Universidade Federal do Amazonas
Instituto de Ciências Exatas
Curso de Bacharelado em Matemática Aplicada

Thiago Parente da Silveira

Um estudo sobre caminhos mínimos com base na teoria dos grafos e otimização matemática

Manaus
2015

Thiago Parente da Silveira

Um estudo sobre caminhos mínimos com base na teoria dos grafos e otimização matemática

Monografia apresentada ao Curso de Bacharelado em Matemática Aplicada da UFAM, como requisito para a obtenção parcial do grau de BACHAREL em Matemática Aplicada.

Orientador: Sandro Dimy Barbosa Bitar

Doutor em Matemática

Manaus

2015

Ficha Catalográfica

Ficha catalográfica elaborada automaticamente de acordo com os dados fornecidos pelo autor.

Silveira, Thiago Parente da

S587u Um estudo sobre caminhos mínimos com base na teoria dos grafos e otimização matemática / Thiago Parente da Silveira . 2015

66.f.: il. color; 31 cm

Orientador: Sandro Dimy Barbosa Bitar

TCC de Graduação (Matemática Aplicada) - Universidade Federal do Amazonas

1.Caminho mínimo 2.Complementaridade não linear 3.Teoria das filas 4.Otimização. I. Bitar, Sandro Dimy Barbosa II.Universidade Federal do Amazonas III.Título.

Thiago Parente da Silveira

Um estudo sobre caminhos mínimos com base na teoria dos grafos e otimização matemática

Monografia apresentada ao Curso de Bacharelado em Matemática Aplicada da UFAM, como requisito para a obtenção parcial do grau de BACHAREL em Matemática Aplicada.

Aprovado em 29 de janeiro de 2015

BANCA EXAMINADORA

Sandro Dimy Barbosa Bitar

Doutor em Matemática

Roberto Cristóvão Mesquita Silva

Doutor em Matemática

Mário Salvatierra Júnior

Doutor em Matemática

Resumo

Neste trabalho são apresentados estudos realizados sobre Teoria dos Grafos, Complementaridade Não Linear e Teoria das Filas. Para o desenvolvimento teórico exploram-se algumas Definições, Teoremas e Corolários considerados essenciais à compreensão dos principais resultados.

Ao final do trabalho, discute-se uma proposta para otimização de caminhos ao longo de um grafo orientado, tendo como parâmetros iniciais o nó de origem e o nó de destino. Os resultados obtidos com as simulações e as possibilidades de aplicação, assim como a implementação para fins comerciais, são analisados.

Palavras-chaves: Caminho mínimo, complementaridade não linear, teoria das filas, otimização.

Abstract

This paper presents studies on Graph Theory, Nonlinear Complementarity and Theory of Queues. For the theoretical development if explores some definitions, theorems and corollaries considered essential to the understanding of the main results.

At the end of the paper, we discuss a proposal for ways of optimization over a directed graph, with the initial parameters the source node and the destination node. The results of the simulations and the application possibilities, as well as the implementation for commercial purposes, are analyzed.

Keywords: Shortest path, nonlinear complementarity, theory of queues, optimization.

Agradecimentos

A Deus, o maior matemático de todos. Ele nos mostrou isso ao usar a matemática como o idioma na qual escreveu o universo. Quanto mais estudamos a matemática, mais podemos conhecer à Deus.

Aos meus pais, que me apoiaram durante toda a minha vida, especialmente durante a minha graduação. Que me ensinaram o quão importante são os estudos para a formação de uma pessoa e também que a disciplina é fundamental para que se alcancem os objetivos um dia planejados.

Ao Professor Doutor Sandro Bitar, que me acompanhou durante toda a minha faculdade. Pela sua orientação, amizade e principalmente pela paciência, sem a qual este trabalho não se realizaria.

Aos meus amigos da matemática e de outros cursos que sempre me incentivaram a seguir em frente com total determinação.

Aos professores do Departamento de Matemática da Universidade Federal do Amazonas pelos seus ensinamentos e aos funcionários do curso, que durante esses anos contribuíram de algum modo para o meu enriquecimento pessoal e profissional.

"Porque melhor é a sabedoria do que as jóias; e de tudo o que se deseja nada se pode comparar a ela".

Provérbios 8.11 (Bíblia Sagrada)

Sumário

Introdução

A Matemática Aplicada apresenta-se hoje como uma área de pesquisa extremamente importante à sociedade cujo interesse por parte dos cientistas pode ser confirmado com base no volume de publicações científicas já produzidas. Buscam-se os benefícios sociais diretos ou indiretos, uma vez que os modelos matemáticos atuam no apoio à busca de soluções de problemas relacionados às diversas áreas do conhecimento. A importância da Matemática Aplicada se ampliou nas últimas décadas face à necessidade de uma gestão de recursos e produção otimizada, sujeita às diversas tecnologias sofisticadas.

Um desafio para matemática é a proposição de algoritmos que nos conduzam, através de algoritmos, às soluções ótimas ou aproximadas para problemas de interesse científico, dentre eles destacam-se uma classe considerada de difícil solução (NP-difícil); são os chamados problemas de Otimização Combinatória. Alguns desses problemas são tratados com o uso de métodos eficientes para solução; outros utilizam métodos não tão rápidos, mas necessário, motivado pela sua complexidade computacional.

Este trabalho expõe alguns temas relacionados à Teoria dos Grafos, Otimização Matemática e Teoria das Filas, objetivando a correta compreensão dos conceitos e principais Teoremas investigados e, em segundo plano, a formulação de um modelo matemático para a solução de um problema relacionado a caminhos mínimos sobre sistemas urbanos semaforizados.

Inicialmente é feito um desenvolvimento dos principais resultados referentes à Teoria dos Grafos, demonstrando Teoremas e Corolários relativos ao assunto. Como aplicação dos conhecimentos adquiridos, buscou-se o entendimento de um problema clássico: o Problema do Caminho Mínimo em Grafos.

Em seguida, após a compilação desse conhecimento, realizou-se uma pesquisa sobre sobre Redes em Grafos e Complementaridade Não Linear. Um resultado sobre equivalência do Problema de Complementaridade Não Linear e o Equilíbrio de Tráfego foi investigado. Essa discussão é apresentada no Capítulo 2. A caracterização para o problema do Equilíbrio de Tráfego, assim como as demonstrações pertinentes a equivalência são apresentadas nesse capítulo.

Seguinte aos resultados da Teoria dos Grafos e de Complementaridade Não Linear, investigou-se um dos capítulos mais importante da Pesquisa Operacional: Teoria das Filas. Os resultados do estudo servem para dar sustentação à proposta de um modelo matemático para o problema do caminho mínimo apresentada no Capítulo 4.

Finalmente, discute-se a implementação de um algoritmo baseado no estudo realizado ao longo deste trabalho para o problema do caminho mínimo. Como hipótese assume-se uma rede conexa cujas arestas são valoradas com números reais não negativos. A origem, o destino, a taxa de chegada em cada ponto de atendimento da rede e o índice de congestionamento do sistema são utilizados como parâmetros do modelo durante a simulação.

Os principais Teoremas e Corolários necessários à fundamentação matemática do modelo final são discutidos e demonstrados no decorrer do trabalho.

1 Elementos da Teoria dos Grafos

Neste capítulo apresenta-se um embasamento teórico sobre a Teoria dos Grafos, objetivando a modelagem e aplicação dessa ferramenta ao Problema do Caminho Mínimo (PCM). Assim sendo, nas seções seguintes, dedica-se especial atenção às Definições e Teoremas relativos ao tema.

1.1 Definições e Propriedades da Teoria dos Grafos

Definição 1.1.1. Um grafo G consiste de um conjunto V de elementos, que são denominados vértices, um conjunto A de elementos denominados arestas, e uma função de incidência ψ que associa a cada aresta α de G um par não ordenado de vértices (não necessariamente distintos) de G, denominados extremidades de α.

O número de vértices de um grafo será simbolizado por $|V|$ ou por n, e o número de arestas será denotado por $|A|$ ou por m. Quando não houver risco de confusão, será denotado por $V(G)$ o conjunto dos vértices do grafo G e por $A(G)$ o conjunto de arestas de G.

Grafos podem ser representados por diagramas, onde cada vértice é representado por um ponto e cada aresta por uma linha ligando os pontos que representam as suas extremidades. As extremidades de uma aresta são *incidentes* à aresta, e vice-versa. Os extremos de uma aresta são *adjacentes* (mesmo que coincidam). Arestas com pelo menos um extremo em comum também são ditas *adjacentes*.

Definição 1.1.2. Uma aresta é chamada de laço quando incide duas vezes no mesmo vértice, caso contrário diz-se que se trata de uma ligação.

Definição 1.1.3. Um multigrafo é um grafo onde existem pelo menos dois vértices que são ligados por mais de uma aresta.

Na Figura 1.1(a) temos um grafo com um laço no vértice B e na Figura 1.1(b) temos um multigrafo com três arestas ligando os vértices D_1 e C_1.

