# Extensão: Uma aplicação do problema do carteiro chinês direcionado na coleta de lixo urbano

Felipe Carvalho Góes[a]

[a] *UFBA, Graduando em Ciência da Computação, Salvador, Bahia, Brasil*

**Abstract**

O presente trabalho tem por objetivo sumarizar e replicar a abordagem apresentada no artigo "Uma aplicação do problema do carteiro chinês direcionado na coleta de lixo urbano" da Silva *et al.* (2020) bem como esmiuçar conceitos da teoria dos grafos envolvidos. O algoritmo proposto pelos autores para determinação de rotas mínimas para veículos coletores de lixo será aplicado a uma seção do bairro do Canela, em Salvador. Adicionalmente, será proposto um algoritmo alternativo e comparada a eficiência.

*Keywords:* Sumarização, Problema do Carteiro Chinês, Grafos Eulerianos, Coleta de lixo urbano

## 1. Introdução

Formalmente, um grafo é uma tripla, consistindo em um conjunto de vértices, V(G), um conjunto de arestas, E(G), e uma relação que associa a cada aresta dois vértices West *et al.* (2001). Fundamentalmente, um grafo é uma estrutura que descreve relações, representa elementos e as relações entre eles A. DE OLIVEIRA (2016). Assim, a área de estudo da Teoria dos Grafos tem ampla natureza prática, permitindo modelar problemas reais em grafos de forma a serem resolvidos utilizando conceitos e resultados de Teoria dos Grafos.

Problemas de contagem, minimização, maximização envolvendo conceitos como emparelhamento, coloração, caminhos, ciclos, cobertura, em áreas como comunicação, finanças, projeto de computadores, redes, transportes, logística, RH, física, química, compiladores, são exemplos Loureiro (n.d.).

O artigo "Uma aplicação do problema do carteiro chinês direcionado na coleta de lixo urbano" da Silva *et al.* (2020) aplica a teoria dos grafos para

lidar com a problemática da eficiência das rotas de caminhões de coleta de lixo urbano. A primeira parte da introdução destina-se a estabelecer a relevância da proposta de pesquisa. Os autores apontam que, segundo Beliën *et al.* (2014), a coleta de resíduos sólidos é a operação mais importante e cara do ciclo do descarte dos resíduos sólidos, sendo responsável, segundo Vecchi *et al.* (2019), por cerca de 50% a 80% do custo de operação de limpeza pública. Portanto, otimizar rotas dos caminhões de coleta é um problema relevante.

> A coleta dos resíduos sólidos está relacionada aos problemas de otimização de rotas, uma vez que apresenta alto capital envolvido em termos de recursos financeiros, mão-de-obra e custos operacionais variáveis, que precisam ser minimizados. Assim, quanto melhor for a rota de coleta dos resíduos, maior será a redução nos custos, como também, dos efeitos ambientais gerados (emissão de gás carbônico do veículo coletor, alto consumo de combustível e acúmulo de lixo por não atender todas as ruas) Hannan *et al.* (2018). da Silva *et al.* (2020).

A segunda parte apresenta o encaixe teórico do problema das rotas de coleta de lixo com a Teoria dos Grafos ao introduzir os problemas de roteamento.

Os problemas de roteamento são problemas de cobertura em arcos e em nós Wøhlk & Laporte (2018). A literatura mostra que, a depender dos parâmetros da coleta, a problemática das rotas de coleta de lixo pode ser abordadas sobre diversos problemas de roteamento. Um exemplo é Moro *et al.* (2013) que aplica o Problema do Caixeiro Viajante para a otimização da coleta de lixo rural. Se tratando de coleta urbana em que todas as ruas (arcos) precisam ser percorridas ao menos uma vez, o Problema do Carteiro Chinês é uma alternativa para busca da solução ótima. "Os problemas de roteamento em arcos, em especial o Problema do Carteiro Chinês (PCC) otimiza rotas a partir da cobertura de todos os arcos (segmentos de ruas), tornando a coleta de resíduos sólidos viável e minimizando os custos operacionais." da Silva *et al.* (2020).

