-objective optimization function. This function simultaneously minimizes three objectives: the total length of the traces, the number of crossings between traces, and the length difference between them. In other words, the ants aim to find the shortest possible route that maintains the same length as the other traces without crossing them. To evaluate these objectives, multiple heuristic strategies were applied. The proposed algorithm was implemented and tested in seven fundamental validation scenarios. The developed heuristics were analyzed and statistically compared based on the average results of hundreds of executions. Additionally, the program was applied to three scenarios derived from related works. The results demonstrate that the proposed approach is both viable and effective for PCB trace routing, meeting length matching requirements and contributing to solving design challenges.

Keywords: Automatic Routing. Printed Circuit Boards. Ant Colony Optimization. Multi-Agent Algorithms.

LISTA DE FIGURAS

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

ACO	*Ant Colony Optimizaiton*
ACS	*Ant Colony System*
CAD	*Computer-Aided Design*
CI	Circuito Integrado
CNC	*Computer Numerical Control*
FPGA	*Field Programmable Gate Array*
IEM	Interferência Eletromagnética
MOACO	*Multi-Objective Ant Colony Optimization*
PAMC	Pesquisa em Árvore de Monte Carlo
PCB	*Printed Circuit Board*
PCI	Placa de Circuito Impresso
PSO	*Particle Swarm Optimization*
RFD	*River Flow Dynamics*
RNP	Rede Neural Profunda
SPICE	*Simulation Program with Integrated Circuit Emphasis*
TSP	*Travelling Salesman Problem*
VLSI	*Very-Large-Scale Integration*

SUMÁRIO

SUMÁRIO

INTRODUÇÃO

PLACAS de Circuito Impresso (PCI) são estruturas onde componentes eletrônicos são fixados e conectados. Presentes em praticamente todos os equipamentos eletrônicos, as PCIs são formadas por camadas laminadas que intercalam materiais condutores e isolantes, como cobre e fibra de vidro ou fenolite. Nas camadas condutoras, são gravadas as trilhas de conexão e os planos de fixação dos componentes, formando circuitos eletrônicos que funcionam como uma rede de "fios"dispostos sobre um plano. Dependendo da complexidade do circuito, múltiplas camadas de condutores podem ser sobrepostas.

Inicialmente as camadas condutoras são planos contínuos de cobre, onde o projeto do circuito é gravado através da remoção do material nas áreas onde não deve haver conexão. O projeto é impresso na placa deixando expostas as regiões de cobre que devem ser removidas. Em seguida, a placa é imersa em uma solução corrosiva que elimina as partes indesejadas, deixando apenas as trilhas e planos do projeto impresso.

No passado, o projeto era feito à mão, porém, com o avanço da engenharia elétrica e eletrônica, o tamanho dos componentes e das trilhas foram reduzidos e as tolerâncias e limitações eletromagnéticas se tornaram mais estritas. Atualmente, programas de computadores CAD (*Computer-Aided Design*) desenvolvidos especificamente para a elaboração de circuitos eletrônicos são necessários para a criação do projeto. Estes programas permitem a elaboração de circuitos complexos, não apenas por possibilitarem o desenho de traços em pequena escala (que seria impossível a mão livre), mas também por realizarem a verificação das tolerâncias eletromagnéticas de todas as trilhas do circuito, avisando ao usuário quais as regiões problemáticas. Para circuitos complexos, essa verificação de tolerâncias sem o auxílio do programa, seria extremamente laboriosa, quiçá impossível.

Circuitos de alta frequência estão sujeitos a diversas problemas causadas pelo posicionamento dos componentes e características geométricas das trilhas na PCI. Como reflexão, interferência (*Crosstalk*), atenuação, atraso, instabilidade (*jitter*) e ruídos. Diversas técnicas de projeto são necessárias para mitigar e evitar estes problemas (ZHANG;

KROOSWYK; OU, 2015). As trilhas devem respeitar limitação estritas de distância, espessura, comprimentos máximo e mínimo e número de camadas.

Particularmente para trilhas de dados e *clock* em circuitos de alta frequência, é necessário garantir que estas tenham o mesmo comprimento para que não haja desvio de fase e atraso entre os sinais, e minimizar efeitos de capacitâncias e indutâncias parasíticas. Estes problemas ocorrem quando um sinal viaja uma distância maior que os outros, chegando ao destino em momentos diferentes e prejudicando a integridade do sinal (ANAND; SINGH; LADWAL, 2019). Devido a limitações espaciais e do posicionamento de componentes na PCI não é possível que estas trilhas sejam retas. Determina-se a trilha com o maior comprimento dentre as trilhas que devem ser combinadas, e então são acrescentadas curvas nas outras, de forma que todas tenham o mesmo comprimento. As dimensões destas curvas devem ser cuidadosamente especificadas, pois podem introduzir alguns dos problemas anteriormente citados ao invés de mitiga-los (TEXAS INSTRUMENTS, 2006).

Embora existam ferramentas capazes de realizar o roteamento automático de trilhas de PCI, o processo de combinação de comprimento é realizado de forma manual. Necessitando da avaliação e interferência do projetista em cada caso para garantir o funcionamento correto do circuito.

Neste trabalho, apresentamos uma variação do algoritmo ACO típico adaptado para realizar o roteamento de trilhas em PCIs, com foco no casamento de comprimentos entre as trilhas. A área da PCI é dividida em uma grade que representa o espaço de busca do algoritmo. Para cada trilha, as posições inicial e final são conhecidas e localizadas em uma célula da grade. O roteamento de cada trilha é realizado por uma colônia de formigas individual. As formigas de uma colônia devem viajar da posição inicial até a posição final de uma trilha para encontrar a rota que melhor atende às restrições de uma função multiobjetivo. Essa função minimiza simultaneamente três objetivos: o comprimento da trilha, o número de cruzamentos entre trilhas e a diferença de comprimento entre as trilhas. Em outras palavras, as formigas buscam encontrar a trilha mais curta possível que tenha o mesmo comprimento das demais trilhas, sem cruzar com elas. Diversas heurísticas foram propostas para atender aos objetivos. O algoritmo e as combinações de heurísticas foram testadas e comparadas em sete cenários fundamentais. Em seguida, foram aplicadas em três cenários oriundos de outros trabalhos como forma de comparação com outras técnicas.

Esta dissertação está estruturada em seis capítulos. O Capítulo 1 aborda a importância do leiaute do circuito na PCI, destacando os principais problemas e interferências que podem ocorrer em circuitos de alta frequência, suas causas e impactos. Também discute a relevância das restrições de comprimento das trilhas e do casamento de comprimentos como estratégias para mitigar problemas e interferências.

O Capítulo 2 apresenta um levantamento bibliográfico sobre técnicas de roteamento automático de PCIs e casos correlacionados, classificando os trabalhos em duas categorias. Aqueles que utilizam técnicas exatas e aqueles baseados em inteligência computacional.

O Capítulo 3 descreve o algoritmo de otimização por colônia de formigas (ACO), explicando sua aplicação clássica ao problema do caixeiro viajante e detalhando as equações fundamentais da modelagem. Também aborda melhorias como a técnica de elitismo e discute outras aplicações de roteamento que empregam o ACO, com ênfase em temas relevantes para esta dissertação, como otimização multiobjetivo, espaços de busca em grade, múltiplas colônias e integrações com programas CAD.

O Capítulo 4 apresenta a técnica desenvolvida neste trabalho, detalhando sua estrutura, inovações e adaptações específicas para o problema de roteamento em PCIs de alta frequência. O espaço de busca é detalhado e as múltiplas estratégias heurísticas desenvolvidas e suas equações são apresentadas e explicadas. O fluxograma do algoritmo é apresentado e seu funcionamento explicado, assim como os hiperparametros. Por fim são feitas observações a respeito da complexidade do algoritmo.

O Capítulo 5 apresenta os resultados experimentais obtidos com a aplicação da técnica desenvolvida, avaliando seu desempenho em diferentes cenários. Os cenários são descritos em detalhes e organizados em três categorias: irrestritos, restritos e externos. As combinações de heurísticas propostas são testadas e comparadas nos cenários irrestritos e restritos, permitindo identificar suas vantagens e limitações. Por fim, as melhores configurações identificadas são aplicadas aos cenários externos.

Por fim, o Capítulo 6 reúne as conclusões da dissertação, com uma análise comparativa e uma discussão sobre seus resultados. Este capítulo também aponta sugestões para trabalhos futuros, ampliando as possibilidades de pesquisa e aplicação.

Capítulo 1

ROTEAMENTO DE TRILHAS DE PCI

O Avanço da tecnologia eletrônica nas últimas décadas tem impulsionado a evolução dos circuitos e sistemas digitais. A miniaturização dos componentes, combinada com melhorias nos processos de fabricação, possibilitou o aumento das frequências de *clock* e a maior densidade de transistores, alcançando bilhões de unidades em um único chip. Esse aumento no poder de processamento e na complexidade dos circuitos exige um projeto de PCI cada vez mais sofisticado. No entanto, esses avanços também trazem desafios em termos de interferência eletromagnética (IEM) e integridade de sinal, uma vez que a proximidade dos transistores e a alta velocidade dos sinais podem induzir interferências e distorções, o que exige soluções para mitigar esses efeitos e garantir a confiabilidade dos sistemas.

No projeto de PCIs para circuitos de alta frequência, a integridade do sinal é crucial. À medida que as frequências de operação aumentam, fenômenos como diafonia, reflexões, instabilidade e atenuação de sinal podem comprometer a transmissão de dados e a temporização dos sinais, resultando em perda de informações e falhas no desempenho do circuito.

O comportamento eletromagnético de um circuito é influenciado por fatores de projeto, como o roteamento das trilhas, o leiaute da PCI, a qualidade das terminações e a distribuição do sinal de *clock*. Em circuitos de alta frequência, onde os sinais possuem tempos de subida e descida rápidos, até mesmo pequenas flutuações podem gerar distorções. A PCI desempenha um papel essencial na integridade do sinal, com suas características físicas como largura e comprimento das trilhas, espessura das camadas de cobre, espaçamento entre trilhas e qualidade da camada dielétrica, impactando diretamente o desempenho do circuito, (ANAND; SINGH; LADWAL, 2019).

As propriedades do meio dielétrico, como a densidade do trançado das fibras e a quantidade de resina utilizada, também afetam a propagação do sinal. A constante dielétrica e a distribuição do material isolante influenciam a velocidade de propagação e as perdas associadas às trilhas, afetando a impedância característica da PCI. Quando a impedância não é corretamente controlada, podem ocorrer reflexões e outros tipos de IEM (TEXAS INSTRUMENTS, 2014).

Outras características da PCI, como a escolha do material base e a densidade de componentes, também influenciam como os sinais são transmitidos e como o circuito responde a interferências externas. O leiaute da placa, com um roteamento cuidadoso das trilhas e a escolha adequada de vias e terminais, é fundamental para minimizar a interferência entre sinais de alta e baixa frequência, como diafonia e ruído de *clock*. Para garantir a confiabilidade do sistema, é essencial compreender e controlar esses fenômenos desde a definição do leiaute até a escolha dos componentes e materiais.

A Seção 1.1 detalha os problemas e interferências mais comuns em circuitos de alta frequência, bem como suas causas. A Seção 1.2 discute as restrições de comprimento nas trilhas. A Seção 1.3 apresenta a técnica de casamento de comprimento, essencial para o alinhamento temporal de sinais críticos. Por fim, a Seção 1.4 reúne as considerações finais sobre o capítulo apresentado.

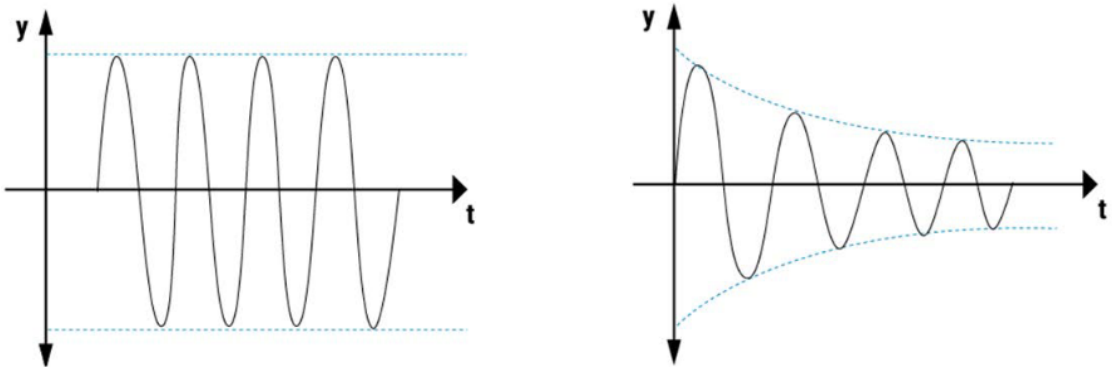
1.1 Problemas e Interferências Eletromagnéticas

Esta seção explora os principais tipos de IEM que afetam PCIs e circuitos de alta frequência, discutindo suas causas, impactos e como esses fenômenos comprometem a qualidade do sinal e o funcionamento adequado dos sistemas eletrônicos.

1.1.1 Atenuação

Atenuação do sinal é a redução da intensidade de um sinal conforme ele se propaga em um meio de transmissão, como cabos, trilhas em PCIs ou através do ar. Esse fenômeno é causado por perdas intrínsecas no meio, como resistência elétrica, absorção dielétrica, efeito pelicular em altas frequências e perdas por radiação. A atenuação pode comprometer a integridade do sinal, dificultando sua distinção em relação ao ruído, especialmente em transmissões de longa distância ou em frequências elevadas.

Em PCIs, a atenuação é influenciada por diversos fatores, como a resistência dos condutores, a qualidade do material dielétrico entre as camadas da placa e a geometria das trilhas. Para sinais de alta frequência, os efeitos do efeito pelicular (*skin effect*) tornam-se mais pronunciados, concentrando a corrente na superfície dos condutores e aumentando as perdas, (ZHANG; KROOSWYK; OU, 2015). Além disso, problemas de projeto, como trilhas excessivamente longas ou mal projetadas, podem amplificar as perdas por radiação e IEM. A atenuação excessiva pode prejudicar o desempenho do sistema, reduzindo a margem de erro e aumentando a suscetibilidade a ruídos. Por isso, é fundamental considerar as características do meio de transmissão no projeto, utilizando materiais de baixa perda, técnicas de impedância controlada e rotas de sinal otimizadas. A Figura 1 ilustra o efeito da atenuação do sinal. A Figura 1a mostra um sinal ideal, que mantém a amplitude constate ao longo do tempo, enquanto a Figura 1b exibe um sinal sofrendo atenuação ao longo do tempo.



(a) Sinal ideal. (b) Sinal atenuando ao longo do tempo.

Figura 1: Exemplo de atenuação (SIERRA CIRCUITS, 2023).

1.1.2 Atraso de Propagação

Atraso de propagação é um fenômeno natural em circuitos eletrônicos, associado ao tempo que um sinal leva para trafegar no meio em que é propagado. Particularmente em circuitos digitais, esse atraso está relacionado ao tempo de resposta dos transistores e à quantidade destes que operam em sequência, resultando na soma dos atrasos individuais. Esse tempo de resposta, ou atraso, costuma ser descrito na folha de dados dos componentes e normalmente é levado em consideração na etapa de projeto do circuito.

Em PCIs, o atraso de propagação também pode ser introduzido por características geométricas das trilhas, por roteamento inadequado ou por IEM decorrentes desse rote-

amento. No caso de circuitos de alta frequência, o atraso entre trilhas onde o sinal deve propagar no mesmo intervalo de tempo, como no caso de trilhas de dados, é tipicamente causado pela diferença de comprimento entre essas trilhas. Isso faz com que os sinais cheguem ao destino em momentos diferentes, comprometendo o funcionameto correto do circuito. A Figura 2 exemplifica o atraso de propagação.

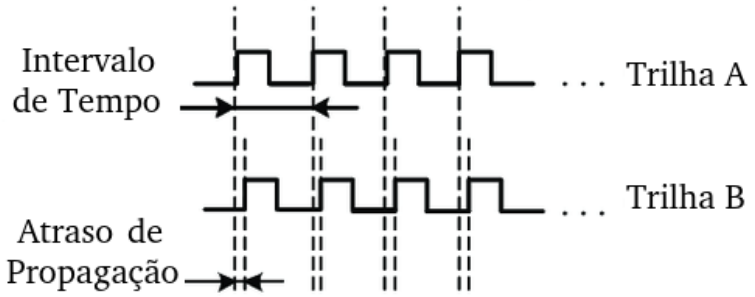


Figura 2: Exemplo de atraso de propagação (ARKANI et al., 2015).

1.1.3 Reflexão

Reflexões ocorrem quando um sinal elétrico que trafega por uma linha de transmissão (como uma trilha em uma PCI ou um cabo) encontra uma mudança de impedância, como uma descontinuidade no meio ou uma terminação inadequada. Parte da energia da onda é refletida de volta em direção à fonte, enquanto o restante continua para a carga. Esse fenômeno pode causar distorção do sinal, perdas de energia e problemas de funcionamento em circuitos digitais e analógicos. A Figura 3 exemplifica o efeito da reflexão do sinal. A linha pontilhada vermelha representa o sinal ideal, enquanto as linhas tracejada verde e contínua azul mostram, respectivamente, o sinal na origem e no destino, ambas sob o efeito das reflexões. Essas reflexões causam *overshoots* e *undershoots*, conforme ilustrado no gráfico.

Em PCIs, a reflexão pode ser causada por características físicas das trilhas, como variações na espessura, largura ou comprimento, que resultam em descontinuidades de impedância ao longo do trajeto. Além disso, o uso de vias, mudanças bruscas na geometria das trilhas, ou a transição entre diferentes camadas da PCI também pode introduzir descontinuidades que geram reflexões. Outro fator relevante é a ausência de terminação adequada nas extremidades das trilhas, o que amplifica as reflexões, especialmente em circuitos de alta frequência, (SIERRA CIRCUITS, 2023).

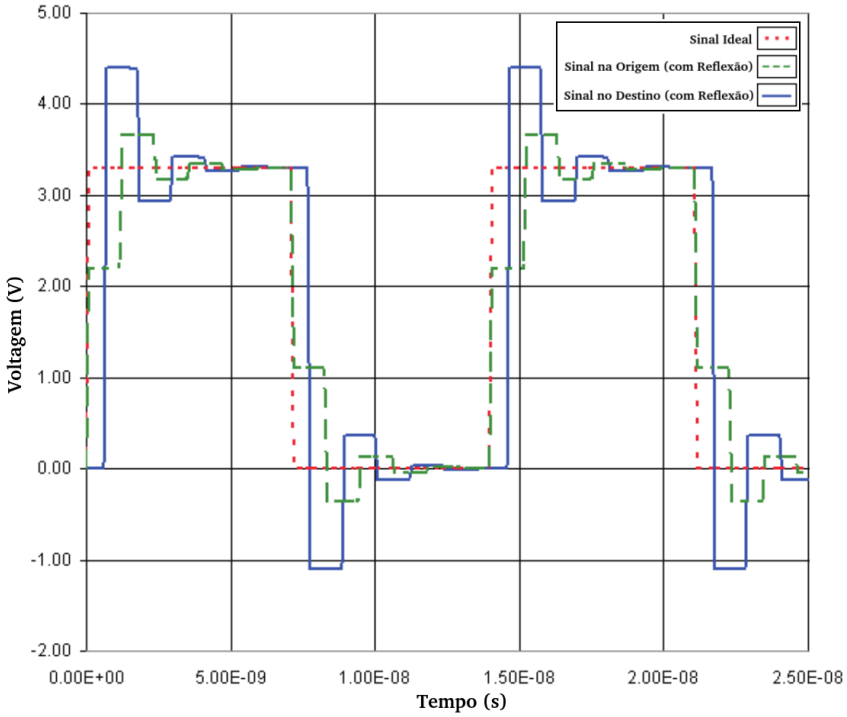


Figura 3: Exemplo do efeito da reflexão (TEXAS INSTRUMENTS, 2006).

1.1.4 Diafonia

Diafonia (*crosstalk*) é uma forma de IEM que ocorre quando o sinal presente em uma trilha de uma linha de transmissão é acoplado indesejadamente a uma trilha adjacente. Esse acoplamento gera distorções no sinal, resultando na perda de integridade da transmissão e, consequentemente, afetando o desempenho e a confiabilidade dos circuitos.

O fenômeno é causado pelo acoplamento capacitivo ou indutivo entre as trilhas. Em termos de acoplamento capacitivo, isso ocorre quando as linhas estão próximas o suficiente para formar um capacitor, permitindo que a energia elétrica se transfira de uma trilha para outra. O acoplamento indutivo resulta da criação de campos magnéticos ao redor das trilhas, que induzem correntes nas trilhas adjacentes, afetando os sinais transmitidos. Em ambos os casos, os campos eletromagnéticos gerados ao longo de uma trilha afetam as trilhas vizinhas, levando à interferência e à degradação do sinal.

A Figura 4 exemplifica o fenômeno da diafonia. O gráfico superior apresenta o sinal agressor, enquanto o gráfico inferior ilustra como o sinal do gráfico superior induz picos e vales nas mudanças de estado (retângulos vermelhos), resultando em distorções no sinal transmitido. Isso é particularmente problemático em circuitos de alta frequência, onde até mesmo pequenas interferências podem resultar em falhas no sistema.

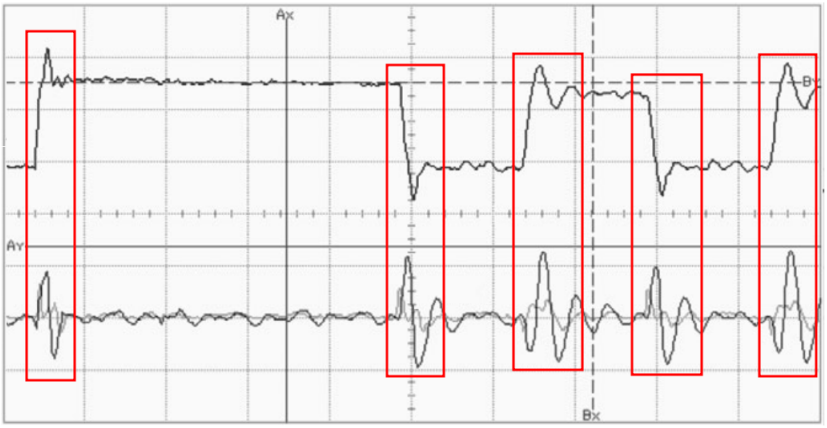


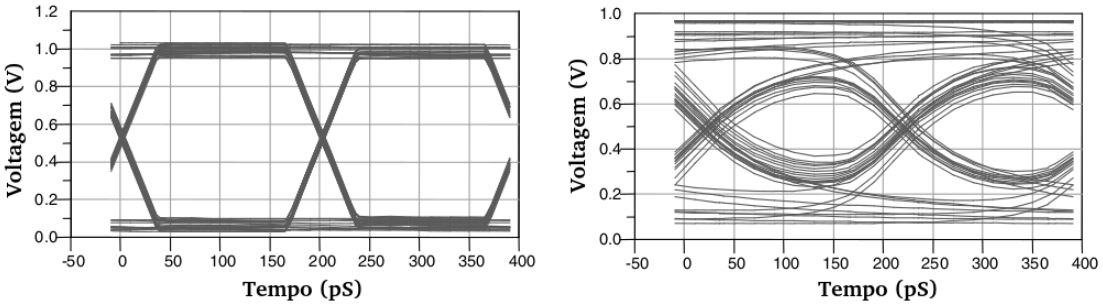
Figura 4: Exemplo de diafonia (BOGATIN, 2009).

Em circuitos de alta frequência (como aqueles que operam na faixa de GHz), a diafonia se torna um desafio ainda maior devido à alta taxa de variação dos sinais. Em frequências elevadas, os sinais têm tempos de subida e descida muito rápidos, o que amplia os efeitos da diafonia. O aumento da largura de banda dos sinais torna a integridade do sinal ainda mais sensível a interferências externas. Quando o sinal de uma trilha afeta outra, a distorção pode causar problemas como perda de dados, erro de sincronização, e atrasos nos sinais de temporização.

1.1.5 Instabilidade

Instabilidade (*jitter*) é uma IEM caracterizada por variações no tempo de propagação de um sinal. Essas flutuações temporais ocorrem quando o sinal não é transmitido de maneira consistente, resultando em atrasos imprevisíveis entre os eventos de um sinal digital, como os flancos de subida e descida. A Instabilidade pode ser causado por uma série de fatores, como ruído elétrico, interferência de sinais adjacentes, variações na qualidade do sinal de *clock* e reflexões em linhas de transmissão.

A Figura 5 apresenta dois diagramas de olho que ilustram a instabilidade. O diagrama de olho é gerado ao sobrepor múltiplos períodos de um sinal digital. Os diferentes ciclos de sinal são alinhados de modo que suas variações sejam sobrepostas, criando uma representação visual que mostra os flancos de subida e descida e as variações no nível do sinal em diferentes instantes. A Figura 5a ilustra o caso desejável de um sinal com pouca instabilidade. Em contrapartida, a Figura 5b apresenta um caso ruim, com muita instabilidade.



(a) Caso desejável, pouca instabilidade. (b) Muita instabilidade.

Figura 5: Exemplo de instabilidade (BOGATIN, 2009).

Em circuitos digitais de alta frequência, até pequenas variações no tempo de propagação podem ter impactos significativos, podendo comprometer a integridade de sinais temporizados resultando em falhas na comunicação, perda de dados e degradação da performance geral do sistema.

1.2 Restrições de Comprimento de Trilhas

As restrições de comprimento máximo e mínimo em trilhas de PCI (*Printed Circuit Boards*) desempenham um papel crucial no desempenho elétrico dos circuitos, especialmente em aplicações de alta frequência. Essas restrições são definidas para garantir a integridade do sinal e minimizar problemas como reflexões, interferências e diferenças no tempo de propagação entre sinais. Os comprimentos máximos e mínimos das trilhas estão diretamente relacionados à frequência de operação do circuito, além de outros fatores do projeto.

O comprimento máximo de uma trilha é estabelecido para evitar perdas excessivas e degradações no sinal. Trilhas longas aumentam a atenuação devido à resistência e capacitância parasitas, além de introduzirem reflexões significativas, que podem causar oscilações indesejadas. Em circuitos de alta frequência, trilhas longas também podem atuar como antenas, intensificando a emissão de ruído eletromagnético e comprometendo o desempenho geral do sistema (BOGATIN, 2009).

Por outro lado, o comprimento mínimo é essencial em aplicações onde os sinais precisam de um tempo de propagação específico. Em sistemas nos quais múltiplos sinais devem chegar simultaneamente ao destino, as restrições de comprimento mínimo são aplicadas para ajustar o tempo de propagação de trilhas mais curtas (TEXAS INSTRUMENTS,

2006). Isso é frequentemente feito por meio do uso de serpentinas para ”compensar”o comprimento. Além disso, minimizar o comprimento das trilhas ajuda a reduzir o uso da superfície de cobre, o que, por consequência, pode tornar a PCI mais compacta e eficiente.

1.3 Casamento de Comprimento Entre Trilhas

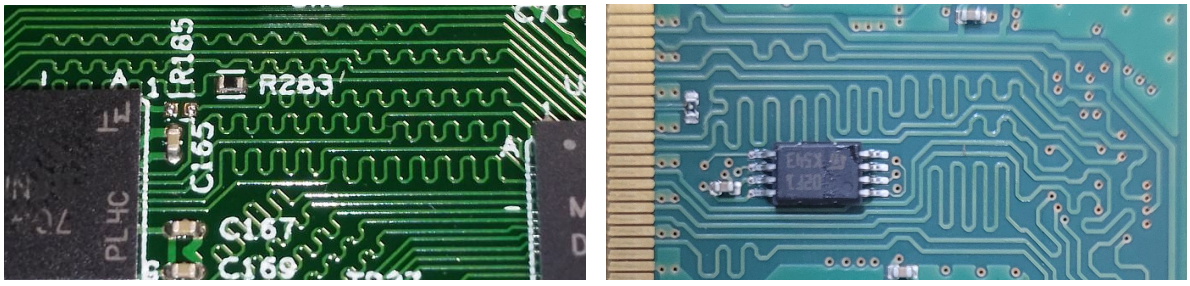
O casamento de comprimento entre trilhas em PCIs de alta frequência é uma técnica crucial para garantir a integridade do sinal e o desempenho adequado do circuito. Essa prática visa assegurar que sinais que precisam ser sincronizados, como os sinais de dados e de *clock*, cheguem ao destino ao mesmo tempo.

Em circuitos de alta frequência, onde os sinais apresentam tempos de subida e descida rápidos, até mesmo pequenas diferenças nos comprimentos das trilhas podem resultar em atrasos significativos na chegada do sinal ao destino. Isso pode causar falhas de temporização, como erros de sincronização, perda de pacotes de dados ou distorções no sinal. O casamento de comprimento busca minimizar essas discrepâncias, ajustando o comprimento das trilhas de modo que os sinais atinjam o receptor simultaneamente, respeitando as restrições de tempo impostas pelas altas frequências de operação.

O casamento de comprimento entre trilhas em PCIs de alta frequência é frequentemente realizado com curvas para estender o comprimento das trilhas de maneira controlada. Essas curvas ajudam a compensar diferenças nos tempos de propagação dos sinais. Quando o espaço é limitado, técnicas como serpentina ou laços são usadas para aumentar o comprimento sem expandir a área linearmente. Porém, é importante planejar as curvas cuidadosamente, pois elas podem afetar a impedância e introduzir indutância ou capacitância, impactando a integridade do sinal (SIERRA CIRCUITS, 2023). A técnica deve equilibrar o comprimento da trilha e minimizar os efeitos de IEM para garantir a confiabilidade do circuito. A Figura 6 apresenta exemplos reais de trilhas com casamento de comprimento em circuitos de alta frequência.

1.4 Considerações Finais

Neste capítulo, foi destacado o papel fundamental do leiaute da PCI, do roteamento cuidadoso das trilhas e da escolha adequada de materiais e componentes como fatores cruciais para garantir a qualidade do sinal e a confiabilidade dos circuitos. A compreensão



(a) Trilhas de conexão entre dois CIs.

(b) Trilhas em um módulo de memória.

(c) Trilhas em uma placa-mãe.

(d) Trilhas em uma placa-mãe.

Figura 6: Exemplos de PCIs com trilhas com casamento de comprimento.

e o controle desses aspectos permitem o desenvolvimento de PCIs eficientes e robustas, capazes de atender aos rigorosos requisitos de desempenho em circuitos de alta frequência.

Também foram abordados os principais problemas e interferências eletromagnéticas que podem surgir no processo de roteamento de trilhas em PCIs destinadas a circuitos de alta frequência. Fenômenos como diafonia, reflexões, atenuação do sinal e instabilidade foram discutidos como fatores que podem comprometer a integridade do sinal e, consequentemente, o desempenho dos sistemas eletrônicos. Além disso, foi destacada a importância de respeitar as restrições de comprimento máximo e mínimo das trilhas, que estão diretamente relacionadas à frequência de operação e têm impacto significativo na integridade e no sincronismo dos sinais. Por fim, enfatizou-se a técnica de casamento de comprimento de trilhas como uma solução indispensável para mitigar os efeitos negativos dessas interferências, garantindo que sinais críticos, como os de dados e *clock*, cheguem ao destino com o alinhamento temporal necessário para o correto funcionamento do sistema.

O trabalho desenvolvido nesta dissertação utiliza algoritmos de inteligência de enxame para realizar o roteamento automático de trilhas de PCI, otimizando três objetivos simultaneamente. Encontrar o comprimento mínimo das trilhas, realizar o casamento de comprimento entre elas e minimizar o número de cruzamentos.

Capítulo 2

TRABALHOS RELACIONADOS

ESTE capítulo apresenta trabalhos relacionados ao problema de roteamento de trilhas de PCIs. Durante o processo de roteamento automático, dependendo das características elétricas e da complexidade do circuito que está sendo resolvido, diversos cenários podem ser abordados. Dentre estes cenários, destacam-se: A conexão de componentes; O posicionamento de componentes e o dimensionamento da PCI; A distribuição de trilhas de sinal de *clock* para minimizar atrasos; O casamento de comprimento de trilhas para mitigar interferências e outros problemas eletromagnéticos.