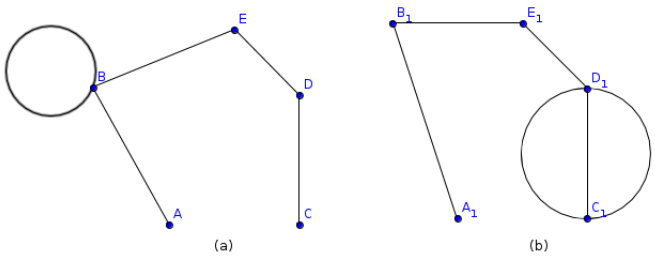

Figura 1.1: (a) Grafo com um laço no vértice B
b) Multigrafo com arestas mútiplas entre os vértices C_1 e D_1

Com base nas definições 1.1.2 e 1.1.3 será definido um tipo de grafo na qual será baseado a maior parte desse capítulo.

Definição 1.1.4. Um grafo simples é um grafo que não tem laços e nem é um multigrafo.

Definição 1.1.5. O grau de um vértice v é o número de arestas que incidem em v e será denotado por $d(v)$.

Teorema 1.1.1. *Para todo grafo G, a soma dos graus de seus vértices é o dobro do número de suas arestas, ou seja,*

$$\sum_{v \in V(G)} d(v) = 2|A|$$

Demonstração. Quando soma-se os graus dos vértices, contam-se as extremidades das arestas uma vez. Como cada aresta tem duas extremidades, cada aresta é contada duas vezes. Note que este fato é independente do grafo possui laços ou não. De fato, se G possui um laço em x_0, então a aresta incide duas vezes no vértice, acrescentando o grau do vértice em dois. □

Corolário 1.1.1. *Todo grafo G possui um número par de vértices com grau ímpar.*

Demonstração. Se o número de vértices com grau ímpar fosse ímpar a soma dos graus de todos os vértices seria ímpar. Mas a soma dos graus é o dobro do número de arestas e, portanto é um número par. □

Definição 1.1.6. Um grafo é dito completo quando qualquer par de vértices é ligado por uma aresta. Um grafo completo com n vértices será denotado por K_n.

A Figura 1.2 ilustra dois grafos completos.

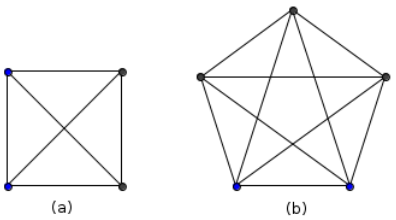

Figura 1.2: (a) Grafo completo com 4 vértices, ou seja, um K_4
(b) Grafo completo com 5 vértices, ou seja, um K_5

Teorema 1.1.2. *Um K_n possui $\binom{n}{2}$ arestas.*

Demonstração. De fato, sendo G um K_n, o grau de cada vértice é $n-1$. Sejam, $v_1, v_2, \ldots, v_n$ os vértices de G. Pelo Teorema 1.1.1 tem-se que

$$\sum_{i=1}^{n} d(v_i) = 2|A|$$

ou seja,

$$n(n-1) = 2|A| \Rightarrow |A| = \frac{n(n-1)}{2} = \binom{n}{2}$$

□

Definição 1.1.7. Um passeio em um grafo G é uma sequência de vértices $v_1, v_2, \ldots, v_k$ tal que o vértice v_i é adjacente ao vértice v_{i+1}, ou seja, quaisquer dois vértices consecutivos na sequência tem que serem ligados por uma aresta. Em um passeio pode-se repetir um ou mais vértices. Se o primeiro vértice do passeio coincide com o último diz-se que se trata de um passeio fechado, caso contrário trata-se de um passeio aberto.

Definição 1.1.8. Um caminho é um passeio onde não há vértices repetidos (com possível excessão das extremidades). O primeiro e o último vértice do caminho são chamados extremidades do caminho. Se estes forem o mesmo, diz-se que se trata de um ciclo. O número de arestas em um caminho é chamado comprimento. Um ciclo de comprimento k é denotado por k-ciclo. Um caminho também pode ser definido como sendo um ciclo sem uma das arestas.

Definição 1.1.9. Um grafo $H = (U, B)$ é dito subgrafo de $G = (V, A)$ se $U \subseteq V$ e $B \subseteq A$

Uma noção importante na Teoria dos Grafos, é a de *grafo conexo.*

Definição 1.1.10. Um grafo G é dito conexo se dados dois vértices quaisquer u e v, existe um caminho com extremidades u e v, ou seja, se existe um caminho ligando estes dois vértices. Caso contrário se trata de um grafo desconexo.

Se um grafo G for desconexo, então G terá subgrafos conexos; por exemplo, o subgrafo consistindo de um único vértice v_0 é conexo. Um *componente conexo* H é um subgrafo maximal que é conexo, em outras palavras, H é um componente conexo se ele é conexo mas qualquer outro subgrafo de G que contém H é desconexo.

Talvez o resultado mais antigo em teoria dos grafos tenha sido descoberto por Euler, considerado o maior matemático do século $XVIII$.

Tudo começou com um desafio recreativo que os cidadãos de Konisberg (hoje: Kaliningrado) propuseram. A cidade era dividida em quatro distritos por braços do rio Pregel, que eram conectados por sete pontes. Era agradável caminhar por ali, atravessando as pontes, e daí surgiu a questão: é possível um passeio de modo que se atravessasse todas as pontes exatamente uma vez?

Euler publicou um artigo em 1736, no qual ele demonstrava que tal passeio era impossível. Euler concluiu que se tal passeio existisse, ele tinha que começar em um distrito e necessariamente teria que terminar em outro, mas no total haviam quatro distritos, o que não tornava possível tal passeio. Este resultado é geralmente considerado como o primeiro Teorema da teoria dos grafos (LOVÁSZ, 2013).

Definição 1.1.11. Um grafo G conexo com m arestas é dito euleriano se existe um passeio fechado com comprimento m, em outras palavras, um grafo é euleriano se saindo de um vértice, passa-se por todas as arestas exatamente uma vez e ao mesmo vértice retorna-se. Se o grafo não é euleriano, mas tem um passeio aberto que passe por todas as arestas exatamente uma vez, ele é dito semi-euleriano.

Teorema 1.1.3. *Se todo vértice de um grafo (não necessariamente simples) possui grau maior do que ou igual a* 2*, então o grafo contém um ciclo.*

Demonstração. Se o grafo contém um laço ou arestas múltiplas, não há o que demonstrar. Suponha que se trata de um grafo simples. Seja v_0 o vértice na qual se inicia um passeio.

Ao sair de v_0 chegamos em v_1. Como v_1 tem grau maior do que ou igual a 2, tem-se que sair de v_1. Repetindo este procedimento, ao chegar em um vértice v_k, tem-se duas opções: ou se está retornando a v_k, criando assim um ciclo, ou então, está se chegando pela primeira vez em v_k, mas como $d(v_k) \geq 2$ tem-se que sair de v_k. Como o número de vértices é finito, repetindo este procedimento mais algumas vezes, uma hora terá que se retornar a algum vértice na qual já se passou, criando assim um ciclo. □

Teorema 1.1.4. *Um grafo é euleriano se, e somente se, todo vértice tem grau par.*

Demonstração. Seja G um grafo euleriano com m arestas. Cada vez que o passeio passa por um vértice, ele utiliza duas arestas, uma para entrar e outra para sair. Logo o grau de cada vértice deve ser obrigatoriamente par.

Suponha que todos os vértices de G tenham grau par. Sendo G conexo todos os vértices tem que ter grau maior do que ou igual a 2. Pelo Teorema 1.1.3, G contém um ciclo (que é um passeio fechado). Dentre todos os passeios fechados em G, escolhe-se o um passeio T que tenha comprimento máximo. Se T tem comprimento m, o Teorema está demonstrado. Suponha que isso não aconteça. Considere agora o grafo H resultante da retirada das arestas de T. Como retira-se um número par de arestas de cada vértice de T, e todos os vértices do grafo tem grau par, pelo menos uma das componentes de H tem um vértice em comum com T e todos os seus vértices tem grau par, assim sendo H tem uma trilha fechada que passa por todos os seus vértices, e pode-se então formar uma trilha maior concatenando T com a trilha H. Mas isto contraria a maximalidade de T. □

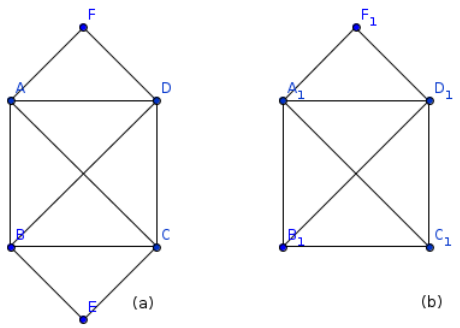

Figura 1.3: (a) Grafo euleriano
(b) Grafo semi-euleriano

Uma questão semelhante ao problema das Pontes de Konisberg foi levantada

por outro famoso matemático, o irlandês William R. Hamilton em 1856. Um *ciclo hamiltoniano* é um ciclo que contém todos os nós de um grafo. O Problema do Ciclo Hamiltoniano é o problema de se decidir se dado um grafo tem ou não um ciclo hamiltoniano.

Ciclos hamiltonianos parecem bem semelhantes a passeios eulerianos: ao invés de exigir que toda aresta seja usada uma vez, deseja-se que todo nó seja usado exatamente uma vez. Mas muito menos se conhece sobre eles do que grafos eulerianos. Euler mostrou como decidir se um grafo é ou não euleriano, mas nenhuma maneira eficiente é conhecida para se verificar se um grafo tem ou não um ciclo hamiltoniano, e nenhuma condição necessária e suficiente para existência de um ciclo hamiltoniano é conhecida.

