Universidade Estadual do Sudoeste da Bahia

Departamento de Ciências Exatas e Tecnológicas

Lucas Amorim Almeida

Teoria dos Grafos e o Problema do Carteiro Chinês

Vitória da Conquista

2018

Lucas Amorim Almeida

Teoria dos Grafos e o Problema do Carteiro Chinês

Trabalho de Conclusão de Curso apresentado ao colegiado do curso de Licenciatura em Matemática da Universidade Estadual do Sudoeste da Bahia – Campus de Vitória da Conquista, para a obtenção do título de Licenciado em Matemática, sob orientação do Prof. Dr. Júlio César dos Reis.

Vitória da Conquista

2018

Lucas Amorim Almeida

Teoria dos Grafos e o Problema do Carteiro Chinês

Trabalho de Conclusão de Curso apresentado como requisito para obtenção do título de Licenciado em Matemática da Universidade Estadual do Sudoeste da Bahia.

Trabalho aprovado. Vitória da Conquista, _____/ _____/ 2018.

Componentes da banca examinadora:

Prof. Dr. Júlio César dos Reis - UESB
orientador

Prof.Ms. Jandresson Dias Pires – IFNMG
2º membro

Prof. Ms. Edson Patricio Barreto de Almeida - IFBA
3º membro

Vitória da Conquista
2018

AGRADECIMENTOS

A Deus pela vida e força para enfrentar e superar as dificuldades.

A meu orientador, Prof. Dr. Júlio César dos Reis pelo seu apoio e confiança, além das várias contribuições para a realização desse trabalho.

A minha família pelo apoio incondicional, em especial a minha mãe, principal modelo de como enfrentar os momentos difíceis da vida.

A todos os professores do Curso de Licenciatura em Matemática, pelos ensinamentos.

A meus colegas de turma, pelos insubstituíveis momentos de alegria que tivemos.

RESUMO

Este trabalho tem por objetivo apresentar alguns conceitos básicos da Teoria de Grafos para que possamos entender o que ocorre quando resolvemos um Problema de Roteamento, chamado Problema do Carteiro Chinês (PCC). Desde a segunda metade do século XX vemos grandes avanços no desenvolvimento e em aplicações da Teoria dos Grafos, aqui tratamos esse problema, usando exemplos fictícios, em pequenas regiões da cidade de Vitória da Conquista. O PCC será abordado de três formas considerando as ruas dos setores escolhidos na cidade, são elas: quando todas as ruas são de mão dupla, quando todas são de mão única e quando existem ruas de mão dupla e mão única. Devemos mencionar que empregaremos algoritmos, modelos de Programação Linear Inteira e o software LINDO para resolver o problema.

Palavras-chave: Teoria dos Grafos, Problema do Carteiro Chinês, aplicação.

ABSTRACT

This paper aims to present some basic concepts of Graph Theory for us to understand what happens when we solve a routing problem called Chinese Postman Problem (CPP). Since the second half of the twentieth century we have seen great advances in the development and application of Graph Theory, here we treat this problem by using fictitious examples in small regions of the city of Vitória da Conquista. The CCP will be approached in three ways considering the streets of the chosen sectors in the city, they are: when all the streets are two-way, when all are single-handed and when there are two-way and one-way streets. We must mention that we will employ algorithms, Integer Linear Programming models and LINDO software to solve the problem.

Keywords: Graph Theory, Chinese Postman Problem, application.

LISTA DE ILUSTRAÇÕES

SUMÁRIO

INTRODUÇÃO

A Teoria dos Grafos tem origens no século XVIII, o seu desenvolvimento se deu principalmente na segunda metade do século XX. Ela possui várias aplicações e o grafo, seu elemento básico, pode ser percebido em várias situações do cotidiano, por exemplo, o que chamamos de árvore genealógica nada mais é que um grafo, também os mapas conceituais de determinados conteúdo e tabelas de campeonatos podem ser vistos como grafos.

Um grafo é composto por dois conjuntos (no capítulo 1 veremos uma definição mais formal do que seria um grafo) um conjunto de vértices e outro do que chamamos ligações (que são relações entre vértices). E como veremos os grafos podem ser utilizados para representas várias coisas e uma delas são regiões de cidades, nas quais as ruas e os cruzamentos entre elas são os elementos do grafo. Ainda poderíamos atribuir valores paras as ligações, como por exemplo, o comprimento da rua.

Com isso, poderíamos pensar em vários problemas de deslocamento em cidades como, por exemplo: Qual é o menor caminho entre dois pontos numa cidade? Na Teoria dos Grafos esses problemas são chamamos de problemas de percursos de custo mínimo.

Um problema muito interessante surge da abordagem de uma cidade como grafo, que é dada uma região da cidade (com suas ruas, seus respectivos comprimentos e as interseções entre elas). Será que podemos percorrer essa região usando todas as ruas de forma que a distância total percorrida seja a mínima possível? Se sim, quais as condições que essa região deve possuir?

Diante dessas questões, definimos o objetivo do nosso trabalho como: Entender os conceitos da Teoria dos Grafos necessários para modelar uma região da cidade e empregar alguns resultados, algoritmos e modelos de Programação Linear Inteira para determinar um percurso nessa região utilizando todas as ruas de forma que a distância percorrida ao final desse trajeto seja a mínima possível.

Assim, exibiremos um problema semelhante, ao que foi posto anteriormente, que intrigava as pessoas em uma cidade da Prússia, que o brilhante matemático Leonhard Euler (1707 – 1783) resolveu esse problema e como consequência disso, criou a Teoria dos Grafos.

Nos capítulos 1 e 2 apresentamos as ferramentas necessárias, fornecidas pela Teoria dos Grafos, para conseguir modelar uma região através de grafo. No capítulo 3, apresentaremos o que chamamos de Problema do Carteiro Chinês, que é o problema de encontrar um circuito de custo mínimo que utiliza todas as ligações de um grafo, e para resolvê-lo vamos nos servir de algoritmos, Programação Linear e do software LINDO. E, finalmente, no capítulo 4 exibimos uma aplicação **fictícia** desse problema na coleta de lixo em algumas regiões de Vitória da Conquista.

1 A ESTRUTURA DOS GRAFOS

"Qualquer coleção de coisas, chamadas de *elementos*, é um conjunto." (LOVÁSZ; PELIKÁN; VESZTERGOMBI, 2013, p. 5, grifo dos autores).

A definição de conjunto é fundamental para uma outra, a definição de grafo, sobre a qual é baseado o nosso trabalho. Resolvemos apresentá-la antes, inclusive, da tradicional introdução histórica, pois o que não é mencionado nessa definição acaba sendo exposto quando observamos a cardinalidade (ou a quantidade de elementos) da união entre dois conjuntos[1] é que os elementos de um conjunto não devem ser repetidos dentro dele, isso é, de forma mais vulgar que, se dois elementos são iguais significa que o conjunto vê apenas um deles e é como se o outro nem existisse.

Dito isso, vamos apresentar (depois de uma breve contextualização histórica) uma definição de multiconjuntos (ou *multiset* em inglês) que é, a grosso modo, um conjunto com elementos repetidos. Essas definições, a de conjunto e de multiconjunto, são fundamentais para entender o que é a *estrutura discreta*[2] chamada *grafo*, sobre a qual se baseia uma interessantíssima teoria. E essa *Teoria dos Grafos* possui inúmeras aplicações, inclusive, neste trabalho lidamos com um problema que consiste num modelo baseado em grafos.

Devido à importância que a Teoria de Grafos tem em nossa pesquisa, dedicaremos este capítulo a apresentar alguns conceitos dela por meio de definições e exemplos. Primeiro, mostraremos um pouco da história dessa teoria e, após isso, uma definição formal do que seja um grafo, atrelando a ela algumas outras noções importantes. E prosseguiremos, apresentando alguns exemplos de grafos. Agora, vamos apresentar um pouco sobre a história da Teoria dos Grafos.

1.1 UM POUCO SOBRE A HISTÓRIA DOS GRAFOS

Tudo começa por causa de Königsberg e suas pontes.Segundo (ROSEN, 2010) o primeiro registro de uso da Teoria dos Grafos na história é devido ao matemático suíço Leonhard Euler (1707 – 1783) no ano de 1736 quando utilizou um modelo de grafo (que será definido mais a diante) para resolver o problema das pontes de Königsberg.

Königsberg era uma cidade prussiana, hoje ela se chama Kalingrad e fica na Rússia. Em Königsberg haviam quatro porções de terra dividas pelo rio Pregel, as quais incluíam duas

[1] Dados dois conjuntos A e B então $|A \cup B| = |A| + |B| - |A \cap B|$, onde $|X|$ é a quantidade de elementos (ou cardinalidade) de um conjunto X.

[2] Caso o leitor se interesse, (ROSEN, 2010) traz no prefácio de seu livro *Matemática discreta e suas aplicações* sob o título *objetivos de um curso de Matemática Discreta* dentre outras uma definição do que seja uma estrutura discreta.

regiões das margens do rio, a ilha de Kneiphof e a região entre os braços (ROSEN, 2010). E sobre o rio haviam sete pontes ligando essas seções, como ilustra a figura 1.

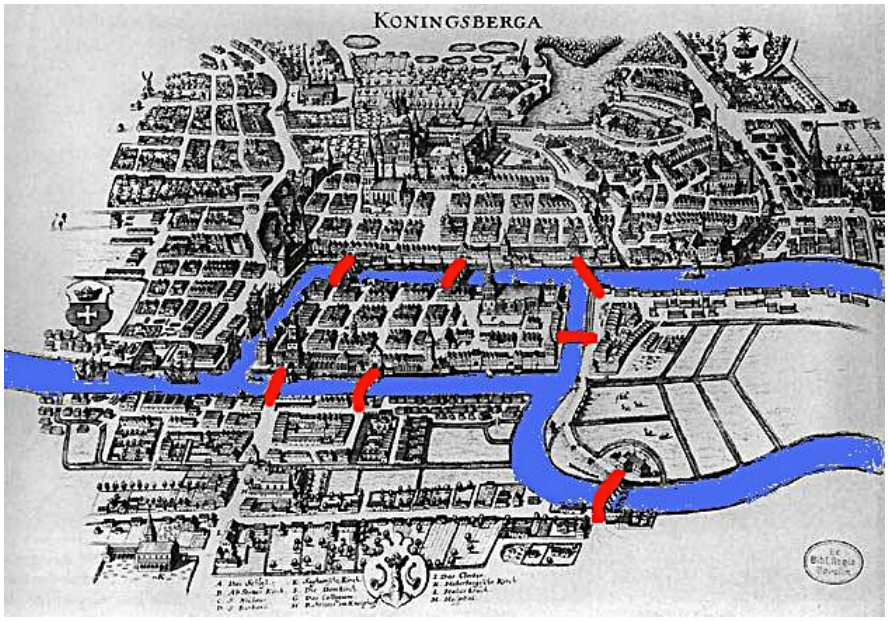


Figura 1 – Ilustração da cidade de Königsberg. Disponível em: <http://www.engeene.it/algoritmi-e-street-view-i-7-ponti-di-kaliningrad/>.

Os habitantes da cidade costumavam fazer longos passeios pela cidade aos domingos e imaginavam se era possível iniciar um percurso de algum lugar de Königsber, atravessar todas as pontes sem cruzar uma ponte mais de uma vez e voltar para o início do percurso (ROSEN, 2010). Daí surge o problema das 7 pontes de Königsberg, que consiste em procurar um percurso no qual se atravessa cada ponte uma, e somente uma, vez. Para resolver esse problema foi necessário rotular as porções de terra conforme a figura 2 a qual se associa o modelo da figura 3.

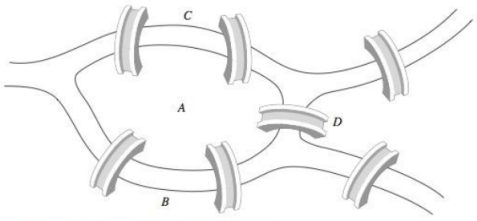


Figura 2 – Ilustração do problema das 7 pontes de Königsberg.Disponível em: (ROSEN, 2010, p. 634)

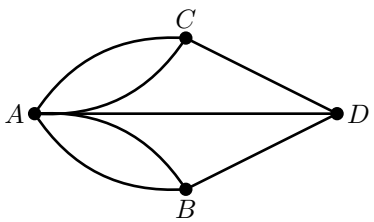


Figura 3 – Modelo do problema das pontes de Königsberg

Ao resolver esse problema Euler também propôs um importante teorema para a Teoria dos Grafos.

Apesar de ter suas origens no século XVIII a Teoria dos Grafos se desenvolveu somente a algumas décadas, na segunda metade do século XX, impulsionada pelos problemas de otimização organizacional (NETTO, 2011). Antes disso, houveram apenas algumas incursões nessa teoria, por exemplo quando, em 1847, Kirchhoff utilizou modelos de grafos no estudo de circuitos elétricos, ou ainda, dez anos mais tarde quando Cayley fez o mesmo, com estrutura de árvore, em isômeros dos hidrocarbonetos alifáticos. E Jordan (1869) estudou as árvores com um olhar matemático (NETTO, 2011).

Outras incursões na Teoria dos Grafos também se deram na então conjectura das 4 cores proposta por Guthrie, que foi provada somente em 1976. No entanto, o interesse pela teoria aumentou, em todo o mundo, após a publicação dos trabalhos de Ford e Fulkerson, Berge e Ore a partir de 1956. A imensa maioria das obras sobre Grafos são de datas posteriores a 1970, também devido a contribuição do avanço dos computadores (NETTO, 2011).

1.2 UMA DEFINIÇÃO DE GRAFO

Como prometemos, agora, vamos definir formalmente o que seria uma Multiconjunto. Essa definição e as subsequentes podem ser encontradas em (STANLEY, 2011).

Definição 1.1. (*Multiconjunto*) Seja S um conjunto. Um *multiconjunto* M de S é um par (S, v), onde v é uma função definida por $v : S \longrightarrow \mathbb{N}$. A imagem de $x \in S$ pela função v é o número de repetições x e ela também é chamada de *multiplicidade de x*.

Observação 1.1. Todo conjunto é um multiconjunto, em que a multiplicidade de cada elemento dele é 1. A saber, neste trabalho consideraremos $\mathbb{N} = \{1, 2, \cdots\}$, logo não é permitido que a multiplicidade de um elemento seja nula. Também, os conjuntos e multiconjuntos considerados nesse trabalho serão **finitos**.

Exemplo 1.1. Considere $S = \{2, 3, 5\}$ então $M = \{2, 2, 2, 2, 3, 5, 5\}$ é um multiconjunto de S. A multiplicidade de 2 é $v(2) = 4$, a de 3 é $v(3) = 1$ e a de 5 é $v(5) = 2$ com isso, a cardinalidade de M é $|M| = v(2) + v(3) + v(5) = 7$.

Exemplo 1.2. Considerando o conjunto finito $S = \{-2,\ \frac{2}{3},\ 1,\ 5\}$. $M = \{-2,\ \frac{2}{3},\ 1,\ 1\}$ não é um multiconjunto sobre S, pois $v(5) = 0$, isto é, $5 \notin M$.

Finalmente, apresentaremos uma definição formal de grafo. Dentre as várias formas de se definir um grafo (que em seu núcleo dizem a mesma coisa), optamos uma adaptação da definição exibida por (NETTO, 2011).

Definição 1.2. Chamamos de *grafo* a estrutura $G = (V,\ E)$, onde V é um conjunto discreto e E é um multiconjunto cujos elementos, $u \in E$, são definidos em função dos elementos de V.

Observação 1.2.
1. Note que o conjunto de vértices V pode ser finito ou infinito. Se V for finito o grafo é finito, e se V for infinito o grafo será infinito (ROSEN, 2010). Mas, aqui trataremos somente de multiconjunto e grafos finitos.
2. Os elementos do conjunto V serão chamados de vértices.
3. Os $u \in E$ são chamados de ligações. E a uma ligação está associada, dentre outras, a possibilidade de passar de um vértice para outro ou para ele próprio, o que consistem em um laço;

Exemplo 1.3. Um exemplo que pode dar uma ideia do que seja um grafo, e também de algo que pode ser modelado por eles, é apresentado por (LOVÁSZ; PELIKÁN; VESZTERGOMBI, 2013, p. 1). "Alice convida seis pessoas para uma festa [...]. Quando eles chegam, apertam as mãos de cada um". Para essa situação podemos associar um grafo onde todas as sete pessoas (Alice e os seis convidados) formam o conjunto dos vértices. Note que o "aperto de mão" se caracteriza como ligação, pois é uma relação que associa duas pessoas na festa, um par de vértices. Logo, todos os apertos de mão formam o multiconjunto das ligações, que nesse caso será somente um conjunto.

Exemplo 1.4. A internet pode ser modelada a partir de um grafo, onde cada página representaria um vértice. E as ligações seriam os hipertextos (hiperlinks) que possibilitam a passagem de uma página para outra.

Com esses exemplos, percebemos que objetos (os vértices) e qualquer relação entre eles (as ligações), poderia ser modelada através um grafo, desde que o conjunto dos objetos envolvido seja enumerável.

Essa definição de grafo não é muito comum em textos introdutórios como em (ROSEN, 2010), (LOVÁSZ; PELIKÁN; VESZTERGOMBI, 2013), entre outros, pois eles afirmam que E é um conjunto. E isso não dá margem para que exista mais de uma ligação entre dois vértices. Obviamente um fato derivado dessa definição baseada em multiconjunto é justamente a possibilidade de existir mais de uma ligação entre dois vértices.

Definição 1.3. Seja M um multiconjunto em um conjunto S com n elementos. Dizemos que p é o *grau de multiplicidade* de M se $p = \max\{v(x_i);\ x_i \in S\}$.

Observação 1.3.
1. Se o grau de multiplicidade de um multiconjunto M é 1 então é um conjunto.
2. O multiconjunto considerado na definição 1.2, tenha grau de multiplicidade finito.

Tendo em vista as definições 1.2 e 1.3 dizemos que um p-grafo é um grafo $G = (V, E)$ cujo grau de multiplicidade de E é p. Assim, os p-grafos são, nada mais que, grafos em que o maior número de ligações entre dois de seus vértices é p. Dentre os p-grafos o de maior interesse para esse trabalho são os 1-grafos.

1.3 ORIENTAÇÃO DAS LIGAÇÕES EM UM GRAFO

A orientação de um grafo está diretamente relacionada com a natureza do multiconjunto das ligações. Ressaltamos, ainda, que esquematicamente, a orientação pode ser visualizada através da presença ou não de direção nas ligações, indicadas por setas.

Caso o leitor se habilite a ler (NETTO, 2011) e (ROSEN, 2010) perceberá que estes autores se referem às ligações de um grafo como orientadas ou não orientadas, mas há outros autores como (FILHO; JUNQUEIRA, 2006) que se referem às ligações por direcionadas ou não direcionadas. Também é muito usual se referir às ligações utilizando os termos "undirected arcs"(arcos não direcionados) ou "directed arcs"(arcos direcionados) em trabalhos escrito na língua inglesa, por exemplo, (AHUJA; MAGNANTI; ORLIN, 1993), (KAPPAUF; KOEHLER, 1979) e (FORD; FULKERSON, 1962). Ainda, é possível que se encontre na literatura autores que se refiram a grafos como redes (networks) o que é muito comum nos trabalhos escritos em inglês, por exemplo, (FILHO; JUNQUEIRA, 2006), (AHUJA; MAGNANTI; ORLIN, 1993), (KAPPAUF; KOEHLER, 1979) utilizam essa nomenclatura. Com isso, optamos seguir, neste trabalho, a nomenclatura usada por (NETTO, 2011) e também, **não** iremos utilizar o termo rede para nos referir aos grafos.

Antes de prosseguirmos aos tipos de grafos, cabe ressaltar que o conjunto das partes de um conjunto X será denotado por $\mathcal{P}(X)$ e o conjunto das partes de X, com n elementos será denotado por $\mathcal{P}_n(X)$. Ainda, considere X^n como o conjunto formado por todas as n-uplas ordenadas possíveis de X. Veja que $1 \leq n \leq |X|$, onde $|X|$ é a quantidade de elementos em X. Por fim, dado um multiconjunto M definimos, $\Lambda(M)$ como sendo o conjunto em que todos os elementos iguais de M serão representados por um, e somente um, elemento em $\Lambda(M)$.

Definição 1.4. (*grafo não orientado*) Seja $G = (V,\ E)$ um grafo que, em particular, se tenha $\Lambda(E) \subseteq \mathcal{P}_1(V) \cup \mathcal{P}_2(V)$ e $\text{gr}(E) = p$, G será chamado de *p-grafo não orientado*.

Observação 1.4.

