UNIVERSIDADE ESTADUAL DE LONDRINA

RENATA MASCARI

UMA ABORDAGEM HEURÍSTICA PARA MINIMIZAR O TEMPO DE PREPARAÇÃO DE UMA MÁQUINA TUBETEIRA

Londrina
2011

RENATA MASCARI

UMA ABORDAGEM HEURÍSTICA PARA MINIMIZAR O TEMPO DE PREPARAÇÃO DE UMA MÁQUINA TUBETEIRA

Dissertação de mestrado apresentada ao Departamento de Matemática da Universidade Estadual de Londrina, como requisito parcial para a obtenção do Título de mestre em Matemática Aplicada e Computacional.

Orientador: Prof. Dr. Robinson Samuel Vieira Hoto

Londrina
2011

Catalogação elaborada pela Divisão de Processos Técnicos da Biblioteca Central da Universidade Estadual de Londrina

Dados Internacionais de Catalogação -na-Publicação (CIP)

M395a Mascari, Renata.
Uma abordagem heurística para minimizar o tempo de preparação de uma máquina tubeteira / Renata Mascari. – Londrina, 2011.
xix, 97 f. : il.

Orientador: Robinson Samuel Vieira Hoto.
Dissertação (Mestrado em Matemática Aplicada e Computacional) – Universidade Estadual de Londrina, Centro de Ciências Exatas, Programa de Pós -Graduação em Matemática Aplicada e Computacional, 2011.
Inclui bibliografia.

1. Programação heurística – Teses. 2. Problema do caixeiro viajante – Teses. 3. Otimização matemática – Teses. 4. Pesquisa operacional – Teses. 5. Métodos de simulação – Teses. I. Hoto, Robinson Samuel Vieira. II. Universidade Estadual de Londrina. Centro de Ciências Exatas. Programa de Pós -Graduação em Matemática Aplicada e Computacional. III. Título.

CDU 519.85

RENATA MASCARI

UMA ABORDAGEM HEURÍSTICA PARA MINIMIZAR O TEMPO DE PREPARAÇÃO DE UMA MÁQUINA TUBETEIRA

Dissertação de mestrado apresentada ao Departamento de Matemática da Universidade Estadual de Londrina, como requisito parcial para a obtenção do Título de mestre em Matemática Aplicada e Computacional.

BANCA EXAMINADORA

_____

Prof. Dr. Robinson Samuel Vieira Hoto
UEL – Londrina – PR

_____

Prof. Dr. Celso Carnieri
UFPR – PR

_____

Profa. Dra. Sílvia Galvão de Souza Cervantes
UEL – Londrina – PR

Londrina, 11 de abril de 2011.

Aos meus pais, que sempre
me apoiaram e me ensinaram o caminho...

AGRADECIMENTOS

Agradeço a Deus, pelas bençãos proporcionadas a mim, por sempre guiar os meus passos e me dar forças para seguir em frente.

Aos meus pais Lacir e Eliane por dedicarem suas vidas à minha formação, por me criarem em um ambiente de respeito, honestidade e amor, os quais foram imprescindíveis para a minha caminhada, e à minha irmã Daniela por estarem ao meu lado nos momentos de alegria e de dificuldades, por acreditarem em mim e me estimularem na busca pelo conhecimento. Agradeço também por não medirem esforços para que mais esta etapa da minha vida fosse concluída.

Ao meu orientador Robinson Hoto, por compartilhar seu conhecimento, pela atenção, paciência e amizade no decorrer do trabalho, e à Adriana Borssoi pelos momentos de amizade e sinceridade.

Ao colega Marcos Okamura Rodrigues pela colaboração no desenvolvimento do programa que é apresentado no trabalho.

Ao meu namorado Vlademir pelo apoio, amor e admiração dedicados a mim durante todos estes anos.

Aos meus amigos de graduação Camila Fogaça, Ewerton Lemes, Poliane Cristina, Rodrigo Nunes Monteiro e Carlos Eduardo (*in memorian*).

Ao meu amigo Rodrigo Banhos, que durante os anos da minha graduação esteve presente em momentos especiais da minha vida acadêmica e pessoal. Pelas conversas, conselhos, pelas risadas e pelo companheirismo.

Aos meus irmãos na fé da Igreja Jesus Cristo Vivo que me ajudaram e me fortaleceram com suas orações.

Ao professor Albo Carlos Cavalheiro, a quem devo grande parte do meu conhecimento e crescimento acadêmico.

Ao professor Ricardo Ferreira, pelo incentivo desde o começo da minha graduação para que eu realizasse um curso de pós-graduacão.

Aos professores que estiveram presentes desde o início desta caminhada e que contribuíram para minha formação.

Agradeço a CAPES, pelo auxílio financeiro concedido.

“A alegria está na luta, na tentativa, no sofrimento envolvido. Não na vitória propriamente dita.”

Mahatma Gandhi

MASCARI, Renata. **Uma abordagem heurística para minimizar o tempo de preparação de uma máquina tubeteira**. 2011. 116 f. Dissertação (Mestrado em Ensino de Ciências e Educação Matemática) – Universidade Estadual de Londrina, Londrina, 2011.

RESUMO

Indústrias dos mais diversos segmentos já se conscientizaram da importância de efetuarem um planejamento inteligente de seus tempo e custos na produção de seus produtos, e para resolver tal problema buscam meios de otimizar seus processos com o auxílio de modelos matemáticos e ferramentas computacionais. Neste trabalho abordamos métodos heurísticos para otimizar o tempo produtivo de uma tubeteira (máquina que confecciona tubetes). Tubetes são tubos feitos pela colagem de fitas de papel, as quais são depositadas em rolos que recebem o nome de bolachas, sendo que algumas delas podem ser aproveitadas entre a confecção de dois tubetes. Apresentamos um modelo matemático para minimizar a quantidade de trocas e movimentos de bolachas, bem como as implementações das heurísticas Vizinho mais Próximo, Melhor Vizinho mais Próximo, Adaptações das heurísticas 2-Opt e 3-Opt e ainda uma heurística de permutação denominada Melhor Configuração em linguagem C^{++} utilizando o IDE (ambiente de desenvolvimento integrado) *WxDev* - C^{++}. Os resultados obtidos pelas simulações apresentaram melhoria em relação aos obtidos por uma indústria do segmento.

Palavras-Chave: Otimização. Problema do caixeiro viajante generalizado. Tubeteira. Implementação. Heurística.

MASCARI, Renata. **Uma abordagem heurística para minimizar o tempo de preparação de uma máquina tubeteira**. 2011. 116 f. Dissertação (Mestrado em Ensino de Ciências e Educação Matemática) – Universidade Estadual de Londrina, Londrina, 2011.

ABSTRACT

Industries from different segments have been realized the importance of effecting an intelligent design of their time and costs in the production of their products, and to solve this problem they are looking for ways to optimize their processes with the aid of mathematical models and computational tools. In this work we approach heuristic methods to optimize the time of preparation of a tubettes machine. Tubettes are tubes made by gluing strips of paper that are packed in paper reels, and some of them may be used between one and making another tubettes. We presents a mathematical model for the minimization of changing reels and moviments and also implementations for the heuristics Nearest Neighbor, an improvement of a nearest neighbor, an adaptation of the heuristics 2-Opt and 3-Opt and an heuristic of permutation called Best Configuration using the IDE (integrated development environment) *WxDev* - C^{++}. The results obtained by simulations presented improvement over those used by the company.

Keywords: Optimization. Generalized traveling salesman problem. Tubettes machine. Implementation. Heuristic.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

PCV	Problema do Caixeiro Viajante
PCVG	Problema do Caixeiro Viajante Generalizado
VMP	Vizinho Mais Próximo
MVMP	Melhor Vizinho mais Próximo
2-Part	Biparticionamento
3-Part	Triparticionamento
MC	Melhor Configuração
MVMP-MC	Melhor Vizinho mais Próximo com a Melhor Configuração
MVMP-2-Part	Melhor Vizinho mais Próximo Biparticionado
MVMP-3-Part	Melhor Vizinho Mais Próximo Triparticionado
MVMP-2-Part-MC	Melhor Vizinho Mais Próximo Biparticionado com a Melhor Configuração
MVMP-3-Part-MC	Melhor Vizinho Mais Próximo Triparticionado com a Melhor Configuração

SUMÁRIO

INTRODUÇÃO

Nas últimas décadas, estudos vêm sendo feitos a fim de desenvolver técnicas que encontrem soluções satisfatórias para as mais diversas empresas, as quais buscam o maior retorno possível em suas atividades, seja ele qualidade no atendimento, aumento na produtividade, redução de custos e consequentemente aumento de lucros.

Nestas situações, o problema consiste nas tomadas de decisão, ou seja, qual a ordem de produção, qual produto deve ser fabricado e em qual quantidade, qual a relação de gasto e benefício para cada produto. Assim, o planejamento da produção influencia significativamente na qualidade do serviço e no custo necessário para todo o processo.

Estas questões têm como solução a otimização do processo realizado pela empresa, segundo Chaves (2009 *apud* BLUM E ROLI, (2008), p. 27) um problema de Otimização P, pode ser descrito como uma tripla (S, Ω, f), tal que:

1. S é o espaço de busca definido sobre um conjunto finito de variáveis de decisão $X_i, i = 1, 2, ..., n$. Caso essas variáveis tenham domínios discretos, P é chamado problema de otimização discreta (ou problema de otimização combinatória), e em casos de domínios contínuos trata-se de um problema de otimização contínua. Problemas com variáveis mistas também existem;

2. Ω é um conjunto de restrições entre as variáveis;

3. $f : S \to \mathbb{R}^+$ é a função objetivo que especifica um valor positivo para cada elemento (ou solução) de S.

A fim de resolver ocasiões práticas e reais, modelos matemáticos são utilizados para a organização de produção de indústrias, uma vez que estes modelos permitem a obtenção de decisões que retornem um custo mínimo ou mesmo a maximização dos lucros.

A Pesquisa Operacional contempla os modelos e métodos de programação matemática, os quais se adequam aos mais diversos problemas de planejamento, como por exemplo, o Problema do Caixeiro Viajante, o Problema de Roteamento de Veículos ou ainda o Problema de Programação de Horários.

Estudamos aqui, o caso de uma indústria que fabrica tubetes, sendo que a ordem de confecção dos mesmos pode retornar um custo muito alto ao final da produção. O estudo de tal problema foi inicializado em Fenato (2008), tendo sua modelagem feita através de um Problema do Caixeiro Viajante e sua generalização com resolução obtida por meio de uma modelagem exata.

Neste trabalho é feita uma abordagem heurística, segundo a qual foi desenvolvido um programa que tem como objetivo minimizar o custo de produção, aumentando assim os lucros. Ainda, para este trabalho o cálculo dos custos das sequências de produção foi reformulado, permitindo-nos apresentar resultados reais que forneçam benefícios concretos à empresa.

Feitas estas considerações, a relevância deste trabalho encontra-se não apenas na adequação de uma situação real para um modelo matemático, mas também na utilização da matemática computacional para alcançar resultados satisfatórios. Cabe ainda, ressaltar a relação estreita entre a matemática e a área computacional, uma vez que para a resolução de problemas de planejamento não é possível separar tais ferramentas.

No capítulo 1 apresentamos a definição do Problema do Caixeiro Viajante Generalizado, bem como suas formulações e ainda técnicas de redução para um Problema do Caixeiro Viajante.

O capítulo 2 contextualiza o problema da Máquina Tubeteira, apresentando uma modelagem por meio do Problema do Caixeiro Viajante e sua generalização, e ainda propomos um novo cálculo dos custos.

As técnicas heurísticas utilizadas para a resolução do problema são apresentadas no Capítulo 3, no qual encontram-se os algoritmos e considerações sobre tais métodos.

O capítulo 4 contempla as características do programa desenvolvido por meio de um ambiente de interface gráfica, juntamente com suas opções e funcionalidades.

Ainda, o capítulo 5 é constituído pelos resultados obtidos pelas heurísticas implementadas em linguagem C^{++}.

Por fim, o apêndice A apresenta noções básicas sobre grafos e o apêndice B é formado pelas carteiras utilizadas para os testes numéricos.

CAPÍTULO 1
O PROBLEMA DO CAIXEIRO VIAJANTE GENERALIZADO

Neste capítulo apresentaremos a definição e formulações matemáticas para o Problema do Caixeiro Viajante Generalizado (PCVG), suas principais aplicações e como tal problema surgiu na literatura. Serão descritos também mecanismos de redução de um PCVG para um PCV e algumas técnicas utilizadas para sua resolução.

A necessidade e importância do presente capítulo são atribuídas ao fato do Problema do Caixeiro Viajante Generalizado ser a base teórica para tratar o problema de minimização do tempo de preparação de uma máquina tubeteira, de modo que, tal contextualização será descrita e abordada no capítulo seguinte.

1.1 INTRODUÇÃO

O Problema do Caixeiro Viajante foi citado pela primeira vez por (Hassler Whitney, 1934), em um trabalho na Princeton University, e é um dos problemas mais tradicionais e estudados da otimização Combinatória. O PCV consiste em visitar n clientes, exigindo-se partir de um depósito central e visitar cada cliente exatamente uma vez. O objetivo de tal problema é que essa sequência de visitas seja feita por meio de um ciclo Hamiltoniano de custo mínimo (Apêndice A).

O Problema do Caixeiro Viajante Generalizado é uma extensão do clássico Problema do Caixeiro Viajante, no qual os vértices estão divididos em grupos, denominados *clusters*, e tem como objetivo visitar pelo menos um vértice de cada *cluster*, por meio de um ciclo Hamiltoniano de custo mínimo. Temos ainda uma outra versão deste problema conhecida como E-PCVG, com a restrição adicional de que exatamente um vértice de cada *cluster* seja visitado.

Na generalização de um PCV temos ainda os casos de PCVG simétrico e assimétrico, os quais serão discutidos no decorrer do trabalho.

Tais problemas adaptam-se facilmente a vida real, em situações como entregas de encomendas, transporte escolar, visitas a clientes e ainda sequenciamento de tarefas, como por exemplo o Problema da Tubeteira, Fenato (2008), que é o problema abordado nesta dissertação.

O PCVG pertence à classe dos problemas NP-difícil, pois o mesmo pode ser interpretado como um PCV quando temos apenas um vértice em cada *cluster* e sendo o PCV um problema NP-difícil (Garey and Johnson, 1979) fica claro tal dedução.

1.2 Revisão Bibliográfica

Dantzig *et al.* (1954) apresentaram uma formulação para o PCV, a qual será descrita ainda neste capítulo, Laporte e Nobert (1973), desenvolveram métodos exatos para a resolução deste problema.

Lin e Kernighan (1973) utilizaram métodos heurísticos, e ainda Golden *et al.* (1980) contribuíram com heurísticas para solucionar o problema do Caixeiro Viajante.

Balas e Christofides (1981) trabalharam relaxações Lagrangeanas para o PCV.

Adrabinski (1983) realizou experimentos computacionais para heurísticas referentes a tal problema.

Recentemente, vem-se desenvolvendo heurísticas e metaheurísticas, para auxiliarem na resolução do PCV, uma vez que em grandes instâncias é impossível resolvê-lo por meio de uma modelagem exata.

O Problema do Caixeiro Viajante Generalizado (PCVG) foi inserido por Henry - Labordere (1969), onde foi apresentada uma aplicação no sequenciamento de arquivos de computador, Srivastava *et al.* (1969) modelaram um problema de sequenciamento de visitas-auxílio a clientes por agências governamentais e utilizaram programação dinâmica para solucioná-lo.

Podemos encontrar aplicações do PCVG em programação de processos de máquinas em indústrias, roteamento postal e layout de redes, (Noon 1988) e (Noon e Bean 1991).

Em Noon e Bean (1991), foi desenvolvida uma formulação exata para o PCVG.

Muitos estudos sobre o PCVG e suas aplicações foram realizados com o intuito de desenvolverem métodos de redução para um PCV, pois o mesmo possui grande quantidade de estudos e heurísticas envolvendo-o. A primeira técnica de redução de um PCVG para um PCV foi realizada por Lien e Ma (1983). Dimitrijevic e Saric (1997) também contribuíram para o estudo desta transformação.

1.3 Considerações sobre PCVG e Equality-PCVG

O Problema do Caixeiro Viajante Generalizado é definido sobre um grafo valorado $G = (V, E)$, onde V é o conjunto formado por *clusters* (conjuntos de vértices) a serem visitados e E o conjunto composto pelas arestas (i, j).

O conjunto V é a união de m *clusters*, tais que $V = C_1 \cup C_2 \cup ... \cup C_m$ onde $C_i \cap C_j = \emptyset$, para todo $i, j, i \neq j$.

Definimos um Problema do Caixeiro Viajante Generalizado como E-PCVG, quando o mesmo tem a exigência de visitar cada *cluster* exatamente uma vez, ou ainda, visitar exatamente um vértice de cada *cluster*.

As arestas pertencentes ao conjunto E são definidas apenas entre vértices de *clusters* distintos, isto é, não existem arestas que interligam vértices de um mesmo *cluster* (arestas *intraclusters*). Tal restrição deve-se ao fato de estarmos estudando o E-PCVG. Ainda, a cada aresta $(i, j) \in E$, associa-se um custo não negativo c_{ij}.

Definição 1. *Uma rota T de um grafo G é um subgrafo de G que é um ciclo sobre V, isto é, se $T = (V, E_T)$, então cada vértice de T é de grau 2, T é conexo e $|E_T| = |V|$.*

Observação 1. *Note que, $|E_T| = |V|$, significa dizer que o número de vértices é igual ao número de arestas em uma rota.*

As soluções factíveis são todas as rotas de G, assim, assumindo que exista pelo menos uma rota, o objetivo do E-PCVG pode ser definido como o problema de determinar uma rota de custo mínimo com m arestas, que inclui exatamente um vértice de cada *cluster*.

Temos então que determinar primeiramente a ordem dos *clusters* a serem visitados e feito isto definir qual vértice será visitado em cada *cluster*, de modo que o custo total seja minimizado.

Como o PCVG, também temos o E-PCVG pertencente à classe dos problemas NP-difícil.

1.3.1 Equivalência entre PCVG e Equality-PCVG

Os casos PCVG e E-PCVG tornam-se equivalentes quando os custos do problema em questão satisfazem a desigualdade triangular para qualquer tripla de vértices (i, j, k).

Figura 1.1: Rota factível PCVG

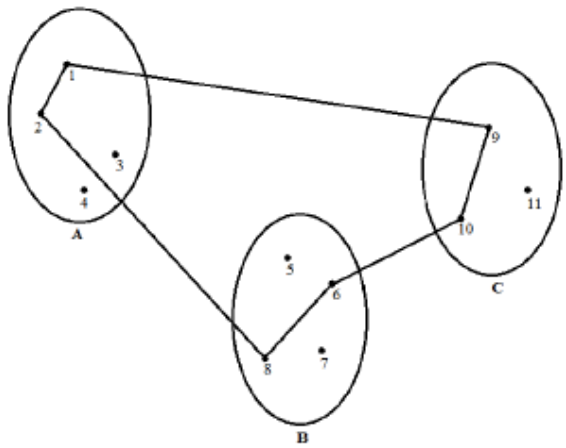
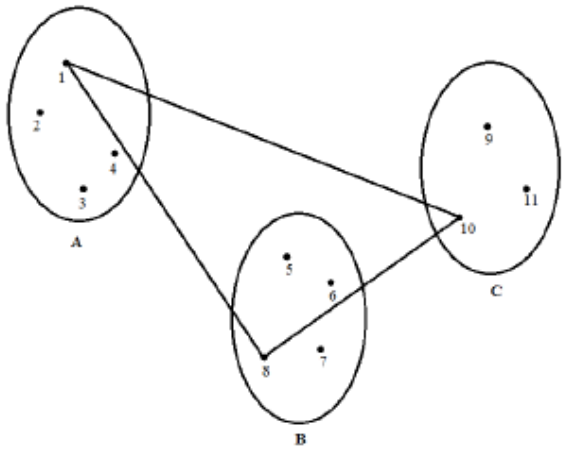


Figura 1.2: Rota factível Equality-PCVG e PCVG



Se os custos satisfazem a desigualdade triangular para qualquer tripla de vértices, em particular, para $(1, 2, 8)$, $(8, 6, 10)$ e $(10, 9, 1)$ temos $c_{18} \leq c_{12} + c_{28}$. Assim, escolheremos

apenas a aresta $(1,8)$ ao invés de $(1,2)$ e $(2,8)$, pois a condição de que pelo menos um vértice de cada *cluster* seja visitado é satisfeita, e então podemos escolher entre as duas possibilidades. Analisando da mesma forma, temos $c_{810} \leq c_{86} + c_{610}$ e $c_{101} \leq c_{109} + c_{91}$, obtendo assim a rota $R_1 = (1,8,10,1)$ que totaliza um custo menor em relação a rota $R_2 = (1,2,8,6,10,9,1)$. Assim escolheremos sempre a rota que visita exatamente um vértice de cada *cluster* (E-PCVG).

1.4 PCVG Simétrico e Assimétrico

O PCVG simétrico é definido sobre um grafo não-direcionado $G = (V, E)$, onde os custos relacionados às arestas pertencentes ao conjunto E satisfazem a igualdade $c_{ij} = c_{ji}$ para todos os vértices $i, j \in V$.

O PCVG assimétrico é definido sobre um grafo direcionado $G = (V, E)$, onde os custos relacionados às arestas diferem entre si em relação a ordem em que os vértices são visitados, isto é, $c_{ij} \neq c_{ji}$ para algum (i, j), e ainda utilizando a aresta (i, j) ao invés de (j, i), a função objetivo retornará valores distintos.

1.5 Formulações Matemáticas

Nesta seção apresentamos as formulações de Noon e Bean (1989) e Salazar, Fischetti e Toth (1995) que descrevem o PCVG, bem como uma discussão abrangendo semelhanças e diferenças entre as duas formulações.

O Problema do Caixeiro Viajante Generalizado é modelado como um Problema de Programação Linear Inteira, no qual é definida uma variável binária que corresponde à tomada de decisão de um vértice ou aresta pertencer ou não a solução obtida.

1.5.1 Formulações de Noon e Bean

Noon e Bean (1989) apresentam uma formulação para o E-PCVG, na qual é definida a variável binária x_{ij} onde:

$$x_{ij} = \begin{cases} 1, & \text{se a aresta } (i,j) \text{ faz parte da solução} \\ 0, & \text{caso contrário.} \end{cases}$$

$$\min z = \sum_{(i,j) \in E} c_{ij} x_{ij} \tag{1.1}$$

sujeito a:

$$\sum_{i \in C_I} \sum_{j \notin C_I} x_{ij} = 1, \text{ para todos os conjuntos } C_I \tag{1.2}$$

$$\sum_{i \notin C_I} \sum_{j \in C_I} x_{ij} = 1, \text{ para todos os conjuntos } C_I \tag{1.3}$$

$$\sum_{\substack{i \in V, \\ (i,j) \in E}} x_{ij} - \sum_{\substack{k \in V, \\ (j,k) \in E}} x_{jk} = 0, \text{ para todo } j \in V, \tag{1.4}$$

$$\sum_{i \in \Omega} \sum_{i \in C_I} \sum_{j \notin \Omega} \sum_{j \in C_J} x_{ij} \geq 1, \text{ para todo conjunto } \Omega, \tag{1.5}$$

tal que, $2 \leq |\Omega| \leq m - 2$, subconjuntos próprios da coleção de *clusters*.

$$\sum_{(i,j) \in C_I} x_{ij} = 0, \ I = 1, ..., m \tag{1.6}$$

$$x_{ij} \in \{0, 1\}, \text{ para todo } (i, j) \in E, \tag{1.7}$$

A função objetivo 1.1 é assim definida a fim de obter-se uma sequência de vértices a serem visitados, com um custo total mínimo.

As restrições 1.2 e 1.3 são impostas para garantir que apenas uma aresta chegue e uma aresta saia de cada *cluster*.

Outra restrição necessária é a de garantir que a solução seja ininterrupta, para isto definimos a restrição 1.4, ou seja, para todo $i \in V$ tal que $x_{ij} = 1$ deve existir $k \in V$ onde $x_{jk} = 1$, assim garantimos que há uma aresta de chegada e outra de saída no mesmo vértice do *cluster*. Por outro lado se $x_{ij} = 0$ deve existir $k \in V$ onde $x_{jk} = 0$, isto é, se não há aresta de chegada no vértice j, não pode haver aresta de saída deste vértice.

Já a equação 1.5 não permite que sejam aceitas soluções com subrotas entre os *clusters*. A imposição de que $2 \leq |\Omega| \leq m - 2$, é de fácil entendimento já que, se tivermos um subconjunto com $m - 1$ elementos não haverá uma subrota, pois o ponto separado do conjunto não terá um outro vértice para se ligar, não formando assim uma subrota. Logo, este caso não precisa ser analisado. Por outro lado, para conjuntos com $m - 2$ ou menos elementos conseguimos formar no mínimo duas subrotas, pois teremos os elementos do conjunto Ω interligados e os elementos do conjunto formado pelos vértices restantes, que poderão ser ligados uma vez que tal conjunto é formado por dois ou mais elementos.

No caso E-PCVG exatamente um vértice de cada *cluster* deve ser visitado, assim para que não existam arestas entre dois vértices de um mesmo *cluster* é imposta a condição 1.6.

E em 1.7 é exigida a binaridade da variável x_{ij}.

Para a modelagem do PCVG basta substituir 1.6 pela seguinte equação que permite a existência de arestas entre o mesmo *cluster*:

$$\sum_{(i,j) \in C_I} x_{ij} \geq 0, \ I = 1, ..., m.$$

1.5.2 Formulação de Salazar, Fischetti e Toth

Em Salazar, Fischetti e Toth (1995), outra formulação é apresentada para o E-PCVG, e para tal serão introduzidas as notações utilizadas neste contexto:

Para cada $S \subseteq N$, consideremos $E = \{(i,j) : i,j \in N, i \neq j\}$ o conjunto de arestas e c_e o custo associado a cada aresta $e \in E$ e os conjuntos:

$$E(S) = \{[i,j] \in E : i \in S, j \in S\},$$

$$\delta(S) = \{[i,j] \in E : i \in S, j \notin S\},$$

$$\mu(S) = |\{h : C_h \subseteq S\}|,$$

$$\nu(S) = |\{h : C_h \cap S \neq \emptyset\}|.$$

Para $v \in N$ denotamos $C_{h(v)}$ o *cluster* que contém apenas o vértice v e $W = \{v \in N : |C_{h(v)}| = 1\}$.

Considere, $$y_v = \begin{cases} 1, & \text{se o vértice } v \in N \text{ foi visitado} \\ 0, & \text{caso contrário.} \end{cases}$$

Assim, segue o modelo de programação linear inteira proposto pelos autores:

$$\min v(E - PCVG) = \sum_{e \in E} c_e x_e \tag{1.8}$$

sujeito a:

$$\sum_{e \in \delta(v)} x_e = 2y_v, \text{ para } v \in N \tag{1.9}$$

$$\sum_{v \in C_h} y_v = 1, \text{ para } h = 1, ..., m \tag{1.10}$$

$$\sum_{e \in \delta(S)} x_e \geq 2(y_i + y_j - 1), \text{ para todo } S \subset N, 2 \leq |S| \leq n-2, i \in S, j \in (N/S) \tag{1.11}$$

$$x_e \in \{0,1\}, \text{ para } e \in E \tag{1.12}$$

$$y_v \in \{0,1\}, \text{ para } v \in N \tag{1.13}$$

Em 1.9, temos $\delta(v) = \{(v,j) \in E : v \in \{v\} \text{ e } j \notin \{v\}\}$ e ao percorrermos este conjunto por meio somatório, para cada $v \in N$ eliminamos todas as arestas do tipo (v,v). Garantimos também que, se v é visitado, teremos duas arestas, uma de chegada em v e outra de saída de v.

A restrição 1.10 garante que exatamente um vértice de cada *cluster* seja visitado.

Para 1.11, já foi discutido o motivo de exigir os conjuntos S com tal cardinalidade, esta hipótese, impõe ainda que para cada subconjunto separando dois vértices ($2 \leq |S| \leq n-2$), deve-se ter pelo menos duas arestas passando por este conjunto (uma de chegada no conjunto e uma de saída do conjunto). Desse modo, são eliminadas as subrotas.

Por fim, 1.12 e 1.13 garantem a integralidade de x_e e y_v.

Para a versão PCVG, basta substituirmos na formulação acima, a equação 1.10 por

$$\sum_{v \in C_h} y_v \geq 1, \text{ para } h = 1, ..., m,$$

e assim pelo menos um vértice de cada *cluster* será visitado.

1.5.3 Considerações

A formulação de Noon e Bean (1989) define apenas uma variável binária x_{ij}, a qual garante o fluxo contínuo na solução e também que em cada vértice exista exatamente uma aresta de chegada e uma de saída. Já a formulação de Salazar, Fischetti e Toth (1995) utiliza-se de duas variáveis binárias, x_e e y_v que analisa se um vértice foi ou não visitado, afim de garantir a continuidade da rota e que todos os vértices tenham grau 2. A variável binária x_{ij} é definida segundo seus índices $i, j \in V$, que referem-se ao vértice j ser ou não visitado imediatamente após o vértice i, e por sua vez a variável binária x_e é determinada pelo fato da aresta e ser percorrida (utilizada) na solução, uma vez que são considerados como índices as arestas $e \in E$

Em relação às restrições de eliminação de subrotas ambas as formulações consideram subconjuntos da coleção de *cluster* com a mesma cardinalidade, citada nas equações 1.5 e 1.11. A restrição de Salazar, Fischetti e Toth (1995) analisa o número de arestas incidentes em conjuntos que separam pelo menos dois vértices, enquanto Noon e Bean (1989) garantem que quaisquer dois subconjuntos Ω da coleção de *cluster* sejam ligados por meio de pelo menos uma aresta.

1.6 Um Caso Particular

Quando temos exatamente um vértice em cada *cluster*, obtemos o Problema do Caixeiro Viajante (PCV), que pode ser analisado como um caso particular do PCVG.

O PCV tem sido amplamente estudado devido a pelo menos três de suas características: possui grande aplicação prática, grande dificuldade de obtenção da solução exata, e uma significativa relação com outros modelos (Laporte e Martello, 1990).

No século XIV o matemático irlandês Sir William Rowan Hamilton e o matemático inglês Thomas Penyngton Kirkman relataram dificuldades sobre o Problema do Caixeiro Viajante e em 1857 Hamilton inventou o jogo icosiano e o vendeu para um comerciante em Londres, de modo que o jogo foi comercializado por toda a Europa.