O artigo segue com o referencial teórico em que introduz o conceito de grafo, grafo não dirigido, grafo dirigido, grafo misto, passeio (tour), trilha, caminho, ciclo, circuito, grafo euleriano, circuito euleriano, passeio de euler e detalha o Problema do Carteiro Chinês Direcionado (PCCD). Além disso,

apresenta os modelos matemáticos para composição de grafos eulerianos que serão auxiliares ao algoritmo que soluciona o problema.

Após a introdução teórica, os autores apresentam a metodologia utilizada na pesquisa, explicam o passo a passo de funcionamento da solução desenvolvida e os inputs a essa solução que são a abstração das rotas em grafos e a matriz de distâncias obtida pela coleta dos dados geográficos. Então, apresentam os resultados encontrados ao aplicar o algoritmo em dois bairros da cidade de Recife-PE e comparar as distâncias mínimas obtidas nas rotas ótimas com o comprimento das rotas atuais, fornecidas pela Secretaria de Infraestrutura e Serviços Urbanos de Recife. Por fim, concluem o trabalho revisando as descobertas e indicando os próximos passos.

O presente trabalho tem por objetivo esmiuçar conceitos, recriar a abordagem do artigo objeto e disponibilizar um programa simples que permita a resolução do PCCD para redes urbanas, devolvendo a rota ótima do caminhão de lixo a partir da entrada de um grafo que represente o mapa da área a ser coberta pelo caminhão.

Este artigo estrutura-se em uma linha similar. Em seguida a essa introdução serão esmiuçados os conceitos teóricos envolvidos na modelagem e solução do problema, apresentada a solução proposta no artigo sumarizado e um algoritmo alternativo a essa solução, análise de eficiência, aplicação da solução em uma seção do bairro do Canela em Salvador, análise de resultados e conclusão.

## 2. Embasamento teórico

[1]

[1]As defições fornecidas foram extraidas dos seguintes materias:
- West *et al.* (2001)
- da Silva *et al.* (2020)
- https://www.inf.ufsc.br/grafos/definicoes/definicao.html#:~:text
- https://www.ibilce.unesp.br/Home/Departamentos/MatematicaAplicada/socorro/aula2_camin_conexo.pdf
- https://www.ibilce.unesp.br/Home/Departamentos/MatematicaAplicada/socorro4029/representa_grafos2.pdf
- https://www.ime.usp.br/~pf/algoritmos_para_grafos/aulas/weightedgraphs.html
- https://www.ime.usp.br/~pf/algoritmos_em_grafos/aulas/forte.html

**Definition 2.1 (Grafo).** *Um grafo é uma tripla, consistindo em um conjunto de vértices, V(G), um conjunto de arestas, E(G), e uma relação que associa a cada aresta dois vértices*

Uma aresta corresponde a um par não orientado $i, j \in E$, sendo $i, j \in V(G)$, que indicam uma ligação não direcionada entre eles. De forma análoga, um arco é formado por um par orientado $(i, j) \in A$, onde os vértices $i, j \in V(G)$, que indica uma ligação direcionada do vértice i para j.

**Definition 2.2 (Grafo direcionado).** *Um grafo direcionado ou digrafo é uma tripla consistindo em um conjunto de vértices, V(G), um conjunto de arestas, E(G), e uma relação que associa a cada aresta um par ordenado de vértices*

Arestas orientadas, que são pares ordenados, usualmente são denominadas arcos e o conjunto de arcos A(G).

**Definition 2.3 (Grafo misto).** *Um grafo é dito misto quando existem arcos e arestas no mesmo grafo.*

**Definition 2.4 (Grau).** *O grau de um vértice é dado pelo número de arestas que lhe são incidentes.*

No caso do grafo ser dirigido a noção de grau é especializada em:

- Grau de saída (outdegree): O grau de saída de um vértice v corresponde ao número de arcos que partem de v.
- Grau de entrada (indegree): O grau de entrada de um vértice v corresponde ao número de arcos que chegam a v.