1. Para efeito de simplificação do texto, a menos que haja necessidade, nos referiremos aos 1-grafos não orientado simplesmente por *grafos não orientados*. A saber, quando o grau de multiplicidade p é maior que 1 os p-grafos também podem ser chamados de multigrafos, como se pode observar em alguns trabalhos em Teoria de Grafos o que não o caso do nosso.

2. Os elementos de E, em um grafo não orientado, serão chamados de arestas.

3. Do fato de $\Lambda(E) \subseteq \mathcal{P}_1(V) \cup \mathcal{P}_2(V)$, mais precisamente quando $u \in \mathcal{P}_2(V)$ implica que $u = \{v_1,\ v_2\}$, para alguns $v_1, v_2 \in V$, porém, note que $\{v_1,\ v_2\} = \{v_2,\ v_1\}$. Isso significa que a ordem em que v_1 e v_2 aparecem não importa na **representação** da aresta.

4. Quando o grafo for não orientado o multiconjunto das arestas será denotado por E, a menos que se explicite o contrário.

Notação 1.1. Aqui utilizaremos a mesma notação para aresta que (NETTO, 2011). Assim uma aresta $u \in E$ será denotada por $u = (v_1, v_2)$, onde $v_1,\ v_2 \in V$ e (v_1, v_2) é um **par não ordenado**.

Perceba que com base no item 3 da observação 1.4 e na notação estabelecida para designar as arestas temos que $u = (v_1, v_2) = (v_2, v_1),\ v_1,\ v_2 \in V$, ou seja, uma aresta pode ser representada tanto por (v_1, v_2) quanto por (v_2, v_1).

Exemplo 1.5. Veja que o grafo não orientado $G = (\{1,2,3\}, E)$, onde E pode ser definido como $E = \{\{1,2\},\ \{1,3\}, \{2,1\}\}$, ou conforme a notação que adotamos, $E = \{(1,2),\ (1,3),\ (2,1)\}$. Perceba que pelo item 3 da observação 1.4 temos $(1,2) = (2,1)$, entretanto isso não significa, necessariamente, que elas são uma única aresta, mas que podemos representá-la tanto por $(1,2)$ quanto por $(2,1)$, pois como podemos observar $(1,2), (2,1) \in E$, isto é, elas são duas arestas que ligam o mesmo par de vértices. Com isso, o grau de multiplicidade de E é $\mathrm{gr}(E) = 2$ implicando que G é um 2-grafo não orientado.

Diante das várias possibilidades para definir E em um grafo $G = (V, E)$, vamos apresentar um tipo de grafo que não se encaixa na definição 1.4, o qual chamaremos de *grafo orientado*. E para distinguir um do outro durante o texto, utilizaremos a representação $G = (V, A)$ para grafos orientados e $G = (V, E)$ para grafos não orientados, a menos que se explicite o contrário.

Definição 1.5. (*Grafo orientado*) Um grafo $G = (V,\ A)$ em que $\mathrm{gr}(A) = p$ e $\Lambda(A) \subseteq V^2$ é chamado de *p-grafo orientado*.

Observação 1.5.

1. Para efeito de simplificação do texto nos referiremos aos 1-grafos orientado simplesmente por *grafo orientado*.
2. Os elementos de A, em um grafo orientado, serão chamados de arcos.
3. Do fato de $\Lambda(A) \subseteq V^2$, onde $V^2 = V \times V$, implica que para alguns $v_1, v_2 \in V$ que determinam $u = (v_1,\ v_2)$ nota-se que $(v_1,\ v_2) \neq (v_2,\ v_1)$, ao contrário da observação 1.4. Isso é, os arcos agora são representados por **pares ordenados** e isso faz com que a ordem em que v_1 e v_2 aparecem no arco defina o seu sentido.
4. G ainda pode ser chamado de *dígrafo*, mas não faremos isto nesse trabalho.

Exemplo 1.6. Com o mesmo conjunto de vértice do exemplo 1.5 termos: $G = (\{1,2,3\}, A)$, e A pode ser definido como $A = \{(1,2),\ (1,3),\ (2,1)\}$. Note que, ao contrário do exemplo 1.5 teremos $(1,2)$ e $(2,1)$ não podem ser a mesma representação para um arco e pelo item 3 da observação 1.5 eles representam arcos em sentidos opostos. Também, ao contrário do exemplo 1.5, o grau de multiplicidade de A é $\mathrm{gr}(A) = 1$ e isso implica que G é 1-grafo orientado.

Cabe ressaltar que as noções de ligações orientadas e não orientadas são mutuamente exclusivas, ou seja, em um grafo se uma ligação é não orientada ela não pode ser orientada

e vice-versa, mais a frente veremos como relacionar arcos e arestas. Quando chamamos um grafo de orientado ou de não orientado estamos afirmando que todas suas ligação são ou não orientadas.

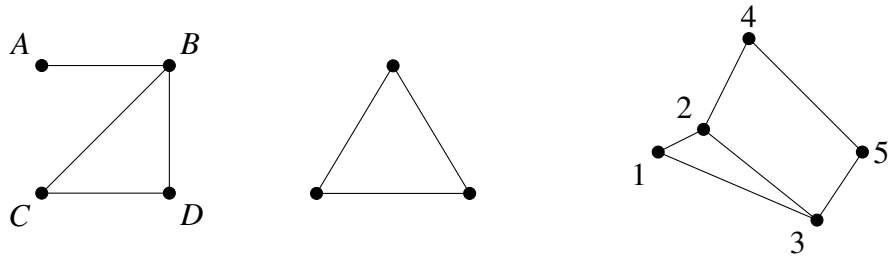
Ainda, existem grafos que possuem, ao mesmo tempo, ligações que são orientadas e outras que são não orientadas a esses grafos damos um nome *grafos mistos*. E esses grafos são muito importantes para esse trabalho.

Notação 1.2. Representaremos um grafo misto como $G = (V, A \cup E)$, onde obviamente $A \cup E$ é um multiconjunto formado tanto por arestas quanto por arcos.

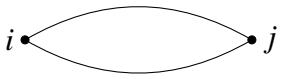
1.4 REPRESENTAÇÃO ESQUEMÁTICA DE GRAFOS

Podemos representar um grafo de várias formas, porém a mais emblemática é sua representação esquemática, na qual os vértices serão representados por pontos ou por regiões limitadas e suas ligações serão representadas por figuras que unam dois vértices, a saber utilizaremos setas e linhas (contínuas e sem orientação) para representar arcos e arestas, respectivamente.

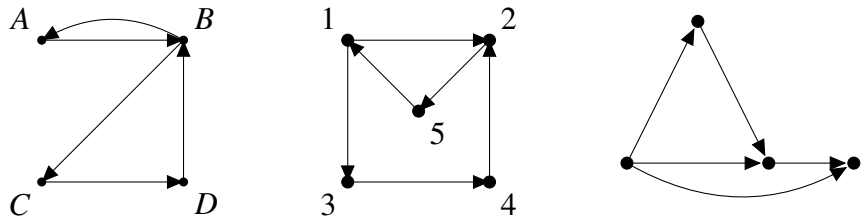
Exemplo 1.7. E, a seguir, apresentamos os seguintes exemplos de grafos não orientados



Observe que as arestas do grafo são representadas por linhas ininterruptas que vão de um vértice a outro. Essas linhas podem ser segmentos de reta ou curvas. Quando mais de uma aresta ligam os mesmos vértices, i e j, usamos as curvas para que não haja confusão entre as arestas, da seguinte forma:

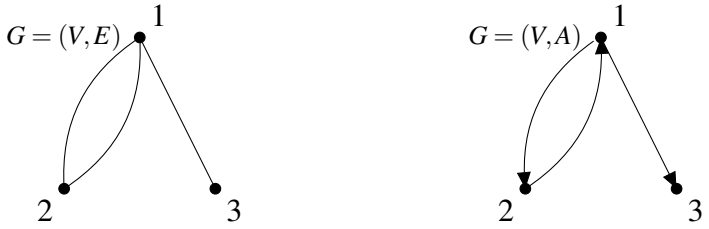


Exemplo 1.8. Agora, os próximos grafos expostos são exemplos de grafos orientado.

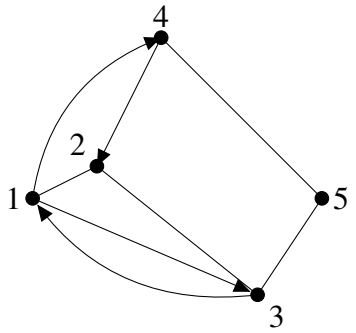


Note que os arcos ligando os vértices são setas, podendo ser curvadas para distinguir dois, ou mais, arcos que estejam no mesmo sentido.

Dessa forma, as representações esquemática dos grafos dos exemplos 1.5 e 1.6 – em que respectivamente temos, $G = (V,E)$, com $E = \{(1,2),\ (1,3),(2,1)\}$ e $G = (V,A)$, onde $A = \{(1,2),\ (1,3),\ (2,1)\}$, em ambos os grafos, $V = \{1,2,3\}$ – são:

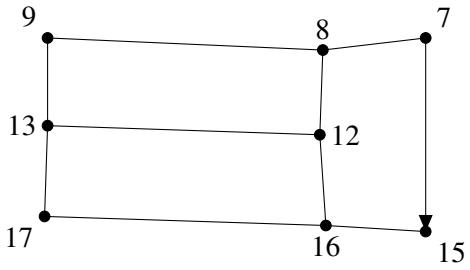


A seguir, apresentaremos um exemplo de representação esquemática de um grafo misto.



Note que podemos representar as ruas de uma região da cidade por um grafo, fazendo corresponder a cada ligação uma rua e a cada vértice um cruzamento entre duas ou mais ruas. Como nas cidades existem ruas de mão única e ruas de mão dupla podemos ainda corresponder a cada rua de mão única um arco e a cada rua de mão dupla uma aresta, e nesse caso, o grafo que representa essa região será um grafo misto.

Exemplo 1.9. A seguir, exibimos um grafo misto que representa algumas ruas de determinada região no Loteamento Vila Marina, bairro Felícia, na cidade de Vitória da Conquista.

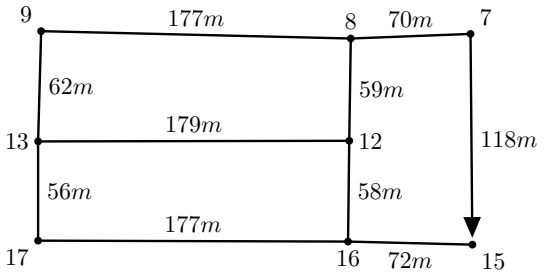


1.4.1 Equivalência orientado – não orientado

Antes de continuarmos, se faz necessário estabelecer alguma relação entre grafos orientados e não orientados, como uma equivalência, por exemplo. Seguindo com a ideia de tentar estabelecer uma relação de equivalência entre grafos orientados e não orientados, vemos que (NETTO, 2011) define tal relação entre as ligações dos grafos. Assim, mantendo os vértices, e sabendo que se há uma aresta entre dois vértices podemos seguir de um vértice para outro tanto em um sentido quanto no outro. Desse modo, é fácil identificar a relação entre uma aresta e dois arcos, em sentidos opostos.

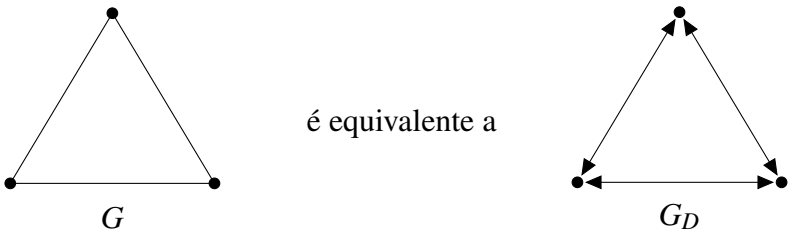
Definição 1.6. Uma ligação diz-se valorada quando a essa ligação está associado um valor (ou peso).

Exemplo 1.10. Podemos valorar as ligações do grafo do exemplo 1.9, de forma que os valores atribuídos as elas sejam correspondentes ao comprimento das ruas.



Definição 1.7. (*equivalência orientado – não orientado*) $G_D(V,A)$ é um grafo orientado que está associado a um grafo não orientado $G = (V,E)$, de modo que a cada aresta de G corresponde biunivocamente a dois arcos de sentidos opostos de G_D.

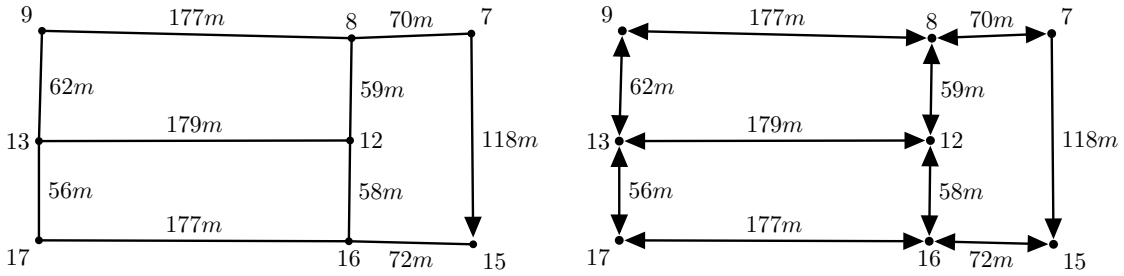
Exemplo 1.11. Veja como seria, por exemplo, um grafo orientado G_D equivalente a um dos grafos do exemplo 1.7.



Pensar na equivalência entre arestas e arcos, dispostos em sentidos opostos, em grafos valorados sobre as ligações é, por um lado, simples e, por outro, complicado. Veja que, se temos uma aresta (i,j) com valor x_{ij} basta substituir por dois arcos em sentidos opostos, ambos com mesmo valor, agora, se tivermos dois arcos em sentidos opostos com mesmo valor, ou seja, dados dois arcos $u = (i,j)$ e $v = (j,i)$ com $x_u = x_v$, onde x_u e x_v são os valores de u e v, respectivamente, devemos substituí-los por uma aresta de mesmo valor. Entretanto, não

aplicamos a essa equivalência quando os valores associados aos arcos em sentidos opostos são diferentes.

Exemplo 1.12. Tomando como referência o grafo do exemplo 1.10 a equivalência orientado-não orientado pode ser aplicada da seguinte forma:



Note que os valores são os mesmos para os arcos nos sentidos opostos, mas não repetimos esses valores para não soar repetitivo. Por exemplo, o arco $(7,8)$ está valorado por $70m$ que é o mesmo valor do arco $(8,7)$.

1.5 OPERAÇÕES EM GRAFOS

Nessa seção discutiremos operações unárias em grafos, mais especificamente, a operação de contração de vértices.

Podemos definir várias operações entre grafos ou entre os elementos de um grafo (vértices e ligações). As operações entre mais de um grafo são chamadas de operações binárias e algumas dessas operações se assemelham bastante as operações entre conjuntos. Em uma operação binária, nota-se que ao operar dois grafos G_1 e G_2 (quaisquer orientados ou não) tem-se como resultado um outro grafo G, não necessariamente diferente dos grafos iniciais a depender da operação. Para exemplificar, temos dentre as operações binárias a união de dois grafos.

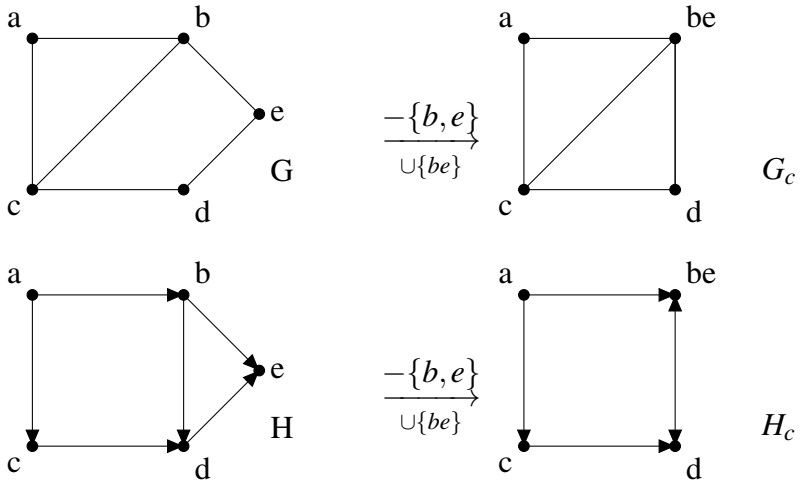
As operações internas aos grafos, ou seja, operações envolvendo os elementos de um grafo, são chamadas de operações unárias. Em uma operação unária, um 1-grafo será transformado em um novo grafo, através dessa operação. Dentre as operações unárias, temos especial interesse na contração de vértices.

Definição 1.8. (*Contração de vértices*) Dado 1-grafo $G = (V,E)$, orientado ou não, a *contração de vértice* transforma dois vértices $v_1,\ v_2 \in V$ em um novo vértice v_1v_2 criando assim um novo grafo $G_c(V_c,E_c)$ tal que $V_c = V - \{v_1,v_2\} \cup \{v_1v_2\}$ e a ligação entre v_1 e v_2 é removida do grafo, além disso, todas as ligações contendo v_1 e v_2 passam a conter v_1v_2.

Observação 1.6.

1. G_c continua sendo 1-grafo.
2. A ordem no rótulo do vértice v_1v_2 indica que o vértice v_2 foi contraído ao vértice v_1.

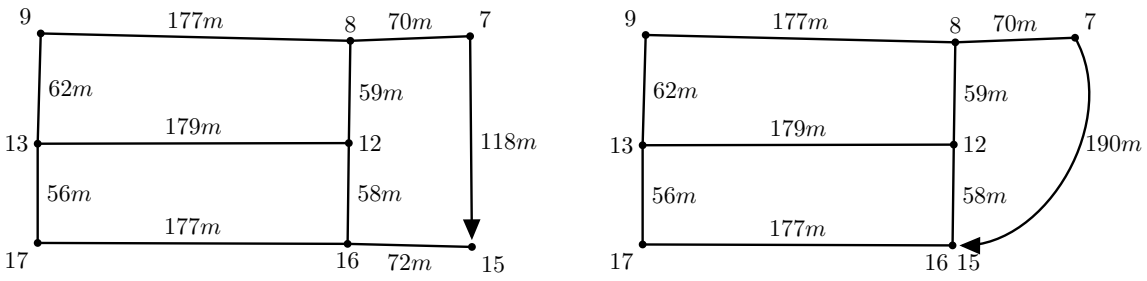
Exemplo 1.13. A seguir apresentamos um exemplo de contração de vértice baseado no exemplo apresentado por (NETTO, 2011, p. 15).



O leitor deve se atentar ao fato que em um grafo orientado o multiconjunto das ligações é formado por pares ordenados. Com isso, $(be,\ d) \neq (d,\ be)$ e assim o grau de multiplicidade do multiconjunto dos arcos é 1, ou seja, um conjunto, e implicando que H é um 1-grafo.

Sobre a contração de vértices em grafos mistos, não há nada a acrescentar, pois a definição 1.8 não depende da orientação das ligações do grafo. Entretanto, devemos fazer uma ressalva para o caso do grafo ser valorado (para grafos de qualquer um dos tipos apresentados aqui), porque dada a contração do vértice v_1 para o v_2, onde (v_1, v_2) valorada por x e, por exemplo, um vértice v que seja adjacente a v_1, onde (v, v_1) é valorada com y, assim, a partir da operação criaremos uma nova ligação (v, v_1v_2) e essa será valorada por $x+y$. Para deixar mais claro, apresentamos um exemplo sobre a contração de vértices em grafos valorados a seguir.

Exemplo 1.14. Mostraremos nesse exemplo com seria uma contração de vértice no grafo apresentado no exemplo 1.10.



Note que, $190m = 72 + 118$.

2 PERCURSOS E ALGORITMOS

No capítulo anterior apresentamos a estrutura de um grafo e algumas definições subsequente a ela. Neste capítulo, iremos complementar o que foi posto sobre Teoria de Grafos, dentro do que for necessário para a aplicação no problema a ser estudado. Com isso, vamos exibir definições, exemplos e alguns resultados relacionados a percursos em grafos.

2.1 VÉRTICES ADJACENTES

A primeira definição apresentada será a de adjacência. Ela é muito importante porque nos diz quando um vértice é vizinho de outro e também como o leitor pode esperar essa definição é bem intuitiva.

Definição 2.1. Dois vértices em um grafo são ditos adjacentes quando existe uma ligação entre eles.

Uma ligação é dita incidente em um vértice se ele constitui uma, e apenas uma, de suas extremidades. Também, duas ligações (arestas ou arcos) são adjacentes quando essas duas incidem sobre um mesmo vértice.