Observação 2. *O jogo icosiano é um quebra-cabeças que consiste em passar por todos os vértices apenas uma vez. Caminhos dessa forma são definidos hoje como Ciclos Hamiltonianos.*

Por volta de 1930, o estudo do PCV foi iniciado por Karl Menger em Vienna e Harvard, no ano de (1934), Hassler Whitney citou o PCV em um trabalho na Universidade de Princeton.

Descreveremos a formulação de Dantzig-Fulkerson-Johnson para o PCV, a qual aborda o mesmo como um problema de programação $0-1$ sobre um grafo valorado $G = (V, E)$ (Goldbarg e Luna, 2000).

$$\min z = \sum_{j=1}^{n} \sum_{i=1}^{n} c_{ij} x_{ij} \tag{1.14}$$

sujeito a:

$$\sum_{i=1}^{n} x_{ij} = 1, \text{ para todo } j \in N \tag{1.15}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \text{ para todo } i \in N \tag{1.16}$$

$$\sum_{i,j \in S} x_{ij} \leq |S| - 1, \text{ para todo } S \subset N \tag{1.17}$$

$$x_{ij} \in \{0, 1\}, \text{ para todo } (i, j) \in N \tag{1.18}$$

A variável x_{ij} assume valor igual a 1 (um) se a aresta (i, j) é usada na solução e 0 (zero) caso contrário.

As restrições 1.15 e 1.16 garantem respectivamente, que para cada i e para cada j tem-se exatamente uma aresta incidente j e uma aresta incidente i. Já a equação 1.17 elimina soluções que possuem subrotas, como podemos ver nas figuras 1.3 e 1.4:

Figura 1.3: Solução não-viável (subrotas)

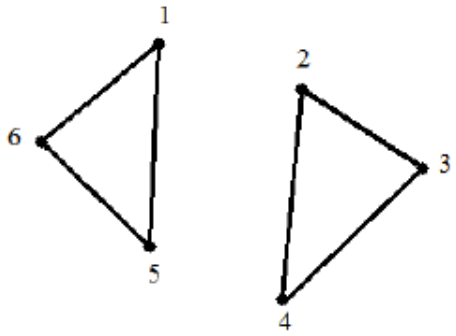
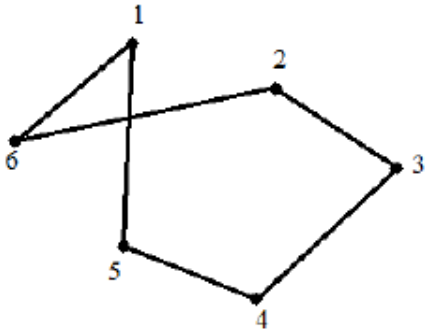


Figura 1.4: Solução viável



Na Figura 1.3, se considerarmos o subconjunto $S = \{1, 5, 6\} \subset N$, temos $\sum_{i,j \in S} x_{ij} = 3$, ou seja a restrição de eliminação de subrotas não é satisfeita, e desse modo o programa não aceitará esta rota como solução. Por outro lado, na Figura 1.4, para qualquer subconjunto $S \subset N$ temos a restrição 1.17 satisfeita.

A implementação do PCV pode-se dar de maneira exata ou heurística. Para resolvermos o problema de maneira exata podemos utilizar, por exemplo, o método *Branch and Bound*, e para soluções heurísticas podemos considerar as heurísticas de construção, refinamento, composição de rotas e ainda as metaheurísticas.

1.7 Redução de um PCVG para um PCV

Como já foi citado, o Problema do Caixeiro Viajante Generalizado pode ser adaptado a várias situações reais, contudo muitos pesquisadores hesitam em aplicá-lo na resolução de seus problemas devido sua complexidade para encontrar uma solução ótima ou próxima da ótima. Neste contexto, alguns autores desenvolveram métodos com a finalidade de transformar um PCVG em um PCV.

A ideia de redução de um PCVG em um PCV pode justificar-se em dois aspectos básicos: primeiramente o PCVG é uma extensão do PCV e portanto possuem similaridades e ainda, muitos dos métodos eficientes existentes para o PCVG são extensões de métodos desenvolvidos para o PCV.

Lien e Ma (1993) abordaram e introduziram na literatura a transformação de um PCVG em um PCV, porém em tal estudo o número de vértices do PCV obtido era muito grande, chegando até mesmo a ser três vezes o número de vértices do PCVG associado. Temos também a contribuição de Dimitrijevic e Saric (1997), os quais diminuíram o número de vértices do PCV correspondente para duas vezes o número de vértices do PCVG original (Behzad, Modares, 2002, p. 1).

O método proposto a seguir foi desenvolvido por Behzad e Modares (2002), no qual o número de vértices é o mesmo nas versões do PCVG e do PCV associado.

Definição 2. *Define-se como g-rota cada solução factível que é um caminho fechado e inclui pelo menos um vértice de cada cluster.*

Os autores assumem que o grafo do PCVG é direcionado e que não existem arestas *intra-clusters* (que ligam vértices de um mesmo *cluster*) no grafo, supõe-se ainda que não há interseção entre os *clusters* e o objetivo é determinar uma g-rota mínima que visita cada *cluster* exatamente uma vez.

Segundo Behzad e Modares (2002 *apud* LIEN E MA, 1983, p. 2) as hipóteses não são restritivas e todo PCVG pode ser transformado neste problema.

Definição 3. *Considere um PCVG com p clusters, representado pelo grafo $G = (V, E)$. Seja v_i^r o i-ésimo vértice do cluster r. Definimos um PCV sobre um grafo direcionado G', associado com G, tal que,*

1. *O conjunto de vértices de G e G' são idênticos;*
2. *Todos os vértices de cada cluster de G são conectados por arestas em um ciclo de G'. O vértice que sucede v_i^r no ciclo é denotado por $v_{i(s)}^r$;*
3. *Os elementos da matriz de custo de G' são definidos como*

$$c'(v_i^r, v_{i(s)}^r) = 0 \tag{1.18}$$

$$c'(v_i^r, v_j^t) = c'(v_{i(s)}^r, v_j^t) + M, \quad r \neq t \tag{1.19}$$

onde

$$M > \sum_{(i,j)\in E} c(i,j). \tag{1.20}$$

Baseados em 1.19, os autores afirmam que se o custo $c'(v_i^r, v_j^t)$ tornar-se "infinito", tem-se a implicação de que v_i^r não está conectado a v_j^t.

Definição 4. *Um caminho em G' é definido como um "caminho cluster"se consiste de todos os vértices de um único cluster.*

Definição 5. *Um ciclo Hamiltoniano em G' cujo custo é menos que $(p+1)M$ é definido como um "ciclo Hamiltoniano clusterizado"(ou ciclo C-Hamiltoniano).*

Lema 1. *Todo ciclo C-Hamiltoniano atravessa apenas ao longo de um caminho cluster, ou seja, chega e sai de cada cluster exatamente uma vez.*

Demonstração: Pela definição da matriz de custos de G', temos que o custo de mover-se de um *cluster* a outro é pelo menos M. Por outro lado, sendo que um ciclo Hamiltoniano visita todos os *clusters* seu custo será no mínimo pM.
Porém, se não passarmos ao longo dos *"caminhos cluster"*, visitaremos pelo menos um *cluster* mais de uma vez. Isto implica que o custo total do ciclo Hamiltoniano não pode ser menor que $(p+1)M$, o que completa a demonstração.

□

Definição 6. *Definimos uma correspondência um a um entre os ciclos C-Hamiltonianos em G' e as g-rotas em G da seguinte forma:*

1. *Considere um ciclo C-Hamiltoniano de G' e conecte os primeiros vértices de seus "caminhos clusters"aos clusters correspondentes. O resultado será uma g-rota do PCVG (mais precisamente E-PCVG);*
2. *Considere uma g-rota em G que inclua $... \to v_i^r \to v_j^t \to ..., r \neq t$.*
   Substitua v_i^r pelo r-ésimo "caminho cluster"de G' começando por v_i^r, e então conecte o último vértice deste caminho ao próximo "caminho cluster"que inicia-se com v_j^t.

O resultado obtido pelo procedimento descrito forma um ciclo Hamiltoniano.

Teorema 1. *O custo de toda g-rota em G é igual ao custo do ciclo C-Hamiltoniano em G' menos pM.*

Demonstração: Segue diretamente das definições 3 e 6 e do lema 1.

Observação 3. *Uma vez que pM é constante não interfere na solução ótima do PCVG.*

CAPÍTULO 2
CONTEXTUALIZAÇÃO DO PCVG E O PROBLEMA DA TUBETEIRA

Este capítulo apresenta o problema do tempo de preparação de uma máquina Tubeteira (utilizada para confecção de tubetes) da empresa Sonoco de Londrina, descrevendo características da tubeteira e dos tubetes.

O estudo de tal problema iniciou-se em Fenato (2008), visto que na literatura não foi encontrado nenhum trabalho que tratasse do problema especificamente, o qual foi modelado por meio dos Problemas do Caixeiro Viajante (PCV) e Caixeiro Viajante Generalizado (PCVG) e solucionado de maneira exata. Ainda não existem bibliografias que contemplem alguma resolução para o problema, porém considerando o trabalho inicial de Fenato (2008) buscamos por meio da abordagem heurística apresentar uma solução satisfatória à indústria em questão.

O presente trabalho apresenta além de uma modelagem que condiz com a realidade da empresa, uma interface gráfica que permitirá à empresa obter sequências de fabricação de tubetes que minimizem o tempo de preparação da máquina tubeteira sem a necessidade de utilizar um *solver*, ou mesmo aceitar soluções inferiores para o caso de carteiras com um número elevado de tubetes a serem confeccionados.

Os dados citados nas seções 2.1, 2.2 e 2.3 são baseados na dissertação de Fenato (2008).

2.1 TUBETES

Os tubetes (Figura 2.1) são fabricados pela sobreposição ordenada de várias camadas de papel reciclado, coladas umas sobre as outras com tensões tecnicamente determinadas, formando assim um tubo rígido e oco, cuja espessura e rigidez dependem principalmente da quantidade de camadas e do tipo de papel utilizado em sua fabricação. Tais camadas de papel são provenientes de bolachas (Figura 2.2), que são fitas de papel reciclado enroladas em formato de grandes bobinas. Os tubetes diferenciam-se pela resistência à pressão (rigidez), espessura da parede, diâmetro interno e comprimento.

Podemos destacar como principais finalidades do tubete, seu uso como embalagem de

Figura 2.1: Tubete

Figura 2.2: Bolachas

proteção e conservação dos produtos de outras empresas e como barreira rígida para produtos com embalagens mais frágeis às pressões mecânicas, evitando que mercadorias sejam fisicamente danificados enquanto são deslocados de um local para outro.

O número de camadas de papel, as quais diferem segundo a largura, espessura, gramatura e composição será determinado de acordo com a aplicação do tubete e o pedido do cliente. Tal número é limitado pela capacidade da tubeteira, isto é, o número máximo de bolachas que a mesma suporta ao decorrer da confecção de um tubete. A tubeteira da Sonoco, comporta um número máximo de 34 bolachas para confeccionar um tubete.

A tabela 2.1 apresenta a nomenclatura das camadas e suas respectivas posições na constituição de um tubete.

Tabela 2.1: Camadas de papel constituintes do tubete.

Nome	InsidePly (face interna)	Miolo	TopPly	Primeira Cobertura	Segunda Cobertura
Posição	1ª	2ª a 31ª	Ante - penúltima	Penúltima	Última
Número máximo de camadas	1	30	1	1	1

Fonte: Fenato, 2008.

As posições das camadas de papel de um tubete são as mesmas das bolachas na tubeteira, as quais devem ser posicionadas no estaleiro respeitando a ordem da configuração de um tubete da esquerda para a direita. Conforme informações técnicas, o número mínimo de bolachas usadas na produção de um tubete qualquer é três, sendo uma para a posição *inside ply*, outra para a posição *top ply* e uma para a primeira cobertura.

2.2 TUBETEIRA

A tubeteira é composta pelo estaleiro, o tanque de cola e a sobrepositora.

Depois de ordenadas segundo a composição do tubete, as bolachas são alocadas no estaleiro (Figura 2.3), para então serem desenroladas até o tanque de cola (Figura 2.4).

O tanque de cola, é formado por um reservatório na parte inferior, onde deposita-se a cola que será elevada e derramada sobre as fitas de papel que passam por entre as duas palhetas.

A sobrepositora (Figura 2.5) recebe as fitas de papel que passam pelo tanque de cola e as sobrepõem, uma a uma, com uma tensão adequada, formando um tubo de comprimento "infinito", do qual serão cortados os tubetes numa serra circular acoplada à sobrepositora.

Figura 2.3: Estaleiro

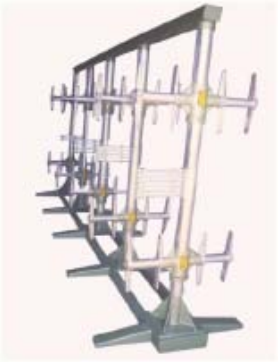


Figura 2.4: Tanque de cola

Figura 2.5: Sobrepositora

2.3 DESCRIÇÃO DO PROBLEMA

No processo de fabricação dos tubetes, o tempo de preparação da máquina tubeteira representa em média 40% (quarenta por cento) do tempo total de produção, conforme dados técnicos fornecidos pela empresa, isto se deve ao fato de várias bolachas serem trocadas entre a fabricação de dois tubetes. Desse modo, a otimização desse processo consiste em minimizar o número de trocas de bolachas necessárias entre a confecção dos tubetes.

Será considerado ainda o diâmetro do mandril, o qual define o diâmetro interno de um tubete. A troca de mandril é usualmente feita após a confecção de todos os tubetes com o mesmo diâmetro interno, pois o custo operacional (tempo gasto) para a troca de um mandril é significativamente superior ao custo da troca de uma bolacha.

Fenato (2008) e Hoto *et al.* (2010) abordaram o problema da Tubeteira de duas formas: primeiramente como um Problema do Caixeiro Viajante, considerando as configurações originais dos tubetes e como um Problema do Caixeiro Viajante Generalizado na versão *Equality*, permitindo a existência de uma posição vazia no estaleiro com o objetivo de minimizar as trocas de bolachas entre a fabricação dos tubetes.

Assim, o problema da Tubeteira é definido sobre um grafo valorado $G = (V, E)$, onde V é o conjunto de tubetes (vértices) a serem fabricados e E o conjunto das arestas (i, j) que representam a fabricação do tubete j logo após a produção de i. A cada elemento de E foi associado um custo c_{ij}, que representa o número de trocas e/ou movimentos de bolachas entre os tubetes i e j. Ainda, G trata-se de um grafo completo, uma vez que ao ser fabricado um tubete qualquer i pode-se confeccionar qualquer outro tubete j.

Associamos à troca de dois mandris M_i e M_j, um custo m_{ij} elevado, a fim de que a mesma seja evitada enquanto houver tubetes com o mesmo diâmetro a serem fabricados, ou seja, serão produzidos todos os tubetes com o mesmo mandril e somente então haverá a troca.

Nosso objetivo é determinar um ciclo Hamiltoniano de custo mínimo e para isto definimos um tubete (vértice) fictício O, que representa o vértice inicial e final da sequência de tubetes a serem fabricados sendo que os custos da forma $c_{kO} = c_{Ok} = 0$, para qualquer tubete k.

2.4 Definição da Matriz de Custos

Nesta seção serão descritas duas maneiras para que sejam calculados os custos associados às arestas (i, j).

Em Fenato (2008) o cálculo de custos foi feito considerando-se apenas trocas de bolachas e propomos nesta dissertação analisar os custos segundo os movimentos de bolachas.

2.4.1 Custo em Relação à Trocas de Bolachas

A proposta inicial deste trabalho consistia na implementação de heurísticas para a resolução do problema da Tubeteira, assim, inicialmente os custos eram calculados de maneira análoga à descrita em Fenato (2008).

Neste caso, o custo c_{ij} é associado à aresta (i, j) segundo a quantidade de trocas de bolachas que houve entre a fabricação de dois tubetes i e j.

Por exemplo, considere os tubetes 1 e 2, com as respectivas configurações:

Tabela 2.2: Trocas de bolachas entre os tubetes 1 e 2.

Tubete	Configuração
1	10 20 60 60 90 90 100
2	10 30 30 30 40 40 100 90

Na tabela acima, os círculos em vermelho representam as trocas de bolachas necessárias para a fabricação do tubete 2 imediatamente após a confecção do tubete 1, ocorrendo assim 6 trocas.

Este modelo para cálculo de custos é razoável para um modelo inicial, porém não condiz com a realidade da empresa em questão, uma vez que o tempo dispendido para o posicionamento das novas bolachas depende das posições em que as mesmas encontram-se no estaleiro.

Para abranger esta situação, é desenvolvido nesta dissertação uma nova abordagem referente ao cálculo da matriz de custos.

2.4.2 Custo em Relação à Movimentos de Bolachas

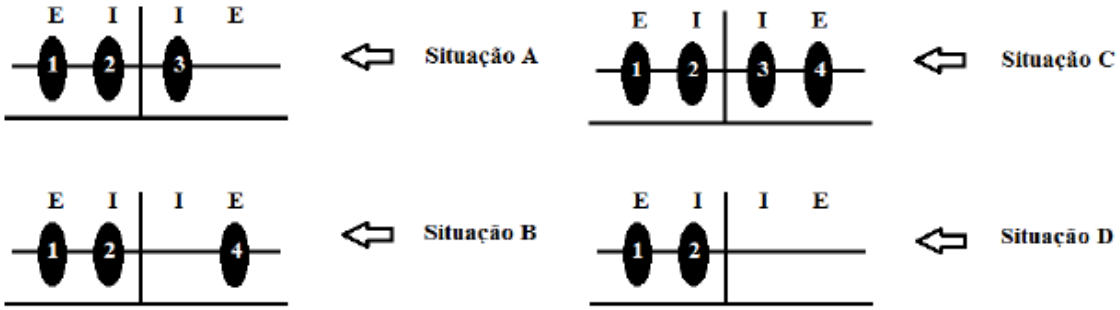
A fim de apresentar um modelo que descreva e solucione as reais necessidades da empresa, propomos o cálculo dos custos considerando a quantidade de movimentos necessários para as trocas das bolachas que compõem as configurações dos tubetes a serem fabricados sequencialmente.

Neste momento, fazem-se necessárias informações sobre a alocação das bolachas no estaleiro, sendo que a troca ou reposicionamento de uma bolacha dependerá fortemente da colocação que ocupa no mesmo.

O estaleiro é dividido em fileiras, nas quais é possível alocar no máximo 4 bolachas. Desse modo, as bolachas posicionadas nas extremidades do estaleiro são classificadas como externas e as bolachas que correspondem às posições centrais são denominadas bolachas internas.

Na figura 2.6 apresentamos uma ilustração de possibilidades de alocação das bolachas no estaleiro:

Figura 2.6: Possíveis alocações de bolachas



Sem perda de generalidade, para a análise dos possíveis movimentos realizados em uma bolacha, será considerada apenas a parte direita do estaleiro.

1. Situação A para a situação A, temos duas possibilidades: a bolacha permanecer na posição, o que não realiza movimentos e ainda retirar a bolacha da posição interna 3 e colocar outra boalcha na posição interna 3 o que fornece 2 movimentos.
2. Situação A para a situação B, temos duas possibilidades: apenas deslocar a bolacha da posição 3 (interna) para a posição 4 (externa) o que totaliza 1 movimento, ou ainda, retirar a bolacha alocada em 3 (interna) e colocar outro bolacha na posição 4 (externa), o que retorna um total de 2 movimentos.
3. Situação A para a situação C, temos duas possibilidades: manter a bolacha já alocada na posição 3 e colocar uma nova bolacha na posição externa 4, o que realiza 2 movimentos, ou retirar a bolacha da posição interna 3, colocar uma nova bolacha em 3 e ainda posicionar outra bolacha na posição externa 4, o que totaliza 3 movimentos.
4. Situação A para a situação D, existe apenas a possibilidade da retirada da bolacha posicionada em 3, o que realiza 1 movimento.

5. Situação B para a situação B, também temos duas possibilidades, a trivial, na qual a bolacha permenece alocada sem qualquer movimento e a retirada da bolacha e alocação de outra bolacha na posição externa 4, o que totaliza 2 movimentos.

6. Situação B para a situação A, temos duas possibilidades: deslocar a bolacha da posição externa 4 para a posição interna 3, realizando assim 1 movimento, ou retirar a bolacha posicionada em 4 e colocar outra bolacha na posição 3, fornecendo então 2 movimentos.

7. Situação B para a situação C, pode-se apenas deslocar a bolacha posicionada em 4 para a posição 3 e colocar uma nova bolacha na posição 4, o que retorna 2 movimentos, ou ainda, retirar a bolacha da posição externa 4, e colocar uma bolacha para a posição 3 e outra bolacha para a posição 4, totalizando 3 movimentos.

8. Situação B para a situação D, temos apenas uma solução que é a retirada da bolacha posicionada em 4, o que realiza 1 movimento.

9. Situação C para a situação C, neste caso temos três possibilidades, primeiramente analisamos o caso trivial, em que nenhuma bolacha é retirada de sua posição, e assim não é realizado movimento algum, em segundo podemos ter apenas a troca da bolacha da posição externa 4, assim retira-se a bolacha posicionada em 4 e então coloca-se outra bolacha nesta posição o que totaliza 2 movimentos, e ainda se a troca da bolacha ocorrer apenas na posição interna 3 ou mesmo nas duas posições são necessárias a retirada das bolachas posicionadas em 3 e 4 e a alocação de novas bolachas nestas posições, o que realiza 4 movimentos.

10. Situação C para a situação A, existem duas possibilidades: apenas retirar a bolacha da posição externa 4 realizando assim 1 movimento, ou retirar as bolachas das posições 3 e 4 e então alocar uma bolacha na posição interna 3, o que fornecerá 3 movimentos.

11. Situação C para a situação B, podemos apenas retirar a bolacha da posição externa 4 e deslocar a bolacha da posição interna 3 para a posição externa 4 fornecendo um total de 2 movimentos, ou ainda, a retirada das bolachas posicionadas em 3 e 4 e alocação de uma bolacha na posição 4, o que retorna 3 movimentos.

12. Situação C para a situação D, temos apenas um caso, que é a retirada das bolachas posicionadas em 3 e 4, realizando 2 movimentos.

13. Situação D para a situação D, temos apenas o caso trivial, nenhum movimento é realizado.

14. Situação D para a situação A, existe apenas a possibilidade da alocação de uma bolacha na posição interna 3, o que realiza 1 movimento.

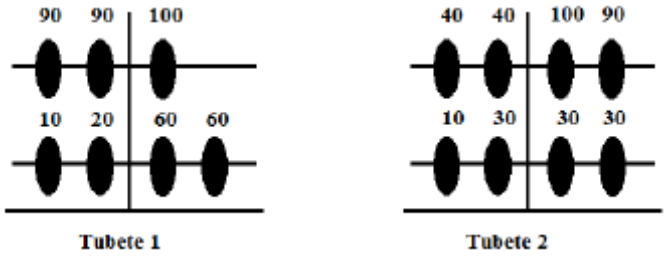
15. Situação D para a situação B, equivalente ao caso anterior, temos apenas um movimento com a colocação de uma bolacha na posição externa 4.

16. Situação D para a situação C, são necessários 2 movimentos para a alocação da bolacha na posição interna 3 e para a alocação da bolacha na posição externa 4.

A situação B, é considerada apenas para o caso em que o problema da Tubeteira é modelado como um E-PCVG, devido a necessidade da posição interna 3 encontrar-se vazia.

Para os tubetes 1 e 2 (Figura 2.7) será analisado o custo necessário para suas fabricações em relação a movimentos de bolachas.

Figura 2.7: Bolachas alocadas - Tubetes 1 e 2



Para a primeira fileira são realizados 8 (oito) movimentos, sendo 4 (quatro) para a troca da bolacha 20 pela bolacha 30 e 4 (quatro) na troca das duas bolachas 60 por duas bolachas 90. Para a segunda fileira, temos um total de 4 movimentos para a troca das bolachas 90 pelas bolachas 40 e ainda mais um movimento para a alocação da bolacha 90 na posição externa 8.

Assim, são necessários 13 movimentos para a fabricação do tubete 2 imediatamente após a confecção do tubete 1.

Ao considerar-se movimentos de bolachas em função de suas posições no estaleiro, a função objetivo retornará um valor superior ao caso em que são computadas apenas trocas. Este fato por sua vez não significa ineficiência ou diminuição na qualidade da modelagem, uma vez que neste caso o problema da Tubeteira está descrito de maneira estritamente próxima à realidade presente na produção de tubetes em sequência.

2.5 Modelando o Problema da Tubeteira

Para o problema da Tubeteira foi proposto em Fenato (2008) que sua modelagem seria feita por meio do Problema do Caixeiro Viajante e sua generalização. Neste trabalho continuamos com tal abordagem, porém algumas considerações foram acrescentadas de modo que todas as restrições do problema em estudo estivessem presentes na formulação e adaptação para um PCV e um PCVG.

Inicialmente, a função objetivo foi considerada de modo que são adicionados não apenas os custos relacionados às trocas e/ou movimentos de bolachas, mas também o custo da troca de mandril, o que direciona o programa a não optar pela troca de mandril em momentos desnecessários, Fenato (2008) solucionou o problema acrescentando um número elevado de bolachas a cada tubete. Ainda, relacionamos o custo com o número máximo de bolachas em cada carteira.

2.5.1 O Problema da Tubeteira como um PCV

Nesta seção o Problema da Tubeteira é modelado como um PCV definido sobre um grafo valorado completo $G = (V, E)$, onde V é o conjunto formado pelos tubetes considerados como vértices que devem ser visitados exatamente uma vez. Nosso objetivo é determinar a ordem de fabricação dos tubetes que apresente o menor número de trocas e/ou movimentos de bolachas, priorizando a confecção sucessiva de tubetes com o mesmo mandril.

Para o cálculo dos custos considerando trocas de bolachas e movimentos de bolachas define-se a variável binária x_{ij} tal que:

$$x_{ij} = \begin{cases} 1, & \text{se a aresta } (i,j) \text{ faz parte da solução} \\ 0, & \text{caso contrário.} \end{cases}$$

2.5.1.1 Modelagem considerando trocas de bolachas

Neste caso, o problema da Tubeteira é apresentado como o seguinte PCV:

$$\min z' = \sum_{j=1}^{n} \sum_{i=1}^{n} x_{ij}(c_{ij} + m_{ij}) \tag{2.1}$$

sujeito a:

$$\sum_{i=1}^{n} x_{ij} = 1, \text{ para todo } j, j = 1, 2, ..., n \tag{2.2}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \text{ para todo } i = 1, 2, ..., n \tag{2.3}$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \text{ para todo } S \subset \{1, 2, ..., n\} \tag{2.4}$$

$$0 \leq c_{ij} \leq B_{max}, \tag{2.5}$$

onde B_{max} é o número máximo de bolachas necessárias para confeccionar um tubete.

$$m_{ij} = \begin{cases} B_{max} + 1, & \text{se } M_i \neq M_j \\ 0, & \text{caso contrário.} \end{cases} \tag{2.6}$$

$$x_{ij} \in \{0, 1\}, \text{ para todo } i, j \in \{1, 2, ..., n\}, i \neq j \tag{2.7}$$

Na função objetivo 2.1, somam-se os custos fictícios relacionados às trocas de mandril, a fim de que tubetes com mandris iguais sejam produzidos em sequência. Assim, para a obtenção do custo total de trocas de bolachas necessários para produzir os tubetes, é necessário eliminar os custos fictícios de trocas de mandril que foram realizadas durante a confecção dos tubetes, ou ainda:

$$z = z' - \sum_{j=1}^{n} \sum_{i=1}^{n} x_{ij} m_{ij} = z' - (M - 1)(B_{max} + 1),$$

uma vez que, se o número de mandris diferentes é M, teremos $(M-1)$ trocas de mandril e o custo relacionado a cada troca de mandril é $(B_{max}+1)$.

É apresentada aqui uma solução distinta para a restrição de que tubetes possuindo o mesmo mandril sejam fabricados em sequência.

2.5.1.2 Modelagem considerando movimentos de bolachas

Para esta situação, o diferencial na modelagem está na limitação dos custos e ainda na atribuição do custo fictício para a troca de mandril. Logo, o problema da Tubeteira é modelado como o seguinte PCV:

$$\min z' = \sum_{j=1}^{n}\sum_{i=1}^{n} x_{ij}(c_{ij}+m_{ij}) \tag{2.8}$$

sujeito a:

$$\sum_{i=1}^{n} x_{ij} = 1, \text{ para todo } j, j = 1,2,...,n \tag{2.9}$$

$$\sum_{j=1}^{n} x_{ij} = 1, \text{ para todo } i = 1,2,...,n \tag{2.10}$$

$$\sum_{i\in S}\sum_{j\in S} x_{ij} \leq |S|-1, \text{ para todo } S \subset \{1,2,...,n\} \tag{2.11}$$

$$0 \leq c_{ij} \leq 2B_{max}, \tag{2.12}$$

onde B_{max} é o número máximo de bolachas necessárias para confeccionar um tubete.

$$m_{ij} = \begin{cases} 2B_{max}+1, & \text{se } M_i \neq M_j \\ 0, & \text{caso contrário.} \end{cases} \tag{2.13}$$

$$x_{ij} \in \{0,1\}, \text{ para todo } i,j \in \{1,2,...,n\}, i \neq j \tag{2.14}$$

Na função objetivo, os custos fictícios relacionados às trocas de mandril são somados ao seu valor, de modo que tubetes com mandris iguais sejam produzidos em sequência.

Para obtermos o custo total de movimentos de bolachas necessários para a produção dos tubetes, basta eliminar os custos fictícios de trocas de mandril realizadas durante a confecção da carteira, ou ainda:

$$z = z' - \sum_{j=1}^{n}\sum_{i=1}^{n} x_{ij}m_{ij} = z' - (M-1)(2B_{max}+1),$$

uma vez que, se o número de mandris diferentes é M, teremos $(M-1)$ trocas de mandril e o custo relacionado a cada troca de mandril é $(2B_{max}+2)$.

Assim, é apresentada aqui uma solução distinta para a restrição de que tubetes possuindo o mesmo mandril sejam fabricados em sequência.