1.2 Árvores

Nesta seção faremos um estudo sobre um tipo de grafo que possui propriedades particulares e de interesse para o estudo realizado sobre Caminhos Mínimos que será apresentado ao final deste trabalho.

Definição 1.2.1. Um grafo $G = (V, A)$ é chamado de árvore se ele é conexo e não contém qualquer ciclo como subgrafo.

A árvore mais simples tem um vértice e nenhuma aresta. A segunda árvore mais simples contém dois vértices e uma aresta ligando eles.

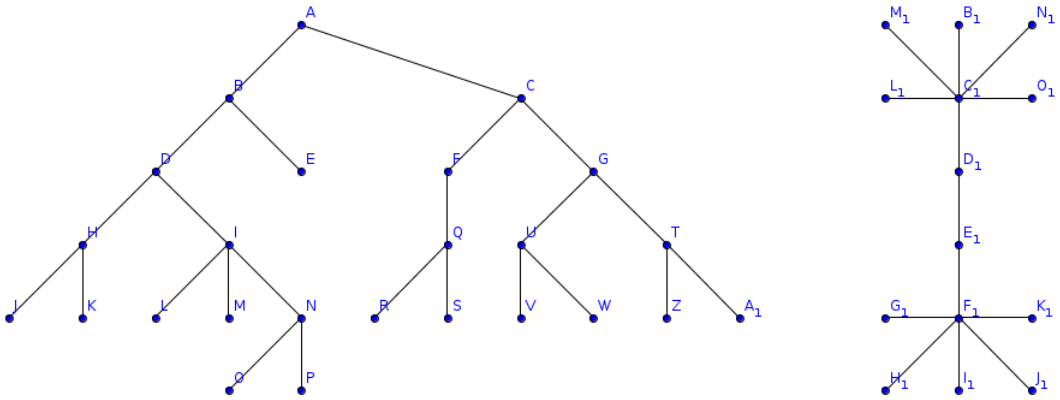

Figura 1.4: Árvores

Note que as duas propriedades que definem árvores funcionam em direções opostas: conectividade significa que o grafo não pode ter "arestas de menos"e, não ter ciclos, significa que o grafo não pode ter "arestas demais". Para ser mais preciso, se um

grafo é conexo, então ao adicioná-lo uma nova aresta ele permanece conexo. Entretanto ao remover uma de suas arestas, o grafo pode permanecer ou não conexo. Se um grafo não contém qualquer ciclo, então ao remover qualquer aresta o grafo permanecerá sem ter ciclos, mas ao adicionar um nova aresta ao grafo, este pode ter ou não um ciclo.

Teorema 1.2.1. *Um grafo G é uma árvore se, e somente se, ele é conexo, mas a remoção de qualquer de suas arestas resulta em um grafo desconexo.*

Demonstração. Suponha que G é uma árvore. Então pela definição de árvore G é conexo. Deseja-se provar então que após a remoção de qualquer aresta, ele não permanece conexo. Assuma que ao remover uma aresta uv do grafo G ele permaneça conexo. Então o grafo resultante terá um caminho P entre os vértices u e v, mas se for colocada a aresta uv de volta, o caminho P e a aresta uv formará um ciclo em G, o que contradiz a definição de árvores.

Suponha agora que G é um grafo conexo tal que a remoção de qualquer aresta resulta em um grafo desconexo. Suponha que G não é uma árvore. Suponha também que G contenha um ciclo C. Então ao remover qualquer aresta de C, o grafo resultante ainda é conexo, uma contradição. Portanto G é uma árvore. □

Teorema 1.2.2. *Um grafo G é uma árvore se, e somente se, ele não contém nenhum ciclo, mas a adição de qualquer nova aresta cria um ciclo.*

Demonstração. Suponha que G é uma árvore. Considere dois vértices u e v. Como G é conexo, então existe um caminho P entre u e v. Ao adicionar uma nova aresta ligando u e v então o grafo G terá um ciclo formado por P e por uv. Como u e v foram tomados arbitrariamente, a adição de qualquer aresta forma um ciclo.

Agora suponha que G é um grafo sem ciclos, mas que a adição de qualquer aresta cria um ciclo. Sejam u e v dois vértices quaisquer. Se for adicionada uma aresta ligando u e v, por hipótese G terá um ciclo. Assim sendo, sebe-se que um caminho é um ciclo sem uma das arestas. Então, ao retirar a aresta uv tem-se que existe um caminho entre u e v, logo G é conexo, pois u e v foram tomados arbitrariamente. Como G não contém ciclos e é conexo, G é uma árvore. □

Considere um grafo conexo G sobre n vértices, e uma aresta α desse grafo. Ao remover a aresta α deste grafo ele pode permanecer ou não conexo. Se o grafo ficar

desconexo, então diz-se que a aresta α é uma *aresta de corte* ou então *ponte*. Pelo Teorema 1.2.1 percebe-se que toda aresta de uma árvore é uma ponte.

Definição 1.2.2. Seja G um grafo. Um subgrafo de G com o mesmo conjunto de vértices que é também uma árvore é dito árvore geradora de G.

Um dos processos para se encontrar uma árvore geradora de um grafo conexo é o seguinte: se existe uma aresta que não é uma ponte, então a remova. Repita este procedimento até que a remoção de qualquer outra aresta resulte em um grafo desconexo.

Frequentemente usam-se árvores que tem um nó especial, que se chama *raiz*. Pode-se tomar qualquer árvore, selecionar qualquer um de seus vértices, e chamá-lo de raiz. Uma árvore com uma raiz determinada é chamada *árvore enraizada*.

Teorema 1.2.3. *Toda árvore com pelo menos dois vértices tem pelo menos dois vértices de grau 1.*

Demonstração. Seja G uma árvore com pelo menos dois vértices. Começa-se um passeio de um vértice v_0 de grau 1 da árvore, que existe, pois caso não existisse, todos os vértices teriam grau maior do que ou igual a dois, visto que G é conexo, pelo Teorema 1.1.3 G teria um ciclo, o que não pode pois G é uma árvore. Esse passeio é feito de tal maneira que não deseja-se voltar de um vértice sobre a aresta na qual se entrou nesse vértice; isso é possível, a menos que se tome um vértice de grau 1, e nesse caso a demonstração estaria concluída. Suponha que isso não aconteça, então em algum momento, tem-se que retornar a um vértice que já foi visitado, pois o número de vértices é finito. Ora, então os vértices e arestas percorridos entre as duas visitas formam um ciclo, uma contradição, pois G é uma árvore e não contém ciclos. □

O seguinte procedimento é conhecido como *Procedimento de Crescimento de Árvore*, ou simplesmente (PCA).

Algoritmo PCA

Passo 1

Comece com apenas um vértice;

Passo 2

Repita o que se segue um número qualquer de vezes: dada uma etapa k do processo, na

próxima etapa crie um novo vértice e ligue-o por uma nova aresta a qualquer vértice de G.

Fim do Algoritmo

O próximo Teorema caracteriza um grafo que é uma árvore e, além disso, faz uma demonstração de como uma árvore é obtida.

Teorema 1.2.4. *Todo grafo obtido pelo Procedimento de Crescimento de Árvore é uma árvore, e toda árvore pode ser obtida dessa maneira.*

Demonstração. Seja G um grafo obtido pelo P.C.A. Para apenas um vértice, tem-se uma árvore. O que será mostrado é que nunca será criado um grafo que não seja uma árvore, em outras palavras, se G é uma árvore, e G' for obtido de G criando-se um novo vértice v e conectando-o a um vértice u de G, então G' é uma árvore. De fato, pois G' é conexo, já que quaisquer dois vértices de G podem ser conectados por um caminho em G (G é uma árvore), enquanto que v pode ser conectado a qualquer outro vértice w primeiro indo a u e depois conectando u a v tem-se então um caminho entre w e v. Além do mais, G não tem ciclo: v tem grau 1 e portanto nenhum ciclo pode passar por v, mas um ciclo que não passa por v seria um ciclo em G, o que não pode acontecer pois G é uma árvore. Logo G' também é uma árvore.

Seja G uma árvore. Se o número de vértices é 1, então a árvore surge por construção, pois essa é a maneira na qual se inicia o P.C.A. Assuma que G é uma árvore com pelo menos dois vértices. Assim pelo Teorema 1.2.3, G tem um vértice de grau 1 (pelo menos dois na verdade). Seja v um vértice com grau 1 de G, juntamente com a aresta com extremidade v, para obter o grafo G'. Afirma-se que G' é uma árvore. De fato, G' é conexo: quaisquer dois vértices de G' podem ser conectados por um caminho de G, e esse caminho não pode passar por v, pois v tem grau 1. Portanto esse caminho também é um caminho em G'. Além do mais, G' não contém nenhum ciclo pois G não contém. Assuma que toda árvore com menos vértices que G surge pela construção, em particular G' assim o faz. Mas então G surge de G', por uma iteração a mais do segundo passo, completando a prova do Teorema. □

O Procedimento de Crescimento de Árvore pode ser usado para estabelecer propriedades das árvores. Talvez a mais importante delas concerne ao número de arestas. O número de arestas de uma árvore depende somente do número de vértices.