Notação 2.1. Em um grafo $G = (V,E)$ a notação para o conjunto dos vértices adjacentes a um vértice $v \in V$ será $\Gamma(v)$.

Os grafos ainda podem ser representados, entre outros, por listas de adjacência e matriz de adjacência. As duas são de grande importância quando trabalhamos com algoritmos sejam eles implementados manualmente ou computacionalmente. Para maiores informações sobre as representações dos grafos indicamos a leitura de (NETTO, 2011) ou (ROSEN, 2010).

Além da orientação de um grafo, outras noções importantes que se fazem necessárias para o desenvolvimento do nosso trabalho são as noções de grau dos vértices, percursos e circuitos em grafos. Todas as definições e estabelecidas nessa seção são baseadas em (NETTO, 2011).

Definição 2.2. Em um grafo orientado $G = (V,A)$, dizemos que $b \in V$ é *sucessor (antecessor)* de $a \in V$ se existir um arco $(a,b) \in A$ ($(b,a) \in A$).

Como um vértice pode ter mais de um sucessor ou antecessor estabeleceremos uma notação para os *Conjuntos de sucessores* e de *antecessores*. Dessa forma, denotaremos os conjuntos dos sucessores e de antecessores de um dado vértice v por $\Gamma^+(v)$ e $\Gamma^-(v)$, respectivamente.

Observação 2.1. Em grafos não orientados temos $\Gamma(v) = \Gamma^+(v) = \Gamma^-(v)$. Neste caso, não faz sentido fazer distinção entre sucessores ou antecessores de vértices em grafos não orientados,

por isso, nos referimos a $\Gamma(v)$, **no caso de grafos não orientados**, como conjunto dos vértices adjacentes a v

A seguir, exibimos uma definição do que é o grau de um vértice. Essa é uma noção que associa os vértices e as ligações que neles incidem.

Definição 2.3. O grau $d(v)$ de um vértice v é o número total de ligações que nele incidem.

Observação 2.2.

1. Em grafos orientados, teremos $d(v) = d^+(v) + d^-(v)$, onde $d^-(v)$ é o número de arcos que incidem interiormente[1] em v, também chamado de semi-grau interior e $d^+(v)$ é o número de arcos que incidem exteriormente[2] em v, também chamado de semi-grau exterior.
2. Em grafos não orientados teremos $d(v) = |\omega(v)|$, em que $|\omega(v)|$ é uma noção para grafos não orientados e corresponde ao número de arestas que incidem no vértice v e $\omega(v)$ é o conjunto dessas arestas.
3. Em um grafo misto, $G = (V, A \cup E)$, dado $v \in V$ temos $d(v) = d^+(v) + d^-(v) + |\omega(v)|$.

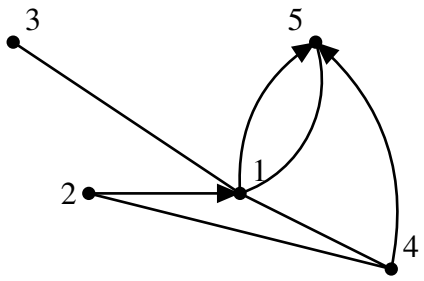
Exemplo 2.1. Com base no grafo misto que será apresentado a seguir:

a) $\Gamma^+(1) = \{3,4,5\}$;

b) $\Gamma^-(1) = \{2,3,4,5\}$;

c) $d(1) = 5$,

onde $d^+(1) = d^-(1) = 1$ e $|\omega(1)| = 3$



2.2 PERCURSOS EM GRAFOS

Agora que vimos que um vértice pode ter vértices adjacentes podemos pensar em sequências de vértices de forma que, cada vértice nessa sequência seja adjacente ao anterior.

Definição 2.4. Um percurso em um grafo é uma sequência de ligações sucessivamente adjacentes, ou seja, é uma sequência de ligações em cadeia que, duas a duas, partilham um vértice.

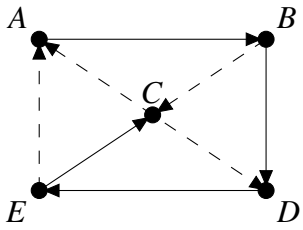
Se nessa sequência existirem n ligações, diremos que esse percurso terá comprimento n. Um percurso com comprimento $n > 1$ que se inicia e termina em um mesmo vértice é chamado de *circuito*.

[1] Um arco $u \in A$ incide interiormente em vértice $v \in V$, quando u é tal que $u = (x, v)$, onde $x \in V$

[2] Da mesma forma, um arco $u \in A$ incide exteriormente em vértice $v \in V$, quando $u = (v, x)$, onde $x \in V$

Notação 2.2. A notação para designar um percurso que inicia em um vértice i e termina em um vértice j será: $P_{ij} = (i, i_1, i_2, \cdots, j_1, j_2, \cdots, j)$. Adotaremos notação semelhante para os circuitos exceto pelo índice da letra usada para denotá-lo onde utilizamos P_i para designar um circuito que se inicia no vértice i.

Exemplo 2.2. No grafo a seguir destacamos, por linhas **não** tracejadas, um percurso entre A e C que pode ser representado por $P_{AC} = (A, B, D, E, C)$.



Existem muitas aplicações que utilizam percursos. É comum que nessas aplicações se procure um percurso de custo mínimo e para encontrar esses percursos mínimos podemos utilizar algoritmos como o algoritmo de Dijkstra. Como o leitor poderá constatar, os percursos são de grande utilidade em nosso trabalho. Seria intuitivo pensar se um percurso pode utilizar todos os vértices de um grafo ou todas as suas ligações. Esses percursos são chamados de *percursos abrangentes* e eles utilizam pelo menos uma vez as ligações ou os vértices de um grafo.

Definição 2.5. Dizemos que um circuito é *euleriano* quando utiliza cada ligação do grafo uma única vez. Um grafo euleriano é um grafo que possuí um percurso euleriano.

No problema das pontes de Königsberg, procurava-se um circuito euleriano na cidade que utilizava todas as pontes que eram as arestas do grafo associado. E, como veremos mais adiante no trabalho, Euler mostrou que esse circuito não existia. Não vamos fazer o mesmo agora, pois ainda não temos condições de enunciar o resultado que nos permite verificar a existência de circuitos eulerianos em grafos.

Ao remover a unicidade da definição 2.5, isso é, quando permitimos um circuito que usa as ligações do grafo uma ou mais vezes, teremos um circuito chamado de *pré-euleriano*.

Veremos nesse trabalho que existe aplicações para circuitos que utiliza todos as ligações num grafo. Mais especificamente, podemos encontrar uso para um circuito pré-euleriano de custo mínimo em um grafo valorado sobre as ligações. Por exemplo, minimizar a rota de um caminhão da coleta de lixo consiste na busca por um circuito pré-euleriano de custo mínimo. E, é nisso que o nosso trabalho consiste.

2.3 CONEXIDADE

A noção de conexidade está diretamente ligada a possibilidade de se encontrar um percurso até um vértice, a partir de outro, ou seja, conexidade está relacionada a seguinte pergunta: "Será possível atingir um vértice em um grafo a partir de um outro?". Agora definiremos formalmente atingibilidade.

Definição 2.6. Em um grafo $G = (V,\ E)$ dizemos que um vértice $v_n \in V$ é *atingível* a partir de um outro vértice v quando há uma sequência de sucessores que se inicia em v e termina em v_n, ou seja, se há um percurso entre v e v_n.

A saber, o conjunto dos vértices atingíveis a partir de v é chamado de *fecho transitivo* de v sendo denotado por $\hat{\Gamma}(v)$.

Em um grafo poderá ocorrer, obviamente, o caso de um vértice v_2 ser atingível a partir outro vértice v_1 e que, v_1 seja atingível a partir de v_2. Neste caso, diremos que v_1 e v_2 são mutuamente atingíveis. A conexidade está relacionada com a possibilidade de passar de um vértice para outro utilizando-se dos arcos ou arestas existentes. Essa ideia, de passagem de um vértice para outro, está relacionada com o conceito de atingibilidade, mas, também está relacionada a conexidade, que é aplicada a todo o grafo, enquanto a atingibilidade é utilizada em pares de vértices.

Observação 2.3.

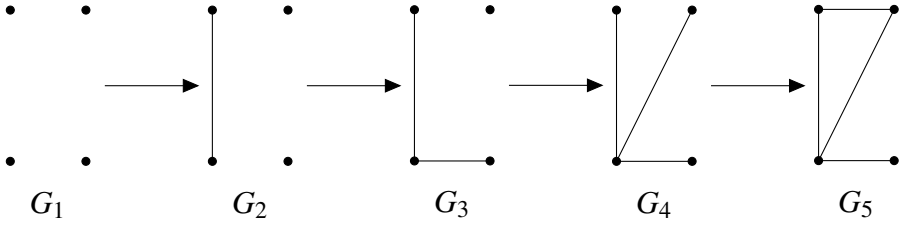
1) O conjunto dos vértices de um grafo G atingíveis a partir de um vértice x é chamado de fecho transitivo direto, sendo denotado por $\hat{\Gamma}^{+}(x)$. Também, o conjunto dos vértices, em um grafo, a partir dos quais um vértice x é atingível é denominado de fecho transitivo inverso, sendo denotado por $\hat{\Gamma}^{-}(x)$

2) Se G é um grafo não orientado $\hat{\Gamma}(x) = \hat{\Gamma}^{+}(x) = \hat{\Gamma}^{-}(x)$

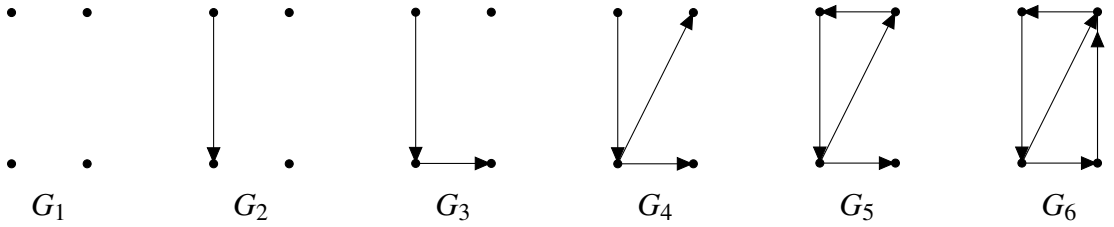
Finalmente, apresentaremos a definição de conexidade em grafos.

Definição 2.7. Um grafo $G = (V,\ E)$ é dito *conexo* quando todos os vértices do grafo são, dois a dois, mutuamente atingíveis. Ou seja, quando, dados dois vértices de G, existir um percurso os unindo. Quando um grafo não é conexo diremos que ele é não-conexo ou, simplesmente, desconexo.

Também, é comum existir variações dessa definição em grafos orientados nas quais se considera, ou não, o sentido dos arcos nos percursos entre os vértices. A saber, em grafos orientados quando se considera o sentido das ligações para verificar a atingibilidade mutua, diz-se que o grafo é fortemente conexo ou f-conexo. Veja, agora, dois exemplos bastante didáticos, extraídos de (NETTO, 2011, p. 31). Neles, podemos ver a construção de grafos conexos a partir da adição de ligações em grafos. O primeiro, diz respeito ao caso do grafo ser não orientado e o segundo, caso o grafo seja orientado.



Observe que G_1 é desconexo e a partir dele vamos adicionando arestas até que em certo momento ele se torna conexo. Perceba que os grafos G_1, G_2 e G_3 são desconexos, pois existem vértices isolados, e G_4 e G_5 são conexos, porque quando escolhemos um vértice qualquer conseguimos encontrar um percurso ligando-o aos outros três vértices, ainda que eles não sejam adjacentes.



Note que os grafos G_1, G_2 e G_3 são desconexos, também, G_4 e G_5 são conexos **se desconsiderarmos a direções dos arcos**. Já G_6 é f-conexo. Cabe, aqui, ressaltar que quando nos referimos a conexidade em grafos mistos os tratamos por f-conexo ao invés de simplesmente usar o termo conexo.

2.4 RESULTADOS SOBRE CIRCUITOS EULERIANOS

Vamos apresentar, nessa seção, alguns resultados que nos ajudarão a verificar se um grafo é euleriano, ou não. O primeiro deles é o Teorema de Euler e fornece as condições necessárias e suficiente para a existência de circuitos eulerianos em grafos não orientados.

Teorema 2.1. (Euler) Um grafo $G = (V,\ E)$ não orientado e conexo possui um circuito euleriano se, e somente se, $\forall v \in V$ se verificar que o grau de v é par.

Demonstração:

($\Rightarrow$) Seja $G = (V,\ E)$ euleriano.

Ao procurarmos neste grafo um circuito euleriano podemos escolher um vértice e daí prosseguir atravessando os demais vértices, apagando as arestas[3] utilizadas.

[3] Na prática, não apagamos uma aresta em seu sentido literal, apenas a marcamos a partir daí consideraremos que esta aresta não existe no grafo (durante a busca do percurso euleriano) para que futuramente se saiba que já a percorremos.

Logo, ao atravessarmos algum vértice v, seu grau diminuirá em duas unidades (pois apagamos duas arestas que incidem sobre ele, uma para a entrada e outra para a saída do vértice). Portanto, todos os vértices intermediários no percurso deverão ter grau par. Caso contrário, seria impossível anular todos os graus neste processo. E mais, o vértice inicial também deverá ter grau par, isso porque ao usarmos uma das arestas incidentes a ele no início do processo deixamos o vértice com grau ímpar, permitindo, assim, a anulação do seu grau no fim do percurso.

($\Leftarrow$) Seja $G(V,\ E)$ um grafo onde todos os seus vértices tenha grau par.

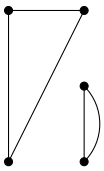
Considere uma aresta $(v,u) \in E$ cuja supressão mantenha a conexidade de G e procuremos um percurso μ_{vu} que não utilize (v,u). Ao apagarmos as arestas do caminho à medida que as percorremos. Existem duas possibilidades:

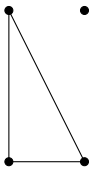
(i) o percurso utiliza todas as arestas restantes no grafo, e então μ_{vu} juntamente com (v,u) formam um percurso euleriano;

(ii) o percurso não utiliza todas as arestas que restam ao grafo. Assim, pelo mesmo raciocínio anterior e também por hipótese, os graus finais não nulos dos vértices intermediários serão pares, e isso permitirá a definição de ciclos secundários saindo e voltando para μ_{vu}, que podem ser percorridos apagando-se as arestas até que todos os graus sejam anulados. Logo, ao chegarmos a u, atravessa-se (v,u), o que anula o grau dos vértices v e u. Portanto, forma-se um percurso euleriano.

E isso encerra a demonstração.

A saber, existem grafos não orientados que são desconexos e, também, eulerianos. Esses são casos não previstos no teorema de Euler, pois em grafos desconexos existem vértices que não são atingíveis a partir de algum outro e, com isso, não podemos traçar percursos entre eles, mas isso não, necessariamente, implica que alguma aresta deixe de ser percorrida. Veja que, grafos desconexos que possuem vértices *isolados*, que são vértices sobre os quais não incidem ligações, podem ser euleriano. Por exemplo:

G_1 : 

G_2 : 

Agora, devemos observar que ainda há a necessidade que todos os vértices tenham grau par. Quando estudamos grafos dessa forma, podemos particionar o conjunto dos vértices de forma a criar, pelo menos dois, subgrafos[4] os quais não possuem ligações entre si. Desses subgrafos, somente um deve possuir ligações e obedecer as condições do Teorema de Euler, isso é ser conexo e não ter vértices de grau ímpar.

[4] Dado $G = (V,E)$, um subgrafo será um outro grafo $H = (W,F)$ tal que $W \subseteq V$ e $F \subseteq E$.

Com base nisso, notamos que G_1 possui todos os vértices com grau par, mas não é possível percorrer todas as arestas em um circuito, pois ao particionar o conjunto de vértice em dois, da forma que mencionamos, ambos possuem arestas e dessa forma se percorremos as arestas de um subgrafo, não percorremos as do outro e vice-versa.

Já G_2 também possui todos os vértices de grau par, inclusive o vértice isolado que tem grau 0 e como somente uma das componentes possui arestas, então G_2 é euleriano.

Sobre o problema das pontes de Königsberg, agora temos condições de verificar a não existência de um circuito euleriano no grafo associado ao problema. Pode se constatar que o grafo associado ao problema, figura 3, é conexo e que, no entanto, possui todos os vértices de grau ímpar. Portanto, ele não é euleriano.

Definição 2.8. Um grafo qualquer $G = (V,E)$ será dito simétrico quando a relação associada a E for simétrica, em outros termos, G será simétrico quando:

$$(v,\ w) \in E \Longleftrightarrow (w,\ v) \in E,\ \forall v,w \in V$$

Note que, todo grafo não orientado é simétrico. Isso se justifica, pois, a cada aresta em um grafo não orientado fazemos corresponder a dois arcos, em um grafo G_D (orientado), em sentidos opostos, isso conforme a definição 1.7.

Definição 2.9. Um grafo qualquer $G = (V,E)$ é dito pseudo-simétrico quando se observa que:

$$d^+(v) = d^-(v),\ \forall v \in V$$

Ainda, é natural se pensar que grafos não orientados são também pseudo-simétricos, e mais que os grafos simétricos são pseudo-simétricos. Essas afirmações são de fácil verificação, mas não parece ser pertinente apresentá-las nesse trabalho. Agora, exibiremos dois resultados sobre a existência de circuitos eulerianos em grafos orientados e mistos, nessa ordem.

Teorema 2.2. Uma condição necessária e suficiente para que um grafo orientado e conexo admita um percurso euleriano é que ele seja pseudo-simétrico.

Observação 2.4. Para a finalidade de nosso trabalho, convém interpretar esse teorema da seguinte forma: se o grau de cada vértice de um grafo for par e a quantidade de vértices incidindo interiormente a qualquer vértice for a mesma que incide exteriormente, então esse grafo possuirá um percurso euleriano. Além disso, não apresentaremos a demonstração desse teorema, visto que sua demonstração é muito parecida a do teorema 2.1, exceto pela necessidade de considerar arcos no lugar das arestas e de considerar os semi-graus dos vértices num ciclo.

Uma observação semelhante a observação feita após o teorema 2.1 também pode ser verificada nesse caso.

Observação 2.5. Um grafo $G = (V,A)$ desconexo com vértices isolados e com um subgrafo $H = (V',A)$ f-conexo, onde V' é o conjunto dos vértices não isolados, que obedece a condição de pseudo-simetria do teorema 2.2 será euleriano. Isso se verifica pois, os vértices isolados não possuem arcos incidindo sobre eles, com isso, não fazem parte do circuito e, além disso, o subgrafo H é euleriano, visto que obedece ao teorema 2.2. Portanto, G é euleriano.

A seguir, vamos apresentar um resultado, formulado por Ford e Fulkerson, que estabelece condições necessárias e suficientes para a existência de percursos eulerianos em grafos mistos.

Teorema 2.3. (Ford e Fulkerson) O grafo misto $G = (V,\ A \cup E)$ é euleriano se, e somente se:

(a) G é conexo;

(b) A todo vértice de G é incide um número par de ligações[5];

(c) Para todo $S \subseteq V$, a diferença entre o número de arcos de S para $V - S$ e o número de arcos de $V - S$ para S é menor ou igual ao número de arestas entre S e $V - S$.

Não apresentaremos a demonstração desse teorema, pois Ford e Fulkerson o demonstram determinando um fluxo (ou circulação) viável no grafo, conceito esse que não definimos nesse trabalho, ainda que os fluxos não estejam tão distantes do que fazemos nesse trabalho. Novamente, uma ponderação semelhante a observação 2.5 pode ser feita aqui, nas mesmas condições, exceto pela pseudo-simetria que será substituída pelas condições (b) e (c) do teorema 2.3.

Também, não vamos fazer extenso uso do teorema 2.3, visto que a verificação do item (c) é muito laboriosa porque como afirmam (LOVÁSZ; PELIKÁN; VESZTERGOMBI, 2013) e (STANLEY, 2011) o número de subconjuntos de um conjunto com n elementos é 2^n. E mesmo considerando que o item vale para o conjunto $\emptyset$ e seu complementar (todo o conjunto dos vértice), ainda precisaríamos fazer $2^{n-1} - 1$ testes para o item, isso porque ao testarmos o item para um conjunto também testamos para seu complementar, e isso reduz a quantidade de testes pela metade. Então, para evitar a fadiga, nos grafos trabalhados nos capítulos futuros vamos tentar, sempre que for possível, não verificar a validade do item (c).

2.5 ALGORITMOS EM GRAFOS

O Algoritmo de Dijkstra é um dentre os algoritmos utilizados para a determinação de um percurso de menor custo entre vértices a partir de um vértice inicial. Recomendamos esse algoritmo, pois ele fornece as distâncias mínimas entre o vértice inicial e todos os outros vértices do grafo. A seguir, mostramos o Algoritmo de Dijkstra apresentado por (NETTO, 2011). Consideraremos no algoritmo um grafo $G = (V,E)$ e o vértice inicial como 1.

