RTC REVISTA TÉCNICO-CIENTÍFICA DO IFSC
PUBLICAÇÃO DO IFSC
INSTITUTO FEDERAL Santa Catarina

Proposta de melhoria para o sequenciamento da furação em placas de circuito impresso

An approach to improve the drilling sequencing in printed circuit boards

Lucas William Zimmer | https://orcid.org/0000-0002-1170-8560
Carise Elisane Schmidt | https://orcid.org/0000-0003-2274-1953
Vinícius Berndsen Peccin | https://orcid.org/0000-0002-1846-4933

RESUMO

As Placas de Circuito Impresso (PCI) são componentes fundamentais dos produtos eletrônicos. Em sua produção, uma das etapas que demanda maior tempo é a furação, geralmente realizada por fresadoras de comando numérico computadorizado. Considerando essa etapa, este estudo aborda uma proposta de melhoria no tempo do processo de furação, focado na trajetória executada pela fresadora a cada troca de ferramenta. O objetivo é obter uma sequência de furação que minimize o deslocamento total da ferramenta. Um modelo matemático de programação linear inteira e duas abordagens heurísticas são utilizados para geração dessa sequência. Testes computacionais são conduzidos em um conjunto de instâncias, geradas a partir de layouts de PCIs. Os resultados mostram que as abordagens heurísticas são mais eficientes em fornecer soluções quando a instância possui mais de 10 furos do mesmo padrão, gerando rotas até 18,89% mais curtas, além de requerer um tempo de processamento, no mínimo, 544 vezes inferior, quando comparadas ao modelo matemático com tempo limite de processamento. Eles também mostram que a utilização de técnicas de otimização de baixa complexidade pode contribuir com a melhoria do processo produtivo, aumentando a sua eficiência.
Palavras-chave: roteirização em nós; problema do caixeiro viajante; programação matemática.

ABSTRACT

Printed Circuit Boards (PCBs) are essential components in electronic devices. Among the manufacturing steps, drilling is one of the most time-consuming, typically performed by computer numerical control (CNC) machines. This paper presents an improvement approach to reduce drilling time by optimizing the CNC toolpath after each tool change. The goal is to minimize the total tool displacement through an optimized drilling sequence. An integer linear programming model and two heuristic methods are proposed for sequence generation. Computational experiments on instances derived from PCB layouts demonstrate that the heuristics are more effective for instances with more than 10 similar holes, achieving toolpaths up to 18.89% shorter and processing times at least 544 times faster than the mathematical model within a fixed time limit. The results indicate that low-complexity optimization techniques can enhance production efficiency in PCB manufacturing.
Keywords: node routing; traveling salesman problem; mathematical programming.

Recebido em: 10/07/2025. Aprovado em: 10/03/2026.
Avaliado pelo sistema duplo-anônimo. Publicado conforme as normas da ABNT.
DOI: https://doi.org/10.35700/2316-8382.2026.v16.4020

1 INTRODUÇÃO

Com o avanço acelerado da tecnologia, o ritmo de desenvolvimento de novos dispositivos eletrônicos também disparou. Inovações cada vez mais presentes no cotidiano são geradas por componentes eletrônicos cada vez menores, que integrados formam esses dispositivos (Khandpur, 2006; Rao *et al.*, 2022). Eles são utilizados para realizar e simplificar grande parte das nossas tarefas, das mais simples às mais complexas, tornando o dia a dia cada vez mais conectado e tecnológico.

As placas de circuito impresso (PCIs) são a tecnologia de interconexão mais utilizada em dispositivos eletrônicos, integrando componentes em um mesmo circuito (Khandpur, 2006; Rao *et al*., 2022) e, por isso, são fundamentais na maioria desses dispositivos (Angelopoulos *et al.*, 2019; Laisupannawong; Intiyot; Jeenanunta, 2021; Shi; Zhu; Chen, 2022). Uma PCI é composta, basicamente, por camadas isolantes e trilhas condutoras de eletricidade, responsáveis por transmitir sinais entre os componentes.

Por conta do aumento na demanda por dispositivos eletrônicos, a indústria de PCIs tornou-se mais competitiva (Laisupannawong; Intiyot; Jeenanunta, 2021). A produção de uma PCI é complexa e envolve etapas anteriores à montagem: elaboração do diagrama elétrico e do layout (definição da posição dos componentes), transferência do layout para a placa e impressão, finalizando com a furação (Khandpur, 2006).

A furação é uma etapa crítica devido ao tempo demandado para mover a broca entre pontos (Wang, 2019). Com múltiplos padrões de furos, cerca de 70% do tempo total é gasto em movimentação e troca de ferramentas (Abidin; Ab Rashid; Mohamed, 2019). Em um sistema de produção em grande escala, pequenas melhorias na trajetória podem gerar reduções significativas de custos (Dewil *et al.*, 2019). A furação de PCIs normalmente é realizada por uma fresadora orientada por controle numérico computadorizado (CNC) (Dhouib, 2022), com movimentos baseados no sistema de coordenadas cartesianas e o posicionamento da ferramenta em relação a um ponto de referência fixo.

A otimização do caminho da ferramenta busca reduzir o deslocamento total entre pontos de furação consecutivos (Rico-Garcia *et al*., 2020). Com uma sequência de furação mais eficiente, reduz-se o tempo de execução e aumenta-se a taxa de produção (Wang, 2019), atendendo um dos principais objetivos de empresas de manufatura de PCIs, que é melhorar a eficiência dos processos de produção em termos de tempo, energia e custos (Khatiwada *et al.*, 2020).