2.5.2 O Problema da Tubeteira como um E-PCVG

O problema da Tubeteira será interpretado como um E-PCVG, ao permitir-se a existência de uma posição vazia no estaleiro para a confecção de cada tubete, ainda, é necessário que a primeira posição do estaleiro esteja ocupada por uma bolacha.

Desse modo o problema é definido sobre um grafo valorado completo $G = (V, E)$, onde V é a união da coleção de *clusters*, assim:

$$V = S_1 \bigcup S_2 \bigcup ... \bigcup S_n, \quad \text{onde} \quad S_p \bigcap S_q = \emptyset, \quad \text{para} \quad p \neq q.$$

Cada *cluster* $S_i, i = 1, ..., n$ é formado pelas possíveis configurações das bolachas constituintes de um tubete ao permutar-se a posição vazia e o conjunto E é constituído pelas arestas que ligam os vértices do grafo, às quais associamos um custo c_{ij} que representa a quantidade de trocas e/ou movimentos de bolachas necessários para a fabricação do tubete j imediatamente após o tubete i. Para esta abordagem as arestas *intraclusters* não são consideradas, pois isso acarretaria a fabricação do mesmo tubete duas ou mais vezes.

Nossa meta, identicamente ao modelo PCV, é definir uma ordem de fabricação dos tubetes que apresente o menor número de trocas e/ou movimentos de bolachas, priorizando a confecção sucessiva de tubetes com o mesmo mandril.

Primeiramente é definida a variável binária x_{ij} tal que:

$$x_{ij} = \begin{cases} 1, & \text{se a aresta } (i,j) \text{ faz parte da solução} \\ 0, & \text{caso contrário.} \end{cases}$$

2.5.2.1 Modelagem considerando trocas de bolachas

Para o caso em que consideramos uma posição vazia no estaleiro, o problema da Tubeteira é modelado como um E-PCVG da seguinte maneira:

$$\min z' = \sum_{(i,j)\in E} x_{ij}(c_{ij} + m_{ij}) \tag{2.15}$$

sujeito a:

$$\sum_{\substack{i\in S_p \\ (i,j)\in E}} \sum_{\substack{j\notin S_p \\ (i,j)\in E}} x_{ij} = 1, \quad \text{para todos conjuntos } S_p, p = 1, 2, ..., n \tag{2.16}$$

$$\sum_{\substack{i\notin S_p \\ (i,j)\in E}} \sum_{\substack{j\in S_p \\ (i,j)\in E}} x_{ij} = 1, \quad \text{para todos os conjuntos } S_p, p = 1, 2, ..., n \tag{2.17}$$

$$\sum_{\substack{i\in V \\ (i,j)\in E}} x_{ij} - \sum_{\substack{k\in V \\ (j,k)\in E}} x_{jk} = 0, \quad \text{para todo } j \in V \tag{2.18}$$

$$\sum_{i\in\Omega} \sum_{i\in S_p} \sum_{j\notin\Omega} \sum_{j\in S_q} x_{ij} \geq 1, \quad \text{para todo conjunto } \Omega, \tag{2.19}$$

tal que, $2 \leq |\Omega| \leq m - 2$, subconjuntos próprios da coleção de *clusters*.

$$\sum_{(i,j)\in S_p} x_{ij} = 0, \quad p = 1, 2, ..., n \tag{2.20}$$

$$0 \leq c_{ij} \leq B_{max} + 2, \tag{2.21}$$

onde B_{max} é o número máximo de bolachas necessárias para confeccionar um tubete.

$$m_{ij} = \begin{cases} B_{max} + 3, & \text{se } M_i \neq M_j \\ 0, & \text{caso contrário.} \end{cases} \tag{2.22}$$

$$x_{ij} \in \{0, 1\}, \text{ para todo } i, j \in \{1, 2, ..., n\}, i \neq j \tag{2.23}$$

Na função objetivo 2.15, os custos fictícios relacionados às trocas de mandril são somados juntamente com os custos relacionados às trocas de bolachas, de modo que os tubetes com o mesmo mandril são produzidos em sequência. Para obtermos apenas o custo total de trocas de bolachas necessários para a produção entre dois tubetes i e j é suficiente eliminar-se os custos fictícios de trocas de mandril que foram realizadas durante a produção, ou seja:

$$z = z' - \sum_{(i,j)\in E} x_{ij} m_{ij} = z' - (M - 1)(B_{max} + 2),$$

sendo que o número de mandris diferentes é M, teremos $(M - 1)$ trocas de mandril e o custo relacionado a cada troca de mandril é $(B_{max} + 2)$.

Na modelagem descrita, atribuímos nas restrições a associação de um custo elevado referente à troca de mandril.

2.5.2.2 Modelagem considerando movimentos de bolachas

Para o caso em que consideramos uma posição vazia no estaleiro, o problema da Tubeteira é modelado como um E-PCVG da seguinte maneira:

$$\min z' = \sum_{(i,j)\in E} x_{ij}(c_{ij} + m_{ij}) \tag{2.23}$$

sujeito a:

$$\sum_{\substack{i\in S_p \\ (i,j)\in E}} \sum_{\substack{j\notin S_p \\ (i,j)\in E}} x_{ij} = 1, \text{ para todos os conjuntos } S_p, p = 1, 2, ..., n \tag{2.24}$$

$$\sum_{\substack{i\notin S_p \\ (i,j)\in E}} \sum_{\substack{j\in S_p \\ (i,j)\in E}} x_{ij} = 1, \text{ para todos os conjuntos } S_p, p = 1, 2, ..., n \tag{2.25}$$

$$\sum_{\substack{i\in V \\ (i,j)\in E}} x_{ij} - \sum_{\substack{k\in V \\ (i,j)\in E}} x_{jk} = 0, \text{ para todo } j \in V \tag{2.26}$$

$$\sum_{i\in\Omega} \sum_{i\in S_p} \sum_{j\notin\Omega} \sum_{j\in S_q} x_{ij} \geq 1, \text{ para todo conjunto } \Omega, \tag{2.27}$$

tal que, $2 \leq |\Omega| \leq m-2$, subconjuntos próprios da coleção de *clusters*.

$$\sum_{(i,j)\in S_p} x_{ij} = 0, \quad p = 1, 2, ..., n \tag{2.28}$$

$$0 \leq c_{ij} \leq 2B_{max} + 2, \tag{2.29}$$

onde B_{max} é o número máximo de bolachas necessárias para confeccionar um tubete.

$$m_{ij} = \begin{cases} 2B_{max} + 3, & \text{se } M_i \neq M_j \\ 0, & \text{caso contrário.} \end{cases} \tag{2.30}$$

$$x_{ij} \in \{0, 1\}, \text{ para todo } i, j \in \{1, 2, ..., n\}, i \neq j \tag{2.31}$$

Para o cálculo apenas do custo total de movimentos de bolachas necessários para a produção entre dois tubetes i e j basta eliminarmos os custos fictícios de trocas de mandril realizadas durante a confecção da carteira, ou seja:

$$z = z' - \sum_{(i,j)\in E} x_{ij} m_{ij} = z' - (M-1)(2B_{max} + 3),$$

sendo que o número de mandris diferentes é M, teremos $(M-1)$ trocas de mandril e o custo relacionado a cada troca de mandril é $(2B_{max} + 3)$.

Modelado o problema da Tubeteira para os casos PCV e E-PCVG, considerando trocas e movimentos de bolachas, nosso objetivo é a implementação de heurísticas que minimizem as trocas e/ou movimentos de bolachas entre a fabricação dos tubetes, uma vez que Fenato (2008) e Hoto *et al.* (2010) realizaram simulações no *solver Xpress-MP* (Dash Optimization, 2000), mas, não exploraram o emprego de heurísticas na resolução do problema.

2.6 CONSIDERAÇÕES

É relevante ressaltar que ao considerar o problema da Tubeteira como um PCV, a quantidade de vértices a ser analisada é igual à quantidade de tubetes a serem fabricados. Por outro lado, para a abordagem deste problema segundo um E-PCVG o número de vértices presentes no grafo é proporcional à quantidade de bolachas que constituem os tubetes devido o fato da posição vazia percorrer todas as posições de bolachas, exceto a primeira posição, a qual deve sempre conter uma bolacha.

Para a visualização de que há um aumento significativo de vértices quando passamos de um Problema do Caixeiro Viajante para um Problema do Caixeiro Viajante Generalizado, apresentamos a tabela 2.3:

Tabela 2.3: Quantidade de vértices para os casos PCV e E-PCVG - Problema da Tubeteira

Carteira	PCV	PCVG
1	10	79
2	16	246
3	10	205
4	8	132
5	7	122
6	6	154
7	6	104

A seguir, tal situação é exemplificada por meio de uma carteira simples, assim sejam os tubetes A, B, C e D com suas respectivas configurações dadas pela tabela 2.4:

O grafo completo do problema sem considerar uma posição vazia no estaleiro e uma rota factível são ilustradas pelas figuras 2.8 e 2.9:

Tabela 2.4: Configurações dos tubetes.

Tubete	Configuração das Bolachas
A	(200, 100, 102, 103, 2000)
B	(200, 100, 101, 103, 2000)
C	(200, 101, 102, 2000)
D	(200, 101, 102, 103, 2000)

Fonte: Fenato, 2008.

Figura 2.8: Grafo completo dos tubetes A, B, C, D

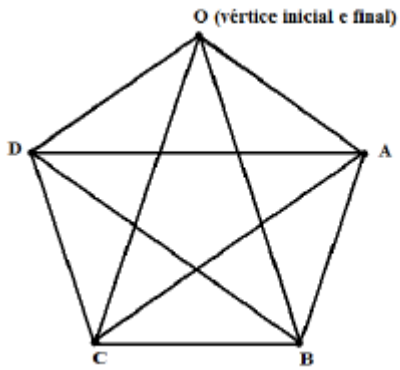
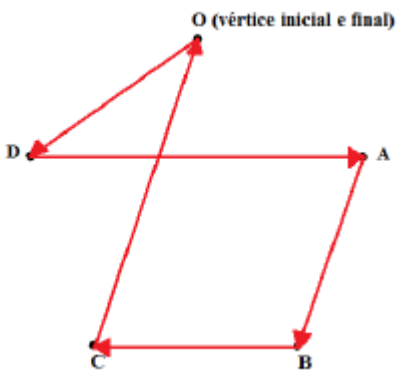


Figura 2.9: Rota factível



A tabela 2.5 apresenta as possíveis configurações de bolachas para cada um dos tubetes A, B, C, D, onde a posição vazia é representada por ∅:

A figura 2.10 representa o grafo para o exemplo da carteira formada pelos tubetes A, B, C, D, considerando uma posição vazia no estaleiro, onde cada vértice é representado pelo tubete e sua respectiva configuração e a figura 2.11 apresenta uma rota factível para o problema:

Figura 2.10: Grafo considerando posições vazias no estaleiro

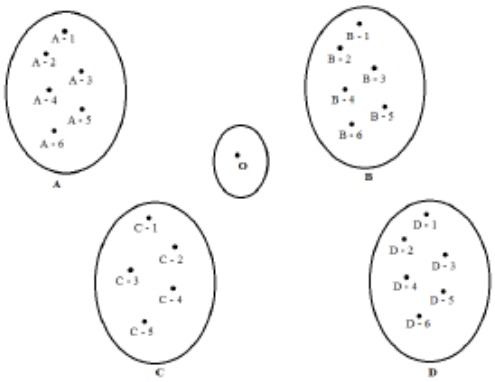
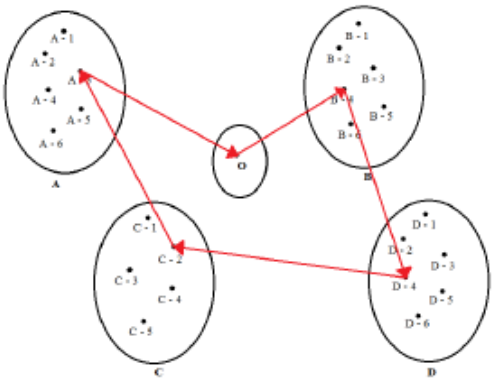


Figura 2.11: Rota factível



Descritas tais observações, propomos aqui uma abordagem heurística para o problema da Tubeteira, a fim de que soluções de mesma qualidade ou muito próximas do ótimo do problema sejam determinadas.

Tabela 2.5: Configurações dos tubetes considerando posições vazias.

Tubete - Configuração	Configuração das Bolachas
A - 1	(200, 100, 102, 103, 2000, ∅)
A - 2	(200, ∅, 100, 102, 103, 2000)
A - 3	(200, 100, ∅, 102, 103, 2000)
A - 4	(200, 100, 102, ∅, 103, 2000)
A - 5	(200, 100, 102, 103, ∅, 2000)
B - 1	(200, 100, 101, 103, 2000, ∅)
B - 2	(200, ∅, 100, 101, 103, 2000)
B - 3	(200, 100, ∅, 101, 103, 2000)
B - 4	(200, 100, 101, ∅, 103, 2000)
B - 5	(200, 100, 101, 103, ∅, 2000)
C - 1	(200, 101, 102, 2000, ∅)
C - 2	(200, ∅, 101, 102, 2000)
C - 3	(200, 101, ∅, 102, 2000)
C - 4	(200, 101, 102, ∅, 2000)
D - 1	(200, 101, 102, 103, 2000, ∅)
D - 2	(200, ∅, 101, 102, 103, 2000)
D - 3	(200, 101, ∅, 102, 103, 2000)
D - 4	(200, 101, 102, ∅, 103, 2000)
D - 5	(200, 101, 102, 103, ∅, 2000)

CAPÍTULO 3

HEURÍSTICAS APLICADAS AO PROBLEMA DA TUBETEIRA

Este capítulo apresenta a necessidade da utilização dos métodos heurísticos para a resolução de problemas combinatórios.

Consideramos heurísticas clássicas, adaptações e ainda uma heurítica desenvolvida a fim de solucionar o problema em questão, contemplamos ainda a classificação das mesmas segundo suas funções e principais características e algoritmos que descrevem o funcionamento das mesmas.

3.1 INTRODUÇÃO

A fim de resolver problemas combinatórios, uma opção simples é a análise de todas as possíveis combinações para então determinar a que melhor satisfaz a função objetivo. Por outro lado, situações reais apresentam um número de possibilidades muito elevado, tornando a técnica de enumeração inviável, uma vez que, para resolvermos um problema de roteirização de maneira exata (solução por enumeração), analisaríamos $(n-1)!$ rotas, quantidade que cresce rapidamente com um pequeno aumento de n.

Ao tratar-se de aplicações práticas, como por exemplo, empacotamento de objetos, roteamento de veículos ou mesmo planejamento de um processo de produção, além de uma solução satisfatória, existe a exigência ou mesmo necessidade de que tais problemas sejam resolvidos em um curto espaço de tempo.

A tabela abaixo, descreve o tempo que um computador capaz de realizar um bilhão de adições por segundo levaria para calcular um problema de roteirização de maneira exata:

Para a resolução do Problema do Caixeiro Viajante e do Problema do Caixeiro Viajante Generalizado simétricos são analisadas $\frac{(n-1)!}{2}$ rotas possíveis, assim, considerando um problema com 20 vértices, temos 6.10^{16} rotas possíveis. Neste caso, uma modelagem exata não é capaz de determinar uma solução para o problema, ou ainda que seja feita, despenderá um tempo muito grande e em alguns casos o programa desenvolvido pode não

N	Rotas/s	$(n-1)!$	Cálculo Total
5	250 milhões	24	insignificante
10	110 milhões	362880	$0,003$ s
15	71 milhões	87 bilhões	20 min.
20	53 milhões	$1,2.10^{17}$	73 anos
25	42 milhões	$6,2.10^{23}$	470 milhões de anos

Fonte: http://www.mat.ufrgs.br.

retornar resposta alguma, como em Fenato (2008).

As técnicas heurísticas foram desenvolvidas a fim de resolver problemas combinatórios com um alto grau de complexidade, auxiliando de forma inteligente o processo de enumeração, selecionando soluções promissoras e analisando assim um espaço de busca reduzido, permitindo o estudo de problemas com uma dimensão elevada.

Com a utilização de heurísticas, obtemos uma solução viável em tempo computacional razoável, no entanto não é possível determinar se esta solução é ótima, ou mesmo quão próxima ela está da solução ótima.

A fim de apresentar resultados satisfatórios, foram desenvolvidas heurísticas que auxiliam tanto no processo de construção da solução, as chamadas heurísticas Construtivas, quanto na melhoria da mesma, as heurísticas de Refinamento.

Uma heurística construtiva, ou míope, para o PCV, consiste em tentar encontrar uma boa rota, considerando a cada iteração somente o próximo passo, ou seja, o critério de escolha é basicamente local (Campello, Maculan, 1994).

As heurísticas de Construção possuem a característica de pararem no momento em que uma solução factível é encontrada sem a tentativa de melhora.

As heurísticas de Refinamento partem de uma solução inicial factível, obtida por meio de uma heurística construtiva, em busca de uma melhoria na função objetivo. Os passos realizados para o melhoramento da solução, geralmente são baseados na exclusão e inclusão de arestas ou vértices a esta solução (inicial).

Utilizamo-nos de heurísticas presentes na literatura e ainda apresentamos adaptações das mesmas a fim de abranger todas as particularidades do problema da Tubeteira.

3.2 Heurística e o Problema da Tubeteira

Para a contextualização do problema da Tubeteira segundo um PCV, consideramos a heurística de construção Vizinho mais Próximo e as heurísticas de refinamento Melhor Vizinho mais Próximo, Biparticionamento e Triparticionamento, que são adaptações das heurísticas clássicas 2-Opt e 3-Opt.

Considerando o problema da Tubeteira como um E-PCVG além das heurísticas utilizadas para o caso PCV, foi implementada a heurística de refinamento Melhor Configuração, que após a determinação de uma solução viável, permuta as configurações dos tubetes sem alterar a ordem em que os mesmos se encontram.

A fim de melhorar as soluções determinadas inicialmente combinamos as heurísticas citadas acima, implementando assim as heurísticas Melhor Vizinho mais Próximo Biparticionado, Melhor Vizinho mais Próximo Triparticionado, Melhor Vizinho mais Próximo com a Melhor Configuração, Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração e ainda Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração.

Uma vez que a abordagem por meio de um PCV possui apenas um tipo de configuração para cada tubete a heurística Melhor Configuração se limita a resolver o caso generalizado.

As heurísticas implementadas para a resolução do problema da Tubeteira e seus respectivos algoritmos são descritos ao decorrer das próximas seções. Ainda, as implementações foram realizadas utilizando o IDE (ambiente de desenvolvimento integrado) $WxDev - C^{++}$ e linguagem C^{++}.

3.2.1 Vizinho Mais Próximo

A heurística Vizinho mais Próximo (VMP) é uma heurística construtiva gulosa, apresentada originalmente em (Bellmore & Nemhauser, 1968), e tem como base uma ideia simples que consiste em escolher aleatoriamente um vértice inicial i e então selecionar o vértice mais próximo j que ainda não foi visitado, formando assim a aresta (i, j) que fará parte da solução. O processo se encerra quando todos os vértices forem visitados.

Note que, a menos que o grafo seja completo, este procedimento pode ter dificuldades para encontrar uma rota, mesmo que exista, pois podem restar vértices que não são ligados por aresta alguma.

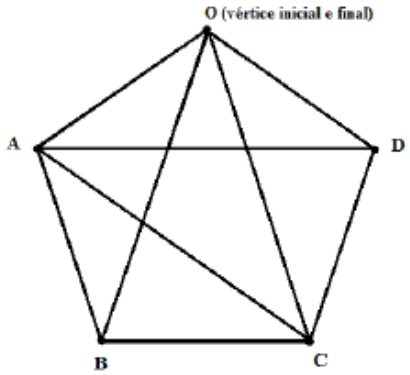
Exemplo 1. *Considere um carteiro que deve entregar correspondências em 4 sítios A, B, C e D, a partir de um depósito O, por meio de um ciclo Hamiltoniano de custo mínimo. Os custos entre as cidades são dados pela tabela* 3.1 *e o grafo não-completo G deste problema é representado pela figura* 3.1.

Tabela 3.1: Custos entre os sítios

Sítios	A	B	C	D
A	0	12	10	14
B	12	0	13	-
C	10	13	0	15
D	14	-	15	0

Suponha que a heurística escolha aleatoriamente o sítio A como o primeiro vértice a ser visitado a partir do depósito. Assim, o próximo sítio a ser visitado será C, uma vez que é o mais próximo de A, formando assim a aresta (A, C) com custo 10. Após visitar o sítio C, o carteiro irá para o sítio B, acrescentando a aresta (C, B) de custo 13 à rota. Por fim, resta o sítio D a ser visitado, porém não existe aresta entre os sítios B e D, de modo que é impossível formar uma rota mesmo que existam R_1 = (O, B, A, C, D, O) e R_2 = (O, C, A, B, D, O), ambas com custo total de 37 unidades e soluções factíveis para o problema.

Figura 3.1: Grafo não-completo G



Mesmo em grafos completos esta heurística pode fornercer uma solução ruim, por ser forçada a escolher arestas muito distantes nos últimos passos. Outro problema encontrado nesta heurística, é o fato do primeiro vértice ser escolhido arbitrariamente, o que pode acarretar a obtenção de uma solução muito distante da ótima.

Exemplo 2. *Suponha que um viajante deva visitar uma única vez as cidades 1, 2, 3 e 4 a partir de um depósito O, por meio de um ciclo Hamiltoniano de custo mínimo. Considere um grafo completo G para este problema e C a matriz simétrica de custos entre as cidades.*

$$C = (c_{ij}) = \begin{bmatrix} 0 & 8 & 10 & 14 \\ 8 & 0 & 9 & 16 \\ 10 & 9 & 0 & 7 \\ 14 & 16 & 7 & 0 \end{bmatrix}$$

Se a heurística escolher o vértice 2 para ser visitado a partir do depósito, a próxima cidade escolhida será 1, formando assim a aresta $(2, 1)$ de custo 8, logo após a cidade 1 será visitada a cidade 3, acrescentando à solução a aresta $(1, 3)$ com custo 10 e assim resta a cidade 4 a ser visitada, o que adiciona a aresta $(3, 4)$ com custo 7, obtendo assim a rota R_1 = (O, 2, 1, 3, 4, O) que totaliza um custo de 25 unidades.
Se a heurística selecionar a cidade 3 a ser visitada a partir do depósito O, a cidade 4 será escolhida para ser visitada após 3, formando a aresta $(3, 4)$ de custo 7. Logo após 4, será visitada a cidade 1, acrescentando à rota a aresta $(4, 1)$ com custo 14 e assim o caixeiro visitará a cidade 2 que incluirá a aresta $(1, 2)$ de custo 8 à solução o que nos fornece a rota R_2 = (O, 3, 4, 1, 2, O) que fornece um custo total de 29 unidades. Note que o aumento significativo de 5 unidades deu-se por causa do custo elevado entre as cidades 4 e 1.
Temos ainda como solução ótima as rotas R_3 = (O, 4, 3, 2, 1, O) e R_4 = (O, 1, 2, 3, 4, O) que são equivalentes devido a simetria do problema e possuem custo total de 24 unidades.

Um algoritmo para a heurística Vizinho mais Próximo pode ser descrito como segue:

---

algoritmo VMP (sol, cus, m, n, p, c)
escolha o tubete (p, c) como vértice inicial da solução
soma $\leftarrow 0$

para $i \leftarrow 1$ até $m - 1$ **faça**
acrescente à solução o vértice de um cluster não visitado mais próximo do vértice do cluster anterior
acrescente à soma o custo da aresta percorrida
retorne soma
fim-algoritmo

Devido a simplicidade da heurística, podemos escolher um vértice que fornecerá uma solução muito ruim. Por exemplo, considere a carteira 1 (Apêndice B) formada por 10 tubetes com 9 tipos diferentes de bolachas.

Utilizando a implementação realizada em linguagem C^{++} obtemos duas soluções com diferentes valores para a função objetivo:

Iniciando a heurística a partir do tubete fictício O, com a fabricação do tubete 1, temos a rota (O, 1 - 1, 4 - 1, 5 - 1, 2 - 1, 10 - 1, 8 - 1, 9 - 7, 6 - 7, 3 - 11, 7 - 11, O) que retorna custo 49, formada pelo tubete e sua respectiva configuração.

Por outro lado, quando o processo se inicia pelo tubete 8, temos a rota (O, 8 - 1, 9 - 7, 2 - 7, 10 - 8, 4 - 9, 5 - 6, 6 - 8, 1 - 11, 7 - 11, 3 - 11, O) que possui custo 44, formada pelo tubete e sua respectiva configuração.

3.2.2 Melhor Vizinho mais Próximo

A heurística de refinamento Melhor Vizinho mais Próximo (MVMP) consiste em analisar todas as rotas possíveis construídas pela heurística do Vizinho mais Próximo, isto é, considerando cada tubete como inicial a partir do tubete fictício O, e dentre elas escolher a solução de menor custo, ou ainda, a solução que retorne o menor valor da função objetivo.

No exemplo 2, considere as rotas obtidas pela heurística de construção Vizinho mais Próximo:
R_1 = (O, 2, 1, 3, 4, O) com custo de 25 unidades;
R_2 = (O, 3, 4, 1, 2, O) com custo de 29 unidades;
R_3 = (O, 4, 3, 2, 1, O) com custo de 24 unidades e
R_4 = (O, 1, 2, 3, 4, O) com custo de 24 unidades.

A heurística Melhor Vizinho mais Próximo analisa as rotas factíves R_1, R_2, R_3 e R_4 e retorna como solução final a rota que possui o menor custo, ou ainda, o menor valor da função objetivo, no caso as rotas R_3 ou R_4 que são equivalentes.

A seguir um algoritmo para a heurística Melhor Vizinho mais Próximo:

```
algoritmo MVMP (sol, cus, m, n)
    soma ← ∞
    para i ← 1 até m faça
        para j ← 1 até n faça
        soma* ← VMP(sol*, cus, m, n, i, j)
        se soma* < soma então faça
            soma ← soma*
            sol ← sol*
    retorne soma
fim-algoritmo
```

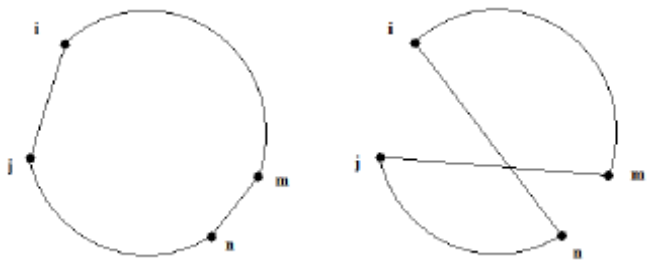
Se a solução construída pela heurística Vizinho mais Próximo iniciar com o tubete que fornece a rota com o menor custo, a heurística de refinamento não apresentará mudanças.

3.2.3 2-Opt e 3-Opt

Croes (1958 *apud* BELLMORE, (1958), p. 551) propôs a heurística de refinamento 2-Opt enquanto Lin (1965 *apud* BELLMORE, (1958), p. 551) apresentou a heurística 3-Opt.

O algoritmo 2 - Opt basicamente remove duas arestas da rota e reconecta os dois caminhos criados, (quando removemos duas arestas temos dois caminhos desconectados) obtendo uma nova rota, esta modificação na rota é denominada de movimento 2 - Opt. Existe apenas uma maneira de reconectar estes dois caminhos a fim de ainda termos uma rota viável (Figura 3.2).

Figura 3.2: Solução factível 2 - Opt



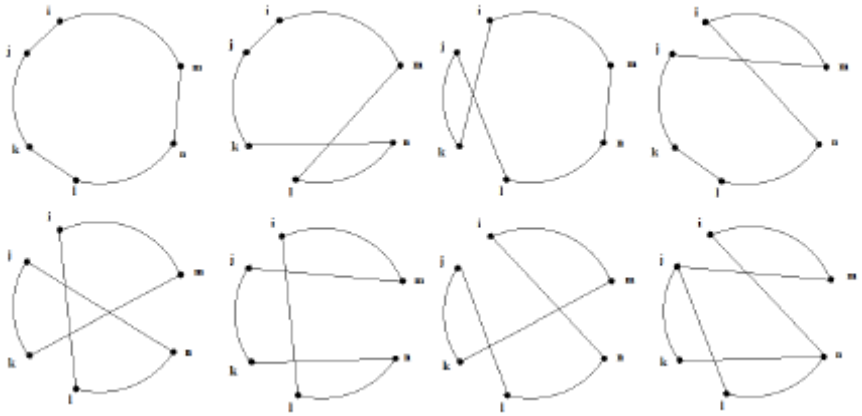
Considera-se o caminho reconectado como solução apenas se a rota obtida for mais curta (com o menor custo) em relação à rota anterior.

O processo continua removendo as arestas e reconectando a rota por meio de arestas não pertencentes à solução até não existirem mais melhoras do tipo 2 - Opt a serem feitas. Diz-se que a rota final obtida é agora 2 - ótima.

O algoritmo 3 - Opt trabalha similarmente, porém ao invés de remover duas arestas, são removidas três. No algoritmo 3-Opt ao serem removidas três arestas existem 7 soluções factíveis alternativas (Figura 3.3).

Termina-se a busca quando não existem mais movimentos 3 - Opt que possibilitem a melhora da rota atual. Um movimento 3 - Opt pode ser visto como dois ou três movimentos 2 - Opt, desse modo no caso de uma rota ser 3 - ótima é 2 - ótima também.

Figura 3.3: Soluções factíveis 3 - Opt



3.2.4 Biparticionamento

A heurística Biparticionamento (2-Part) é uma adaptação da heurística clássica 2-Opt e consiste em eliminar uma aresta de uma solução (rota) viável e religar dois vértices extremos da solução por meio de uma nova aresta, sem que sejam formadas subrotas.

O procedimento é realizado em todas as posições que permitam o particionamento da solução. A melhor solução (rota) encontrada durante a realização do procedimento é atualizada ao final do algoritmo.

Abaixo um algoritmo para a heurística Biparticionamento:

```
algoritmo Biparticionamento (sol, soma, n)
    soma** ← soma
    para i ← 1 até n − 1 faça
        para j ← i + 1 até n − 1 faça
        particione a solução removendo a aresta entre os vértices i e i + 1
        se soma* < soma então faça
            soma ← soma*
            sol** ← sol*
    soma ← soma**
    sol ← sol**
    retorne soma
fim-algoritmo
```

3.4.2.1 Melhor vizinho mais próximo biparticionado

A heurística Melhor Vizinho mais Próximo Biparticionado (MVMP-2-Part) é um refinamento para a solução obtida por meio do Melhor Vizinho mais Próximo, a partir da qual a heurística Biparticionamento é aplicada na melhor solução encontrada até que não haja mais melhora. No caso de melhora a solução é então atualizada.