A Seção 2.1 aborda alguns trabalhos que utilizaram técnicas determinísticas para realizar o roteamento de trilhas de circuitos. A Seção 2.2 cita trabalhos que utilizaram estratégias baseadas em inteligência computacional. A Seção 2.3 contém as considerações finais sobre o capítulo apresentado.

2.1 Técnicas Exatas

Em (KAHNG et al., 2011b), diversos algoritmos e estratégias de roteamento são abordadas para realizar o roteamento global de trilhas de circuitos eletrônicos integrados. Durante o roteamento global o espaço disponível para as trilhas é particionado em regiões de roteamento, e identifica-se quais trilhas necessitam de determinadas regiões. Neste processo, diversas restrições são levadas em consideração, como o comprimento das trilhas, atraso de propagação de sinais e interferências eletromagnéticas. No entando, as trilhas competem pela disponibilidade de regiões de roteamento, e locais de congestão são identificados. Para minimizar estes conflitos, duas estratégias distintas podem ser adotadas: o roteamento sequencial ou o roteamento simultâneo. No roteamento sequencial, uma ordem para o roteamento é determinada e as trilhas são traçadas uma por vez. Quando uma trilha

viola alguma restrição, ela é removida e seu processo reinicia. O roteamento sequencial necessita de muitas iterações para resolver o problema. Nesta estratégia, são aplicados algoritmos baseados em busca em árvores binárias e resolvedores de labirintos, como o algoritmo de Dijkstra (DIJKSTRA, 1959) e A* (HART; NILSSON; RAPHAEL, 1968). No roteamento simultâneo, todas as trilhas são traçadas em conjunto, evitando o surgimento de regiões congestionadas e violações das restrições. Nesta estratégia, são utilizados algoritmos baseados em programação linear inteira (MA; WONG, 2012). A etapa de roteamento global é considerada uma etapa "grosseira" no processo de roteamento, que é refinado na etapa seguinte denominada roteamento detalhado.

Em (KAHNG et al., 2011a) são abordadas técnicas para realizar o roteamento detalhado de circuitos eletrônicos integrados. A etapa de roteamento detalhado segue a etapa de roteamento global, levando em consideração o resultado obtido por este roteamento global. Enquanto o roteamento global define rotas gerais para a conexão entre componentes, o roteamento detalhado lida com os detalhes específicos das trilhas individuais do circuito. Nessa etapa todas as trilhas devem obrigatoriamente respeitar todas as restrições do problema. Esta etapa leva em consideração os conflitos de rota, as características do processo de fabricação e a geometria do encapsulamento, além de todas as restrições eletromagnéticas das etapas anteriores. Nessa etapa as trilhas são distribuidas em camadas distintas. O problema é subdividido em problemas menores que então são resolviso independentemente, dessa forma o tempo de execução e o cusco computacional escalam linearmente com o tamanho do circuito. São utilizados algoritmos de minimização e busca em grafos, e algoritmos que reproduzem padrões geométricos para resolver conflitos de rota.

Em (KAHNG et al., 2011c) técnicas de roteamento especializado são abordadas. Roteamento especializado aplica-se ao roteamento de circuitos analógicos e trilhas de PCIs de uma ou múltiplas camadas. Também são incluídas nessa categoria trilhas que necessitam de tratamento específico, como trilhas de sinal de *clock* e trilhas de sinais diferenciais (comunicação serial). Quando os roteamentos global e detalhado não são realizados, o roteamento de área é utilizado para conectar componentes. O roteamento de área minimiza o comprimento das vias, o número de vias, o número de camadas, o atraso de propagação de sinais e evita o surgimento de efeitos capacitivos parasíticos entre vias vizinhas. O roteamento de área utiliza apenas trilhas verticais e horizontais para resolver

o problema. Os algoritmos desta estratégia utilizam o cálculo da distância Euclidiana e da distância de Manhattan para realizar o roteamento. Além disso, também podem realizar o roteamento ordenado das vias para minimizar o comprimento. Na categoria de roteamento especializado, estão inclusas técnicas denominadas roteamento *Non-Manhattan*, também conhecidas como roteamento octilínio. Estas técnicas permitem movimentos diagonais para minimizar o comprimento das trilhas. Os algoritmos de roteamento octilíneo utilizam grafos e técnicas de busca em labirintos para realizar o roteamento.

Em (LEE, 1961), é introduzido um algoritmo de busca para conexão de caminhos. Este algoritmo sempre consegue encontrar o menor caminho entre dois pontos, se for possível conectar estes pontos. Embora este algoritmo não tenha sido desenvolvido especificamente para o roteamento de trilhas de PCIs, um dos cenários de teste apresentado é a conexão de componentes de um circuito. Este algoritmo tem a desvantagem de ter a complexidade $\mathcal{O}(MN)$ para um espaço de busca de tamanho $M \times N$. Portanto, os requisitos de memória e o tempo de convergência do algoritmo aumentam com a extensão do espaço de busca.

Em (MALAVASI; SANGIOVANNI-VINCENTELLI, 1993), é apresentada uma técnica desenvolvida para o roteamento em área de circuitos analógicos baseada no algoritmo resolvedor de labirintos nomeado A* (HART; NILSSON; RAPHAEL, 1968). O roteamento em área consiste em gerar o projeto do circuito que será utilizado no processo de fabricação de um circuito integrado. O processo de roteamento é controlado por um conjunto de pesos definidos pelo usuário que afetam um função de custo que minimiza fenômenos parasíticos eletromagnéticos. Os pesos podem ser definidos automaticamente a partir da especificação de performance elétrica do circuito. O algoritmo realiza roteamento simétrico em casos de circuitos com arquiteturas diferenciais. Além disso, também é capaz de criar planos de aterramento para limitar a ocorrência de diafonia (*crosstalk*) entre trilhas. O algoritmo foi utilizado para criar o projeto do circuito de amplificadores operacionais de baixa complexidade.

Em (CONG A. B. KAHNG; TSAO, 1998), expande-se a estratégia de roteamento com árvores binárias para o caso de atraso mínimo em trilhas de sinal de *clock* através da introdução do conceito de região de fusão. A região de fusão é o local onde as diferentes trilhas que levam o sinal de *clock* aos componentes se unem. O algoritmo encontra os melhores locais para a região de fusão, balanceando a compensação entre as restrições de compri-

mento máximo da trilha e atraso mínimo do sinal de *clock*. Os experimentos realizados demonstram que os resultados encontrados pelo algoritmo balanceiam satisfatoriamente as restrições do problema.

No artigo (CHU; WONG, 2008), é proposto um algoritmo baseado em árvores binárias para minimizar o comprimento das trilhas no roteamento de PCIs que equilibra tempo de execução e precisão. O algoritmo utiliza tabelas pré-calculadas para acelerar o processo de busca em árvores binárias. Como essa estratégia tem o desempenho melhor quando aplicada a circuitos com poucas trilhas, o algoritmo particiona problemas com muitas trilhas em problemas menores até que as tabelas possam ser utilizadas. O usuário pode configurar o equlíbrio entre tempo de execução e precisão do resultado. O algoritmo foi testado em dezoito problemas de teste de circuitos industriais. Os resultados foram comparados com os resultados obtidos por outras técnicas de roteamento baseadas em árvores binárias. O algoritmo proposto solucionou os problemas mais rapidamente em relação as técnicas de referência, sendo cinco ou sessenta vezes mais rápido dependendo da técnica com que foi comparado.

Em (LEE; MARKOV, 2010), é introduzido um novo algoritmo para otimizar o roteamento de trilhas de se sinal de *clock* em circuitos integrados. O algoritmo busca o equilíbrio entre desvio de *clock*, atraso, potência e tolerância a variações do circuito. O circuito é avaliado através da realização de análises com o simulador de circuitos SPICE (*Simulation Program with Integrated Circuit Emphasis*) durante o processo iterativo. Uma possível solução é gerada e em seguida avaliada pelo simulador, a função de custo é ajustada e o processo se repete até a obteção de um resultado que respeita e equilibra as restrições do problema. O algoritmo é avaliado e validado em casos de teste disponibilizados pela empresa IBM para o processo de fabricação de circuitos integrados de $45nm$.

Em (OZDAL, 2005), é proposto um algoritmo de alocação de recursos baseado em relaxação Lagrangiana (EVERETT, 1963). As restrições utilizadas são os comprimentos máximo e mínimo da trilha. O algoritmo de alocação de recursos é baseado no algoritmo de congestão negociada, inicialmente proposto para o roteamento de recursos em FPGAs (*Field Programmable Gate Array*) (BETZ; ROSE, 1996). Este algoritmo detecta regiões congestionadas, e faz com que, quando possível, as trilhas sigam por caminhos alternativos menos congestionados. Por fim, a trilha que passa pela região congestionada é a que mais necessita da região para atender a função objetivo. No fim do processo iterativo, a

diferença de comprimento entre as trilhas é pequena. O algoritmo é aplicado em diversos cenários, como placas de duas camadas, circuitos de alta frequência e componentes com matriz de pinos, demonstrando sua eficácia e as vantagens de utilizar o comprimento mínimo como restrição na função objetivo.

Em (XU et al., 2005), é utilizada uma estratégia de roteamento ambíguo para minimizar falhas no roteamento. A estratégia consiste em avaliar multiplas soluções para uma mesma trilha simultaneamente durante o processo iterativo. Dependendo do valor da função de custo, trilhas ambiguas podem ser adicionadas ou removidas. Além de avaliar o roteamento ambíguo, a função de custo também minimiza o comprimento das trilhas. A estratégia é formulada como um resolvedor de labirinto, utilizando um algoritmo baseado em relaxação Lagrangiana para obter o resultado. A técnica é testada em três problemas sintéticos de roteamento de trilhas de uma PCI de seis camadas. Os resultados obtidos são comparados com os resultados de um resolvedor de labirintos convencional que encontra a menor distância entre dois pontos. A estratégia proposta é melhor em todos os casos e demonstra que a insersão de trilhas ambíguas durante o processo de roteamento aumenta o sucesso de roteamento de trilhas em até 200%.

Em (MA; WONG, 2012), é introduzido um algoritmo baseado em programação linear inteira para resolver o problema de rota de fuga aplicado ao roteamento de trilhas de PCI. O problema de rota de fuga consiste em: dada uma região retangular R e um conjunto S de retângulos dentro desta região, encontrar para cada retângulo a direção para fora da região R, de forma que a trajetória dentro da região seja mínima e que não haja interseção entre os caminhos. Esse problema caracteriza o roteamento de componentes com matriz de pinos em sua superfície inferior ou de conexão de barramentos. O algoritmo proposto seleciona uma direção cartesiana para “fuga“ de cada pino e minimiza iterativamente a quantidade de escolhas de uma mesma direção e o caminho percorrido na região retangular. O método foi testado em um conjunto de PCIs industriais para o roteamento de barramentos e foi capaz de encontrar a solução ótima.

Em (HU; ROY; MARKOV, 2010), é introduzida uma técnica com três inovações para minimizar o comprimento de trilhas no roteamento de circuitos integrados. A primeira consiste na representação de trilhas sem bifurcações. Essa representação simplifica o roteamento de trilhas com diversos pontos de destino, como trilhas de sinal de *clock*. A segunda inovação é uma função de penalização trigonométrica. Essa função faz com que a

penalização de uma solução que viola as restrições do problemas aumente rapidamente com o número de iterações, fazendo com que o roteador consiga sair do mínimos locais em busca do mínimo global. A terceira inovação é o ajuste dinâmico dos pesos dos parâmetros da função de custo. O ajuste é feito durante o processo iterativo para minimizar a quantidade de trilhas que violam as restrições do problema.

Em (LEI; MAK, 2016), é proposto um algoritmo também baseado em programação linear inteira para resolver o problema de rota de fuga. Porém neste caso, levando em consideração não apenas trilhas individuais, mas também pares diferenciais, que devem ter o mesmo comprimento. Esse novo algoritmo obtém um comprimento de trilhas menor, e reduz o número de camadas de cobre na PCI necessárias para resolver determinados problemas.

Em (WANG; SHI; CHENG, 2006), é proposto um algoritmo de triangulação para minimizar o número de camadas necessárias para resolver o problema de rota de fuga no roteamento de componenetes com matriz de pinos. A técnica é focada na aplicação em componentes com muitos pinos, com matrizes de dimensões de $35x35$ e $40x40$. Nesses casos não é possível encontrar a rota de fuga em apenas uma camada da PCI. Os pinos são agrupados em padrões triangulares. Os padrões triangulares são gerados em ordem reversa, da última para a primeira camada. A sequência de pinos triangular maximiza a soma de pinos de fuga nas camadas acumuladas e, assim, minimiza o número de camadas de roteamento de fuga. Um caso de teste é apresentado para ilustrar a abordagem.

Em (YAN; WONG, 2009), é proposto um algoritmo para resolver o problema de rota de fuga no roteamento de componentes com matriz de pinos. O algotirmo aprimora as técnicas anteriores permitindo que as trilhas realizem movimentos diagonais em 45 graus. Essa funcionalidade permite o roteamento em casos onde apenas movimentos ortoganais não são suficientes, além de reduzir a ocorrência de regiões congestionadas. O algoritmo é avaliado em oito casos de teste e tem seu desempenho comparado com a versão do algoritmo que só é capaz de realizar os movimentos ortogonais convencionais. O algoritmo proposto demanda um tempo de execução maior, porém foi capaz de realizar o roteamento em todos os casos, enquanto que a versão ortogonal só foi capaz de realizar o roteamento sem violar as restrições do problema em dois casos.

Em (LUO et al., 2010), é proposto um algoritmo para resolver o problema de rota de fuga para realizar o roteamento entre componentes com matriz de pinos em uma PCI.

O algoritmo realiza o roteamento de todas as trilhas simultaneamente e decide a ordem de roteamento dinâmicamente a partir do surgimento de regiões congestionadas. Durante o processo iterativo, o algoritmo detecta as regiões congestionadas e realiza uma etapa de retrocesso para redirecionar as trilhas que passam por estas regiões. O algoritmo foi testado em catorze casos de testes de circuitos industriais e teve seu desempenho comparado com uma ferramenta comercial contemporânea. A técnica introduzida foi capaz de realizar o roteamento em todos os casos e a ferramenta comercial só foi capaz de realizar o roteamento na metado dos casos.

Em (KUMTONG; DANKLANG; SRIBORRIRUX, 2015), é introduzida um técnica denominada Sequencia Definida (*Set Sequence*), para o roteamento de trilhas entre pinos de circuitos integrados (CIs). Essa técnica realiza uma etapa anterior ao roteamento das trilhas. Nessa etapa é estabelecida a ordem em que as trilhas devem ser traçadas para evitar que ocorra o cruzamento das trilhas na etapa de roteamento. As conexões que devem ser realizadas são qualificadas através do cálculo do produto Cartesiano dos pinos que devem ser conectados entre os CIs. A técnica é capaz de identificar as conexões que não são possíveis de serem realizadas sem que haja cruzamento entre as trilhas. É sugerido que a informação sobre essas conexões possa ser utilizada em etapas posteriores para traça-las em camadas distintas da PCI.

2.2 Técnicas baseadas em Inteligência Computacional

Em (ZHANG et al., 2020), é proposto um algoritmo hierárquico baseado em pesquisa de árvore de Monte Carlo (PAMC) para o roteamento de trilhas de PCI. A PAMC é um algoritmo de busca que utiliza métodos de simulação estatística (Monte Carlo) para encontrar soluções aproximadas para problemas com grande espaço de estados e incerteza. Isto a partir da obtenção de amostras aleatórias no espaço de busca e da construção de uma árvore de busca com os resultados (GELLY; SILVER, 2011). Neste trabalho, a PCI é representada como uma grade, e o problema de roteamento é tratado como uma busca da rota mais curta entre os pinos de origem e destino. A PAMC hierárquica consiste em duas PAMCs. A primeira encontra uma rota intermediária e um objetivo secundário que consiste em uma célula na grade. A segunda é responsável de encontrar duas rotas ótimas: uma do pino de origem ao objetivo secundário e a outra do objetivo secundário ao pino de destino. O objetivo secundário consiste em uma posição intermediária entre a origem e o

destino. Em seguida as duas rotas são concatenadas e obtém-se a rota final otimizada. A vantagem de utilizar esta técnica é que a segunda PAMC é capaz de realizar a busca de forma mais eficiente a partir do objetivo secundário gerado pela primeira PAMC e por fim obter uma rota melhor. Os autores afirmam que a PAMC hierárquica é capaz de obter a rota ótima com maior eficiência computacional quando comparada com outros métodos existentes.

Em (HE et al., 2022), é proposta uma abordagem para o roteamento de circuitos que envolve PAMC e uma rede neural profundas (RNP) que pode ser reconfigurada para quase qualquer restrição ou objetivo sem alterações no algoritmo. A rede realiza uma etapa de simulação no processo de busca da PAMC. Essa estratégia melhora a eficiência do processo em cenários de alta complexidade com o espaço de busca extenso. Para cada trilha que deve ser traçada, a rede neural gera um conjunto de movimentos possíveis a partir da origem, e a probabilidade de um movimento levar ao local de destino. Por sua vez, a PAMC realiza a busca nos movimentos gerados pela rede neural e encontra os possíveis caminhos que vão da origem ao destino. Esses caminhos são então recompensados ou penalizados pela PAMC. O valor da função de recompensa cresce com o número de trilhas que podem ser traçadas. Outras restrições adotadas para o problema também afetam este valor, como o comprimentos mínimo e máximo das trilhas e a diferença entre os comprimentos de todas as trilhas traçadas. A técnica é testada em circuitos gerados aleatoriamente e em projetos de *hardware* de código aberto. Seu desempenho é comparada com o algoritmo A* (HART; NILSSON; RAPHAEL, 1968), obtendo resultados melhores em um terço dos casos testados.

Em (AYOB et al., 2010), um algoritmo de otimização por enxame de partículas (*Particle Swarm Optimization* - PSO) é utilizado para inserir trilhas de *buffer* no processo de roteamento. A trilha de *buffer* é utilizada para minimizar o atraso. O espaço de busca representa a superfície da PCI e as partículas representam possíveis caminhos no roteamento. A função objetivo é o atraso do sinal da conexão. A rota das trilhas e as trilhas de *buffer* são obtidas simultaneamente durante o processo de otimização do PSO.

Em (YUSOF et al., 2011), a técnica de utilização do PSO para inserção de trilhas de buffer é aprimorada. É introduzida uma abordagem que utiliza dois PSOs binários. O PSO binário tem o espaço de busca discreto (KENNEDY; EBERHART, 1997). O primeiro PSO realiza o roteamento das trilhas minimizando o comprimento. O segundo determina

os melhores locais para inserção de trilhas de *buffer*. A função de aptidão do segundo PSO é formulada a partir do cálculo de atraso de circuitos RLC. O desempenho do algoritmo é comparado com a versão apresentada em (AYOB et al., 2010). A técnica proposta é capaz de obter um atraso menor mas necessita de um número maior de iterações para convergir.

Em (NATH; GHOSH; SARKAR, 2015), é proposta uma técnica que utiliza um PSO discreto com características de algoritmos genéticos para a minimização de comprimento de trilhas. Na técnica denominada DPSO-MU, uma etapa de mutação, inspirada em algoritmos genéticos, é acrescentada no processo iterativo do PSO. Nesta etapa, algumas partículas são escolhidas aleatoriamente e tem sua posição alterada para uma posição com maior valor na função de aptidão. Essa nova posição é construída com partes dos vetores de posição das partículas com maior aptidão nas gerações anteriores. O algoritmo proposto tem a taxa de convergência e estabilidade aprimoradas em relação ao PSO discreto sem a etapa de mutação.

Em (BERLIER, 2011), é proposta uma estratégia utilizando algoritmos genéticos em paralelo para posicionar componentes na placa e determinar a ordem de roteamento. A ordem de roteamento é otimizada para minimizar o comprimento da trilha. O paralelismo é realizado através de recursos de computação em nuvem, onde uma máquina de 64 núcleos é utilizada para executar diversas populações do algoritmo genético simultaneamente. A técnica é aplicada com sucesso no projeto de circuitos reais. O tempo de execução da estratégia paralela é 50 vezes mais rápido quando comparado com a execução do algoritmo genético de forma sequencial.

Em (SINHA; BAISHNAB, 2017), é proposto um modelo híbrido que combina ACO com a dinâmica de formação de rios (*River Formation Dynamics*, RFD) para enfrentar os desafios do roteamento em dispositivos VLSI (*Very-Large-Scale Integration*). A técnica de RFD é inspirada no processo natural de erosão e depósito que forma cursos de água, modelando o comportamento de agentes artificiais que alteram o ambiente ao longo de suas rotas, criando canais preferenciais. Os resultados obtidos demonstraram que o modelo proposto alcançou melhor desempenho em comparação com as técnicas tradicionais utilizadas, apresentando soluções de maior qualidade. Contudo, isso foi alcançado ao custo de maior demanda por memória e poder de processamento.

Em (BADRIYAH; SETYORINI; YULIAWAN, 2016), um algoritmo genético é utilizado para determinar o posicionamento dos componentes na PCI e o algoritmo proposto em

(LEE, 1961) é aplicado na realização do roteamento das trilhas entre os componentes. A função de aptidão do algoritmo genético minimiza o tamanho da PCI, posiciona os componente o mais próximo possível sem que haja sobreposição e distribui na placa componentes com alta emissão de calor. A técnica é aplicada em circuitos reais e tem seus resultados comparados com resultados gerados por ferramentas comerciais. O algoritmo de roteamento usado, embora lento e com alta demanda de memória, obtém resultados satisfatórios para circuitos de baixa complexidade.

No trabalho (PANDIARAJ; SIVAKUMAR; SRIDEVI, 2017), é proposto um algoritmo genético baseado em Evolução Diferencial (STORN; PRICE, 1997) para minimizar o comprimento das trilhas no circuito interno de um circuito integrado. O algoritmo leva em consideração o comprimento de trilhas em circuitos de várias camadas. Durante o processo iterativo as camadas do circuito são particionadas e sobrepostas de forma que o circuito continue funcional e o comprimento das trilhas é minimizado. O comprimento das trilhas e o número de camadas são utilizado na função de aptidão do algoritmo genético.

No trabalho (LIAO et al., 2019), uma abordagem baseada em redes neurais com aprendizado profundo por reforço é utilizada no roteamento de trilhas de PCI. O modelo gera uma política ótima para o roteamento baseada na variedade de problemas que foram vistos durante o processo de treinamento da rede neural. O algoritmo é treinado para minimizar o comprimento das trilhas e tem seu desempenho comparado com o algoritmo A* (HART; NILSSON; RAPHAEL, 1968). Para validar o algoritmo um gerador de problemas de roteamento com restrições diversas é desenvolvido e utilizado. A estratégia proposta supera o algoritmo A* na maioria dos casos, encontrando trilhas que melhor satisfazem as restrições do problema.

Em (KIM et al., 2020), um roteador é gerado com uma rede neural com aprendizado por reforço considerando a integridade dos sinais na PCI. O modelo treinado leva em consideração regiões de diafonia, quantidade de trilhas e o comprimento das trilhas, priorizando a minimização deste comprimento. Quando comparado com os algoritmos de Lee (LEE, 1961) e A* (HART; NILSSON; RAPHAEL, 1968) em um caso de teste específico, o modelo proposto obtém resultados melhores em termos da minimização do comprimento das trilhas e da ocorrência de diafonia. Além disso, também é capaz de realizar o roteamento de PCIs de múltiplas camadas. Estas são tarefas que os outros dois algoritmos, em suas implementações convencionais, não são capazes de realizar.

2.3 Considerações Finais

Neste capítulo foram apresentados, de maneira breve, trabalhos relacionados ao roteamento de trilhas de PCIs. Os trabalhos apresentados foram separados em duas categorias: Técnicas Exatas e Técnicas baseadas em otimização computacional. As técnicas exatas utilizam estratégias determinísticas com iterações sequenciais. Algumas destas técnicas são utilizadas como pontos de referência para avaliar o desempenho de novas estratégias. Em particular os algoritmos introduzidos em (LEE, 1961) e (HART; NILSSON; RAPHAEL, 1968) são utilizados dessa forma. As técnicas baseadas em inteligência computacional utilizam estratégias heurísticas, que são capazes de realizar o roteamento em casos mais complexos. Em contrapartida, essas estratégias não garantem a solução ótima, mas sim uma solução que equilibra as restrições do problema. Para as técnicas nas duas catagorias, a capacidade de obter de uma solução satisfatória depende da complexidade do circuito e das restriçoes adotadas para o problema. O tempo de execução e o custo computacional são elevados em casos muito complexos, inviabilizando o roteamento.

Capítulo 3

OTIMIZAÇÃO POR COLÔNIA DE FORMIGAS

ESTE capítulo apresenta o algoritmo de Otimização por Colônia de Formigas (*Ant Colony Optimization*, ACO) em sua forma canônica, uma aplicação clássica do algoritmo e variações da meta-heurística relevantes para o problema de roteamento de trilhas de PCI.

A Seção 3.1 introduz o problema típico de aplicação do ACO. A Seção 3.2 apresenta o ACO aplicado ao problema do caixeiro-viajante, suas equações e o pseudo-código do algoritmo. A Seção 3.3 descreve variações na forma canônica do algoritmo para enfatizar boas soluções e exploração do espaço de busca. A Seção 3.4 apresenta trabalhos com aplicações do ACO que foram considerados relevantes. Por fim, a seção 3.5 contém as considerações finais sobre o conteúdo deste capítulo.

3.1 Problema do Caixeiro-viajante

O problema do caixeiro-viajante (*Travelling Salesman Problem*, TSP) é um desafio típico de otimização combinatória que foi estudado extensivamente. O primeiro algoritmo ACO, assim como suas variações subsequentes, foram inicialmente testados e validados no TSP (DORINGO; STÜTZLE, 2004). O TSP é um problema NP-difícil, o que significa que não existe um algoritmo polinomial conhecido que possa resolver o problema em tempo razoável.

O TSP pode ser formulado da seguinte maneira: Dada uma lista de cidades e as distâncias entre cada par de cidades, o objetivo é encontrar o caminho mais curto que visita cada cidade exatamente uma vez e retorna à cidade de origem. Em outras palavras, deve-se determinar a rota mais eficiente que minimize a distância total percorrida. Formalmente,

se tivermos n cidades, existem $n!$ (fatorial de n) possíveis rotas diferentes para o caixeiro-viajante. O TSP pode ser representado por um grafo ponderado $G = (N, A)$, onde N é o conjunto de nós do grafo, representando as cidades, e A o conjunto de arcos que conecta os nós. Cada arco $(i, j) \in A$ é atribuído um valor de comprimento d_{ij}, que é a distância entre as cidades i e j, onde $i, j \in N$. O objetivo no TSP consiste em encontrar o circuito Hamiltoniano de comprimento mínimo no grafo. Circuito Hamiltoniano é um caminho fechado que visita cada nó do grafo exatamente uma vez até retornar a origem. Uma solução ótima para o TSP consiste na permutação π dos nós de índices $1, 2, ..., n$, de forma que o comprimento seja mínimo. O comprimento do caminho é definido por:

$$f(\pi) = \left(\sum_{i=1}^{n-1} d_{\pi(i)\pi(i+1)} \right) + d_{\pi(n)\pi(1)} \tag{1}$$

3.2 Algoritmo de Otimização por Colônia de Formigas

Otimização por Colônia de Formigas é um algoritmo inspirado no comportamento das formigas na busca por alimentos. O algoritmo tem como objetivo encontrar a solução ótima para um problema, usando uma abordagem de busca baseada em um conjunto de agentes (formigas). O algoritmo começa criando uma população de formigas, que exploram o espaço de busca para encontrar a melhor solução. Cada formiga tem uma posição e usa uma heurística e umo rastro de feromônio para guiar sua busca. A heurística é uma função que mede quão boa é uma solução em relação a outras soluções. O rastro de feromônio é uma substância química que as formigas depositam ao caminhar, que consiste em uma forma de comunicação entre as formigas, indicando a qualidade da solução encontrada por cada formiga. Em cada iteração do algoritmo, as formigas constroem soluções incrementais para o problema, usando a heurística e o rastro de feromônio como guias. As formigas depositam feromônio nas posições que percorrem e, assim, as melhores soluções são reforçadas ao longo do tempo. As formigas são influenciadas pela quantidade de feromônio presente em cada posição do espaço de busca e pelo comprimento da melhor solução encontrada até o momento. Dessa forma, tendem a explorar as regiões com maior concentração de feromônio e a se mover na direção da melhor solução conhecida.

A probabilidade de uma formiga k na posição i escolher se deslocar para a posição j é dada pela equação:

$$p_{ij}^{k} = \frac{(\tau_{ij})^{\alpha} \cdot (\eta_{ij})^{\beta}}{\sum_{l \in N_i^k} (\tau_{il})^{\alpha} \cdot (\eta_{il})^{\beta}}, \quad j \in \mathcal{N}_i^k, \tag{2}$$

onde: τ_{ij} é o feromônio associado ao caminho (i, j). α e β são parâmetros para determinar a influência do feromônio e da informação heurística. $\mathcal{N}_i^k$ é a vizinhança de possíveis movimentos da formiga k na posição i. $\eta_{i,j}$ é o valor heurístico que representa a atratividade da formiga visitar a posição j depois de visitar a posição i.

A equação de atualização do rastro de feromônio do ACO é dada por:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \sum_{k=1}^{m} \Delta\tau_{ij}^k, \tag{3}$$

τ_{ij} é a quantidade de feromônio depositada na posição (i, j). ρ é a taxa de evaporação do feromônio, que controla a quantidade de feromônio que evapora do rastro de feromônio em cada iteração. Geralmente, assume um valor entre 0 e 1. ΔT_{ij} é a quantidade de feromônio depositada na aresta (i, j), que depende da qualidade da solução encontrada pela formiga k na iteração. m é o número de formigas na colônia.

A equação da quantidade de feromônio depositado $\Delta\tau_{ij}^t$ pode ser calculada como:

$$\Delta\tau_{ij}^k = \frac{Q}{L_k}, \tag{4}$$

Q é a constante de depósito de feromônio, que controla a quantidade de feromônio depositada no rastro de feromônio por soluções de boa qualidade. L_k é o comprimento da solução encontrada pela formiga k.

Algoritmo 1 ACO canônico

```
Inicializa o feromônio τ_ij em todas as células
Configura os parâmetros: Q, α, β, ρ, t_max
para t = 1 a t_max faça
  para cada formiga k faça
    Inicializa formiga k na origem
    enquanto a formiga k não chegar na posição destino faça
      para cada célula vizinha j faça
        Calcula a probabilidade de transição p_ij^k utilizando a Equação (2)
      fim para
      Seleciona próxima posição j baseado na probabilidade p_ij^k
      Move a formiga k para a célula j
    fim enquanto
    Calcula o comprimento da rota L_k da formiga k
  fim para
  Atualiza o nível de feromônio em todas as células com a Equação (3)
fim para
```

O Algoritmo 1 apresenta o pseudocódigo fundamental do ACO, detalhando a inicialização, a construção incremental de soluções pelas formigas e a atualização do rastro

de feromônio. Sua estrutura iterativa permite melhorias contínuas até alcançar o número máximo de iterações ou outro critério de parada, equilibrando de forma eficiente a exploração e a intensificação para aprimorar as soluções progressivamente.

Na etapa de inicialização, são definidos os níveis de feromônio e os parâmetros principais, como α, β, ρ, Q e $t_{\max}$. Durante cada iteração, as formigas constroem soluções incrementais ao avaliar probabilisticamente suas opções, conforme a Equação (2). Ao final de cada ciclo, o feromônio é atualizado utilizando a Equação (3), incorporando os efeitos de evaporação (ρ) e reforço positivo, proporcional à qualidade das soluções ($\Delta\tau_{ij}^{k}$).

3.3 Variantes do ACO

Esta seção apresenta evoluções diretas do ACO. Variações que introduzem melhoramentos na deposição de feromônio e na exploração do espaço de busca. O elitismo reforça as melhores soluções, assegurando sua preservação ao longo das iterações. O ranqueamento prioriza as soluções mais bem classificadas, atribuindo maior peso a elas na atualização do feromônio. A variante *max-min* limita a quantidade de feromônio, prevenindo convergência prematura e promovendo um equilíbrio mais eficiente entre exploração e intensificação. O sistema de colônia de formigas realiza uma atualização de feromônio mais eficiente e uma escolha de caminhos mais agressiva, favorecendo a exploração e a convergência mais rápida para boas soluções.