Do ponto de vista matemático, o processo de furação de PCI é um problema de otimização combinatória (Dhouib, 2022), no qual se busca a melhor combinação de variáveis para maximizar ou minimizar uma função objetivo (Reinelt, 1994). A determinação da melhor sequência de furação, isto é, a que minimiza a distância total percorrida, pode ser modelada como um Problema do Caixeiro Viajante (PCV) (Wang, 2019; Khatiwada *et al.*, 2020).

Esse problema é um dos mais famosos dentro da área de otimização combinatória (Laporte, 2010; Bock *et al*., 2025). Ele consiste em visitar um conjunto de cidades exatamente uma vez e retornar à origem, minimizando a distância total. Nesta aplicação, a ferramenta corresponde ao caixeiro e os pontos de furação às cidades, com origem no ponto zero da máquina. Como uma PCI pode demandar diferentes diâmetros de furo, após concluir um padrão de furação, a fresadora deve retornar à posição inicial para realizar a troca da ferramenta e, só então, perfurar uma nova sequência de pontos. Assim, uma rota é gerada para cada padrão de furação a ser executado.

Sob a perspectiva da complexidade computacional, o PCV é NP-difícil (Laporte *et al*., 2000), o que exige alto poder computacional à medida que o número de pontos cresce (Lenstra; Rinnooy Kan, 1981; Bock *et al*., 2025). As abordagens de solução propostas na literatura se dividem, em geral, em métodos exatos e heurísticos. Métodos exatos podem obter a solução ótima, mas a um custo computacional elevado, enquanto heurísticas podem encontrar soluções próximas da ideal (ou até ótimas), porém sem garantia teórica, e com melhor desempenho computacional.

Diversos estudos abordam a otimização do caminho da ferramenta, visando melhorar a eficiência do processo. Medina-Rodríguez *et al.* (2012) apresentaram um algoritmo paralelo, baseado na Otimização de Colônia de Formigas, para encontrar uma sequência eficiente de furação. Rocha e Tostes (2013) e Ramos, Kamassury e Duarte (2016), abordaram a otimização desse processo utilizando Algoritmos Genéticos.

Mais recentemente, Wang (2019) aplicou um Algoritmo Genético com estratégia de reprodução por combinação total, aumentando a probabilidade de gerar bons indivíduos e melhorando a qualidade da população inicial com um algoritmo guloso. Khatiwada *et al.* (2020) aplicam um algoritmo evolutivo para otimizar o caminho da ferramenta quando há grande número de furos. Rico-Garcia *et al.* (2020) propõem uma versão discreta da Otimização baseada em Ensino-Aprendizagem com processamento paralelo.

Métodos híbridos também foram utilizados para tratar o problema. Abbas, Hamza e Aly (2014) aplicaram um método que une a abordagem Colônia de Formigas com as vantagens da busca local e do arranjo do padrão dos furos. Lim, Kanagaraj e Ponnambalam (2016) combinaram um Algoritmo Genético com uma estratégia evolutiva para otimizar a furação e minimizar o tempo de movimentação da ferramenta durante a usinagem. Al-Janan e Liu (2016) propuseram uma abordagem que combina um Algoritmo Genético com o método de Taguchi, Karuppusamy e Kang (2017) combinaram o algoritmo A* com um Algoritmo Genético para determinar o caminho ideal da ferramenta. Recentemente, Dhouib *et al.* (2022), propuseram um método híbrida, integrando um Algoritmo de Otimização de Colônia de Formigas e da heurística Dhouib-Matrix-TSP1, que trabalha com métricas estatísticas descritivas.

A otimização da rota de furação em PCIs, voltada à minimização do deslocamento total da ferramenta, é um problema clássico e amplamente estudado na literatura, especialmente com o uso de metaheurísticas. Este trabalho compara heurísticas de baixa complexidade computacional com uma formulação de programação linear inteira, sob limite de tempo. Discute-se o trade-off entre qualidade da solução e tempo de geração da rota, com foco na obtenção de sequências de furação mais eficientes do ponto de vista operacional. A investigação é conduzida em caráter teórico e computacional, com avaliação em instâncias de teste, não contemplando validação por implantação em linha de produção.

A sequência deste artigo está estruturada conforme segue. Na Seção 2, são descritas as instâncias de teste, detalhadas as abordagens de resolução aplicadas e descritos os testes computacionais realizados. Na Seção 3, os resultados são detalhados e discutidos. Por fim, na Seção 4, são apresentadas as considerações finais.

2 METODOLOGIA

Nesta seção, são apresentadas as etapas do estudo. Inicialmente, detalha-se o conjunto de instâncias de teste. Na sequência, relaciona-se a aplicação em estudo a um problema clássico de

otimização combinatória. Por fim, apresentam-se uma formulação matemática baseada em programação linear inteira e os algoritmos heurísticos aplicados para obtenção da solução.

2.1 Instâncias de teste

Foram utilizadas 12 instâncias de teste, extraídas de layouts públicos de PCIs disponíveis em formato Gerber. Na escolha das instâncias, priorizou-se a disponibilidade dos arquivos e a viabilidade de execução dos experimentos, buscando contemplar diferentes graus de dificuldade, caracterizados principalmente pelo número total de furos e pelo número de padrões distintos. A Tabela 1 sintetiza as características de cada instância, incluindo o total de furos, a quantidade de padrões, e o número mínimo e máximo de furos por padrão.