Teorema 1.2.5. *Toda árvore que tem n vértices possui $n-1$ arestas.*

Demonstração. De fato, começa-se com um vértice e nenhuma aresta, e a cada passo, um novo vértice e uma nova aresta são adicionados, e essa diferença de 1 é mantida. □

Uma questão bastante natural com respeito ao estudo de árvores é a seguinte: quantas árvores existem sobre n vértices? Antes de responder a essa questão, outra deve ser respondida: quando que duas árvores são diferentes?

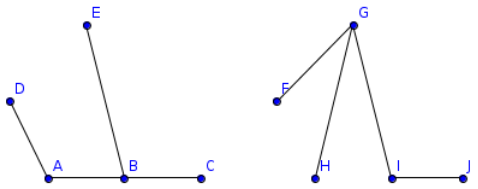

Figura 1.5: Árvores iguais

Definição 1.2.3. Sejam $G = (V, A)$ e $H = (U, B)$ dois grafos. Um isomorfismo entre os grafos G e H é uma aplicação $\varphi : V \to U$, tal que $\{c, d\} \in A \Rightarrow \{\varphi(c), \varphi(d)\} \in U$.

Consideram-se duas árvores iguais se pode-se rearrumar os vértices de uma árvore de modo que se obtém a outra árvore. Mais exatamente, consideram-se duas árvores iguais quando há um *isomorfismo* entre as duas árvores, ou seja, uma correspondência um pra um entre os vértice de uma árvore com os vértices da outra árvore que preserve as adjacências. Árvores *rotuladas* são aquelas em que se fixam os vértices, dando nomes aos mesmos. Se estiver trabalhando com árvores *não rotuladas*, árvores isomorfas não possuem distinção alguma. Árvores não rotuladas são diferentes quando não existe um isomorfismo entre elas.

1.2.1 Armazenamento de árvores

Agora estudaremos uma estratégia para o problema de armazenar árvores. Existem várias maneiras de fazer isso, mas do ponto de vista computacional é importante utilizar a menor quantidade de memória possível. Suponha que por algum motivo apareça a necessidade de armazenar uma árvore rotulada, por exemplo a árvore da Figura 1.6.

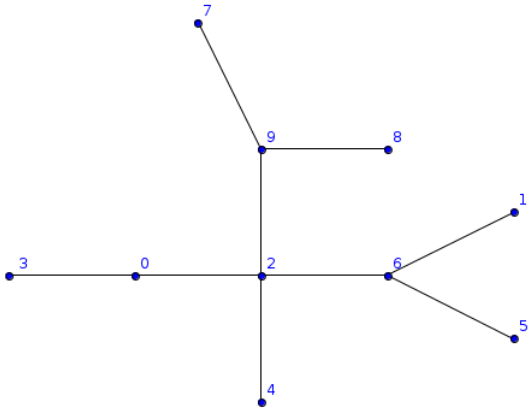

Figura 1.6: Árvore rotulada

A maneira na qual se realizará este processo depende de quais dados deseja-se recuperar e com que frequência. No momento, a única preocupação será com a quantidade de memória necessária para tal. Deseja-se então armazenar uma árvore de tal maneira que ela ocupe o mínimo de espaço possível no computador.

Suponha que se tem uma árvore G sobre n vértices. Uma maneira de se armazenar esta árvore é usando uma matriz binária, conhecida como *matriz de adjacência*, definida da seguinte maneira: é uma matriz quadrada de ordem n onde a i-ésima linha e a i-ésima coluna representam o mesmo vértice, e a entrada $A_{ij} = 0$ se o vértice i não está ligado ao vértice j e $A_{ij} = 1$ se existe uma aresta entre os vértices i e j. Claramente esta é uma matriz simétrica com 0 na diagonal principal pois G é uma árvore e não possui laços. Precisa-se de um bit para armazenar cada entrada desta matriz, portanto isso necessita de n^2 bits. Pode-se economizar um pouco usando apenas a parte inferior da diagonal principal (pois a matriz é simétrica e possui 0 na diagonal principal), no entanto isto ainda necessita de $\frac{n^2-n}{2}$ bits.

Uma maneira melhor é especificar cada árvore listando todas as suas arestas. Pode-se especificar cada aresta por suas extremidades. Será conveniente arranjar essa lista em um vetor cujas colunas representam essas arestas. Por exemplo, a árvore da Figura 1.6 pode ser codificada por:

$$\begin{matrix} 7 & 8 & 9 & 6 & 3 & 0 & 2 & 6 & 6 \\ 9 & 9 & 2 & 2 & 0 & 2 & 4 & 1 & 5 \end{matrix}$$

Ao invés de uma matriz com n linhas, obtém-se uma matriz com duas linhas. Paga-se um pouco por isso, pois passa-se a trabalhar com uma matriz cujas entradas

são inteiros entre 0 e $n-1$ em vez de uma matriz binária. Mas isto certamente vale a pena: mesmo se forem contados os bits, escrever o rótulo de um vértice toma $log_2 n$ bits, portanto a matriz inteira ocupa apenas $2nlog_2 n$ bits, o que é muito menor do que $\frac{n^2-n}{2}$ se n é grande (e geralmente é).

Agora será definido uma maneira de se armazenar árvores, conhecido como *código de pai* De agora em diante, o vértice com rótulo 0 terá um papel especial. Será considerado esse vértice como o vértice "raiz"da árvore. Então pode-se listar os dois vértices extremos de uma aresta listando a extremidade mais distante da raiz primeiro e depois a extremidade mais próxima da raiz. Portanto para toda aresta, o vértice escrito abaixo é o pai do vértice escrito acima. Para a ordem na qual listam-se as arestas, tomar-se-à a ordem de seus primeiros vértices. Para a árvore na Figura 1.6, obtém-se a matriz

1	2	3	4	5	6	7	8	9
6	0	0	2	6	2	9	9	2

Algo especial nessa matriz é o fato de que na primeira linha aparecem os inteiros de 1 até $n-1$ exatamente ordenados. Por conseguinte, sabe-se de antemão que se tiver uma árvore sobre n vértices, e esta for armazenada usando o código de pai, então na primeira linha da matriz consistirá de $1, 2, \ldots, n-1$. Com isso, pode-se suprimir a primeira linha sem perder qualquer informação e armazenar apenas a segunda linha. Portanto pode-se especificar a árvore por uma lista de $n-1$ números, cada um entre 0 e $n-1$. Isso toma $(n-1)log_2 n$ bits. Essa codificação não é ótima, no sentido de que nem todo código retorna uma árvore, mas esse método é quase ótimo.

Agora será descrito um método parecido com o código de pai, este é chamado *códido de Prüfer*, que atribuirá a qualquer árvore rotulada de n vértices uma lista de comprimento $n-2$ e não $n-1$, consistindo dos números $0, 1, \ldots, n-1$.

O código de Prüfer pode ser considerado um refinamento do código de pai. Ainda será considerado 0 como raiz, e ordena-se as duas extremidades de uma aresta tal que filho vem primeiro, mas ordenam-se as arestas (as colunas da matriz) não pela magnitude de sua primeira extremidade porém um pouco diferentemente, mais diretamente relacionado à própria árvore.

Portanto, novamente será construída uma matriz com duas linhas cujas colunas correspondem as arestas, e cada aresta é listada de modo que o vértice mais distante da raiz esteja no topo e seu pai embaixo. A questão é a ordem na qual listam-se as arestas.

A regra é a seguinte: procura-se por um vértice de grau 1 (esses vértices são conhecido como folhas, exceto a raiz quando a mesma possui grau 1), com o menor rótulo, e escreve-se essa aresta com essa extremidade. Com base na Figura 1.6, a primeira coluna será formada pelos vértices 1 e 6 respectivamente. Então, remove-se este vértice e a aresta da árvore e repete-se o argumento até que todas as arestas sejam listadas. A matriz que obtem-se é chamada de *código estendido de Prüfer* da árvore (chama-se estendido porque, como será mostrado, precisa-se apenas de uma parte desse código como Prüfer "real"). O código de Prüfer da árvore da Figura 1.6 é

$$\begin{matrix} 1 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 2 \\ 6 & 0 & 2 & 6 & 2 & 9 & 9 & 2 & 0 \end{matrix}$$

Algo que ainda não está claro é o fato de que o código de Prüfer é melhor que o código de pai. Mas uma primeira observação é que a última entrada da segunda linha da matriz é sempre 0, pois este vértice é a raiz e é sempre o último a ser atingido. Mas aparentemente não há vantagens, pois na primeira linha guarda-se os números $1, 2, \ldots, n-1$ também, mas de maneira não ordenada. Apesar de não ser visível, a primeira linha é um tanto supérflua, e pode ser determinada pela segunda linha, como diz o Teorema 1.2.6.

Teorema 1.2.6. *A segunda linha de um código de Prüfer determina a primeira.*

Demonstração. De fato, pois cada entrada na primeira linha do código estendido de Prüfer é o menor inteiro que não ocorre na primeira linha antes dele, nem na segunda linha abaixo ou depois dele, pois quando essa entrada (por exemplo a k-ésima entrada da primeira linha) foi registrada, então os vértices antes dela na primeira linha foram removidos (juntamente com as arestas correspondentes às $k-1$ colunas). As entradas remanescentes na segunda coluna são exatamente aqueles vértices que são pais ao mesmo tempo, o que significa que eles não são folhas. Isto descrever como a primeira linha pode ser reconstruída a partir da segunda. □

Portanto, não é preciso armazenar toda a matriz do códido estendido de Prüfer para armazenar a árvore, basta armazenar a segunda linha. Na verdade, como se sabe que a última entrada é 0, não é necessário o armazenamento desta entrada. A lista consistindo das primeiras $n-2$ entradas da segunda linha é conhecido como *código de Prüfer* da árvore. Por conseguinte, este código é uma lista de comprimento $n-2$ onde cada entrada é um número inteiro entre 0 e $n-1$, o que é bem semelhante ao código

de pai, sendo apenas um dígito mais curto. Mas a principal característica do código de Prüfer é que ele é ótimo, como diz o Teorema 1.2.7.