3.3.1 Elitismo

O primeiro melhoramento no algoritmo ACO original consiste na chamada estratégia elitista, ou simplesmente elitismo. A estratégia consiste em reforçar as posições pertencentes ao melhor caminho encontrado desde o início da execução do algoritmo. Este caminho é denotado como T^{bs} (*best-so-far*). Esse reforço pode ser considerado como um feromônio depositado por uma formiga adicional nomeada *bs* na etapa de atualização do rastro de feromônio. Com este termo adicional, a equação de atualização do rastro de feromônio fica da seguinte forma,

$$\tau_{ij} = (1-\rho)\tau_{ij} + \left(\sum_{k=1}^{m} \Delta\tau_{ij}^{k}\right) + e\tau_{ij}^{bs}, \tag{5}$$

e define a importância dada ao melhor caminho encontrado até o momento.

Resultados computacionais sugerem que a estratégia elitista com um parâmetro e apropriado permite que melhores caminhos sejam encontrados pelo ACO de forma mais rápida (DORIGO; GAMBARDELLA, 2000).

3.3.2 Ranqueamento

Outro melhoramento do ACO consiste no ranqueamento das formigas. Nessa variante cada formiga deposita uma quantidade de feromônio que depende da sua classificação. A melhor formiga até o momento (*best-so-far*) deposita a maior quantidade de feromônio a cada iteração.

Antes da atualização do rastro de feromônio, as formigas são ordenadas de forma crescente pelo tamanho do caminho construído. Ou seja, quanto maior o caminho, maior será o ranque da formiga. A quantidade de feromônio depositado é regulado de acordo com o ranque r da formiga e uma constante ω. Em cada iteração, apenas as $\omega-1$ formigas mais bem classificadas e a melhor formiga até o momento podem depositar feromônio. A melhor formiga até o momento não necessariamente faz parte das formigas desta iteração. Esta formiga deposita a maior quantidade de feromônio, multiplicado pelo peso ω. As outras formigas depositam uma quantidade de feromônio multiplicado pelo maior valor entre 0 e $\omega - r$. Dessa forma, a equação de atualização de feromônio é dada por:

$$\tau_{ij} = (1-\rho)\tau_{ij} + \left(\sum_{r=1}^{\omega-1}(\omega - r)\Delta\tau_{ij}^{r}\right) + \omega\tau_{ij}^{bs}. \tag{6}$$

Resultados experimentais sugerem que a estratégia de ranqueamento tem a performance ligeiramente melhor que a elitista e significativamente melhor que o ACO canônico (BULLNHEIMER; HARTL; STRAUSS, 1999).

3.3.3 *Max-Min*

Na estratégia *Max-Min*, apenas a melhor formiga até o momento (*best-so-far*) deposita feromônio. Para evitar a estagnação do feromônio e a convergência para uma solução subótima, o feromônio τ_{ij} é limitado no intervalo entre τ_{min} e τ_{max}. Além disso, o rastro de feromônio é inicializada no valor máximo τ_{max}, que associado a uma taxa de evaporação ρ pequena, intensifica a exploração do espaço de busca no início da busca. Finalmente, nesta abordagem, o rastro de feromônio é reinicializado quando não ocorrem melhorias por um determinado número de iterações consecutivas. A equação de deposição de feromônio

para a estratégia *Max-Min* é:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \tau_{ij}^{bs}, \tag{7}$$

$$\tau_{ij} = \begin{cases} \tau_{max}, & \text{se } \tau_{ij} > \tau_{max} \\ \tau_{min}, & \text{se } \tau_{ij} < \tau_{min} \end{cases}. \tag{8}$$

O limite superior do rastro de feromônio é limitado por $Q/\rho L^{bs}$ (DORINGO; STÜTZLE, 2004). Esse valor é utilizado para estimar o valor de τ_{max}. Toda vez que um novo melhor caminho é encontrado, o valor de τ_{max} é atualizado. O valor de τ_{min} é atualizado para τ_{max}/a, onde a é uma constante arbitrária. Resultados experimentais indicam que, para evitar a estagnação do feromônio, a limite inferior τ_{min} é mais importante que o limite superior. No entanto, τ_{max} é relevante para determinar o feromônio durante a reinicialização do rastro de feromônio (STÜTZLE, 1999).

3.3.4 Sistema de Colônia de Formigas

Sistema de Colônia de Formigas (ACS, *Ant Colony System*) é uma variante que apresenta três principais inovações em relação ao ACO. A primeira inovação envolve o uso de uma regra de escolha de ações mais agressiva, permitindo uma exploração mais intensa da experiência adquirida pelas formigas. A segunda é que a evaporação e o depósito de feromônio ocorrem apenas nas arestas pertencentes à melhor solução encontrada até o momento. A terceira inovação é que, após cada movimento, a formiga reduz a quantidade de feromônio na aresta que ela percorre, o que favorece a exploração de caminhos alternativos.

Durante a construção da solução, uma formiga k na posição i escolhe se mover para a posição j baseada na seguinte regra de escolha pseudo-aleatória proporcional:

$$j = \begin{cases} argmax_{l \in \mathcal{N}_i^k}[\tau_{il}.\eta_{il}^{\beta}], & \text{se } q \leq q_0 \\ p_{ij}^k, & \text{senão} \end{cases}, \tag{9}$$

onde q é uma variável aleatória distribuída uniformemente entre 0 e 1, q_0 é um constante que define probabilidade de escolha da posição da vizinhança que tem o maior valor para $\tau_{il}.\eta_{il}^{\beta}$ ou da escolha aleatória utilizando a equação 2. Com probabilidade q_0, a formiga escolhe o melhor movimento possível com base nos trilhos de feromônio e nas informações heurísticas. Com probabilidade $1 - q_0$, a formiga realiza uma exploração mais direcionada para outros caminhos. O parâmetro q_0 permite ajustar o grau de exploração e a concentração da busca em torno da melhor solução encontrada ou em novas soluções.

A atualização global do feromônio no ACS é realizada apenas pela formiga que encontrou a melhor solução até o momento, o que reduz a complexidade computacional por iteração. A equação de atualização global do feromônio é dada por:

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho\Delta\tau_{ij}^{bs}. \tag{10}$$

Além disso, o ACS utiliza uma atualização local do feromônio, onde, ao se mover de i para j, a formiga reduz a quantidade de feromônio naquela aresta para promover a exploração de outros caminhos. A equação para essa atualização local é:

$$\tau_{ij} = (1 - \xi)\tau_{ij} + \xi\tau_0, \tag{11}$$

onde τ_0 é o valor inicial do feromônio e ξ é um parâmetro de atualização. Esse procedimento ajuda a evitar a convergência prematura e aumenta a exploração do espaço de busca.

3.4 Aplicações do ACO

Esta seção apresenta aplicações do algoritmo ACO que são consideradas relevantes no contexto desta dissertação. Foram selecionados estudos que utilizam o ACO com uma ou mais das seguintes características. Espaço de busca em grade, onde as soluções são exploradas em uma malha estruturada. Otimização multiobjetivo, que envolve a busca de soluções que atendem a múltiplos critérios de desempenho simultaneamente. Múltiplas colônias, onde várias colônias de formigas operam paralelamente, cada uma especializada em diferentes aspectos do problema ou buscando diferentes soluções. Aplicação em programas CAD, utilizando o ACO para otimizar o roteamento de trajetórias em projetos ou sistemas de manufatura assistidos por softwares de projeto.

Em (MIAO et al., 2021) é proposta uma variação do algoritmo ACO para o planejamento de trajetórias de robôs móveis em ambientes internos. Para aumentar a eficiência em tempo real e a segurança no planejamento de trajetórias, fatores de orientação angular e de exclusão de obstáculos são incorporados na probabilidade de transição do ACO. O espaço de busca do algoritmos é representado por uma grade. A distância Euclidiana entre a posição atual do robô e a posição final ou intermediária é utilizada como informação heurística da formiga. Os resultados experimentais mostram que o algoritmo desenvolvido é capaz de encontrar trajetórias globalmente otimizadas, com alta eficiência em tempo real e excelente estabilidade no planejamento de trajetórias.

Em (HATEM et al., 2021) o algoritmo ACO é utilizado para otimizar trajetórias em máquinas de controle numérico computadorizado (*Computer Numerical Control*, CNC) com o objetivo de reduzir tempos de fabricação. A abordagem proposta visa minimizar o tempo de deslocamento da ferramenta de corte durante operações de fresamento e perfuração, otimizando o percurso e a sequência ideal de operações. Essas sequências que são baseadas em códigos de usinagem gerados por programas CAD. Esses processos resultaram em melhorias consistentes de aproximadamente 10,41% para fresamento e 16,58% para perfuração.

O trabalho (ABBAS; HAMZA; MOHAMED, 2014) apresenta um estudo de otimização de planejamento de percurso para operações de furação em componentes com grande número de furos dispostos em padrões circulares concêntricos, comum em equipamentos como trocadores de calor e separadores alimentícios. O problema foi modelado como um TSP e resolvido com Algoritmos Genéticos, uma versão básica do ACO, e uma variante modificada do ACO que inclui uma matriz de feromônios inicializada com viés para penalizar longas distâncias. Simulações numéricas demonstraram que a variante modificada do ACO supera as abordagens tradicionais, reduzindo em até 15% a distância total percorrida em leiautes complexos comparado ao caminho espiral gerado por softwares CAD. A aplicabilidade da abordagem foi validada em uma tela separadora com 2100 furos, mostrando sua escalabilidade.

Em (ALAYA; GHEDIRA, 2007) é proposto um algoritmo genérico baseado no ACO para resolver problemas de otimização multiobjetivo, denominado m-ACO. O algoritmo é parametrizado pelo número de colônias de formigas e trilhas de feromônio. Foram testadas quatro variantes do m-ACO no problema da mochila multiobjetivo, destacando-se a variante m-ACO4, que utiliza uma única colônia com múltiplas trilhas de feromônio, associando cada trilha a um critério a ser otimizado. Comparações com algoritmos evolutivos da literatura mostraram que o m-ACO4 geralmente supera os concorrentes, encontrando soluções superiores as dos EAs testados.

Em (LÓPEZ-IBÁÑEZ; STÜTZLE, 2010) apresenta-se um algoritmos flexível e configurável para algoritmos ACO aplicados a problemas de otimização multiobjetivo (MOACO, *Multi-objective Ant Colony Optimization*). A abordagem, fruto da síntese de algoritmos MOACO existentes, permite tanto a instância de algoritmos conhecidos quanto a criação de novas combinações. O MOACO permite a utilização de múltiplas colônias que

constroem soluções com trilhas de feromônios independentes. As colônias cooperam compartilhando soluções para atualizar rastros de feromônios e para detectar soluções que melhor equilibram os objetivos. A configuração automática demonstrou resultados superiores aos algoritmos MOACO de estado da arte no problema TSP com dois objetivos. Destaca-se o potencial do método proposto para expandir o uso de técnicas de configuração automática em algoritmos de otimização multiobjetivo, promovendo avanços em problemas complexos.

O trabalho (ZHANG et al., 2019) aborda o problema de roteamento de veículos com janelas de tempo flexíveis como um problema multiobjetivo, onde é possível atender clientes fora do intervalo de tempo exigido, dentro de uma tolerância, permitindo reduzir custos logísticos às custas da satisfação do cliente. A solução proposta baseia-se no ACO, aprimorado com operadores de mutação que melhoram a busca local e exploram soluções globais. A abordagem gera soluções eficientes, mostrando resultados competitivos com os melhores publicados. Este método destaca o equilíbrio entre custos e satisfação do cliente, sendo aplicável a diversos problemas reais.

3.5 Considerações Finais

Neste capítulo, foram abordados os fundamentos e aplicações do ACO. Inicialmente, o TSP foi introduzido como exemplo típico de aplicação, demonstrando a eficiência do ACO em problemas combinatórios. Em seguida, os conceitos fundamentais do algoritmo foram explicados, incluindo variações para melhorar a convergência, a exploração do espaço de busca e preservar as melhores soluções. Adicionalmente, aplicações com características relevantes foram apresentadas, como otimização multiobjetivo, roteamento com múltiplas colônias e integração com ferramentas CAD, ilustrando a versatilidade do ACO em diferentes contextos.

Essas discussões fornecem uma base sólida para os desenvolvimentos subsequentes desta dissertação, nos quais o ACO será adaptado e aplicado a problemas específicos de roteamento de trilhas em PCIs.

Capítulo 4

ROTEADOR PROPOSTO

ESTE capítulo apresenta o roteador de trilhas de PCI baseado no algoritmo de otimização por colônia de formigas. Nele, são detalhados os componentes fundamentais do modelo, incluindo a função objetivo, o espaço de busca, as estratégias heurísticas e o papel do feromônio. Também são abordadas as heurísticas para avaliação de cruzamentos e casamento de comprimentos, além da estrutura geral do algoritmo, os hiperparâmetros e a análise de sua complexidade computacional. Esses elementos estabelecem a base teórica e prática necessária para compreender e avaliar o desempenho do roteador em diferentes cenários.

O algoritmo ACO é especialmente adequado para o roteamento de trilhas em PCIs devido à sua eficácia na resolução de problemas de roteamento. Originalmente desenvolvido para desafios semelhantes, o ACO encontra soluções ótimas ou quase ótimas de forma eficiente. O roteamento de trilhas em PCIs envolve restrições complexas, o que se alinha bem com as forças do ACO. Embora amplamente utilizado em diversos cenários de roteamento, o ACO é subutilizado no roteamento de trilhas em PCIs. Nossa abordagem aproveita suas capacidades para enfrentar os desafios específicos do casamento de comprimento das trilhas de dados e de clock em circuitos de alta frequência.

A Seção 4.1 apresenta a função objetivo utilizada no processo de otimização. A Seção 4.2 explica o espaço de busca do algoritmo e como uma PCI é representada. A Seção 4.3 discute as duas estratégias que o algoritmo pode utilizar para calcular a informação heurística das formigas. A Seção 4.4 descreve o papel do feromônio no algoritmo, explicando como ele é atualizado e influencia a exploração do espaço de busca. A Seção 4.5 detalha as três heurísticas que podem ser utilizada pelo algoritmo para avaliar o cruzamentos de trilhas. A Seção 4.6 aborda as três heurísticas que podem ser aplicadas no

processo de casamento de comprimentos das trilhas. A Seção 4.7 apresenta a estrutura geral do algoritmo, descrevendo as etapas principais e as interações entre as colônias durante a otimização. A Seção 4.8 discute os hiperparâmetros utilizados no algoritmo. A Seção 4.9 analisa a complexidade computacional do algoritmo, considerando o impacto da quantidade de trilhas, área da PCI e número de formigas por colônia. Por fim, a seção 4.10 contém as considerações finais sobre o conteúdo deste capítulo.

4.1 Função Objetivo

O roteador de trilhas com ACO otimiza três objetivos simultaneamente. O comprimento da trilha, o número de cruzamentos com outras trilhas, e o casamento de comprimento entre as trilhas. A função de otimização $\mathcal{O}pt_k^{(c)}$ calcula a média ponderada destes três objetivos. No ACO, uma colônia é responsável por realizar o roteamento de uma trilha. Para uma formiga k, de uma colônia c, a função objetivo que avalia a trilha é:

$$\mathcal{O}pt_k^{(c)} = \frac{\omega_1 \Lambda_k^{(c)} + \omega_2 \Upsilon_k^{(c)} + \omega_3 \Gamma_k^{(c)}}{\omega_1 + \omega_2 + \omega_3}, \tag{12}$$

onde $\Lambda_k^{(c)}$ é o comprimento da trilha, $\Upsilon_k^{(c)}$ é a quantidade de vezes que essa trilha se cruza com outras trilhas, $\Gamma_k^{(c)}$ é o casamento de comprimento calculado pela soma das diferenças de comprimento desta trilha com as outras trilhas da solução. ω_1, ω_2 e ω_3 são pesos para regular a importância dos objetivos.

4.2 Espaço de Busca

O espaço de busca do roteador representa a superfície de cobre de uma PCI retangular. A área da PCI é dividida em células quadradas do mesmo tamanho. As dimensões da PCI são especificadas no algoritmo em linhas e colunas, uma placa com m linhas e n colunas tem $m \times n$ células. As linhas são contadas de baixo para cima, e as colunas da esquerda para a direita. Cada célula representa uma possível posição que pode ser visitada por uma formiga do ACO. As posições são enumeradas a partir do canto inferior esquerdo. Ou seja, a posição 1 é a célula localizada na primeira linha da primeira coluna, por sua vez, a posição $n + 1$ é a célula logo acima desta, na segunda linha e primeira coluna. A posição i está localizada em (m_i, n_i).

As formigas do ACO visitam as células e depositam feromônio nestas células em busca do caminho ótimo para a trilha. Um circuito é composto por diversas trilhas

interligando os componentes, no roteador estipulou-se que cada trilha é traçada por uma colônia distinta. Portanto, o número de colônias no roteador ACO é igual ao número de trilhas na PCI, esse número é representado por $\mathbb{C}$. O número de formigas é igual para todas as colônias, esse valor é representado por $\mathbb{K}$. Cada colônia em $\mathbb{C}$ tem a quantidade de formigas $\mathbb{K}$, para um total de $\mathbb{C} \times \mathbb{K}$ formigas. Cada trilha tem as posições inicial e final conhecidas. Essas posições são representadas por:

$$s^{(c)} = (m_s^{(c)}, n_s^{(c)}) \quad e \quad g^{(c)} = (m_g^{(c)}, n_g^{(c)}),\ c \in \mathbb{C}, \tag{13}$$

onde $s^{(c)}$ é a posição inicial e $g^{(c)}$ é a posição final da trilha traçada pela colônia c.

A Figura 7 apresenta o espaço de busca de uma PCI com dimensões 6×17 células e 3 trilhas. Os pontos verdes indicam as posições iniciais das trilhas os pontos vermelhos as posições finais. As colonias são inicializadas nos pontos verdes e as formigas realizam a busca para determinar a trajetória ótima para os pontos vermelhos.

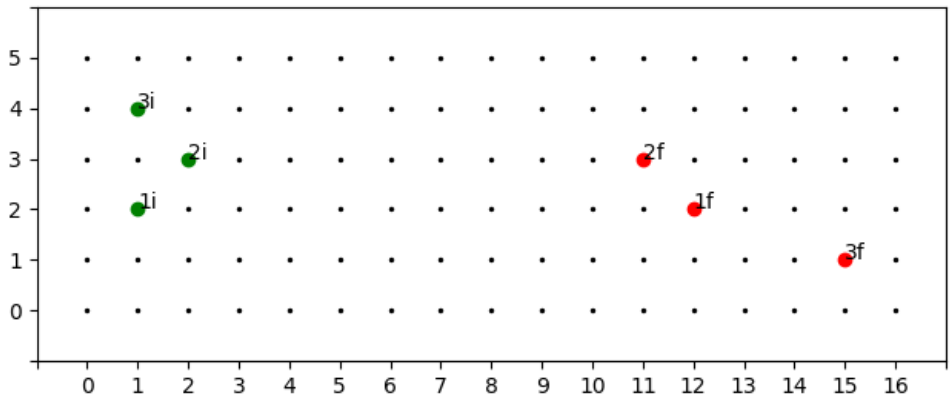


Figura 7: Representação do espaço de busca de uma PCI com 3 trilhas.

O caminho construído por uma formiga k da colônia c é representado por $\lambda_k^{(c)}$ e o comprimento deste caminho por $\Lambda_k^{(c)}$. Esse caminho é o conjunto de posições escolhidas pela formiga para ir de $s^{(c)}$ até $g^{(c)}$. O comprimento é o número de posições no caminho. Uma formiga k da colônia c na posição i pode se deslocar para qualquer posição na sua vizinhança, desde que esta posição não seja a posição inicial ou final de outra colônia. A vizinhança possível da posição i é representada por $\mathcal{N}_i$. Ela é composta pelas células diretamente a direita, esquerda, acima e abaixo desta posição. Ou seja:

$$\mathcal{N}_i = [(m_i, n_i + 1), (m_i, n_i - 1), (m_i + 1, n_i), (m_i - 1, n_i)]. \tag{14}$$

A probabilidade de uma formiga k pertencente a colônia c na posição i se deslocar para a posição j é dada por:

$$p_{ij}^{(k_c)} = \frac{\left(\tau_{ij}^{(c)}\right)^{\alpha} \cdot \left(\eta_{ij}^{(c)}\right)^{\beta}}{\sum_{l \in \mathcal{N}_i} \left(\tau_{il}^{(c)}\right)^{\alpha} \cdot \left(\eta_{il}^{(c)}\right)^{\beta}}, \tag{15}$$

onde $\tau_{ij}^{(c)}$ é o feromônio da colônia c associado a posição j, $\eta_{ij}^{(c)}$ é a informação heurística para a colônia c na posição j, α e β são constantes para determinar a influência do feromônio e da informação heurística e $\mathcal{N}_i$ é a vizinhança factível para a posição i.

4.3 Informação Heurística da formiga

A informação heurística representa a atratividade de uma formiga por uma posição. No ACO canônico, essa informação é calculada pelo inverso da distância entre as duas posições, ou seja, a formiga é atraída com mais intensidade para uma posição cuja distância é menor. Como no roteador com ACO o espaço de busca é dividido em células quadradas de dimensões iguais, e a vizinhança factível é composta apenas pelas células diretamente aos lados, acima e abaixo da posição atual, a distância é a mesma para toda a vizinhança. Dessa forma, apenas o feromônio influenciaria a escolha da próxima posição. No algoritmo apresentado neste capítulo, duas estratégias distintas são propostas para o cálculo da informação heurística.

A primeira estratégia consiste na utilização da Distância Euclidiana (DEC) entre a posição j e $g^{(c)}$, que é a posição final da trilha traçada pela colônia c. Dessa forma, as formigas são atraídas pelas posições que mais se aproximam de sua posição final. Portanto, a informação heurística é calculada pela seguinte equação:

$$\eta_j^{(c)} = \left(\sqrt{(m_g^{(c)} - m_j)^2 + (n_g^{(c)} - n_j)^2}\right)^{-1} \tag{16}$$

A segunda estratégia consiste no inverso da soma dos feromônios das outras colônias (ISF) na posição j. Esta abordagem faz com que a formiga seja atraída para posições menos utilizadas por formigas de outras colônias, evitando regiões "congestionadas". Além disso, incentiva o surgimento de soluções sem cruzamentos com outras trilhas. Neste caso, a informação heurística para a colônia c na posição j é calculada por:

$$\eta_j^{(c)} = \left(\sum_{u \in \mathbb{C}} \tau_{ij}^{(u)}\right)^{-1}, u \neq c \tag{17}$$

4.4 Atualização e Depósito de Feromônio

A atualização do feromônio é dada pela equação:

$$\tau_{ij}^{(c)} = (1-\rho)\cdot\tau_{ij}^{(c)} + \left(\sum_{k=1}^{\mathbb{K}} \Delta\tau_{ij}^{k^{(c)}}\right) + \Delta\tau_{ij}^{bs^{(c)}}, \tag{18}$$

onde ρ é a constante de evaporação do feromônio. $\Delta\tau_{ij}^{k^{(c)}}$ é o feromônio depositado pela formiga k da colônia c na posição ij. $bs^{(c)}$ é a formiga da colônia c que construiu o melhor caminho até o momento. $\Delta\tau_{ij}^{bs^{(c)}}$ é o fator de elitismo, que reforça o feromônio depositado na melhor solução encontrada até o momento.

A função de quantidade de feromônio depositado $\Delta\tau_{ij}^{k^{(c)}}$ é dada por:

$$\Delta\tau_{ij}^{k^{(c)}} = \frac{Q}{f_{opt}(k^{(c)})}, \tag{19}$$

onde Q é uma constante que regula a quantidade de feromônio depositado. A função de otimização $f_{opt}(k^{(c)})$ avalia três características da solução construída pela formiga k da colônia c. O comprimento do caminho $\Lambda_k^{(c)}$, o número de cruzamentos com outras trilhas $\Upsilon_k^{(c)}$ e o casamento de comprimento $\Gamma_k^{(c)}$. A função de otimização está no denominador pois estes três objetivos são minimizados.

O comprimento $\Lambda_k^{(c)}$ é o número de células que fazem parte do caminho $\lambda_k^{(c)}$ construído pela formiga k da colônia c multiplicado pelo tamanho do passo da formiga. $\mathcal{N}_{\lambda_k^{(c)}}$ representa o número de células no caminho. O tamanho do passo S_k é o lado das células que formam o espaço de busca.

$$\Lambda_k^{(c)} = \mathcal{N}_{\lambda_k^{(c)}} \times S_k \tag{20}$$

4.5 Cruzamento Entre Trilhas

O parâmetro $\Upsilon_k^{(c)}$ quantifica o cruzamento entre trilhas e pode ser calculado de três formas. A primeira abordagem consiste no somatório do feromônio das outras colônias (SFO) para identificar o quanto a colônia atual utiliza posições muito utilizadas pelas outras colônias. Esta estratégia identifica regiões de congestão mas mede cruzamentos entre trilhas indiretamente. O número de formigas pode perturbar e atrasar a concentração de feromônio nas etapas iniciais do processo de otimização. Essa estratégia é calculada pela equação:

$$\Upsilon_k^{(c)} = \sum_{j\in\lambda_k^{(c)}} \sum_{u\in\mathbb{C}} \tau_{ij}^{(u)}, u \neq c \tag{21}$$

A segunda abordagem consiste em quantificar quantas vezes o caminho construído pela formiga k da colônia c utiliza uma posição que também é utilizada pelo melhor caminho construído das outras colônias (MCO). Ou seja, conta quantas vezes esta trilha cruza com as outras trilhas levando em consideração apenas os melhores resultados encontrados até o momento pelas outras colônias.

$$\Upsilon_k^{(c)} = \sum_{j \in \lambda_k^{(c)}} \sum_{u \in \mathbb{C}} f_c(u, j), u \neq c \tag{22}$$

$$f_c(u, j) = \begin{cases} 1, & \text{se } j \in \lambda_{bs}^{(u)} \\ 0, & \text{senão} \end{cases} \tag{23}$$

A terceira abordagem leva em consideração todos os caminhos construídos pelas outras colônias (TCO).

4.6 Casamento de Comprimento

Conforme demonstrado pela vizinhança de uma célula na Equação 14, uma das restrições impostas ao algoritmo é a limitação dos movimentos possíveis. As formigas podem realizar apenas movimentos cartesianos (para cima, baixo, direita e esquerda), enquanto movimentos diagonais são proibidos. Essa restrição simplifica o algoritmo ao garantir que todos os passos realizados pelas formigas tenham o mesmo comprimento, além de reduzir o número total de soluções possíveis no espaço de busca. No entanto, essa limitação impõe um fator restritivo no casamento de comprimentos. O comprimento mínimo de uma trilha torna-se crítico, pois, se for um valor par, não pode ser ajustado para um comprimento ímpar e vice-versa, devido à ausência de movimentos diagonais no modelo. Essa característica foi denominada paridade da trilha no contexto deste trabalho. Como consequência, a medição do casamento de comprimentos deve incorporar uma tolerância de ± 1 movimento, permitindo que trilhas com diferenças de comprimento dentro desse intervalo sejam consideradas casadas.

A função $\Gamma_k^{(c)}$ que mede o casamento de comprimento pode ser calculada de três formas. A primeira forma é o módulo da diferença entre o comprimento da trilha e o maior comprimento entre os melhores caminhos encontrados até o momento (DMC). Esta estratégia tende a aumentar o comprimento das trilhas menores, mas tem o risco de fazer o maior comprimento estagnar e não diminuir, possivelmente fazendo o processo convergir

para um mínimo local. Essa abordagem é calculada pela equação:

$$\Gamma_k^{(c)} = |\Lambda_k^{(c)} - max(V_\Lambda^{bs})|, \tag{24}$$

onde V_Λ^{bs} é um vetor com os comprimentos dos melhores caminhos encontrados até o momento para todas as trilhas.

$$V_\Lambda^{bs} = [\Lambda_{bs}^{(1)}, \Lambda_{bs}^{(2)}, ..., \Lambda_{bs}^{(c)}, ..., \Lambda_{bs}^{(\mathbb{C})}]. \tag{25}$$

A segunda forma de calcular o casamento de comprimento é pela soma dos valores absolutos das diferenças de comprimento entre o caminho construído pela formiga k da colônia c e os melhores caminhos das outras colônias (SDA).

$$\Gamma_k^{(c)} = \sum_{u \in \mathbb{C}} |\Lambda_k^{(c)} - \Lambda_{bs}^{(u)}|, u \neq c. \tag{26}$$

Essa abordagem aproxima todos os comprimentos. No entanto, caso a maioria das trilhas com posições iniciais e finais mais próximas casem seus comprimentos em um tamanho inferior ao comprimento mínimo da trilha que deve ser obrigatoriamente maior, o processo pode estagnar em um mínimo local. A maior trilha não pode reduzir seu comprimento e como as menores casaram seus comprimentos elas tendem a não crescerem para o comprimento da maior trilha.

A terceira estratégia é a soma das diferenças quadráticas (SDQ). Nessa abordagem, as maiores diferenças tem mais importância na função objetivo, mitigando a situação de estagnação da abordagem anterior.

$$\Gamma_k^{(c)} = \sum_{u \in \mathbb{C}} (\Lambda_k^{(c)} - \Lambda_{bs}^{(u)})^2, u \neq c. \tag{27}$$

4.7 Algoritmo

A Figura 8 apresenta o fluxograma simplificado do roteador de trilhas de circuito com ACO que utiliza as equações apresentadas neste capítulo.

Na primeira etapa os hiper-parâmetros do algoritmo e as características do espaço de busca são lidos de um arquivo e validados. Estas configurações são os pesos dos objetivos, α, β, ρ e outras constantes, número de formigas por colônia ($\mathbb{K}$), dimensões da PCI, posições inicias e finais das trilhas, número de iterações, critério de parada e seleção de quais heurísticas serão utilizadas nos cálculos. Ainda nesta etapa as matrizes de feromônio e outras variáveis são alocadas na memória.

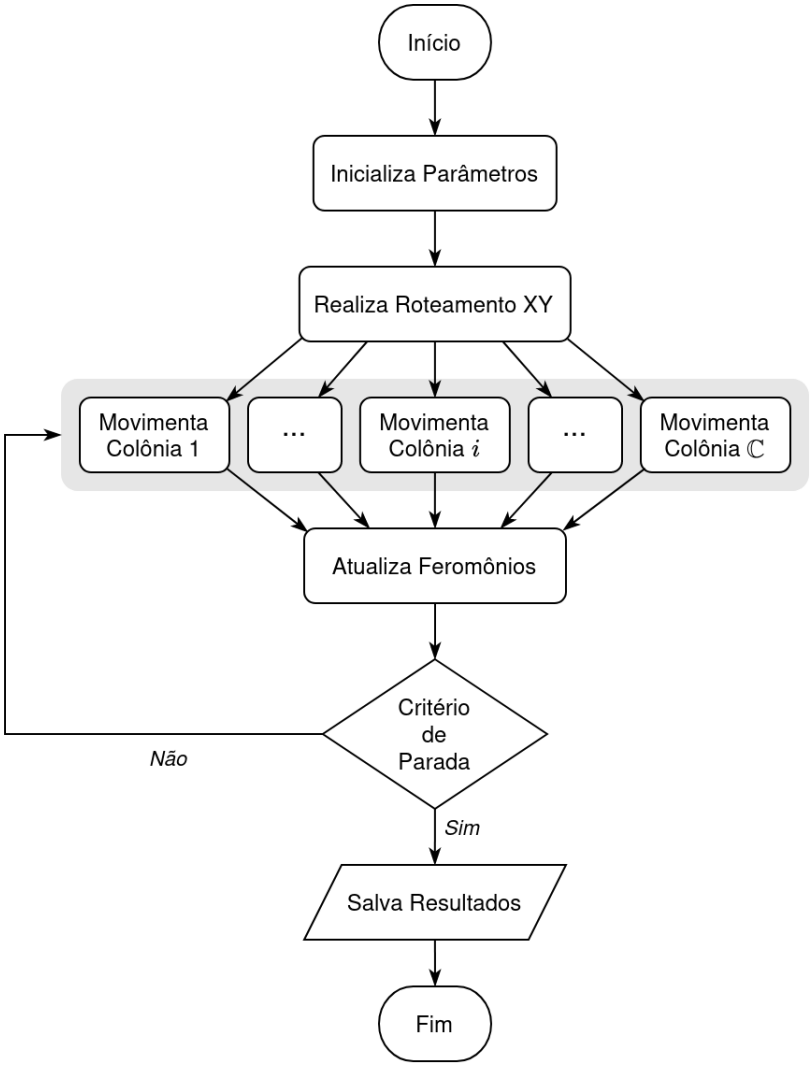


Figura 8: Fluxograma do roteador de trilhas de PCI

Na segunda etapa é realizado o roteamento preliminar para cada trilha, utilizando o algoritmo XY. Este algoritmo é muito simples e consiste em gerar um caminho que avança na direção X até atingir a coluna da posição final e então avança na direção Y até chegar na posição final. Este roteamento é realizado sem levar em consideração nenhuma restrição do problema ou do espaço de busca, utilizado apenas como ponto de partida para processo de otimização.