Tabela 1 — Características das instâncias de teste utilizadas

Instância	Nº total de furos	Nº de padrões distintos de furos	Nº de furos do mesmo padrão	
			Mínimo	Máximo
1	13	3	4	5
2	35	9	1	10
3	52	4	4	38
4	86	7	3	26
5	92	1	92	92
6	92	10	1	44
7	94	9	1	48
8	194	8	6	87
9	220	1	220	220
10	400	1	400	400
11	769	2	9	760
12	1038	7	3	989

Fonte: Dados da pesquisa (2025).

2.2 Descrição do problema e modelo matemático

O problema de encontrar a melhor sequência de pontos durante a execução de um tipo específico de furo em uma PCI pode ser modelado como um Problema do Caixeiro Viajante (PCV). Em sua forma clássica, o PCV consiste em determinar a rota mais curta que visita cada vértice de um grafo exatamente uma vez e retorna à origem (Laporte, 2010; Bock *et al.*, 2025).

Na aplicação em estudo, o conjunto de vértices é representado pelo ponto zero da máquina (origem da rota) e pelos pontos da placa com o mesmo padrão de furação, com distâncias conhecidas entre eles. Para cada padrão, a fresadora precisa realizar uma sequência de movimentos, saindo do ponto zero, passando por todos os pontos com esse padrão e retornando à origem para a troca de ferramenta. Ao minimizar a distância total percorrida pela ferramenta em cada rotas, otimiza-se o tempo total de execução da etapa de furação.

Considere um grafo completo e direcionado $G = (V, A)$, onde $V = \{1,2, \dots, n\}$ representa o conjunto de vértices e $A = \{(i, j): i, j \in V, i \neq j\}$ representa o conjunto de arcos, para o qual é definida uma matriz de custos $C = (c_{ij})$. A matriz C satisfaz a desigualdade triangular $c_{ij} \leq c_{ik} + c_{kj}$ para todo $i, j \in V$, com c_{ij} representando a distância euclidiana entre os vértices $i, j \in V$, de forma que $c_{ij} = c_{ji}$.

O problema é modelado via programação linear inteira, utilizando a formulação matemática proposta por Miller, Tucker e Zemlin (1960). Define-se x_{ij} como uma variável binária que assume valor 1 caso o arco $(i,j) \in A$ seja atravessado no percurso da rota, e 0 em caso contrário; u_i como uma variável auxiliar, contínua e não negativa. A formulação matemática é apresentada a seguir.

$$\text{Min } Z = \sum_{i=1}^{n} \sum_{\substack{j=1 \\ j \neq i}}^{n} c_{ij} x_{ij}, \tag{1}$$

sujeito à:

$$\sum_{i=1, i \neq j}^{n} x_{ij} = 1, \qquad j = 1,2, \ldots., n, \tag{2}$$

$$\sum_{j=1, j \neq i}^{n} x_{ij} = 1, \qquad i = 1,2, \ldots., n, \tag{3}$$

$$u_i - u_j + n x_{ij} \leq n - 1, \qquad \forall\, i, j = 2, \ldots, n, i \neq j, \tag{4}$$

$$u_i \in \mathbb{R}_+, \quad \forall\, i \in V, \tag{5}$$

$$x_{ij} \in \{0,1\}, \quad \forall\, (i,j) \in A. \tag{6}$$

A função objetivo (1) minimiza a distância total da rota. As restrições (2) e (3) são de designação e garantem que cada vértice seja visitado exatamente uma vez. As restrições (4) eliminam subciclos, enquanto (5) e (6) definem o domínio das variáveis u_i e x_{ij}, respectivamente.

2.3 Abordagens heurísticas

Para solucionar cada PCV são propostas duas abordagens heurísticas. Ambas são compostas por duas etapas: i) construção de uma rota inicial; ii) melhoria da rota inicialmente gerada. As abordagens diferem apenas na etapa de geração da rota inicial: a primeira utiliza o algoritmo de inserção *Vizinho mais Próximo* (VMP) e a segunda utiliza o algoritmo *Inserção do mais Distante* (IMD). Na etapa de melhoria da rota, ambas aplicam o algoritmo *2-opt*. A seguir, os algoritmos são detalhados.

2.3.1 Algoritmo de inserção do vizinho mais próximo

O algoritmo VMP é uma heurística gulosa simples para construção de rota. Inicialmente, é escolhido um vértice arbitrário i, que é adicionado à rota como vértice inicial. Neste estudo, esse vértice é sempre o ponto zero da máquina. Em seguida, adiciona iterativamente o vértice ainda não visitado que minimiza o custo da aresta a partir do último vértice inserido. Em seguida, um processo iterativo é iniciado para inserção dos demais vértices. O processo se repete até que todos os vértices sejam incluídos. Por fim, a rota é fechada com o retorno ao vértice inicial, permitindo o cálculo do custo total. O pseudocódigo ilustrado na Figura 1 detalha o seu funcionamento.

Figura 1 — Pseudocódigo do algoritmo VMP

Algorithm 1 Vizinho mais Próximo

Seja i um vértice aleatório do grafo.
$i_0 \leftarrow i$
Insira o vértice i_0 na rota.
$V \leftarrow \{1, 2, ..., n\} \setminus \{i_0\}$
while $V \neq \emptyset$ **do**
    Seja $j \in V$ um vértice tal que c_{ij} é mínimo.
    Insira o vértice j na rota.
    $V \leftarrow V \setminus \{j\}$
    $i \leftarrow j$
end while
Insira o vértice i_0 na rota.

Fonte: Elaborado pelos autores (2025).

2.3.2 Algoritmo de inserção do mais distante

O algoritmo IMD é uma heurística construtiva de inserção. O pseudocódigo ilustrado na Figura 2 detalha o funcionamento do algoritmo.