[5] No texto original de Ford e Fulkerson aparece "arcs" ao invés de links, no entanto esses arcos são de forma implícita o que eles chamam de "directed arcs" e "undirected arcs", ou seja, arcos direcionado ou não direcionados, (direcionado seria pelo que convencionamos orientado)

Algoritmo 2.1. (Algoritmo de Dijkstra)
Início

$d_{1\,1} \leftarrow 0$; $d_{1\,i} \leftarrow \infty$, $\forall i \in V - \{1\}$; $S \leftarrow \{1\}$; $A \leftarrow V$; $F \leftarrow \emptyset$;
$rot(i) \leftarrow 0$, $\forall i$
Enquanto $A \neq \emptyset$ **faça**

$r \leftarrow v \in V$ tal que $d_{1\,v} = \min_{i \in A}\{d_{1\,i}\}$;
$F \leftarrow F \cup \{r\}$; $A \leftarrow A - \{r\}$; $S \leftarrow A \cap \Gamma^{+}(r)$;
Para $i \in S$ **faça**

$p \leftarrow \min\{d_{1i},\ d_{1r} + v_{ri}\}$, onde v_{ir} é o custo da ligação que vai de i para r.
Se $p < d_{1i}$ **então faça**

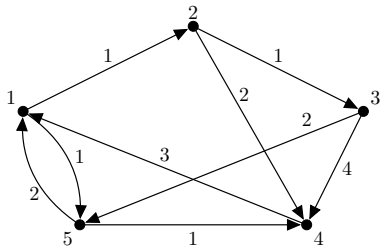
$d_{1i} \leftarrow p$
$rot(i) \leftarrow r$

Fim;

Fim;

Fim;
Fim;

Exemplo 2.3. Dado o grafo a seguir, buscaremos os percursos de custo mínimo a partir do vértice 1, (exemplo apresentado por (NETTO, 2011)).



Início: $d_{11} \leftarrow 0$; $d_{1i} \leftarrow \infty, \forall i \in V - \{1\}$; $S \leftarrow \{1\}$; $A \leftarrow V$; $F \leftarrow \emptyset$; $rot(i) \leftarrow 0, \forall i$;

Para $k = 1:\ r = 1;\ F = \{1\}; A = \{2,3,4,5\}; S = A \cap \{2,5\} = \{2,5\}$;

Para $i \in \{2,5\}:\ p = \min\{\infty,\ (0+1)\} = 1 < d_{12};\ d_{12} = 1; rot(2) = 1$;

$p = \min\{\infty,\ (0+1)\} = 1 < d_{15};\ d_{15} = 1; rot(5) = 1$;

Para $k = 2:\ r = 2;\ F = \{1,2\}; A = \{3,4,5\}; S = A \cap \{3,4\} = \{3,4\}$;

Para $i \in \{3,4\}:\ p = \min\{\infty,\ (1+1)\} = 2 < d_{13};\ d_{13} = 2; rot(3) = 2$;

$p = \min\{\infty,\ (1+2)\} = 3 < d_{14};\ d_{14} = 3; rot(4) = 2$;

Para $k = 3:\ r = 5;\ F = \{1,2,5\}; A = \{3,4\}; S = A \cap \{1,4\} = \{4\}$;

Para $i \in \{4\}:\ p = \min\{3,\ (1+1)\} = 2 < d_{14};\ d_{14} = 2; rot(4) = 5;$

Para $k = 4:\ r = 3;\ F = \{1,2,3,5\}; A = \{4\}; S = A \cap \{4,5\} = \{4\};$

Para $i \in \{4\}:\ p = \min\{2,\ (2+4)\} = 2 = d_{14};\ d_{14} = 2; rot(4) = 5;$

Para $k = 5:\ r = 4;\ F = \{1,2,3,4,5\}; A = \emptyset; S = \emptyset;$ **Fim**.

E obtemos o vetor $rot = (0,1,2,5,1)$ que pode ser interpretado como: o caminho para 2 vem do vértice 1; para o vértice 3 vem de 2 que, por sua vez, vem de 1; e o caminho para 4 vem do vértice 5, mas o caminho para 5 vem diretamente de 1.

O Algoritmo Húngaro é uma técnica que objetiva encontrar um acoplamento de cardinalidade máxima com custo mínimo em um grafo bipartido valorado, no qual os valores geralmente são não negativos. Um acoplamento é um subconjunto de ligações M tal que nenhum vértice seja incidente a mais de uma ligação de M. Também consideramos um grafo bipartido como grafo que possui uma partição $\mathbf{P} = \{W_1, W_2;\ W_1 \cap W_2 = \emptyset\}$ do conjunto de vértices tal que **não existem ligações** entre quaisquer dois vértices de W_1, o mesmo para os vértices de W_2. Assim, esse acoplamento de custo mínimo é o que queremos encontrar quando precisamos adicionar arestas ao grafo original para que ele se torne euleriano.

Agora apresentaremos o Algoritmo húngaro, também exposto em (NETTO, 2011)

Algoritmo 2.2. (algoritmo húngaro)

Primeira fase

1. **Subtrair** o menor elemento de cada linha de todos os elementos dessa mesma linha.
2. Se em cada coluna obtivemos um zero, ir para 4. Se obtivemos **exatamente** um zero em cada coluna, temos uma solução ótima, **fim**.
3. **Subtrair** de todos os elementos de cada coluna, *que não possuir zero*, o menor valor dessa coluna, voltar para 2.
4. **Marcar** um *zero* (fazendo, assim, uma alocação) e **inabilitar** a linha e a coluna as quais esse zero pertença. **Repetir** esse passo até que não haja mais zeros disponíveis.
5. Se um acoplamento perfeito foi obtido, **fim**; senão passar à segunda fase;

Note que no passo 4. o zero a ser marcado é o primeiro da linha e se houver outros nessa mesma linha ou na coluna, pelo fato delas terem sido inabilitadas, estes não poderão ser utilizados como uma alocação.

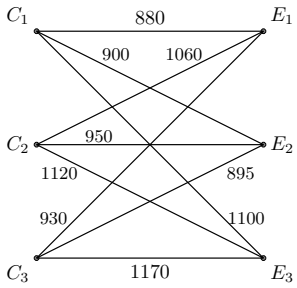
Segunda Fase

1. **Marcar** as linhas que não receberam alocação no passo 4. da primeira fase.

2. **Marcar** as colunas que possuem um zero em alguma das linhas marcadas.
3. **Marcar** as linhas que receberam alocação nas colunas marcadas.
4. **Repetir** os passos 2. e 3. até que não se possa mais marcar linhas ou colunas.
5. **Riscar** todas as linhas não marcadas e todas as colunas marcadas.
6. **Subtrair** o menor elemento não marcado todos os elementos não marcados. E somá-lo aos elementos que estejam, simultaneamente, em uma linha e em uma coluna riscada.
7. **Voltar** ao passo 4. da primeira fase.

Exemplo 2.4. Três casais (C_1, C_2 e C_3) querem matricular seus filhos em uma escola de forma que os gastos mensais com os estudos dos filhos sejam os menores possíveis, mas existem três delas e cada uma possui apenas uma vaga. Cada escola (E_1, E_2 e E_3) demanda um gasto diferente para cada um dos casais. A escola E_1 tem custo de R$ 880,00 para o casal C_1, R$ 1.060,00 para o C_2 e R$ 930,00 para C_3. A escola E_2 custa R$ 900,00 do casal C_1, R$ 950,00 para C_2 e R$ 895,00 para C_3. E E_3 custa R$ 1.100,00 para o casal C_1, R$ 1.120,00 para C_2 e R$ 1.170,00 para C_3.

A esse problema podemos associar o seguinte grafo que é um grafo bipartido, com isso poderemos utilizar o algoritmo húngaro para resolvê-lo.



A esse grafo podemos associar a seguinte matriz e nela cada entrada é o custo de uma ligação.

$$D(V) = \begin{array}{c} \begin{matrix} c_1 & c_2 & c_3 \end{matrix} \\ \begin{pmatrix} 880 & 1060 & 930 \\ 900 & 950 & 895 \\ 1100 & 1120 & 1170 \end{pmatrix} \end{array} \begin{matrix} E_1 \\ E_2 \\ E_3 \end{matrix}$$

Os elementos destacados em $D(V)$ são os menores de cada linha que serão subtraídos.

$$\xrightarrow{Passo\ 1} \begin{pmatrix} 0 & 180 & 50 \\ 5 & 55 & 0 \\ 0 & 20 & 70 \end{pmatrix} \xrightarrow{Passos\ 2\ e\ 3} \begin{pmatrix} 0 & 160 & 50 \\ 5 & 35 & 0 \\ 0 & 0 & 70 \end{pmatrix}$$

E essa última matriz possui um zero em cada linha e coluna. Portanto, encerramos a aplicação do algoritmo não sendo necessária a aplicação da segunda fase. Os zeros destacados, em verde, são as alocações e elas, nesse exemplo significam que o casal C_1 deve mandar seu filho para a escola E_1, o casal C_2 deve mandar o seu para a escola E_3 e o casal C_3 deve mandar seu filho para a escola E_2.

2.5.1 Um algoritmos de busca de circuitos eulerianos

Aqui, vamos apresentar um algoritmo para a determinação, se for possível, de circuitos eulerianos em grafos. Esse algoritmo se baseia em grafos reduzidos. Grafo reduzido são grafos que têm ligações e vértices removidos, em relação ao grafo original. Na aplicação do algoritmo, vamos removendo esses elementos à medida que os inserimos na solução do problema (CARVALHO, 2017). Note também que já utilizamos grafos reduzidos na demonstração do Teorema 2.1 quando apagávamos as arestas.

O algoritmo que será exposto e utilizado nesse trabalho é o algoritmo de Hierholzer, também poderíamos utilizar um outro algoritmo para o mesmo fim, como por exemplo, o algoritmo de Fleury apresentado em (CARVALHO, 2017). Esses algoritmos são realizados em tempo polinomial (CARVALHO, 2017) e partem da hipótese que já foi verificada a existência de um circuito euleriano. Segundo (ROSEN, 2010, p. 193), "A complexidade temporal de um algoritmo pode ser expressa em termos de número de operações usadas num algoritmo quando a entrada tem um determinado tamanho." e é uma forma de medir a eficácia de um algoritmo. Um algoritmo tem complexidade polinomial (realizados em tempo polinomial) se forem da ordem $\Theta(n^b)$, com $b \geq 1$.

Algoritmo de Hierholzer

Segundo (CARVALHO, 2017) algoritmo de Hierholzer foi proposto em 1873 e foi um dos primeiros a traçar circuitos eulerianos em grafos. Ele consiste em percorrer ligações em ciclos[6] sucessivos que retornam para um vértice inicial pertencente a um ciclo inicial. Dessa forma, enquanto houver um vértice com ligações não percorridas, devemos começar percursos nesse vértice e tentar retornar a ele percorrendo as ligações não percorridas.

Considere, no algoritmo, E' como o conjunto das ligações não percorridas; K como o grafo reduzido criado a partir de G; e H como o conjunto das ligações que definem um circuito em K.

Algoritmo 2.3. (*algoritmo de Hierholzer*)

Início

Entrada: $G = (V, E)$;

Escolher um vértice $v \in V$;

[6] Entenda *ciclo* como um sinônimo de circuito.

Construir um ciclo C_v, a partir de v, percorrendo as ligações de G aleatoriamente;

$E' \leftarrow E - C_v$;

$K \leftarrow (V, E')$;

Enquanto $E' \neq \emptyset$, faça:

Escolher um vértice v tal que $d(v) > 0$ e $v \in C$;

Construir um circuito H a partir de v, percorrendo as ligações de K aleatoriamente;

$E' \leftarrow E' - H$;

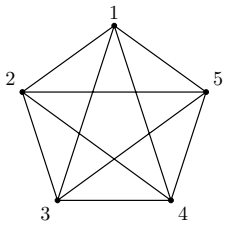
$C_v \leftarrow H \cup C_v$;

$H \leftarrow \emptyset$;

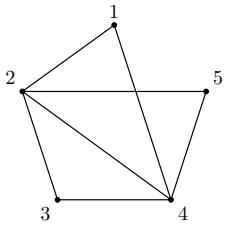
fim;

Fim;

Exemplo 2.5. Neste exemplo, que é muito simples, utilizaremos o Algoritmo de Hierholzer para encontrar um circuito euleriano no grafo $G = (\{1,2,3,4,5\}, E)$ a seguir.

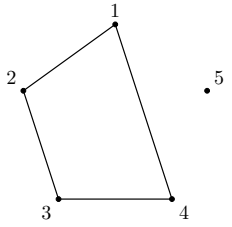


Primeiramente, o grafo é euleriano, pois se verifica o Teorema de Euler (Teorema 2.1). Assim, escolhendo o vértice 1, podemos tomar o ciclo aleatório $C_1 = (1,5,3,1)$ e remover de E as arestas usadas em C_1 obtendo o seguinte grafo $K = (\{1,2,3,4,5\}, E')$



Como $E' \neq 0$ fazemos a o que é pedido dentro do laço (enquanto ... faça) e obtemos os seguintes resultados em suas iterações:

1ª iteração: Escolhendo 5 adicionaremos o ciclo $C_5 = (5,4,2,5)$ ao ciclo principal C_1, resultando:



$H = (\{1,2,3,4,5\}, E')$ e $C_1 = (1,5,4,2,5,3,1)$

2ª iteração: Escolhendo 3 adicionaremos o ciclo $C_3 = (3,4,1,2,3)$ ao ciclo principal C_1. E terminamos o algoritmo, pois $E' = \emptyset$, com o ciclo $C_1 = (1,5,4,2,5,3,4,1,2,3,1)$.

Veremos como todos esses algoritmos – do algoritmo 2.1 até o algoritmo 2.3 – funcionam juntos no momento de resolver um exemplo no próximo capítulo.

3 O PROBLEMA DO CARTEIRO CHINÊS

O Problema do Carteiro Chinês, sendo conhecido por PCC ou CPP (do inglês Chinese Postman Problem), é basicamente um problema de roteamento de ligações em grafos. Isso é, consiste em encontrar uma rota, ou circuito fechado de custo mínimo em um grafo. Esse circuito é pelo menos pré-euleriano, ou seja, um circuito que utiliza todas as ligações pelo menos uma vez. Isso significa que o Problema do Carteiro Chinês é posto sobre um grafo, por isso é essencial saber alguns conceitos da Teoria dos Grafos e alguns deles foram apresentadas nos capítulos anteriores.

Este problema pode ser relacionado ao famoso problema das pontes de Königsberg, mencionado na contextualização histórica do primeiro capítulo desse trabalho. Ressalta-se que mesmo não sendo possível encontrar um circuito euleriano no grafo associado ao problema, como ele é conexo conseguimos encontrar alguns circuitos pré-eulerianos e uns deles possuem custo mínimo. Claramente, se um grafo for euleriano o circuito euleriano terá custo mínimo. Ainda, o PCC é considerado da classe NP-completo, quando se trabalha em um grafo misto.

Pela última frase do parágrafo anterior é natural pensar que a resolução do Problema do Carteiro Chinês dependerá da forma como se dispõem as ligações, ou seja, se todas são orientadas, ou não. Veremos como a orientação das ligações interfere na resolução do PCC.

3.1 PROCEDIMENTOS PARA A RESOLUÇÃO DO PCC

Para se resolver um PCC temos basicamente dois caminhos. O primeiro e mais comum de ser utilizado consiste no uso modelos de Programação Linear. E o segundo é um procedimento baseado na aplicação de alguns algoritmos da Teoria de Grafos em sequências. Ambos os caminhos para a resolução serão apresentados neste capítulo. Mas ressaltamos que utilizamos somente um procedimento com uso de algoritmos da Teoria dos Grafos.

3.1.1 Um procedimento que utiliza algoritmos da Teoria dos Grafos

Um procedimento para a resolução do PCC para grafos não orientados pode ser encontrado em (NETTO, 2011), este procedimento consiste na aplicação sucessiva de algoritmos da Teoria dos Grafos. Esse autor não propõe formalmente outros procedimentos, apenas indica como o procedimento para resolver o problema pode ser adaptado para grafos orientados.

A seguir apresentamos o procedimento proposto por (NETTO, 2011) para a resolução do PCC em grafos não orientados. Esse procedimento também é proposto por (AHUJA; MAGNANTI; ORLIN, 1993) com o mesmo intuito, entretanto, ao contrário de (NETTO, 2011), esses

autores não propõe uma alteração no procedimento para resolver o Problema do Carteiro em grafos orientado.

Procedimento 3.1. 1. Verificar se G é euleriano; caso positivo ir para 6.

2. Determinar um conjunto T, formado pelos vértices de grau ímpar de G.

3. Determinar as distâncias d_{vw} para $v,w \in T,\ v < w$, usando por exemplo o algoritmo de Dijkstra (algoritmo 2.1).

4. Seja $D(S) = [d_{vw}]$ uma matriz obtida com os valores obtidos no passo anterior, com $d_{vv} = \infty$. Aplicar a $D(T)$ o algoritmo Húngaro (algoritmo 2.2);

5. Acrescentar a G uma aresta (k,l) para cada alocação reciproca k,l feita pelo algoritmo e atribuir o valor d_{kl} a essa aresta.

6. Aplicar um algoritmo de busca de circuitos eulerianos, por exemplo, o algoritmo de Hierholzer (algoritmo 2.3).

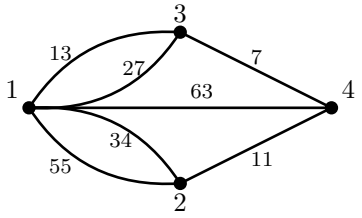
Observação 3.1. Ao concretizar o 5º passo do procedimento 3.1 obtermos um conjunto $E' = (k,l)$, para cada alocação k,l e l,k (uma aresta para duas alocações) e, com isso, acabamos por criar um grafo $G' = (V, E \cup E')$, o qual chamaremos de grafo auxiliar. Note que, custo final do circuito encontrado em G', e por consequência em G, será:

$$C_f = \sum_{(i,j)\in E} C_{ij} + \sum_{(k,l)\in E'} C_{kl}$$

Veja que a verificação da existência de um percurso euleriano é feita utilizando o Teorema de Euler (2.1), pois G seria um grafo não orientado. Também, o primeiro passo do procedimento é comum em todas as formas de se resolver um PCC, pois como afirmado anteriormente, se um grafo é euleriano então o circuito de custo mínimo será justamente o circuito euleriano. Com isso, independente das abordagens de resolução do PCC o primeiro passo é usar um dos teoremas para verificação de existência de circuitos eulerianos, apresentados na seção anterior.

3.1.2 Um exemplo da aplicação do procedimento 3.1

Dado o grafo associado ao problema das pontes de Königsberg o qual denotaremos por $G = (V,E)$, atribuiremos valores arbitrários, não-negativos, às suas arestas como se segue:



Como vimos anteriormente este grafo não é euleriano e com esses valores associados às suas arestas podemos pensar em aplicar o procedimento 3.1 para encontrar um circuito de custo mínimo em G e, neste caso, seguimos diretamente para o passo 2.

Note que o conjunto de todos os vértices de grau ímpar é $T = \{A,B,C,D\} = V$. Perceba, ainda, que não seria necessário aplicar o algoritmo de Dijkstra (algoritmo 2.1), visto que o grafo tem apenas 4 vértices e por isso poderíamos encontrar as distâncias mínimas apenas comparando-as, mas vamos aplicá-lo e mostrar aqui apenas a execução de uma iteração.