A etapa seguinte consiste na movimentação das colônias, onde as formigas constroem caminhos. Cada colônia é responsável por uma trilha da PCI, e seu processo de construção só depende de sua própria matriz de feromônio e do resultado encontrado pelas outras colônias na iteração anterior. Dessa forma, a construção de caminhos das colônia na iteração atual são independes uns dos outros, pois dependem apenas de informações passadas que são comuns a todas as colônias. Essa estratégia permite que o processo de

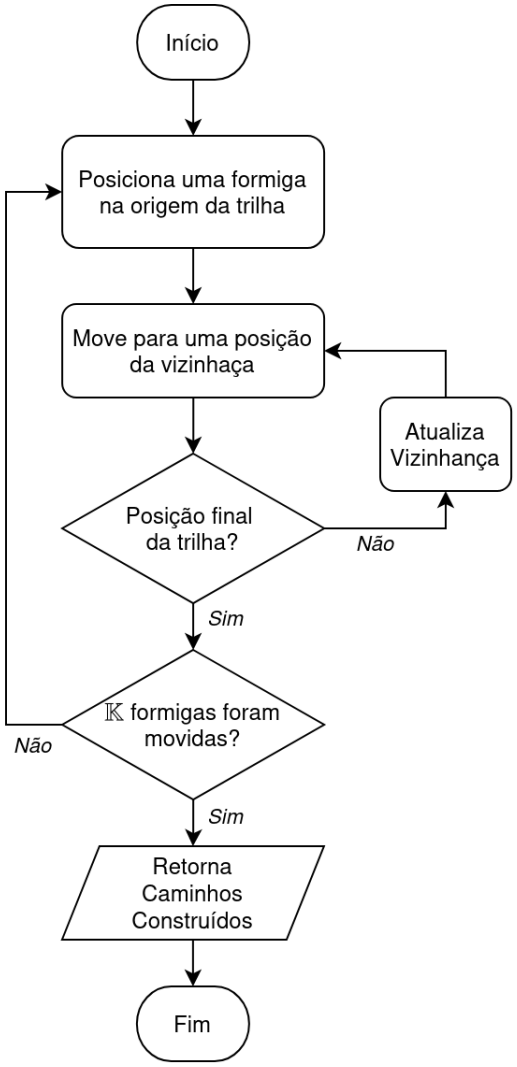


Figura 9: Fluxograma do processo de construção de caminhos de uma colônia.

movimentação de cada colônia seja realizado em um processo independente, executado em paralelo. A Figura 9 apresenta o fluxograma do algoritmo para a movimentação das formigas de uma colônia. A escolha do próxima posição da formiga é realizada utilizando a função de probabilidade, conforme a Equação 15.

Quando todos os processos de movimentação das colônias são finalizados, o algoritmo segue para a etapa seguinte. Nesta etapa os resultados são avaliados e as matrizes de feromônio de cada colônia são atualizadas com a Equação 18.

Finalmente, o caso o critério de parada tenha sido atingido o processo chega ao fim e os resultados são salvos para serem avaliados e exibidos posteriormente. Caso contrário o processo iterativo continua da etapa de movimentação das colônias. O processo é limitado pelo número de iterações, este o critério de parada obrigatório. Além do número de iterações outros dois critérios de parada podem ser utilizados. O primeiro interrompe

a execução do algoritmo quando um resultado sem cruzamentos com casamento de comprimento entre as trilhas é encontrado. O segundo critério interrompe a execução quando um resultado considerado ótimo é encontrado. Um resultado ótimo consiste em uma solução sem cruzamentos com casamento de comprimento entre as trilhas cujo comprimento é mínimo. O comprimento mínimo é o menor comprimento possível para a trilha que tem a maior distância entre os pontos inicial e final. Este critério de parada só é válido para um problema cujo resultado ótimo é conhecido e foi utilizado para validar o algoritmo nos cenários de teste apresentados e avaliados no próximo capítulo. Este segundo critério de parada não é aplicável para um problema desconhecido ou de alta complexidade em uma aplicação real.

4.8 Hiperparâmetros

O algoritmo ACO possui diversos hiperparâmetros que precisam ser calibrados conforme a aplicação. Além dos parâmetros típicos, o algoritmo apresentado nas seções anteriores introduz novas constantes que devem ser configuradas de forma adequada. A Tabela 1 apresenta esses hiperparâmetros e uma breve descrição de suas funções. Os valores para os parâmetros α, β, ρ e τ_0 são comumente adotados em aplicações do ACO (DORINGO; STÜTZLE, 2004). O valor da intensidade do elitismo (e) acelera a convergência, mas valores muito altos podem levar à convergência para ótimos locais, limitando a exploração do espaço de soluções. O parâmetro Q deve ser ajustado para equilibrar o crescimento da intensidade do feromônio depositado pelas formigas. As constantes $omega_1$, $omega_2$ e $omega_3$ controlam a importância relativa dos diferentes objetivos no problema, sendo que os valores na tabela refletem uma abordagem ingênua de atribuir a mesma ou quase a mesma importância a todos os objetivos. No ACO, o número de formigas em uma colônia ($\mathbb{K}$) é tipicamente escolhido com base no tamanho do espaço de busca. No caso do algoritmo proposto, esse valor seria o número de colunas (m) multiplicado pelo número de linhas (n). No entanto, como o algoritmo utiliza múltiplas colônias, para espaços de busca grandes e com muitas trilhas, uma abordagem mais eficiente consiste em escolher um valor que balanceie a exploração do espaço de busca e o tempo de execução.

A técnica introduzida neste capítulo propõe diversas heurísticas distintas para mensurar os objetivos de cruzamentos ($\Upsilon_k^{(c)}$) e casamento de comprimento de trilhas ($\Gamma_k^{(c)}$) e para o cálculo da informação heurística da formiga ($\eta_j^{(c)}$). Portanto, na execução do

Tabela 1: Hiperparâmetros do Algoritmo

Hiperparâmetro	Descrição
α	Influência do feromônio
β	Influência da informação heurística
ρ	Taxa de evaporação
τ_0	Feromônio inicial
e	Intensidade do Elitismo
Q	Constante de Depósito de Feromônio
ω_1	Importância da minimização de comprimento
ω_2	Importância da minimização dos cruzamentos
ω_3	Importância do casamento de comprimento
$\mathbb{K}$	Número de formigas por colônia

algoritmo, além dos hiperparâmetros, essas estratégias também devem ser selecionadas. A Tabela 2 apresenta de forma organizada a função as possíveis heurísticas e suas siglas.

Tabela 2: Heurísticas

Função	Heurística	Sigla
$\eta_j^{(c)}$	Distância Euclidiana	DEC
	Inverso da soma de feromônios das outras colonias	ISF
$\Upsilon_k^{(c)}$	soma de feromônios das outras colonias	SFO
	Melhor caminho das outras colônias	MCO
	Todos os caminhos construídos	TCO
$\Gamma_k^{(c)}$	Diferença com o maior dos melhores caminhos construídos	DMC
	Soma das diferenças absolutas	SDA
	Soma das diferenças quadráticas	SDQ

4.9 Complexidade do Algoritmo

Para avaliar a complexidade de um algoritmo para encontrar todos os caminhos possíveis entre um par de pontos em uma grade com dimensões de n linhas e m colunas, deve-se considerar os possíveis movimentos que é possível fazer em uma posição. Para o tipo de problema resolvido pelo algoritmo deste trabalho, existem quatro possíveis movimentos, para cima, para baixo, para a esquerda e para direita. A complexidade na Notação *Big* O para o pior caso (mais complexo) neste este tipo de cenário é dada por $O(4^{n\times m})$. A complexidade para k pares de pontos é $k \times O(4^{n\times m})$, que pode ser simplificado para $O(4^{n\times m})$.

Para o algoritmo proposto neste capítulo, a complexidade leva em consideração o tamanho $n \times m$ da grade, o número de iterações t, o número de colônias $\mathbb{C}$, o número

de formigas por colônia $\mathbb{K}$ e as restrições das heurísticas e múltiplos objetivos. Para os objetivos propostos, pode-se considerar que a complexidade varia entre $O(n^2 \times m^2)$ e $O(n^3 \times m^3)$. Combinando todos estes fatores, a complexidade do ACO para roteamento de trilhas de PCI varia entre $O(t \times \mathbb{C} \times \mathbb{K} \times n^2 \times m^2)$ e $O(t \times \mathbb{C} \times \mathbb{K} \times n^3 \times m^3)$.

4.10 Considerações Finais

Neste capítulo, foram apresentados e discutidos os principais elementos que compõem o algoritmo proposto para roteamento de trilhas em PCIs, incluindo sua estrutura, estratégias e restrições. A função objetivo, descrita na Seção 4.1, foi detalhada para evidenciar os critérios que orientam a busca por soluções otimizadas. Na Seção 4.2, foram explicados o espaço de busca e a representação da PCI, que desempenham papel central na definição das possibilidades de solução.

Foram abordadas as estratégias heurísticas e o uso de feromônios, que são fundamentais para o comportamento do algoritmo. Essas seções exploraram diferentes abordagens para cálculo de informações heurísticas, avaliação de cruzamentos e casamento de comprimentos das trilhas, destacando as opções implementadas e seus impactos no desempenho do algoritmo.

A estrutura geral do algoritmo foi apresentada na Seção, enfatizando a interação entre as colônias de formigas e as etapas do processo de otimização. Além disso, a análise dos hiperparâmetros e da complexidade computacional forneceu uma visão detalhada dos fatores que influenciam o desempenho, considerando aspectos como a quantidade de trilhas, a área da PCI e o número de formigas por colônia.

Esses elementos fornecem a base teórica e prática para o funcionamento do algoritmo, preparando o terreno para a validação e análise de resultados nos capítulos subsequentes. As escolhas e restrições adotadas refletem um equilíbrio entre simplicidade e eficácia, direcionando o algoritmo para resolver problemas de roteamento e casamento de comprimento em PCIs de alta frequência.

Capítulo 5

RESULTADOS DE DESEMPENHO

NESTE capítulo, são apresentados e analisados os resultados obtidos com a aplicação do Roteador de Trilhas de PCI proposto no Capítulo 4. Na Seção 5.1, os cenários utilizados na execução do algoritmo são descritos. Na Seção 5.2, as diversas combinações de heurísticas são avaliadas e comparadas. A Seção 5.3 apresenta os resultados de roteamento obtidos para cada um dos cenários propostos. Na Seção 5.4, são exibidos os resultados da execução do algoritmo em cenários provenientes de trabalhos relacionados, comparando nossa técnica com outras abordagens. Por fim, a Seção 5.5 apresenta as considerações finais deste capítulo.

5.1 Cenários Avaliados

Esta seção apresenta todos os cenários avaliados com o algoritmo desenvolvido. Os cenários são separados em dois grupos, cenários propostos e cenários externos. cenários propostos são aqueles introduzidos por este trabalho, elaborados para testar, validar e avaliar o algoritmo desenvolvido. Cenários externos são aqueles provenientes de outros estudos e foram utilizados para realizar comparações com outras técnicas.

5.1.1 Cenários Propostos

Esta Subseção apresenta os sete cenários propostos. Esses cenários foram projetados com o objetivo de avaliar a eficácia e a robustez do algoritmo desenvolvido neste trabalho. Esses cenários foram elaborados para simular diferentes condições e situações de roteamento em PCIs, variando em termos de complexidade, densidade de trilhas e requisitos de casamento de comprimento. Cada cenário foi configurado de forma a explorar aspectos específicos do desempenho do algoritmo, como a capacidade de minimizar cruzamentos,

igualar comprimentos e otimizar o uso do espaço disponível. A construção desses cenários permite validar o funcionamento do algoritmo em situações controladas e analisar seu comportamento em diferentes contextos.

A Figura 10 apresenta a configuração do cenário 1. Nesse cenário, duas trilhas devem ser roteadas em um espaço de busca composto por 6 linhas e 17 colunas, totalizando 102 células disponíveis para visitação pelas formigas do algoritmo. As duas trilhas estão alinhadas, com suas posições iniciais e finais localizadas na mesma linha, mas possuem comprimentos mínimos diferentes. A trilha maior (trilha 1) deve ser traçada em linha reta ao longo do eixo x, enquanto a trilha menor (trilha 2) deve realizar desvios ao longo do eixo y para igualar o comprimento. A Figura 11 apresenta quatro possíveis soluções ótimas, todas com comprimento 14, obtidas por meio de resolução manual.

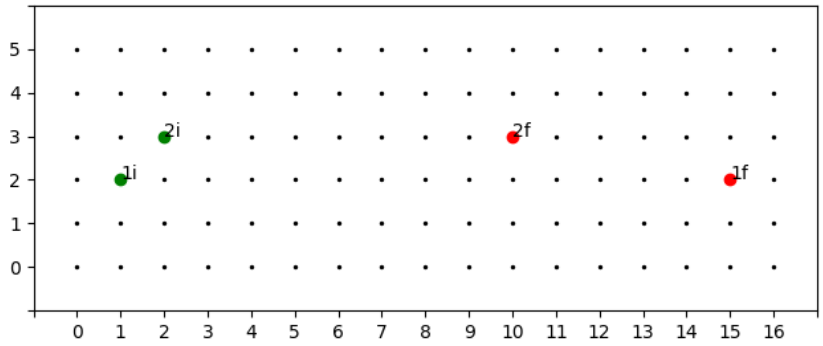


Figura 10: Representação do espaço de busca do cenário 1.

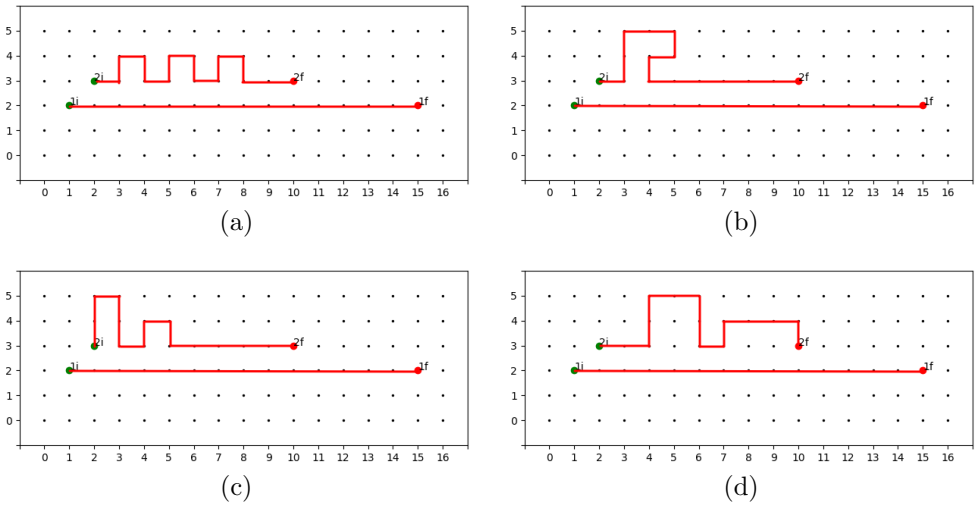


Figura 11: Resultados ótimos com comprimento 14 para o cenário 1.

A Figura 12 apresenta a configuração do cenário 2. Nesse cenário, duas trilhas devem ser roteadas em um espaço de busca composto por 6 linhas e 17 colunas, totalizando

102 células disponíveis para visitação pelas formigas do algoritmo. As duas trilhas possuem suas posições iniciais e finais localizadas nas mesmas colunas, porém com posições finais situadas em linhas diferentes. Uma das trilhas precisa contornar a outra, enquanto esta utiliza o espaço entre as posições iniciais e finais para realizar seu traçado. A Figura 13 apresenta quatro possíveis soluções ótimas, todas com comprimento 17, obtidas por meio de resolução manual.

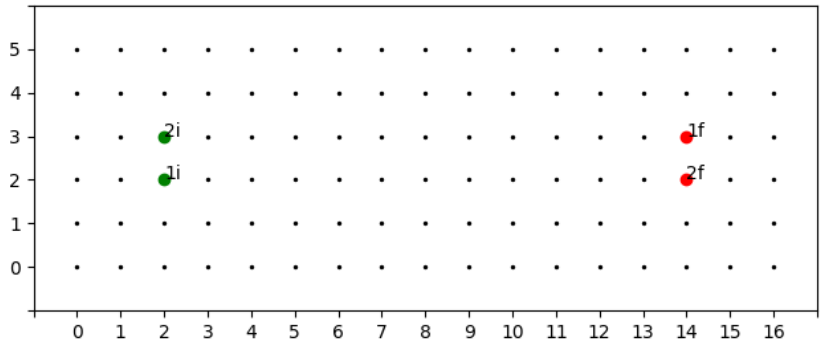
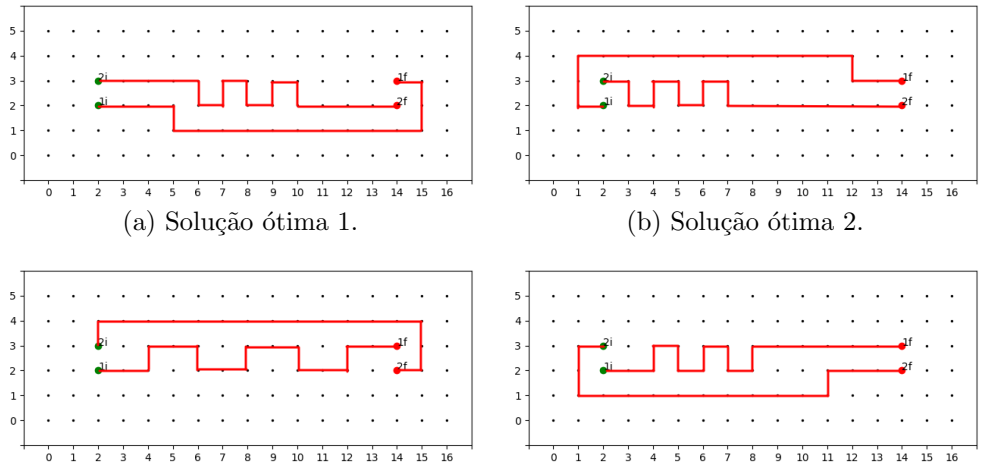


Figura 12: Representação do espaço de busca do cenário 2.



(a) Solução ótima 1.

(b) Solução ótima 2.

(c) Solução ótima 3.

(d) Solução ótima 4.

Figura 13: Soluções ótimas com comprimento 17 para o cenário 2.

A Figura 14 apresenta a configuração do cenário 3. Nesse cenário, três trilhas devem ser roteadas em um espaço de busca composto por 6 linhas e 17 colunas, totalizando 102 células disponíveis para visitação pelas formigas do algoritmo. As trilhas 1 e 2 possuem suas posições iniciais e finais alinhadas em colunas diferentes, enquanto a trilha 3 tem suas posições distribuídas em linhas e colunas distintas. Nesse cenário, a trilha 3 determina o comprimento mínimo e deve contornar as outras duas, enquanto estas competem pelo espaço livre para igualar seus comprimentos ao da trilha 3. A Figura 15 apresenta quatro

possíveis soluções ótimas, todas com comprimento 17, obtidas por meio de resolução manual.

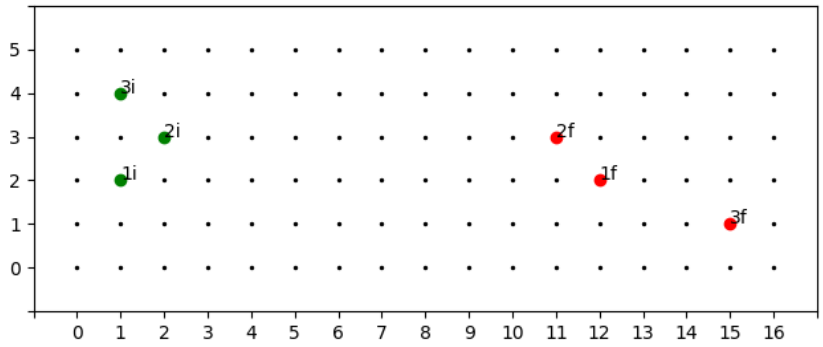
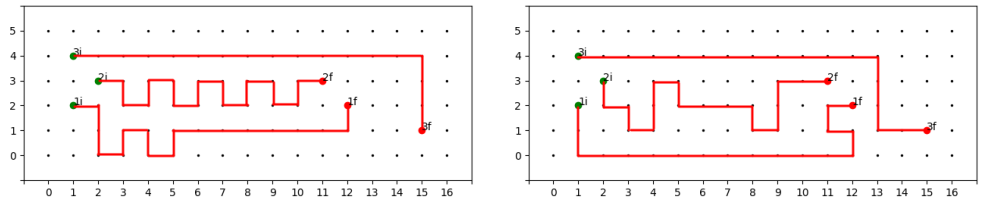
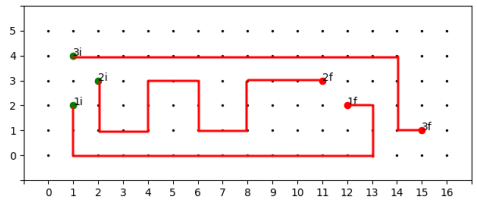


Figura 14: Representação do espaço de busca do cenário 3.

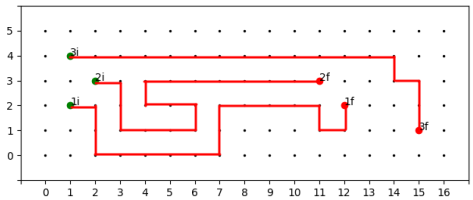


(a) Solução ótima 1.

(b) Solução ótima 2.



(c) Solução ótima 3.



(d) Solução ótima 4.

Figura 15: Soluções ótimas com comprimento 17 para o cenário 3.

A Figura 16 apresenta a configuração do cenário 4. Neste cenário, quatro trilhas devem ser roteadas em um espaço de busca composto por 15 linhas e 21 colunas, totalizando 315 células. Este cenário simula a interconexão entre um CI com encapsulamento quadrado QFP (*Quad Flat Package*), representado pelas posições iniciais das trilhas, e um CI do tipo DIP (*Dual In-line Package*), representado pelas posições finais. A trilha 4 determina o comprimento mínimo e deve contornar a trilha 3. A Figura 17 apresenta quatro possíveis soluções ótimas, todas com comprimento 14, obtidas por meio de resolução manual.

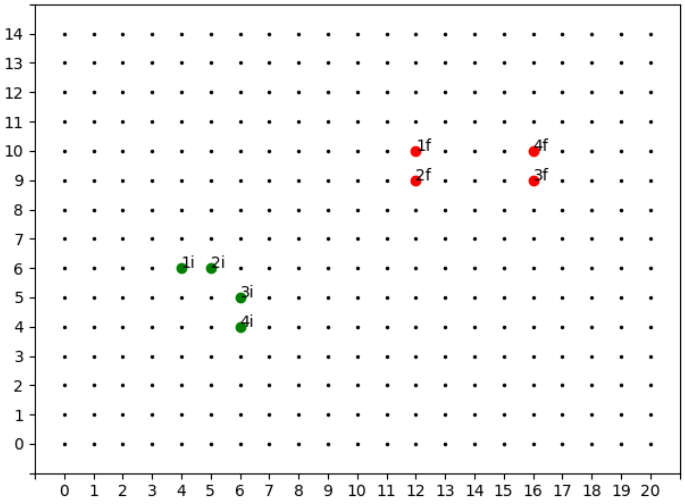
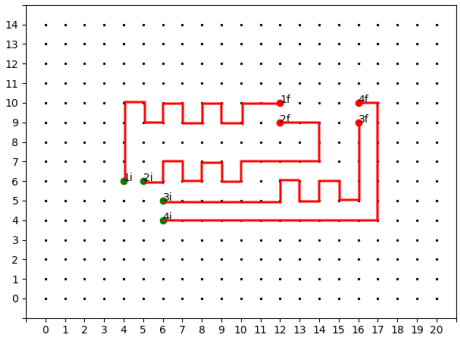
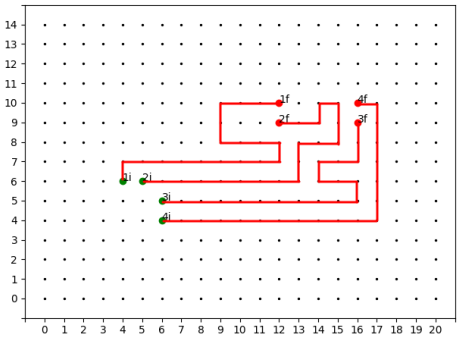


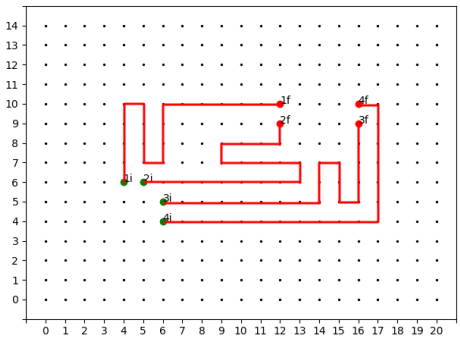
Figura 16: Representação do espaço de busca do cenário 4.



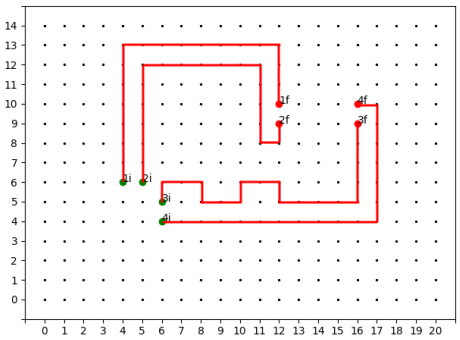
(a) Solução ótima 1.



(b) Solução ótima 2.



(c) Solução ótima 3.



(d) Solução ótima 4.

Figura 17: Soluções ótimas com comprimento 18 para o cenário 4.

A Figura 18 apresenta a configuração do cenário 5. Neste cenário, cinco trilhas devem ser roteadas em um espaço de busca composto por 11 linhas e 15 colunas, totalizando 165 células. A trilha 3 define o comprimento mínimo. Este cenário apresenta uma grande quantidade de espaço livre para o roteamento, permitindo maior flexibilidade na disposição das trilhas. A Figura 19 apresenta quatro possíveis soluções ótimas, todas com comprimento 11, obtidas por meio de resolução manual.

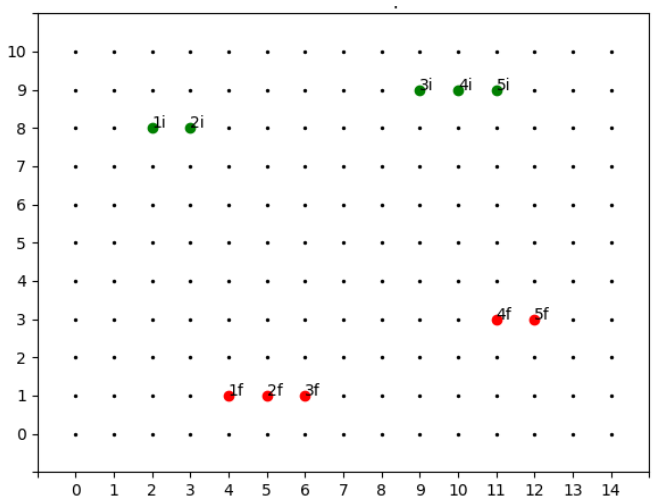
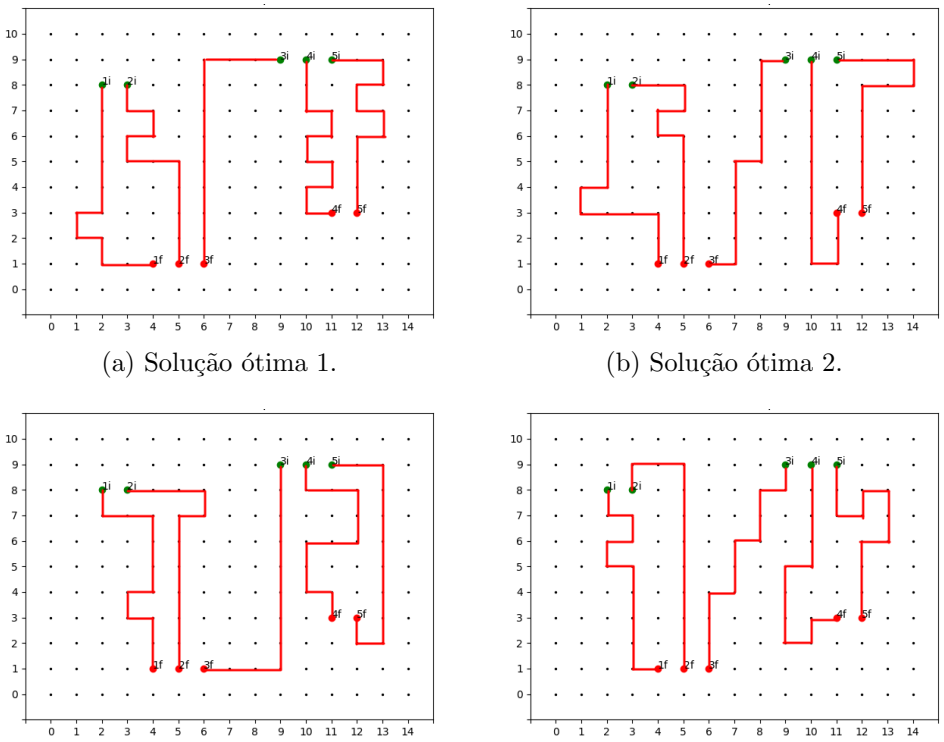


Figura 18: Representação do espaço de busca do cenário 5.



(a) Solução ótima 1.

(b) Solução ótima 2.

(c) Solução ótima 3.

(d) Solução ótima 4.

Figura 19: Soluções ótimos com comprimento 11 para o cenário 5.

A Figura 20 apresenta a configuração do cenário 6. Neste cenário seis trilhas devem ser traçada em um espaço de busca composto por 15 linhas e 21 colunas, totalizando 315 células. Este cenário emula a conexão de um CI tipo DIP (*Dual In-line Package*) com barramentos próximos (que podem ser outros CIs). A trilha 1 dita o comprimento mínimo. As trilhas 2 e 3 tem suas posições iniciais e finais próximas, de forma que devem competir

pelo espaço com a trilha 1 e buscar espaço livre nas extremidades da placa. As tilhas 4, 5 e 6 tem suas posições iniciais e finais alinhadas e equidistantes, de forma que devem competir pelo espaço entre estas posições. Levando-se em consideração as distâncias entre as posições, o número de células e o comprimento mínimo, este cenário tem muito espaço disponível. A Figura 21 exibe quatro possíveis soluções ótimas, todas com comprimento 11, obtidas através de resolução manual.