Figura 2 — Pseudocódigo do algoritmo IMD

Algorithm 2 Inserção do mais Distante

$V \leftarrow \{1, 2, ..., n\}$
Sejam $i, j \in V$ dois vértices tais que c_{ij} é máximo.
Inicie uma rota com a sequência $i - j - i$.
$S \leftarrow \{i, j\}$
$V \leftarrow V \setminus \{i, j\}$
while $V \neq \emptyset$ **do**
    Sejam $i \in S$ e $k \in V$ dois vértices tais que c_{ik} é máximo.
    Encontre uma aresta $\{i, j\}$ da rota, que minimiza $c_{ik} + c_{kj} - c_{ij}$.
    Insira o vértice k na rota, entre os vértices i e j.
    $V \leftarrow V \setminus \{k\}$
    $S \leftarrow S \cup \{k\}$
end while

Fonte: Elaborado pelos autores (2025).

Inicialmente, seleciona-se um par de vértices, i e j, cuja aresta possui custo máximo, formando uma rota inicial $i - j - i$. Em seguida, o algoritmo seleciona iterativamente o vértice k ainda fora da rota que está mais distante de algum vértice já inserido. Esse vértice é então inserido na posição que minimiza o custo de inserção, isto é, entre os dois vértices da aresta $\{i, j\}$ cuja substituição por $\{i, k\}$ e $\{k, j\}$ resulta no menor acréscimo de custo. O processo se repete até que todos os vértices tenham sido inseridos à rota.

2.3.3 Algoritmo 2-opt

O algoritmo 2-opt é um algoritmo iterativo de busca local. A partir de uma rota arbitrária, ele avalia movimentos de troca envolvendo dois arcos não adjacentes, reconectando seus vértices e recalculando o custo da rota. Os pseudocódigos ilustrados nas Figuras 3 e 4 apresentam, respectivamente, a rotina de troca utilizada e o pseudocódigo do algoritmo completo.

Dada uma rota inicial arbitrária, o algoritmo inicia o processo iterativo selecionando dois arcos não adjacentes e aplica a troca, reconectando os vértices desses arcos, conforme descrito na Figura 3. Se o movimento reduz o custo total, a rota é atualizada incorporando a troca. O processo se repete enquanto houver melhoria, encerrando quando nenhum movimento adicional reduz o custo. Um pseudocódigo do algoritmo completo é ilustrado na Figura 4.

Figura 3 — Pseudocódigo da operação de troca realizada pelo algoritmo 2-opt

Algorithm 3 Troca 2-opt

1: **function** TROCA2OPT(r, i, j)
2: $r' \leftarrow r[0, ..., i-1] \| \overline{r[i, ..., j]} \| r[j+1, ..., n]$
3: return r'
4: **end function**

Fonte: Adaptado de Viccari e Miyazawa (2022).

Figura 4 — Pseudocódigo do algoritmo 2-opt

Algorithm 4 2-opt

1: Seja n a quantidade de vértices a visitar
2: Seja r uma solução inicial
3: Seja C a matriz de custos do grafo
4: **function** 2OPT(r, n, C)
5: **repeat**
6: $melhoria \leftarrow 0$
7: **for all** i, j tal que $1 \leq i < j \leq n$ **do**
8: $r' \leftarrow$TROCA2OPT(r, i, j)
9: **if** $C(r') < C(r)$ **then**
10: $r' \leftarrow r$
11: $melhoria \leftarrow 1$
12: **break** loop da linha 7
13: **end if**
14: **end for**
15: **until** $melhoria = 0$
16: **end function**

Fonte: Adaptado de Viccari e Miyazawa (2022).

2.4 Testes computacionais

Para avaliar e comparar as abordagens propostas, foram realizados testes computacionais, onde cada uma das três abordagens de solução foi aplicada a cada instância e a cada padrão de furo. Os custos obtidos em cada rota de uma mesma instância foram somados, assim como o tempo de

execução. Dessa forma, para cada instância e para cada abordagem, reportam-se a distância total percorrida pela ferramenta e o tempo necessário para gerar a solução.

Nos testes envolvendo o modelo de programação linear inteira, expresso por (1)–(6), impôs-se um limite de tempo de processamento de 3.600 segundos por rota (padrão de furação), como critério uniforme de execução. Para padrões com grande quantidade de furos, a formulação pode apresentar degradação de escalabilidade; motivo pelo qual os resultados do solver são analisados sob limite de tempo. Para as abordagens heurísticas VMP+2-opt e IMD+2-opt, não houve limitação no tempo de execução.

Os algoritmos foram implementados em C++. Para resolver o modelo matemático, foi utilizado o solver Gurobi, versão 11.0.0, em configuração padrão. Os testes foram conduzidos em uma máquina com processador Intel Core i7-1165G7 de 2.80GHz e 16.00 GB de memória RAM.

3 RESULTADOS E DISCUSSÕES

Nesta seção, são apresentados os resultados dos testes computacionais. A Tabela 2 compara as soluções geradas pela formulação matemática (1)–(6) e pelas duas abordagens heurísticas. Para cada instância, os valores descritos correspondem a soma dos resultados gerados em cada padrão de furação. Para o modelo, reportam-se, respectivamente: a melhor solução, o limitante inferior médio, o *gap* percentual médio e o tempo total de processamento (segundos). Na sequência, para cada combinação de abordagem heurística, reportam-se a solução encontrada e tempo de processamento (segundos), respectivamente. A melhor solução por instância está destacada em negrito e, em caso de empate, todas as melhores soluções estão destacadas.