Abaixo um algoritmo que descreve o funcionamento da heurística Melhor Vizinho mais Próximo Biparticionado:

```
algoritmo MVMP-2-Part (sol, cus, m, n)
    soma ← ∞
    para i ← 1 até m faça
        para j ← 1 até n faça
        soma* ← VMP(sol*, cus, m, n, i, j)
        enquanto soma** ← 2-part(sol*, soma*, m) < soma* faça
            soma* ← soma**
        se soma* < soma então faça
            soma ← soma*
            sol ← sol*
    retorne soma
fim-algoritmo
```

3.2.5 Triparticionamento

O Triparticionamento (3-Part) é baseado na heurística clássica 3-Opt. A heurística Triparticionamento remove duas arestas de uma solução viável e religa os vértices extremos das partições formadas através de duas novas arestas, sem permitir a formação de subrotas.

Descrevemos a seguir um algoritmo para a heurística Triparticionamento:

---

algoritmo Triparticionamento (sol, soma, n)
  soma** ← soma
  para $i \leftarrow 1$ até $n-2$ **faça**
    para $j \leftarrow i+1$ até $n-1$ **faça**
    particione a solução removendo a aresta entre os vértices i e $i+1$ e a aresta entre os vértices j e $j+1$
    sol* ← melhor solução dentre as possíveis reconexões das partições criadas
    se soma* < soma **então faça**
      soma ← soma*
      sol** ← sol*
  soma ← soma**
  sol ← sol**
  retorne soma
fim-algoritmo

3.2.5.1 Melhor vizinho mais próximo triparticionado

A heurística Melhor Vizinho mais Próximo Triparticionado (MVMP-3-Part) tem como objetivo refinar a solução encontrada através do Melhor Vizinho mais Próximo, aplicando a heurística Triparticionamento na sequência determinada inicialmente. Caso ocorra uma melhoria a solução é atualizada.

O algoritmo a seguir descreve o funcionamento da heurística Melhor Vizinho mais Próximo Triparticionado:

```
algoritmo MVMP-3-Part (sol, cus, m, n)
    soma ← ∞
    para i ← 1 até m faça
        para j ← 1 até n faça
        soma* ← VMP(sol*, cus, m, n, i, j)
        enquanto soma** ← 3-part(sol*, soma*, m) < soma* faça
            soma* ← soma**
        se soma* < soma então faça
            soma ← soma*
            sol ← sol*
    retorne soma
fim-algoritmo
```

3.2.6 Melhor Configuração

A heurística de refinamento Melhor Vizinho mais Próximo não apresentou melhoras significativas na obtenção de sequências com custo mínimo. Assim, percebemos a necessidade de uma heurística que pudesse alcançar as características intrínsecas do problema em questão e então desenvolvemos a heurística denominada como Melhor Configuração (MC).

Tal heurística se limita a analisar se é necessário realizar trocas nas configurações dos tubetes sem que a sequência obtida seja alterada.

Por exemplo, se a fabricação do tubete 1 se dá utilizando a configuração C - 4 e o próximo tubete a ser fabricado é o 5 com configuração C - 3, a heurística irá manter o primeiro tubete e sua configuração, e apenas permutará as outras configurações possíveis para o tubete 5. Assim, no caso de alguma configuração apresentar melhoria, digamos C - 4, a solução é atualizada.

O algoritmo abaixo descreve o funcionamento da heurística Melhor Configuração:

```
algoritmo Melhor Configuração (sol, soma, m, n)
    soma ← ∞
    para i ← 1 até m faça
        para j ← 1 até n faça
        crie sol* substituindo o nó do cluster i da solução por outro nó
        do mesmo cluster mas com configuração j
        se soma* < soma então faça
            soma ← soma*
            sol ← sol*
    retorne soma
fim-algoritmo
```

3.2.6.1 Melhor vizinho mais próximo com a melhor configuração

Após a obtenção de uma rota por meio da heurística Melhor Vizinho mais Próximo aplicamos a heurística Melhor Configuração a fim de que o processo retorne uma solução mais próxima do ótimo do problema.

Um algoritmo para a heurística Melhor Vizinho mais Próximo com a Melhor Configuração (MVMP-MC) é apresentado abaixo:

---

algoritmo MVMP-MC (sol, cus, m, n)
    soma $\leftarrow \infty$
    para $i \leftarrow 1$ até m **faça**
        para $j \leftarrow 1$ até n **faça**
            soma* $\leftarrow$ VMP(sol*, cus, m, n, i, j)
            soma* $\leftarrow$ MC(sol*, soma*, m, n)
        se soma* $<$ soma **então faça**
            soma $\leftarrow$ soma*
            sol $\leftarrow$ sol*
    retorne soma
fim-algoritmo

3.2.6.2 Melhor vizinho mais próximo biparticionado com a melhor configuração

Para esta combinação de heurísticas, consideramos a solução obtida pela heurística Melhor Vizinho mais Próximo Biparticionado e aplicamos sobre ela a heurística Melhor Configuração.

Descrevemos a seguir um algoritmo para a heurística Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração (MVMP-2-Part-MC):

---

algoritmo MVMP-2-Part-MC (sol, cus, m, n)
    soma $\leftarrow \infty$
    para $i \leftarrow 1$ até m **faça**
        para $j \leftarrow 1$ até n **faça**
        soma* $\leftarrow$ VMP(sol*, cus, m, n, i, j)
        enquanto soma** $\leftarrow$ 2-part(sol*, soma*, m) $<$ soma* **faça**
            soma* $\leftarrow$ soma**
        soma* $\leftarrow$ MC(sol*, soma*, m, n)
        se soma* $<$ soma **então faça**
            soma $\leftarrow$ soma*
            sol $\leftarrow$ sol*
    retorne soma
fim-algoritmo

3.2.6.3 Melhor vizinho mais próximo triparticionado com a melhor configuração

Analogamente à heurística anterior, a partir da solução fornecida pela heurística Melhor Vizinho mais Próximo Triparticionado aplica-se a heurística Melhor Configuração.

Segue um algoritmo da heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração (MVMP-3-Part-MC):

```
algoritmo MVMP-3-Part-MC (sol, cus, m, n)
    soma ← ∞
    para i ← 1 até m faça
        para j ← 1 até n faça
        soma* ← VMP(sol*, cus, m, n, i, j)
        enquanto soma** ← 3-part(sol*, soma*, m) < soma* faça
            soma* ← soma**
        soma* ← MC(sol*, soma*, m, n)
        se soma* < soma então faça
            soma ← soma*
            sol ← sol*
    retorne soma
fim-algoritmo
```

CAPÍTULO 4
UM PROTÓTIPO COMPUTACIONAL PARA AUXILIAR NA PREPARAÇÃO DE UMA MÁQUINA TUBETEIRA

Neste capítulo apresentamos o programa desenvolvido para a resolução do Problema da Tubeteira, por meio de um ambiente de desenvolvimento integrado (IDE) gratuito, *WxDev-C++*.

4.1 INTRODUÇÃO

Para o desenvolvimento desta dissertação foi proposto a implementação de heurísticas que viabilizassem a obtenção de sequências satisfatórias para a confecção de carteiras de tubetes. Assim, além de implementarmos heurísticas clássicas e ainda uma heurística abrangendo as características da Máquina Tubeteira, optamos pelo desenvolvimento de uma interface, denominada como interface gráfica de usuário (GUI).

A implementação foi realizada em linguagem *C++*, utilizando o IDE (ambiente de desenvolvimento integrado) gratuito *WxDev-C++*, o qual é baseado no também IDE *Dev-C++*.

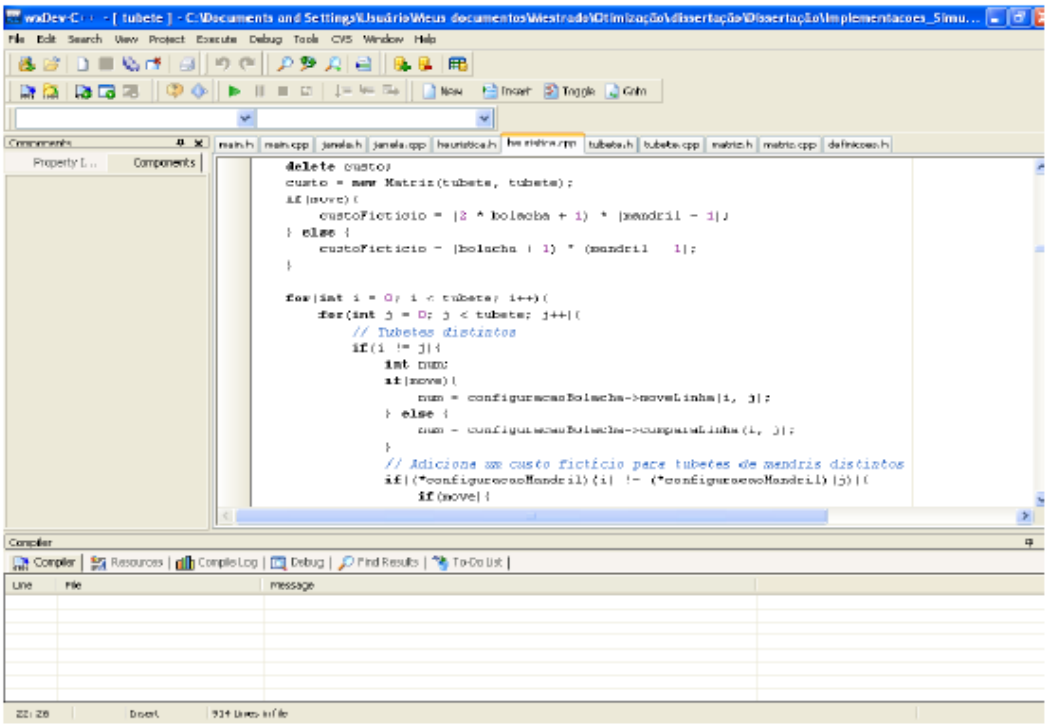
A vantagem da utilização do *WxDev - C++* está na possibilidade do desenvolvimento e criação de aplicações *wxWidgets*, as quais auxiliam na criação de uma interface gráfica de usuário (GUI) que representa as ações disponíveis por meio de ícones (botões) gráficos.

4.2 EXECUÇÃO DO PROGRAMA – AMBIENTES E BOTÕES GRÁFICOS

O *software* denominado aqui como Máquina Tubeteira tem como objetivo minimizar a quantidade de trocas e/ou movimentos de bolachas, ou ainda, retornar a sequência que realiza o menor número de trocas e/ou movimentos entre as bolachas de uma carteira.

A execução do programa é realizada de maneira simples, descrita pelas subseções seguintes.

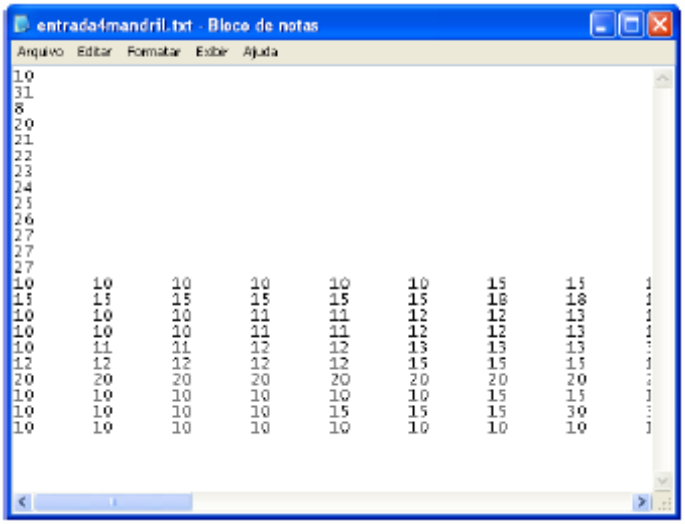
Figura 4.1: Janela do Programa



4.2.1 Dados de Entrada – Botão Abrir

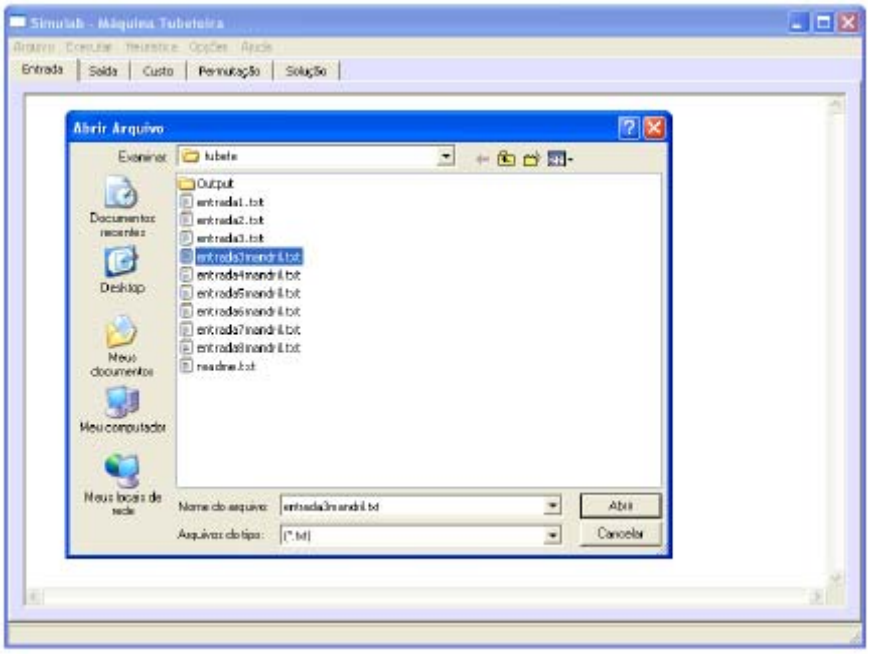
O programa recebe como dado de entrada um arquivo ".txt"(Figura 4.2) contendo a quantidade de tubetes que a carteira possui, o número máximo de bolachas que serão alocadas no estaleiro, a quantidade e descrição dos diferentes tipos de mandril necessários para a fabricação da carteira de tubetes e ainda a configuração das bolachas que compõem cada tubete.

Figura 4.2: Dados de entrada



Para a leitura de um arquivo de dados, basta clicar na opção "Abrir" (Figura 4.3) que se encontra no Menu do programa.

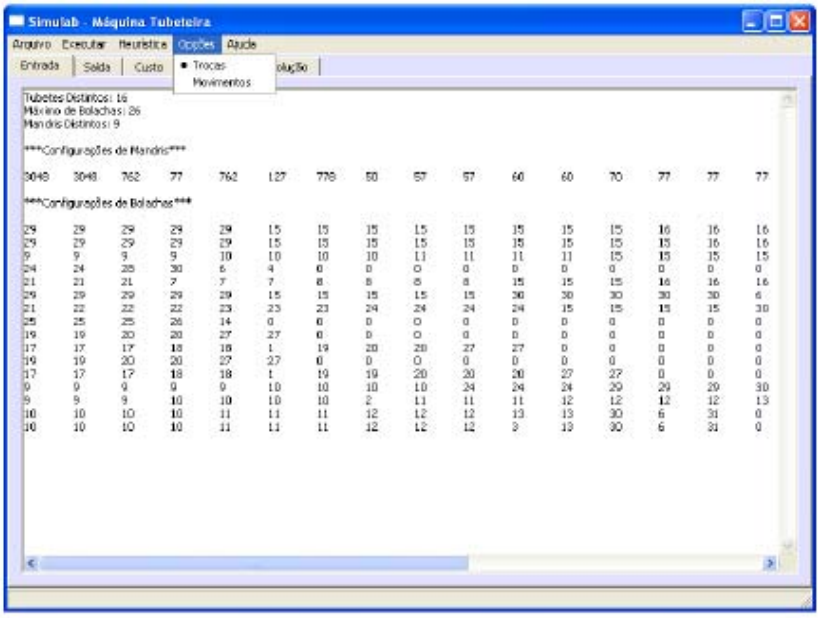
Figura 4.3: Botão Abrir



4.2.2 Cálculo da Matriz de Custos

Após a escolha da carteira a ser fabricada, é necessário optar pelo cálculo dos custos em relação à trocas ou movimentos de bolachas (Figura 4.4), e então todos os resultados obtidos serão baseados nesta opção.

Figura 4.4: Botão Custos

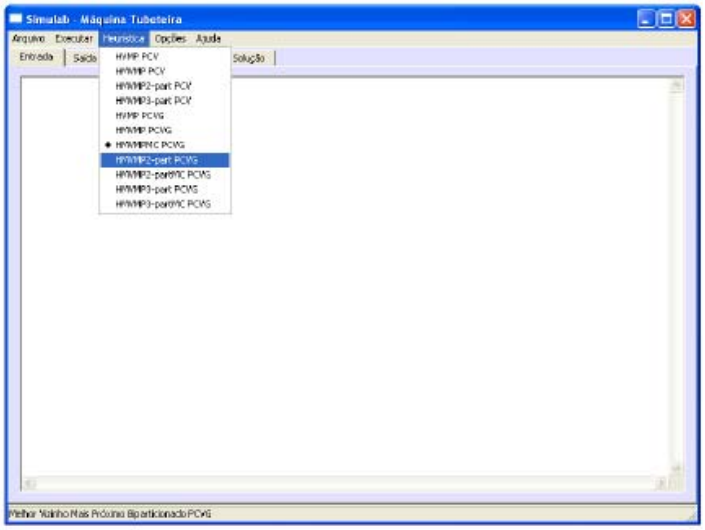


4.2.3 Execução das Heurísticas

Para a obtenção dos resultados computacionais estão disponíveis as heurísticas a serem executadas. Ao total foram implementadas 11 (onze) heurísticas, das quais 4 (quatro) são utilizadas para o caso da Máquina Tubeteira como um PCV e 7 (sete) abordam o problema como um PCVG.

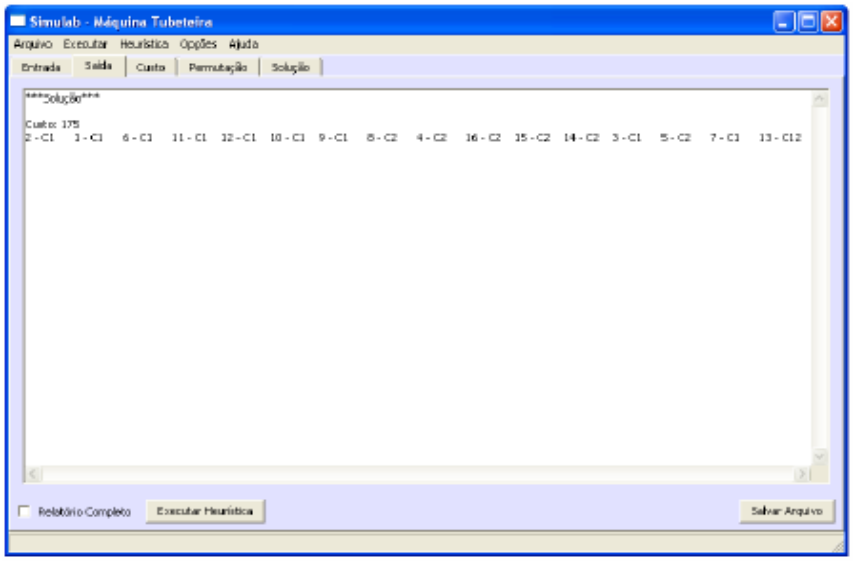
Para cada heurística a ser executada pelo programa, tem-se a indicação do caso em que deve ser utilizada, ou seja, considerando ou não uma posição vazia no estaleiro (Figura 4.5).

Figura 4.5: Opções de Heurísticas



Assim que foi selecionada a heurística, basta clicar no botão executar heurística e automaticamente o programa redireciona o usuário para a aba denominada "Saída" (Figura 4.6), fornecendo assim a sequência obtida e o custo necessário para sua fabricação.

Figura 4.6: Ambiente Saída



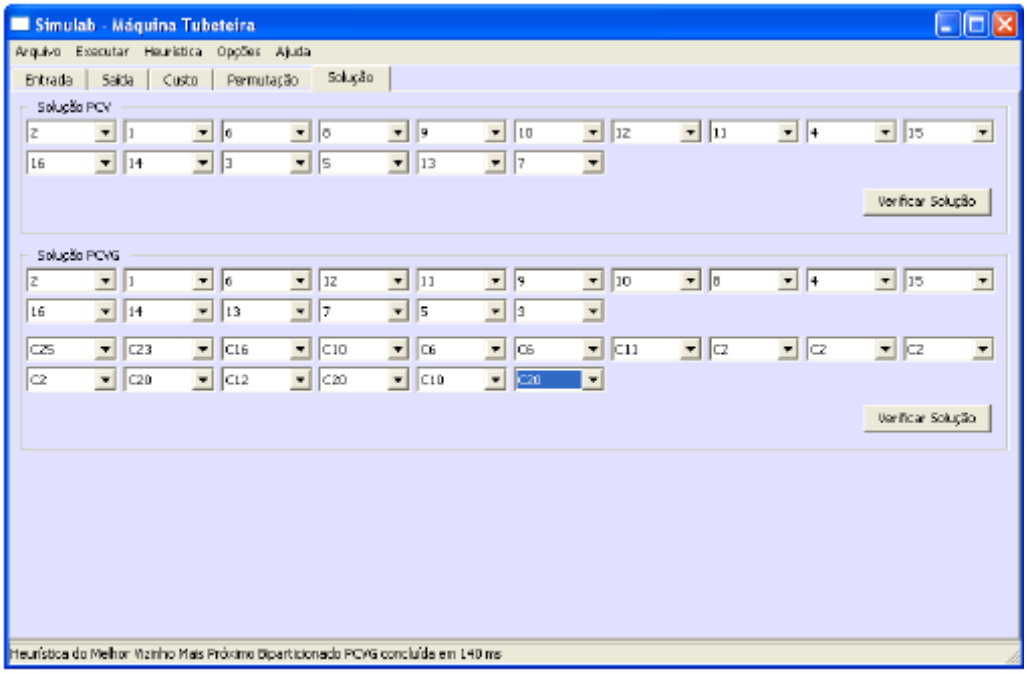
Nesta fase, o objetivo do programa foi alcançado, uma vez que as heurísticas executadas forneceram a sequência com o menor custo para a fabricação de uma carteira de tubetes.

4.2.4 Ambiente Solução

Ainda, para fins de comparação, o programa Máquina Tubeteira possui no ambiente "Solução" (Figura 4.7) a verificação de custos de sequências, no qual digita-se uma solução viável qualquer. Ao clicar o botão "Verificar solução" obtemos a quantidade de trocas e/ou movimentos que a mesma realiza.

Esta ferramenta nos auxiliou para a comparação entre os resultados obtidos através das heurísticas implementadas e as sequências obtidas por Fenato (2008) e ainda comparar ambos os resultados, com a solução fornecida (e utilizada) pela empresa.

Figura 4.7: Ambiente Solução



4.2.5 Considerações

O programa desenvolvido executou de maneira satisfatória as carteiras que a empresa nos forneceu como exemplares. Em relação ao tempo de execução, as simulações foram realizadas imediatamente sem necessidade de qualquer divisão de carteira.

Outro diferencial deste programa está na possibilidade de que sejam testadas carteiras diversas, sem a necessidade de novas implementações ou mesmo alterações no código, uma vez que para as implementações que Fenato (2008) realizou, a cada nova carteira é necessária a implementação de um novo programa contendo as particularidades de cada carteira.

Para o desenvolvimento do programa, como citado acima, utilizamos um IDE gratuito, o que permite a execução do mesmo em qualquer ambiente, não apenas em laboratórios ou ainda somente com a compra e licença de *softwares*, que é o caso do programa apresentado em Fenato (2008).

CAPÍTULO 5
RESULTADOS COMPUTACIONAIS

Neste capítulo apresentamos os resultados obtidos pelas implementações das heurísticas abordadas no capítulo 3. Os dados utilizados para os testes numéricos foram fornecidos pela empresa Sonoco, os quais correspondem às configurações descritas pelas carteiras 1, 2, 3, 4, 5, 6 e 7, que encontram-se no Apêndice B.

Em um primeiro momento consideramos o custo das sequências por meio de trocas de bolachas e em seguida analisamos o custo das rotas por meio de movimentos de bolachas.

Em Fenato (2008) e Hoto *et al.* (2010) foram realizados testes apenas para as duas primeiras carteiras, sendo assim, as implementações que os autores realizaram foram atualizadas e reformuladas para que fosse possível a comparação entre a modelagem exata e a abordagem heurística apresentada nesta dissertação, bem como uma comparação com a solução utilizada pela empresa durante a fabricação das carteiras. Fenato (2008) e Hoto *et al.* (2010) utilizaram para as simulações das carteiras 1 e 2, um computador com processador *Intel Core 2 Duo* de 3 GHz e 4 GB de *Ram*.

Realizamos os testes das sete carteiras, em plataforma *Windows* numa máquina com processador *Intel Core i7* de 2.67 *GHz* e 12 *GB* de memória *RAM*. A mesma máquina foi utilizada para simular e obter as respostas obtidas pela implementação de Fenato (2008) realizada no *solver Xpress-MP*.

5.1 Custos em Relação à Trocas de Bolachas

Apresentamos os resultados obtidos considerando como custo a quantidade de trocas de bolachas necessárias para a fabricação dos tubetes que compõem determinada carteira, proposta feita por Fenato (2008) e Hoto *et al.* (2010).

As sequências obtidas pela heurística Vizinho mais Próximo foram omitidas, uma vez que a mesma fornece diferentes resultados que dependem fortemente da escolha do primeiro vértice.

5.1.1 Carteira 1

Com os dados fornecidos pela carteira 1, temos o total de 10 tubetes, todos com o mesmo mandril e suas configurações utilizam no máximo 10 bolachas no estaleiro.

5.1.1.1 Resultados sem considerar posições vazias

A implementação da heurística Melhor Vizinho mais Próximo retornou a sequência (O, 8, 9, 2, 10, 5, 4, 1, 7, 3, 6, O), que totaliza 45 trocas de bolachas.

Por sua vez, a heurística Melhor Vizinho mais Próximo Biparticionado encontrou a sequência (O, 3, 7, 1, 4, 5, 10, 2, 9, 8, 6, O), totalizando 45 trocas.

A heurística Melhor Vizinho mais Próximo Triparticionado forneceu a sequência, (O, 8, 9, 2, 10, 4, 5, 6, 1, 7, 3, O), propondo 44 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) encontraram a sequência (O, 8, 9, 2, 10, 4, 5, 6, 1, 7, 3, O), com um total de 44 trocas.

Ainda, a sequência utilizada pela empresa é (9, 8, 2, 5, 6, 3, 10, 1, 7, 4), a qual realiza 56 trocas.

Na tabela 5.1 apresentamos um resumo do resultado.

Tabela 5.1: Resultados obtidos para a carteira 1 sem considerar posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	56	-
Xpress-MP	44	21, 4 %
MVMP	45	19, 6 %
MVMP/2-Part	45	19, 6 %
MVMP/3-Part	44	21, 4 %

5.1.1.2 Resultados considerando posições vazias

Para este caso as sequências de tubetes a serem fabricados são formadas pelo tubete e sua configuração respectivamente.

A heurística Melhor Vizinho mais Próximo retornou a sequência (O, 8 - 2, 9 - 1, 2 - 1, 10 - 1, 5 - 1, 4 - 1, 1 - 1, 6 - 7, 3 - 1, 7 - 1, O) que totaliza 42 trocas de bolachas.

A heurística Melhor Vizinho mais Próximo Biparticionado determinou a sequência (O, 9 - 1, 8 - 3, 10 - 3, 2 - 3, 4 - 1, 5 - 6, 6 - 1, 1 - 1, 7 - 1, 3 - 1, O), que propõe 40 trocas de bolachas.

Utilizando a heurística Melhor Vizinho mais Próximo Triparticionado obtivemos a sequência (O, 9 - 1, 8 - 4, 10 - 4, 2 - 4, 5 - 4, 6 - 1, 4 - 4, 1 - 1, 7 - 1, 3 - 1, O), também com um total de 39 trocas de bolachas.

Com a utilização da heurística Melhor Vizinho mais Próximo com a Melhor Configuração obtivemos a sequência (O, 4 - 1, 2 - 3, 10 - 3, 5 - 3, 6 - 9, 1 - 1, 7 - 1, 3 - 1, 8 - 2, 9 - 1, O), com um total de 42 trocas.

A heurística Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração forneceu a sequência (O, 9 - 1, 8 - 2, 10 - 7, 2 - 3, 4 - 1, 5 - 6, 6 - 9, 1 - 1, 7 - 1, 3 - 1, O), que realiza 39 trocas.

Por fim, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração retornou a sequência (O, 9 - 1, 8 - 2, 10 - 2, 2 - 2, 5 - 2, 6 - 1, 4 - 4, 1 - 1, 7 - 1, 3 - 1, O) realizando 38 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) encontraram a sequência (O, 3 - 9, 7 - 9, 1 - 9, 4 - 3, 6 - 9, 5 - 2, 10 - 2, 2 - 2, 8 - 2, 9 - 5, O), com um total de 38 trocas.

Ainda, a sequência utilizada pela empresa é (9, 8, 2, 5, 6, 3, 10, 1, 7, 4), a qual realiza 56 trocas.

Na tabela 5.2 apresentamos um resumo do resultado.

Tabela 5.2: Resultados obtidos para a carteira 1 considerando posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	56	-
Xpress-MP	38	32, 1 %
MVMP	42	25 %
MVMP/2-Part	40	28, 6 %
MVMP/3-Part	39	30, 4 %
MVMP/MC	42	25 %
MVMP/2-Part/MC	39	30, 4 %
MVMP/3-Part/MC	38	32, 1 %

5.1.2 Carteira 2

A carteira 2 é constituída por um total de 16 tubetes, os quais utilizam no máximo 26 bolachas posicionadas no estaleiro para sua fabricação e 9 tipos diferentes de mandris.

5.1.2.1 Resultados sem considerar posições vazias

A sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 7, 13, 14, 15, 16, 4, 8, 9, 10, 12, 11, 6, 1, 2, 3, 5, O), propondo a realização de 181 trocas.

As heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado retornaram a mesma sequência, (O, 2, 1, 6, 11, 12, 10, 9, 8, 4, 16, 15, 14, 13, 7, 5, 3, O), que totaliza 180 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 2, 1, 6, 8, 9, 10, 12, 11, 4, 15, 16, 14, 3, 5, 12, 7, O), propondo 180 trocas.

A empresa utilizou a sequência (1, 2, 6, 5, 3, 4, 16, 15, 14, 7, 13, 8, 10, 9, 12, 11), que realiza 224 trocas.

Na tabela 5.3 apresentamos um resumo do resultado.

Tabela 5.3: Resultados obtidos para a carteira 2 sem considerar posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	224	-
Xpress-MP	180	19, 7 %
MVMP	181	19, 2 %
MVMP/2-Part	180	19, 7 %
MVMP/3-Part	180	19, 7 %

5.1.2.2 Resultados considerando posições vazias

Para o caso generalizado, a sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 6 - 1, 1 - 1, 2 - 1, 3 - 1, 5 - 2, 7 - 1, 13 - 12, 14 - 2, 15 - 2, 16 - 2, 4 - 2, 8 - 2, 9 - 1, 10 - 1, 12 - 1, 11 - 1, O), a qual realiza 177 trocas.

As heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado retornaram a sequência (O, 2 - 1, 1 - 1, 6 - 1, 11 - 1, 12 - 1, 10 - 1, 9 - 1, 8 - 2, 4 - 2, 16 - 2, 15 - 2, 14 - 2, 3 - 1, 5 - 2, 7 - 1, 13 - 12, O), que totaliza 175 trocas de bolachas.

Aplicando a heurística Melhor Configuração o resultado obtido foi (O, 6 - 6, 1 - 6, 2 - 1, 3 - 1, 5 - 2, 7 - 1, 13 - 12, 14 - 2, 15 - 2, 16 - 2, 4 - 2, 8 - 2, 9 - 1, 10 - 7, 12 - 1, 11 - 1, O), realizando um total de 174 trocas.

Com a utilização das heurísticas Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração obtivemos a sequência (O, 2 - 1, 1 - 6, 6 - 6, 11 - 6, 12 - 1, 10 - 7, 9 - 1, 8 - 2, 4 - 2, 16 - 2, 15 - 2, 14 - 2, 13 - 12, 7 - 1, 5 - 2, 3 - 1, O), que realiza 172 trocas.

Por fim, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração forneceu a sequência (O, 2 - 1, 1 - 1, 6 - 1, 11 - 1, 12 - 1, 10 - 7, 9 - 1, 8 - 2, 4 - 2, 16 - 2, 15 - 2, 14 - 17, 13 - 17, 7 - 1, 5 - 2, 3 - 1, O), na qual são necessárias 172 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010), após 100 horas de execução em um computador com processador *Intel Core 2 Duo* de 3 GHz e 4 GB de *Ram* não obtiveram uma solução viável. Como alternativa, os autores dividiram o problema em quatro partes e obtiveram a sequência (O, 2 - 26, 1 - 1, 6 - 17, 12 - 11, 11 - 1, 9 - 1, 10 - 1, 8 - 3, 4 - 3, 15 - 3, 16 - 3, 14 - 1, 13 - 13, 7 - 21, 5 - 11, 3 - 21, O) que realiza 179 trocas de bolachas.

Note que com a utilização desta divisão o resultado obtido foi um sub-ótimo do problema, uma vez que a heurística MVMP-3Part-MC propôs uma sequência com um total de 172 trocas.

A sequência utilizada pela empresa foi (1, 2, 6, 5, 3, 4, 16, 15, 14, 7, 13, 8, 10, 9, 12, 11), que totaliza 224 trocas.

Na tabela 5.4 apresentamos um resumo do resultado.

Tabela 5.4: Resultados obtidos para a carteira 2 considerando posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	224	-
Xpress-MP	179	20, 1 %
MVMP	177	21 %
MVMP/2-Part	175	21, 9 %
MVMP/3-Part	174	22, 3 %
MVMP/MC	174	22, 3 %
MVMP/2-Part/MC	172	23, 2 %
MVMP/3-Part/MC	172	23, 2 %

5.1.3 Carteira 3

A carteira 3 é composta por 10 tubetes e um número máximo de 31 bolachas posicionadas no estaleiro para a confecção dos mesmos. Temos ainda 8 tipos diferentes de mandris.

5.1.3.1 Resultados sem considerar posições vazias

Neste caso, para as três heurísticas propostas para a resolução sem posições vazias, a sequência fornecida foi (O, 1, 8, 10, 9, 5, 4, 3, 6, 7, 2, O), propondo 142 trocas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 1, 8, 10, 9, 5, 4, 3, 7, 6, 2, O), que realiza 142 trocas de bolachas.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6, 7, 8, 9, 10), que totaliza 187 trocas.

Na tabela 5.5 apresentamos um resumo do resultado.

Tabela 5.5: Resultados obtidos para a carteira 3 sem considerar posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	187	-
Xpress-MP	142	24, 1 %
MVMP	142	24, 1 %
MVMP/2-Part	142	24, 1 %
MVMP/3-Part	142	24, 1 %

5.1.3.2 Resultados considerando posições vazias

Uma vez realizados os testes numéricos a sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 1 - 1, 8 - 1, 10 - 1, 9 - 1, 5 - 1, 4 - 1, 3 - 1, 6 - 1, 7 - 1, 2 - 1, O), propondo a realização de 142 trocas.

As heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado retornaram a mesma sequência, (O, 1 - 1, 8 - 1, 10 - 1, 9 - 2, 5 - 2, 4 - 1, 3 - 1, 6 - 2, 7 - 2, 2 - 1, O), que totaliza 138 trocas de bolachas.

A heurística Melhor Vizinho mais Próximo com a Melhor Configuração obteve a sequência (O, 1 - 1, 8 - 1, 10 - 1, 9 - 2, 5 - 2, 4 - 1, 3 - 1, 6 - 1, 7 - 1, 2 - 1, O), com um total de 139 trocas de bolachas.

Com a utilização da heurística Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração a sequência fornecida foi (O, 1 - 1, 8 - 1, 10 - 1, 9 - 2, 5 - 2, 4 - 1, 3 - 1, 6 - 2, 7 - 2, 2 - 1, O), na qual são necessárias 138 trocas de bolachas.

Ainda, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração retornou a sequência (O, 2 - 1, 6 - 17, 7 - 17, 3 - 1, 4 - 1, 5 - 2, 9 - 2, 10 - 1, 8 - 1, 1 - 1, O), totalizando 138 trocas de bolachas.

Por meio da implementação realizada por Fenato (2008) e Hoto *et al.* (2010) obtivemos a sequência (O, 1 - 1, 8 - 27, 10 - 13, 9 - 2, 5 - 2, 4 - 13, 3 - 16, 6 - 2, 7 - 2, 2 - 1, O), a qual propõe 138 trocas de bolachas.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6, 7, 8, 9 ,10), que realiza 187 trocas.

Na tabela 5.6 apresentamos um resumo do resultado.

Tabela 5.6: Resultados obtidos para a carteira 3 considerando posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	187	-
Xpress-MP	138	26, 2 %
MVMP	142	24, 1 %
MVMP/2-Part	138	26, 2 %
MVMP/3-Part	138	26, 2 %
MVMP/MC	139	25, 7 %
MVMP/2-Part/MC	138	26, 2 %
MVMP/3-Part/MC	138	26, 2 %

5.1.4 Carteira 4

Para a confecção da carteira 4, temos um total de 8 tubetes e o número máximo de bolachas posicionadas no estaleiro é 25, são utilizados ainda 5 tipos diferentes de mandris.

5.1.4.1 Resultados sem considerar posições vazias

Considerando a heurística Melhor Vizinho mais Próximo a sequência fornecida foi (O, 5, 4, 3, 2, 8, 7, 1, 6, O), propondo a realização de 100 trocas de bolachas.

O resultado obtido pela heurística Melhor Vizinho mais Próximo Biparticionado foi (O, 2, 3, 4, 5, 8, 7, 1, 6, O), que realiza 98 trocas de bolachas.

Por fim, a heurística Melhor Vizinho mais Próximo Triparticionado propõe 92 trocas de bolachas por meio da sequência (O, 1, 7, 8, 5, 2, 4, 3, 6, O).

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 6, 3, 4, 2, 5, 8, 7, 1, O), que realiza 92 trocas de bolachas.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6, 7, 8), que realiza 116 trocas.

Na tabela 5.7 apresentamos um resumo do resultado.

Tabela 5.7: Resultados obtidos para a carteira 4 sem considerar posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	116	-
Xpress-MP	92	20, 7 %
MVMP	100	13, 8 %
MVMP/2-Part	98	15, 5 %
MVMP/3-Part	92	20, 7 %

5.1.4.2 Resultados considerando posições vazias

A sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 5 - 1, 4 - 1, 3 - 1, 2 - 1, 8 - 1, 7 - 1, 1 - 2, 6 - 1, O), propondo a realização de 98 trocas.

A heurística Melhor Vizinho mais Próximo Biparticionado retornou a sequência, (O, 2 - 2, 3 - 2, 4 - 2, 5 - 2, 8 - 1, 7 - 1, 1 - 2, 6 - 1, O), que totaliza 94 trocas de bolachas.

Ainda, a heurística Melhor Vizinho mais Próximo Triparticionado forneceu a sequência de fabricação (O, 1 - 2, 7 - 1, 8 - 1, 5 - 2, 2 - 2, 4 - 2, 3 - 2, 6 - 1, O), que realiza 89 trocas.

Com a utilização ds heurística Melhor Vizinho mais Próximo com a Melhor Configuração obtivemos a sequência (O, 5 - 1, 4 - 1, 3 - 1, 2 - 1, 8 - 1, 7 - 1, 1 - 2, 6 - 1, O), com um total de 98 trocas.

Utilizando a heurísitica Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração a sequência fornecida foi (O, 2 - 2, 3 - 2, 4 - 2, 5 - 2, 8 - 1, 7 - 1, 1 - 2, 6 - 1, O), na qual são necessárias 94 trocas de bolachas.

A heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração retornou a sequência (O, 1 - 2, 7 - 1, 8 - 1, 5 - 2, 2 - 2, 4 - 2, 3 - 2, 6 - 1, O), que propõe 89 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 6 - 24, 3 - 17, 4 - 17, 2 - 12, 5 - 6, 8 - 11, 7 - 15, 1 - 4, O), que totaliza 89 trocas de bolachas.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6, 7, 8), que realiza 116 trocas.

Na tabela 5.8 apresentamos um resumo do resultado.

Tabela 5.8: Resultados obtidos para a carteira 4 considerando posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	116	-
Xpress-MP	89	23, 3 %
MVMP	100	13, 8 %
MVMP/2-Part	94	18, 9 %
MVMP/3-Part	89	23, 3 %
MVMP/MC	98	15, 5 %
MVMP/2-Part/MC	94	18, 9 %
MVMP/3-Part/MC	89	23, 3 %

5.1.5 Carteira 5

A carteira 5 é composta por 7 tubetes os quais utilizam-se de no máximo 23 bolachas posicionadas no estaleiro e 5 tipos diferentes de mandris.

5.1.5.1 Resultados sem considerar posições vazias

Com os dados fornecidos, as heurísticas Melhor Vizinho mais Próximo, Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado retornaram a sequência (O, 7, 6, 2, 4, 3, 5, 1, O), que totaliza 65 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a mesma sequência (O, 1, 5, 3, 4, 2, 6, 7, O) que realiza 65 trocas de bolachas.

A sequência utilizada pela empresa é (1, 2, 3, 4, 5, 6, 7), a qual realiza 72 trocas.

Na tabela 5.9 apresentamos um resumo do resultado.

Tabela 5.9: Resultados obtidos para a carteira 5 sem considerar posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	72	-
Xpress-MP	65	9, 7 %
MVMP	65	9, 7 %
MVMP/2-Part	65	9, 7 %
MVMP/3-Part	65	9, 7 %

5.1.5.2 Resultados considerando posições vazias

As heurísticas implementadas retornaram a sequência (O, 1 - 1, 3 - 13, 4 - 13, 5 - 13, 6 - 13, 2 - 1, 7 - 1, O), que totaliza 64 trocas de bolachas.

Utilizando o programa desenvolvido em Fenato (2008) a sequência determinada foi (O, 7 - 16, 2 - 12, 6 - 13, 5 - 13, 4 - 13, 3 - 13, 1 - 22, O) realizando 64 trocas.

Ainda, a sequência utilizada pela empresa é (1, 2, 3, 4, 5, 6, 7), a qual realiza 72 trocas.

Na tabela 5.10 apresentamos um resumo do resultado.

Tabela 5.10: Resultados obtidos para a carteira 5 considerando posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	72	-
Xpress-MP	64	11, 1 %
MVMP	64	11, 1 %
MVMP/2-Part	64	11, 1 %
MVMP/3-Part	64	11, 1 %
MVMP/MC	64	11, 1 %
MVMP/2-Part/MC	64	11, 1 %
MVMP/3-Part/MC	64	11, 1 %

5.1.6 Carteira 6

Na carteira 6, temos um total de 6 tubetes, um número máximo de 27 bolachas posicionadas no estaleiro e 5 tipos diferentes de mandris.

5.1.6.1 Resultados sem considerar posições vazias

Considerando a heurística Melhor Vizinho mais Próximo, a sequência fornecida foi (O, 4, 1, 2, 3, 5, 6, O), propondo a realização de 69 trocas de bolachas.

O resultado obtido pelas heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado foi (O, 6, 5, 3, 2, 1, 4, O), que também realiza 69 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 6, 5, 3, 2, 1, 4, O), que realiza 69 trocas de bolachas.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6), que realiza 87 trocas de bolachas.

Na tabela 5.11 apresentamos um resumo do resultado.

Tabela 5.11: Resultados obtidos para a carteira 6 sem considerar posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	87	-
Xpress-MP	69	20, 7 %
MVMP	69	20, 7 %
MVMP/2-Part	69	20, 7 %
MVMP/3-Part	69	20, 7 %

5.1.6.2 Resultados considerando posições vazias

Por meio das heurísticas Melhor Vizinho mais Próximo e Melhor Vizinho mais Próximo com a Melhor Configuração a sequência encontrada foi (O, 4 - 1, 1 - 1, 2 - 1, 3 - 1, 5 - 1, 6 - 1, O), propondo a realização de 69 trocas.

As heurísticas Melhor Vizinho mais Próximo Biparticionado, Melhor Vizinho mais Próximo Triparticionado, Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração e Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração retornaram a sequência, (O, 6 - 1, 5 - 1, 3 - 1, 2 - 1, 1 - 1, 4 - 1, O), que totaliza 69 trocas de bolachas.

A sequência fornecida pelo modelo de Fenato (2008) e Hoto *et al.* foi (O, 1 - 1, 4 - 1, 3 - 1, 2 - 1, 5 - 1, 6 - 20, O), a qual realiza 69 trocas de bolachas.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6), que realiza 87 trocas.

Na tabela 5.12 apresentamos um resumo do resultado.

Tabela 5.12: Resultados obtidos para a carteira 6 considerando posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	87	-
Xpress-MP	69	20, 7 %
MVMP	69	20, 7 %
MVMP/2-Part	69	20, 7 %
MVMP/3-Part	69	20, 7 %
MVMP/MC	69	20, 7 %
MVMP/2-Part/MC	69	20, 7 %
MVMP/3-Part/MC	69	20, 7 %

5.1.7 Carteira 7

Na carteira 7, temos um total de 6 tubetes os quais necessitam de no máximo 23 bolachas posicionadas no estaleiro e 5 tipos de mandris.

5.1.7.1 Resultados sem considerar posições vazias

Utilizando as heurísticas Melhor Vizinho mais Próximo, Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado, a sequência fornecida foi (O, 1, 2, 6, 5, 3, 4, O), propondo a realização de 94 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 4, 3, 1, 2, 6, 5, O), que realiza 94 trocas de bolachas.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6), que realiza 107 trocas de bolachas.

Na tabela 5.13 apresentamos um resumo do resultado.

Tabela 5.13: Resultados obtidos para a carteira 7 sem considerar posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	107	-
Xpress-MP	94	12, 1 %
MVMP	94	12, 1 %
MVMP/2-Part	94	12, 1 %
MVMP/3-Part	94	12, 1 %

5.1.7.2 Resultados considerando posições vazias

Neste caso, a sequência fornecida pelas heurísticas Melhor Vizinho mais Próximo e Melhor Vizinho mais Próximo com a Melhor Configuração foi (O, 3 - 1, 1 - 2, 2 - 2, 6 - 1, 5 - 1, 4 - 1, O), propondo a realização de 93 trocas.

As heurísticas Melhor Vizinho mais Próximo Biparticionado, Melhor Vizinho mais Próximo Triparticionado, Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração e Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuraçãc retornaram a sequência (O, 5 - 1, 6 - 1, 2 - 2, 1 - 2, 3 - 1, 4 - 1, O), totalizando 93 trocas de bolachas.

Fenato (2008) e Hoto *et al.* (2010) forneceram a sequência (O, 5 - 22, 6 - 12, 2 - 2, 1 - 11, 3 - 21, 4 - 21, O), que realiza 93 trocas de bolachas.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6), que realiza 107 trocas.

Na tabela 5.14 apresentamos um resumo do resultado.

Tabela 5.14: Resultados obtidos para a carteira 7 considerando posições vazias

Método	Total de Trocas	Percentual de Redução (comparado com a empresa)
Empresa	107	-
Xpress-MP	93	13, 1 %
MVMP	93	13, 1 %
MVMP/2-Part	93	13, 1 %
MVMP/3-Part	93	13, 1 %
MVMP/MC	93	13, 1 %
MVMP/2-Part/MC	93	13, 1 %
MVMP/3-Part/MC	93	13, 1 %

5.2 Custos em Relação à Movimentos de Bolachas

Neste momento são apresentados os resultados obtidos considerando como custo o número de movimentos de bolachas necessários para a confecção dos tubetes que compõem as diferentes carteiras.

Fenato (2008) considerou apenas a ideia de relacionar os custos com o número de trocas de bolachas para a modelagem exata. Para efeito de comparação foi implementado um programa que recebe como entrada as soluções fornecidas pelo modelo de Fenato (2008) e retorna a quantidade necessária de movimentos para executá-las.

Para esta abordagem foram utilizadas as carteiras citadas no Apêndice B.

5.2.1 Carteira 1

Para a carteira 1, temos 10 tubetes a serem fabricados, os quais possuem o mesmo mandril e utilizam no máximo 10 bolachas para suas configurações.

5.2.1.1 Resultados sem considerar posições vazias

Utilizando a heurística Melhor Vizinho mais Próximo obtivemos a sequência (O, 7, 1, 4, 2, 10, 5, 6, 8, 9, 3, O), que totaliza 98 movimentos de bolachas.

As heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado retornaram a sequência (O, 9, 8, 6, 5, 10, 2, 4, 1, 7, 3, O), com o total de 98 movimentos.

Fenato (2008) e Hoto *et al.* (2010) encontraram a sequência (O, 8, 9, 2, 10, 4, 5, 6, 1, 7, 3, O), com um total de 103 movimentos.

Ainda, a sequência utilizada pela empresa é (9, 8, 2, 5, 6, 3, 10, 1, 7, 4), a qual realiza 116 movimentos.

Na tabela 5.15 apresentamos um resumo do resultado.

Tabela 5.15: Resultados obtidos para a carteira 1 sem considerar posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	116	-
Xpress-MP	103	11, 2 %
MVMP	98	15, 5 %
MVMP/2-Part	98	15, 5 %
MVMP/3-Part	98	15, 5 %

5.2.1.2 Resultados considerando posições vazias

A heurística Melhor Vizinho mais Próximo retornou a sequência (O, 7 - 1, 1 - 1, 4 - 4, 2 - 3, 10 - 1, 9 - 2, 5 - 1, 6 - 1, 8 - 1, 3 - 1, O), formada pelo tubete e sua respectiva configuração, que realiza 88 movimentos de bolachas.

As heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado forneceram a sequência (O, 9 - 1, 8 - 2, 10 - 2, 2 - 2, 4 - 2, 5 - 2, 6 - 1, 1 - 2, 3 - 2, 7 - 2, O), que totaliza 86 movimentos de bolachas.

Com a utilização das heurísticas Melhor Vizinho mais Próximo com a Melhor Configuração e Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração obtivemos a sequência (O, 3 - 9, 8 - 2, 9 - 2, 2 - 1, 10 - 1, 4 - 4, 1 - 1, 5 - 4, 6 - 7, 7 - 1, O), que totaliza 84 movimentos.

Por fim, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração retornou a sequência (O, 6 - 1, 5 - 4, 4 - 1, 2 - 3, 10 - 1, 9 - 2, 8 - 2, 3 - 2, 1 - 2, 7 - 2, O) realizando 83 movimentos de bolachas.

Fenato (2008) e Hoto *et al.* (2010) encontraram a sequência (O, 3 - 9, 7 - 9, 1 - 9, 4 - 3, 6 - 9, 5 - 2, 10 - 2, 2 - 2, 8 - 2, 9 - 5, O), com um total de 89 movimentos.

Ainda, a sequência utilizada pela empresa é (9, 8, 2, 5, 6, 3, 10, 1, 7, 4), a qual realiza 116 movimentos.

Na tabela 5.16 apresentamos um resumo do resultado.

Tabela 5.16: Resultados obtidos para a carteira 1 considerando posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	116	-
Xpress-MP	89	23, 3 %
MVMP	89	23, 3 %
MVMP/2-Part	88	24, 1 %
MVMP/3-Part	86	25, 9 %
MVMP/MC	84	27, 6 %
MVMP/2-Part/MC	84	27, 6 %
MVMP/3-Part/MC	83	28, 4%

5.2.2 Carteira 2

Na carteira 2, temos um total de 16 tubetes os quais necessitam no máximo de 26 bolachas posicionadas no estaleiro para sua fabricação e 9 tipos diferentes de mandris.

5.2.2.1 Resultados sem considerar posições vazias

A sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 2, 1, 6, 8, 4, 15, 16, 14, 3, 5, 9, 10, 12, 11, 13, 7, O), propondo a realização de 321 movimentos.

A heurística Melhor Vizinho mais Próximo Biparticionado retornou a sequência (O, 6, 2, 1, 5, 3, 14, 16, 15, 4, 8, 11, 12, 10, 9, 13, 7, O), que totaliza 318 movimentos de bolachas.

Utilizando a heurística Melhor Vizinho mais Próximo Triparticionado a sequência obtida foi (O, 3, 5, 1, 2, 6, 11, 12, 10, 9, 8, 4, 15, 16, 14, 13, 7, O), realizando 316 movimentos.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 2, 1, 6, 8, 9, 10, 12, 11, 4, 15, 16, 14, 3, 5, 13, 7, O), totalizando 321 movimentos.

A empresa utilizou a sequência (1, 2, 6, 5, 3, 4, 16, 15, 14, 7, 13, 8, 10, 9, 12, 11), que realiza 370 movimentos.

Na tabela 5.17 apresentamos um resumo do resultado.

Tabela 5.17: Resultados obtidos para a carteira 2 sem considerar posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	370	-
Xpress-MP	321	13, 2 %
MVMP	321	13, 2 %
MVMP/2-Part	318	14, 1 %
MVMP/3-Part	316	14, 6 %

5.2.2.2 Resultados considerando posições vazias

Para a carteira 2 a sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 6 - 5, 1 - 5, 2 - 1, 5 - 12, 3 - 1, 14 - 4, 4 - 1, 15 - 1, 16 - 1, 8 - 1, 9 - 1, 10 - 1, 12 - 1, 11 - 1, 13 - 1, 7 - 2, O), propondo a realização de 302 movimentos.

A heurística Melhor Vizinho mais Próximo Biparticionado retornou a sequência (O, 2 - 1, 1 - 1, 6 - 1, 11 - 1, 12 - 1, 10 - 1, 9 - 1, 8 - 1, 16 - 1, 15 - 1, 4 - 1, 14 - 4, 3 - 1, 5 - 2, 7 - 1, 13 - 12, O), que totaliza 302 movimentos de bolachas.

A heurística Melhor Vizinho mais Próximo Triparticionado forneceu a sequência (O, 3 - 1, 5 - 12, 2 - 1, 1 - 1, 6 - 1, 11 - 1, 12 - 1, 10 - 1, 9 - 1, 8 - 1, 16 - 1, 15 - 1, 4 - 1, 14 - 4, 13 - 12, 7 - 1, O), com um total de 300 movimentos de bolachas.

Aplicando as heurísticas Melhor Vizinho mais Próximo com a Melhor Configuração e Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração o resultado obtido em ambos os casos foi (O, 6 - 1, 1 - 12, 2 - 1, 5 - 12, 3 - 1, 14 - 4, 4 - 1, 15 - 1, 16 - 1, 8 - 1, 9 - 1, 10 - 8, 12 - 1, 11 - 1, 13 - 13, 7 - 1, O), realizando um total de 292 movimentos.

Por fim, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração forneceu a sequência (O, 6 - 1, 1 - 12, 2 - 1, 5 - 12, 3 - 1, 14 - 4, 4 - 1, 16 - 2, 15 - 2, 11 - 1, 12 - 1, 10 - 8, 9 - 1, 8 - 1, 13 - 12, 7 - 1, O), para a qual são necessários 290 movimentos de bolachas.

Fenato (2008) e Hoto *et al.* (2010), após 100 horas de execução em um computador com processador *Intel Core 2 Duo* de 3 GHz e 4 GB de *Ram* não obtiveram uma solução viável. Como alternativa, os autores dividiram o problema em quatro partes e obtiveram a sequência (O, 2 - 26, 1 - 1, 6 - 17, 12 - 11, 11 - 1, 9 - 1, 10 - 1, 8 - 3, 4 - 3, 15 - 3, 16 - 3, 14 - 1, 13 - 13, 7 - 21, 5 - 11, 3 - 21, O), totalizando 310 movimentos.

A sequência utilizada pela empresa foi (1, 2, 6, 5, 3, 4, 16, 15, 14, 7, 13, 8, 10, 9, 12, 11), que realiza 370 movimentos.

Na tabela 5.18 apresentamos um resumo do resultado.

Tabela 5.18: Resultados obtidos para a carteira 2 considerando posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	370	-
Xpress-MP	310	16, 2 %
MVMP	315	14, 9 %
MVMP/2-Part	302	18, 4 %
MVMP/3-Part	300	18, 9 %
MVMP/MC	292	21, 1 %
MVMP/2-Part/MC	292	21, 1 %
MVMP/3-Part/MC	290	21, 6 %

5.2.3 Carteira 3

Na carteira 3, temos um total de 10 tubetes e um número máximo de 31 bolachas posicionadas no estaleiro para a confecção dos tubetes e 8 tipos diferentes de mandris.

5.2.3.1 Resultados sem considerar posições vazias

A heurística Melhor Vizinho mais Próximo forneceu a sequência (O, 1, 8, 9, 10, 3, 4, 5, 6, 7, 2, O), com um total de 246 movimentos de bolachas.

Utilizando a heurística Melhor Vizinho mais Próximo Biparticionado obtivemos a sequência (O, 1, 8, 10, 9, 4, 3, 5, 6, 7, 2, O), totalizando 244 movimentos.

Por fim a heurística Melhor Vizinho mais Próximo Triparticionado forneceu a sequência (O, 1, 8, 10, 9, 3, 4, 5, 7, 6, 2, O), propondo a realização de 240 movimentos.

Fenato (2008) e Hoto *et al.* (2010) encontraram a sequência (O, 1, 8, 10, 9, 5, 4, 3, 7, 6, 2, O), que realiza 246 movimentos de bolachas.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6, 7, 8, 9 ,10), que realiza 313 movimentos.

Na tabela 5.19 apresentamos um resumo do resultado.

Tabela 5.19: Resultados obtidos para a carteira 3 sem considerar posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	313	-
Xpress-MP	246	21, 4 %
MVMP	246	21, 4 %
MVMP/2-Part	244	22, 1 %
MVMP/3-Part	240	23, 3 %

5.2.3.2 Resultados considerando posições vazias

Uma vez realizados os testes numéricos, a sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 1 - 9, 8 - 9, 9 - 5, 10 - 1, 3 - 1, 4 - 1, 5 - 2, 6 - 2, 7 - 2, 2 - 1, O), propondo a realização de 234 trocas.

A heurística Melhor Vizinho mais Próximo Biparticionado retornou a sequência, (O, 1 - 4, 8 - 4, 10 - 1, 9 - 1, 4 - 1, 3 - 1, 5 - 2, 6 - 2, 7 - 2, 2 - 1, O), que totaliza 232 trocas de bolachas.

Por outro lado, a heurística Melhor Vizinho mais Próximo Triparticionado forneceu a sequência (O, 1 - 4, 8 - 4, 10 - 1, 9 - 1, 3 - 1, 4 - 1, 5 - 2, 6 - 2, 7 - 2, 2 - 1, O), que realiza um total de 232 movimentos de bolachas.

A heurística Melhor Vizinho mais Próximo com a Melhor Configuração retornou a sequência (O, 1 - 9, 8 - 9, 9 - 5, 10 - 1, 3 - 1, 4 - 1, 5 - 2, 6 - 2, 7 - 2, 2 - 1, O), com um total de 234 movimentos de bolachas.

Com a utilização da heurística Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração a sequência fornecida foi (O, 1 - 4, 8 - 4, 10 - 1, 9 - 1, 4 - 1, 3 - 1, 5 - 2, 6 - 2, 7 - 2, 2 - 1, O), na qual são necessárias 232 movimentos de bolachas.

Ainda, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração retornou a sequência (O, 1 - 4, 8 - 4, 10 - 1, 9 - 1, 4 - 1, 3 - 1, 5 - 2, 7 - 22, 6 - 1, 2 - 2, O), totalizando 231 movimentos de bolachas.

Considerando a abordagem realizada por Fenato (2008) a sequência (O, 1 - 1, 8 - 27, 10 - 13, 9 - 2, 5 - 2, 4 - 13, 3 - 16, 6 - 2, 7 - 2, 2 - 1, O), que totaliza 246 movimentos.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6, 7, 8, 9 ,10), que realiza 313 movimentos.

Na tabela 5.20 apresentamos um resumo do resultado.