**Definition 2.5 (Grafo conexo).** *Um grafo é dito conexo se existir pelo menos um caminho entre cada par de vértices do grafo. Caso contrário, o grafo é chamado de desconexo.*

**Definition 2.6 (Grafo ponderado).** *Dado um grafo G(V, A). G é dito ponderado ou valorado se existe uma função c que associa para cada aresta a de G um número Ca. Diremos que Ca é o custo da aresta a.*

Em um grafo com custos nos arcos, a matriz de adjacências, adj, tem dupla função: além de indicar a presença e ausência de arcos, ela armazena os custos dos arcos. Se os vértices v e w são adjacentes então adj[v][w] é o custo do arco v-w. Senão, adj[v][w] vale 0.

**Definition 2.7 (Fortemente conexo).** *Um grafo é fortemente conexo se para qualquer par(v, w) de seus vértices existe um caminho de v para w e também um caminho de w para v.*

**Definition 2.8 (Passeio).** *Dado um grafo G(V, E), um passeio ou tour em G consiste de uma sequência finita alternada de vértices e arestas, começando e terminando por vértices, tal que cada aresta é incidente ao vértice que a precede e ao que a sucede.*

**Definition 2.9 (Trilha).** *Dado um grafo G(V, E), uma trilha em G consiste de uma sequência finita alternada de vértices e arestas, começando e terminando por vértices, tal que cada aresta aparece apenas uma vez e é incidente ao vértice que a precede e ao que a sucede.*

**Definition 2.10 (Caminho).** *Dado um grafo G(V, E), um caminho em G consiste de uma sequência finita alternada de vértices e arestas, começando e terminando por vértices, tal que cada aresta é incidente ao vértice que a precede e ao que a sucede e não há repetição de vértices. Em outras palavras, um caminho é um trajeto onde não há repetição de vértices.*

**Definition 2.11 (Circuito).** *Uma trilha no qual o vértice inicial e o final são iguais é chamado de trilha fechada ou circuito.*

**Definition 2.12 (Ciclo).** *Um trajeto fechado no qual nenhum vértice (com exceção do inicial e do final) aparece mais de uma vez é chamado de Ciclo ou caminho fechado.*

**Definition 2.13 (Circuito euleriano).** *Dado um grafo G(V, E). Um circuito euleriano é uma trilha fechada que passa por todas as arestas de G*

**Definition 2.14 (Grafo euleriano).** *Um grafo é euleriano se possui um circuito euleriano*

**Definition 2.15 (Tour euleriano).** *Dado um grafo G(V, E). Um tour euleriano é um passeio que passa por todas as arestas de G ao menos uma vez.*

**Definition 2.16 (Matriz de adjascência).** *É uma matriz nxn, denotada por* $X = [xij]$ *e definida como: - xij* $= 1$ *se existe uma aresta entre os vertices vi e vj, - xij* $= 0$ *caso contrário.*

**Definition 2.17 (Supergrafo).** *Um grafo G é dito supergrafo ou grafo aumentado de um grafo H se G conter H, ou seja, H for subgrafo de G.*

A teoria dos grafos introduz uma série de problemas relevantes com alta aplicabilidade prática. O problema objeto é o problema do carteiro chinês.

O Problema do Carteiro Chinês (PCC) é um problema de otimização que pertence à classe de problemas de roteamento de arcos definido por Kwan (1962) como segue: "Um carteiro tem que cobrir seu local de trabalho, antes de retornar ao posto. O problema é encontrar a menor distância de percurso para o carteiro".

Para que o Problema do Carteiro Chinês tenha solução é necessário que o grafo seja euleriano. Assim, conforme o Teorema de Euler ele precisa ser fortemente conexo e ter para todo vértice grau par ou para todo vértice grau de entrada igual ao grau de saída caso grafo direcionado. Se o grafo é euleriano, o problema se reduz a encontrar um circuito euleriano neste grafo Vasconcelos (2017). Para essa busca, qualquer algoritmo que busca circuitos eulerianos em grafos eulerianos como o Algoritmo de Fleury e o Algoritmo de Hierholzer pode ser utilizado.