Tabela 2 — Resultados comparativos entre as abordagens de solução

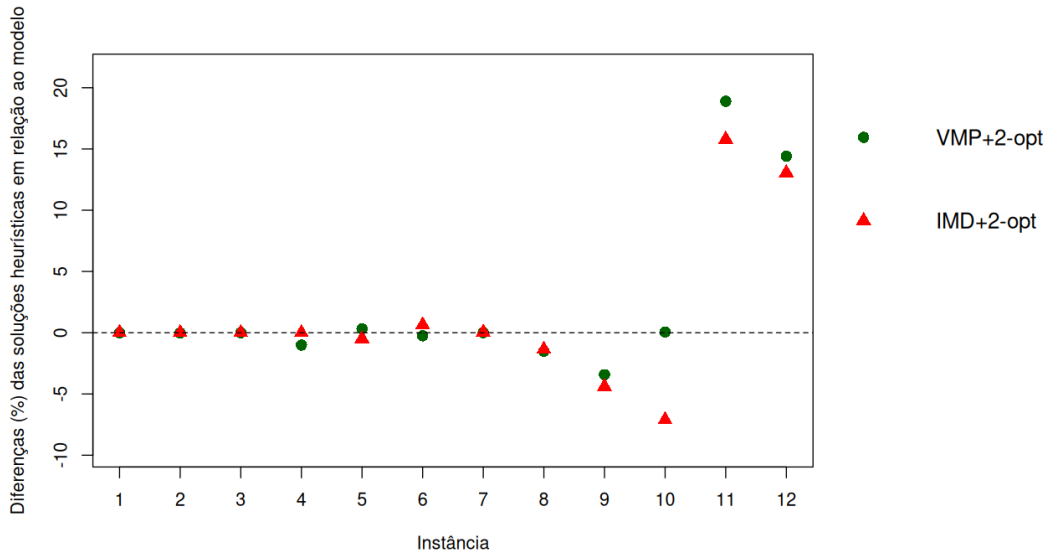
Instância	Modelo de programação linear				Heurísticas			
					VMP + 2-opt		IMD + 2-opt	
	Melhor solução	Limitante Inferior	Gap (%)	Tempo (s)	Solução	Tempo (s)	Solução	Tempo (s)
1	**1067,41**	1067,41	0,00	< 1	**1067,41**	< 1	**1067,41**	< 1
2	**2701,19**	2701,19	0,00	< 1	2701,61	< 1	**2701,19**	< 1
3	**3849,55**	3849,22	0,03	3600	**3849,55**	< 1	**3849,55**	< 1
4	7265,94	6861,38	32,78	7222	7338,96	< 1	**7265,94**	< 1
5	2644,55	2574,09	2,66	3600	**2636,00**	< 1	2657,50	< 1
6	**4351,70**	43123,09	1,62	3604	4362,63	< 1	4379,45	< 1
7	**9269,86**	8567,09	52,47	19579	**9269,86**	< 1	9273,94	< 1
8	**1153,00**	1096,62	33,11	31559	1170,38	< 1	1168,68	< 1
9	**1192,92**	1140,32	4,41	7042	1233,78	< 1	1245,33	2
10	1131,38	989,73	12,52	3605	1130,78	< 1	**1211,87**	7
11	6011,48	4488,21	29,34	3601	**4875,86**	< 1	5061,89	64
12	5896,86	4826,06	38,98	8093	**5047,28**	< 1	5126,84	99
Média	3877,99	3539,47	17,33	7625	3723,68	< 1	3750,80	14

Fonte: Dados da pesquisa (2025).

Para facilitar a análise visual, a Figura 5 apresenta, por instância, a diferença percentual entre as soluções das abordagens heurísticas e a solução da formulação matemática (1)–(6), tomando a linha

tracejada como referência. Valores positivos indicam que a heurística produziu rota mais curta do que o modelo matemático no tempo limite, e valores negativos indicam o contrário.

Figura 5 — Variação percentual entre as soluções das abordagens heurísticas e o modelo, por instância



Fonte: Elaborado pelos autores (2025).

Os resultados da Tabela 2 indicam que o modelo de programação linear inteira encontrou a solução ótima em apenas 2 das 12 instâncias, considerando o limite de tempo estabelecido. Nessas instâncias, o número máximo de furos em um mesmo padrão foi 10, e a solução ótima foi obtida instantaneamente. Porém, considerando todo o conjunto de instâncias, o tempo médio de processamento foi de 7625 segundos, com um *gap* médio entre a melhor solução encontrada e o limitante inferior do respectivo problema igual a 17,33%, chegando a 52,47% para uma instância em particular. Esses resultados evidenciam a baixa eficiência da formulação, em termos de tempo de processamento. Em comparação, o tempo médio do modelo foi ao menos 544 vezes maior que o das heurísticas (IMD+2-opt).

As abordagens heurísticas geraram soluções instantâneas quando a rota inicial foi gerada pelo algoritmo VMP, e demandaram, em média, 14 segundos quando a rota inicial foi gerada pelo algoritmo IMD. Em termos de qualidade, em relação ao modelo de programação linear, as soluções das heurísticas foram no máximo 3,43% (instância 9) e 7,11% (instância 10) piores, respectivamente, quando os algoritmos VMP+2-opt e IMD+2-opt foram usados. Por outro lado, também chegaram a ser 18,89% (instância 11) e 15,80% (instância 11) melhores do que as do modelo, respectivamente, para VMP+2-opt e IMD+2-opt. As maiores melhorias ocorreram nas instâncias com maior número de furos em um mesmo padrão (instâncias 11 e 12).