Teorema 1.2.7. *Toda lista de números entre* 0 *e* $n-1$*, de comprimento* $n-2$*, é um código de Prüfer de alguma árvore sobre* n *vértices.*

Demonstração. É bem simples, considere uma de $n-2$ entradas com cada entrada assumindo valores inteiros entre 0 e $n-1$. Então acrescenta-se 0 após a última entrada aumentando o tamanho da lista. Após isso, escreve-se uma linha acima desta lista de mesmo comprimento, de maneira que acima de cada entrada na primeira linha seja o menor inteiro que não ocorre na primeira linha antes dele e nem na segunda abaixo ou depois dele (o que é sempre possível pois esta condição exclui no máximo $n-1$ valores de n possíveis). O resultado é um código de Prüfer estendido que é uma árvore que é única. De fato, pois se duas listas distintas representassem a mesma árvore, então as matrizes teriam que ser iguais, o que implicaria que cada entrada teria que ser igual a sua correspondente na outra lista, sendo então as duas listas iguais. □

1.2.2 Número de árvores rotuladas e não rotuladas

Teorema 1.2.8. ***(Teorema de Cayley)*** *O número de árvores rotuladas sobre* n *vértices é* n^{n-2}*.*

Demonstração. De fato, toda lista de $n-2$ elementos com cada entrada entre 0 e $n-1$ corresponde a uma árvore sobre n vértices, assim sendo o número de sequências é n^{n-2}. □

O número de árvores não rotuladas sobre n vértices, que será denotado por T_n, é bem mais difícil de se trabalhar. Nenhuma fórmula simples como a do Teorema 1.2.8 é conhecida para se determinar T_n.

É claro que o número de árvores não rotuladas é menor do que o número de árvores rotuladas; toda árvore rotulada pode ser rotulada de muitas maneiras. Se for desenhada uma árvore não rotulada, pode-se rotular seus vértices de $n!$ maneiras. As árvores rotuladas que se obtém dessa maneira não são necessariamente todas diferentes. Por exemplo, se a árvore for um estrela, então independentemente de como se permutam os rótulos das folhas, obtém-se a mesma árvore rotulada.

Teorema 1.2.9. *O número de* T_n *de árvores não rotuladas sobre* n *vértices satisfaz*

$$\frac{n^{n-2}}{n!} \leq T_n$$

Demonstração. Se sabe que cada árvore rotulada pode ser rotulada de no máximo $n!$ maneiras. Como o número de árvores rotuladas é n^{n-2}, segue que:

$$\frac{n^{n-2}}{n!} \leq T_n$$

□

Note que o Teorema 1.2.9 representa apenas um limitante inferior para T_n. Como calcular então um limitante superior? Informalmente, como deve-se descrever uma árvore, se deseja-se apenas o "formato"dela e não se está preocupado com qual vértice possui qual rótulo?

Tome uma árvore G sobre n vértices, e especifique uma de suas folhas como sendo a raiz. Em seguida desenhe G no plano sem cruzar as arestas, isso pode sempre ser feito e quase sempre se desenham árvores dessa maneira. Agora imagine que as arestas da árvore são paredes perpendiculares ao plano. Começando na raiz, caminhe ao redor desse sistema de paredes, mantendo sempre a parede à sua direita. Chama-se o passeio sobre uma aresta de um "passo". Como existem $n-1$ arestas e cada arestas tem dois lados, será dado $2(n-1)$ passos antes de se retornar a raiz.

A cada vez que se dá um passo para longe da raiz (isto é, um passo de um pai para um de seus filhos), escreve-se 1, e cada vez que se dá um passo em direção a raiz, escreve-se 0. Dessa maneira se determina uma lista binária com $2(n-1)$ elementos. Esse código é conhecido como *código planar* da árvore não rotulada. O código planar tem a seguinte propriedade: toda árvore não rotulada é unicamente determinada pelo seu código planar (LOVÁSZ, 2013).

Teorema 1.2.10. *O número T_n de árvores não rotuladas satisfaz*

$$T_n \leq 4^{n-1}$$

Demonstração. Como o número de possíveis códigos planares é no máximo $2^{2(n-1)}$, obtem-se que o número de árvores não rotuladas é no máximo 4^{n-1} □

Teorema 1.2.11. *O número T_n de árvores não rotuladas sobre n vértices satisfaz:*

$$\frac{n^{n-2}}{n!} \leq T_n \leq 4^{n-1}$$

Demonstração. Pelo Teorema 1.2.9 se obtém a primeira parte da desigualdade e pelo Teorema 1.2.10 se obtem a segunda parte. □

1.2.3 Árvore ótima

Começamos esta seção apresentando uma discussão sobre um problema real que pode ser tratado matematicamente como um modelo em grafos. Suponha país com n cidades deseja construir uma nova rede telefônica para ligar todas as cidades. É claro que não é necessária a construção de uma linha para cada par de cidades, mas de fato a rede tem que ser conexa. Suponha que por algum motivo o país não deseja construir uma linha direta entre cidades que podem ser atingidas por outro meio. Do ponto de vista da teoria dos grafos, o que se está a procura é de um grafo conexo "mínimo"com n vértices, isto é, uma árvore.

Sabe-se que independentemente de qual árvore será feita, será necessário pelo menos $n - 1$ linhas. Apesar deste número ser fixo, não será qualquer árvore que será escolhida, pois algumas linhas com certeza não são tão fáceis de se construir, e todas não possuem o mesmo custo. O objetivo então é construir uma árvore geradora cujo custo total (a soma dos custos de suas arestas) seja mínima. Suponha que se esteja de mão de todos os custos.

Uma ideia bem natural para se resolver este problema, seria gerar todas as árvores possíveis, calcular o custo de cada uma delas e simplesmente optar pela melhor. Mas essa com certeza não é a melhor ideia. Pelo Teorema 1.2.8 que o número de árvores rotuladas sobre n vértices é n^{n-2}. Portanto, para 10 cidades teriam que ser analisadas 10^8 (cem milhões) de árvores possíveis, e para 20 cidades o número é absurdamente grande. Logo, analisar árvore por árvore não é a melhor solução.

Suponha que o governo queira projetar a árvore de maneira otimista, ou seja, a cada passo, escolhe-se o melhor passo possível. A cada passo, o governo levanta o investimento necessário para a construção da linha, esse método é conhecido como *método guloso* ou *Algoritmo Guloso*. Após o término da construção, o resultado é uma árvore onde cada vértice representa uma cidade e cada aresta representa uma linha telefônica. Essa árvore é de fato tão barata quanto possível, mas se a tarefa for um pouco diferente, a mesma abordagem pode levar a resultados muito ruins.

Suponha que por razões de confiabilidade, o governo exige que entre quaisquer duas cidades deve haver pelo menos dois caminhos possíveis sem qualquer aresta em comum. Para isso, $n-1$ linhas não são suficientes, mas n linhas bastam, o que se tem que fazer é criar um ciclo através de todas as cidades. Com isso, a nova tarefa é encontrar um ciclo com n arestas passando pelos n vértices de modo que o custo total de construir suas arestas é mínimo.

Esse problema é bastante conhecido na Otimização Matemática, é chamado Problema do Caixeiro Viajante. Neste caso, o método guloso não retorna a melhor solução possível (LOVÁSZ, 2013).

Retornando ao problema de se encontrar a árvore de custo mínimo. No contexto de árvores geradoras, o algoritmo guloso é conhecido como *Algoritmo de Kruskal*. A árvore obtida pelo método guloso será denotada por F.

Teorema 1.2.12. *Seja F uma árvore construída pelo método guloso. Então qualquer outra árvore G custará no mínimo tanto quanto a árvore gulosa.*

Demonstração. Seja G uma árvore diferente da árvore gulosa F. Considere o processo de construção de F, e o passo quando primeiro se toma uma aresta de F que não é uma aresta de G. Seja α essa aresta. Se α for adicionada a G, obtém-se um ciclo C. Esse ciclo não está inteiramente contido em F, logo existe uma aresta f que não é aresta de F. Se for adicionada a aresta α a G e então remove-se f, obtém-se uma terceira árvore H. Será mostrado que H é no máximo tão cara quanto G. Isso quer dizer que α é no máximo tão cara quanto f. Suponha que f seja mais barata que α. Ora, pelo método otimista (ou método guloso) então f teria que ter sido escolhida ao invés de α, mas já que não escolhida, então foi descartada porque estaria formando um ciclo C' com as arestas de F já selecionadas. Mas todas essas arestas previamente selecionadas são arestas de G, pois inspeciona-se o passo em que a primeira aresta que não está em G foi adicionada a F. Como a própria f é uma aresta de G, segue que todas as arestas de C' são arestas de G, o que é impossível pois G é uma árvore. Portanto, f não pode ser mais barata que α e assim G não pode ser mais barata que H. Assim sendo, substitui-se G por H que não é mais cara, e que tem a vantagem de coincidir com a F em mais arestas. Isso implica que se H for diferente de F, repete-se o mesmo argumento, obtendo-se árvores que não são mais caras que G, e que coincidem com a F em mais e mais arestas. Como o número de arestas é finito, em um determinado passo se obterá a própria F, que não é mais cara

que a G. □

Retornando-se à questão de se encontrar um ciclo mais barato possível através de todas as cidades dadas: tem-se n cidades (vértices), e para quaisquer duas delas se tem o "custo"de se conectá-las diretamente. Tem-se que encontrar um ciclo com esses vértices tal que o custo do ciclo (a soma dos custos de suas arestas) seja o mínimo possível.