Se o grafo não for euleriano deverão ser adicionadas cópias de arestas (ou arcos) no grafo para que este se torne euleriano. Neste caso o problema se torna definir quais arestas (ou arcos) devem ser copiadas de modo que o grafo tenha custo mínimo Vasconcelos (2017). Após adicionadas estas arestas (ou arcos) obtêm-se um grafo aumentado que é euleriano e, então, basta encontrar o circuito euleriano dentro deste grafo.

Vale destacar, contudo, que ao buscarmos um circuito euleriano no grafo aumentado G' estamos obtendo um tour euleriano sobre o grafo original, pois G' foi composto pela cópia de arestas.

Segundo Wang e Wen (2002) o Problema do Carteiro Chinês para grafos não direcionados e grafos direcionados possui algoritmos e modelos matemáticos que apresentam solução exata em tempo polinomial.

O artigo objeto e o presente artigo limitam-se ao estudo do Problema do Carteiro Chinês direcionado pois será o de maior utilidade para estudo das rotas. Seja G(V, A) grafo direcionado onde cada arco $a \in A$ têm um custo não negativo associado. O Problema do Carteiro Chinês Direcionado

(PCCD) consiste em identificar um tour de mínimo comprimento que se inicie em algum nó, passe por todos os arcos da rede pelo menos uma vez e retorne ao nó inicial Ahuja *et al.* (1993).

Para o problema da rota da coleta de lixo serão compostos grafos direcionados que representam a rede de ruas da região de coleta. Os vértices representam os cruzamentos e os arcos as ruas, sendo a direção do arco o sentido da rua. Ruas de mão dupla serão adicionadas no grafo como dois arcos em sentidos opostos.

## 3. Implementação

Conforme discutido na seção anterior, a solução do Problema do Carteiro Chinês é simples quando o grafo é euleriano, pois basta buscar o seu circuito euleriano. Contudo, no contexto prático, é comum o grafo sobre o qual buscamos o tour euleriano não ser euleriano e solucionar o PCCD para esses grafos passa por torná-lo euleriano. Para isso, cópias adicionais de alguns arcos precisam ser incluídas.

A literatura introduz diversas abordagens para composição do grafo aumentado euleriano com custo mínimo como a utilização do algoritmo de Dijkstra e emparelhamentos Vasconcelos (2017), programação inteira e programação linear Wang & Wen (2002). A solução adotada pelo artigo objeto é a abordagem de Wang e Wen (2002) que apresenta um modelo matemático de programação linear para obter a solução exata do PCCD.

$$Minimize \quad \sum_{(i,j)\in A} cijxij \qquad (01)$$

Sujeito a:

$$\sum_{(i,j)\in A} xij - \sum_{(i,j)\in A} xji \qquad \forall i \in A \qquad (02)$$

$$xij \geq 1 \text{ e inteiro} \qquad \forall (i,j) \in A \qquad (03)$$

A modelagem consiste em adicionar uma variável de decisão $xij$ que representa o número de vezes que o arco (i,j) é percorrido em G e considerar o custo dessa aresta (cij). As restrições (02 e 03) forçam o grafo a ser euleriano e a função objetivo (01) é minimizar o custo total do tour do carteiro. Resolver a expressão retorna a distância mínima (object value) e a repetição de cada arco (valor de cada variável de decisão).

Para a resolução do PCCD o artigo objeto introduz um programa simples em python que receberá a matriz de distâncias do grafo e determinará o percurso ótimo do caminhão de lixo em 3 passos descritos baixo.

Procedimento em três passos para determinar o tour do caminhão

**Passo 1 Entrada:** Matriz das distâncias dos segmentos de ruas (comprimento dos arcos). **Processo:** Escrever o Modelo de Programação Linear (MPL) para o PCCD em um arquivo de texto. **Saída:** Arquivo com extensão (.lp) para entrada no solver

**Passo 2 Entrada:** Arquivo com extensão (.lp) para entrada no solver. **Processo:** Resolver o MPL, com o objetivo de determinar o número de vezes que cada segmento de rua deverá ser percorrido para minimizar a distância total do trajeto do caminhão de lixo. **Saída:** Quantidade de vezes que cada segmento de rua é percorrido e a distância mínima total.