Aplicação do algoritmo 2.1 para o vértice 1

$d_{11} \leftarrow 0$; $d_{1i} \leftarrow \infty$, $\forall i \in T - \{1\}$; $S \leftarrow \{1\}$; $A \leftarrow T$; $F \leftarrow \emptyset$; $rot_1(i) \leftarrow 0$;

Como $A = T \neq \emptyset$, temos:

$r \leftarrow 1$, pois $d_{1v} = \min\{0,\infty,\infty,\infty\} = 0 = d_{11}$;
$F \leftarrow \emptyset \cup \{1\}$; $A \leftarrow T - \{1\}$; $S \leftarrow A \cap \Gamma^+(1) = \{2,3,4\}$;
Para $i \in S$, fazemos

$i = 2 : p \leftarrow 34$, pois $34 = \min\{d_{12}, d_{11}+v_{12}, d_{11}+v'_{12}\} = \{\infty, 34, 55\}$;
como $34 = p < \infty = d_{12}$, temos que $d_{12} \leftarrow 34$ e $rot_1(2) \leftarrow 1$;
$i = 3 : p \leftarrow 13$, pois $13 = \min\{d_{13}, d_{11}+v_{13}, d_{11}+v'_{13}\} = \{\infty, 13, 27\}$;
como $13 = p < \infty = d_{13}$, temos que $d_{13} \leftarrow 34$ e $rot_1(3) \leftarrow 1$;
$i = 4 : p \leftarrow 63$, pois $63 = \min\{d_{14}, d_{11}+v_{14}\} = \{\infty, 63\}$; como
$63 = p < \infty = d_{14}$, temos que $d_{14} \leftarrow 63$ e $rot_1(4) \leftarrow 1$;

Agora, $A = \{2,3,4\} \neq \emptyset$, E assim encerramos a primeira iteração do algoritmo. Alertamos que com as demais iterações os valores armazenados em $rot_1(i),\ i \in \{1,2,3,4\}$ podem ser alterados. E isso realmente acontecerá. Prosseguindo com a aplicação do algoritmo 2.1 encontraremos o vetor $rot_1 = (1,1,1,2)$, porque $rot_1(4)$ teve seu valor atualizado em alguma iteração. Posto isso, podemos interpretar esse vetor da seguinte forma:

$$\begin{array}{cc} 1 \quad 2 \quad 3 \quad 4 & \longleftarrow \text{Posições} \\ \left(0 \quad 1 \quad 1 \quad 2 \right) & \longleftarrow \textit{Vetor } rot_1 \end{array}$$

Considerando os fins da aplicação do algoritmo $rot(1)$ não tem significado claro, uma vez que no próximo passo na matriz $D(T)$ cada linha representara um vetor $rot_i,\ i \in T$ e o elemento a_{11} dessa matriz correspondente ao $rot_1(1)$ será ∞.

$rot_1(2) = 1$, ou seja, a segunda posição do vetor rot_1 significa que o percurso (com menor custo) de 1 até 2 vem diretamente de 1, cujo custo é $d_{12} = 34$. Da mesma forma $rot_1(3) = 1$, e com isso, o percurso de menor custo (com $d_{13} = 13$) indo de 1 até 3, vem diretamente de 1. Agora, $rot_1(4) = 2$ significa que 4 vem diretamente de 2 que por sua vez vem de 1, logo o percurso de 1 até 4 inicia em 1, passa por 2 para finalmente chegar em 4, e seu custo é $d_{14} = 45$.

Ao aplicar a primeira fase do algoritmo 2.1 para os demais vértices de T e prosseguindo para o passo 2. do procedimento 3.1 encontraremos a seguinte matriz $D(T)$:

$$D(T) = \begin{array}{c} \begin{matrix} 1 & 2 & 3 & 4 \end{matrix} \\ \begin{pmatrix} \infty & 34 & 13 & 45 \\ 34 & \infty & 18 & 11 \\ 13 & 18 & \infty & 7 \\ 45 & 11 & 7 & \infty \end{pmatrix} \end{array} \begin{matrix} 1 \\ 2 \\ 3 \\ 4 \end{matrix}$$

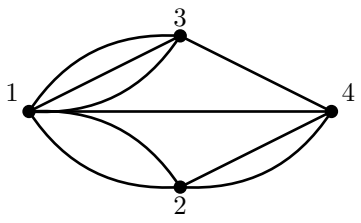
Agora, vamos aplicar o algoritmo húngaro (algoritmo 2.2) nessa matriz. Localizando os elementos mínimos de cada linha e subtraindo:

$$\begin{pmatrix} \infty & 34 & \boxed{13} & 45 \\ 34 & \infty & 18 & \boxed{11} \\ 13 & 18 & \infty & \boxed{7} \\ 45 & 11 & \boxed{7} & \infty \end{pmatrix} \Longrightarrow \begin{pmatrix} \infty & 21 & 0 & 32 \\ 21 & \infty & 7 & 0 \\ 6 & 11 & \infty & 0 \\ 38 & 6 & 0 & \infty \end{pmatrix}$$

Note que, não obtivemos 0 em duas colunas. Então, encontraremos os mínimos de cada coluna e subtraímos de cada elemento da coluna, estaremos aptos a verificar o passo 4.

$$\begin{pmatrix} \infty & 21 & \boxed{0} & 32 \\ 21 & \infty & 7 & \boxed{0} \\ \boxed{6} & 11 & \infty & 0 \\ 38 & \boxed{6} & 0 & \infty \end{pmatrix} \Longrightarrow \begin{pmatrix} \infty & 15 & 0 & 32 \\ 15 & \infty & 7 & 0 \\ 0 & 5 & \infty & 0 \\ 32 & 0 & 0 & \infty \end{pmatrix}$$

Observe que esse é um acoplamento perfeito. Portanto, não será necessário passar à segunda fase do algoritmo. De forma resumida, interpretamos esse resultado como: devemos acrescentar um arco indo do vértice correspondente à linha para o vértice que corresponde à coluna na posição em que uma alocação (destacada em verde) foi obtida. Ou seja, incluiremos ao grafo G os seguintes arcos $(1,3)$, $(2,4)$, $(3,1)$ e $(4,2)$. Entretanto, podemos utilizar a equivalência entre arestas e arcos (definição 1.7) para substituir os arcos $(1,3)$ e $(3,1)$ por uma **aresta** $(1,3)$ e os arcos $(2,4)$ e $(4,2)$ por outra **aresta** $(2,4)$, também, o custo dessas arestas serão 13 e 11, respectivamente. Dessa forma obtemos $E' = \{(1,3),(2,4)\}$ e por consequência o seguinte grafo $G' = (V, E \cup E')$:



O custo de qualquer circuito euleriano nesse grafo G' é $C_f = 13+27+7+63+11+55+34+13+11 = 234$. E para finalizar a resolução do PCC vamos aplicar o algoritmo de Hierholzer (algoritmo 2.3). A partir do vértice 1, traçamos um circuito aleatório, por exemplo:

$$C_1 = (1,2,4,1) \Longrightarrow C = \{(1,2),(2,4),(4,1)\}$$

De agora em diante, determinamos E_1, mesmo que essa não seja a notação adotada no algoritmo, porque o intuito é somente diferenciar de E' que é o multiconjunto das arestas de G'. Então, $E_1 = E' - C = \{(1,2),(2,4),(3,4),(1,3),(1,3),(1,3)\}$. Definimos, então um outro grafo $K = (V, E_1)$. Como $E'' \neq \emptyset$, para o vértice 1 (novamente escolhido aleatoriamente, dentre os elementos de V) definiremos um novo circuito $H_1 = (1,3,1)$, também definido aleatoriamente sobre K, onde $H = \{(1,3),(3,1)\}$.

Novamente, definimos $E_2 = E_1 - H = \{(1,2),(2,4),(3,4),(1,3)\}$. Agora, determinamos C_1' como $C_1' = C_1 \cup H_1$. Continuando a aplicação do algoritmo de Hierholzer encontraremos um percurso C_1, sobre G, definitivo $C_1 = (1,3,4,2,1,2,4,1,3,1)$, o qual é a solução do PCC.

3.2 O PCC E A PROGRAMAÇÃO LINEAR

Apesar de existirem procedimento para resolver um Problema do Carteiro Chinês utilizando uma série de algoritmos, quando pesquisamos sobre o tema é muito comum de nos depararmos com resoluções que utilizam modelos de *Programação Linear*. E estes modelos mudam a depender da orientação das ligações em um grafo.

Segundo (BAZARAA; JARVIS; SHERALI, 2010), Programação Linear (PL) consiste na minimização ou maximização (isso é, uma otimização) de uma função linear, chamada de função objetivo, essa otimização que satisfaz um conjunto restrição (equações lineares ou inequações de restrições, também lineares).

Nesse sentido, um problema de Programação Linear é um problema onde $z = c_1X_1 + c_2X_2 + \cdots + c_nX_n$ é a função objetivo a ser otimizada. Os coeficientes $c_1, c_2, \cdots, c_n$ são os coeficientes de custos (que são conhecidos) e $X_1, X_2, \cdots, X_n$ são as variáveis de decisão que serão determinadas. Simbolicamente, isso significa:

$$\min \qquad z = \sum_{i=1}^{n} C_i X_i$$

Sujeito as restrições:

$$\begin{array}{ccccccccc}
a_{11}X_1 & + & a_{12}X_2 & + & \cdots & + & a_{1n}X_n & \geq & b_1 \\
a_{21}X_1 & + & a_{22}X_2 & + & \cdots & + & a_{2n}X_n & \geq & b_2 \\
\vdots & & \vdots & & & & \vdots & & \vdots \\
a_{m1}X_1 & + & a_{m2}X_2 & + & \cdots & + & a_{mn}X_n & \geq & b_m
\end{array}$$

$$X_1, X_2, \cdots, X_n \geq 0$$

A inequação $\sum_{j=1}^{n} a_{ij}X_j \geq b_i$ é a i-ésima restrição e os coeficientes a_{ij}, com $i = 1, 2, \cdots, m$ são chamadas de restrições técnicas e os b_i representam os requisitos mínimos a serem satisfeitos. O conjunto dos valores de $X_1, X_2, \cdots, X_n$ que satisfazem as restrições é chamado de uma solução viável.

Quando em um modelo de PL a natureza das variáveis de decisão é inteira, ou seja, quando $X_i \in \mathbb{Z}$, para $i = 1, 2, \cdots, n$ e obedecendo a condição de não-negatividade, ele é dito ser um modelo de Programação Linear Inteira (PLI).

Como discutimos anteriormente para cada tipo de grafo, em relação a orientação das ligações, podemos encontrar várias formas de resolvê-lo. Bem como, encontramos várias formulações de modelos de Programação Linear Inteira a depender o tipo de orientação das ligações dos grafos.

3.2.1 Modelo para resolução do PCC em grafos orientados

Os autores (AHUJA; MAGNANTI; ORLIN, 1993) apresentam a seguinte forma de resolver o Problema do Carteiro Chinês, para o caso orientado, considerando um grafo[1] $G = (N, A)$ f-conexo:

> Em um percurso ótimo, um carteiro pode atravessar arcos mais de uma vez. O percurso de comprimento mínimo minimiza a soma dos comprimentos dos arcos atravessados repetidamente. Vamos denotar por X_{ij} o número de vezes que o carteiro atravessa o arco (i, j) em um percurso. Qualquer percurso do carteiro deve satisfazer as seguintes condições:
>
> $$\sum_{j:(i,j)\in A} X_{ij} - \sum_{j:(j,i)\in A} X_{ji} = 0 \; \forall i \in N;$$
>
> $$X_{ij} \geq 1, \; \forall (i, j) \in A$$
>
> (AHUJA; MAGNANTI; ORLIN, 1993, p. 741, tradução nossa)[2]

Simbolicamente, esse trecho poderia ser reescrito da seguinte forma – adaptando a notação usada pelo autor e considerando $G = (V, A)$ f-conexo.

[1] A notação dos grafos nos trabalhos em inglês é $G = (N, A)$ e N é o conjunto de vértices, os autores da língua utilizam *node* para se referir aos vértices, e esse é o motivo dele utilizarem N ao invés de V.

[2] In an optimal walk, a postal carrier might traverse arcs more than once. The minimum length walk minimizes the sum of lengths of the repeated arc traversals. Let X_{ij} denote the number of times the postal carrier traverses arc (i, j) in a walk. Any carrier walk must satisfy the following conditions:

$$\sum_{j:(i,j)\in A} X_{ij} - \sum_{j:(j,i)\in A} X_{ji} = 0 \; \forall i \in N;$$

$$X_{ij} \geq 1, \; \forall (i, j) \in A$$

(AHUJA; MAGNANTI; ORLIN, 1993, p. 741)

Modelo 3.1.

$$\min \sum_{(i,j)\in A} C_{ij}X_{ij} \tag{3.1}$$

Sujeitos as restrições:

$$\sum_{j:(i,j)\in A} X_{ij} - \sum_{j:(j,i)\in A} X_{ji} = 0,\ \forall i \in V; \tag{3.2}$$

$$X_{ij} \geq 1,\ \forall (i,j) \in A \tag{3.3}$$

$$X_{ij} \text{ é inteiro não-negativo}, \forall (i,j) \in A \tag{3.4}$$

Onde, C_{ij} é o custo de um arco (i,j).

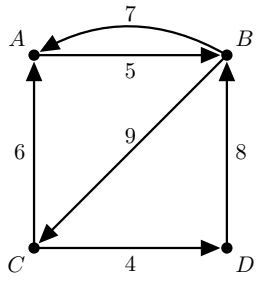
A expressão 3.1 é somente uma transcrição do trecho citado anteriormente e a restrição 3.3 nos indica que todas os arcos devem ser percorridos pelo menos uma vez. A condição 3.4 força todas as variáveis a serem inteiras não-negativas.

Ao fim da aplicação do modelo 3.1 a um grafo orientado G podemos construir um outro grafo G', que possui os mesmos vértices de G e todos seus arcos, mas algumas deles são replicados. Além disso, G' deve ser euleriano.

O que determina a quantidade de réplicas de um arco (i,j) em G' é o valor da variável X_{ij}, ou seja, se $X_{ij} = k$, para algum $k \in \mathbb{N}$, podemos interpretar isso como a quantidade k de vezes que o arco foi atravessado ou que o grafo construído G' possui k cópias de (i,j).

Nesse grafo é que a restrição 3.2 assume todo o seu potencial. Pois, como afirmamos anteriormente G' deve ser euleriano e como ele é orientado o teorema 2.2 deve ser válido, isso é, ele deve ser pseudo-simétrico e é essa restrição que faz com que a quantidade de arcos incidindo interiormente em um vértice i seja igual a quantidade de arcos incidindo exteriormente no mesmo vértice. Além do que, a restrição 3.3 garante a f-conexidade em G'.

Exemplo 3.1. Vamos resolver um PCC para o seguinte grafo $G = (V,A)$ orientado:



Perceba que G não é euleriano porque não se verifica o teorema 2.2, pois, por exemplo, $d^+(C) = 2 \neq 1 = d^-(C)$ e G não é pseudo-simétrico. Com isso, vamos utilizar o modelo 3.1 para resolver o PCC nesse grafo.

$$\min \qquad z = 5X_{AB} + 7X_{BA} + 9X_{BC} + 4X_{CD} + 8X_{DB} + 6X_{CA}$$

Sujeito as restrições:

$$X_{AB} - X_{BA} - X_{CA} = 0$$
$$X_{BC} + X_{BA} - X_{AB} - X_{DB} = 0$$
$$X_{CD} + X_{CA} - X_{BC} = 0$$
$$X_{DB} - X_{CD} = 0$$

$$X_{AB} \geq 1$$
$$X_{BA} \geq 1$$
$$X_{BC} \geq 1$$
$$X_{CD} \geq 1$$
$$X_{DB} \geq 1$$
$$X_{CA} \geq 1$$

X_{AB}, X_{BA}, X_{BC}, X_{CD}, X_{DB}, X_{CA} são inteiros não-negativos.

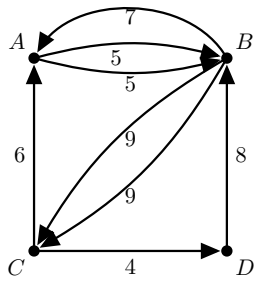
O pensamento que se aplica a restrição $X_{AB} - X_{BA} - X_{CA} = 0$ é o mesmo que se aplica as restrições $X_{BC} + X_{BA} - X_{AB} - X_{DB} = 0$; $X_{CD} + X_{CA} - X_{BC} = 0$; $X_{DB} - X_{CD} = 0$. E ele é: Fixado o vértice A igualar a diferença entre as variáveis X_{AB} e X_{BA} e X_{CA} a 0 determina que no grafo gerado G' os arcos incindo sobre A ser tais que $d^+(A) = d^-(A)$, isso é, os arcos (A,B), (B,A) e (C,A) devem ser replicados, ou não, de forma que a quantidade de arcos incindindo interiormente em A seja a mesma que a quantidade de arcos incidindo exteriormente.

Para resolver os problemas de PLI que surgirem nesse trabalho utilizaremos um software chamado LINDO (Ver Apêndice A), em sua versão 6.1. Este software utiliza uma linguagem bem próxima a escrita não sendo preciso fazer grandes alterações no modelo como foi apresentado aqui. O Lindo também nos fornece o custo total de um percurso euleriano nesse grafo auxiliar.

De posse dos resultados obtidos no LINDO construímos a seguinte tabela:

Valor	Variáveis
1	X_{BA} X_{CD} X_{DB} X_{CA}
2	X_{AB} X_{BC}

O custo resultante da aplicação desse modelo de PLI no Lindo é: 53. Observe na tabela que os valores presentes na coluna valor são a quantidade de vezes que o respectivo arco aparecerá num grafo auxiliar G' a seguir, com os respectivos custos.



Nesse grafo auxiliar aplicaremos o algoritmo de Hierholzer para encontrar um percurso euleriano. E um circuito encontrado é:

$$P_1 = (A,B,A,B,C,D,B,C,A)$$

3.2.2 Modelo para a resolução do PCC em grafos mistos

Dado um Grafo misto $G = (V,\ E \cup A)$. Vamos apresentar um modelo de PLI que se encontra exposto em (SHERAFAT, 2004).

Modelo 3.2. Considere $\Gamma(i)$ como o conjunto dos vértices adjacentes ao vértice i. Seja X_{ij} uma variável inteira indicando quantas vezes a ligação (i, j) é percorrida de i a j numa solução ótima do PCC em um grafo misto. Então, o problema pode ser formulado como:

$$min \sum_{(i,j)\in E} C_{ij}\left(X_{ij} + X_{ji}\right) + \sum_{(i,j)\in A} C_{ij}X_{ij} \tag{3.5}$$

Sujeito as restrições:

$$\sum_{j\in\Gamma(i)} \left(X_{ij} - X_{ji}\right) = 0,\ \forall i \in V \tag{3.6}$$

$$X_{ij} + X_{ji} \geq 1,\ \forall (i,j) \in E \tag{3.7}$$

$$X_{ij} \geq 1,\ \forall (i,j) \in A \tag{3.8}$$

$$X_{ij} \text{ inteiro não-negativo},\ \forall (i,j) \in A \cup E \tag{3.9}$$

Observação 3.2. A aplicação desse modelo acaba por criar um grafo auxiliar G' o qual será euleriano. Ainda é interessante notar que o modelo 3.2 exige, na função objetivo, que uma aresta seja substituída por dois arcos em sentidos opostos o grafo auxiliar resultante terá o sentido em que devemos percorrer cada aresta.

Sobre esse modelo, podemos interpretar C_{ij} como o custo de (i, j) e X_{ij} como a variável de tomada de decisão; a equação 3.5 como a função objetivo; a equação 3.6, como uma restrição

equivalente a uma condição de pseudo-simetria para G'; a inequação 3.7 aliada a condição 3.9 pode ser interpretada como a condição que impede que uma aresta não seja percorrida, ou seja, cada aresta será percorrida pelo menos uma vez; a inequação 3.8 nos obriga a percorre cada arco de G pelo menos uma vez, ou ainda que o grafo auxiliar G' possua pelo menos todos os arcos de G; e finalmente a condição 3.9 é uma consequência do fato de não termos meia ligação, isto é, em um circuito que utiliza certa ligação devemos percorrê-la completamente e essa condição é a condição que torna o modelo um modelo de PLI.

Antes de prosseguirmos (SHERAFAT, 2004) apresenta o seguinte resultado, atribuindo-o a Minieka e não o demonstra, sobre a solução de um PCC. Também, vamos apresentar esse resultado sem demonstração.