Tabela 5.20: Resultados obtidos para a carteira 3 considerando posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	313	-
Xpress-MP	246	21, 4 %
MVMP	234	25, 2 %
MVMP/2-Part	232	25, 9 %
MVMP/3-Part	232	25, 9 %
MVMP/MC	234	24, 6 %
MVMP/2-Part/MC	232	25, 9 %
MVMP/3-Part/MC	231	26, 2 %

5.2.4 Carteira 4

Na carteira 4, temos um total de 8 tubetes e o número máximo de bolachas posicionadas no estaleiro é 25. Tem-se ainda 5 tipos diferentes de mandris.

5.2.4.1 Resultados sem considerar posições vazias

Considerando a heurística Melhor Vizinho mais Próximo, a sequência fornecida foi (O, 1, 7, 8, 5, 4, 3, 2, 6 , O), propondo a realização de 161 movimentos de bolachas.

O resultado obtido pelas heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado foi (O, 1, 7, 8, 2, 3, 4, 5, 6, O), que realiza 155 movimentos de bolachas.

Fenato (2008) e Hoto *et al.* (2010) encontraram a sequência (O, 6, 3, 4, 2, 5, 8, 7, 1, O), com um total de 161 movimentos.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6, 7, 8), que realiza 170 movimentos.

Na tabela 5.21 apresentamos um resumo do resultado.

Tabela 5.21: Resultados obtidos para a carteira 4 sem considerar posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	170	-
Xpress-MP	161	5, 3 %
MVMP	161	5, 3 %
MVMP/2-Part	155	8, 8 %
MVMP/3-Part	155	8, 8 %

5.2.4.2 Resultados considerando posições vazias

A sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 1 - 4, 7 - 1, 8 - 1, 5 - 2, 4 - 2, 3 - 2, 2 - 1, 6 - 1, O), propondo a realização de 151 movimentos.

Ainda, as heurísticas Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado forneceram a sequência de confecção (O, 2 - 1, 3 - 2, 4 - 2, 5 - 2, 8 - 1, 7 - 1, 1 - 4, 6 - 1, O), que realiza 146 movimentos.

Com a utilização da heurística Melhor Configuração, obtivemos a sequência (O, 1 - 4, 7 - 1, 8 - 1, 5 - 2, 4 - 2, 3 - 2, 2 - 1, 6 - 1, O), com um total de 151 movimentos.

Utilizando a heurísitica Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração a sequência fornecida foi (O, 2 - 1, 3 - 2, 4 - 2, 5 - 2, 8 - 1, 7 - 1, 1 - 4, 6 - 1, O), na qual são necessárias 146 movimentos de bolachas

Ainda, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração retornou a sequência (O, 2 - 1, 4 - 1, 3 - 1, 5 - 8, 8 - 1, 7 - 1, 1 - 4, 6 - 1 , O), que propõe 146 movimentos de bolachas.

Considerando o programa realizado por Fenato (2008) e Hoto *et al.* (2010) encontramos a sequência (O, 6 - 24, 3 - 17, 4 - 17, 2 - 12, 5 - 6, 8 - 11, 7 - 15, 1 - 4, O), com um total de 149 movimentos de bolachas.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6, 7, 8), que realiza 170 movimentos.

Na tabela 5.22 apresentamos um resumo do resultado.

Tabela 5.22: Resultados obtidos para a carteira 4 considerando posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	170	-
Xpress-MP	149	12, 4 %
MVMP	151	11, 2 %
MVMP/2-Part	146	14, 1 %
MVMP/3-Part	146	14, 1 %
MVMP/MC	151	11, 2 %
MVMP/2-Part/MC	146	14, 1 %
MVMP/3-Part/MC	146	14, 1 %

5.2.5 Carteira 5

A carteira 5 é composta por 7 tubetes os quais utilizam-se de no máximo 23 bolachas posicionadas no estaleiro e 5 tipos diferentes de mandris.

5.2.5.1 Resultados sem considerar posições vazias

Com os dados fornecidos, as heurísticas Melhor Vizinho mais Próximo, Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado retornaram a sequência (O, 1, 3, 4, 5, 6, 2, 7, O), que totaliza 122 movimentos de bolachas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 1, 5, 3, 4, 2, 6, 7, O), com um total de 136 movimentos.

A sequência utilizada pela empresa é (1, 2, 3, 4, 5, 6, 7), a qual realiza 122 movimentos.

Na tabela 5.23 apresentamos um resumo do resultado.

Tabela 5.23: Resultados obtidos para a carteira 5 sem considerar posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	122	-
Xpress-MP	136	-
MVMP	122	0 %
MVMP/2-Part	122	0 %
MVMP/3-Part	122	0 %

5.2.5.2 Resultados considerando posições vazias

Neste caso, as heurísticas implementadas retornaram todas a mesma sequência, (O, 1 - 2, 2 - 2, 4 - 2, 3 - 2, 5 - 2, 6 - 2, 7 - 2, O), formada pelo tubete e sua respectiva configuração, que totaliza 104 movimentos de bolachas.

O modelo de Fenato (2008) e Hoto *et al.* (2010) obteve a sequência (O, 7 - 16, 2 - 12, 6 - 13, 5 - 13, 4 - 13, 3 - 13, 1 - 22, O), com um total de 112 movimentos.

Ainda, a sequência utilizada pela empresa é (1, 2, 3, 4, 5, 6, 7), a qual realiza 122 trocas.

Na tabela 5.24 apresentamos um resumo do resultado.

Tabela 5.24: Resultados obtidos para a carteira 5 considerando posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	122	-
Xpress-MP	112	8, 2 %
MVMP	104	14, 8 %
MVMP/2-Part	104	14, 8 %
MVMP/3-Part	104	14, 8 %
MVMP/MC	104	14, 8 %
MVMP/2-Part/MC	104	14, 8 %
MVMP/3-Part/MC	104	14, 8 %

5.2.6 Carteira 6

Na carteira 6, temos um total de 6 tubetes, um número máximo de 27 bolachas posicionadas no estaleiro e 5 tipos diferentes de mandris.

5.2.6.1 Resultados sem considerar posições vazias

Considerando as heurísticas Melhor Vizinho mais Próximo, Melhor Vizinho mais Próximo Biparticionado e Melhor Vizinho mais Próximo Triparticionado a sequência fornecida foi (O, 3, 2, 5, 6, 1, 4, O), propondo a realização de 127 movimentos de bolachas.

Fenato (2008) e Hoto *et al.* (2010) encontraram a sequência (O, 6, 5, 3, 2, 1, 4, O), com um total de 132 movimentos.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6), que realiza 166 movimentos de bolachas.

Na tabela 5.25 apresentamos um resumo do resultado.

Tabela 5.25: Resultados obtidos para a carteira 6 sem considerar posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	166	-
Xpress-MP	132	20, 5 %
MVMP	127	23, 5 %
MVMP/2-Part	127	23, 5 %
MVMP/3-Part	127	23, 5 %

5.2.6.2 Resultados considerando posições vazias

Na carteira 6, por meio das heurísticas Melhor Vizinho mais Próximo, Melhor Vizinho mais Próximo Biparticionado, Melhor Vizinho mais Próximo Triparticionado, Melhor Vizinho mais Próximo com a Melhor Configuração, Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração e Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração a sequência encontrada foi (O, 3 - 2, 2 - 2, 5 - 2, 6 - 2, 1 - 1, 4 - 1, O), propondo a realização de 124 movimentos.

A implementação de Fenato (2008) e Hoto *et al.* (2010) encontrou a sequência (O, 1 - 1, 4 - 1, 3 - 1, 2 - 1, 5 - 1, 6 - 20, O), com um total de 134 movimentos.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6), que realiza 166 movimentos.

Na tabela 5.26 apresentamos um resumo do resultado.

Tabela 5.26: Resultados obtidos para a carteira 6 considerando posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	166	-
Xpress-MP	134	19, 3 %
MVMP	124	25, 3 %
MVMP/2-Part	124	25, 3 %
MVMP/3-Part	124	25, 3 %
MVMP/MC	124	25, 3 %
MVMP/2-Part/MC	124	25, 3 %
MVMP/3-Part/MC	124	25, 3 %

5.2.7 Carteira 7

Na carteira 7, temos um total de 6 tubetes os quais necessitam de no máximo 23 bolachas posicionadas no estaleiro e 5 tipos de mandris.

5.2.7.1 Resultados sem considerar posições vazias

Considerando a heurística Melhor Vizinho mais Próximo a sequência fornecida foi (O, 3, 2, 6, 5, 1, 4, O), propondo a realização de 164 movimentos de bolachas.

A heurística Melhor Vizinho mais Próximo Biparticionado retornou a sequência (O, 3, 1, 2, 5, 6, 4, O), totalizando 158 movimentos.

Utilizando a heurística Melhor Vizinho mais Próximo Triparticionado a sequência obtida foi (O, 3, 2, 5, 6, 1, 4, O), que realiza 158 movimentos de bolachas.

Fenato (2008) e Hoto *et al.* (2010) determinaram a sequência (O, 4, 3, 1, 2, 6, 5, O), com um total de 164 movimentos.

A empresa utilizou a sequência (1, 2, 3, 4, 5, 6), que realiza 178 movimentos de bolachas.

Na tabela 5.27 apresentamos um resumo do resultado.

Tabela 5.27: Resultados obtidos para a carteira 7 sem considerar posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	178	-
Xpress-MP	164	7, 9 %
MVMP	164	7, 9 %
MVMP/2-Part	158	11, 2 %
MVMP/3-Part	158	11, 2 %

5.2.7.2 Resultados considerando posições vazias

Para a carteira 7, a sequência fornecida pela heurística Melhor Vizinho mais Próximo foi (O, 3 - 1, 2 - 1, 6 - 1, 5 - 1, 1 - 1, 4 - 1, O), propondo a realização de 164 trocas.

A heurística Melhor Vizinho mais Próximo Biparticionado retornou a sequência (O, 3 - 1, 1 - 1, 2 - 1, 5 - 1, 6 - 1, 4 - 1, O), totalizando 158 movimentos de bolachas. Ainda, a heurística Melhor Vizinho mais Próximo Triparticionado retornou a sequência (O, 3 - 1, 1 - 2, 6 - 1, 5 - 1, 2 - 1, 4 - 1, O), que totaliza 154 movimentos.

Utilizando a heurística Melhor Vizinho mais Próximo com a Melhor Configuração foi obtida a sequência (O, 4 - 1, 2 - 1, 6 - 1, 5 - 1, 1 - 2, 3 - 1, O), a qual propõe 160 movimentos de bolachas. A heurística Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração retornou a sequência (O, 3 - 1, 1 - 2, 2 - 1, 5 - 1, 6 - 1, 4 - 1, O), propondo 154 movimentações. Por fim, a heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração obtive a sequência (O, 3 - 1, 1 - 2, 6 - 1, 5 - 1, 2 - 1, 4 - 1, O), com um total de 154 movimentos de bolachas.

Fenato (2008) e Hoto *et al.* (2010) obtiveram a sequência (O, 5 - 22, 6 - 12, 2 - 2, 1 - 11, 3 - 21, 4 - 21, O), com um total de 160 movimentos.

A sequência utilizada pela empresa foi (1, 2, 3, 4, 5, 6), que realiza 178 movimentos.

Na tabela 5.28 apresentamos um resumo do resultado.

Tabela 5.28: Resultados obtidos para a carteira 7 considerando posições vazias

Método	Total de Movimentos	Percentual de Redução (comparado com a empresa)
Empresa	178	-
Xpress-MP	160	10, 1 %
MVMP	164	7, 9 %
MVMP/2-Part	158	11, 2 %
MVMP/3-Part	154	13, 5 %
MVMP/MC	160	10, 1 %
MVMP/2-Part/MC	154	13, 5 %
MVMP/3-Part/MC	154	13, 5 %

5.3 Considerações Finais

Apresentamos a seguir, as tabelas que contemplam as 7 carteiras utilizadas para os testes numéricos e o número de trocas de bolachas fornecido pela empresa e pelas heurísticas abordadas nesta dissertação sem considerar posições vazias (Tabela 5.29) e considerando posições vazias (Tabela 5.30):

Tabela 5.29: Resultados obtidos sem considerar posições vazias

Carteira	MVMP	MVMP/ 2-Part	MVMP/ 3-Part	Solução da Empresa	Melhor Solução
1	45	45	44	60	44
2	181	180	180	227	180
3	142	142	142	188	142
4	100	98	92	117	92
5	65	65	65	72	65
6	69	69	69	87	69
7	94	94	94	107	94

Tabela 5.30: Resultados obtidos considerando posições vazias

Carteira	MVMP	MVMP/ 2-Part	MVMP/ 3-Part	MVMP/ MC	MVMP/ 2-Part/ MC	MVMP/ 3-Part/ MC	Solução da Empresa	Melhor Solução
1	44	41	41	40	40	38	60	38
2	181	175	175	175	172	172	227	172
3	142	138	138	139	138	138	188	138
4	100	94	89	98	94	89	117	89
5	65	64	64	64	64	64	72	64
6	69	69	69	69	69	69	87	69
7	94	93	93	93	93	93	107	93

Ainda, seguem as tabelas que fornecem a quantidade necessária de movimentos de bolachas para a confecção das sete carteiras disponibilizadas pela empresa para os testes numéricos (Tabela 5.31 e Tabela 5.32).

Tabela 5.31: Resultados obtidos sem considerar posições vazias

Carteira	MVMP	MVMP/ 2-Part	MVMP/ 3-Part	Solução da Empresa	Melhor Solução
1	98	98	98	116	98
2	321	318	316	370	316
3	246	244	240	313	240
4	161	155	155	170	155
5	122	122	122	122	122
6	127	127	127	166	127
7	164	158	158	178	158

Os resultados obtidos por meio de heurísticas clássicas e adaptações das mesmas forneceram competitiva qualidade quando comparados com os obtidos por meio da modelagem

Tabela 5.32: Resultados obtidos considerando posições vazias

Carteira	MVMP	MVMP/ 2-Part	MVMP/ 3-Part	MVMP/ MC	MVMP/ 2-Part/ MC	MVMP/ 3-Part/ MC	Solução da Empresa	Melhor Solução
1	146	89	88	86	82	82	83	116
2	315	302	300	292	292	290	370	290
3	240	236	234	236	232	231	313	231
4	161	146	146	151	146	146	170	146
5	104	104	104	104	104	104	122	104
6	124	124	124	124	124	124	166	124
7	164	158	154	160	154	154	178	154

exata utilizando o *solver Xpress-MP*, o que indica a relevância do estudo de heurísticas para a resolução do problema, uma vez que sua utilização forneceu reduções nos custos de até 32, 1% para o caso de trocas de bolachas e 28, 4% calculando os custos como movimentos de bolachas.

As heurísticas atingiram a melhor solução conhecida em todos os casos em que temos esta informação. Para o caso da carteira 2, no qual Fenato (2008) e Hoto *et al.* (2010) subdividiram a carteira, o resultado obtido pelas heurísticas apresentou uma redução maior do que a apresentada pelos autores.

Apresentamos aqui a abordagem em que os custos são considerados como trocas de bolachas para fins de conhecimento de todo o processo de resolução do problema por meio da modelagem matemática. Assim, para a utilização da empresa, a metodologia que fornece a redução real de custos do processo é a modelagem que calcula as movimentações das bolachas, ou ainda, que levam em consideração não apenas o número de bolachas distintas na sequência, mas também suas posições e quantas bolachas devem se deslocar para que estas bolachas sejam trocadas. A modelagem considerando a movimentação de bolachas é a metodologia utilizada pela empresa para a reposição de bolachas.

Ainda, a abordagem do problema por meio de heurísticas é válida não apenas pelo retorno de uma resposta viável, mas também pela rapidez em que solucionam o problema e fornecem uma sequência de fabricação com um custo reduzido.

Como realizamos uma abordagem heurística, não temos como garantir a otimalidade das respostas para o caso dos custos em relação aos movimentos, nem mesmo quão próximas estão do ótimo do problema, uma vez que a modelagem exata apresentada por Fenato (2008), é baseada apenas na troca de bolachas.

Desse modo, buscamos nos Algoritmos Genéticos uma possibilidade de melhora de solução. Porém, após o estudo do assunto e do modo como o método contribui para a procura da solução, encontramos alguns obstáculos, os quais não foram solucionados.

Os Algoritmos Genéticos utilizam-se de operadores cruzamento e mutação para a geração de novas soluções, assim uma vez que estes cruzamentos são realizados de maneira genérica, sem atribuição de custos, nos deparamos com o problema de ordenação em relação à fabricação de tubetes com o mesmo mandril, isto é, ao cruzarmos as soluções os mandris iguais não permaneceriam em sequência, gerando assim uma solução não-viável sendo o custo da troca de mandril muito elevado.

Nossa proposta para investigações futuras consiste no desenvolvimento de cruzamentos específicos para o Problema da Tubeteira, em busca de novas soluções por meio dos Algoritmos Genéticos.

CONCLUSÃO

O desenvolvimento deste trabalho iniciou-se com o estudo do Problema do Caixeiro Viajante Generalizado, suas principais formulações e maneiras de resolvê-lo. Assim, nos remetemos ao aprofundamento na resolução do problema de uma máquina Tubeteira, estudo que inicializou-se em Fenato (2008).

Consideramos para este problema a utilização de heurísticas, desenvolvendo assim um *software* para a resolução das diversas carteiras que a empresa possa produzir, realizamos ainda modificações na modelagem segundo um E-PCVG descrita por Fenato (2008) a fim de abranger a restrição apresentada pelos diferentes tipos de mandris que compõem uma carteira.

Ainda, a relevância deste trabalho está no caso em que são estudadas carteiras com um número elevado de tubetes a serem fabricados, pois a modelagem exata torna-se inviável e ineficiente nestas situações. Neste trabalho propomos o cálculo de custos segundo movimentações das bolachas no estaleiro a fim de apresentar um modelo que atenda às reais necessidades e dificuldades da empresa em questão, uma vez que nosso objetivo é nos aproximar ao máximo da situação descrita pelo fabricante, para que hajam verdadeiras reduções de custos e consequentemente um aumento significativo nos lucros da indústria.

Como sugerido em Fenato (2008), implementamos heurísticas clássicas como, Melhor Vizinho mais Próximo, adapatações de 2-Opt e 3-Opt e foi desenvolvida também uma heurística que analisa as configurações possíveis de uma carteira devido a permutação da posição vazia, denominada como Melhor Configuração.

A abordagem por meio de heurísticas proposta nesta dissertação mostrou-se promissora para a redução do tempo gasto na preparação da máquina tubeteira utilizada pela empresa Sonoco de Londrina. Com a implementação das heurísticas Construtivas e de Refinamento obtivemos a melhor solução conhecida para cada carteira analisada, sendo que para a carteira 2 o resultado fornecido é superior ao obtido por Fenato (2008), uma vez que o autor subdividiu o problema, aceitando assim como solução um sub-ótimo do problema. Alcançamos reduções de custos de até 32, 1% para o custo segundo trocas de bolachas e 28, 4% para o caso em que são considerados os movimentos de bolachas, ainda estas reduções foram retornadas ambas pela heurística Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração, a qual retornou um melhor desempenho para todas as carteiras analisadas.

Realizamos ainda uma investigação para a utlização de Algoritmos Genéticos na resolução do problema, porém foram encontradas dificuldades para a restrição de que tubetes de mesmo mandril sejam fabricados consecutivamente ao considerarmos os operadores genéticos cruzamento e mutação encontrados na literatura. Nossa proposta para trabalhos futuros é o possível desenvolvimento de operadores que permitam garantir proles que mantenham as características necessárias para obter uma sequência viável para a empresa.

REFERÊNCIAS

[1] A. Behzad and M. Modares, A New Efficient Transformation of the Generalized Traveling Salesman Problem into Traveling Salesman Problem, in Proceedings of the 15th International Conference of Systems Engineering (ICSE) , Las Vegas, Nevada, August 6-8, 2002.

[2] Adranbinski, A. and Suslom M. M. (1983), Computational Experiments with Some Approximation Algorithms for the Traveling Salesman Problem, Zastos Mat 18, 91-95.

[3] A. L. Henry-Labordere. The record balancing problem: a dynamic programming solution of a generalized traveling salesman problem, Revue Francaise D Informatique DeRecherche Operationnelle 3 (NB2) 43-49, 1969.

[4] Balas, E. and Christofides, N. A. (1981), A Restricted Lagrangean Approach to the Traveling Salesman Problem, Math. Progr. 21, 19-46.

[5] Bäck, T., Fogel, D.B. & Michalewicz, Z. (eds.) "Evolutionary Computation 1: Basic Algorithms and Operators", Institute of Physics Publishing, 2000.

[6] Bellmore, M; Nemhauser, G. The Traveling Salesman Problem: A survey. Operations Research, v. 16, n. 3, p. 538 - 558, 1958.

[7] Blum, C.; Aguilera, M. J. B.; Roli, A.; Sampels, M. (Ed.). Hybrid metaheuristics: studies in computational intelligence. Berlin/Heidelberg: Springer, 2008. p. 1-30.

[8] Campello, C. R., Maculan, N. 1994. Algoritmos e Heurísticas. Editora da Universidade Federal Fluminense, Niterói, RJ.

[9] Christofides, N. (1976), Worst-case Analysis of a New Heuristic for the Traveling Salesman Problem, Technical Report, GSIA, Carnegie-Mellon University.

[10] Chaves, A. A.: Uma Metaheurística Híbrida com Busca por Grupamentos Aplicada a Problemas de Otimização Combinatória. Tese de Doutorado. Doutorado em Computação Aplicada. Instituto Nacional de Pesquisas Espaciais, INPE, Brasil. 2009.

[11] Croes, G. A. A method for solving traveling salesman problems. In: Operations Research n. 6, p. 791-812, 1958.

[12] Dantzig, G., Fulkerson, R., Johnson, S. 1954. Solution of a large-scale traveling-salesman problem. Journal of the operations research society of America, 2(4), 393-410.

[13] Davis, L., Applying Adaptive Algorithms to Epistatic Domains, Proceedings of the International Joint Conference on Artificial Intelligence, pp. 162 - 164, 1985.

[14] Dias, A. H. F. Algoritmos Genéticos Aplicados a Problemas com Múltiplos Objetivos. 136f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal de Minas Gerais - UFMG, Belo Horizonte, 2000.

[15] Fenato, A. J. Um Modelo de Caixeiro Viajante Generalizado para Minimizar o Tempo de Preparação de uma Máquina Tubeteira. 69f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Estadual de Londrina - UEL, Londrina, 2008.

[16] Feo, T., Resende, M. (1995) Greedy Randomized adaptative search procedures, Journal of Global Optimization Algorithms, vol. 6, pgs 109 - 133 - (Kluver Academic Publishers, Boston).

[17] Garey, M. R. and Johnson, D. S. Computers and Intractability: a guide to the theory of NP-Completeness. San Francisco, Freeman, 1979.

[18] G. Laporte, H. Mercure, Y. Nobert. Generalized traveling salesman problem through n sets of nodes: The asymmetric case, Discrete Applied Mathematics, 18, 185-197, 1987.

[19] G. Laporte, Y. Nobert (1983). Generalized traveling salesman problem through n sets of nodes: An integer programming approach, INFOR 21 (1), 61-75, 1983.

[20] Glover, F. (1986). Future paths for integer programming and links to artificial intelligence, Computers and Operations Research 5: 553-549.

[21] Goldbarg, M.C., Luna, H.P.L. Otimização Combinatória e Programação Linear. 3.ed. Rio de Janeiro: Editora Campus, 2000.

[22] Goldberg, D.E. Genetic Algorithms in Search, Optimization and Machine Learning. Addison-Wesley, 1989.

[23] Golden, B. L., Bodin, L., Doyle, T. and Stewart Jr., W. (1980), Approximate Travelling Salesman Algorithms, Opns. Res 28, 694-711.

[24] Gondran, M., Minoux, M. Graphs and Algorithms. Wiley-Interscience, 1984.

[25] Holland J.H., Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, Ann Arbor, Mich., 1975.

[26] Hoto, R. S. V.; Borssoi, A. H.; Maculan, Nelson; Fenato, Alexandre. Reducing the Setup of a Tubettes Machine. Revista IEEE América Latina vol.8, no.1, p. 101-106, 2010.

[27] H. Whitney, Analytic extensions of differentiable functions defined in closed sets, Transactions of the American Mathematical Society, vol. 36 (1934), pp. 63-89, Lemma 2.

[28] Laporte, G.; Martello, S. 1990. The Selective Traveling Salesman Problem. Discrete Appl. Math. 26. p. 193 – 207.

[29] Laporte G. and Norbert Y. (1973), Exact Algorithms for the Traveling Salesman Problem, Université du Québec.

[30] Lin, S. Computer solutions of the traveling salesman problem. In: Bell System. Technical. Journal. n.. 44, p. 2245-2269, 1965.

[31] Lin, S., Kernighan, B.W., 1973. An effective heuristic algorithm for the traveling-salesman problem. Operations research, 21(2), 498-516.

[32] Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution Programs. 3rd edition. Springer-Verlag. Berlin, Germany (1996).

[33] Nemhauser, G.L.; Wolsey L. Integer and Combinatorial Optimization. John Wiley & Sons, 1988.

[34] Nilsson, C. Heuristic Algorithms For Travelling Salesman Problem. Linköping University. Último acesso em 19/08/2010: http://www.ida.liu.se/ TDDB19/reports_2003/htsp.pdf.

[35] Noon. C.E. The Generalized Traveling Salesman Problem. Unpublished Dissertation, Department of Industrial and Operations Research, University of Tennessee, 1988.

[36] Noon, C. E., Bean; J. C. A Lagrangian Based Approach for the Asymmetric Generalized Traveling Salesman Problem. Operations Research n.39, p. 623-632, 1991.

[37] Noronha, T. F., DaSilva, M. M., Aloise, D. J. (2000) Uma Abordagem sobre Estratégias Metaheurísticas - Relatório Técnico, UFRN - UnP.

[38] Pinheiro, F. A. Aplicação de um Algoritmo Genético no estudo das Perdas e do Controle de Tensão em Sistemas Elétricos de Potência. 78f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal de Minas Gerais - UFMG, Belo Horizonte, 1998.

[39] Reeves, C. R. Modern Heuristic Techniques for Combinatorial Problems, Blackwell Scientific Publication, 1993.

[40] Rosenkrantz, R.; Sterns, R; Lewis, P. 1977. An Analysis of Several Heuristics for the Traveling Salesman Problem. SIAM j. Com. 6. p. 563 – 581.

[41] S. S. Srivastava, S. Kumar, R.C. Garg, P. Sen. Generalized traveling salesman problem through n sets of nodes, CORS J. 7, 1969.

[42] Soares, G. L., Algoritmos Genéticos: Estudo, Novas Técnicas e Aplicações. 137f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal de Minas Gerais - UFMG, Belo Horizonte, 1997.

[43] Vasconcelos, J. A - "Optimisation de Forme des Structures Électromagnétiques", Tese de Doutorado, Ecole Centrale de Lyon, Junho 1994, França.

[44] V. Dimitrijevic and Z. Saric. Efficient Transformation of the Generalized Traveling Salesman Problem into the Traveling Salesman Problem on Digraphs, Information Sci., 102, 65-110, 1997.

[45] Y. Lien, E. Ma and B. W. Wah. Transformation of the Generalized Traveling Salesman Problem into the Standard Traveling Salesman Problem, Information Sci. 74, 177-189, 1993.

[46] http://www.mat.ufrgs.br/ portosil/caixeiro.html

APÊNDICES

APÊNDICE A – Artigo submetido à Revista TEMA

Um Protótipo Computacional para o Problema da Tubeteira

RENATA MASCARI, Programa de Pós-Graduação em Matemática Aplicada e Computacional, UEL - Universidade Estadual de Londrina, 86051-980 Londrina, PR, Brasil.

Resumo. Neste trabalho abordamos métodos heurísticos para otimizar o tempo produtivo de uma tubeteira (máquina que confecciona tubetes). Tubetes são tubos feitos pela colagem de fitas de papel, as quais são depositadas em rolos que recebem o nome de bolachas, sendo que algumas delas podem ser aproveitadas entre a confecção de dois tubetes. Apresentamos um modelo matemático para minimizar a quantidade de movimentos de bolachas, bem como as implementações das heurísticas Vizinho mais Próximo, Melhor Vizinho mais Próximo, Adaptações das heurísticas 2-Opt e 3-Opt e ainda uma heurística de permutação denominada Melhor Configuração em linguagem C^{++} utilizando o IDE (ambiente de desenvolvimento integrado) *WxDev* - C^{++}. Os resultados obtidos pelas simulações apresentaram melhoria em relação aos obtidos por uma indústria do segmento.

Palavras-chave. Otimização, Problema do caixeiro viajante generalizado, Heurística.

A.1 Introdução

Nas últimas décadas, estudos vêm sendo feitos a fim de desenvolver técnicas que encontrem soluções satisfatórias para as mais diversas empresas, as quais já se conscientizaram da importância de efetuarem um planejamento inteligente para sua produção. Para resolver tal problema as indústrias buscam meios de otimizar seus processos com o auxílio de modelos matemáticos e ferramentas computacionais, buscando o maior retorno possível em suas atividades, seja ele qualidade no atendimento, aumento na produtividade, redução de custos e consequentemente aumento de lucros. Nestas situações, o problema consiste nas tomadas de decisão, ou seja, qual a ordem de produção, qual produto deve ser fabricado e em qual quantidade, qual a relação de gasto e benefício para cada produto. Assim, o planejamento da produção influencia significativamente na qualidade do serviço e no custo necessário para todo o processo. Neste artigo, iremos expor o estudo de um caso em que buscamos otimizar o tempo de preparação de uma máquina

denominada Tubeteira.

A Tubeteira é uma máquina que fabrica tubos rígidos (tubetes) pela sobreposição ordenada de várias camadas de papel reciclado (acondicionados em fitas denominadas bolachas, ver figura 1), coladas umas sobre as outras com tensões tecnicamente determinadas. A espessura e rigidez dos tubetes dependem, principalmente, da quantidade de camadas e do tipo de papel utilizado em sua fabricação.

Tubetes (Figura 2) são utilizados como embalagens de proteção e conservação para produtos de outras empresas (tubolatas). São empregados também como suporte para serem enrolados filmes plásticos, fitas adesivas, etc. Os tubetes diferenciam-se pela resistência à pressão (rigidez), espessura da parede, diâmetro interno e comprimento, os quais são determinados pela necessidade do cliente.