**Passo 3 Entrada:** Número de vezes que cada segmento de rua é percorrido e a distância mínima. **Processo:** Sequenciar o percurso do caminhão de lixo percorrendo todas as ruas, por meio do algoritmo de Fleury modificado. **Saída:** Tour para o caminhão.

Fonte: da Silva *et al.* (2020) - Artigo objeto

Os autores do artigo objeto não disponibilizaram o código ou qualquer arquivo de referência e a implementação teve que ser refeita.

O programa desenvolvido consiste em um arquivo python que implementa o fluxo descrito no quadro anterior. Inicialmente, deverão ser fornecidos um arquivo csv com a lista de vértices descritos cada qual por um id e um arquivo csv com a relação de arcos, descritos por vértice de início e vértice de origem, e sua extensão. Os arquivos são lidos e cria-se uma representação interna da matriz de adjacência. O programa então gera o modelo matemático de programação linear. Seguindo a diretriz do artigo, foi utilizado o solver SoPlex e o python criará um arquivo.lp com o modelo em formato SoPlex[2] (Passo 1). O usuário deve entrar o arquivo.lp no solver (Passo 2) e o programa aguarda

[2]https://uvadoc.uva.es/bitstream/handle/10324/19454/TFG-I-511.pdf;jsessionid=D61E7659E841B5C1170ADBC597EAF1E6?sequence=1

o nome do arquivo.csv com o resultado da análise. Com o valor de repetição de cada arco uma nova matriz de adjacência será criada internamente e ela será submetida ao Algoritmo de Hierholzer para delimitação do circuito euleriano (o artigo objeto utiliza o algoritmo de Fleury para delimitação do circuito euleriano porém na implementação do presente artigo foi utilizado o algoritmo de Hierholzer uma vez mais eficiente - Passo 3).

O Algoritmo de Fleury é um algoritmo que tem por objetivo identificar um circuito euleriano em um grafo euleriano. O algoritmo se baseia no seguinte procedimento:

1. Compor uma lista vazia T que armazenará o circuito euleriano encontrado
2. Escolher um vértice arbitrário de partida e adicioná-lo a T
3. Escolher e marcar uma aresta incidente ao nó atual que respeite a regra da ponte
4. Percorrer a aresta e adicionar o novo nó atual em T
5. Se todas as arestas incidentes ao nó atual estiverem marcadas retornar a lista T que conterá o circuito euleriano. Se não, repita os passos 3 e 4.

Regra da Ponte:

Se uma aresta v, w é uma ponte no grafo reduzido, então v, w só deve ser escolhida pelo algoritmo de Fleury caso não haja outra opção.

Ao respeitar a regra da ponte, garantimos que todas as arestas do grafo serão percorridas. O Algoritmo foi formulado originalmente para grafos não direcionados mas pode ser ajustado para grafos direcionados. O pseudocódigo para o algoritmo de Fleury modificado é destacado na Figura 1.

O lado negativo do algoritmo de Fleury é que depende de um bom algoritmo para identificar se é ponte e isso pode ser custoso.

Adicionalmente, o presente artigo propõe um algoritmo alternativo para encontrar o circuito euleriano, o Algoritmo de Hierholzer. O Algoritmo de Hierholzer dispensa a identificação de pontes e se baseia no seguinte procedimento:

1. Compor uma pilha S auxiliar e uma lista T que irá armazenar o circuito, inicialmente vazias
2. Escolher um vértice arbitrário de partida e adicioná-lo em S
3. Escolher e marcar uma aresta qualquer incidente ao vértice atual
4. Percorrer a aresta e adicionar o novo vértice atual em S
5. Se todas as arestas incidentes ao vértice atual estiverem marcadas, desempilhe o vértice atual de S, adicione a T e com o vértice que está no topo da pilha repita o passo 5. Se não, repita os passos 3 e 4.
6. Se S está vazio, retorne T que contém o circuito euleriano

O pseudocódigo para o Algoritmo de Hierholzer é destacado na Figura 2.

O algoritmo de Fleury na implementação que segue tem complexidade descrita como O((A)*x), sendo x a complexidade de tempo no pior caso do algoritmo escolhido para verificar se uma aresta é ponte a exemplo de O(V+A) no caso de um algoritmo baseado em DFS. O algoritmo de Hierholzer, por sua vez, é mais eficiente ao dispensar a verificação de pontes, executando em O(V+A).