Este problema é um dos mais importantes na área de *Otimização Combinatória*, conhecido como *Problema do Caixeiro Viajante*, e pode aparecer sob vários aspectos diferentes. Este nome vem da versão para o problema onde um vendedor viajante tem que visitar todas as cidades na região e depois retornar a própria casa,e obviamente ele deseja minimizar os custos dessa viagem. É fácil imaginar que este problema aparece em conexão com o "desenho"de rotas de entregas ótimas para os correios, rotas ótimas para coleta de lixo, etc.

A seguinte questão importante leva ao mesmo problema matemático. Uma máquina tem que perfurar um número de buracos em uma placa de circuitos impressos, e então retornar para o ponto inicial. Nesse caso, a quantidade importante é o tempo em que a máquina leva para mover a ponta perfurante de um buraco para o próximo, pois o tempo total que uma dada placa tem que gastar na máquina determina o número de placas que podem ser processadas em um dia. Portanto, se for tomado o tempo necessário para mover a ponta de um buraco para outro como o "custo"dessa aresta, precisa-se encontrar um ciclo com os buracos como vértices e com o custo mínimo.

O problema do caixeiro viajante é intimamente relacionado aos ciclos hamiltonianos. Antes de tudo, uma volta do caixeiro viajante é simplesmente um ciclo hamiltoniano no grafo completo sobre um conjunto de vértices. Mas há uma conexão mais interessante: *o problema de se descobrir se um dado grafo tem um ciclo hamiltoniano pode ser reduzido ao Problema do Caixeiro Viajante* (LOVÁSZ, 2013).

1.3 Redes

Agora faremos um embasamento no estudo de redes com o intuito de modelar e resolver o problema do caminho mínimo. Quando se associam valores e/ou arestas, o grafo designa-se geralmente por **rede**. Neste caso, fala-se em *nós* e *arcos* em vez de vértices e arestas, respectivamente. Uma rede pode ser interpretada por $G = (N, A, C)$,

em que (N, A) é um grafo e C corresponde ao conjunto de valores associados aos arcos "comprimentos": ao arco (i, j) está associado o valor c_{ij}. De uma maneira geral, os conceitos utilizados em grafos são extensíveis às redes.

Considere um caminho p de S para T, na rede G. O comprimento do caminho p corresponde à soma dos comprimentos dos arcos que pertencem àquele caminho, ou seja,

$$C(p) = \sum_{(i,j)\in p} c_{ij}$$

O conjunto de todos os caminhos de S para T, numa rede qualquer G será identificado por P.

Define-se **árvore mínima** (**árvore de caminhos mais curtos**) com origem em S, como a árvore que contém todos os nós de N acessíveis a partir de S, em que para cada nó n_2 é o caminho mais curto (de comprimento mínimo) na rede G que liga S a n_2.

1.3.1 O Problema do Caminho Mínimo

Os problemas de caminho mais curto (ou caminho mínimo), são fundamentais e frequentes quando se estuda redes de transportes e de comunicação. Este problema surge quando se pretende determinar o caminho mais curto, entre um ou vários pares de nós de uma rede.

Existem três tipos de problemas de caminho mais curto:

(i) de um nó para outro;

(ii) de um nó para todos os outros;

(iii) entre todos os pares de nós;

Sejam S e T dois nós de uma rede $G = (N, A, C)$, em que a cada arco é associado apenas um valor (chamaremos de comprimento do arco). O comprimento de um caminho de S para T, é a soma dos comprimentos dos arcos que o compõem.

O problema do caminho mais curto entre os nós S e T, tem por objetivo determinar o caminho de valor mínimo existente em P, ou seja, determinar $p \in P$ tal que $C(p) \leq C(q)$, $\forall q \in P$.

Algumas observações relacionadas com este tipo de problema:

i) o comprimento de um caminho é maior do que o de qualquer dos seus subcaminhos;

ii) qualquer subcaminho de um caminho mais curto, é ele próprio um caminho mais curto;

iii) para uma rede com n nós, qualquer caminho mais curto tem no máximo $n-1$ arcos (no caminho mais curto entre dois nós, não existem nós repetidos);

Matematicamente, este problema pode ser formulado como um problema de programação inteira da seguinte maneira:

$$Z = \sum_{i \in N} \sum_{j \in N} c_{ij} x_{ij}$$

$$h_1 = \sum_{j \in N} x s_j - 1$$

$$h_2 = \sum_{i \in N} x_{ij} - \sum_{k \in N} x_{jk}, \forall j \in N - \{S, T\}$$

$$h_3 = \sum_{i \in N} x_{iT} - 1$$

onde $h_1 = h_2 = h_3 = 0$ e

$$x_{ij} = \begin{cases} 1, \text{ se (i,j) pertence ao caminho} \\ 0, \text{ se (i,j) não pertence ao caminho} \end{cases}$$

Existem vários algoritmos eficientes para resolver problemas de caminho mais curto. Dentre eles um dos mais conhecidos é o de Dijkstra.

1.3.2 Algoritmo de Dijkstra

Este algoritmo, que foi apresentado por Dijkstra e que só pode ser aplicada a redes cujos arcos têm associados comprimentos não negativos, baseia-se num processo de rotulação dos nós da rede e classificação dos respectivos rótulo. A cada nó i é atribuído um rótulo $[\xi_i, \pi_i]$, o qual pode ser permanente ou temporário. Isto quer dizer o seguinte:

- $[\xi_i, \pi_i]$ permanente, representa o caminho mais curto de S para i;
  - $\xi_i \leftarrow$ nó que antecede i no caminho mais curto de S para i;

  - $\pi_i \leftarrow$ valor do caminho mais curto de S para i;

- $[\xi_i, \pi_i]$ temporário, representa um caminho mais curto de S para i;

  - $\xi_i \leftarrow$ nó que antecede i no melhor caminho, até a etapa atual de S para i;
  - $\pi_i \leftarrow$ valor do melhor caminho até a etapa atual de S para i;

O rótulo temporário de um nó representa um limite superior da distância mais curta de S ao nó determinado, uma vez que o caminho que lhe está associado pode ser ou não o caminho mais curto.

O algoritmo consiste em rotular os nós da rede, começando por S (origem), de uma forma ordenada, segundo as distâncias de cada nó a S: escolher o nó com rótulo temporário com menor valor de π, que passa a ser um nó com rótulo permanente, para depois serem varridos todos os nós adjacentes, de forma a atualizar os rótulos destes (temporários). O algoritmo termina quando não existirem nós com rótulos temporários. Inicialmente apenas o nó S é permanente, sendo os restantes temporários.

Algoritmo de Dijkstra

Passo 1

$[\xi_S, \pi_S] = [S, 0]$ *(caminho mais curto para S custa 0 e não tem nós intermediários)*

$[\xi_i, \pi_i] = [S, C_{S,i}]$, $\forall i \in N - \{S\}$ e $(S, i) \in A$ *(os nós adjacentes a S são rotulados com $\xi_i = S$ e $\pi_i = C_{S,i}$ que é o custo do arco (S, i))*

$[\xi_S, \pi_S] = [S, \infty]$, $\forall i \in N - \{S\}$ e $(S, i) \in \complement A$ *(os nós que não são adjacentes a S são rotulados com $\xi_i = S$ e $\pi_i = \infty$ para representar que não existe o arco (S, i))*

Temporários $= N - \{S\}$ *(Temporários é o conjunto de nós com rótulos temporários)*

Permanentes $= \{S\}$ *(Permanentes é o conjunto de nós com rótulos permanentes)*

Passo 2

Se Temporários $= \emptyset$ *(todos os nós tem rótulos permanentes)* então

  STOP *(critério de parada)*

Senão

  $v \in Temporarios$ é tal que π_v é mínimo *($v : \pi_v = min\{\pi_x, x \in Temporarios\}$)*

  $Temporarios = Temporarios - \{v\}$ *(v deixa de ser temporário)*

  $Permanentes = Permanentes \cup \{v\}$ *(v passa a ser permanente)*

Passo 3

Para todo $j \in N$ tal que $(v, j) \in A$ e $j \in Temporarios$ faz

Se $\pi_v + C_{v,j} < \pi_j$ então

$\pi_j = \pi_v + C_{v,j}$ *(o custo atual de se chegar a j é substituído pelo custo de se chegar a v mais o custo do arco (v, j))*

$\xi_j = v$ *(o nó antecessor de j no caminho mais curto de S para j passa a ser v)*

Senão

Regressar ao Passo 2

Fim do Algoritmo

O algoritmo apresentado, determina o caminho mais curto entre um dado nó S e todos os outros nós da rede. Portanto, no fim do algoritmo, para se verificar se existe caminho entre S e um nó qualquer v, basta analisar o valor de π_v: se $\pi_v = \infty$, então não existe caminho. Se existir caminho mais curto de S para v, este pode ser determinado percorrendo (em sentido inverso) a primeira parte dos rótulos dos nós (a saber ξ) de v até S, da seguinte forma:

$Caminho = \{v\}$

$i = v$

Enquanto $i \neq S$ faz

$Caminho \leftarrow Caminho \cup \{i\}$

Exemplo 1: Encontrar o caminho mínimo da rede apresentada na Figura 1.7 entre o nó $S = 1$ e todos os outros nós da rede.