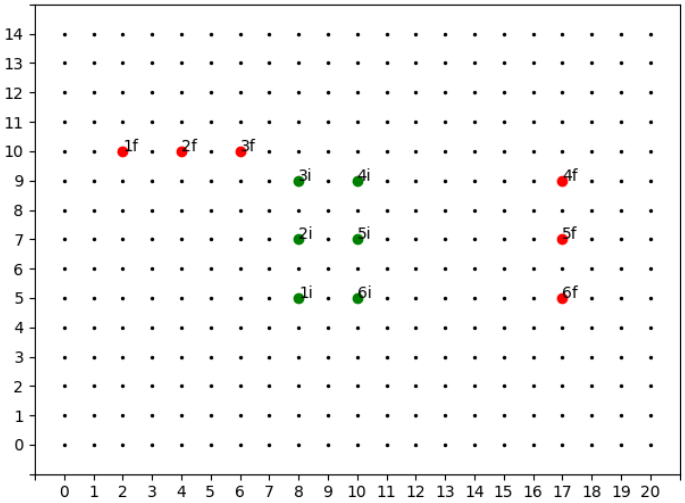
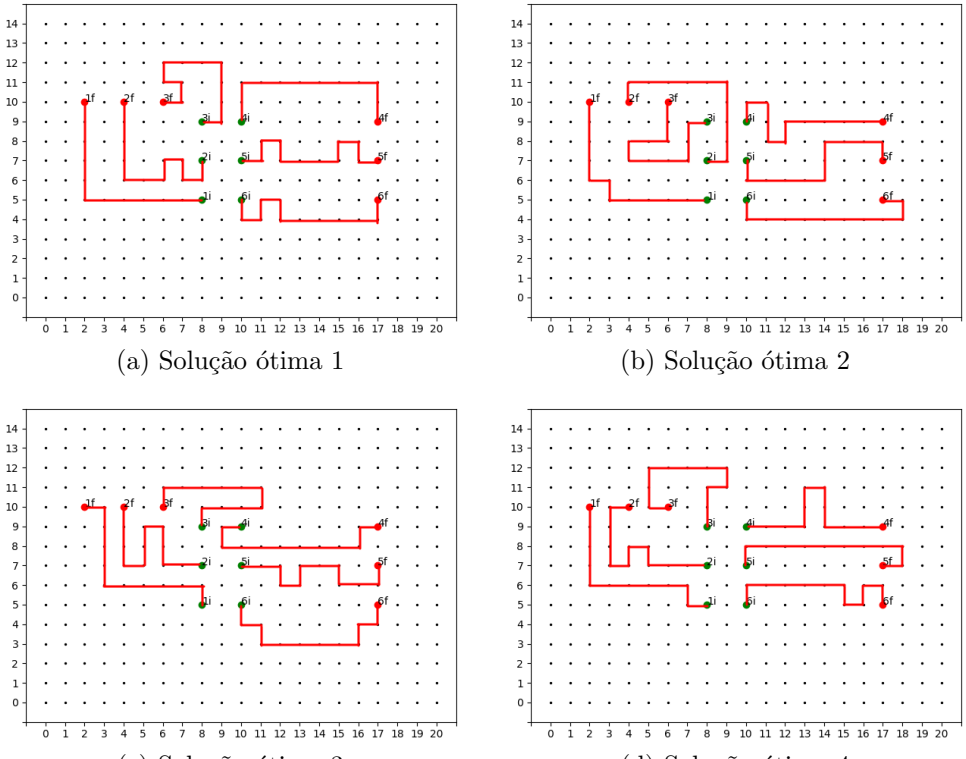


Figura 20: Representação do espaço de busca do cenário 6.



(a) Solução ótima 1

(b) Solução ótima 2

(c) Solução ótima 3

(d) Solução ótima 4

Figura 21: Resultados ótimos com comprimento 11 para o cenário 6.

A Figura 22 apresenta a configuração do cenário 7. Neste cenário, seis trilhas devem ser roteadas em um espaço de busca composto por 15 linhas e 21 colunas, totalizando 315 células. Este cenário é uma extensão do cenário 4, com a adição de duas trilhas extras. A trilha 6 define o comprimento mínimo. A proximidade entre as posições iniciais das trilhas intensifica a competição pelo espaço central da placa, tornando o roteamento mais desafiador. A Figura 23 exibe quatro possíveis soluções ótimas, todas com comprimento 22, obtidas por meio de resolução manual.

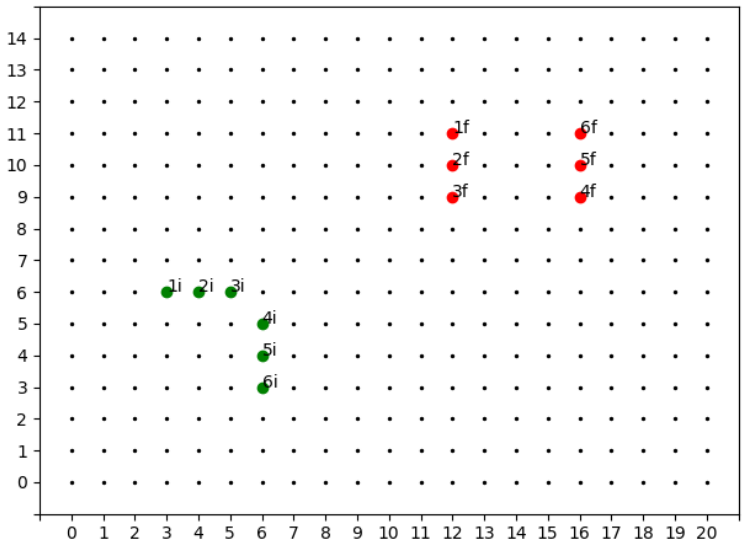
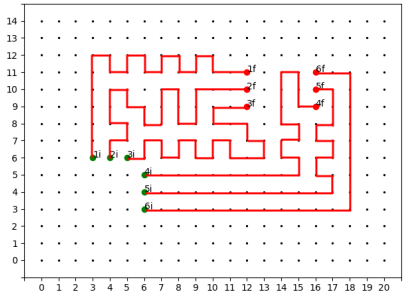
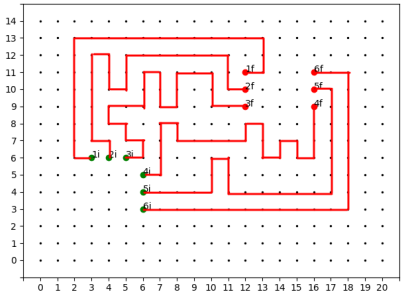


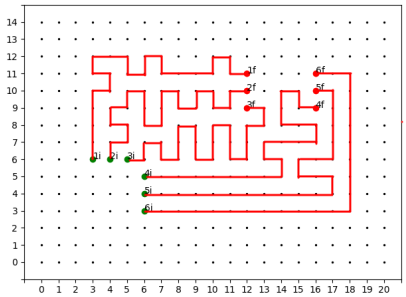
Figura 22: Representação do espaço de busca do cenário 7.



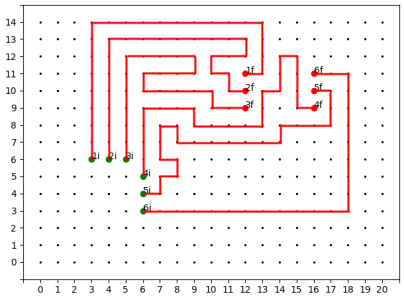
(a) Solução ótima 1.



(b) Solução ótima 2.



(c) Solução ótima 3.



(d) Solução ótima 4.

Figura 23: Soluções ótimas com comprimento 22 para o cenário 7.

5.1.2 Cenários Externos

Os cenários externos são provenientes de estudos e algoritmos distintos e foram utilizados para fins de comparação com o método desenvolvido. Esses cenários foram resolvidos com técnicas alternativas que adotam objetivos e restrições diferentes, adicionando um nível extra de desafio para a comparação direta. Uma característica comum nos cenários externos é a presença de áreas proibidas, representadas por quadrados cinzas no espaço de busca. Essas áreas exigem que o algoritmo ajuste sua estratégia para encontrar caminhos viáveis dentro das restrições impostas. Essa característica permite avaliar a robustez e flexibilidade do algoritmo em situações mais complexas e variadas. Três cenários com essas características foram testados. A escolha desses cenários ocorreu pela semelhança na representação do espaço de busca, que é subdividido em células numa grade. Essa semelhança possibilitou a aplicação do algoritmo desenvolvido com poucas alterações, sendo necessário apenas tratar a inicialização das posições proibidas.

A Figura 24 apresenta a configuração do cenário externo 1 (E1). Neste cenário, quatro trilhas devem ser traçadas em um espaço de busca composto por 16 linhas e 23 colunas, totalizando 368 células, das quais 76 são áreas proibidas. A trilha 3 determina o comprimento mínimo e possui apenas um trajeto viável. A Figura 25 mostra quatro possíveis soluções ótimas, todas com comprimento 32, obtidas por meio de resolução manual.

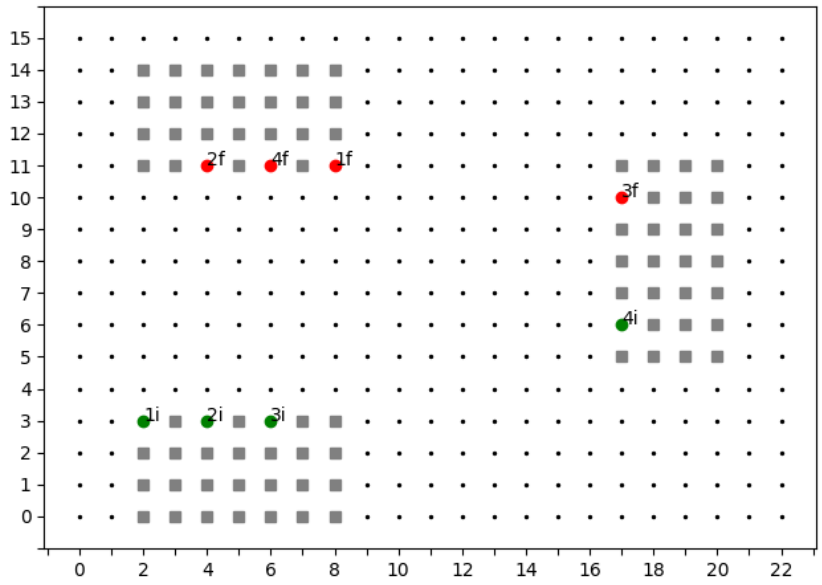
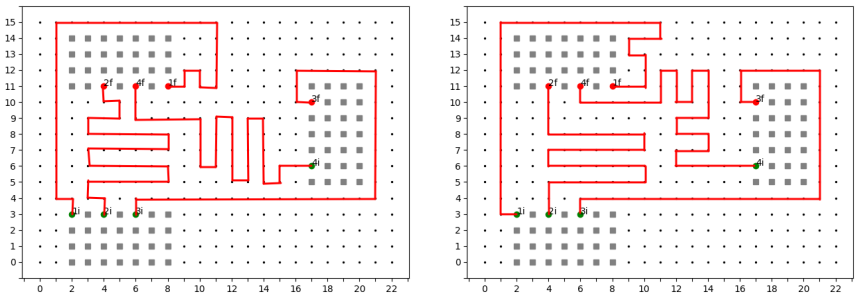
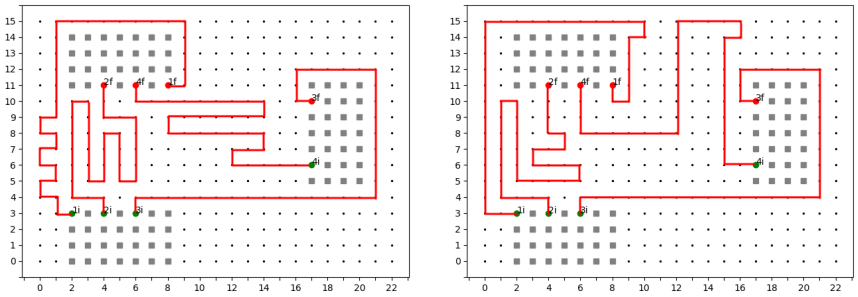


Figura 24: Representação do espaço de busca do cenário E1.



(a) Solução ótima 1. (b) Solução ótima 2.



(c) Solução ótima 3. (d) Solução ótima 4.

Figura 25: Soluções ótimas com comprimento 32 para o cenário E1.

A Figura 26 apresenta a configuração do cenário externo 2 (E2). Neste cenário, seis trilhas devem ser traçadas em um espaço de busca composto por 16 linhas e 23 colunas, totalizando 368 células, das quais 80 são áreas proibidas. A trilha 4 determina o comprimento mínimo e possui mais de um trajeto viável. A Figura 27 mostra quatro possíveis soluções ótimas, todas com comprimento 26, obtidas por meio de resolução manual.

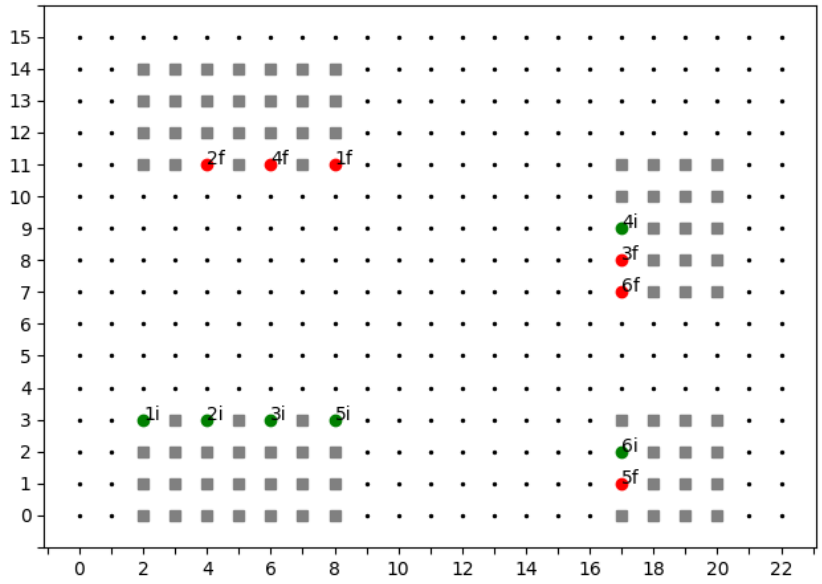
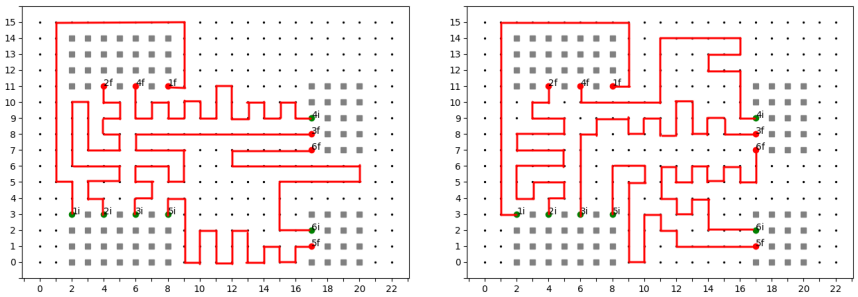
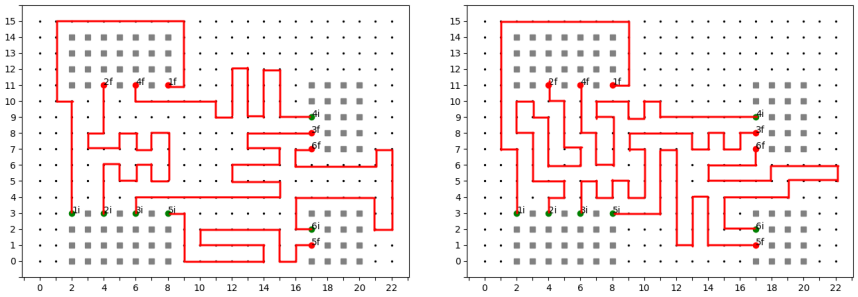


Figura 26: Representação do espaço de busca do cenário E2.



(a) Solução ótima 1. (b) Solução ótima 2.



(c) Solução ótima 3. (d) Solução ótima 4.

Figura 27: Soluções ótimas com comprimento 23 para o cenário E2.

A Figura 28 apresenta a configuração do cenário externo 3 (E3). Neste cenário, cinco trilhas devem ser traçadas em um espaço de busca composto por 30 linhas e 30 colunas, totalizando 900 células, das quais 97 são proibidas. A trilha 1 determina o comprimento mínimo. A Figura 29 exibe quatro possíveis soluções ótimas, todas com comprimento 27, obtidas por meio de resolução manual.

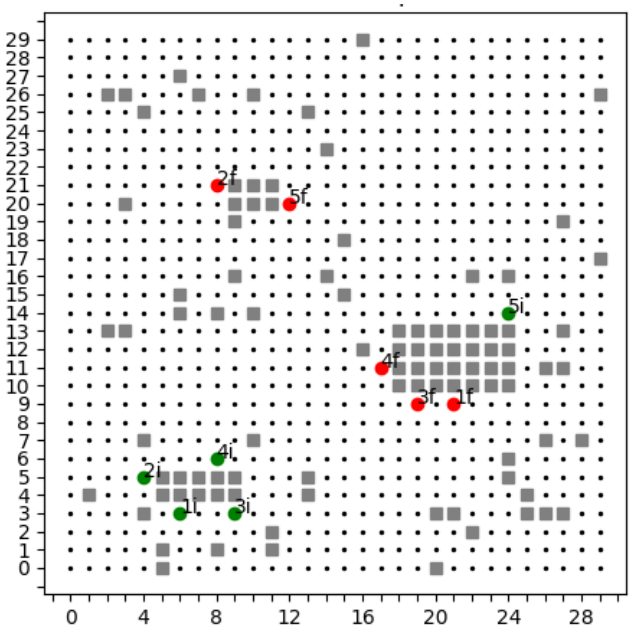
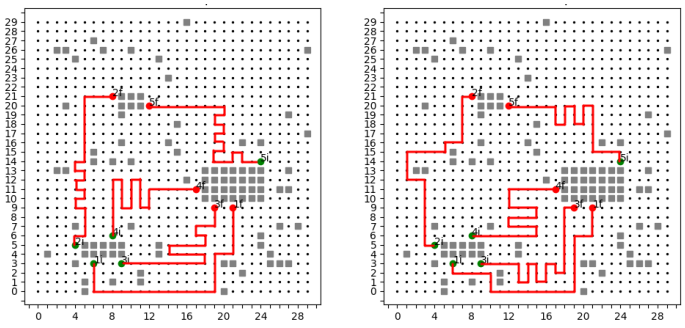
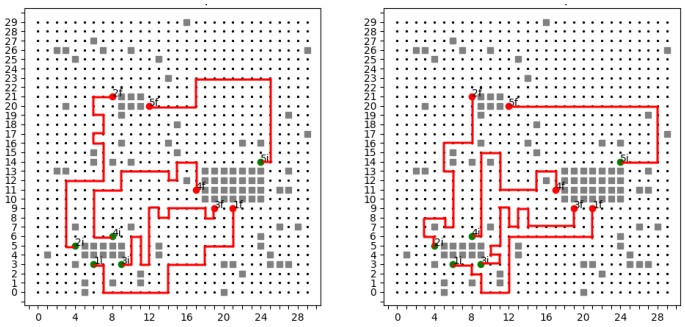


Figura 28: Representação do espaço de busca do cenário E3.



(a) Solução ótima 1. (b) Solução ótima 2.



(c) Solução ótima 3. (d) Solução ótima 4.

Figura 29: Soluções ótimas com comprimento 27 para o cenário E3.

5.2 Comparação das Heurísticas

No capítulo anterior, foram propostas diversas heurísticas distintas para o roteador de trilhas com ACO. Duas estratégias foram apresentadas para o cálculo da informação heurística da formiga: a distância euclidiana (DEC) e o inverso da soma dos feromônios das outras colônias (ISF). Três estratégias foram propostas para a avaliação do casamento de comprimento: a diferença de comprimento em relação à trilha mais longa (DMC), a soma das diferenças absolutas (SDA) e a soma das diferenças quadráticas (SDQ). Além disso, foram sugeridas três heurísticas para avaliar o cruzamento entre trilhas: evitar os melhores caminhos das outras colônias (MCO), evitar todos os caminhos das outras colônias (TCO) e evitar as trilhas de feromônios das outras colônias (SFO). A Tabela 3 apresenta as dezoito possíveis combinações dessas heurísticas para os cálculos da informação heurística da formiga, casamento de comprimento e cruzamento entre trilhas.

Todos os casos foram avaliados com o mesmo conjunto de hiperparâmetros, apresentados na Tabela 4. É possível que em determinado caso uma seleção diferente de parâmetros possa obter melhores resultados, no entanto, para fins de comparação, foram escolhidos valores típicos e a mesma importância foi dada a todos os objetivos.

Tabela 3: Combinações das heurísticas.

Caso	Heurística da Formiga	Comprimento	Cruzamento
1	DEC	DMC	MCO
2			TCO
3			SFO
4		SDA	MCO
5			TCO
6			SFO
7		SDQ	MCO
8			TCO
9			SFO
10	ISF	DMC	MCO
11			TCO
12			SFO
13		SDA	MCO
14			TCO
15			SFO
16		SDQ	MCO
17			TCO
18			SFO

O algoritmo foi executado com todas as combinações de heurísticas nos sete cenários propostos. Os resultados apresentados correspondem às médias de cem execuções para cada combinação aplicada a cada cenário. Para avaliar as combinações de heurísticas, foram consideradas três métricas principais: o percentual de resultados ótimos, o número médio de iterações e o tempo médio de execução. Outras métricas obtidas, mas não utilizadas diretamente na análise, estão disponíveis no Apêndice A.

O critério de parada adotado foi a obtenção de um resultado ótimo ou o alcance do limite máximo de cem mil iterações. Um resultado é considerado ótimo quando todas as trilhas possuem o mesmo comprimento, sendo este o comprimento mínimo obtido pela resolução manual do cenário, e quando não há cruzamentos entre as trilhas.

Todas as execuções foram realizadas em um computador equipado com um processador *i9-12900KF* e 128, GB de memória RAM DDR5. A Subseção 5.2.1 apresenta os resultados detalhados para cada combinação de heurísticas nos diferentes cenários. Já a Subseção 5.2.2 traz uma análise consolidada do conjunto de resultados obtidos.

5.2.1 Resultados por Caso

Considerando apenas as três métricas avaliadas, a informação heurística da formiga exerceu a maior influência sobre os resultados. Assim, os resultados dos casos que utilizam a

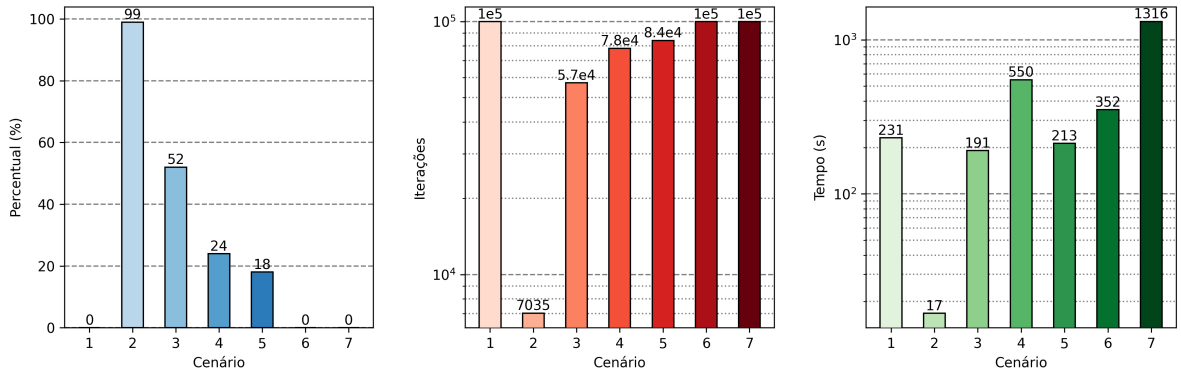
Tabela 4: Hiperparâmetros utilizados nos testes.

Hiperparâmetro	Valor
α	1
β	2
ρ	0.5
τ_0	1
e	1
Q	1
ω_1	33
ω_2	33
ω_3	34
$\mathbb{K}$	100
Iteração Máxima	100000

mesma estratégia são bastante semelhantes, variando pouco percentualmente, mas mantendo as mesmas características e tendências. Foram propostas duas estratégias para o cálculo da informação heurística da formiga. A primeira estratégia utiliza a distância Euclidiana (DEC) entre a posição atual e a posição final da trilha, favorecendo o objetivo de minimizar o comprimento da trilha. Os casos de 1 a 9 utilizam DEC. A segunda estratégia consiste no inverso da soma dos feromônios (ISF) das outras colônias, favorecendo o objetivo de minimizar cruzamentos entre trilhas. Os casos de 10 a 18 utilizam ISF. Portanto, os casos de 1 a 9 apresentam resultados semelhantes entre si, assim como os casos de 10 a 18.

A Figura 30 apresenta os resultados para o caso 1. As heurísticas utilizadas são a distância Euclidiana (DEC) para a informação heurística da formiga, a diferença com o maior caminho (DMC) para avaliar o casamento de comprimento e evitar os melhores caminhos das outras colônias (MCO) para minimizar o cruzamento entre trilhas.

A Figura 30a apresenta o percentual de resultados ótimos obtidos pelo caso 1 em cada cenário. Essa caso não obteve resultados ótimos para os cenários 1, 6 e 7. O maior percentual de resultados ótimos foi 99% obtido no cenário 2. O segundo maior percentual de ótimos foi de 52%. os cenários 4 e 5 obtiveram respectivamente 24% e 18% de ótimos. A Figura 30b apresenta o número médio de iterações. A execução nos cenários 1, 6 e 7 ocorreu até o número máximo de iterações, o que reflete a ausência de resultados ótimos para estes cenários. Nesses casos o processo convergiu para mínimos locais obtendo resultados sub-ótimos. Os resultados com menos número de iterações foram obtidos no cenário 2, que precisou de em média 7035 iterações até convergir para um ótimo. A Figura

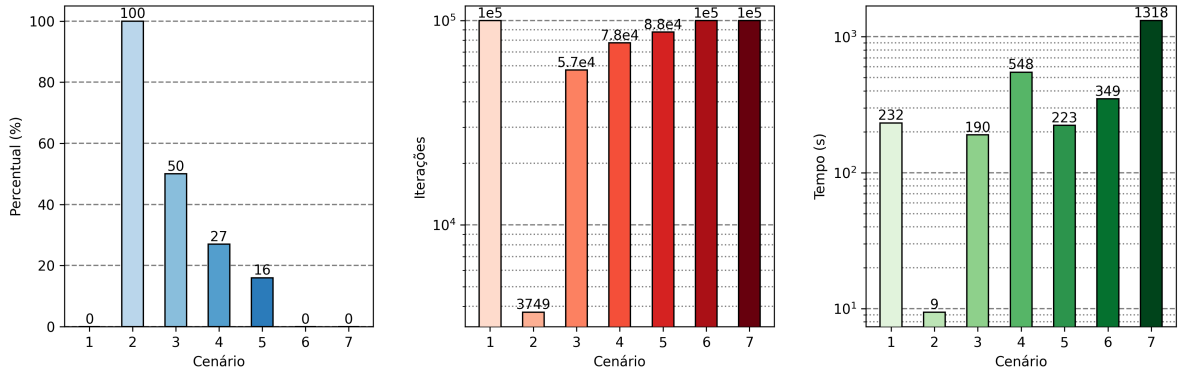


(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 30: Resultados da execução do caso 1 para os sete cenários propostos.

30c apresenta o tempo médio de execução em segundos. A execução mais rápida foi a do cenário 2, levando em média 16.8 segundos. A execução mais demorada foi para o cenário 7, em média 1316 segundos ou 21.9 minutos, o que reflete o maior número de trilhas e a ausência de ótimos para este cenário.

A Figura 31 apresenta os resultados para o caso 2. As heurísticas utilizadas são a distância Euclidiana (DEC) para a informação heurística da formiga, a diferença com o maior caminho (DMC) para avaliar o casamento de comprimento e evitar todos os caminhos das outras colônias (TCO) para minimizar o cruzamento entre trilhas.

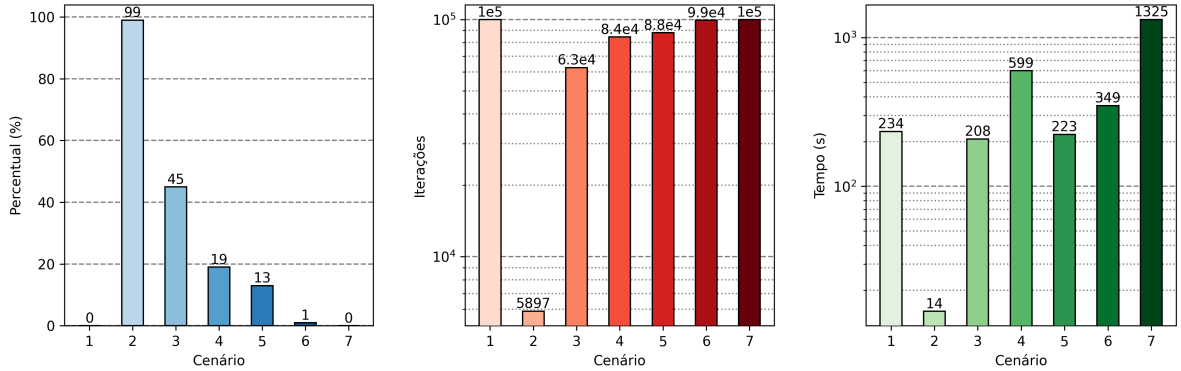


(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 31: Resultados da execução do caso 2 para os sete cenários propostos.

A Figura 32 apresenta os resultados obtidos para o caso 3. As heurísticas aplicadas incluem a distância Euclidiana (DEC) como informação heurística principal para guiar as formigas, a diferença com o maior caminho (DMC) para avaliar o casamento de com-

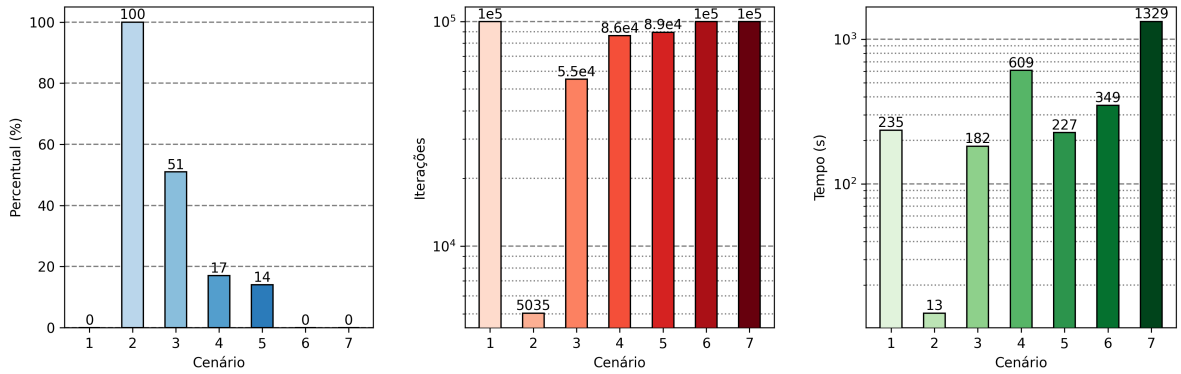
primentos, e a estratégia de evitar trilhas de feromônios de outras colônias (SFO) para minimizar os cruzamentos entre trilhas. Notavelmente, este foi o único caso entre os que utilizam DEC que conseguiu alcançar resultados ótimos no cenário 6. No entanto, como apenas um resultado ótimo foi encontrado, é provável que isso tenha ocorrido devido ao acaso.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 32: Resultados da execução do caso 3 para os sete cenários propostos.

A Figura 33 apresenta os resultados para o caso 4. As heurísticas utilizadas são a distância Euclidiana (DEC) para a informação heurística da formiga, a soma das diferenças absolutas (SDA) para avaliar o casamento de comprimento e evitar os melhores caminhos das outras colônias (MCO) para minimizar o cruzamento entre trilhas.