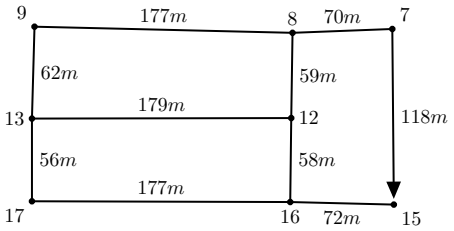
Teorema 3.1. Em qualquer solução ótima do PCC em grafos mistos, apenas um dos três seguintes casos ocorre para qualquer aresta (v_i, v_j):

(i) $X_{ij} = 0$ e $X_{ji} \geq 1$;

(ii) $X_{ji} = 0$ e $X_{ij} \geq 1$;

(iii) $X_{ij} = X_{ji} = 1$

Exemplo 3.2. Considere o seguinte grafo $G = (V, E)$ apresentado no exemplo 1.14:



Note que G não é euleriano, pois nele não se verifica o item (b) do teorema de Ford e Fulkerson (teorema 2.3), porque $d(13) = d(12) = 3$ não é par. Logo, para encontrar um circuito de custo mínimo podemos utilizar o modelo 3.2 da seguinte forma:

$$\begin{aligned} \textit{min} \qquad & z = 70(X_{78} + X_{87}) + 177(X_{89} + X_{98}) + 59(X_{812} + X_{128}) + 177(X_{913} + X_{139}) + \\ & 179(X_{1213} + X_{1312}) + 58(X_{1216} + X_{1612}) + 56(X_{1317} + X_{1713}) + 72(X_{1516} + X_{1615}) + \\ & 177(X_{1617} + X_{171}6) + 118X_{715} \end{aligned}$$

Sujeito as restrições:

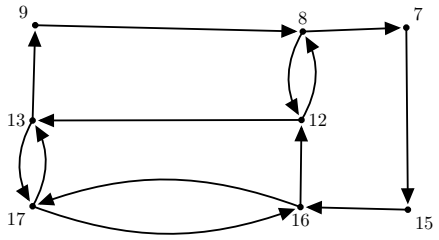
$$\begin{aligned} & X_{715} + X_{78} - X_{87} = 0 \\ & X_{87} - X_{78} + X_{89} - X_{98} + X_{812} - X_{128} = 0 \\ & X_{98} - X_{89} + X_{913} - X_{139} = 0 \\ & X_{128} - X_{812} + X_{1213} - X_{1312} + X_{1216} - X_{1612} = 0 \\ & X_{139} - X_{913} + X_{1312} - X_{1213} + X_{1317} - X_{1713} = 0 \end{aligned}$$

$X_{1516} - X_{1615} - X_{715} = 0$
$X_{1612} - X_{1216} + X_{1615} - X_{1516} + X_{1617} - X_{1716} = 0$
$X_{1713} - X_{1317} + X_{1716} - X_{1617} = 0$
$X_{78} + X_{87} \geq 1$
$X_{89} + X_{98} \geq 1$
$X_{812} + X_{128} \geq 1$
$X_{913} + X_{139} \geq 1$
$X_{1213} + X_{1312} \geq 1$
$X_{1216} + X_{1612} \geq 1$
$X_{1317} + X_{1713} \geq 1$
$X_{1516} + X_{1615} \geq 1$
$X_{1617} + X_{1716} \geq 1$
$X_{715} \geq 1$
X_{ij} é inteiro, não-negativo, para qualquer $(i, j) \in E \cup A$.

Após a aplicação no software LINDO 6.1 obtemos $1320m$ como valor mínimo para a função objetivo obedecendo as restrições. Também, obtemos a quantidade de vezes e o sentido que cada ligação é percorrida:

valor	variáveis
0	$X_{78}\ X_{89}\ X_{913}\ X_{1312}\ X_{1216}\ X_{1615}$
1	$X_{87}\ X_{98}\ X_{812}\ X_{128}\ X_{139}\ X_{1213}\ X_{1612}\ X_{1317}\ X_{1713}\ X_{1516}\ X_{1617}\ X_{1716}\ X_{715}$

E a partir dos resultados obtidos construímos o grafo auxiliar G'.



Note que cada arco em G' possui o mesmo valor atribuído as arestas em G que incidem sobre os mesmos vértices. Também, é fácil de ver que G' é pseudo-simétrico e conexo, com isso se verificam as condições do teorema 2.2, logo G' é euleriano. E ao aplicar o algoritmo de Hierholezer encontramos o seguinte circuito euleriano.

$$P_9 = (9, 8, 7, 15, 16, 12, 8, 12, 13, 17, 16, 17, 13, 9)$$

Essa solução, apesar de tudo, poderia ser melhorada a depender do problema que está sendo modelado. Por exemplo, se tentamos minimizar a rota de um carro de som que passe por essa região, seria desnecessário percorrer os arcos $(8,12)$ e $(12,8)$, por exemplo, pois seus comprimentos são pequenos e bastaria percorrer os arcos $(9,8)$ e $(8,7)$ que o som seria ouvido das casas da rua representada pelos primeiros arcos. Mas isso, renderia outro problema que seria o de encontrar um ciclo de custo mínimo sobre um conjunto L de ligações, e esse, mesmo que seja derivado de um PCC, não é o objetivo desse trabalho, e ainda teríamos que considerar a possibilidade desse ciclo mínimo ser justamente o circuito encontrado no problema.

4 ALGUNS EXEMPLOS DE APLICAÇÕES

Nesse capítulo vamos visualizar como seria a aplicação de um Problema do Carteiro Chinês na cidade de Vitória da Conquista. Estudaremos, através de exemplos **fictícios** como seria a determinação da coleta de lixo em determinados locais (setores) escolhidos de forma a representar algumas situações que consideramos interessantes, como por exemplo, ter somente ruas de mão dupla, ou ruas que podem ser consideradas de mão única e possuir ruas em ambas condições. Também, vamos mostrar como seria a construção de um modelo de grafos para determinado setor.

Ressaltamos que **não** levaremos em consideração alguns aspectos, típicos do problema de determinação da rota, para definir os setores sobre os quais aplicaremos o PCC, como por exemplo, a quantidade de lixo produzido no setor, a capacidade de armazenamento do caminhão compactador, sua autonomia (em relação ao combustível) e paradas para tratar das necessidades específicas do trabalho, tempo de coleta, dentre outras. Os problemas apresentados nesse capítulo são apenas para ilustrar algumas implicações que surgem ao determinar o modelo de grafo associado ao setor. Estaremos interessados em, somente, encontrar um percurso mínimo no setor escolhido. Como consequência da forma escolhida para abordar o problema, as ligações do grafo associado serão valoradas considerando apenas o comprimento das ruas a elas associadas.

Esse comprimento será tomado utilizando o Google Maps, que possui uma ferramenta que permite medir a distância entre dois pontos. Obviamente, esses pontos serão marcados de forma, que levando em conta suas posições, a ferramenta nos forneça o comprimento de uma rua. Marcaremos os pontos no centro da intercessão entre duas ruas, não temos um motivo específico para a escolha desse método de marcação e apenas buscamos um padrão para determinar os comprimentos das ruas. Também, faremos uso de aproximações dos comprimentos para que este seja inteiro, faremos isso porque é mais provável que em mais de uma medição podemos encontrar aproximadamente o mesmo valor e, também, por razões estéticas, visto que os procedimentos de resolução do PCC não impedem o uso de números não inteiros para a valoração das ligações.

4.1 UM SETOR COM RUAS DE MÃO DUPLA

Vamos supor que um caminhão deva efetuar a coleta do lixo em um setor delimitado pelas ruas: R. João Gonçalves, R. São Roque, Tv. Princesa Isabel e Tv. José Gonçalves, no bairro Guarani. Veja a seguir, uma imagem que mostra o setor escolhido:

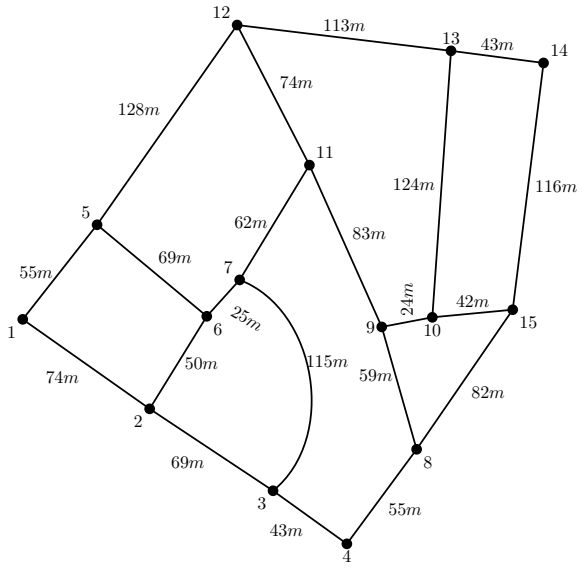
Figura 4 – Região do bairro Guarani. Fonte: *Google Maps*

Note que todas as ruas, no setor escolhidos, são de mão dupla. Com efeito, o grafo associado a essa região é não orientado, porque todas as ruas são de mão dupla, isso nos diz que uma rua pode ser percorrida tanto em um sentido quanto em outro e essa forma de percorrer a rua é mais semelhante a uma aresta do que a um arco. O grafo $G = (V, E)$ será construído tomando como vértices os cruzamentos entre as ruas (ou seja, a intercessão entre duas ou mais ruas definirão um vértice) e as ruas como arestas.

Ora, ainda pode ocorrer ao leitor o questionamento: “Por que não associar as ruas um par de arcos orientados em sentidos opostos?”. Bem, isso seria possível, mas tendo em vista se fizéssemos isso acabaríamos por aplicar o modelo 3.1 o qual nos obriga a percorrer uma mesma rua nos dois sentidos pelo menos uma vez, enquanto que ao aplicarmos o procedimento 3.1 provavelmente encontraremos algumas ruas que serão percorridas em apenas um sentido, como consequência disso o custo do circuito obtido no segundo seria menor que o obtido no primeiro.

4.1.1 Construindo o grafo Associado

Considerando o que foi mencionado anteriormente, sobre a determinação dos vértices e das arestas, o grafo associado a esse setor será:



Agora, vamos aplicar o procedimento 3.1 para encontrar um circuito de custo mínimo através desse grafo. Primeiramente, perceba que G não é euleriano, por exemplo, observe que $|\omega(3)| = |\{(2,3),(3,6),(3,4)\}| = 3$, isso é o grau do vértice 3 é ímpar e dessa forma o teorema 2.1 não se verifica. Assim, prosseguiremos com os demais passos do procedimento.

Do fato de G não ser euleriano, podemos determinar $T = \{2, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15\}$ o conjunto de todos os vértices de G que possuem grau ímpar. Com isso, ao aplicarmos o Algoritmo de Dijkstra e passando ao passo 4. do procedimento 3.1 construímos a matriz:

$$D(T) = \begin{array}{c} \begin{array}{cccccccccccc} 2 & 3 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 15 \end{array} \\ \left(\begin{array}{cccccccccccc} \infty & 69 & 119 & 50 & 75 & 279 & 220 & 244 & 137 & 211 & 324 & 266 \\ 69 & \infty & 188 & 119 & 115 & 319 & 260 & 284 & 177 & 251 & 364 & 326 \\ 119 & 188 & \infty & 69 & 94 & 298 & 239 & 263 & 156 & 128 & 241 & 305 \\ 50 & 119 & 69 & \infty & 25 & 229 & 170 & 194 & 87 & 161 & 274 & 236 \\ 75 & 115 & 94 & 25 & \infty & 204 & 145 & 169 & 62 & 136 & 249 & 211 \\ 279 & 319 & 298 & 229 & 204 & \infty & 59 & 83 & 142 & 216 & 207 & 82 \\ 220 & 260 & 239 & 170 & 145 & 59 & \infty & 24 & 83 & 157 & 141 & 66 \\ 244 & 284 & 263 & 194 & 169 & 83 & 24 & \infty & 107 & 190 & 303 & 42 \\ 137 & 177 & 156 & 87 & 62 & 142 & 83 & 107 & \infty & 74 & 187 & 149 \\ 211 & 251 & 128 & 161 & 136 & 216 & 157 & 190 & 74 & \infty & 113 & 223 \\ 324 & 364 & 241 & 274 & 249 & 207 & 141 & 303 & 187 & 113 & \infty & 166 \\ 266 & 326 & 305 & 236 & 211 & 82 & 66 & 42 & 149 & 223 & 166 & \infty \end{array}\right) \\ \begin{array}{c} 2 \\ 3 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 15 \end{array} \end{array}$$

Ao aplicarmos o Algoritmo Húngaro em $D(T)$ obtemos a seguinte matriz:

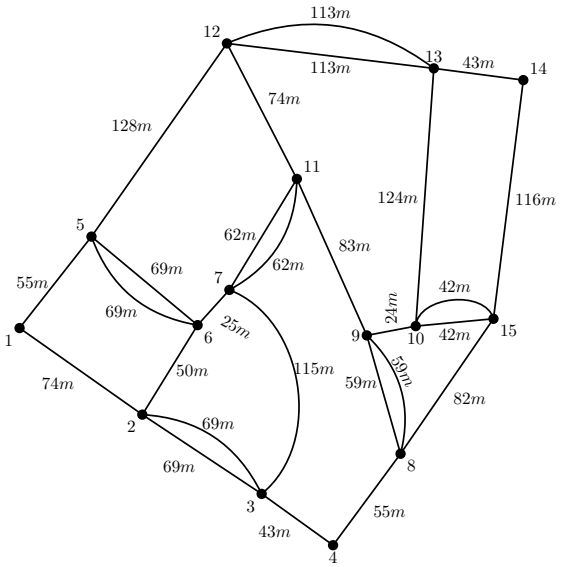
$$D'(T) = \begin{array}{c} \begin{array}{cccccccccccc} 2 & 3 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 & 13 & 15 \end{array} \\ \left(\begin{array}{cccccccccccc} \infty & \boxed{0} & 37 & 37 & 37 & 194 & 170 & 194 & 87 & 161 & 235 & 198 \\ \boxed{0} & \infty & 87 & 87 & 58 & 215 & 191 & 215 & 108 & 182 & 256 & 239 \\ 13 & 63 & \infty & \boxed{0} & 0 & 157 & 133 & 157 & 50 & 22 & 96 & 181 \\ 13 & 63 & \boxed{0} & \infty & 0 & 157 & 133 & 157 & 50 & 124 & 198 & 181 \\ 13 & 34 & 0 & 0 & \infty & 107 & 83 & 107 & \boxed{0} & 74 & 148 & 131 \\ 220 & 241 & 207 & 207 & 157 & \infty & \boxed{0} & 24 & 83 & 157 & 109 & 5 \\ 196 & 217 & 183 & 183 & 133 & \boxed{0} & \infty & 0 & 59 & 133 & 78 & 24 \\ 220 & 241 & 207 & 207 & 157 & 24 & 0 & \infty & 83 & 166 & 240 & \boxed{0} \\ 63 & 84 & 50 & 50 & \boxed{0} & 33 & 9 & 33 & \infty & 0 & 74 & 57 \\ 137 & 158 & 22 & 124 & 74 & 107 & 83 & 116 & 0 & \infty & \boxed{0} & 131 \\ 211 & 232 & 96 & 198 & 148 & 59 & 28 & 190 & 74 & \boxed{0} & \infty & 35 \\ 224 & 265 & 231 & 231 & 181 & 5 & 24 & \boxed{0} & 107 & 181 & 85 & \infty \end{array}\right) \end{array} \begin{array}{c} 2 \\ 3 \\ 5 \\ 6 \\ 7 \\ 8 \\ 9 \\ 10 \\ 11 \\ 12 \\ 13 \\ 15 \end{array}$$

Na matriz $D'(T)$, os zeros destacados são as alocações feitas pelo algoritmo. Note que as alocações estão disposta de forma simétrica nessa última matriz. Então, pela equivalência entre arcos e arestas e, como essas alocações têm mesmo custo é suficiente adicionar uma aresta incidindo sobre os mesmos vértices, ao invés de adicionarmos dois arcos, com sentidos opostos. Em resumo, teremos que incluir as seguintes arestas.

$$(2,3);\ (5,6);\ (7,11);\ (8,9);\ (10,15);\ (12,13)$$

E, os custos dessas arestas serão respectivamente: 69, 69, 62, 69, 59, 42 e 113.

Dito isso, com essas alocações podemos construir um grafo $G' = (V, E')$, no qual poderemos observar que todos os vértices terão grau par, com isso poderemos aplicar um algoritmo de busca. O grafo obtido é apresentado a seguir:



Note que esse grafo obtido através da aplicação do procedimento 3.1 atende as condições do Teorema de Euler (Teorema 2.1), pois ele é conexo e o grau de cada vértice nele é par, logo existe um circuito euleriano nesse grafo. Ainda, podemos afirmar que o custo total de qualquer circuito euleriano nesse grafo é:

$$C_f = \sum_{(i,j)\in E} C_{ij} + \sum_{(k,l)\in E'} C_{kl} = 1919m$$

E da aplicação do algoritmo de Hierholzer teremos o seguinte circuito:

$$P_4 = (4,8,15,14,13,10,15,10,9,8,9,11,7,3,2,1,5,12,13,12,11,7,6,5,6,2,3,4)$$

Esse circuito pode ser traduzido para um itinerário. Iniciando esse itinerário na esquina da Rua João Gonçalves com a R. São Roque, devemos seguir os seguintes passos:

1º passo Siga pela Rua João Gonçalves até a altura da Travessa José Gonçalves e vire à esquerda.

2º passo Prossiga na Tv. José Gonçalves e vire à esquerda na Tv. G. Estrela.

3º passo Vire à esquerda na R. Grimaldo Estrela e faça o retorno na esquina com a Rua João Gonçalves;

4º passo Volte pela R. Grimaldo Estrela até a R. Filinho Candeias

5º passo Vire à esquerda (na R. Filinho Candeias), desça até a R. José Gonçalves e faça o retorno;

6º passo Volte pela R. Filinho Candeias e vire à esquerda na R. Albatroz.

7º passo Seguindo pela R. Albatroz, vire à esquerda na Tv. São Roque e prossiga até a R. São Roque;

8º passo Vire à direita (na R. São Roque), prossiga e vire à direita na segunda rua (ou seja, no vértice 1).

9º passo siga em frente, entre Tv. Princesa Isabel e vire à direita na Tv. José Gonçalves.

10º passo Faça o retorno na esquina com a Tv. G. Estrela, volte até a T. Princesa Isabel e entre na R. Filinho Candeias.

11º passo Vire à direita na R. Albatroz e prossiga até virar à direita na Tv. Albatroz;

12º passo Siga em frente e faça o retorno na esquiva com a Tv. Princesa Isabel;

13º passo Volte e vire à direita na Rua Albatroz;

14º passo Desça até a esquina com a R. São Roque, Vire à esquerda e retorne ao ponto inicial.

4.2 UM SETOR COM RUAS DE MÃO ÚNICA

Em grandes, e médias, cidades os setores com, somente, ruas de mão única assim como os setores com apenas mão dupla são menos frequentes do que aqueles que possuem ruas de ambos os tipos. Para um exemplo de setor com apenas rua de mão única escolhemos uma região do centro da cidade que contém o Terminal de ônibus da Av. Lauro de Freitas. Esse setor será delimitado pelas ruas 2 de julho, Francisco Santos, Monsenhor Olímpio, pela Av. Lauro de Freitas e pela Tv. Ernestina Gusmão.

Figura 5 – Região do Centro de Vitória da Conquista. **Fonte:** Google Maps (Modificado)

4.2.1 Construindo o grafo associado a esse setor

O leitor perceberá que nem todas as ruas desse setor são necessariamente de mão única, por exemplo, a Av. Lauro de Freitas. Entretanto, como ela é uma avenida duplicada, – existe um canteiro central (onde ficam as plataformas de embarque e desembarque de passageiros) entre as pistas sobre, as quais os veículos transitam em sentidos opostos – pode, em certo grau, ser considerada com duas ruas de mão única.

Fato é que existe a possibilidade de um veículo da coleta de lixo percorrer essa via em apenas um sentido. E isso seria possível de duas formas. A primeira é se os comerciantes (pois essa é uma avenida comercial) de ambos os lados colocassem o lixo em um dos lados do canteiro central ou na calçada, da rua no sentido em que o veículo de coleta deve passar, em ambos os

casos teríamos problemas, pois o canteiro central é, como já foi mencionado, onde ficam as plataformas de embarque e desembarque, logo não é possível que se deposite lixo nesse local e ainda teria pessoas transitando nas calçadas (pois o lixo nessa região é coletado de segunda a sexta a partir das 19:00h (PMVC, 2017)), além do que, o volume do lixo pode ser um empecilho para transportá-lo de um lado para outro.

A segunda e menos provável, nos dias atuais, seria deixar para fazer a coleta de madrugada, para que o lixo fosse depositado na calça de um lado ou em determinados pontos do canteiro central, mas a coleta de lixo na cidade começa a partir das 7:00h (PMVC, 2017) e caso fosse possível fazê-la pela manhã, ela deveria terminar antes do horário em que os veículos do transporte coletivo começassem a circular pelo terminal, o que acontece por volta das 6:00h, portanto não seria viável (considerando os horários atuais de coleta) que o lixo fosse coletado nessa faixa horária. Por esses motivos, consideramos que um veículo de coleta deve percorrer a Av. Lauro de Freitas nos dois sentidos, tendo em vista que, se o lixo for depositado em ambos os lados da rua o seu volume total será diluído por toda a extensão da via permitindo que os pedestres transitem sem maiores complicações, e mesmo que isso não ocorra o veículo de coleta terá que percorrer essa avenida nos dois sentidos, considerando a estrutura do grafo associado ao setor.