Como sintetizado na Figura 5, observa-se que os ganhos mais expressivos tendem a se concentrar nas instâncias de maior porte. Nas instâncias menores, as diferenças em relação à solução do modelo, quando negativas, apresentam magnitudes percentuais mais reduzidas.

A Tabela 3 compara os resultados das abordagens heurísticas, discriminando as duas etapas do método. Para cada instância, os valores correspondem à soma dos resultados obtidos em cada

padrão de furação. Para cada abordagem, reporta-se a solução após a etapa de construção da rota e, em seguida, a solução após a etapa de melhoria (2-opt). Os melhores resultados, por instância, são destacados em negrito. Em caso de empate, todos os melhores valores são destacados.

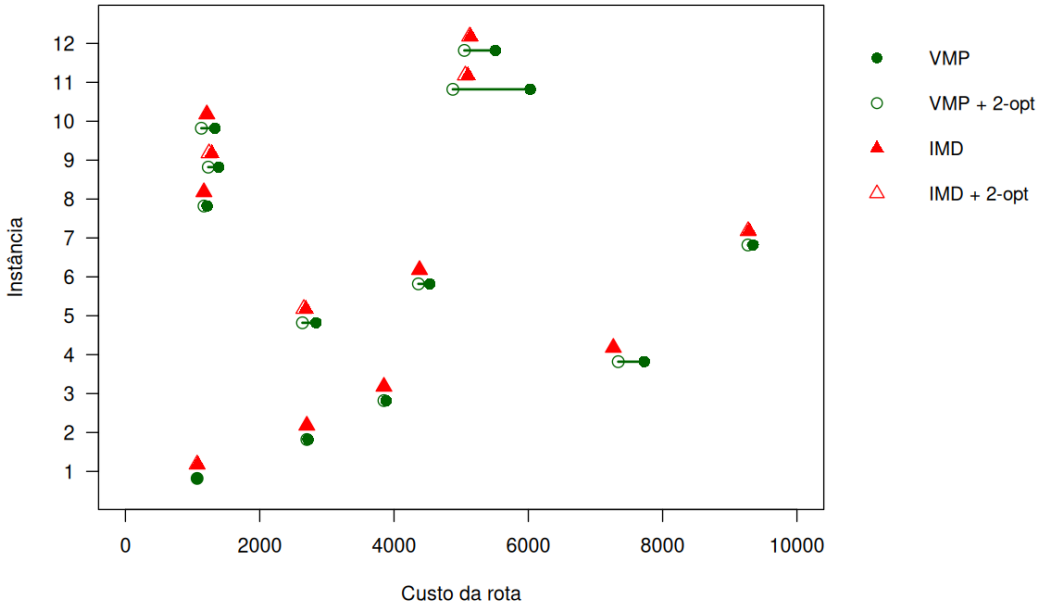
Tabela 3 — Resultados das soluções geradas pelas abordagens heurísticas, por etapa

Instância	VMP + 2-opt		IMD + 2-opt	
	Fase de geração	Fase de melhoria	Fase de geração	Fase de melhoria
1	1069,52	**1067,41**	1078,41	**1067,41**
2	2719,97	2701,61	2701,19	**2701,19**
3	3881,26	**3849,55**	3849,55	**3849,55**
4	7726,59	7338,96	7265,94	**7265,94**
5	2838,46	**2636,00**	2688,23	2657,50
6	4532,12	**4362,63**	4382,88	4379,45
7	9347,15	**9269,86**	9289,54	9273,94
8	1216,67	1170,38	1172,36	**1168,68**
9	1388,61	**1233,78**	1285,68	1245,33
10	1334,28	**1130,78**	1219,66	1211,87
11	6028,20	**4875,86**	5101,16	5061,89
12	5507,95	**5047,28**	5143,36	5126,84
Média	3965,90	3723,68	3764,83	3750,80

Fonte: Dados da pesquisa (2025).

Para facilitar a análise visual do efeito da etapa de melhoria, a Figura 6 apresenta, para cada instância, o custo da rota inicial obtida na etapa de construção e o custo após a etapa de melhoria, para as abordagens VMP e IMD. Assim, o deslocamento horizontal entre os marcadores indica a redução de custo promovida pelo algoritmo 2-opt em cada instância.

Figura 6 — Impacto do algoritmo 2-opt sobre o custo das rotas iniciais, conforme heurística de construção e instância



Fonte: Elaborado pelos autores (2025).

Os resultados da Tabela 3 indicam que, exceto na primeira instância, o algoritmo IMD foi mais eficiente que o VMP na geração da rota inicial, com melhoria média de quase 5% e máxima acima de 15%, para uma instância em particular. Contudo, na segunda fase, o algoritmo 2-opt gerou, em média, melhores soluções quando aplicado sobre rotas iniciais geradas pelo VMP do que pelo IMD. Como resultado, a combinação VMP+2-opt gera soluções, em média, cerca de 1% melhores, podendo chegar a quase 7% para uma instância em particular. Esse efeito pode ser visualmente observado na Figura 6, pelo deslocamento horizontal entre os marcadores de rota inicial e após melhoria (2-opt) em cada instância.

A melhoria promovida pela etapa 2-opt também difere entre as combinações. Para VMP+2-opt, a melhoria média em relação à rota inicial foi de 6,34%, alcançando até 19,12%. Já para IMD+2-opt, a melhoria média foi inferior a 1%, e o melhor caso reduziu a rota inicial em 3,14%. Esses resultados mostram que a escolha do método de construção influencia diretamente o potencial de melhoria do 2-opt e, portanto, a qualidade final da solução.