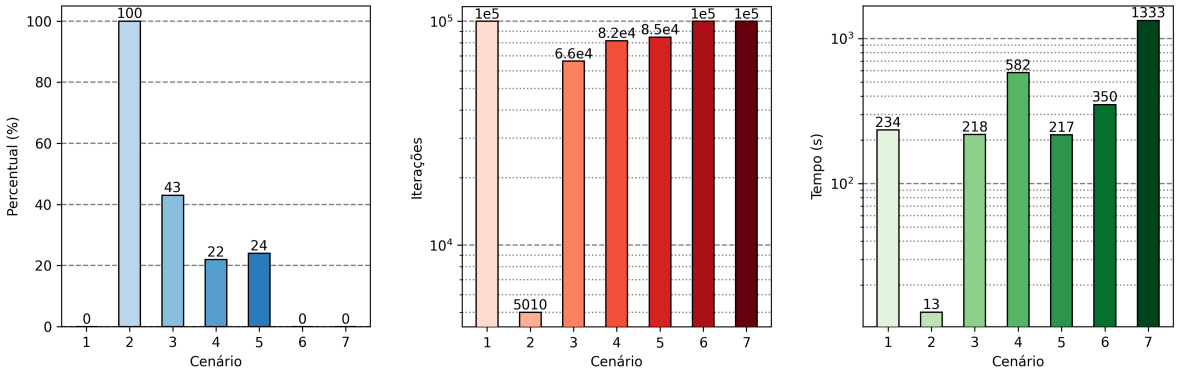


(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 33: Resultados da execução do caso 4 para os sete cenários propostos.

A Figura 34 apresenta os resultados obtidos para o caso 5. As heurísticas aplicadas incluem a distância Euclidiana (DEC) como informação heurística principal para

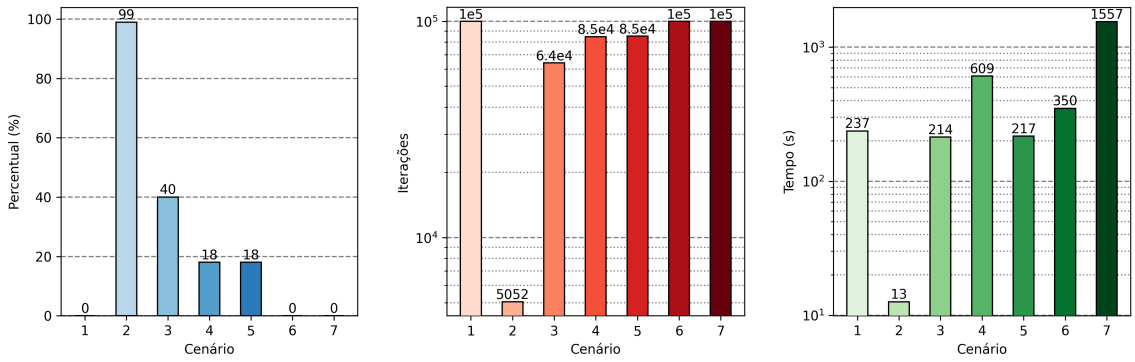
guiar as formigas, a soma das diferenças absolutas (SDA) para avaliar o casamento de comprimentos, e a estratégia de evitar todos os caminhos das outras colônias (TCO) para minimizar os cruzamentos entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 34: Resultados da execução do caso 5 para os sete cenários propostos.

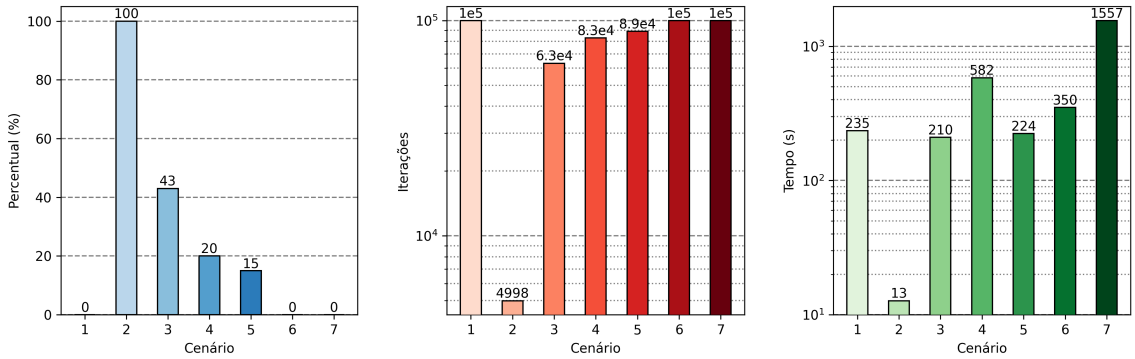
A Figura 35 apresenta os resultados para o caso 6. As heurísticas utilizadas são a distância Euclidiana (DEC) para a informação heurística da formiga, a soma das diferenças absolutas (SDA) para avaliar o casamento de comprimento e evitar as trilhas de feromônios das outras colônias (SFO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 35: Resultados da execução do caso 6 para os sete cenários propostos.

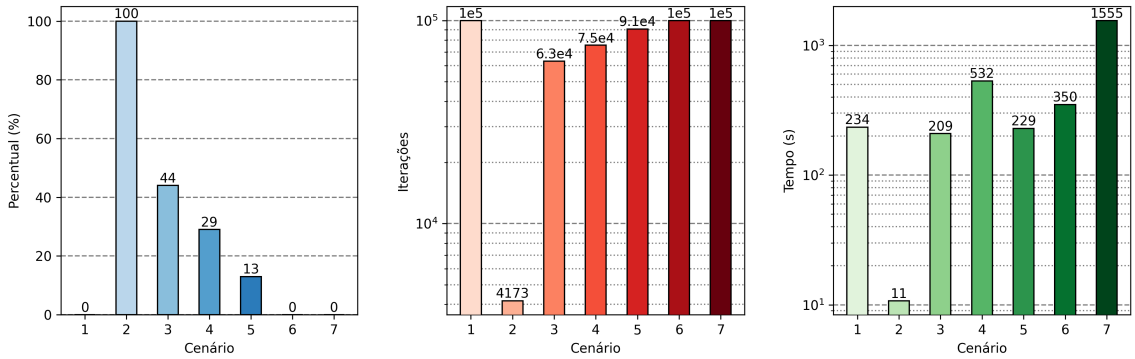
A Figura 36 apresenta os resultados para o caso 7. As heurísticas utilizadas são a distância Euclidiana (DEC) para a informação heurística da formiga, a soma das diferenças quadráticas (SDQ) para avaliar o casamento de comprimento e evitar os melhores caminhos das outras colônias (MCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 36: Resultados da execução do caso 7 para os sete cenários propostos.

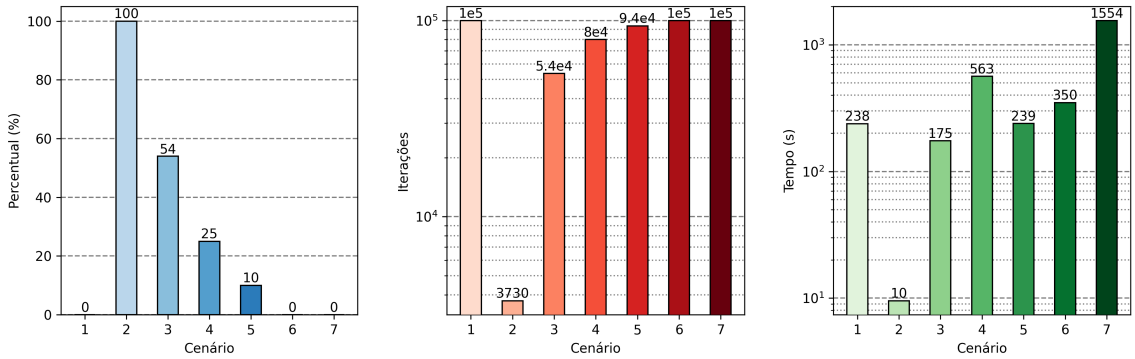
A Figura 37 apresenta os resultados para o caso 8. As heurísticas utilizadas são a distância Euclidiana (DEC) para a informação heurística da formiga, a soma das diferenças quadráticas (SDQ) para avaliar o casamento de comprimento e evitar todos os caminhos das outras colônias (TCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 37: Resultados da execução do caso 8 para os sete cenários propostos.

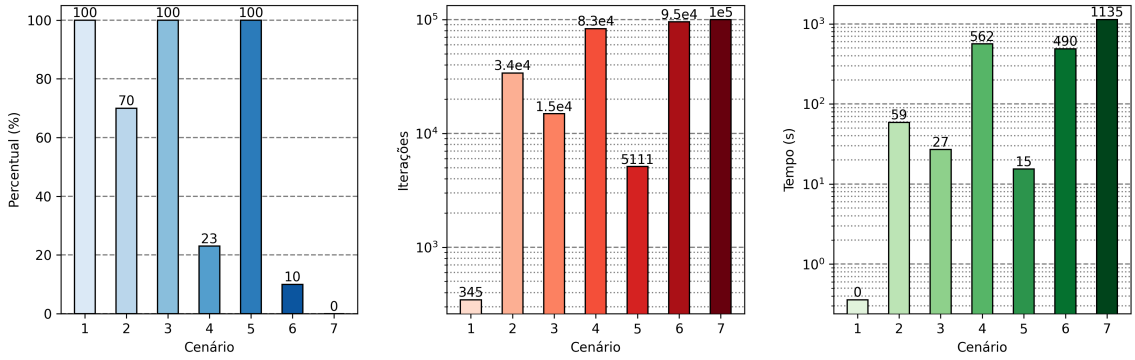
A Figura 38 apresenta os resultados para o caso 9. As heurísticas utilizadas são a distância Euclidiana (DEC) para a informação heurística da formiga, a soma das diferenças quadráticas (SDQ) para avaliar o casamento de comprimento e evitar as trilhas de feromônios das outras colônias (SFO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 38: Resultados da execução do caso 9 para os sete cenários propostos.

A Figura 39 apresenta os resultados para o caso 10. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) para a informação heurística da formiga, a diferença com o maior caminho (DMC) para avaliar o casamento de comprimento e evitar os melhores caminhos das outras colônias (MCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

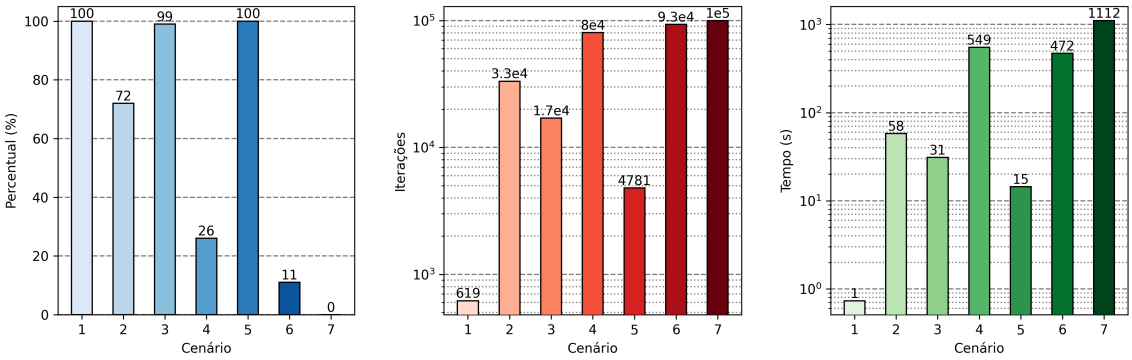
Figura 39: Resultados da execução do caso 10 para os sete cenários propostos.

A Figura 39a apresenta o percentual de resultados ótimos obtidos pelo caso 10 em cada cenário. Este caso obteve 100% de resultados ótimos nos cenários 1, 3 e 5, demonstrando uma forte eficácia em situações menos complexas. Por outro lado, o cenário 7 foi o único em que o algoritmo não alcançou resultados ótimos, destacando os desafios específicos desse cenário. Nos cenários 2, 4 e 6, os percentuais de ótimos alcançados foram 70%, 23% e 10%, respectivamente, evidenciando uma queda de desempenho conforme a complexidade aumenta.

A Figura 39b apresenta o número médio de iterações necessárias para alcançar a convergência. O cenário 1 apresentou o menor número médio de iterações, com apenas 345 iterações até alcançar um ótimo, enquanto o cenário 7 atingiu o número máximo de iterações permitidas, refletindo a ausência de resultados ótimos e a complexidade adicional do cenário.

Por fim, a Figura 39c mostra o tempo médio de execução em segundos para cada cenário. O cenário 1 foi o mais rápido, com um tempo médio inferior a 1 segundo. Em contraste, o cenário 7 apresentou o maior tempo de execução, com uma média de 1135 segundos (aproximadamente 18,9 minutos), o que reflete tanto o maior número de trilhas quanto a ausência de convergência para resultados ótimos. Esses dados evidenciam a correlação entre a complexidade do cenário, o número de trilhas e o desempenho do algoritmo em termos de tempo e iterações.

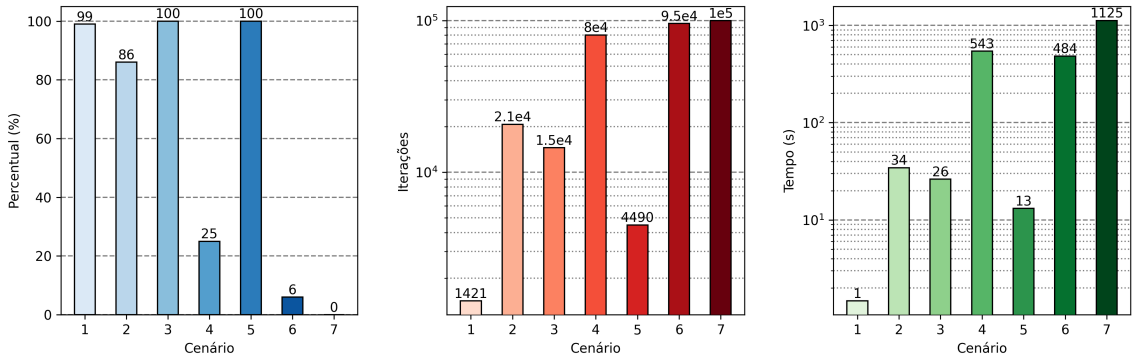
A Figura 40 apresenta os resultados para o caso 11. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) para a informação heurística da formiga, a diferença com o maior caminho (DMC) para avaliar o casamento de comprimento e evitar todos os caminhos das outras colônias (TCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 40: Resultados da execução do caso 11 para os sete cenários propostos.

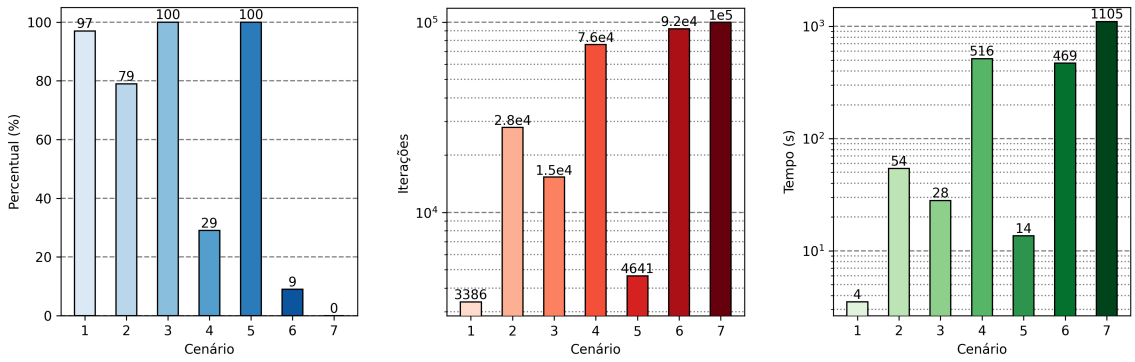
A Figura 41 apresenta os resultados para o caso 12. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) para a informação heurística da formiga, a diferença com o maior caminho (DMC) para avaliar o casamento de comprimento e evitar as trilhas de feromônios das outras colônias (SFO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 41: Resultados da execução do caso 12 para os sete cenários propostos.

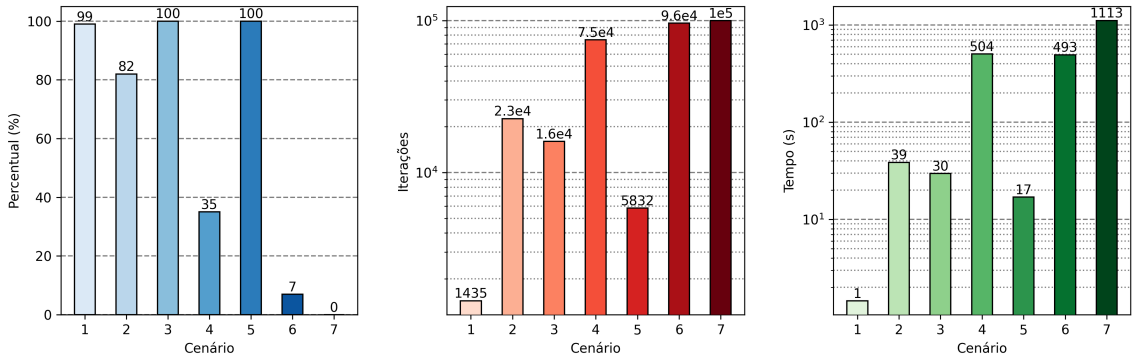
A Figura 42 apresenta os resultados para o caso 13. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) para a informação heurística da formiga, a soma das diferenças absolutas (SDA) para avaliar o casamento de comprimento e evitar os melhores caminhos das outras colônias (MCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 42: Resultados da execução do caso 13 para os sete cenários propostos.

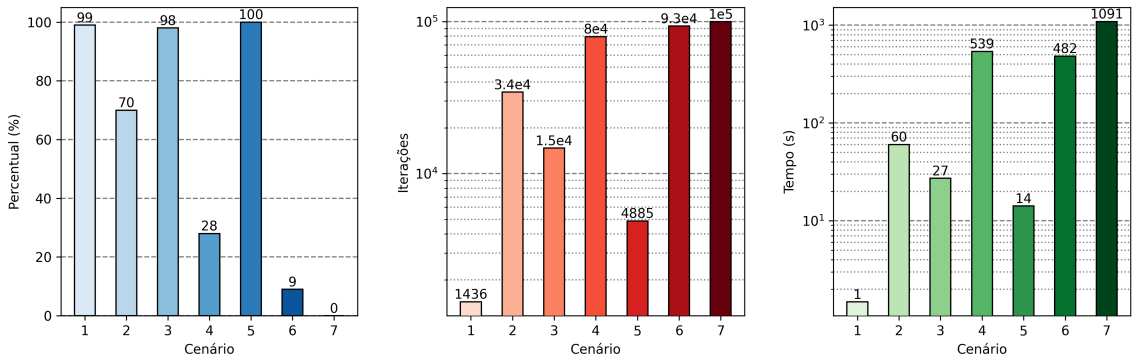
A Figura 43 apresenta os resultados para o caso 14. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) para a informação heurística da formiga, a soma das diferenças absolutas (SDA) para avaliar o casamento de comprimento e evitar todos os caminhos das outras colônias (TCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 43: Resultados da execução do caso 14 para os sete cenários propostos.

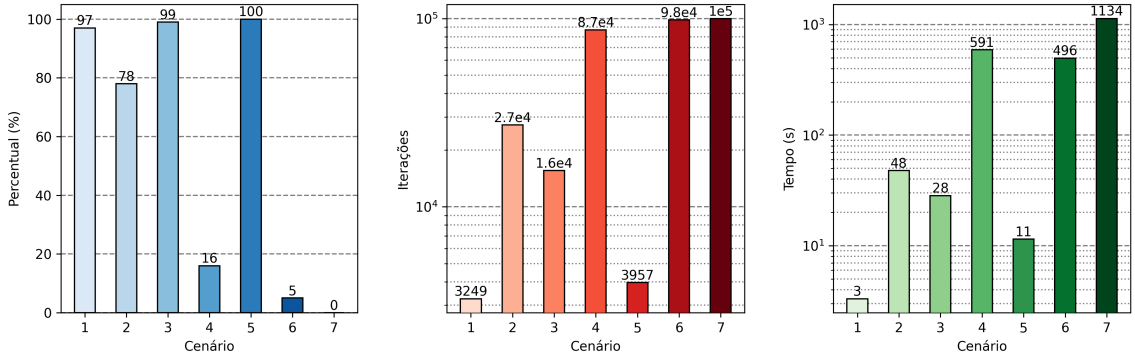
A Figura 44 apresenta os resultados para o caso 15. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) para a informação heurística da formiga, a soma das diferenças absolutas (SDA) para avaliar o casamento de comprimento e evitar as trilhas de feromônios das outras colônias (SFO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 44: Resultados da execução do caso 15 para os sete cenários propostos.

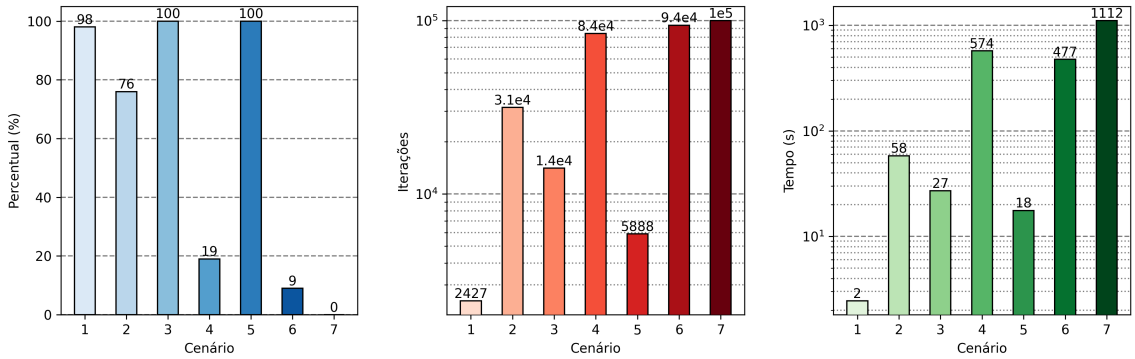
A Figura 45 apresenta os resultados para o caso 16. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) a soma das diferenças quadráticas (SDQ) para avaliar o casamento de comprimento e evitar os melhores caminhos das outras colônias (MCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 45: Resultados da execução do caso 16 para os sete cenários propostos.

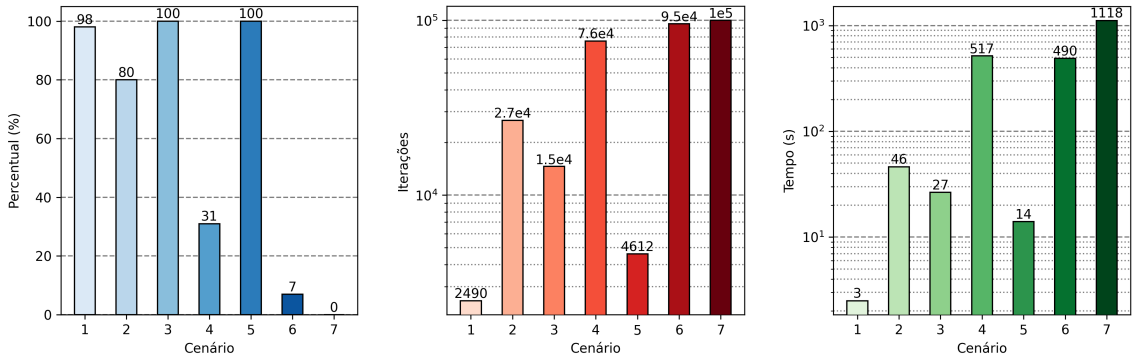
A Figura 46 apresenta os resultados para o caso 17. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) a soma das diferenças quadráticas (SDQ) para avaliar o casamento de comprimento e evitar todos os caminhos das outras colônias (TCO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 46: Resultados da execução do caso 17 para os sete cenários propostos.

A Figura 47 apresenta os resultados para o caso 18. As heurísticas utilizadas são o inverso da soma dos feromônios das outras colônias (ISF) a soma das diferenças quadráticas (SDQ) para avaliar o casamento de comprimento e evitar as trilhas de feromônios das outras colônias (SFO) para minimizar o cruzamento entre trilhas.



(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 47: Resultados da execução do caso 18 para os sete cenários propostos.

5.2.2 Avaliação dos Resultados

A Figura 48 apresenta o percentual acumulado de resultados ótimos por caso. Os dados evidenciam a influência predominante da informação heurística da formiga na obtenção de soluções ótimas. Para os casos de 1 a 9, que utilizam a heurística DEC, os percentuais variaram entre 175% e 193%. Já para os casos que utilizam a heurística ISF (casos de 10 a 18), os percentuais variaram entre 395% e 423%. Apesar do desempenho geral inferior dos casos que utilizam DEC, é notável que esses casos apresentaram melhor desempenho no cenário 2, onde todos alcançaram 100% ou quase isso de resultados ótimos. Por outro lado, os mesmos casos 1 a 9 não obtiveram resultados ótimos no cenário 1, considerado teoricamente o mais simples.

Esse comportamento pode ser explicado pela natureza da heurística DEC, que favorece posições próximas à posição final, priorizando excessivamente o objetivo de minimizar os comprimentos das trilhas. Isso causa uma convergência para caminhos mais curtos, muitas vezes em detrimento dos outros objetivos. A ausência de resultados ótimos no cenário 6 também pode ser atribuída a essa característica, já que esse cenário possui trilhas cujas posições iniciais e finais são muito próximas, levando o algoritmo a convergir para soluções subótimas que minimizam os comprimentos das trilhas, mas não atendem aos outros critérios.

A Figura 49 apresenta o número médio de iterações acumulado por caso. Para os casos de 1 a 9, que utilizam a heurística DEC, os valores acumulados variaram entre 527000 e 540000. Já para os casos de 10 a 18, que utilizam a heurística ISF, os valores acumulados variaram entre 317000 e 335000. Essa discrepância significativa pode ser

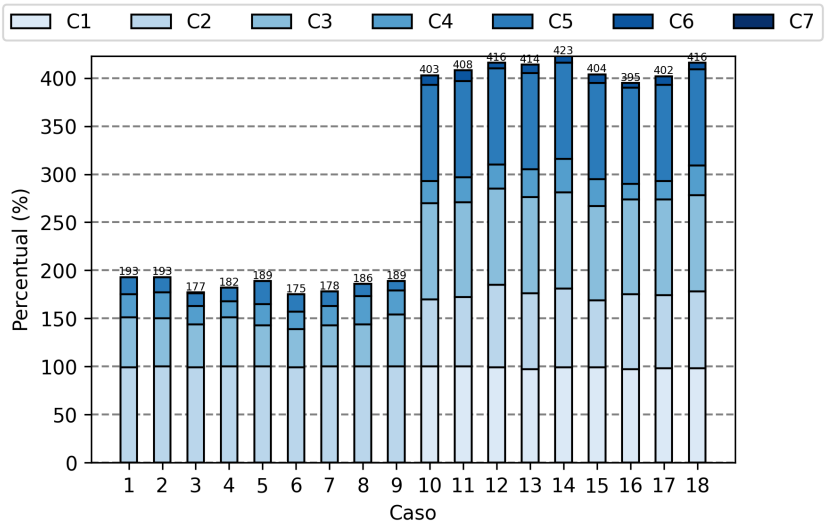


Figura 48: Percentual acumulado de resultados ótimos obtidos em todos os cenários para cada caso

atribuída, em grande parte, à tendência de convergência para resultados subótimos nos casos 1 a 9, o que leva o algoritmo a atingir o número máximo de iterações sem encontrar soluções ótimas com maior frequência.

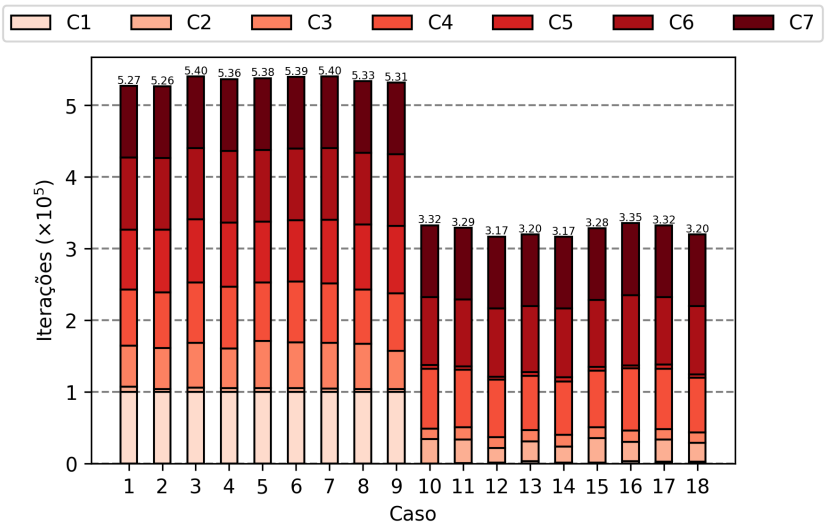


Figura 49: Número médio de iterações acumulado em todos os cenários para cada caso

A Figura 50 apresenta o tempo médio de execução acumulado por caso. Para os casos de 1 a 9, que utilizam a heurística DEC, os valores acumulados variaram entre 2868 e 3197 minutos. Observa-se que os tempos de execução dos casos 6, 7, 8 e 9 são consideravelmente mais elevados em comparação com os casos 1, 2, 3, 4 e 5, principalmente devido ao cenário 7, que apresenta a execução mais lenta nos quatro últimos casos. Para os demais cenários, os tempos de execução são semelhantes entre os casos 1 a 9. Já

para os casos de 10 a 18, que utilizam a heurística ISF, os valores acumulados variaram entre 2190 e 2312 minutos. Nesses casos, os tempos de execução nos cenários 1, 2 e 4 são significativamente inferiores em comparação aos casos que utilizam a heurística DEC. Além disso, parte da discrepância nos tempos acumulados pode ser atribuída à tendência dos casos 1 a 9 de convergirem para resultados subótimos, o que leva o algoritmo a atingir o número máximo de iterações com maior frequência, consequentemente prolongando os tempos de execução.

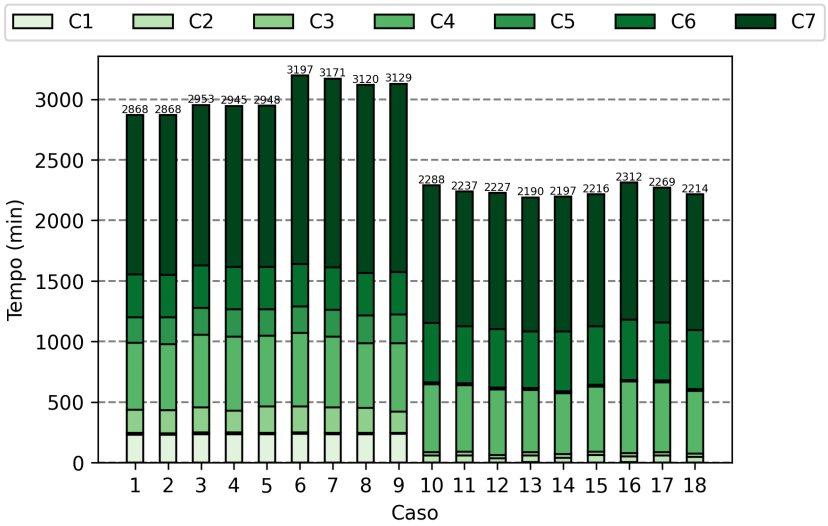
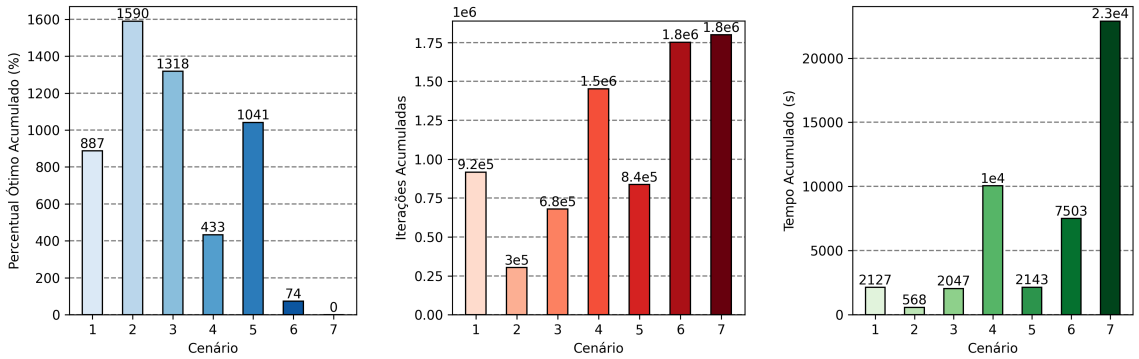


Figura 50: Tempo médio de execução acumulado em todos os cenários para cada caso

Considerando as métricas acumuladas, os casos de combinações de heurísticas que utilizam a estratégia ISF tem o desempenho melhor em 5 dos 7 cenários, obtendo mais resultados ótimos mais rapidamente e com menos iterações. Dentre estes, os três melhores são os casos 14, 12 e 18. É digno de nota que essa vantagem é por uma margem muito pequena.