Existe ainda a possibilidade de o leitor questionar a existência de pelo menos um trecho da Av. Lauro de Freitas que não possui grandes obstáculos. Por exemplo,o pequeno trecho entre a esquina com a Av. Régis Pacheco até primeira rotatória, indo no sentido das plataformas de embarque e desembarque. Mas, é justamente porque esse trecho é pequeno e sem grandes obstáculos que iremos associar a esse trecho um arco, no sentido da esquina com a Av. Régis Pacheco as plataformas. Faremos isso porque a Tv. Monsenhor Olímpio é de mão única e vai na direção da Av. Régis Pacheco e se considerarmos esse trecho como um somente arco no outro sentido, não teríamos a conexidade necessária para ter um circuito pré-euleriano.

Note que vamos associar a Tv. Ernestina Gusmão um arco que vai da Av. Lauro de Freitas para 2 de Julho. No caso da Lauro de Freitas, que possui um canteiro central, um veículo de coleta de lixo é obrigado a transitar nos dois sentidos da via, mas no caso da Tv. Ernestina Gusmão basta que as pessoas coloquem o lixo ou no canteiro central ou no outro lado da via, devido a forma como os estabelecimentos estão dispostos nessa travessa, com isso não há necessidade de associar a esse trecho dois arcos em sentidos opostos. Por uma questão estética e para aproveitar melhor a área da face das folhas iremos rotacionar a figura 5 em 90°.

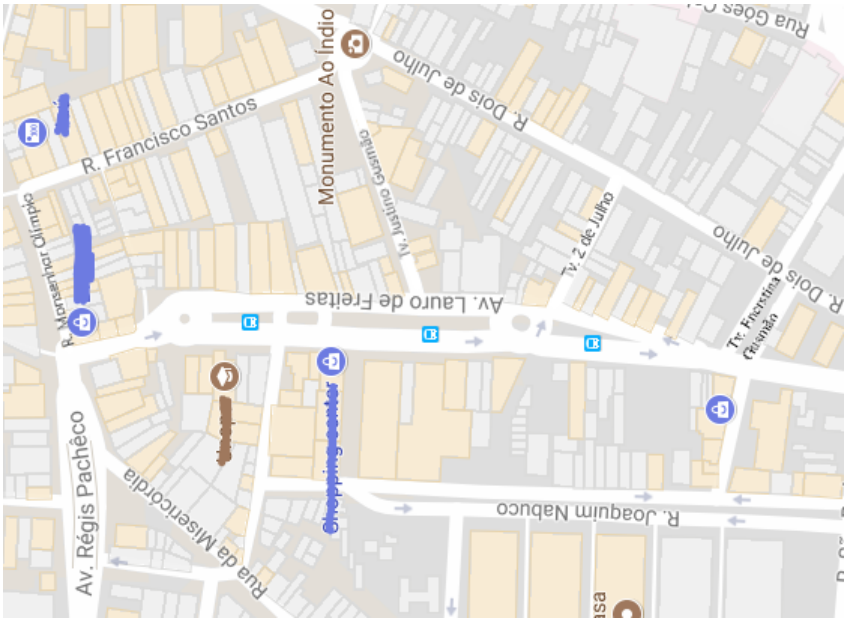
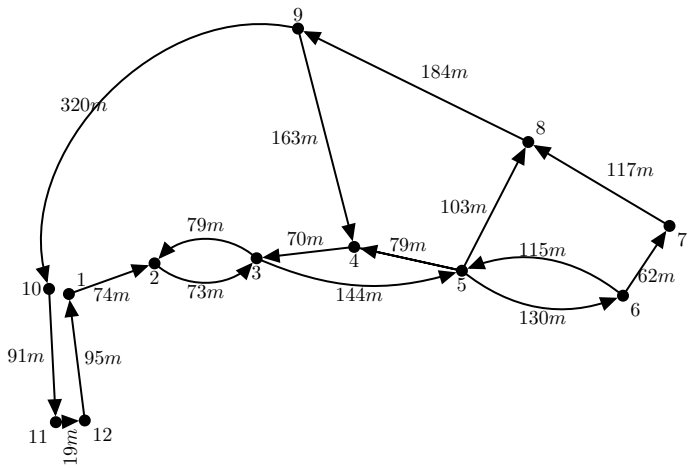


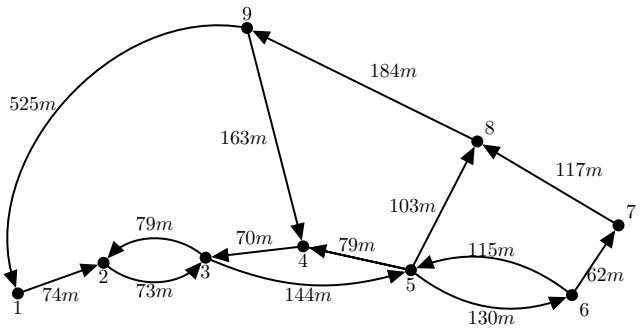
Figura 6 – Figura 5 rotacionada. **Fonte:** Google Maps (Modificado)

Dessa forma o grafo G_1 associado a esse setor será:



4.2.2 Resolvendo o problema

Perceba que o percurso $P_{91} = (9,\ 10,\ 11,\ 12,\ 1)$ pertence a G_1, pois não se pode passar diretamente da rua Monsenhor Olímpio para a Lauro de Freitas. Também, $P_{9\ 1}$ pode ser reduzido a apenas um arco indo de 9 até 1 através da contração sucessiva de vértices de forma que esse novo arco tenha custo $525m$ que é, justamente, a soma dos custos dos arcos de P_{91}. Como consequência da contração dos vértices teremos um grafo G e será sobre esse que resolveremos nosso PCC. Ressaltamos que isso não é feito somente por questões estéticas, mas também para simplificar o modelo de PLI, utilizado para esse caso (que é o modelo 3.1).



Utilizamos apenas 1 para rotular o vértice resultante da contração ao invés de 1011121 para simplificar a notação no modelo de PLI cuja função objetivo será:

$$\min \quad z = 74X_{12} + 73X_{23} + 79X_{32} + 144X_{35} + 70X_{43} + 79X_{54} + 130X_{56} + 103X_{58} + 11X_{65} + 62X_{67} + 117X_{78} + 184X_{89} + 163X_{94} + 525X_{91}$$

Que está sujeita as seguintes restrições:

$$X_{12} - X_{91} = 0$$
$$X_{23} - X_{12} - X_{32} = 0$$
$$X_{32} + X_{35} - X_{23} - X_{43} = 0$$
$$X_{43} - X_{54} - X_{94} = 0$$
$$X_{54} + X_{56} + X_{58} - X_{65} - X_{35} = 0$$
$$X_{65} + X_{67} - X_{56} = 0$$
$$X_{78} - X_{67} = 0$$
$$X_{89} - X_{58} - X_{78} = 0$$
$$X_{91} + X_{94} - X_{89} = 0$$
$$X_{12} \geq 1$$
$$X_{23} \geq 1$$
$$X_{32} \geq 1$$
$$X_{35} \geq 1$$
$$X_{43} \geq 1$$
$$X_{54} \geq 1$$
$$X_{56} \geq 1$$
$$X_{58} \geq 1$$
$$X_{65} \geq 1$$
$$X_{67} \geq 1$$
$$X_{78} \geq 1$$
$$X_{89} \geq 1$$
$$X_{94} \geq 1$$
$$X_{91} \geq 1$$

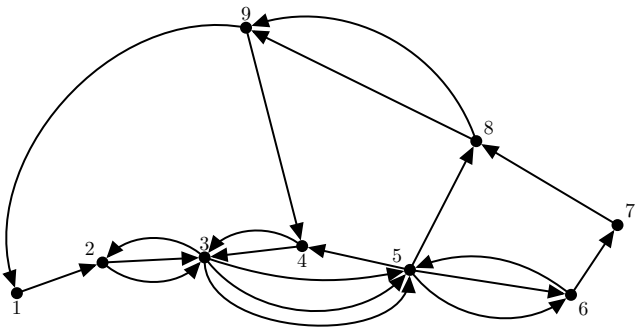
X_{12}, X_{23}, X_{32}, X_{35}, X_{43}, X_{54}, X_{56}, X_{58}, X_{65}, X_{67}, X_{78}, X_{89}, X_{94}, X_{91} inteiros não-negativos.

As restrições do tipo $X_{ij} \geq 1$ além de significarem que as respectivas variáveis terão valores maiores ou iguais a 1, também, podem ser entendidas pela quantidade de réplicas dos arcos (i, j) no grafo que pode ser construído após a aplicação do modelo. Podemos entender as restrições de igualdade de forma análoga ao exemplo 3.1. Isso é, fixando, por exemplo, o vértice 1 e observando todas as ligações que incidem nesse vértice, a restrição $X_{12} - X_{91} = 0$ implica que, no grafo gerado, a quantidade de réplicas de $(1,2)$ é a mesma que a de $(9,1)$.

Ao resolver esse modelo de PLI no LINDO 6.1 (após 10 iterações) obtemos $2663m$ como valor ótimo para z e também os seguintes valores para cada variável de decisão:

Valor	Variáveis
1	X_{12} X_{32} X_{54} X_{58} X_{65} X_{67} X_{78} X_{94} X_{91}
2	X_{23} X_{43} X_{56} X_{89}
3	X_{35}

Com esses valores podemos construir o seguinte grafo auxiliar G':



Esse grafo auxiliar é euleriano, pois se verifica facilmente o teorema 2.2. Agora, resta aplicar o algoritmo de Hierholzer para encontrar um circuito euleriano em G'. Com a aplicação desse algoritmo obtemos o seguinte circuito:

$$C_1 = (1,2,3,2,3,5,4,3,5,8,9,4,3,5,6,5,6,7,8,9,1)$$

E isso significa, em termos de itinerário, que partindo da esquina da Av. Régis Pacheco com a Av. Lauro de Freitas devemos seguir os seguintes passos:

1º passo Siga na Av. Lauro de Freitas até a segunda rotatória e faça o retorno.

2º passo Volte até a primeira rotatória faça o retorno, agora desça até a rotatória onde ficam os fiscais das empresas de ônibus e faça o retorno.

3º passo Volte, pela Av. Lauro de Freitas, até a rotatória onde ficam os banheiros públicos e faça o retorno.

4º passo Novamente, desça até a rotatória onde ficam os ficais e entre na Tv. 2 de Julho.

5º passo Vire à esquerda na R. 2 de Julho e siga até o Monumento ao Índio.

6º passo Entre na Tv. Justino Gusmão e siga até virar à esquerda na Av. Lauro de Freitas.

7º passo Mais uma vez, siga até fazer o retorno na rotatória onde ficam os banheiros.

8º passo Prossiga até a esquina com a Tv. Ernestina Gusmão e encontre uma forma de fazer o retorno e volte até a rotatória onde ficam os fiscais.

9º passo Faça o retorno.

10º passo Siga até virar à esquerda na Tv. Ernestina Gusmão.

11º passo Vire à esquerda na R. 2 de Julho e siga até o Monumento ao Índio.

12º passo Entre na R. Francisco Santos e depois na R. Monsenhor Olímpio.

13º passo Siga pela R. Monsenhor Olímpio atá fazer o retorno no elevado, conhecido popularmente por Bigode de Pedral. E, finalmente, volte para o ponto inicial.

4.3 UM SETOR COM RUAS DE MÃO ÚNICA E MÃO DUPLA

Apesar de setores com ruas de mão única e de mão dupla constituem a maioria dos setores em grandes e médias cidades. Escolhemos o Loteamento Vila Marina, pois é um local com poucas ruas e com ele não teremos problemas em trabalhar no Lindo 6.1 (que possui uma limitação de 50 variáveis inteiras). Este será, portanto, outro exemplo fictício.

Vila Marina é um loteamento do bairro Felícia, com acesso a Av. Juracy Magalhães próximo ao anel viário e ao Shopping Conquista Sul. Veja a seguir, uma imagem do setor, obtidas através do Google Maps.

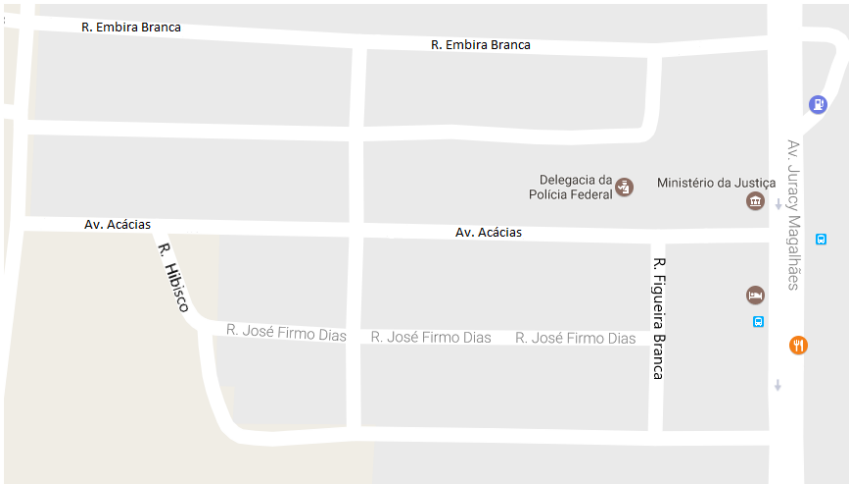
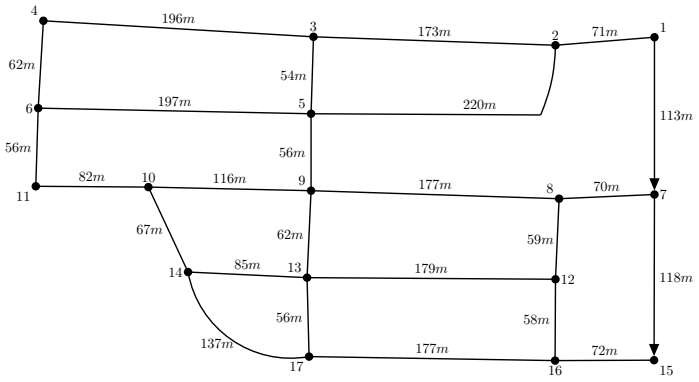


Figura 7 – Lot. Vila Marina. **Fonte:** Google Maps (Modificado)

Agora, podemos construir um grafo $G = (V, E \cup A)$ associado ao setor, sob o qual encontraremos um circuito de custo mínimo que utilize todas as ligações. Fazer isso, é equivalente a resolver um PCC em um grafo misto. A seguir, apresentamos o grafo G, que assim como em todos os exemplos anteriores será valorado sobre as arestas.



4.3.1 Resolvendo o PCC

Ao observarmos o grafo associado ao Loteamento Vila Marina, percebemos que ele não é euleriano. Pois, aplicando o teorema 2.3, ao grafo do loteamento vemos que, apesar de ser conexo ele apresenta vértices de grau ímpar, por exemplo, $|\omega(2)| = |\{(1,2),(2,3),(2,5)\}| = d(2) = 3$, assim o item (a) do teorema, anteriormente referido, não é satisfeito. Com isso, ao invés de tentar encontrar um circuito euleriano em G procuraremos por um circuito pré-euleriano, isto é, devemos obter um grafo auxiliar G' e nele encontrar um circuito euleriano e este será o menor circuito pré-euleriano em G.

Dito isso, devemos aplicar o modelo 3.2. Assim, para o grafo associado ao setor estudado temos o seguinte modelo de Programação Linear Inteira:

$$\begin{aligned}
\textit{min} \qquad z = {} & 71X_{1\,2} + 71X_{2\,1} + 173X_{2\,3} + 173X_{3\,2} + 196X_{3\,4} + 196X_{4\,3} + 62X_{4\,6} + \\
& 62X_{6\,4} + 220X_{5\,2} + 220X_{2\,5} + 54X_{5\,3} + 54X_{3\,5} + 197X_{5\,6} + 197X_{6\,5} + 56X_{6\,11} + 56X_{11\,6} + \\
& 82X_{10\,11} + 82X_{11\,10} + 116X_{10\,9} + 116X_{9\,10} + 56X_{9\,5} + 56_{X}5\,9 + 177X_{8\,9} + 177X_{9\,8} + \\
& 70X_{7\,8} + 70X_{8\,7} + 72X_{15\,16} + 72X_{16\,15} + 58X_{16\,12} + 58X_{12\,16} + 59X_{12\,8} + 59X_{8\,12} + \\
& 177X_{16\,17} + 177X_{17\,16} + 56X_{13\,17} + 56X_{17\,13} + 62X_{13\,9} + 62X_{9\,13} + 179X_{13\,12} + 179X_{12\,13} + \\
& 85X_{13\,14} + 85X_{14\,13} + 137X_{14\,17} + 137X_{17\,14} + 67X_{10\,14} + 67X_{14\,10} + 113X_{1\,7} + 118X_{7\,15}
\end{aligned}$$

Sujeito as restrições:

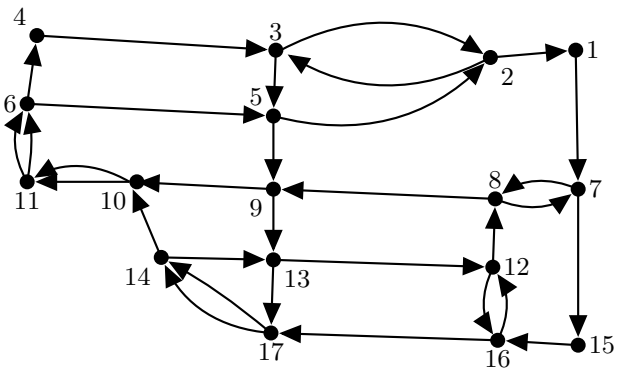
$$\begin{aligned}
& X_{1\,2} + X_{1\,7} - X_{2\,1} = 0 \\
& X_{2\,1} + X_{2\,5} + X_{2\,3} - X_{1\,2} - X_{5\,2} - X_{3\,2} = 0 \\
& X_{3\,2} + X_{3\,4} + X_{3\,5} - X_{2\,3} - X_{4\,3} - X_{5\,3} = 0 \\
& X_{4\,3} + X_{4\,6} - X_{3\,4} - X_{6\,4} = 0 \\
& X_{5\,2} + X_{5\,3} + X_{5\,6} + X_{5\,9} - X_{2\,5} - X_{3\,5} - X_{6\,5} - X_{9\,5} = 0 \\
& X_{6\,5} + X_{6\,4} + X_{6\,11} - X_{4\,6} - X_{5\,6} - X_{11\,6} = 0 \\
& X_{7\,8} - X_{8\,7} + X_{7\,15} - X_{1\,7} = 0 \\
& X_{8\,7} + X_{8\,9} + X_{8\,12} - X_{7\,8} - X_{9\,8} - X_{12\,8} = 0 \\
& X_{9\,5} + X_{9\,8} + X_{9\,10} + X_{9\,13} - X_{5\,9} - X_{8\,9} - X_{10\,9} - X_{13\,9} = 0 \\
& X_{10\,9} + X_{10\,11} + X_{10\,14} - X_{9\,10} - X_{11\,10} - X_{14\,10} = 0 \\
& X_{11\,6} + X_{11\,10} - X_{6\,11} - X_{10\,11} = 0 \\
& X_{12\,8} + X_{12\,13} + X_{12\,16} - X_{8\,12} - X_{13\,12} - X_{16\,12} = 0 \\
& X_{13\,9} + X_{13\,12} + X_{13\,14} + X_{13\,17} - X_{9\,13} - X_{12\,13} - X_{14\,13} - X_{17\,13} = 0 \\
& X_{14\,10} + X_{14\,13} + X_{14\,17} - X_{10\,14} - X_{13\,14} - X_{17\,14} = 0 \\
& X_{15\,16} - X_{16\,15} - X_{7\,15} = 0 \\
& X_{16\,12} + X_{16\,15} + X_{16\,17} - X_{12\,16} - X_{15\,16} - X_{17\,16} = 0 \\
& X_{17\,13} + X_{17\,14} + X_{17\,16} - X_{13\,17} - X_{14\,17} - X_{16\,17} = 0 \\
& X_{1\,2} + X_{2\,1} \geq 1 \\
& X_{2\,3} + X_{3\,2} \geq 1 \\
& X_{2\,5} + X_{5\,2} \geq 1 \\
& X_{3\,5} + X_{5\,3} \geq 1 \\
& X_{3\,4} + X_{4\,3} \geq 1 \\
& X_{4\,6} + X_{6\,4} \geq 1 \\
& X_{5\,6} + X_{6\,5} \geq 1 \\
& X_{6\,11} + X_{11\,6} \geq 1 \\
& X_{10\,11} + X_{11\,10} \geq 1 \\
& X_{10\,14} + X_{14\,10} \geq 1 \\
& X_{9\,10} + X_{10\,9} \geq 1 \\
& X_{5\,9} + X_{9\,5} \geq 1 \\
& X_{8\,9} + X_{9\,8} \geq 1 \\
& X_{7\,8} + X_{8\,7} \geq 1
\end{aligned}$$

$X_{8\ 12} + X_{12\ 8} \geq 1$
$X_{12\ 13} + X_{13\ 12} \geq 1$
$X_{9\ 13} + X_{13\ 9} \geq 1$
$X_{13\ 14} + X_{14\ 13} \geq 1$
$X_{14\ 17} + X_{17\ 14} \geq 1$
$X_{13\ 17} + X_{17\ 13} \geq 1$
$X_{16\ 17} + X_{17\ 16} \geq 1$
$X_{12\ 16} + X_{16\ 12} \geq 1$
$X_{15\ 16} + X_{16\ 15} \geq 1$
$X_{1\ 7} \geq 1$
$X_{7\ 15} \geq 1$
$\forall X_{i\ j} \in \mathbb{Z}$, onde X_{ij} é o custo correspondente a ligação (i, j).