Figura A.1: Bolachas

Figura A.2: Tubete

Com o objetivo de diminuir o tempo de preparação da máquina tubeteira, Fenato (2008) e Hoto *et al.* (2010) modelaram o problema por meio de um Caixeiro Viajante Generalizado, visando a minimização de trocas de bolachas entre a confecção de dois tubetes arbitrários. Os autores também efetuaram simulações no Xpress-MP (Dash Optimization, 2000). Neste artigo, propomos, um aprimoramento da modelagem de Fenato (2008) e Hoto *et al.* (2010) e a utilização das heurísticas Melhor Vizinho mais Próximo, Bipariciomento (adaptação 2-Opt), Triparticionamento (adaptação 3-Opt) e Melhor Configuração, sendo esta última desenvolvida para o Problema da Tubeteira.

A.2 Descrição do Problema

Fenato (2008) e Hoto *et al.* (2010) descrevem uma máquina tubeteira e o funcionamento da mesma durante o processo de fabricação dos tubetes.

Primeiramente, as bolachas são alocadas no estaleiro segundo a composição do tubete a ser fabricado, obedecendo rigorosamente a ordem previamente estabelecida, a fim de obter as características técnicas do tubete. No estaleiro podem ser acondicionadas no máximo 34 bolachas, de onde serão desenroladas até o tanque de cola. O tanque de cola, por sua vez, é formado por um reservatório na parte inferior, onde se deposita a cola que é elevada e derramada sobre as fitas de papel que passam por entre palhetas que retiram o excesso de cola e direcionam a fita. Já a sobrepositora recebe as fitas de papel que passam pelo tanque de cola e as sobrepõem, uma a uma, com uma tensão adequada, formando um tubo de comprimento "infinito", do qual serão cortados os tubetes numa serra circular acoplada à sobrepositora. Existe ainda o diâmetro do mandril, ou seja, o diâmetro interno do tubete. A troca de um mandril despende um tempo muito superior em relação à movimentação de uma bolacha, sendo assim os tubetes que possuem o mesmo mandril devem ser fabricados em sequência.

O tempo de preparação da máquina tubeteira (posicionamento de bolachas no estaleiro) representa 40% do tempo total de produção dos tubetes (Fenato, 2008).

Considera-se ainda a possibilidade de uma posição vazia no estaleiro, de modo que um mesmo tubete terá mais de uma configuração possível para a sua fabricação. Baseados nisso, Fenato (2008) e Hoto *et al.* (2010) definiram um grafo valorado G formado por clusters, onde cada um representa um diferente tipo de tubete, e seus vértices representam as diferentes configurações possíveis das bolachas utilizadas em suas configurações.

Neste artigo acrescentamos à função objetivo uma variável que analisa a troca de mandris distintos e utilizamos uma nova abordagem para o cálculo de custos entre a fabricação de dois tubetes quaisquer, a fim de apresentar um modelo que descreva e solucione as reais necessidades da empresa.

A.3 Abordagem por meio de um Problema do Caixeiro Viajante Generalizado

Exposto o problema e a ideia de resolução, o critério para tomada de decisão consiste na obtenção de um ciclo hamiltoniano de custo mínimo e para isto definimos um tubete (vértice) fictício O que representa o vértice inicial e final da sequência de tubetes a serem fabricados.

Em Fenato (2008) e Hoto *et al.* (2010) o custo c_{ij} é considerado como o número de trocas de bolachas necessárias para a fabricação entre os tubetes i e j, e assim inicialmente trabalhamos com esta proposta de custos, porém esta abordagem não representa a realidade da empresa. Buscamos então determinar o custo de modo a atender as necessidades reais da empresa, considerando então o número de movimentos que são realizados para uma troca de bolacha.

Assim, o custo c_{ij}, o qual será associado à aresta (i, j), que representa o número de movimentos de bolachas entre as confecções dos tubetes i e j, é analisado segundo as possíveis posições formadas pelas carteiras. Como mencionado, o estaleiro comporta 34 bolachas ao máximo, sendo estas divididas em fileiras com 4 posições, as quais podem ser alocadas como na figura 4:

Note que, as bolachas podem então ser denominadas como bolachas internas e externas, e ainda a movimentação de uma bolacha interna despende um tempo maior em relação a remoçao e substituição de uma bolacha externa. A figura 4, apresenta as possíveis distribuições das bolachas em cada fileira. Por exemplo, partindo da situação C, para a substituição da bolacha interna 3 seriam necessários 4 movimentos, enquanto Fenato (2008) relaciona a esta substituição apenas 1 troca, ou seja, não representa o verdadeiro tempo gasto realizado pela empresa. Consideramos os custos da forma $c_{k0} = c_{0k} = 0$, para qualquer tubete k.

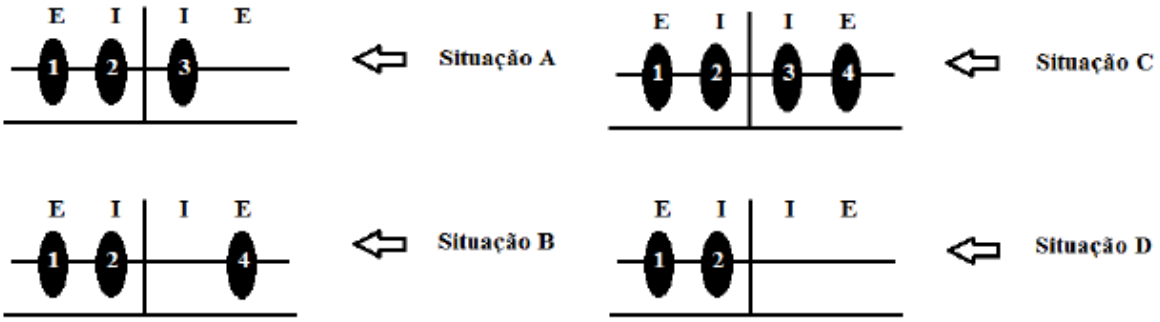


Figura A.3: Possíveis alocações de bolachas

Apresentamos a seguir, o modelo reformulado e considerando como custo os movimentos de bolachas.

Para a modelagem serão considerados n tubetes e a variável de decisão binária x_{ij}, onde $i, j = 1, 2, ..., n, i \neq j$ representam os tubetes, tal que:

$$x_{ij} = \begin{cases} 1, & \text{se o tubete } j \text{ é fabricado imediatamente após o tubete} i \\ 0, & \text{caso contrário.} \end{cases}$$

Considere ainda o custo c_{ij}, o qual será associado à aresta (i, j), que representa o número de movimentos de bolachas entre as confecções dos tubetes i e j.

Definimos a variável m_{ij}, que está associada ao custo fictício referente à troca de mandril:

$$m_{ij} = \begin{cases} 2B_{max} + 3, & \text{se } M_i \neq M_j \\ 0, & \text{caso contrário.} \end{cases}$$

onde B_{max} é o número máximo de bolachas necessárias para confeccionar um tubete

Desse modo, define-se a função objetivo auxiliar como:

$$z' = \sum_{(i,j) \in E} x_{ij}(c_{ij} + m_{ij})$$

O modelo pode ser apresentado como:

$$\min z' = \sum_{j=1}^{n} \sum_{i=1}^{n} x_{ij}(c_{ij} + m_{ij})$$

sujeito a:

$$\sum_{i=1}^{n} x_{ij} = 1, \text{ para todo } j, j = 1, 2, ..., n$$

$$\sum_{j=1}^{n} x_{ij} = 1, \text{ para todo } i = 1, 2, ..., n$$

$$\sum_{i \in S} \sum_{j \in S} x_{ij} \leq |S| - 1, \text{ para todo } S \subset \{1, 2, ..., n\}$$

$$0 \leq c_{ij} \leq 2B_{max} + 2,$$

onde B_{max} é o número máximo de bolachas necessárias para confeccionar um tubete.

$$m_{ij} = \begin{cases} 2B_{max} + 3, & \text{se } M_i \neq M_j \\ 0, & \text{caso contrário.} \end{cases}$$

$$x_{ij} \in \{0, 1\}, \text{ para todo } i, j \in \{1, 2, ..., n\}, i \neq j$$

Na função objetivo auxiliar, os custos fictícios relacionados às trocas de mandril são somados juntamente com os custos relacionados às movimentações de bolachas, de modo que os tubetes com o mesmo mandril possam ser produzidos em sequência. Para obtermos apenas o custo total de movimentos de bolachas necessários para a produção entre dois tubetes i e j é suficiente eliminar-se os custos fictícios de trocas de mandril que foram realizadas durante a produção, ou seja, a função objetivo se reduz a:

$$z = z' - \sum_{(i,j)\in E} x_{ij} m_{ij} = z' - (M-1)(2B_{max}+3),$$

uma vez que o número de mandris diferentes é M, teremos $(M-1)$ trocas de mandril e o custo relacionado a cada troca de mandril é $(2B_{max}+3)$.

Modelado o problema da Tubeteira segundo um E-PCVG considerando movimentos de bolachas, apresentamos as implementações de heurísticas que minimizam movimentos de bolachas entre a fabricação dos tubetes.

A.4 Resolução do Problema por meio de Heurísticas

Para buscarmos uma solução viável, sem o uso de um *solver*, foram utilizadas a heurística construtiva Vizinho mais Próximo e as heurísticas de refinamento Melhor Vizinho mais Próximo, Biparticionamento, Triparticionamento e ainda a heurística Melhor Configuração, que permuta as configurações dos tubetes após ser encontrada uma solução viável, tais heurísicas foram combinadas, de modo a obtermos as heurísticas Melhor Vizinho mais Próximo Biparticionado, Melhor Vizinho mais Próximo Triparticionado, Melhor Vizinho mais Próximo com a Melhor Configuração, Melhor Vizinho mais Próximo Biparticionado com a Melhor Configuração e Melhor Vizinho mais Próximo Triparticionado com a Melhor Configuração.

As heurísticas foram implementadas em linguagem C^{++}, considerando as posições vazias nas configurações dos tubetes. Desenvolvemos ainda uma interface gráfica (Figura 5), por meio do ambiente de desenvolvimento integrado (IDE) *WxDev-C*$^{++}$, que permite ao usuário a visualização das opções disponíveis para as carteiras a serem analisadas.

A heurística do Vizinho mais Próximo (VMP) é uma heurística construtiva gulosa, apresentada originalmente em Bellmore & Nemhauser (1968), e tem como base uma ideia simples que consiste em escolher aleatoriamente um vértice inicial e então selecionar o vértice mais próximo que ainda não foi visitado, formando assim a aresta que fará parte da solução. O processo se encerra quando todos os vértices forem visitados.

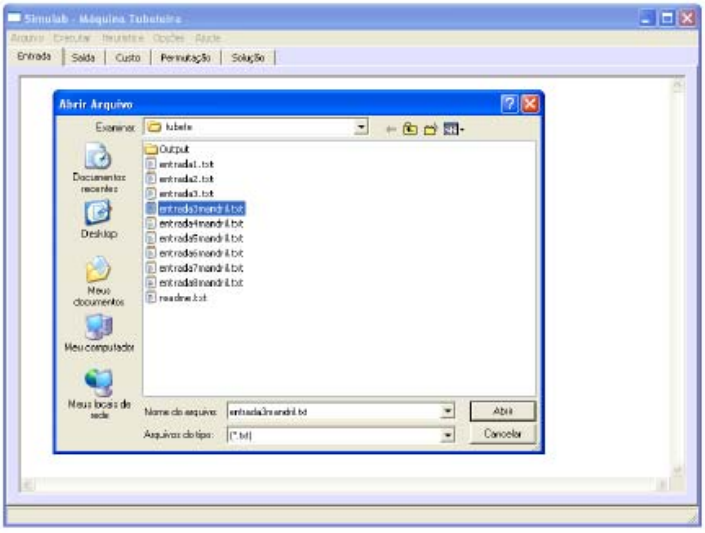


Figura A.4: Interface desenvolvida para o Problema da Tubeteira.

---

algoritmo VMP (sol, cus, m, n, p, c)
  escolha o tubete (p, c) como vértice inicial da solução
  soma $\leftarrow 0$
  para $i \leftarrow 1$ até $m - 1$ **faça**
    acrescente à solução o vértice de um cluster não visitado mais próximo
    do vértice do cluster anterior
    acrescente à soma o custo da aresta percorrida
  retorne soma
fim-algoritmo

A heurística de refinamento Melhor Vizinho mais Próximo (MVMP) consiste em analisar todas as rotas possíveis construídas pela heurística do Vizinho mais Próximo, isto é, considerando cada tubete como inicial a partir do tubete fictício O, e dentre elas escolher a solução de menor custo, ou ainda, a solução que retorne o menor valor da função objetivo.

---

algoritmo MVMP (sol, cus, m, n)
  soma $\leftarrow \infty$
  para $i \leftarrow 1$ até m **faça**
    para $j \leftarrow 1$ até n **faça**
    soma* $\leftarrow$ HVMP(sol*, cus, m, n, i, j)
    se soma* $<$ soma **então faça**
      soma $\leftarrow$ soma*
      sol $\leftarrow$ sol*
  retorne soma

fim-algoritmo

A heurística Biparticionamento (2-Part) é uma adaptação da heurística clássica 2-Opt e consiste em eliminar uma aresta de uma solução (rota) viável e religar dois vértices extremos da solução por meio de uma nova aresta, sem que sejam formadas subrotas. O procedimento é realizado em todas as posições que permitam o particionamento da solução. A melhor solução (rota) encontrada durante a realização do procedimento é atualizada ao final do algoritmo.

---

algoritmo Biparticionamento (sol, soma, n)
  soma** $\leftarrow$ soma
  para $i \leftarrow 1$ até $n - 1$ **faça**
    para $j \leftarrow i + 1$ até $n - 1$ **faça**
    particione a solução removendo a aresta entre os vértices i e $i + 1$
    se soma* $<$ soma **então faça**
      soma $\leftarrow$ soma*
      sol** $\leftarrow$ sol*
  soma $\leftarrow$ soma**
  sol $\leftarrow$ sol**
  retorne soma
fim-algoritmo

O Triparticionamento (3-Part) é baseado na heurística clássica 3-Opt. A heurística Triparticionamento remove duas arestas de uma solução viável e religa os vértices extremos das partições formadas através de duas novas arestas, sem permitir a formação de subrotas.

---

algoritmo Triparticionamento (sol, soma, n)
  soma** $\leftarrow$ soma
  para $i \leftarrow 1$ até $n-2$ **faça**
    para $j \leftarrow i+1$ até $n-1$ **faça**
    particione a solução removendo a aresta entre os vértices i e $i+1$ e a aresta entre os vértices j e $j+1$
    sol* $\leftarrow$ melhor solução dentre as possíveis reconexões das partições criadas
    se soma* $<$ soma **então faça**
      soma $\leftarrow$ soma*
      sol** $\leftarrow$ sol*
  soma $\leftarrow$ soma**
  sol $\leftarrow$ sol**
  retorne soma
fim-algoritmo

A fim de abranger as características intrínsecas do problema em questão desenvolvemos a heurística denominada como Melhor Configuração (MC). Tal heurística se limita a analisar a necessidade de que sejam realizadas trocas nas configurações dos tubetes sem que a sequência obtida seja alterada.

---

algoritmo Melhor Configuração (sol, soma, m, n)
  soma $\leftarrow \infty$
  para $i \leftarrow 1$ até m **faça**
    para $j \leftarrow 1$ até n **faça**
    crie sol* substituindo o nó do cluster i da solução por outro nó do mesmo cluster mas com configuração j
    se soma* $<$ soma **então faça**
      soma $\leftarrow$ soma*
      sol $\leftarrow$ sol*
  retorne soma
fim-algoritmo

Uma vez implementadas as heurísticas, realizamos testes numéricos em carteiras fornecidas pela empresa.

A.5 Resultados Computacionais

Para realizarmos os testes numéricos tivemos acesso a sete carteiras cedidas pela empresa, verificando assim a eficiência das heurísticas apresentadas.

As carteiras são constituídas por diferentes quantidades de tubetes, os quais também possuem configurações distintas para sua fabricação. Na tabela a seguir, as principais características das carteiras, bem como o número de vértices que obtemos ao permitir uma posição vazia no estaleiro:

Tabela A.1: Características da Carteiras

Carteira	Tubetes	Número Máximo de Bolachas	Mandris Distintos	Quantidade de Vértices
1	10	10	1	79
2	16	26	9	246
3	10	31	8	205
4	8	25	5	132
5	7	23	5	122
6	6	27	5	154
7	6	23	5	104

Para o caso em que o custo está relacionado com a movimentação de bolachas, não é possível realizar uma comparação entre a solução obtida e a melhor solução (modelagem exata), pois não foi esta formulação realizado por Fenato (2008) e Hoto *et al.* (2010). Porém, obtivemos reduções de custos em todas as carteiras, fornecendo à empresa um programa que condiz e representa seu dia-a-dia.

Tabela A.2: Movimentos necessários para a fabricação das carteiras.

Carteira	MVMP	MVMP/ 2PART	MVMP/ 3PART	MVMP/ MC	MVMP/ 2PART/ MC	MVMP/ 3PART/ MC
1	89	88	86	84	84	83
2	315	302	300	292	292	290
3	234	232	232	234	232	231
4	151	146	146	151	146	146
5	104	104	104	104	103	104
6	124	124	124	124	124	124
7	164	158	154	160	154	154

Para fins de comparação, desenvolvemos no programa a possibilidade de fornecer uma sequência para que o mesmo retornasse a quantidade de movimentos necessários para sua execução. Assim, a tabela 4 apresenta as respostas obtidas pelas heurísticas e pelas simulações realizadas pelo programa desenvolvido por Fenato (2008) e Hoto *et al.* (2010) e ainda o percentual de redução de custos obtidos pela abordagem descrita neste trabalho:

Tabela A.3: Percentual de Redução utilizando heurísticas.

Carteira	Melhor Solução-Heurística	Xpress-MP	Empresa	Percentual de Redução (Heurística)	Percentual de Redução (Xpress-MP)
1	83	89	116	28, 4%	23, 3%
2	290	310	370	21, 6%	16, 2%
3	231	246	313	26, 2%	21, 4%
4	146	149	170	14, 1%	12, 4%
5	103	112	122	14, 8%	8, 2%
6	124	134	166	25, 3%	19, 3%
7	154	160	178	13, 5%	10, 1%

A.6 Considerações Finais

O programa desenvolvido executou de maneira satisfatória as carteiras que a empresa nos forneceu como exemplares. Em relação ao tempo de execução, as simulações foram realizadas imediatamente sem necessidade de qualquer divisão de carteira.

Outro diferencial deste programa está na possibilidade de que sejam testadas carteiras diversas, sem a necessidade de novas implementações ou mesmo alterações no código, uma vez que para as implementações que Fenato (2008) realizou, a cada nova carteira é necessária a implementação de um novo programa contendo as particularidades de cada carteira.

Para o desenvolvimento do programa, como citado acima, utilizamos um IDE gratuito, o que permite a execução do mesmo em qualquer ambiente, não apenas em laboratórios ou ainda somente com a compra e licença de *softwares*, que é o caso do programa apresentado em Fenato (2008).

As heurísticas implementadas forneceram melhoras significativas nas sequências de fabricação dos tubetes ao compararmos com a solução usada pela empresa e mesmo com as soluções obtidas pelo solver Xpress-MP. O desenvolvimento de uma interface para o programa possibilita a sua utilização pela empresa, atingindo assim o objetivo de fornecer uma solução real e concreta.

Abstract. In this work we approach heuristic methods to optimize the time of preparation of a tubettes machine. Tubettes are tubes made by gluing strips of paper that are packed in paper reels, and some of them may be used between one and making another tubettes. We presents a mathematical model for the minimization of moviments reels and also implementations for the heuristics Nearest Neighbor, an improvement of a nearest neighbor, an adaptation of the heuristics 2-Opt and 3-Opt and an heuristic of permutation called Best Configuration using the IDE (integrated development environment) *WxDev - C++*. The results obtained by simulations presented improvement over those used by the company.

Bibliografia

[1] Bellmore, M; Nemhauser, G. The Traveling Salesman Problem: A survey. Operations Research, v. 16, n. 3, p. 538 - 558, 1958.

[2] Campello, C. R., Maculan, N. 1994. Algoritmos e Heurísticas. Editora da Universidade Federal Fluminense, Niterói, RJ.

[3] DASH OPTIMIZATION, Applications Of Optimization With XpressMP, Tradução para o inglês de Programmation Linéaire de C. Guéret, C. Prins E M. Sevaux, Dash Optimization Ltda, 2000.

[4] FENATO, A. J. Um Modelo de Caixeiro Viajante Generalizado para Minimizar o Tempo de Preparação de uma Máquina Tubeteira. 69f. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Estadual de Londrina - UEL, Londrina, 2008.

[5] GOLDBARG, M.C.; LUNA, H.P.L. Otimização Combinatória e Programação Linear. 3.ed. Rio de Janeiro: Editora Campus, 2000.

[6] GONDRAN, M.; MINOUX, M. Graphs and Algorithms. Wiley-Interscience, 1984.

[7] HOTO, R. S. V.; BORSSOI, A. H.; MACULAN, Nelson; FENATO, Alexandre. Reducing the Setup of a Tubettes Machine. Revista IEEE América Latina vol.8, no.1, p. 101-106, 2010.

[8] NEMHAUSER, G.L.; WOLSEY L. Integer and Combinatorial Optimization. John Wiley & Sons, 1988.

[9] NILSSON, C. Heuristic Algorithms For Travelling Salesman Problem. Linköping University. Último acesso em 19/08/2010: http://www.ida.liu.se/ TDDB19/reports-2003/htsp.pdf.

[10] NOON, C. E., BEAN; J. C. A Lagrangian Based Approach for the Asymmetric Generalized Traveling Salesman Problem. Operations Research n.39, p. 623-632, 1991.

APÊNDICE B – Conceitos básicos sobre Grafos

As notações e definições apresentadas neste apêndice foram baseadas em [Gondran e Minoux].

Um grafo $G = (V, E)$ é formado por um conjunto de vértices V e um conjunto de arestas E. Tais grafos podem ser classificados como completos ou não-completos, direcionados ou não-direcionados, conexo ou desconexo, fechado, com ciclos ou sem ciclos. Ao decorrer deste apêndice definiremos cada classificação, bem como abordaremos definições de ordem de um grafo, árvores e florestas.

B.1 Grafos Direcionados

Definição 1. *Um grafo $G = (V, E)$ é definido como um conjunto V cujos elementos são chamados vértices (ou nós) e um conjunto E cujos elementos $e \in E$ são pares ordenados de vértices, chamados arcos.*

A ordem de um grafo depende do número de vértices contidos no mesmo. Assumindo-se que os vértices são numerados da form $i = 1, ..., N$. Podemos definir a ordem de um grafo:

Definição 2. *Se $|V| = N$ é o número de vértices, então o grafo G é dito ser de ordem N.*

Sendo $e = (i, j)$ um arco de G, então i é o ponto extremo incial e j o ponto extremo final. Um arco com os pontos extremos coincidentes é chamado um ciclo.

Definição 3. *Um p - grafo é um grafo com não mais que p arcos (i, j) entre quaisquer dois vértices i e j, nesta ordem. Em particular, um 1 - grafo é um grafo com não mais que um arco (i, j) para todos i, j.*

B.2 Grafos não-direcionados

No estudo de grafos não direcionados, a cada arco, ou seja, a cada par ordenado de vértices (i, j), associamos um par não-ordenado (i, j), chamado aresta (i, j).

Definição 4. *Uma aresta é um arco, no qual a direção é irrelevante.*

Para o estudo das propriedades de um grafo não-direcionado $G = (V, E)$, será suficiente considerarmos o conjunto E como um conjunto de arestas.

Observação 4. *O conjunto E sempre será considerado, implicitamente, como sendo um conjunto de arcos. Quando o mesmo for um conjunto de arestas, mencionaremos explicitamente. Diremos então que o grafo é não-direcionado.*

Definição 5. *Um grafo é chamado simples, quando não possui ciclos e não há mais que uma aresta entre dois vértices quaisquer.*

B.3 Definições principais

Definição 6. *Dois arcos, ou arestas, são chamados adjacentes se possuem pelo menos um ponto extremo em comum.*

Definição 7. *O grau do vértice i, denotado por $d_G(i)$ é o número de arcos (ou arestas) que tem i como um ponto extremo.*

Definição 8. *Um grafo é chamado simétrico se, para todo par de vérties (i, j) existe uma aresta da (i, j) bem como da forma (j, i).*

Definição 9. *Um 1 - grafo é chamado antissimétrico quando $(i, j) \in E$ implica que $(j, i) \notin E$.*

Definição 10. *Um grafo $G = (V, E)$ é chamado completo se, para todo par de vértices (i, j) existe uma aresta da forma (i, j).*

Observação 5. *Um 1 - grafo é completo se, e somente se, $(i, j) \in E$ implica que $(j, i) \notin E$.*

Definição 11. *Dado $A \subset V$, o subgrafo gerado por A é o grafo $G_A = (A, E_A)$ cujos vértices são os elementos de A e cujos arcos em U_A são os arcos de G que tem seus dois pontos extremos em A.*

B.4 Vizinhança, passeio, trajeto e circuito de um grafo

Uma vizinhança de um vértice qualquer v é formada por todos os vértices adjacentes a ele e denota-se este conjunto de vértices vizinhos por $N_G(v)$.

Definição 12. *Define-se um passeio em um grafo, entre dois vértices x e y quaisquer, como sendo toda a sequência de vértices e arestas no formato da equação $(A.1)$ com eventual repetição de vértices e arestas.*

$$x = v_1, v_1v_2, v_2, ..., v_{k-1}v_k, v_k = y \tag{B.1}$$

Para o caso representado pela equação $(A.1)$ tem-se x e y como vértices extremos do passeio sendo x o vértice inicial e y o vértice final.

Definição 13. *Um trajeto, entre dois vértices quaisquer x e y, feito em um grafo é um passeio entre x e y sem arestas repetidas, podendo haver vértices repetidos.*

Definição 14. *Um caminho entre dois vértices é definido como sendo um trajeto sem vértices repetidos.*

Um caso específico e muito usado na teoria de grafos é o circuito que é representado por um trajeto fechado, onde o vértice inicial coincide com o vértice final.

B.5 Árvores e Florestas

Definição 15. *Uma árvore é um grafo conexo sem ciclos.*

Definição 16. *Um grafo sem ciclos que não é conexo é chamado uma floresta (cada componente conexa é uma árvore).*

Segue da definição que uma árvore é sempre um grafo simples.

Definição 17. *Uma árvore geradora do grafo G, é uma árvore que visita todos os vértices de G.*

B.6 Comprimento em Grafos

Definição 18. *Dado um caminho P (ciclo C) de um grafo G qualquer, designa-se por comprimento de $P(C)$ o número de arestas que o constitui.*

Neste contexto, uma aresta é um caminho de comprimento 1 e um vértice é um caminho de comprimento nulo.

B.7 Ciclos Hamiltonianos

Seja $G = (V, E)$ um grafo conexo de ordem $|V| = N$.

Definição 19. *Um caminho passando uma e somente uma vez através de todo vértice de G é chamado um caminho Hamiltoniano.*

Um caminho Hamiltoniano é portanto elementar e tem comprimento $N - 1$.

Definição 20. *Um ciclo (circuito) Hamiltoniano é um ciclo (circuito) que passa uma e somente uma vez através de todo vértice de G.*

Desse modo, um ciclo Hamiltoniano é elementar, com comprimento N. Ainda, dizemos que um grafo é Hamiltoniano se contém um ciclo Hamiltoniano (no caso não-direcionado) ou um circuito Hamiltoniano (no caso direcionado).

APÊNDICE C – Configurações das Carteiras

Neste apêndice serão apresentadas as configurações das carteiras 1, 2, 3, 4, 5, 6 e 7 considerando suas configurações originais e suas configurações considerando posições vazias.