A Figura 51 apresenta os resultados acumulados por cenário. Com base nesses resultados, o cenário 2 se destaca como o mais fácil de ser resolvido, apresentando o maior número de resultados ótimos e os menores valores para número de iterações e tempo de execução. Em ordem crescente de dificuldade, os próximos cenários são 3, 5, 1, 4 e 6. Dentre esses, o cenário 4 demandou significativamente mais iterações e tempo de execução. O cenário mais difícil foi o 7, que não obteve resultados ótimos e apresentou um tempo de execução elevado. A complexidade e a dificuldade de um cenário são influenciadas por diversos fatores, como o número de trilhas, a distância entre as posições iniciais e finais e a distribuição dessas posições no espaço de busca.



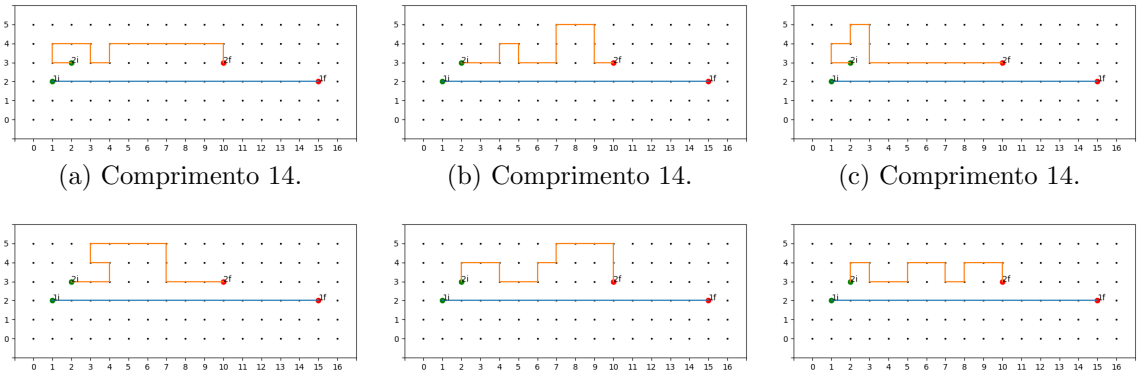
(a) Percentual de resultados ótimos. (b) Número médio de iterações. (c) Tempo médio de execução.

Figura 51: Resultados acumulados por cenário.

5.3 Trilhas Obtidas

Nesta seção, são exibidas as trilhas geradas com a aplicação do algoritmo desenvolvido nos cenários propostos. Todos os resultados foram obtidos utilizando o caso 14.

A Figura 52 apresenta seis resultados ótimos obtidos para o cenário 1. Em determinados casos, a pequena distância entre o inicio e o fim da trilha 2 acarreta na convergência para um resultado em que o caminho desta trilha é uma reta. Isso ocorre devido à forte influência da heurística na escolha do trajeto mais curto, o que leva o algoritmo a priorizar soluções que minimizem a distância, mesmo que isso possa comprometer os outros objetivos. No entanto, nos resultados para o caso 14, a trilha 2 utilizou o espaço adjacente para realizar o casamento de comprimento, destacando-se os padrões geométricos formados nas Figuras 52b e 52f, que são semelhantes a casos encontrados em PCIs reais.

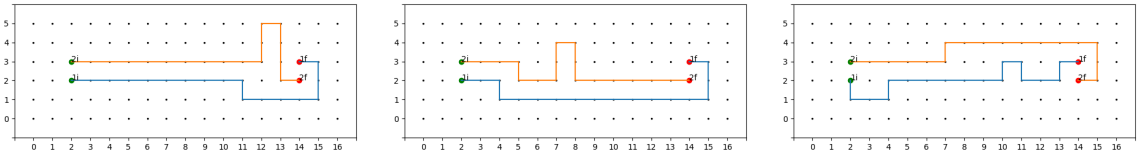


(a) Comprimento 14. (b) Comprimento 14. (c) Comprimento 14.

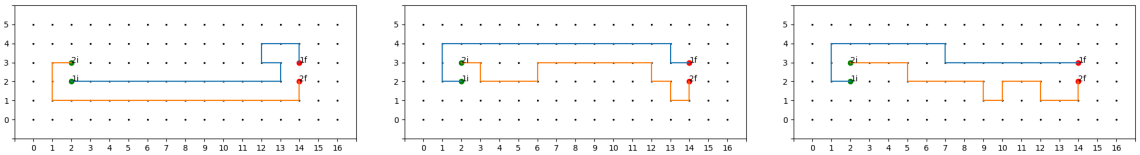
(d) Comprimento 14. (e) Comprimento 14. (f) Comprimento 14.

Figura 52: Seis resultados ótimos para o cenário 1 obtidos pelo algoritmo.

A Figura 53 apresenta seis resultados ótimos obtidos para o cenário 2, que se destacou por conter o maior número de soluções ótimas. Os resultados mostram que qualquer uma das trilhas pode determinar o comprimento mínimo, dependendo de qual trilha foi a primeira a encontrar um caminho que contorna a outra. Nas Figuras 53a, 53b, 53e e 53f, a trilha 1, representada pela cor azul, realiza o movimento de contorno e define o comprimento mínimo. Nos demais casos, essa função é desempenhada pela trilha 2, destacada na cor laranja.



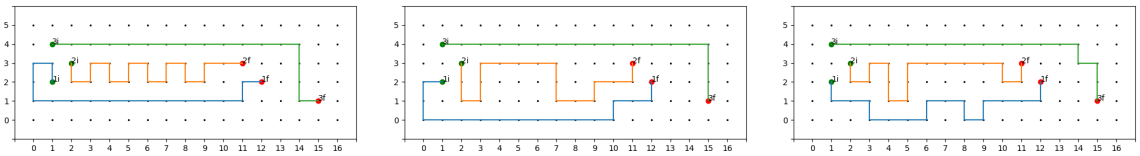
(a) Comprimento 17. (b) Comprimento 17. (c) Comprimento 17.



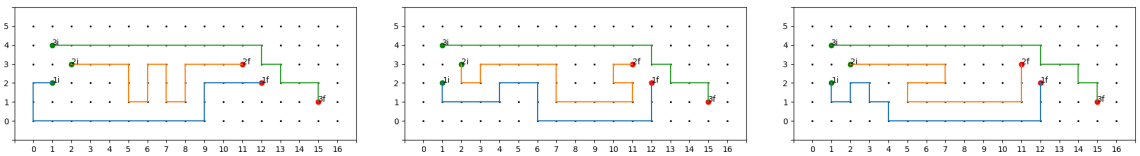
(d) Comprimento 17. (e) Comprimento 17. (f) Comprimento 17.

Figura 53: Seis resultados ótimos para o cenário 2 obtidos pelo algoritmo.

A Figura 54 apresenta seis resultados ótimos obtidos para o cenário 3. Destacam-se os padrões geométricos formados nas Figuras 54a e 54e, que são semelhantes a casos frequentemente encontrados em PCIs reais.



(a) Comprimento 17. (b) Comprimento 17. (c) Comprimento 17.

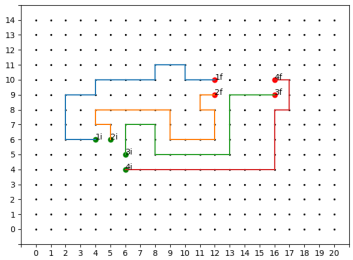


(d) Comprimento 17. (e) Comprimento 17. (f) Comprimento 17.

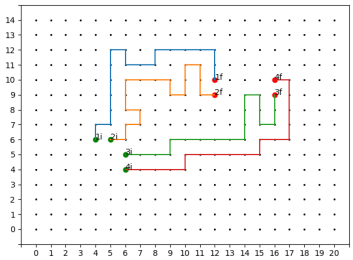
Figura 54: Seis resultados ótimos para o cenário 3 obtidos pelo algoritmo.

A Figura 55 apresenta seis resultados ótimos obtidos para o cenário 4. Este cenário foi o segundo mais demorado em termos de tempo de execução. A proximidade entre as trilhas e a disposição das posições iniciais e finais exige que todas as trilhas utilizem o

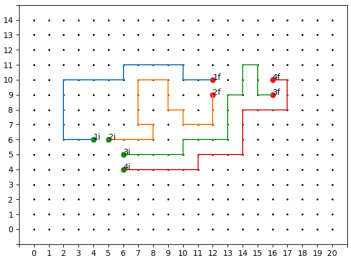
centro do espaço de busca, o que intensifica a competição por esse espaço. Como resultado, as extremidades da placa são pouco ou nada aproveitadas nos roteamentos.



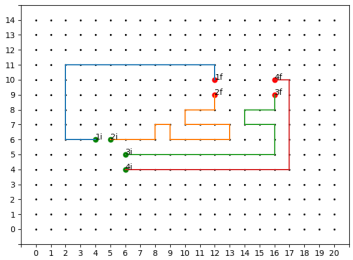
(a) Comprimento 18.



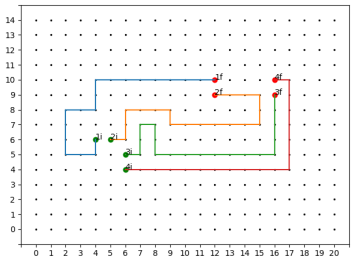
(b) Comprimento 18.



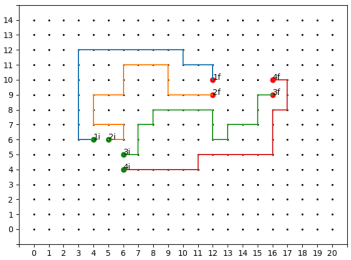
(c) Comprimento 18.



(d) Comprimento 18.



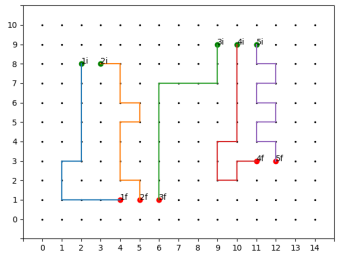
(e) Comprimento 18.



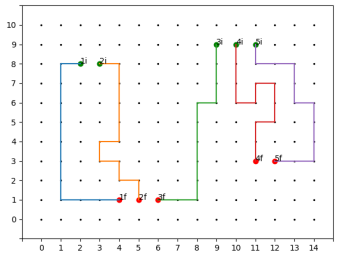
(f) Comprimento 18.

Figura 55: Seis resultados ótimos para o cenário 4 obtidos pelo algoritmo.

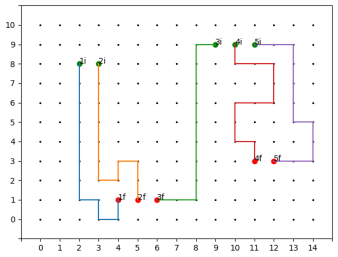
A Figura 56 apresenta seis resultados ótimos obtidos para o cenário 5. Neste cenário, as posições das cinco trilhas estão distribuídas em quatro regiões da placa, o que permite que as trilhas se espalhem de maneira equilibrada sobre o espaço de busca, com pouca necessidade de concorrência entre as colônias.



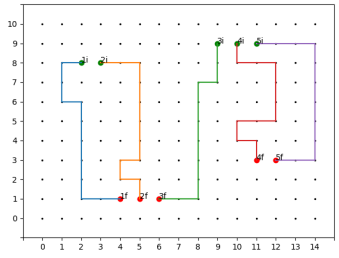
(a) Comprimento 11.



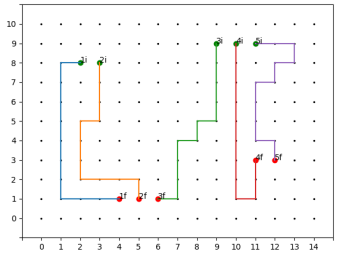
(b) Comprimento 11.



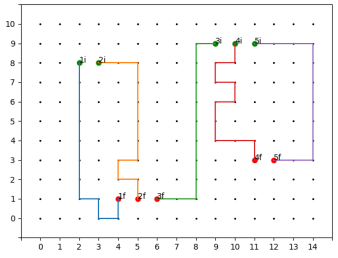
(c) Comprimento 11.



(d) Comprimento 11.



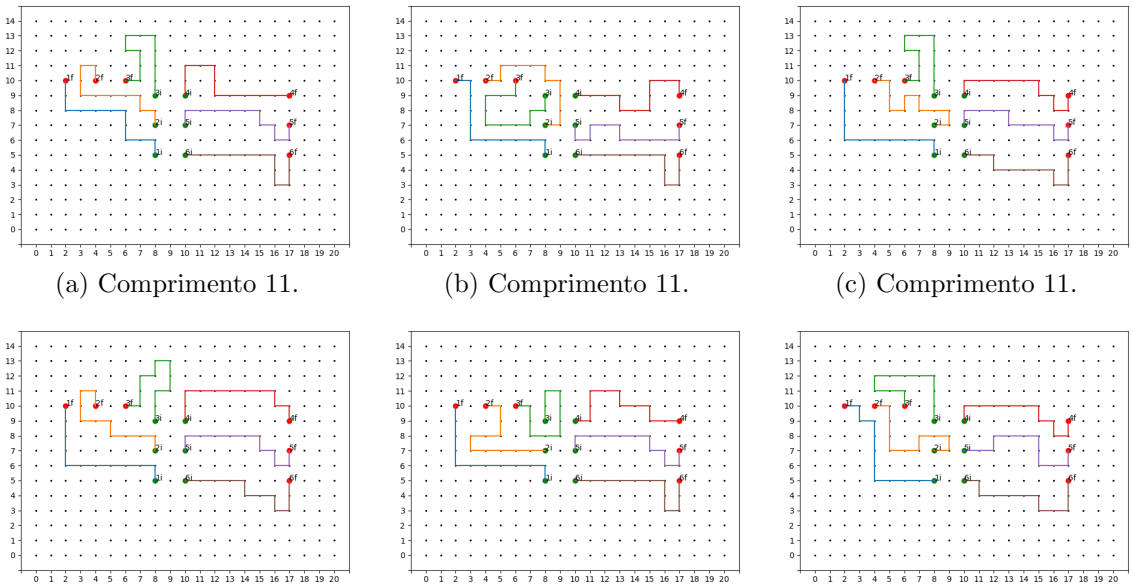
(e) Comprimento 11.



(f) Comprimento 11.

Figura 56: Seis resultados ótimos para o cenário 5 obtidos pelo algoritmo.

A Figura 57 apresenta seis resultados ótimos obtidos para o cenário 6. As trilhas à esquerda possuem distâncias mínimas diferentes entre si, sendo que a trilha 3, de cor verde, tem as posições iniciais e finais muito próximas, o que leva à convergência para o comprimento mínimo e reduz o número de ótimos encontrados. Já as trilhas à direita são equidistantes. A maior parte do roteamento ocorre no centro da placa, sendo que apenas a trilha 3 pode se aproximar das extremidades da placa, como é mostrado na Figura 57d.



(a) Comprimento 11. (b) Comprimento 11. (c) Comprimento 11.

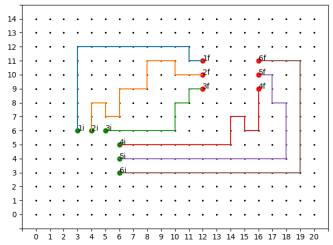
(d) Comprimento 11. (e) Comprimento 11. (f) Comprimento 11.

Figura 57: Seis resultados ótimos para o cenário 6 obtidos pelo algoritmo.

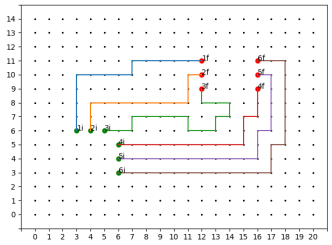
A Figura 58 apresenta seis resultados subótimos obtidos para o cenário 7. Este cenário não obteve resultados ótimos, no entanto, todos os resultados apresentados são razoáveis para circuitos tolerantes a variações no comprimento das trilhas. Também é evidente uma tendência de convergência para o comprimento mínimo, com destaque para a Figura 58d, que apresenta todas as trilhas com os comprimentos mínimos possíveis.

5.4 Comparação com Outros Trabalhos

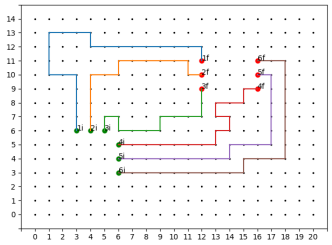
Esta seção apresenta os resultados obtidos em cenários provenientes de trabalhos externos. Esses cenários possuem a particularidade de incluir posições proibidas, uma característica ausente nos cenários propostos. Nos trabalhos originais, os objetivos consistem em encontrar roteamentos viáveis sem cruzamentos. O algoritmo proposto foi aplicado a es-



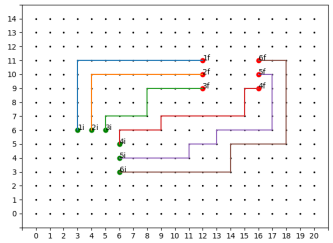
(a) Comprimentos 16, 16, 10, 16, 20 e 24.



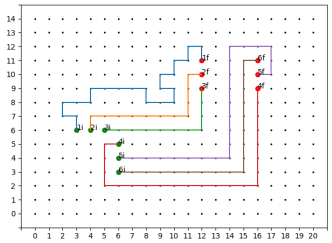
(b) Comprimentos 14, 12, 16, 14, 18 e 22.



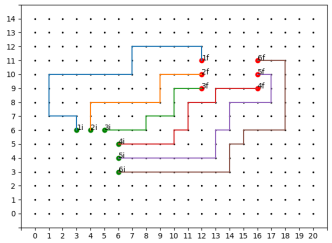
(c) Comprimentos 22, 14, 12, 18 e 22.



(d) Comprimentos 14, 12, 10, 14, 18 e 22.



(e) Comprimentos 22, 12, 10, 22, 18 e 22.

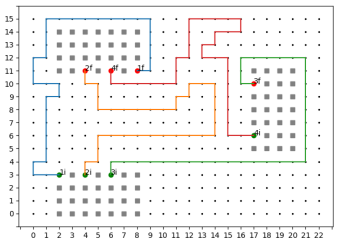


(f) Comprimento 22, 12, 10, 14, 18 e 22.

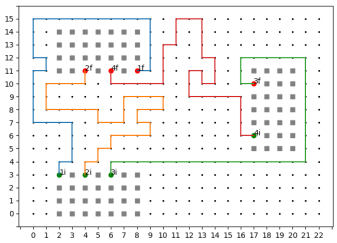
Figura 58: Seis resultados subótimos para o cenário 7 obtidos pelo algoritmo.

ses cenários externos, buscando atender aos três objetivos previamente definidos: obter trilhas com os comprimentos mínimos casados e sem cruzamentos.

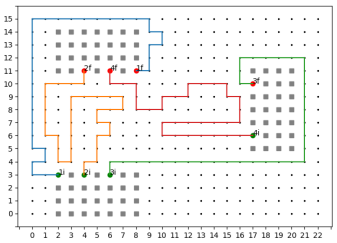
A Figura 59 apresenta seis resultados ótimos obtidos para o cenário E1. A trilha 3 de cor verde dita o comprimento mínimo com apenas uma rota viável, as trilhas 2 e 3 de cores amarela e vermelha competem pelo espaço central da PCI enquanto a trilha 1 de cor azul contorna pela o bloco superior de posições proibidas pela lateral da placa para realizar seu roteamento.



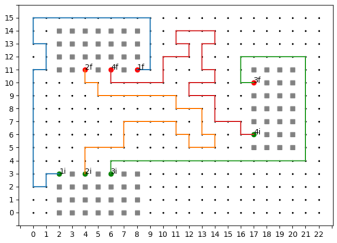
(a) Comprimento 32.



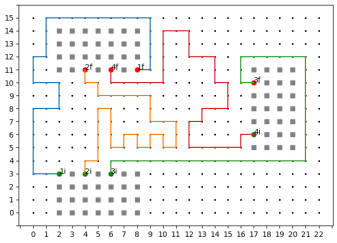
(b) Comprimento 32.



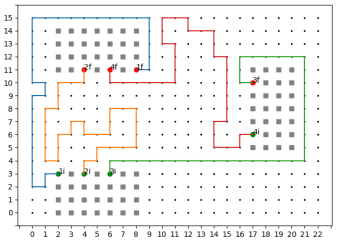
(c) Comprimento 32.



(d) Comprimento 32.



(e) Comprimento 32.



(f) Comprimento 32.

Figura 59: Seis resultados ótimos para o cenário E1 obtidos pelo algoritmo.

A Figura 60, por sua vez, apresenta a solução obtida no trabalho original, que utiliza o método de busca em árvore de Monte Carlo (MCTS) adaptado para realizar o roteamento de trilhas de PCI. A PCI é mapeada em uma grade onde cada pino e componente possui uma coordenada. O objetivo é encontrar um caminho ótimo na grade, do pino inicial ao final, evitando regiões proibidas. Durante o processo de roteamento, as ações que levam a cruzamentos com regiões são consideradas regiões proibidas na próxima iteração. Antes de realizar qualquer movimento no espaço de busca, o algoritmo verifica se o ponto de destino já está ocupado. Se estiver, a ação é invalidada, prevenindo cruzamentos. O método proposto foi comparado com o algoritmo de Lee (LEE, 1961) e obteve uma melhoria de 15.38% na eficiência computacional, com tempos de execução de 22 ms para o MCTS e 26 ms para o algoritmo de Lee, mantendo distâncias de roteamento semelhantes. Para obter convergência perto do ótimo de forma consistente, os autores utilizaram 700 iterações como critério de parada.(ZHANG et al., 2020).

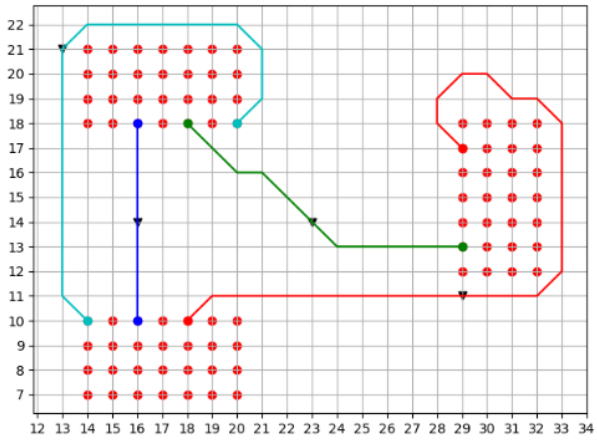
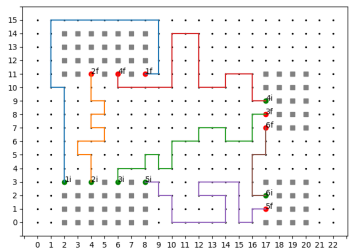


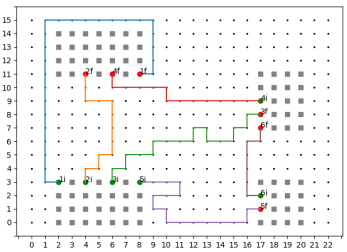
Figura 60: Solução do cenário E1 no trabalho de origem (ZHANG et al., 2020)

A Figura 61 apresenta seis resultados subótimos obtidos para o cenário E2. Apesar da ausência de resultados ótimos neste cenário, os resultados subótimos seriam considerados satisfatórios no contexto do trabalho original, demonstrando que a eficácia da técnica proposta é comparável à da técnica externa. Destaca-se neste cenário a tendência de convergência para comprimentos mínimos, característica também observada nos cenários propostos, onde o objetivo de minimizar as trilhas prevalece sobre os outros dois objetivos. Ainda assim, alguns padrões geométricos característicos do casamento de comprimento podem ser observados em particular nas Figuras 61a e 61e. A Figura 62 apresenta a solução obtida no trabalho original para este problema. Este cenário é do mesmo trabalho que

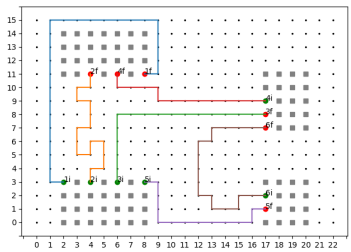
o cenário E1 e utiliza a mesma técnica de busca em árvore de Monte Carlo. No trabalho de origem, este cenário é utilizado como exemplo de um cenário mais complexo para comprovar a robustez do algoritmo MCTS proposto. No entanto, não são apresentadas as métricas de tempo nem comparação com o algoritmo de Lee para este cenário.



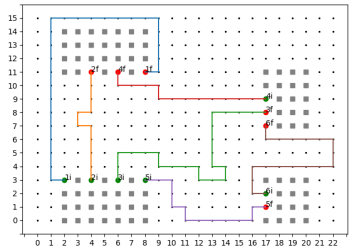
(a) Comprimentos 26, 14, 20, 23, 23 e 7.



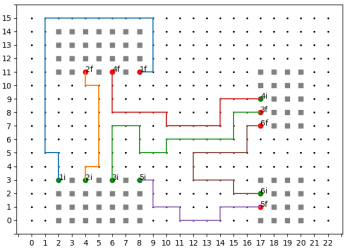
(b) Comprimentos 26, 12, 18, 13, 17 e 7.



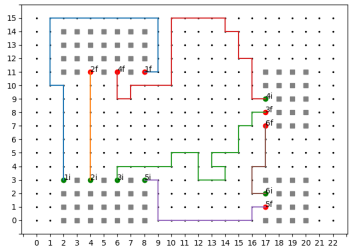
(c) Comprimentos 26, 16, 16, 13, 13 e 17.



(d) Comprimentos 26, 10, 22, 13, 13 e 17.



(e) Comprimentos 26, 10, 20, 17, 13 e 15.



(f) Comprimento 26, 8, 22, 25, 13 e 7 .

Figura 61: Seis resultados subótimos para o cenário E2 obtidos pelo algoritmo.

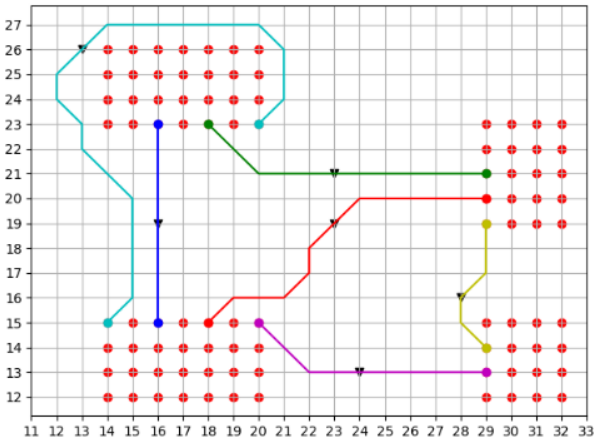
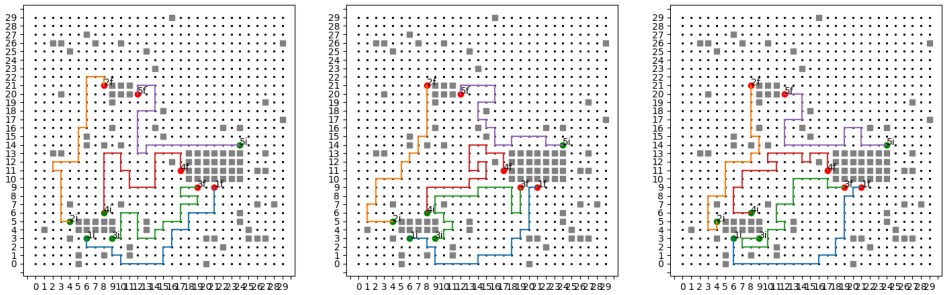


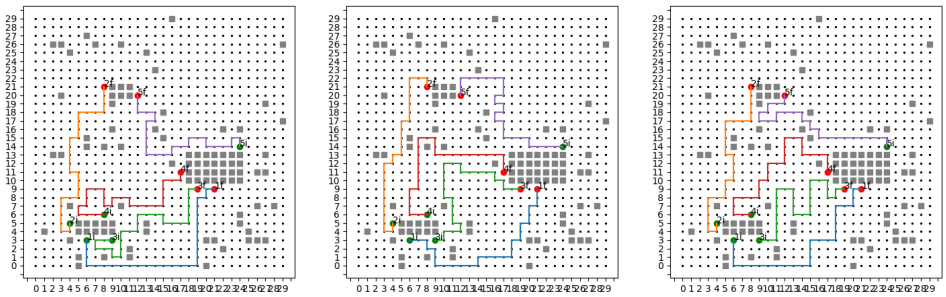
Figura 62: Solução do cenário E2 no trabalho de origem (ZHANG et al., 2020)

A Figura 63 apresenta seis resultados ótimos obtidos para o cenário E3, que registrou um número elevado de soluções ótimas. Este cenário se caracteriza pela presença de diversas posições proibidas espalhadas pelo espaço de busca. O algoritmo proposto demonstrou a capacidade de desviar eficientemente dessas posições proibidas e realizar o

roteamento atendendo aos três objetivos estabelecidos, evidenciando a adaptabilidade do ACO ao espaço de busca.



(a) Comprimento 32. (b) Comprimento 32. (c) Comprimento 32.



(d) Comprimento 32. (e) Comprimento 32. (f) Comprimento 32.

Figura 63: Seis resultados ótimos para o cenário E3 obtidos pelo algoritmo.

A Figura 64 apresenta a solução encontrada no trabalho original para esse problema, que combina busca em árvore de Monte Carlo e redes neurais. O artigo propõe uma abordagem baseada em MCTS para roteamento de circuitos, que trata o problema como um processo de decisão sequencial. A inovação está no uso de uma rede neural profunda para guiar a busca, eliminando a necessidade de conhecimento especializado do domínio ou heurísticas específicas. O sistema aprende a melhorar seu desempenho conforme executa mais roteamentos. Quando comparado com abordagens de roteamento sequencial, como o algoritmo de A* (HART; NILSSON; RAPHAEL, 1968) e o algoritmo de Lee (LEE, 1961), a solução baseada em MCTS com redes neurais demonstrou melhor desempenho em termos de solução de problemas mais complexos. Neste cenários, ambos algoritmos A* e Lee falham no roteamento. Os resultados mostram que 1000 iterações são suficientes para garantir 100% de convergência neste cenário, com 8.2% de trilhas redundantes. Com 5000 iterações o surgimento de trilhas redundantes cai para 4.5%. Nenhuma métrica de tempo é apresentada no trabalho (HE; BAO, 2020). É digno de nota que em (HE et al., 2022) os autores aplica sua técnica a outros cenários com resultados semelhantes.

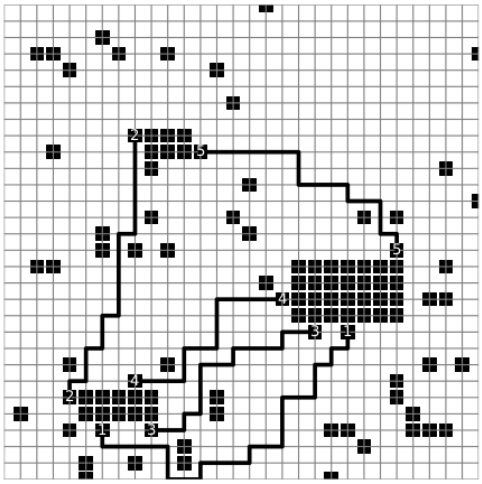


Figura 64: Solução do cenário E3 no trabalho de origem (HE; BAO, 2020)

A Tabela 5 apresenta os resultados de cem execuções do algoritmo proposto em cada cenário externo, utilizando a combinação de heurísticas 14. Foram obtidos 13% e 91% de resultados ótimos nos cenários E1 e E3, respectivamente, enquanto nenhum resultado ótimo foi alcançado no cenário E2. O cenário E3 apresentou o menor número médio de iterações e o menor tempo médio de execução, refletindo o elevado percentual de convergências. Apesar de apresentar mais convergências para resultados ótimos e envolver menos trilhas que o cenário E2, o cenário E1 registrou o tempo de execução mais elevado. Esse aumento no tempo foi influenciado pelo comprimento mínimo necessário maior em E1 em comparação ao E2, 32 e 26 respectivamente. Os comprimentos médios apresentados na quinta coluna demonstram um tendência do nosso algoritmo convergir para resultados com comprimentos menores do que aquele considerado o comprimento mínimo para o casamento de comprimento. Enfatizando a necessidade da escolha de hiper-parâmetros especificamente para as particularidades de cada cenério para balancear efetivamente os objetivos.