Como forma de melhor ilustrar o funcionamento do programa ao permitir visão sobre tudo que está sendo processado foi selecionado um pequeno conjunto de ruas pertencentes ao bairro do Canela em Salvador. Devido ao tamanho reduzido, os dados geográficos foram extraídos manualmente, porém o programa permite entrada de dados extraídos de plataformas como Open Street Map (OSM) contanto que obedeçam ao formato de entrada (veja os arquivos edges.csv e nodes.csv no repositório disponibilizado).

## 4. Resultados

O programa descrito pode ser encontrado no **repositório** e foi implementado em linguagem de programação Python versão 3.10.4 onde foi executado em um computador com CPU Snapdragon (TM) 7c Gen 2, 4GB de RAM e sistema operacional Windows 11 Home (ARM x64). O grafo G representativo da seção do bairro do Canela (área da Figura 3) conta com 10 vértices e 13 arestas e pode ser representado pela matriz de distâncias da Tabela 1. O conjunto “Nó inicial” e “Nó final” representa um arco do grafo e os Nós são identificados por seu ID (no OSM os nós são representados por ids únicos).

Table 1: Arcos de G

| Nó inicial | Nó final | Distância (metros) |
|---|---|---|
| 100 | 200 | 114.54 |
| 200 | 300 | 111.8 |
| 300 | 400 | 122.37 |
| 300 | 800 | 166.0 |
| 300 | 1000 | 78.24 |
| 400 | 100 | 111.56 |
| 500 | 400 | 207.23 |
| 600 | 500 | 88.04 |
| 700 | 600 | 202.35 |
| 800 | 300 | 166.0 |
| 900 | 1000 | 73.13 |
| 1000 | 700 | 128.44 |

Fonte: Elaborado pelo autor (2022)

O grafo G apresenta 24 variáveis de decisão no MPL e foi encontrada solução ótima pelo Soplex. O resultado leva à composição do supergrafo G* pela adição de um único arco, (400, 300) com distância 122.37m, a G e a extensão total do trajeto retornado é 1,765 km.

Como foi selecionada uma seção do bairro não foi possível obter a extensão da rota real do caminhão de lixo, porém, a relevância da otimização pode ser visualizada ao comparar o resultado ótimo obtido com outras rotas que também percorrem todas as ruas da seção. Uma rota mais natural ao conjunto de ruas tem extensão total de 1,981 km, uma diferença de 10,9

Por fim, abaixo tour do caminhão impresso pelo programa.

100-200-300-1000-900-1000-700-600-500-400-300-800-300-400-100

## 5. Considerações finais

A contribuição do trabalho consistiu na entrega de um modelo implementável prevendo algoritmo mais eficiente em relação ao algoritmo originalmente implementado, além de esclarecer o pensamento dos autores do artigo de referência. O programa em python está disponível neste **repositório**.

Assim como no artigo objeto, a principal limitação do trabalho está na representação da rede de ruas apenas por grafos direcionados, pois leva a ser necessário supor que as ruas de mão dupla são grandes o suficiente para forçar a passagem do caminhão de lixo em ambos sentidos. Uma possível extensão seria apresentar um programa que resolvesse o Problema do Carteiro Chinês para grafos mistos.

## References


A. DE OLIVEIRA, Valeriano; RANGEL, Socorro ; A. DE ARAUJO Silvio. 2016. *Teoria dos grafos*. UNESP/IBILCE/DMA.

Ahuja, R., Magnanti, T., & Orlin, J. 1993. *Network flow: theory, algorithms, and applications*. Vol. 1. Prentice hall Upper Saddle River.

Beliën, Jeroen, De Boeck, Liesje, & Van Ackere, Jonas. 2014. Municipal solid waste collection and management problems: a literature review. *Transportation science*, **48**(1), 78–102.

da Silva, Andersson Alves, Lins, Sóstenes Luiz Soares, & da Silva Xavier, Amanda. 2020. Uma aplicação do problema do carteiro chinês direcionado na coleta de lixo urbano. *Brazilian journal of development*, **6**(5), 24640–24659.