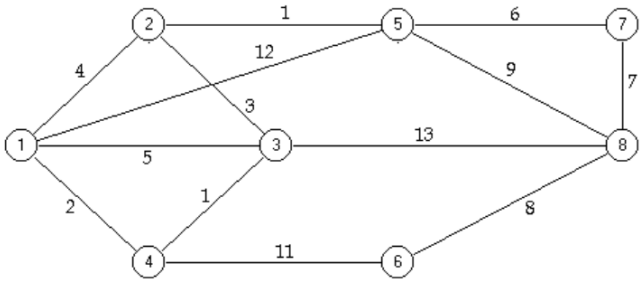

Figura 1.7: Rede com 8 nós

Passo 1:

Colocar rótulo permanente no nó 1 e rótulos temporários nos nós restantes.

Conforme nos diz o algoritmo, no início apenas a origem recebe um rótulo permanente.

$Permanentes = \{1\}$

$Temporarios = \{2, 3, 4, 5, 6, 7, 8\}$

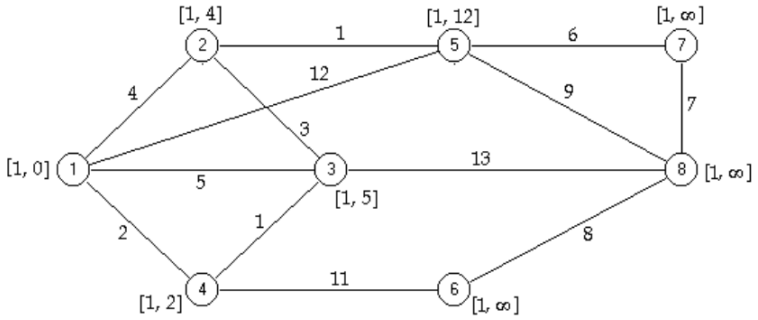

Figura 1.8: Rede após a primeira iteração, apenas a origem com nó permanente

Passo 2:

Como o conjunto de nós temporários não é vazio, então dentre os nós adjacentes à origem procura-se o nó com o menor valor de $\pi_j, j \in Temporarios$. Neste caso, o nó escolhido é $v = 4$, pois $\pi_4 = min\{\pi_2, \pi_3, \pi_4\}$.

$Permanentes = Permanentes \cup \{4\} = \{1, 4\}$

$Temporarios = Temporarios - \{4\} = \{2, 3, 5, 6, 7, 8\}$

Passo 3:

Varrer todos os nós adjacentes à 4, com rótulos temporários e atualizar os seus rótulos. Os nós adjacentes ao nó 4 são os nós 3 e 6.

$\pi_3 = min\{\pi_3, \pi_4 + C_{43}\} = min\{5, 2 + 1\} = 3 \Rightarrow [\xi_3, \pi_3] = [4, 3]$

$\pi_6 = min\{\pi_6, \pi_4 + C_{46}\} = min\{\infty, 2 + 11\} = 13 \Rightarrow [\xi_6, \pi_6] = [4, 13]$

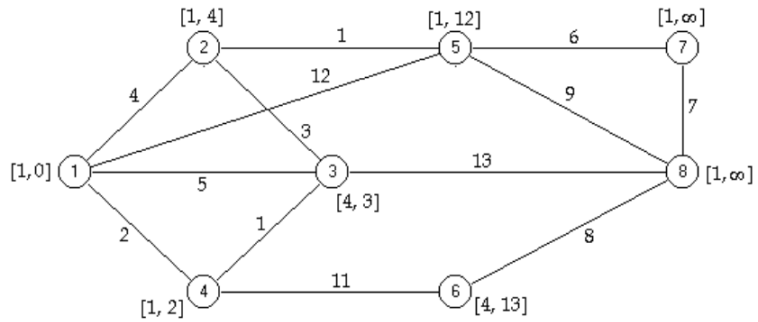

Figura 1.9: Rede com os rótulos dos nós 3 e 6 atualizados

Passo 2:

Como o conjunto de nós temporários não é vazio, então dentre os nós temporários procura-se o nó com o menor valor de $\pi_j, j \in Temporarios$. Neste caso, o nó escolhido é $v = 3$, pois $\pi_3 = min\{\pi_2, \pi_3, \pi_5, \pi_6, \pi_7, \pi_8\}$.

$Permanentes = Permanentes \cup \{3\} = \{1, 3, 4\}$

$Temporarios = Temporarios - \{3\} = \{2, 5, 6, 7, 8\}$

Passo 3:

Varrer todos os nós adjacentes à 3, com rótulos temporários e atualizar os seus rótulos. Os nós adjacentes ao nó 4 com rótulos temporários são os nós 2 e 8.

$\pi_2 = min\{\pi_2, \pi_3 + C_{32}\} = min\{4, 3 + 3\} = 4 \Rightarrow [\xi_2, \pi_2] = [1, 4]$ *(sem alteração)*

$\pi_8 = min\{\pi_8, \pi_3 + C_{38}\} = min\{\infty, 3 + 13\} = 16 \Rightarrow [\xi_8, \pi_8] = [3, 16]$

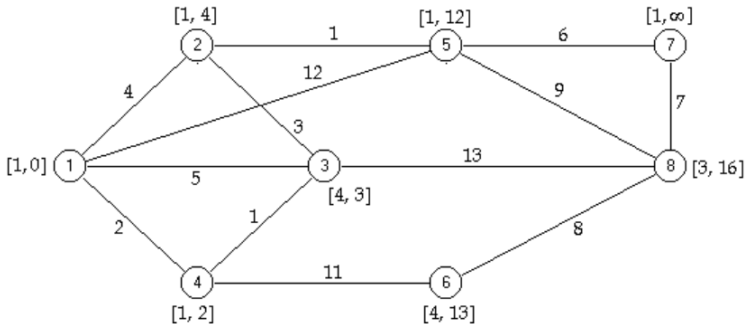

Figura 1.10: Rede com os rótulos dos nós 2 e 8 atualizados

Passo 2:

Como o conjunto de nós temporários não é vazio, então dentre os nós temporários procura-se o nó com o menor valor de $\pi_j, j \in Temporarios$. Neste caso, o nó escolhido é $v = 2$, pois $\pi_2 = min\{\pi_2, \pi_5, \pi_6, \pi_7, \pi_8\}$.

$Permanentes = Permanentes \cup \{2\} = \{1, 2, 3, 4\}$

$Temporarios = Temporarios - \{3\} = \{5, 6, 7, 8\}$

Passo 3:

Varrer todos os nós adjacentes à 2, com rótulos temporários e atualizar os seus rótulos. O nó adjacentes ao nó 2 com rótulo temporário é apenas o nó 5.

$\pi_5 = min\{\pi_5, \pi_2 + C_{25}\} = min\{12, 4 + 1\} = 5 \Rightarrow [\xi_5, \pi_5] = [2, 5]$

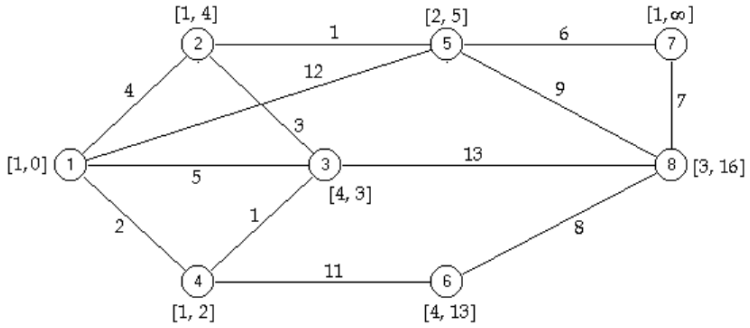

Figura 1.11: Rede com o rótulo do nó 5 atualizado

Passo 2:

Como o conjunto de nós temporários não é vazio, então dentre os nós temporários procura-se o nó com o menor valor de $\pi_j, j \in Temporarios$. Neste caso, o nó escolhido é $v = 5$, pois $\pi_5 = min\{\pi_5, \pi_6, \pi_7, \pi_8\}$.

$Permanentes = Permanentes \cup \{5\} = \{1, 2, 3, 4, 5\}$

$Temporarios = Temporarios - \{5\} = \{6, 7, 8\}$

Passo 3:

Varrer todos os nós adjacentes à 5, com rótulos temporários e atualizar os seus rótulos. Os nós adjacentes ao nó 5 com rótulos temporários são os nós 7 e 8.