Tabela 5: Resultados para o cenários externos.

Cenário	Ótimos (%)	Iterações	Tempo de execução (s)	Comprimento
E1	13	93966 ± 18129.3	293.1 ± 69.8	25.1 ± 3.1
E2	0	100000	216.5 ± 13.8	16.7 ± 1.1
E3	91	23120 ± 28509.9	106.8 ± 1.7	26.3 ± 1.6

Comparando nossos resultados com os obtidos nos outros trabalhos, observamos uma menor taxa de convergência, com tempos de execução e número de iterações superiores. No entanto, é importante destacar que a inclusão do terceiro objetivo, o casamento

de comprimento, aumenta significativamente a complexidade do algoritmo. Além disso, ao contrário dos outros métodos, todos os objetivos são tratados simultaneamente, o que contribui para um processamento mais intenso. Apesar do desempenho aparentemente inferior, nossa técnica foi capaz de obter resultados satisfatórios, considerando nossas restrições e objetivos, mesmo que, em alguns casos, esses resultados tenham sido sub-ótimos, como demonstrado nas Figuras 59, 61 e 63. Finalmente, considerando os comprimentos das trilhas obtidos, fica claro que nossa técnica é capaz de atender satisfatoriamente à busca por trilhas com comprimento mínimo e sem cruzamentos, caso fosse calibrada para priorizar apenas esses objetivos.

5.5 Considerações Finais

Neste capítulo, foram apresentados e analisados os cenários utilizados para avaliar o desempenho do algoritmo proposto, abrangendo tanto os cenários criados especificamente para este trabalho quanto os cenários externos usados para comparações com métodos existentes. O algoritmo foi submetido a execuções repetidas em todos os cenários, demonstrando sua capacidade de alcançar os objetivos estabelecidos. Os resultados confirmam a eficiência do algoritmo em minimizar o comprimento das trilhas, atender aos requisitos de casamento de comprimento e evitar cruzamentos, mesmo sob restrições complexas, como posições proibidas nos cenários externos.

No próximo capítulo, são discutidas as conclusões mais significativas desta dissertação, ressaltando as contribuições do algoritmo para o roteamento em PCIs de alta frequência. Além disso, são apresentadas perspectivas para trabalhos futuros, com foco no aprimoramento do algoritmo e em sua aplicação a problemas mais complexos e situações práticas no contexto de projeto de circuitos eletrônicos.

Capítulo 6

CONCLUSÕES E TRABALHOS FUTUROS

ESTE capítulo apresenta na Seção 6.1 as principais conclusões acerca do algoritmo proposto para o roteamento de trilhas em PCIs com casamento de comprimento, destacando seu desempenho em relação aos cenários avaliados e comparando sua eficácia com outras técnicas. A Seção 6.2 descreve possíveis melhorias no algoritmo e sugestões para trabalhos futuros, com foco em ampliar sua aplicabilidade e aprimorar sua eficiência em cenários mais complexos.

6.1 Conclusões

Nesta dissertação, foi proposto um algoritmo baseado no ACO para o roteamento de trilhas em PCIs com restrições de comprimento, focando na minimização de trilhas, casamento de comprimento e evitando cruzamentos, um desafio típico em projetos de circuitos eletrônicos de alta frequência. O algoritmo foi testado em cenários propostos e cenários externos, permitindo uma avaliação abrangente de sua eficácia.

Os resultados obtidos indicam que o algoritmo proposto é capaz de fornecer soluções viáveis e eficientes, alcançando resultados ótimos em diversos cenários. A comparação com outras técnicas mostrou que o algoritmo apresenta desempenho competitivo, com destaque para sua capacidade de adaptação ao espaço de busca e para a minimização do comprimento das trilhas sem cruzamentos, atendendo aos objetivos estabelecidos. Os cenários 1, 3, 5 e E3, por exemplo, evidenciam um elevado percentual de convergência para soluções ótimas, o que demonstra a robustez do algoritmo.

Os casos de combinações de heurísticas foram avaliados utilizando o mesmo conjunto de hiperparâmetros. No entanto, em determinados casos, ficou evidente a preva-

lência de um objetivo em detrimento dos outros, com a busca pelo comprimento mínimo frequentemente se sobressaindo. Isso demonstra que o mesmo conjunto de hiperparâmetros não é adequado para todos os cenários, sugerindo que ajustes específicos para cada situação poderiam levar a resultados melhores. Encontrar conjuntos de hiperparâmetros adequados é um desafio comum em algoritmos baseados em sistemas multiagentes e inteligência de enxame. A escolha dos parâmetros foi um dos principais desafios na execução do algoritmo e na análise dos resultados, de modo que um conjunto típico e ingênuo foi selecionado para validar a técnica, mesmo que, em alguns casos, isso tenha prejudicado os resultados. O objetivo principal foi comprovar a viabilidade da técnica para realizar o roteamento em PCIs de circuitos de alta frequência com casamento de comprimento, e considera-se que essa meta foi alcançada.

Os padrões geométricos que surgem no roteamento são semelhantes àqueles encontrados em PCIs reais, o que valida a aplicabilidade do algoritmo em cenários práticos. A disposição das trilhas reflete configurações que normalmente surgem em projetos de circuitos eletrônicos, incluindo o respeito a restrições de espaço e as limitações impostas por posições proibidas. Além disso, o comportamento do algoritmo ao lidar com tais restrições e otimizar o roteamento de acordo com os objetivos definidos, como minimização de comprimento e prevenção de cruzamentos, indica que a técnica proposta tem potencial para ser aplicada no projeto de PCIs reais. Esse alinhamento com padrões reais reforça a relevância do algoritmo em contextos industriais e demonstra sua capacidade de gerar soluções viáveis para desafios típicos da área.

6.2 Trabalhos Futuros

Existem várias direções promissoras para o desenvolvimento futuro deste trabalho, com o objetivo de aprimorar a eficácia do algoritmo proposto e expandir suas aplicações. Uma melhoria significativa seria permitir que as formigas realizem movimentos diagonais, o que aumentaria as capacidades de exploração do algoritmo. Essa alteração poderia resultar em soluções mais eficientes, especialmente em cenários mais complexos, onde o espaço de busca é mais restrito.

Além disso, a realização de análises estatísticas sobre o desempenho do algoritmo com diferentes conjuntos de hiperparâmetros seria fundamental para entender seu com-

portamento sob diversas condições. Essas análises permitiriam ajustes no algoritmo, aprimorando sua robustez e adaptabilidade.

Testar o algoritmo em cenários mais complexos e desafiadores também seria essencial para avaliar sua escalabilidade e robustez. A aplicação em cenários de maior dimensão, com mais restrições e variáveis, ajudaria a determinar a eficácia do algoritmo em situações mais próximas de problemas reais.

Outra extensão importante seria adaptar o algoritmo para importar circuitos descritos por modelos SPICE, permitindo tratar malhas de circuitos em vez de simplesmente trilhas com pontos iniciais e finais. Essa modificação ampliaria a aplicabilidade do algoritmo, possibilitando seu uso em projetos mais realistas e abrangentes, típicos de circuitos eletrônicos de grande porte e de alta frequência.

O trabalho em alguns desses aprimoramentos já está em andamento. Esses esforços visam aumentar ainda mais a eficácia do algoritmo e ampliar sua aplicabilidade em tarefas de projeto de PCIs no contexto prático e em projetos de circuitos reais.

REFERÊNCIAS

ABBAS, A.; HAMZA, K.; MOHAMED, M. A. Cnc machining path planning optimization for circular hole patterns via a hybrid ant colony optimization approach. *Mechanical Engineering Research*, v. 4, 09 2014.

ALAYA, I.; GHEDIRA, K. Ant colony optimization for multi-objective optimization problems. *International Conference on Tools with Artificial Intelligence, ICTAI*, v. 1, p. 450 – 457, 11 2007.

ANAND, V.; SINGH, V.; LADWAL, V. K. Study on pcb designing problems and their solutions. In: *2019 International Conference on Power Electronics, Control and Automation (ICPECA)*. [S.l.: s.n.], 2019. p. 1–5.

ARKANI, M. et al. A fpga based time analyser for stochastic methods in experimental physics. *Instruments and Experimental Techniques*, v. 58, p. 350–358, 05 2015.

AYOB, M. N. et al. A particle swarm optimization approach for routing in VLSI. In: *2010 2nd International Conference on Computational Intelligence, Communication Systems and Networks*. Liverpool, UK: IEEE, 2010. p. 49–53.

BADRIYAH, T.; SETYORINI, F.; YULIAWAN, N. The implementation of genetic algorithm and routing Lee for PCB design optimization. In: *2016 International Conference on Informatics and Computing (ICIC)*. Mataram, Indonesia: IEEE, 2016. p. 148–153.

BERLIER, J. A. *A Parallel Genetic Algorithm for Placement and Routing on Cloud Computing Platforms*. Dissertação (Mestrado) — Virginia Commonwealth University, 2011.

BETZ, V.; ROSE, J. Directional bias and non-uniformity in FPGA global routing architectures. In: *Proceedings of International Conference on Computer Aided Design*. San Jose, CA, USA: IEEE, 1996. p. 652–659.

BOGATIN, E. *Signal and Power Integrity - Simplified*. 2nd. ed. USA: Prentice Hall PTR, 2009. ISBN 0132349795.

BULLNHEIMER, B.; HARTL, R.; STRAUSS, C. A new rank based version of the ant system - a computational study. *Central European Journal of Operations Research*, v. 7, p. 25–38, 01 1999.

CHU, C.; WONG, Y.-C. FLUTE: Fast lookup table based rectilinear Steiner minimal tree algorithm for VLSI design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, IEEE, v. 27, n. 1, p. 70–83, 2008.

CONG A. B. KAHNG, C. K. K. J.; TSAO, C. W. A. Bounded-skew clock and Steiner routing. *ACM Transactions on Design Automation of Electronic Systems*, Association for Computing Machinery, New York, NY, USA, 1998.

DIJKSTRA, E. W. A note on two problems in connexion with graphs. *Numerische Mathematik*, Springer, v. 1, n. 1, p. 269–271, 1959.

DORIGO, M.; GAMBARDELLA, L. M. A study of some properties of ant-q. *Lecture Notes in Computer Science*, 10 2000.

DORINGO, M.; STÜTZLE, T. Ant colony optimization for the traveling salesman problem. In: *Ant Colony Optimization*. [S.l.]: The MIT Press, 2004. cap. 3, p. 65–120.

EVERETT, I. H. Generalized lagrange multiplier method for solving problems of optimum allocation of resources. *Operations Research*, v. 11, n. 3, p. 399–417, 1963.

GELLY, S.; SILVER, D. Monte-Carlo tree search and rapid action value estimation in computer Go. *Artificial Intelligence*, Elsevier, v. 175, n. 11, p. 1856–1875, 2011.

HART, P. E.; NILSSON, N. J.; RAPHAEL, B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics*, v. 4, n. 2, p. 100–107, 1968.

HATEM, N. et al. A novel integrating between tool path optimization using an aco algorithm and interpreter for open architecture cnc system. *Expert Systems with Applications*, v. 178, p. 114988, 2021.

HE, Y.; BAO, F. S. *Circuit Routing Using Monte Carlo Tree Search and Deep Neural Networks*. 2020. Disponível em: <https://arxiv.org/abs/2006.13607>.

HE, Y. et al. Circuit routing using Monte Carlo Tree Search and Deep Reinforcement Learning. In: *2022 International Symposium on VLSI Design, Automation and Test (VLSI-DAT)*. Hsinchu, Taiwan: IEEE, 2022. p. 1–5.

HU, J.; ROY, J. A.; MARKOV, I. L. Completing high-quality global routes. In: *Proceedings of the 19th International Symposium on Physical Design*. San Francisco, California, USA: Association for Computing Machinery, 2010. p. 35–41.

KAHNG, A. B. et al. Detailed routing. In: *VLSI Physical Design: From Graph Partitioning to Timing Closure*. [S.l.]: Springer, 2011. cap. 6, p. 129–166.

KAHNG, A. B. et al. Global routing. In: *VLSI Physical Design: From Graph Partitioning to Timing Closure*. [S.l.]: Springer, 2011. cap. 5, p. 129–166.

KAHNG, A. B. et al. Specialized routing. In: *VLSI Physical Design: From Graph Partitioning to Timing Closure*. [S.l.]: Springer, 2011. cap. 7, p. 129–166.

KENNEDY, J.; EBERHART, R. A discrete binary version of the particle swarm algorithm. In: *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*. Orlando, FL, USA: IEEE, 1997. v. 5, p. 4104–4108.

KIM, M. et al. Reinforcement Learning-based auto-router considering signal integrity. In: *2020 IEEE 29th Conference on Electrical Performance of Electronic Packaging and Systems (EPEPS)*. San Jose, CA, USA: IEEE, 2020. p. 1–3.

KUMTONG, W.; DANKLANG, P.; SRIBORRIRUX, W. Pin set sequence selection guideline routing for printed circuit board routing. In: *2015 7th International Conference on Knowledge and Smart Technology (KST)*. Chonburi, Thailand: IEEE, 2015. p. 126–130.

LEE, C. Y. An algorithm for path connections and its applications. *IRE Transactions on Electronic Computers*, EC-10, n. 3, p. 346–365, 1961.

LEE, D.; MARKOV, I. L. Contango: Integrated optimization of soc clock networks. In: *Conference on Design, Automation and Test in Europe*. Dresden, Germany: European Design and Automation Association, 2010. p. 1468–1473.

LEI, S.-I.; MAK, W.-K. Optimizing pin assignment and escape routing for blind-via-based PCBs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, IEEE, v. 35, n. 2, p. 246–259, 2016.

LIAO, H. et al. A deep reinforcement learning approach for global routing. *Journal of Mechanical Design*, ASME, v. 142, n. 6, 2019.

LÓPEZ-IBÁÑEZ, M.; STÜTZLE, T. Automatic configuration of multi-objective aco algorithms. In: *Swarm Intelligence*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 95–106.

LUO, L. et al. B-escape: A simultaneous escape routing algorithm based on boundary routing. In: *Proceedings of the 19th International Symposium on Physical Design*. San Francisco, California, USA: Association for Computing Machinery, 2010. p. 19–25.

MA, Q.; WONG, M. D. F. NP-completeness and an approximation algorithm for rectangle escape problem with application to PCB routing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, IEEE, v. 31, n. 9, p. 1356–1365, 2012.

MALAVASI, E.; SANGIOVANNI-VINCENTELLI, A. Area routing for analog layout. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, IEEE, v. 12, n. 8, p. 1186–1197, 1993.

MIAO, C. et al. Path planning optimization of indoor mobile robot based on adaptive ant colony algorithm. *Computers & Industrial Engineering*, v. 156, p. 107230, 2021.

NATH, S.; GHOSH, S.; SARKAR, S. K. A novel approach to discrete particle swarm optimization for efficient routing in VLSI design. In: *2015 4th International Conference on Reliability, Infocom Technologies and Optimization (ICRITO) (Trends and Future Directions)*. Noida, India: IEEE, 2015. p. 1–4.

OZDAL, M. M. *Routing algorithms for high-performance VLSI packaging*. Tese (Doutorado) — University of Illinois at Urbana-Champaign, 2005.

PANDIARAJ, K.; SIVAKUMAR, P.; SRIDEVI, R. Minimization of wirelength in 3D IC routing by using differential evolution algorithm. In: *2017 IEEE International Conference on Electrical, Instrumentation and Communication Engineering (ICEICE)*. Karur, India: IEEE, 2017. p. 1–5.

SIERRA CIRCUITS. *Signal Integrity eBook*. Sunnyvale, California, 2023. EBook consultado online. Disponível em: <https://www.protoexpress.com/pcb-design-guides/signal-integrity-ebook>.

SINHA, S. N. B.; BAISHNAB, K. L. A hybrid RFD-ACO approach for routing optimization in VLSI physical design. In: *Journal of Information and Optimization Sciences*. [S.l.: s.n.], 2017.

STORN, R.; PRICE, K. Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, Springer, v. 11, p. 341–359, 1997.

STÜTZLE, T. Local search algorithms for combinatorial problems - analysis, improvements, and new applications. In: *DISKI*. [S.l.: s.n.], 1999.

TEXAS INSTRUMENTS. *High-Speed Layout Guidelines*. [S.l.], 2006. Revised in 2017.

TEXAS INSTRUMENTS. *High-Speed Interface Layout Guidelines*. [S.l.], 2014. Revised in 2017.

WANG, R.; SHI, R.; CHENG, C. K. Layer minimization of escape routing in area array packaging. In: *2006 IEEE/ACM International Conference on Computer Aided Design*. San Jose, CA, USA: IEEE, 2006. p. 815–819.

XU, G. et al. Redundant-via enhanced maze routing for yield improvement. In: *Asia and South Pacific Design Automation Conference, 2005.* Shanghai, China: IEEE, 2005. v. 2, p. 1148–1151 Vol. 2.

YAN, T.; WONG, M. D. F. A correct network flow model for escape routing. In: *2009 46th ACM/IEEE Design Automation Conference*. San Francisco, CA, USA: IEEE, 2009. p. 332–335.

YUSOF, Z. M. et al. A two-step binary particle swarm optimization approach for routing in VLSI with iterative RLC delay model. In: *2011 Third International Conference on Computational Intelligence, Modelling & Simulation*. Langkawi, Malaysia: IEEE, 2011. p. 63–67.

ZHANG, C. et al. A hierarchy MCTS algorithm for the automated PCB routing. In: *2020 IEEE 16th International Conference on Control and Automation (ICCA)*. Singapore: IEEE, 2020. p. 1366–1371.

ZHANG, H.; KROOSWYK, S.; OU, J. Chapter 2 - pcb design for signal integrity. In: *High Speed Digital Design*. Boston: Morgan Kaufmann, 2015. p. 27–115.

ZHANG, H. et al. A hybrid ant colony optimization algorithm for a multi-objective vehicle routing problem with flexible time windows. *Information Sciences*, v. 490, p. 166–190, 2019.

Apêndice A

RESULTADOS PARA OS CENÁRIOS PROPOSTOS

Este apêndice apresenta os resultados obtidos para as dezoito combinações de heurísticas apresentadas na Tabela 3 aplicados aos sete cenários de teste. Para avaliar as combinações de heurísticas, seis métricas foram consideradas: percentual de resultados ótimos (PRO), percentual de resultados sem cruzamentos e com casamento de comprimento (RCC), percentual de resultados com cruzamentos (RCR), tempo médio de execução (TME), número médio de iterações (NMI) e comprimento médio das trilhas (CMT). As três últimas métricas são apresentadas junto com o desvio padrão das medições.

Tabela A.1: Resultados das combinações das heurísticas para o cenário 1

Caso	PRO (%)	RCC (%)	RCR (%)	TME (ms)	NMI	CMT
1	0	2	0	230.65 ± 15.3	100000 ± 0	14.52 ± 2.23
2	0	1	0	232.13 ± 16.78	100000 ± 0	14.78 ± 2.25
3	0	1	0	233.97 ± 21.9	100000 ± 0	14.66 ± 2.35
4	0	3	0	235.26 ± 24.87	100000 ± 0	14.69 ± 2.33
5	0	1	0	234.49 ± 19.7	100000 ± 0	14.75 ± 2.21
6	0	3	0	236.97 ± 18.12	100000 ± 0	14.61 ± 2.35
7	0	4	0	234.55 ± 17.47	100000 ± 0	14.63 ± 2.26
8	0	1	0	233.98 ± 15.25	100000 ± 0	14.64 ± 2.23
9	0	1	0	237.78 ± 15.21	100000 ± 0	14.66 ± 2.37
10	100	100	0	0.36 ± 0.93	345.44 ± 710.95	14 ± 0
11	100	100	0	0.73 ± 0.62	618.94 ± 543.87	14 ± 0
12	99	99	0	1.47 ± 10.46	1421.08 ± 9979.67	14.02 ± 0.28
13	97	97	0	3.52 ± 0.59	3386.32 ± 486.75	14 ± 0
14	99	99	0	1.46 ± 0.67	1435.15 ± 497.63	14 ± 0
15	99	99	0	1.49 ± 18.55	1435.74 ± 19620.2	14.01 ± 0.43
16	97	97	0	3.32 ± 10.02	3249.27 ± 9972.88	14.02 ± 0.28
17	98	98	0	2.46 ± 0.8	2427.07 ± 600.52	14 ± 0
18	98	98	0	2.5 ± 12.95	2489.82 ± 14028.94	13.98 ± 0.45

Tabela A.2: Resultados das combinações das heurísticas para o cenário 2

Caso	PRO (%)	RCC (%)	RCR (%)	TME (ms)	NMI	CMT
1	99	100	0	16.83 ± 28.47	7034.61 ± 5326.81	17 ± 0
2	100	100	0	9.43 ± 19.78	3749.37 ± 10408.22	17 ± 0.2
3	99	99	0	14.49 ± 12.63	5896.72 ± 4735.44	17 ± 0
4	100	100	0	12.78 ± 14.21	5035.15 ± 5139.93	17 ± 0
5	100	100	0	13.01 ± 23.42	5010.43 ± 10772.07	17.02 ± 0.2
6	99	99	1	12.62 ± 10.28	5052.17 ± 3984.33	17 ± 0
7	100	100	0	12.71 ± 20	4997.65 ± 10460.2	17.02 ± 0.2
8	100	100	0	10.75 ± 10.6	4173.06 ± 4059.58	17 ± 0
9	100	100	0	9.54 ± 23.15	3730.43 ± 10175.35	17 ± 0
10	70	70	24	58.91 ± 65.43	33867.4 ± 37796.03	16.51 ± 1.13
11	72	72	23	57.74 ± 81.22	33186.83 ± 41229.25	16.39 ± 1.27
12	86	86	13	34.42 ± 79.82	20695.12 ± 42717.49	16.25 ± 1.41
13	79	79	18	54.16 ± 68.33	27954.97 ± 38643.75	16.42 ± 1.26
14	82	82	15	38.6 ± 80.71	22609.85 ± 42950.5	16.14 ± 1.51
15	70	70	25	60.2 ± 80.95	34281.06 ± 44326.91	16.03 ± 1.57
16	78	78	19	47.84 ± 68.3	27170.26 ± 38318.75	16.45 ± 1.22
17	76	76	20	58.05 ± 70.49	31494.81 ± 39263.52	16.37 ± 1.35
18	80	80	16	46.29 ± 66.55	26751.64 ± 37852.4	16.48 ± 1.24

Tabela A.3: Resultados das combinações das heurísticas para o cenário 3

Caso	PRO (%)	RCC (%)	RCR (%)	TME (ms)	NMI	CMT
1	52	52	2	190.9 ± 173.77	57286.45 ± 46791.2	16.46 ± 2.37
2	50	50	4	189.67 ± 166.59	57276.5 ± 45267.73	16.63 ± 2.4
3	45	46	5	208.03 ± 164.76	62618.29 ± 44494.96	16.61 ± 2.31
4	51	51	2	182.44 ± 166.26	55413.45 ± 45397.11	16.52 ± 2.51
5	43	43	3	217.73 ± 170.75	66262.23 ± 44486.39	16.59 ± 2.4
6	40	40	1	214.05 ± 167.96	64096.99 ± 45340.03	16.61 ± 2.31
7	43	46	1	209.68 ± 163.23	63270.35 ± 43971.08	16.52 ± 2.5
8	44	44	4	208.98 ± 164.31	62957.6 ± 44379.55	16.59 ± 2.29
9	54	54	3	174.84 ± 171.8	53733.66 ± 45469.91	16.61 ± 2.4
10	100	100	0	26.97 ± 26.37	14884.41 ± 15265.9	17 ± 0
11	99	99	1	30.98 ± 25.05	17024.25 ± 13944.88	17 ± 0
12	100	100	0	26.24 ± 29.2	14514.03 ± 15669.99	16.95 ± 0.58
13	100	100	0	28.11 ± 26.65	15323.48 ± 14347.76	17 ± 0
14	100	100	0	29.6 ± 24.37	15998.67 ± 14342.09	17 ± 0
15	98	98	0	27.12 ± 21.12	14732.98 ± 11611.52	17 ± 0
16	99	99	0	28.43 ± 31.77	15578.66 ± 16834.85	17 ± 0
17	100	100	0	27.01 ± 27.07	14110.27 ± 15084.94	17 ± 0
18	100	100	0	26.59 ± 23.76	14562.82 ± 13830.67	17 ± 0

Tabela A.4: Resultados das combinações das heurísticas para o cenário 4

Caso	PRO (%)	RCC (%)	RCR (%)	TME (ms)	NMI	CMT
1	24	24	0	550.27 ± 311.09	78240.79 ± 38061.37	16.82 ± 1.73
2	27	27	0	547.53 ± 265.27	77629.25 ± 30988.45	16.61 ± 1.73
3	19	19	0	598.72 ± 286.03	84285.63 ± 35122.78	16.71 ± 1.67
4	17	17	0	609.2 ± 275.61	86414.39 ± 33587.15	16.82 ± 1.62
5	22	22	0	582.1 ± 297.68	81636.23 ± 36654.37	16.71 ± 1.68
6	18	18	0	609.35 ± 309.98	84805.25 ± 38284.77	16.88 ± 1.6
7	20	20	0	582.35 ± 301.76	82958.55 ± 37147.96	16.8 ± 1.64
8	29	29	0	531.63 ± 291.9	75494.26 ± 36279.04	16.83 ± 1.68
9	25	25	0	563.46 ± 265.9	79911.77 ± 32754.57	16.81 ± 1.66
10	23	23	27	561.56 ± 194	83086.32 ± 27720.41	15.24 ± 2.69
11	26	26	33	548.96 ± 245.36	80140.31 ± 35493.82	15.4 ± 2.75
12	25	25	28	543.01 ± 248.24	80258.9 ± 36294.69	15.36 ± 2.71
13	29	29	21	515.96 ± 217.41	76085.71 ± 31243.91	15.23 ± 2.74
14	35	35	23	503.67 ± 230.55	74704.48 ± 32698.59	15.22 ± 2.77
15	28	28	19	539.37 ± 237.41	79507.4 ± 34430.93	15.45 ± 2.75
16	16	16	36	590.8 ± 254.29	87088.06 ± 36381.89	15.52 ± 2.74
17	19	19	30	574.38 ± 264.95	84089.2 ± 37407.63	15.69 ± 2.76
18	31	31	24	516.76 ± 239.31	76009.4 ± 33258.78	15.41 ± 2.71

Tabela A.5: Resultados das combinações das heurísticas para o cenário 5

Caso	PRO (%)	RCC (%)	RCR (%)	TME (ms)	NMI	CMT
1	18	18	2	212.68 ± 69.22	84074.51 ± 26397.64	9.84 ± 1.56
2	16	16	6	222.74 ± 64.4	87674.56 ± 24322.77	9.89 ± 1.55
3	13	13	6	223.45 ± 54.2	87840.99 ± 20602.43	9.82 ± 1.56
4	14	14	2	226.86 ± 78.99	89469.44 ± 29954.06	9.92 ± 1.54
5	24	24	4	216.85 ± 90.14	84719.54 ± 33963.44	9.96 ± 1.54
6	18	18	3	217.3 ± 89.39	85299.24 ± 34127.69	9.97 ± 1.52
7	15	15	7	224.24 ± 92.59	88944.43 ± 35324.11	9.94 ± 1.53
8	13	13	3	228.92 ± 72.5	90738.19 ± 27255.15	9.89 ± 1.55
9	10	10	6	238.95 ± 74.72	93876.42 ± 28338.29	9.96 ± 1.54
10	100	100	0	15.44 ± 14.23	5110.78 ± 4504.95	11 ± 0
11	100	100	0	14.5 ± 15.01	4781.04 ± 4840.2	11 ± 0
12	100	100	0	13.12 ± 14.08	4489.89 ± 4711.44	11 ± 0
13	100	100	0	13.61 ± 14.94	4640.99 ± 4852.98	11 ± 0
14	100	100	0	16.94 ± 15.37	5831.62 ± 4964.52	11 ± 0
15	100	100	0	14.21 ± 12	4884.69 ± 3815.06	11 ± 0
16	100	100	0	11.45 ± 10.47	3956.98 ± 3370.81	11 ± 0
17	100	100	0	17.57 ± 12.81	5887.77 ± 4118.16	11 ± 0
18	100	100	0	14.07 ± 16.21	4611.74 ± 5393.02	11 ± 0

Tabela A.6: Resultados das combinações das heurísticas para o cenário 6

Caso	PRO (%)	RCC (%)	RCR (%)	TME (ms)	NMI	CMT
1	0	0	0	351.65 ± 29.97	100000 ± 0	8.75 ± 2.9
2	0	0	0	349.15 ± 32.09	100000 ± 0	8.79 ± 2.94
3	1	1	0	348.82 ± 30.56	99213.7 ± 0	8.82 ± 2.94
4	0	0	0	349.21 ± 30.6	100000 ± 0	8.74 ± 2.91
5	0	0	0	350.21 ± 30.35	100000 ± 0	8.74 ± 2.93
6	0	0	0	350.21 ± 29.44	100000 ± 0	8.84 ± 2.91
7	0	0	0	350.19 ± 30.91	100000 ± 0	8.71 ± 2.9
8	0	0	0	349.9 ± 30.1	100000 ± 0	8.78 ± 2.94
9	0	0	0	349.6 ± 30.08	100000 ± 0	8.76 ± 2.92
10	10	10	0	489.64 ± 99.9	95141.22 ± 17614.78	8.43 ± 2.73
11	11	11	0	472.42 ± 112.5	93125.54 ± 19782.94	8.63 ± 2.64
12	6	6	0	484.15 ± 100.63	95392.58 ± 17933.05	8.41 ± 2.69
13	9	9	0	468.85 ± 130.46	92261.23 ± 24647.97	8.55 ± 2.68
14	7	7	0	492.94 ± 95.02	95934.12 ± 16629.71	8.51 ± 2.68
15	9	9	0	482.4 ± 109.82	93186.97 ± 19749.31	8.49 ± 2.66
16	5	5	0	496.44 ± 126.57	98160.92 ± 23734.19	8.51 ± 2.63
17	9	9	0	477.41 ± 103.77	93947.27 ± 19339.13	8.61 ± 2.65
18	7	7	0	489.81 ± 68.63	95277.48 ± 10987.31	8.38 ± 2.67

Tabela A.7: Resultados das combinações das heurísticas para o cenário 7

Caso	PRO (%)	RCC (%)	RCR (%)	TME (ms)	NMI	CMT
1	0	0	0	1315.51 ± 141.72	100000 ± 0	17.78 ± 2.89
2	0	0	0	1317.73 ± 143.02	100000 ± 0	17.77 ± 2.85
3	0	0	0	1325.13 ± 137.34	100000 ± 0	17.76 ± 2.72
4	0	0	1	1329.12 ± 137.36	100000 ± 0	17.64 ± 2.77
5	0	0	1	1333.4 ± 142.79	100000 ± 0	17.64 ± 2.95
6	0	0	7	1134.2 ± 91.97	100000 ± 0	16.35 ± 3.87
7	0	0	2	1556.91 ± 143.05	100000 ± 0	17.78 ± 2.89
8	0	0	3	1556.87 ± 142.09	100000 ± 0	17.61 ± 2.78
9	0	0	0	1555.37 ± 141.26	100000 ± 0	17.76 ± 2.95
10	0	0	2	1554.39 ± 142.74	100000 ± 0	17.7 ± 2.75
11	0	0	3	1135.2 ± 101.35	100000 ± 0	16.37 ± 3.96
12	0	0	9	1112.03 ± 108.27	100000 ± 0	16.57 ± 3.98
13	0	0	6	1125.05 ± 91.04	100000 ± 0	16.24 ± 3.94
14	0	0	8	1105.45 ± 92.15	100000 ± 0	16.35 ± 3.84
15	0	0	2	1113.5 ± 98.56	100000 ± 0	16.27 ± 3.89
16	0	0	4	1091.48 ± 101.67	100000 ± 0	16.5 ± 3.91
17	0	0	6	1111.86 ± 96.49	100000 ± 0	16.47 ± 3.96
18	0	0	4	1118.19 ± 94.24	100000 ± 0	16.35 ± 4.08