Hannan, MA, Akhtar, Mahmuda, Begum, RA, Basri, H, Hussain, A, & Scavino, Edgar. 2018. Capacitated vehicle-routing problem model for scheduled solid waste collection and route optimization using pso algorithm. *Waste management*, **71**, 31–41.

Kwan, Mei-Ko. 1962. Graphic programming using odd or even points. *Chinese math*, **1**, 273–277.

Loureiro, Antonio Alfredo Ferreira. *Grafos*. UFMG/ICEx/DCC.

Moro, Matheus Fernando, Schroeder, Wesley, & Cabral de Jesus, Gabriel. 2013. Otimização da rota ótima de coleta seletiva de resíduos na área rerual do munic ípio de missal - paraná, utilizando heuristicas de solução do problema do caixeiro viajante. *In: Xxxiii encontro nacional de engenharia de producao disponível em:https://abepro.org.br/biblioteca/enegep2013*$_t n_s tp_1 82_0 37_2 3303.pdf.acessado$.

Vasconcelos, Rodrigo. 2017. *O problema do carteiro chines dirigido, nao dirigido e misto para otimizacao de rotas com visualizacao grafica da solucao.* Ph.D. thesis, UNIVERSIDADE ESTADUAL DO CEARA.

Vecchi, Thelma PB, Surco, Douglas F, Constantino, Ademir A, & TA, Maria. 2019. Uma abordagem sequencial para otimização de rotas dos caminhões de coleta de resíduos sólidos. *Pages 03–03 of: Congresso de métodos numéricos em engenharia, lisboa. disponível em:. acessado.*

Wang, Hsiao-Fan, & Wen, Yu-Pin. 2002. Time-constrained chinese postman problems. *Computers & mathematics with applications*, **44**(3-4), 375–387.

West, Douglas Brent, *et al.* 2001. *Introduction to graph theory.* Vol. 2. Prentice hall Upper Saddle River.

Wøhlk, Sanne, & Laporte, Gilbert. 2018. A fast heuristic for large-scale capacitated arc routing problems. *Journal of the operational research society*, **69**(12), 1877–1887.

Figure 1: Pseudocodigo do algoritmo de Fleury

**Entrada:**

G = (V, A) // Grafo Euleriano

G' = G com G' = (V', A') // Cópia do grafo original para retornar o tour

T = ∅// Uma lista vazia T para armazenar o tour

n1 ∈ V' // O nó inicial pertence ao grafo G'

**Início:**

T = [n1] // O tour inicia no nó inicial

i = n1 // O nó i recebe o nó inicial

**Enquanto** A' ≠ {∅} **faça:** // Enquanto o conj. de arcos não for vazio, faça:

Escolher um arco a(i,j) adjacente de i que respeite a regra da ponte

T = T u j // Acrescenta o nó rotulado ao tour

A' = A' - (i,j) // Retirar o arco a(i,j)∈ A' do grafo

i = j // Será atribuído ao nó i o nó da próxima iteração j

fim do enquanto

**Retome** T

**fim**

**Saída:**

T contendo um tour Euleriano

Fonte: da Silva *et al.* (2020) - Adaptado pelo autor (2022)

Figure 2: Pseudocodigo do algoritmo de Hierholzer

**Entrada:**

G = (V, A) // Grafo Euleriano

S = ∅ //Uma pilha inicialmente vazia auxiliar

T = ∅ //Uma lista vazia T para armazenar o tour

**Início:**

Escolha um vértice inicial v e insira na pilha S. Todas as arestas iniciam desmarcadas.

**Enquanto** S != ∅ **faça:**

Seja u o vértice do topo da pilha S

Se u possui uma aresta incidente desmarcada para um vértice w, então adicione w a pilha S e marque a aresta (u,w)

Se u não possui aresta incidente desmarcada para um vértice w, então remova u do topo da pilha S e adicione u em T

**Retorne** T

**fim**

**Saída:**

T contendo um tour euleriano

Fonte: Elaborado pelo autor (2022)