Tubete	Configuração das Bolachas
1	(10, 20, 40, 60, 80, 80, 90, 90, 90, 100)
2	(10, 20, 60, 60, 90, 90, 100)
3	(10, 30, 30, 30, 40, 40, 60, 90, 90, 100)
4	(10, 20, 40, 80, 80, 90, 90, 100)
5	(10, 70, 70, 80, 80, 90, 100)
6	(10, 70, 70, 70, 80, 80, 80, 90, 100)
7	(10, 20, 20, 60, 60, 70, 70, 70, 90, 100)
8	(10, 90, 90, 90, 100)
9	(10, 80, 80, 90, 90, 100)
10	(10, 20, 30, 40, 60, 90, 100)

Tabela C.1: Configurações da Carteira 1 sem considerar posições vazias

Tubete - Configuração	Configuração das Bolachas
1 - 1	$(10, 20, 40, 60, 80, 80, 90, 90, 90, 100, \emptyset)$
1 - 2	$(10, \emptyset, 20, 40, 60, 80, 80, 90, 90, 90, 100)$
1 - 3	$(10, 20, \emptyset, 40, 60, 80, 80, 90, 90, 90, 100)$
1 - 4	$(10, 20, 40, \emptyset, 60, 80, 80, 90, 90, 90, 100)$
1 - 5	$(10, 20, 40, 60, \emptyset, 80, 80, 90, 90, 90, 100)$
1 - 6	$(10, 20, 40, 60, 80, \emptyset, 80, 90, 90, 90, 100)$
1 - 7	$(10, 20, 40, 60, 80, 80, \emptyset, 90, 90, 90, 100)$
1 - 8	$(10, 20, 40, 60, 80, 80, 90, \emptyset, 90, 90, 100)$
1 - 9	$(10, 20, 40, 60, 80, 80, 90, 90, \emptyset, 90, 100)$
1 - 10	$(10, 20, 40, 60, 80, 80, 90, 90, 90, \emptyset, 100)$
2 - 1	$(10, 20, 60, 60, 90, 90, 100, \emptyset)$
2 - 2	$(10, \emptyset, 20, 60, 60, 90, 90, 100)$
2 - 3	$(10, 20, \emptyset, 60, 60, 90, 90, 100)$
2 - 4	$(10, 20, 60, \emptyset, 60, 90, 90, 100)$
2 - 5	$(10, 20, 60, 60, \emptyset, 90, 90, 100)$
2 - 6	$(10, 20, 60, 60, 90, \emptyset, 90, 100)$
2 - 7	$(10, 20, 60, 60, 90, 90, \emptyset, 100)$
3 - 1	$(10, 30, 30, 30, 40, 40, 60, 90, 90, 100, \emptyset)$
3 - 2	$(10, \emptyset, 30, 30, 30, 40, 40, 60, 90, 90, 100)$
3 - 3	$(10, 30, \emptyset, 30, 30, 40, 40, 60, 90, 90, 100)$
3 - 4	$(10, 30, 30, \emptyset, 30, 40, 40, 60, 90, 90, 100)$
3 - 5	$(10, 30, 30, 30, \emptyset, 40, 40, 60, 90, 90, 100)$
3 - 6	$(10, 30, 30, 30, 40, \emptyset, 40, 60, 90, 90, 100)$
3 - 7	$(10, 30, 30, 30, 40, 40, \emptyset, 60, 90, 90, 100)$
3 - 8	$(10, 30, 30, 30, 40, 40, 60, \emptyset, 90, 90, 100)$
3 - 9	$(10, 30, 30, 30, 40, 40, 60, 90, \emptyset, 90, 100)$
3 - 10	$(10, 30, 30, 30, 40, 40, 60, 90, 90, \emptyset, 100)$
4 - 1	$(10, 20, 40, 80, 80, 90, 90, 100, \emptyset)$
4 - 2	$(10, \emptyset, 20, 40, 80, 80, 90, 90, 100)$
4 - 3	$(10, 20, \emptyset, 40, 80, 80, 90, 90, 100)$
4 - 4	$(10, 20, 40, \emptyset, 80, 80, 90, 90, 100)$
4 - 5	$(10, 20, 40, 80, \emptyset, 80, 90, 90, 100)$
4 - 6	$(10, 20, 40, 80, 80, \emptyset, 90, 90, 100)$
4 - 7	$(10, 20, 40, 80, 80, 90, \emptyset, 90, 100)$
4 - 8	$(10, 20, 40, 80, 80, 90, 90, \emptyset, 100)$
5 - 1	$(10, 70, 70, 80, 80, 90, 100, \emptyset)$
5 - 2	$(10, \emptyset, 70, 70, 80, 80, 90, 100)$
5 - 3	$(10, 70, \emptyset, 70, 80, 80, 90, 100)$
5 - 4	$(10, 70, 70, \emptyset, 80, 80, 90, 100)$
5 - 5	$(10, 70, 70, 80, \emptyset, 80, 90, 100)$

Tabela C.2: Configurações da Carteira 1 considerando posições vazias

Tubete - Configuração	Configuração das Bolachas
5 - 6	(10, 70, 70, 80, 80, ∅, 90, 100)
5 - 7	(10, 70, 70, 80, 80, 90, ∅, 100)
6 - 1	(10, 70, 70, 70, 80, 80, 80, 90, 100, ∅)
6 - 2	(10, ∅, 70, 70, 70, 80, 80, 80, 90, 100)
6 - 3	(10, 70, ∅, 70, 70, 80, 80, 80, 90, 100)
6 - 4	(10, 70, 70, ∅, 70, 80, 80, 80, 90, 100)
6 - 5	(10, 70, 70, 70, ∅, 80, 80, 80, 90, 100)
6 - 6	(10, 70, 70, 70, 80, ∅, 80, 80, 90, 100)
6 - 7	(10, 70, 70, 70, 80, 80, ∅, 80, 90, 100)
6 - 8	(10, 70, 70, 70, 80, 80, 80, ∅, 90, 100)
6 - 9	(10, 70, 70, 70, 80, 80, 80, 90, ∅, 100)
7 - 1	(10, 20, 20, 60, 60, 70, 70, 70, 90, 100, ∅)
7 - 2	(10, ∅, 20, 20, 60, 60, 70, 70, 70, 90, 100)
7 - 3	(10, 20, ∅, 20, 60, 60, 70, 70, 70, 90, 100)
7 - 4	(10, 20, 20, ∅, 60, 60, 70, 70, 70, 90, 100)
7 - 5	(10, 20, 20, 60, ∅, 60, 70, 70, 70, 90, 100)
7 - 6	(10, 20, 20, 60, 60, ∅, 70, 70, 70, 90, 100)
7 - 7	(10, 20, 20, 60, 60, 70, ∅, 70, 70, 90, 100)
7 - 8	(10, 20, 20, 60, 60, 70, 70, ∅, 70, 90, 100)
7 - 9	(10, 20, 20, 60, 60, 70, 70, 70, ∅, 90, 100)
7 - 10	(10, 20, 20, 60, 60, 70, 70, 70, 90, ∅, 100)
8 - 1	(10, 90, 90, 90, 100, ∅)
8 - 2	(10, ∅, 90, 90, 90, 100)
8 - 3	(10, 90, ∅, 90, 90, 100)
8 - 4	(10, 90, 90, ∅, 90, 100)
8 - 5	(10, 90, 90, 90, ∅, 100)
9 - 1	(10, 80, 80, 90, 90, 100, ∅)
9 - 2	(10, ∅, 80, 80, 90, 90, 100)
9 - 3	(10, 80, ∅, 80, 90, 90, 100)
9 - 4	(10, 80, 80, ∅, 90, 90, 100)
9 - 5	(10, 80, 80, 90, ∅, 90, 100)
9 - 6	(10, 80, 80, 90, 90, ∅, 100)
10 - 1	(10, 20, 30, 40, 60, 90, 100, ∅)
10 - 2	(10, ∅, 20, 30, 40, 60, 90, 100)
10 - 3	(10, 20, ∅, 30, 40, 60, 90, 100)
10 - 4	(10, 20, 30, ∅, 40, 60, 90, 100)
10 - 5	(10, 20, 30, 40, ∅, 60, 90, 100)
10 - 6	(10, 20, 30, 40, 60, ∅, 90, 100)
10 - 7	(10, 20, 30, 40, 60, 90, ∅, 100)

Tabela C.3: Configurações da Carteira 1 considerando posições vazias (continuação)

Tubete	Mandril	Configuração das Bolachas
1	304, 8	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
2	304, 8	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
3	76, 2	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
4	77	(24, 24, 28, 30, 6, 4)
5	76, 2	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
6	127	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
7	77, 8	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
8	50	(25, 25, 25, 26, 14)
9	57	(19, 19, 20, 20, 27, 27)
10	57	(17, 17, 17, 18, 18, 1, 19, 20, 20, 27, 27)
11	60	(19, 19, 20, 20, 27, 27)
12	60	(17, 17, 17, 18, 18, 1, 19, 19, 20, 20, 20, 27, 27)
13	70	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
14	77	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
15	77	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 13, 13, 30, 6, 31)
16	77	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 3, 13, 30, 6, 31)

Tabela C.4: Configurações da Carteira 2 sem considerar posições vazias

Tubete - Configuração	Configuração das Bolachas
1 - 1	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4, ∅)
1 - 2	(29, ∅, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 3	(29, 29, ∅, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 4	(29, 29, 29, ∅, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 5	(29, 29, 29, 29, ∅, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 6	(29, 29, 29, 29, 29, ∅, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 7	(29, 29, 29, 29, 29, 15, ∅, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 8	(29, 29, 29, 29, 29, 15, 15, ∅, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 9	(29, 29, 29, 29, 29, 15, 15, 15, ∅, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 10	(29, 29, 29, 29, 29, 15, 15, 15, 15, ∅, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 11	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, ∅, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 12	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, ∅, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 13	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, ∅, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 14	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, ∅, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 15	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, ∅, 16, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 16	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, ∅, 16, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 17	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, ∅, 16, 16, 16, 16, 16, 16, 6, 4)
1 - 18	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, ∅, 16, 16, 16, 16, 16, 6, 4)
1 - 19	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, ∅, 16, 16, 16, 16, 6, 4)
1 - 20	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, ∅, 16, 16, 16, 6, 4)
1 - 21	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, ∅, 16, 16, 6, 4)
1 - 22	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, ∅, 16, 6, 4)
1 - 23	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, ∅, 6, 4)
1 - 24	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, ∅, 4)
2 - 1	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5, ∅)
2 - 2	(29, ∅, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 3	(29, 29, ∅, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 4	(29, 29, 29, ∅, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 5	(29, 29, 29, 29, ∅, 29, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 6	(29, 29, 29, 29, 29, ∅, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 7	(29, 29, 29, 29, 29, 15, ∅, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 8	(29, 29, 29, 29, 29, 15, 15, ∅, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 9	(29, 29, 29, 29, 29, 15, 15, 15, ∅, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 10	(29, 29, 29, 29, 29, 15, 15, 15, 15, ∅, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 11	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, ∅, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 12	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, ∅, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 13	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, ∅, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 14	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, ∅, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 15	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, ∅, 16, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 16	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, ∅, 16, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 17	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, ∅, 16, 16, 16, 16, 16, 16, 6, 5)
2 - 18	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, ∅, 16, 16, 16, 16, 16, 6, 5)
2 - 19	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, ∅, 16, 16, 16, 16, 6, 5)

Tabela C.5: Configurações da Carteira 2 considerando posições vazias

Tubete - Configuração	Configuração das Bolachas
2 - 20	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, ∅, 16, 16, 16, 16, 16, 6, 5)
2 - 21	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, ∅, 16, 16, 16, 16, 6, 5)
2 - 22	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, ∅, 16, 16, 16, 6, 5)
2 - 23	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, ∅, 16, 16, 6, 5)
2 - 24	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, ∅, 16, 6, 5)
2 - 25	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, ∅, 6, 5)
2 - 26	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 16, 16, 16, 16, 16, 6, ∅, 5)
3 - 1	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4, ∅)
3 - 2	(9, ∅, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 3	(9, 9, ∅, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 4	(9, 9, 9, ∅, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 5	(9, 9, 9, 9, ∅, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 6	(9, 9, 9, 9, 10, ∅, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 7	(9, 9, 9, 9, 10, 10, ∅, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 8	(9, 9, 9, 9, 10, 10, 10, ∅, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 9	(9, 9, 9, 9, 10, 10, 10, 10, ∅, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 10	(9, 9, 9, 9, 10, 10, 10, 10, 11, ∅, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 11	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, ∅, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 12	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, ∅, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 13	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, ∅, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 14	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, ∅, 15, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 15	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, ∅, 15, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 16	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, ∅, 15, 16, 16, 16, 16, 16, 6, 4)
3 - 17	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, ∅, 16, 16, 16, 16, 16, 6, 4)
3 - 18	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, ∅, 16, 16, 16, 16, 6, 4)
3 - 19	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, ∅, 16, 16, 16, 6, 4)
3 - 20	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, ∅, 16, 16, 6, 4)
3 - 21	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, ∅, 16, 6, 4)
3 - 22	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, ∅, 6, 4)
3 - 23	(9, 9, 9, 9, 10, 10, 10, 10, 11, 11, 11, 11, 15, 15, 15, 15, 16, 16, 16, 16, 16, 6, ∅, 4)
4 - 1	(24, 24, 28, 30, 6, 4, ∅)
4 - 2	(24, ∅, 24, 28, 30, 6, 4)
4 - 3	(24, 24, ∅, 28, 30, 6, 4)
4 - 4	(24, 24, 28, ∅, 30, 6, 4)
4 - 5	(24, 24, 28, 30, ∅, 6, 4)
4 - 6	(24, 24, 28, 30, 6, ∅, 4)
5 - 1	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4, ∅)
5 - 2	(21, ∅, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 3	(21, 21, ∅, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 4	(21, 21, 21, ∅, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 5	(21, 21, 21, 7, ∅, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)

Tabela C.6: Configurações da Carteira 2 considerando posições vazias (continuação)

Tubete - Configuração	Configuração das Bolachas
5 - 6	(21, 21, 21, 7, 7, ∅, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 7	(21, 21, 21, 7, 7, 7, ∅, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 8	(21, 21, 21, 7, 7, 7, 8, ∅, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 9	(21, 21, 21, 7, 7, 7, 8, 8, ∅, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 10	(21, 21, 21, 7, 7, 7, 8, 8, 8, ∅, 8, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 11	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, ∅, 15, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 12	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, ∅, 15, 15, 16, 16, 16, 16, 6, 4)
5 - 13	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, ∅, 15, 16, 16, 16, 16, 6, 4)
5 - 14	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, ∅, 16, 16, 16, 16, 6, 4)
5 - 15	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, ∅, 16, 16, 16, 6, 4)
5 - 16	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, ∅, 16, 16, 6, 4)
5 - 17	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, ∅, 16, 6, 4)
5 - 18	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, ∅, 6, 4)
5 - 19	(21, 21, 21, 7, 7, 7, 8, 8, 8, 8, 15, 15, 15, 16, 16, 16, 16, 6, ∅, 4)
6 - 1	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4, ∅)
6 - 2	(29, ∅, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 3	(29, 29, ∅, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 4	(29, 29, 29, ∅, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 5	(29, 29, 29, 29, ∅, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 6	(29, 29, 29, 29, 29, ∅, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 7	(29, 29, 29, 29, 29, 15, ∅, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 8	(29, 29, 29, 29, 29, 15, 15, ∅, 15, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 9	(29, 29, 29, 29, 29, 15, 15, 15, ∅, 15, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 10	(29, 29, 29, 29, 29, 15, 15, 15, 15, ∅, 15, 30, 30, 30, 30, 30, 6, 4)
6 - 11	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, ∅, 30, 30, 30, 30, 30, 6, 4)
6 - 12	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, ∅, 30, 30, 30, 30, 6, 4)
6 - 13	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, ∅, 30, 30, 30, 6, 4)
6 - 14	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, ∅, 30, 30, 6, 4)
6 - 15	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, ∅, 30, 6, 4)
6 - 16	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, ∅, 6, 4)
6 - 17	(29, 29, 29, 29, 29, 15, 15, 15, 15, 15, 30, 30, 30, 30, 30, 6, ∅, 4)
7 - 1	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4, ∅)
7 - 2	(21, ∅, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 3	(21, 22, ∅, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 4	(21, 22, 22, ∅, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 5	(21, 22, 22, 22, ∅, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 6	(21, 22, 22, 22, 23, ∅, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 7	(21, 22, 22, 22, 23, 23, ∅, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 8	(21, 22, 22, 22, 23, 23, 23, ∅, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 9	(21, 22, 22, 22, 23, 23, 23, 24, ∅, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 10	(21, 22, 22, 22, 23, 23, 23, 24, 24, ∅, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 11	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, ∅, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)

Tabela C.7: Configurações da Carteira 2 considerando posições vazias (continuação)

Tubete - Configuração	Configuração das Bolachas
7 - 12	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, ∅, 15, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 13	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, ∅, 15, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 14	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, ∅, 15, 15, 30, 30, 30, 30, 6, 4)
7 - 15	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, ∅, 15, 30, 30, 30, 30, 6, 4)
7 - 16	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, ∅, 30, 30, 30, 30, 6, 4)
7 - 17	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, ∅, 30, 30, 30, 6, 4)
7 - 18	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, ∅, 30, 30, 6, 4)
7 - 19	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, ∅, 30, 6, 4)
7 - 20	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, ∅, 6, 4)
7 - 21	(21, 22, 22, 22, 23, 23, 23, 24, 24, 24, 24, 15, 15, 15, 15, 30, 30, 30, 30, 6, ∅, 4)
8 - 1	(25, 25, 25, 26, 14, ∅)
8 - 2	(25, ∅, 25, 25, 26, 14)
8 - 3	(25, 25, ∅, 25, 26, 14)
8 - 4	(25, 25, 25, ∅, 26, 14)
8 - 5	(25, 25, 25, 26, ∅, 14)
9 - 1	(19, 19, 20, 20, 27, 27, ∅)
9 - 2	(19, ∅, 19, 20, 20, 27, 27)
9 - 3	(19, 19, ∅, 20, 20, 27, 27)
9 - 4	(19, 19, 20, ∅, 20, 27, 27)
9 - 5	(19, 19, 20, 20, ∅, 27, 27)
9 - 6	(19, 19, 20, 20, 27, ∅, 27)
10 - 1	(17, 17, 17, 18, 18, 1, 19, 20, 20, 27, 27, ∅)
10 - 2	(17, ∅, 17, 17, 18, 18, 1, 19, 20, 20, 27, 27)
10 - 3	(17, 17, ∅, 17, 18, 18, 1, 19, 20, 20, 27, 27)
10 - 4	(17, 17, 17, ∅, 18, 18, 1, 19, 20, 20, 27, 27)
10 - 5	(17, 17, 17, 18, ∅, 18, 1, 19, 20, 20, 27, 27)
10 - 6	(17, 17, 17, 18, 18, ∅, 1, 19, 20, 20, 27, 27)
10 - 7	(17, 17, 17, 18, 18, 1, ∅, 19, 20, 20, 27, 27)
10 - 8	(17, 17, 17, 18, 18, 1, 19, ∅, 20, 20, 27, 27)
10 - 9	(17, 17, 17, 18, 18, 1, 19, 20, ∅, 20, 27, 27)
10 - 10	(17, 17, 17, 18, 18, 1, 19, 20, 20, ∅, 27, 27)
10 - 11	(17, 17, 17, 18, 18, 1, 19, 20, 20, 27, ∅, 27)
11 - 1	(19, 19, 20, 20, 27, 27, ∅)
11 - 2	(19, ∅, 19, 20, 20, 27, 27)
11 - 3	(19, 19, ∅, 20, 20, 27, 27)
11 - 4	(19, 19, 20, ∅, 20, 27, 27)
11 - 5	(19, 19, 20, 20, ∅, 27, 27)
11 - 6	(19, 19, 20, 20, 27, ∅, 27)
12 - 1	(17, 17, 17, 18, 18, 1, 19, 19, 20, 20, 20, 27, 27, ∅)
12 - 2	(17, ∅, 17, 17, 18, 18, 1, 19, 19, 20, 20, 20, 27, 27)

Tabela C.8: Configurações da Carteira 2 considerando posições vazias (continuação)

Tubete - Configuração	Configuração das Bolachas
12 - 3	(17, 17, ∅, 17, 18, 18, 1, 19, 19, 20, 20, 20, 27, 27)
12 - 4	(17, 17, 17, ∅, 18, 18, 1, 19, 19, 20, 20, 20, 27, 27)
12 - 5	(17, 17, 17, 18, ∅, 18, 1, 19, 19, 20, 20, 20, 27, 27)
12 - 6	(17, 17, 17, 18, 18, ∅, 1, 19, 19, 20, 20, 20, 27, 27)
12 - 7	(17, 17, 17, 18, 18, 1, ∅, 19, 19, 20, 20, 20, 27, 27)
12 - 8	(17, 17, 17, 18, 18, 1, 19, ∅, 19, 20, 20, 20, 27, 27)
12 - 9	(17, 17, 17, 18, 18, 1, 19, 19, ∅, 20, 20, 20, 27, 27)
12 - 10	(17, 17, 17, 18, 18, 1, 19, 19, 20, ∅, 20, 20, 27, 27)
12 - 11	(17, 17, 17, 18, 18, 1, 19, 19, 20, 20, ∅, 20, 27, 27)
12 - 12	(17, 17, 17, 18, 18, 1, 19, 19, 20, 20, 20, ∅, 27, 27)
12 - 13	(17, 17, 17, 18, 18, 1, 19, 19, 20, 20, 20, 27, ∅, 27)
13 - 1	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4, ∅)
13 - 2	(9, ∅, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 3	(9, 9, ∅, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 4	(9, 9, 9, ∅, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 5	(9, 9, 9, 9, ∅, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 6	(9, 9, 9, 9, 9, ∅, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 7	(9, 9, 9, 9, 9, 10, ∅, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 8	(9, 9, 9, 9, 9, 10, 10, ∅, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 9	(9, 9, 9, 9, 9, 10, 10, 10, ∅, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 10	(9, 9, 9, 9, 9, 10, 10, 10, 10, ∅, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 11	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, ∅, 24, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 12	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, ∅, 24, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 13	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, ∅, 29, 29, 29, 30, 30, 30, 6, 4)
13 - 14	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, ∅, 29, 29, 30, 30, 30, 6, 4)
13 - 15	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, ∅, 29, 30, 30, 30, 6, 4)
13 - 16	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, ∅, 30, 30, 30, 6, 4)
13 - 17	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, ∅, 30, 30, 6, 4)
13 - 18	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, ∅, 30, 6, 4)
13 - 19	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, ∅, 6, 4)
13 - 20	(9, 9, 9, 9, 9, 10, 10, 10, 10, 24, 24, 24, 29, 29, 29, 30, 30, 30, 6, ∅, 4)
14 - 1	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4, ∅)
14 - 2	(9, ∅, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 3	(9, 9, ∅, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 4	(9, 9, 9, ∅, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 5	(9, 9, 9, 10, ∅, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 6	(9, 9, 9, 10, 10, ∅, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 7	(9, 9, 9, 10, 10, 10, ∅, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 8	(9, 9, 9, 10, 10, 10, 10, ∅, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 9	(9, 9, 9, 10, 10, 10, 10, 2, ∅, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 10	(9, 9, 9, 10, 10, 10, 10, 2, 11, ∅, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)

Tabela C.9: Configurações da Carteira 2 considerando posições vazias (continuação)

Tubete - Configuração	Configuração das Bolachas
14 - 11	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, ∅, 11, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 12	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, ∅, 12, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 13	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, ∅, 12, 12, 12, 13, 13, 13, 6, 4)
14 - 14	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, ∅, 12, 12, 13, 13, 13, 6, 4)
14 - 15	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, ∅, 12, 13, 13, 13, 6, 4)
14 - 16	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, ∅, 13, 13, 13, 6, 4)
14 - 17	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, ∅, 13, 13, 6, 4)
14 - 18	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, ∅, 13, 6, 4)
14 - 19	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, ∅, 6, 4)
14 - 20	(9, 9, 9, 10, 10, 10, 10, 2, 11, 11, 11, 12, 12, 12, 12, 13, 13, 13, 6, ∅, 4)
15 - 1	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 13, 13, 30, 6, 31, ∅)
15 - 2	(10, ∅, 10, 10, 10, 11, 11, 11, 12, 12, 12, 13, 13, 30, 6, 31)
15 - 3	(10, 10, ∅, 10, 10, 11, 11, 11, 12, 12, 12, 13, 13, 30, 6, 31)
15 - 4	(10, 10, 10, ∅, 10, 11, 11, 11, 12, 12, 12, 13, 13, 30, 6, 31)
15 - 5	(10, 10, 10, 10, ∅, 11, 11, 11, 12, 12, 12, 13, 13, 30, 6, 31)
15 - 6	(10, 10, 10, 10, 11, ∅, 11, 11, 12, 12, 12, 13, 13, 30, 6, 31)
15 - 7	(10, 10, 10, 10, 11, 11, ∅, 11, 12, 12, 12, 13, 13, 30, 6, 31)
15 - 8	(10, 10, 10, 10, 11, 11, 11, ∅, 12, 12, 12, 13, 13, 30, 6, 31)
15 - 9	(10, 10, 10, 10, 11, 11, 11, 12, ∅, 12, 12, 13, 13, 30, 6, 31)
15 - 10	(10, 10, 10, 10, 11, 11, 11, 12, 12, ∅, 12, 13, 13, 30, 6, 31)
15 - 11	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, ∅, 13, 13, 30, 6, 31)
15 - 12	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 13, ∅, 13, 30, 6, 31)
15 - 13	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 13, 13, ∅, 30, 6, 31)
15 - 14	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 13, 13, 30, ∅, 6, 31)
15 - 15	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 13, 13, 30, 6, ∅, 31)
16 - 1	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 3, 13, 30, 6, 31, ∅)
16 - 2	(10, ∅, 10, 10, 10, 11, 11, 11, 12, 12, 12, 3, 13, 30, 6, 31)
16 - 3	(10, 10, ∅, 10, 10, 11, 11, 11, 12, 12, 12, 3, 13, 30, 6, 31)
16 - 4	(10, 10, 10, ∅, 10, 11, 11, 11, 12, 12, 12, 3, 13, 30, 6, 31)
16 - 5	(10, 10, 10, 10, ∅, 11, 11, 11, 12, 12, 12, 3, 13, 30, 6, 31)
16 - 6	(10, 10, 10, 10, 11, ∅, 11, 11, 12, 12, 12, 3, 13, 30, 6, 31)
16 - 7	(10, 10, 10, 10, 11, 11, ∅, 11, 12, 12, 12, 3, 13, 30, 6, 31)
16 - 8	(10, 10, 10, 10, 11, 11, 11, ∅, 12, 12, 12, 3, 13, 30, 6, 31)
16 - 9	(10, 10, 10, 10, 11, 11, 11, 12, ∅, 12, 12, 3, 13, 30, 6, 31)
16 - 10	(10, 10, 10, 10, 11, 11, 11, 12, 12, ∅, 12, 3, 13, 30, 6, 31)
16 - 11	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, ∅, 3, 13, 30, 6, 31)
16 - 12	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 3, ∅, 13, 30, 6, 31)
16 - 13	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 3, 13, ∅, 30, 6, 31)
16 - 14	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 3, 13, 30, ∅, 6, 31)
16 - 15	(10, 10, 10, 10, 11, 11, 11, 12, 12, 12, 3, 13, 30, 6, ∅, 31)

Tabela C.10: Configurações da Carteira 2 considerando posições vazias (continuação)

Tubete	Mandril	Configuração das Bolachas
1	20	(10, 10, 10, 10, 10, 10, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 17, 17, 17, 17, 17, 18, 18, 18, 18, 19, 19, 19, 19, 20, 25)
2	21	(15, 15, 15, 15, 15, 15, 18, 18, 18, 18, 18, 18, 18, 18, 18, 18, 20, 20, 20, 20, 20, 22, 22, 22, 22, 24, 24, 24, 24, 30, 36)
3	22	(10, 10, 10, 11, 11, 12, 12, 13, 13, 13, 14, 14, 15, 15, 30, 36)
4	23	(10, 10, 10, 11, 11, 12, 12, 13, 13, 13, 14, 14, 30, 36)
5	24	(10, 11, 11, 12, 12, 13, 13, 13, 30, 36)
6	25	(12, 12, 12, 12, 12, 15, 15, 15, 15, 16, 16, 16, 17, 17, 18, 18, 19, 19, 19, 30, 22)
7	26	(20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 20, 22, 22, 22, 22, 22, 22, 22, 22, 22, 22, 30, 22)
8	27	(10, 10, 10, 10, 10, 10, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 17, 17, 17, 17, 17, 18, 18, 18, 18, 20, 25)
9	27	(10, 10, 10, 10, 15, 15, 15, 30, 35)
10	27	(10, 10, 10, 10, 10, 10, 10, 10, 10, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 30, 35)

Tabela C.11: Configurações da Carteira 3 sem considerar posições vazias

Tubete	Mandril	Configuração das Bolachas
1	30	(10, 10, 10, 10, 10, 15, 15, 15, 15, 16, 16, 16, 18, 18, 18, 18, 19, 19, 30, 35)
2	31	(13, 15, 15, 16, 16, 16, 18, 18, 18, 18, 18, 30, 35)
3	31	(10, 10, 10, 12, 12, 12, 12, 13, 13, 13, 13, 15, 15, 15, 15, 16, 16, 16, 16, 18, 18, 18, 18, 30, 35)
4	31	(10, 10, 10, 12, 12, 12, 12, 13, 13, 13, 13, 15, 15, 15, 15, 30, 35)
5	32	(10, 10, 10, 12, 12, 12, 12, 30, 35)
6	33	(10, 10, 10, 10, 10, 10, 10, 10, 10, 15, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 30, 35)
7	34	(10, 10, 10, 10, 10, 10, 15, 15, 15, 15, 16, 16, 16, 30, 35)
8	34	(13, 18, 18, 19, 19, 19, 20, 20, 30, 35)

Tabela C.12: Configurações da Carteira 4 sem considerar posições vazias

Tubete	Mandril	Configuração das Bolachas
1	40	(12, 12, 12, 12, 12, 15, 15, 15, 15, 16, 16, 16, 17, 17, 18, 18, 19, 19, 19, 30, 22)
2	41	(10, 10, 10, 10, 10, 15, 15, 15, 15, 30, 35)
3	42	(10, 10, 10, 10, 10, 15, 15, 15, 15, 16, 16, 16, 18, 18, 18, 19, 19, 19, 30, 35)
4	42	(10, 10, 10, 10, 10, 15, 15, 15, 15, 16, 16, 16, 18, 18, 18, 30, 35)
5	42	(10, 12, 12, 13, 13, 15, 15, 15, 15, 18, 18, 18, 18, 19, 19, 19, 19, 22, 22, 22, 22, 30, 35)
6	43	(10, 12, 12, 13, 13, 15, 15, 15, 15, 18, 18, 18, 18, 30, 35)
7	44	(10, 11, 11, 19, 19, 20, 20, 20, 20, 22, 22, 22, 22, 30, 35)

Tabela C.13: Configurações da Carteira 5 sem considerar posições vazias

Tubete	Mandril	Configuração das Bolachas
1	50	(10, 11, 11, 11, 19, 19, 19, 20, 20, 20, 20, 20, 22, 22, 22, 22, 22, 25, 25, 25, 25, 26, 26, 26, 26, 30, 35)
2	51	(13, 14, 14, 14, 15, 15, 15, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 22, 22, 22, 22, 23, 23, 23, 23, 30, 35)
3	52	(13, 14, 14, 14, 15, 15, 15, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 22, 22, 22, 22, 23, 23, 23, 23, 30, 25)
4	53	(12, 12, 12, 12, 13, 13, 13, 15, 15, 15, 15, 15, 16, 16, 16, 16, 16, 18, 18, 18, 18, 20, 20, 20, 20, 30, 35)
5	54	(15, 15, 15, 15, 16, 16, 16, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 22, 22, 22, 22, 23, 23, 23, 23, 30, 35)
6	54	(15, 15, 15, 15, 16, 16, 16, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 30, 35)

Tabela C.14: Configurações da Carteira 6 sem considerar posições vazias

Tubete	Mandril	Configuração das Bolachas
1	60	(15, 15, 15, 15, 16, 16, 16, 18, 18, 18, 18, 18, 19, 19, 19, 19, 19, 30, 35)
2	61	(18, 18, 18, 18, 20, 20, 20, 30, 35)
3	62	(12, 12, 12, 12, 12, 15, 15, 15, 15, 16, 16, 16, 17, 17, 18, 18, 19, 19, 19, 30, 22)
4	63	(10, 10, 10, 10, 10, 10, 10, 10, 10, 15, 15, 15, 15, 15, 15, 15, 15, 16, 16, 16, 30, 35)
5	64	(15, 18, 18, 18, 18, 19, 19, 19, 19, 22, 22, 22, 23, 23, 25, 25, 29, 29, 29, 30, 35)
6	64	(22, 23, 23, 23, 23, 25, 25, 25, 25, 30, 25)

Tabela C.15: Configurações da Carteira 7 sem considerar posições vazias