Para resolver o modelo de PLI, exibido acima utilizamos o Software Lindo 6.1. O valor mínimo, encontrado após 9388 iterações, da função objetivo é $3289m$, ou $3,289Km$. O grafo auxiliar G' será obtido a partir da interpretação da tabela a seguir.

Valor	Variáveis
0	$X_{1\ 2}\ X_{2\ 5}\ X_{5\ 3}\ X_{3\ 4}\ X_{4\ 6}\ X_{5\ 6}\ X_{11\ 10}\ X_{10\ 14}\ X_{10\ 9}\ X_{9\ 5}\ X_{9\ 8}$ $X_{8\ 12}\ X_{12\ 13}\ X_{13\ 9}\ X_{13\ 14}\ X_{14\ 17}\ X_{17\ 13}\ X_{17\ 16}\ X_{16\ 15}$
1	$X_{2\ 1}\ X_{2\ 3}\ X_{3\ 2}\ X_{5\ 2}\ X_{3\ 5}\ X_{4\ 3}\ X_{6\ 4}\ X_{6\ 5}\ X_{14\ 10}\ X_{9\ 10}\ X_{5\ 9}\ X_{8\ 9}$ $X_{7\ 8}\ X_{8\ 7}\ X_{12\ 8}\ X_{13\ 12}\ X_{9\ 13}\ X_{14\ 13}\ X_{13\ 17}\ X_{16\ 17}\ X_{12\ 16}\ X_{16\ 12}$ $X_{15\ 16}\ X_{1\ 7}\ X_{7\ 15}$
2	$X_{11\ 6}\ X_{10\ 11}\ X_{17\ 14}$

Observe que o valor da variável na tabela define a quantidade de arcos que existirá entre dois vértices no grafo auxiliar, ou seja, quando o valor da variável $X_{i\ j} = 0$ então, no grafo auxiliar, não haverá um arco (i, j), mas se $X_{k\ l} = 1$ teremos um arco indo de k para l. Com os resultados obtidos na execução do Lindo 6.1 construímos o seguinte grafo auxiliar.



Sobre esse grafo auxiliar podemos afirmar que ele é euleriano, pois se verifica o teorema 2.2. Agora, ao aplicar o algoritmo de Hierholzer teremos:

$$C_1 = (1,7,8,9,13,12,8,7,15,16,12,16,17,14,13,17,14,10,11,6,5,9,10,11,6,4,3,5,2,3,2,1)$$

Esse circuito pode ser traduzido para o seguinte itinerário, iniciado na esquina da R. Embira com a Av. Juracy Magalhães.

1º passo Do ponto inicial escolhido siga pela Av. Juracy Magalhães e vire à direita na Av. Acácias.

2º passo Siga por essa avenida entre na segunda rua à esquerda.

3º passo Depois vire à esquerda na R. José Firmo Dias.

4º passo Siga até o final dessa rua e vire à esquerda na R. Figueira Branca e logo em seguida vire à direita na Av. Acácias.

4º passo Siga nessa rua e vire à direita na Av. Juracy Magalhães.

5º passo Prossiga na Av. Juracy Magalhães e entre na primeira rua à direita, vire à direita, na R. Figueira Branca, e faça o retorno na próxima esquina.

6º passo Voltando, quando chegar no fim da R. Figueira Branca vire à direita e siga adiante até o final da rua, lá vire à direita e entre na R. José Firmo Dias.

7º passo Seguindo pela R. José Firmo Dias entre na primeira rua à direita e então vire novamente a direita.

8º passo Prossiga até o final dessa rua e vire à direita.

9º passo Siga pela R. Hibisco até virar à esquerda na Av. Acácias.

10º passo No fim da Av. Acácias vire à direita e entre na primeira rua à direita.

10º passo Siga em frente e entre na primeira rua à direita.

11º passo Prossiga até a Av. Acácias e lá vire à direita.

12º passo Siga na Av. Acácias até o seu final e, mais uma vez, vire à direita.

13º passo Siga nessa rua e vire à direita na R. Embira Branca.

14º passo Seguindo na R. Embira Branca vire à direita na primeira rua.

15º passo Entre na primeira rua à esquerda e vá em frente até virar à esquerda na R. Embira Branca.

16º passo Siga em frente e faça um retorno na primeira esquina e retorne para o ponto inicial.

CONSIDERAÇÕES FINAIS

Os grafos podem ser utilizados para modelar uma infinidade de coisas e nesse trabalho recorremos a eles para modelar regiões de uma cidade. Nos servimos do fato de podermos identificar uma cadeia de vértices e ligações num grafo e de ele admitir valoração sobre essas ligações para descrever setores e, o mais importante, empregamos alguns resultados e algoritmos da Teoria dos Grafos para auxiliar na resolução do problema de encontrar um circuito de custo mínimo.

Contudo, sabemos que as soluções para os problemas apresentados pelos métodos exibidos nesse trabalho são, em sua maioria, inviáveis, visto que quando a coleta de lixo é feita por um caminhão, o mesmo não pode fazer certas manobras que são exigidas nos circuitos dados como soluções, por exemplo, fazer retornos em cruzamentos de ruas estreitas. Isso faz com que várias modificações devam ser feitas nas soluções. Assim, sugerimos que uma análise cuidadosa de viabilidade das soluções seja feita em um trabalho futuro. Também, seria interessante fazer que essas análises viessem acompanhadas de um estudo de caso.

Ao refletir sobre as modificações que devem ser feitas nas soluções para torná-las viáveis e ao observar como é feita a coleta de lixo em certos setores da cidade, pensamos que uma alternativa, para viabilizar uma solução para o PCC, seria encontrar um circuito no grafo que não utilizasse todas as ligações. Para isso, devemos tomar um submulticonjunto (que na maioria dos casos é um subconjunto) de ligações que devem ser percorridas obrigatoriamente ou, equivalentemente, remover algumas ligações desnecessárias.

Todavia, não é fácil estabelecer critérios para remover uma ligação do modelo, por exemplo, não poderíamos tomar somente o valor de uma ligação (comprimento da rua) para determinar se devemos ou não retirá-la, pois pode existir alguma rua que seja representada por uma ligação de pouco valor, mas que o volume de lixo produzido nela seja muito maior que o de outra de maior valor. No entanto, uma vez conhecido o submulticonjunto de ligações que devem ser percorridas seria interessante, isso em um trabalho futuro, determinar critérios para existência de tal circuito e se esses tem custo menor que a solução para o respectivo PCC utilizando todas as ligações.

REFERÊNCIAS

AHUJA, R. K.; MAGNANTI, T. L.; ORLIN, J. B. Netwoks flows: theory, algorithms, and applications. In: ____. [S.l.]: Prentice-Hall, 1993. cap. Additional applications, p. 717 – 764.

BAZARAA, M. S.; JARVIS, J. J.; SHERALI, H. D. Linear programing and netwok flows. In: ____. 4. ed. New Jersey: Wiley, 2010. cap. Introdution, p. 1 – 43.

CARVALHO, M. A. M. . *BCC204 - Teoria dos Grafos*. 2017. Disponível em: <http://www.decom.ufop.br/marco/site_media/uploads/bcc204/16_aula_16.pdf>.

FILHO, M. G.; JUNQUEIRA, R. de A. R. Problema do carteiro chinês: escolha de métodos de solução e análise de tempos computacionais: escolha de métodos de solução e análise de tempos computacionais. *Produção*, v. 16, n. 3, p. 538 – 551, 2006.

FORD, J. L. R.; FULKERSON, D. R. Flows in netwoks. In: ____. [S.l.]: RAND Corporation, 1962. cap. Feasibility theorems and combinatorials applications.

KAPPAUF, C. H.; KOEHLER, G. J. The mixed postman problem. *Distcrete Applied Mathematics 1*, p. 89 – 103, 1979.

LOVÁSZ, L.; PELIKÁN, J.; VESZTERGOMBI, K. *Matemática Discreta*. 2. ed. Rio de Janeiro: SBM, 2013.

NETTO, P. O. B. *Grafos*: teoria, modelo, algoritmos. 5. ed. revista e ampliada. ed. São Paulo: Blucher, 2011.

PMVC, P. M. D. V. D. C. *Quadro de Coleta Torre*. 2017. Disponível em: <http://www.pmvc.ba.gov.br/utilidade-publica-confira-os-horarios-e-dias-de-coleta-de-lixo-em-seu-bairro/>.

ROSEN, K. H. *Matemática discreta e suas aplicações*. 6. ed. Porto Alegre: AMGH, 2010.

SHERAFAT, H. *Algoritmos Heurísticos de Cobertura de Arcos*. Tese (Doutorado) — Universidade Federal de Santa Catarina, Florianópolis, 2004.

SOUZA, M. J. F. *LINDO: manual de referência*. [S.l.], 2004. Disponível em: <http://www.decom.ufop.br/marcone/Disciplinas/OtimizacaoCombinatoria/lindo_p.pdf>.

STANLEY, R. P. Enumerative combinatorics. In: ____. 2. ed. [New York]: Cambridge University Press, 2011. v. 1, cap. What is enumerative combinatoric?, p. 9–221.

SYSTEMS, I. L. *LINDO: user's Manual*. Chicago, Illinois, 2003. Disponível em: <http://www.lindo.com/downloads/PDF/LindoUsersManual.pdf>.

APÊNDICE A – ALGO SOBRE O LINDO

O LINDO é uma ferramenta computacional muito poderosa para resolver problemas de Programação Linear, Programação Linear Inteira e quadrática (SOUZA, 2004). LINDO é a sigla de Linear, Interactive and Discrete Optimizer. As áreas de aplicação onde o LINDO mostrou ser de grande utilidade inclui distribuição de produtos, mistura de ingredientes, produção e agendamento de pessoal, gerenciamento de inventário, entre outras (SYSTEMS, 2003). Este breve tutorial é voltado para uso do LINDO em problemas de PL e de PLI.

A.1 SINTAXE DO LINDO

A sintaxe do Lindo é bastante parecida com a forma como escrevemos um problema de Programação Linear ou Inteira. E os elementos básicos de sua sintaxe são enumerados por (SOUZA, 2004), são eles:

- Uma função objetivo z que deve ser iniciada com os comandos *MAX* (ou maximize) para maximizar essa função ou *MIN* (ou minimize) para minimizá-la.
- Após a função objetivo deve-se fazer a declaração *SUBJECT TO* (ou as equivalentes apresentas por (SOUZA, 2004)). Logo após são declaradas as restrições da função objetivo.
- Para finalizar essa aplicação devemos escrever *END*

Observação A.1. Apesar de ter essa estrutura simples devemos fazer algumas observações:

1. O LINDO **NÃO** faz distinção entre letras maiúsculas e minúsculas. Isso é podemos escrever SUBJECT TO, por exemplo, como Subject to, ou subjecT to, etc. que o LINDO "entende" como se fosse o mesmo comando.
2. Não devemos iniciar variáveis com números ou com caracteres especiais, como por exemplo. E as variáveis utilizadas no Lindo devem conter **no máximo** 8 caracteres.
3. Caso a função objetivo utilize variáveis inteiras, essas devem ser declaradas após o comando END, e precedida pela declaração *GIN*.
4. O Lindo, na versão 6.1 com licença educacional, possui certas limitações, com isso, o problema para ser implementado no LINDO deve conter no máximo 50 variáveis do tipo inteira e até 500 variáveis no total. Também o problema deve ter no máximo 250 restrições e 2.000.000 de *nonzeros*, esses nonzeros em (SYSTEMS, 2003) aparecem como os coeficientes das variáveis nas equações de restrição.

Exemplo A.1. Exemplo modificado, o original consta em (SYSTEMS, 2003).

Maximize 2X + 3Y	Maximize 2X + 3Y
Subject to	Subject to
4X + 3Y < 10	4X + 3Y <= 10
3X + 5Y < 12	3X + 5Y < 12
end	end
	gin 2

gin 2 significa que as duas variáveis utilizadas na função objetivo são inteiras.

A.2 O SOFTWARE

A tela Inicial do Programa é:

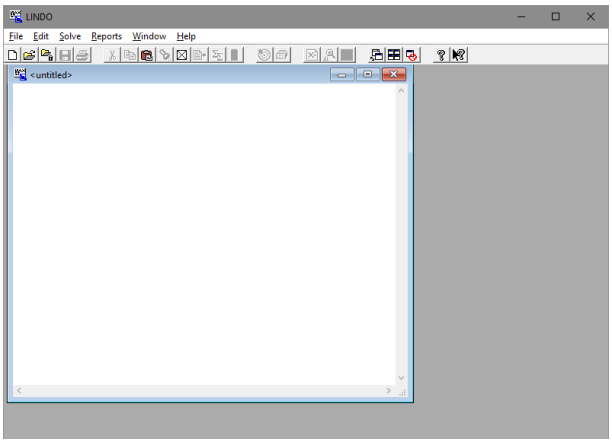


Figura 8 – Tela inicial do Lindo

Para ilustra um uso do LINDO 6.1 utilizaremos um exemplo modificado extraído de (SOUZA, 2004). Um pecuarista tem disponíveis três tipos de ração para gado. Cada tipo tem sua composição em termos de quatro nutrientes. O pecuarista quer misturar essas rações para obter um produto final que satisfaça às exigências mínimas dos animais em termos de nutrientes. A composição e as exigências estão apresentadas no quadro a seguir. E o objetivo é conseguir uma mistura de mínimo custo.

Nutrientes	**% por Kg**			**Exigência mínima em Kg por saco de 100 Kg**
	Ração 1	**Ração 2**	**Ração 3**	
1	30	25	10	6
2	20	30	20	4
3	25	15	30	4
4	25	30	40	6
Custo/Kg	1.00	1.20	1.30	

O modelo de PL inserido no LINDO é:

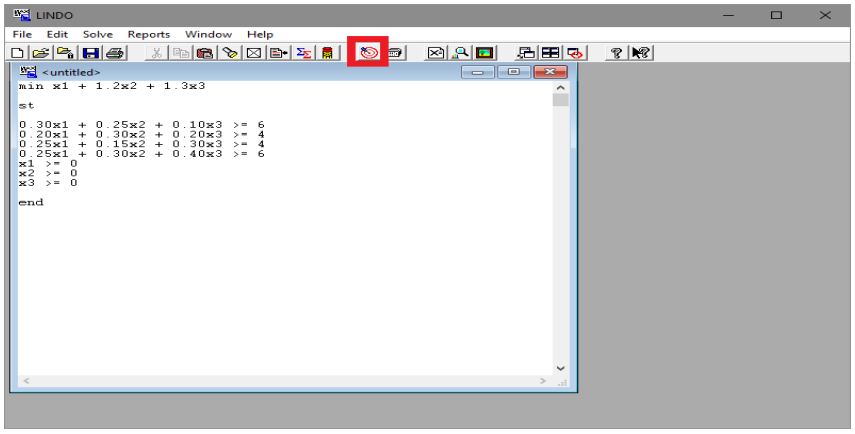


Figura 9 – Modelo de PL do problema implementado no LINDO 6.1

Para resolver o problema de PL basta clicar em Solve (na barra de ferramentas) e em seguida na opção solve, ou então clicar no ícone em destaque na figura 9. Após clicar em solve é a seguinte tela aparece. E para saber os significados dos itens nessa tela consulte (SYSTEMS, 2003).

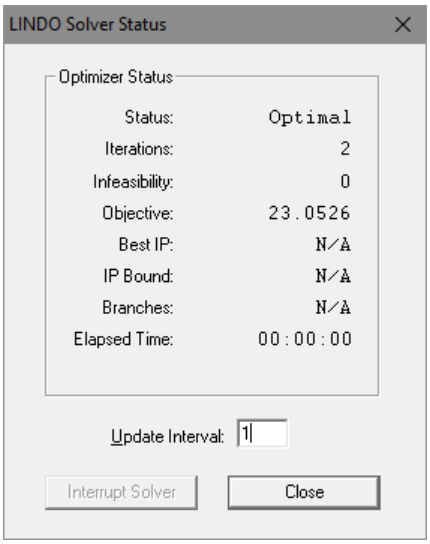


Figura 10 – Tela que aparece durante a resolução do modelo de PL no LINDO

E o resultado é apresentado da seguinte forma:

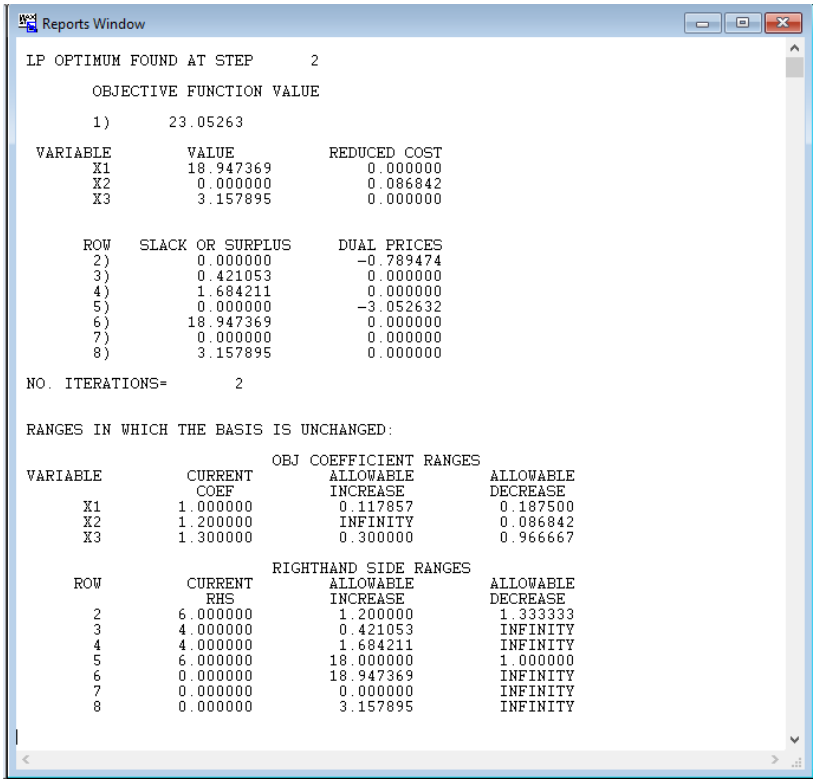


Figura 11 – Tela onde os resultados da resolução do modelo de PL no LINDO.

A estrutura apresentada na figura 11 é uma forma de retorno comum na maioria das aplicações em LINDO, mas pode ocorrer algumas variações. O valor ótimo para a função aparece logo abaixo de “OBJETIVE FUNCTION VALUE” e a tabela logo abaixo dela mostra as variáveis, na primeira coluna, os valores delas, na segunda. Na terceira coluna é apresentado o Reduced Cost que segundo (SYSTEMS, 2003) pode significar a quantidade pela qual o coeficiente da variável na função objetivo teria que melhorar antes que se tornasse viável para incluí-la na solução com valor não-nulo ou o valor de “penalidade” que teríamos que “pagar” para introduzir uma variável na solução.

A segunda tabela apresenta na primeira coluna as restrições, note que a numeração das restrições inicia em 2 e isso pode ser alterado, mas não convém mencionar isso aqui. Na segunda coluna dessa tabela temos o “SLACK OR SURPLUS” mostra o quanto estamos próximos do limite do lado direito de cada variável (e ele aparece do lado direito das inequações, ou equações, das restrições). Na terceira coluna encontramos o “DUAL PRICE” que pode ser interpretado como o valor pelo qual a função objetivo teria que melhorar para cada aumento em uma unidade no lado direito da restrição.

Após essa última tabela é apresentado o número de iterações, mas é possível que apareça outras informações. As demais tabelas que tratam sobre o “RANGE” ou estabilidade e se o leitor se interessar pode consulta (SYSTEMS, 2003) e (SOUZA, 2004), mas há casos em que essas tabelas não são apresentadas nessa janela.