De forma geral, os testes evidenciaram a vantagem das abordagens heurísticas tanto em tempo de processamento quanto na qualidade das soluções. Mesmo que o objetivo final seja reduzir o tempo de furação, o tempo necessário para gerar a rota também é relevante, especialmente quando há muitos furos de um mesmo padrão. Nesses casos, as heurísticas produzem rotas substancialmente melhores com tempo de processamento muito menor. Além disso, são abordagens simples de implementar e dispensam o uso de solver, ao contrário do modelo de programação linear.

4 CONSIDERAÇÕES FINAIS

Este trabalho considerou, em caráter teórico e computacional, um aprimoramento no processo de furação na produção de PCIs, com o objetivo de gerar uma sequência de movimentos que minimize o deslocamento total da ferramenta. Para avaliar os resultados, foram utilizadas instâncias de teste geradas a partir de layouts de placas. As abordagens propostas produziram soluções para instâncias com até 1038 furos e diferentes padrões de furação.

Foram avaliados um modelo de programação linear inteira e duas abordagens heurísticas, ambas em duas fases: geração de uma rota inicial seguida de sua melhoria. Testes computacionais foram realizados para comparar a qualidade das soluções e o tempo de processamento demandado por cada abordagem.

Os resultados indicaram que as abordagens heurísticas geraram rotas até 18,89% mais curtas em relação ao modelo de programação linear, demandando pelo menos 544 vezes menos tempo de processamento. Em média, o algoritmo IMD produziu rotas iniciais melhores do que o algoritmo VMP. Entretanto, após a aplicação do algoritmo 2-opt, a abordagem VMP+2-opt apresentou soluções levemente melhores e mais rápidas do que a IMD+2-opt. A aplicação do 2-opt sobre rotas geradas pelo VMP reduziu a distância total em 6,34% em média, chegando a 19,12% em casos específicos. Esses resultados evidenciam a importância de combinar heurísticas para aumentar a qualidade da solução e reduzir o tempo de obtenção, especialmente em problemas de otimização combinatória complexos.

Por fim, os resultados amostram a aplicação de técnicas de otimização pode contribuir para a melhoria de processos produtivos, como a furação de PCIs. Com a redução no tempo de furação das placas é possível gerar um aumento na quantidade produzida, ampliando a capacidade de produção.

Utilizando uma abordagem relativamente simples, é possível implementar uma proposta de melhoria para tornar o processo mais eficiente.

Como limitação, ressalta-se que as instâncias avaliadas foram obtidas a partir de layouts públicos e não cobrem toda a variabilidade observada em ambiente industrial. Como sugestão de trabalho futuro, propõe-se ampliar o conjunto de testes e realizar a validação prática da proposta em ambiente industrial, por meio de sua implementação em máquina CNC e avaliação em condições reais de produção, considerando tempos de troca de ferramenta e restrições operacionais do processo.

REFERÊNCIAS

ABBAS, Adel T.; HAMZA, Karim; ALY, Mohamed. CNC machining path planning optimization for circular hole patterns via a hybrid ant colony optimization approach. **Journal of Mechanical Engineering Research**, v. 4, n. 2, p. 16-29, 2014. DOI: https://doi.org/10.5539/mer.v4n2p16.

ABIDIN, Najwa W. Z.; AB RASHID, Mohd F. F.; MOHAMED, Nik M. Z. N. A review of multi-holes drilling path optimization using soft computing approaches. **Archives Computational Methods in Engineering**, v. 26, p. 107-118, 2019. DOI: https://doi.org/10.1007/s11831-017-9228-1.

AL-JANAN, Dony H.; LIU, Tung-Kuan. Path optimization of CNC PCB drilling using hybrid Taguchi genetic algorithm. **Kybernetes**, v. 45, n. 1, p. 107-125, 2016. DOI: https://doi.org/10.1108/K-03-2015-0069.

ANGELOPOULOS, Angelos *et al*. Tackling faults in the industry 4.0 era - a survey of machine-learning solutions and key aspects. **Sensors**, v. 20, n. 1, p. 109, 2019. DOI: https://doi.org/10.3390/s20010109.

BOCK, Stefan; BOMSDORF, Stefan; BOYSEN, Nils; SCHNEIDER, Michael. A survey on the traveling salesman problem and its variants in a warehousing context. **European Journal of Operational Research**, v. 322, n. 1, p. 1-14, 2025. DOI: https://doi.org/10.1016/j.ejor.2024.04.014.

DEWIL, Reginald; KÜÇÜKOGLU, Ilker; LUTEYN, Corrine; CATTRYSSE, Dirk. A critical review of multi-hole drilling path optimization. **Archives of Computational Methods in Engineering**, v. 26, p. 449-459, 2019. DOI: https://doi.org/10.1007/s11831-018-9251-x.

DHOUIB, Souhail. Hole drilling route optimization in printed circuit boards using Far-to-Near metaheuristics. **International Journal of Strategic Engineering**, v. 5, n. 1, p. 1-12, 2022. DOI: https://doi.org/10.4018/IJoSE.301568.

DHOUIB, Souhail; ZOUARI, Alaeddine; DHOUIB, Saima; CHABCHOUB, Habib. Integrating the artificial bee colony metaheuristic with Dhouib-Matrix-TSP1 heuristic for holes drilling problems. **Journal of Industrial and Production Engineering**, v. 40, n. 3, p. 177-187, 2022. DOI: https://doi.org/10.1080/21681015.2022.2158499.

KARUPPUSAMY, Naveen S.; KANG, Bo-Yeong. Minimizing airtime by optimizing tool path in computer numerical control machine tools with application of A* and genetic algorithms. **Advances in Mechanical Engineering**, v. 9, n. 12, p. 1-9, 2017. DOI: https://doi.org/10.1177/1687814017737448.

KHANDPUR, Raghbir S. **Printed Circuit Boards:** Design, fabrication and assembly. New York: McGraw-Hill, 2006.

KHATIWADA, Denish; NEPALI, Nabin; CHAULAGAIN, Nabin R.; BHATTARAI, Aayush. Tool path optimization for drilling holes using genetic algorithm. **International Journal of Machine Tools and Maintenance Engineering**, v. 1, n. 1, p. 36.42, 2020.

LAISUPANNAWONG, Teeradech; INTIYOT, Boonyarit; JEENANUNTA, Chawalit. Mixed-integer linear programming model and heuristic for short-term scheduling of pressing process in multi-layer printed circuit board manufacturing. **Mathematics**, v. 9, n. 6, p. 653, 2021. DOI: https://doi.org/10.3390/math9060653.

LAPORTE, Gilbert. A concise guide to the traveling salesman problem. **Journal of the Operational Research Society**, v. 61, n. 1, p. 35-40, 2010. DOI: https://doi.org/10.1057/jors.2009.76.

LAPORTE, Gilbert; GENDREAU, Michel; POTVIN, Jean-Yves; SEMET, Frédréric. Classical and modern heuristics for the vehicle routing problem. **International Transactions in Operational Research**, v. 7, n. 4/5, p. 285-300, 2000. DOI: https://doi.org/10.1111/j.1475-3995.2000.tb00200.x.

LENSTRA, Jan K.; RINNOOY KAN, Alexander H. G. Complexity of vehicle routing and scheduling problems. **Networks**, v. 11, p. 221-228, 1981. DOI: https://doi.org/10.1002/net.3230110211.

LIM, Esmonde W. C.; KANAGARAJ, Ganesan; PONNAMBALAM, S. G. A hybrid cuckoo search-genetic algorithm for hole-making sequence optimization. **Journal of Intelligent Manufacturing**, v. 27, n. 2, p. 417-429, 2016. DOI: https://doi.org/10.1007/s10845-014-0873-z.

MEDINA-RODRÍGUEZ, Nataly; MONTIEL-ROSS, Oscar; SEPÚLVEDA, Roberto; CASTILLO, Oscar. Tool path optimization for computer numerical control machines based on parallel ACO. **Engineering Letters**, v. 20, n. 1, p. 101-108, 2012.

MILLER, Charles E.; TUCKER, Albert W.; ZEMLIN, Robert A. Integer programming formulation of traveling salesman problems. **Journal of the ACM**, v. 7, n. 4, p. 326-329, 1960. DOI: https://doi.org/10.1145/321043.321046.

RAMOS, Endrew B.; KAMASSURY, Jorge K.; DUARTE, Wandesson. Aplicação do algoritmo genético no problema de perfuração em placas de circuito impresso: integrando inteligência computacional e eletrônica. *In*: XLIV Congresso Brasileiro de Educação em Engenharia (COBENGE), 44., 2016, Natal. **Anais do COBENGE**. Natal: Abenge, 2016.

RAO, Chilakapati H.; AVINASH, Kothuru; VARAPRASAD, B.K.S.V.L.; GOEL, Sanket. A review on printed electronics with digital 3D printing: fabrication techniques, materials, challenges and future opportunities. **Journal of Electronic Materials**, v. 51, p. 2747-2765, 2022. DOI: https://doi.org/10.1007/s11664-022-09579-7.

REINELT, Gerhard. **The traveling salesman** - computational solutions for TSP applications. Berlin: Springer Verlag, 1994.

RICO-GARCIA, Hector; SANCHEZ-ROMERO, Jose-Luis; GOMIS, Hector M.; RAO, Ravipudi V. Parallel implementation of metaheuristics for optimizing tool path computation on CNC machining. **Computers in Industry**, v. 123, n. 103322, 2020. DOI: https://doi.org/10.1016/j.compind.2020.103322.

ROCHA JR., Paulo A. S.; DE TOSTES, Maria. E. L. Algoritmo genético aplicado à otimização da furação de placas de circuito impresso. *In*: XI Simpósio Brasileiro de Automação Inteligente (SBAI), 11., 2013, Fortaleza. **Anais do SBAI**. Fortaleza: Sociedade Brasileira de Automática (SBA), 2013.

SHI, Hongyan; ZHU, Tao; CHEN, Zhuangpei. Back-drilling of high-speed printed circuit boards: a review. **The International Journal of Advanced Manufacturing Technology**, v. 121, p. 1483-1499, 2022. DOI: https://doi.org/10.1007/s00170-022-09476-7.

VICCARI, João P B.; MIYAZAWA, Flávio. K. **Algoritmos e heurísticas para o problema do caixeiro viajante com minimização de energia**. Campinas: Instituto de Computação, Universidade Estadual de Campinas, 2022. (Relatório Técnico, PFG-21-40).

WANG, Yuzhen. PCB drill path optimization by improved genetic algorithm. *In*: International Conference on Control, Automation and Robotics (ICCAR), 5., 2019, Beijing. **Anais...** Beijing: IEEE, 2019. p. 744 – 748. DOI: https://doi.org/10.1109/ICCAR.2019.8813494