$\pi_7 = min\{\pi_7, \pi_5 + C_{57}\} = min\{\infty, 5 + 6\} = 11 \Rightarrow [\xi_7, \pi_7] = [5, 11]$

$\pi_8 = min\{\pi_8, \pi_5 + C_{58}\} = min\{16, 5 + 9\} = 14 \Rightarrow [\xi_8, \pi_8] = [5, 14]$

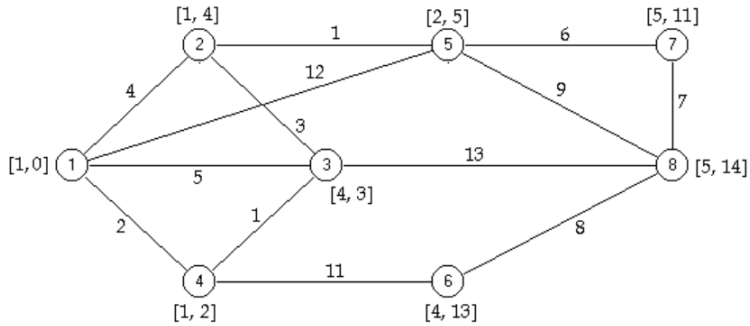

Figura 1.12: Rede com os rótulos dos nós 7 e 8 atualizados

Passo 2:

Como o conjunto de nós temporários não é vazio, então dentre os nós temporários procura-se o nó com o menor valor de $\pi_j, j \in Temporarios$. Neste caso, o nó escolhido é $v = 7$, pois $\pi_7 = min\{\pi_6, \pi_7, \pi_8\}$.

$Permanentes = Permanentes \cup \{7\} = \{1, 2, 3, 4, 5, 7\}$

$Temporarios = Temporarios - \{7\} = \{6, 8\}$

Passo 3:

Varrer todos os nós adjacentes à 7, com rótulos temporários e atualizar os seus rótulos. O nó adjacente ao nó 7 com rótulo temporário é apenas o nó 8.

$\pi_8 = min\{\pi_8, \pi_7 + C_{78}\} = min\{14, 11 + 7\} = 14 \Rightarrow [\xi_8, \pi_8] = [5, 14]$ *(sem alteração)*

Passo 2:

Como o conjunto de nós temporários não é vazio, então dentre os nós temporários procura-se o nó com o menor valor de $\pi_j, j \in Temporarios$. Neste caso, o nó escolhido é $v = 6$, pois $\pi_6 = min\{\pi_6, \pi_8\}$.

$Permanentes = Permanentes \cup \{6\} = \{1, 2, 3, 4, 5, 6, 7\}$

$Temporarios = Temporarios - \{6\} = \{8\}$

Passo 3:

Varrer todos os nós adjacentes à 6, com rótulos temporários e atualizar os seus rótulos. O nó adjacente ao nó 6 com rótulo temporário é apenas o nó 8.

$\pi_8 = min\{\pi_8, \pi_6 + C_{68}\} = min\{14, 13 + 8\} = 14 \Rightarrow [\xi_8, \pi_8] = [5, 14]$ *(sem alteração)*

Passo 2:

Como o conjunto de nós temporários não é vazio, então dentre os nós temporários procura-se o nó com o menor valor de $\pi_j, j \in Temporarios$. Neste caso, o nó escolhido é $v = 8$, pois $\pi_8 = min\{\pi_8\}$.

$Permanentes = Permanentes \cup \{8\} = \{1, 2, 3, 4, 5, 6, 7, 8\}$

$Temporarios = Temporarios - \{8\} = \emptyset$

Passo 3:

Não existem nós adjacentes a 8 com rótulos temporários.

Passo 2:

$Temporarios = \emptyset \Rightarrow$ Fim do algoritmo *(critério de parada)*

O caminho mais curto do nó 1 para todos os demais nós é apresentado na Figura 1.13.

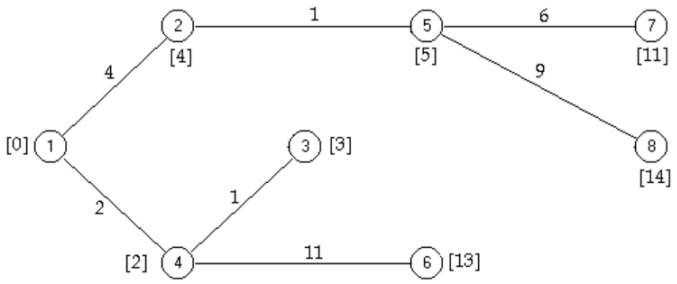

Figura 1.13: Árvore mínima

2 O equilíbrio de tráfego e sua equivalência com complementaridade não linear

Neste capítulo faremos um estudo sobre *Equilíbrio de Tráfego*. Com o estudo e aprendizado obtido no Capítulo 1, apresentaremos uma caracterização para o Equilíbrio de Tráfego.

Concluiremos com a equivalência entre o Problema de Equilíbrio de Tráfego e o Problema da Complementaridade Não Linear.

2.1 Equilíbrio de Tráfego por Complementaridade - Formulação Matemática

Nesta seção apresentamos a definição do problema de equilíbrio de tráfego. Em uma rede os nós podem representar pontos de origem-destino (denotaremos simplesmente por O-D) ou as intersecções de ruas e os arcos representam ruas. Nos referimos ao j-ésimo caminho do i-ésimo par O-D como o caminho j^i; ao fluxo neste caminho denotaremos por F_j^i e ao custo unitário correspondente como $C_j^i(F)$.

As demandas no i-ésimo par O-D serão denotadas por d_i. Com relação aos arcos, nos referimos ao k-ésimo arco da rede como arco k, ao fluxo neste arco como f_k e ao custo unitário correspondente como $c_k(f)$.

Para isso, vamos agora enunciar a Lei de equilíbrio de Wardrop:

"Num equilíbrio, para cada par origem-destino, o custo de viagem em todas as rotas usadas é igual e, este custo é menor ou igual ao custo de viagem das rotas não usadas".

Em outras palavras, se a distribuição está em equilíbrio, então:

i) se um caminho é escolhido para um par origem destino, então ele possui o menor custo possível;

ii) se um caminho possui custo superior ao de qualquer outro caminho, então ele não será escolhido.

Em notação matemática, o modelo de equilíbrio é definido numa rede de transporte (N, A, C) com n nós, k arcos dirigidos e um conjunto I de pares de nós origem-destino. O modelo de equilíbrio de tráfego é definido como sendo:

$$[C_j^i - u_i]F_j^i = 0, j \in J^i, i \in I \tag{2.1}$$

$$C_j^i(F) - u_i \geq 0, j \in J^i, i \in I \tag{2.2}$$

$$\sum_{j \in J^i} F_j^i - d_i(u) = 0, i \in I \tag{2.3}$$

$$F \geq 0 \tag{2.4}$$

$$u \geq 0 \tag{2.5}$$

Nesta formulação temos:

- $I.\equiv$. conjunto de pares origem-destino (O-D);
- $J^i.\equiv$. conjunto de caminhos disponíveis para o i-ésimo par O-D;
- $F_j^i.\equiv$. fluxo no j-ésimo caminho do i-ésimo par O-D;
- $F.\equiv$. vetor de (F_j) com dimensão $n_1 = \sum_i |J^i|$, ou seja, o número total de caminhos;
- $u_i.\equiv$. custo mínimo para o i-ésimo par O-D;
- $u.\equiv$. vetor de (u_i) com dimensão $n_2 = |I|$, ou seja, número total de pares O-D;
- $d_i(u).\equiv$. função demanda para o i-ésimo par O-D;
- $d_i(u) : \mathbb{R}_+^{n_2} \to \mathbb{R}_+$;
- $C_j^i.\equiv$. função custo para o j-ésimo caminho do i-ésimo par O-D;

- $C_j^i(F) : \mathbb{R}_+^{n_1} \to \mathbb{R}_+$

Seja $J. \equiv .\{J^i, i \in I\}$ denotando o conjunto de todos os caminhos disponíveis na rede e vamos assumir que a rede é conexa, ou seja, $|J^i| \geq 1, \forall i \in I$.

As equações (2.1) e (2.2) no modelo de equilíbrio apresentado requerem que, para qualquer par origem-destino, o custo de viagem em todos os caminhos $j \in J^i$ escolhidos sejam o mesmo e igual a u_i, que é menor ou igual ao custo de viagem em qualquer caminho não escolhido. A equação (2.3) exige que a soma dos fluxos ao longo de diferentes caminhos entre qualquer par origem-destino seja igual a demanda total $d_i(u)$. E as equações (2.4) e (2.5) requerem que o fluxo nos caminhos e o custo mínimo devem ser não negativos.

2.2 Equivalência com um problema de Complementaridade Não Linear

Nesta seção definiremos o que é o problema de complementaridade não linear e faremos a sua equivalência com o problema do equilíbrio de tráfego.

Seja $G : \mathbb{R}^n \to \mathbb{R}^n$, $G(x) = (G_1(x), G_2(x), \ldots, G_n(x))$. O problema de Complementaridade Não Linear é encontrar um vetor $x \in \mathbb{R}^n$ satisfazendo:

$$G(x) \geq 0,\ x \geq 0 \text{ e } xG(x) = 0$$

Teorema 2.2.1. *O problema de equilíbrio de tráfego apresentado na Seção* 2.1 *pode ser formulado como um problema de complementaridade não linear.*

Demonstração. Seja $x = [F\ u]^T \in \mathbb{R}^n$ com $n = n_1 + n_2$. Sejam ainda:

$$h_j^i(x) = C_j^i(F) - u_i, j \in J^i, i \in I$$

$$g_i(x) = \sum_{j \in J^i} F_j^i - d_{i(u)}, i \in I$$

Vamos definir $G(x)$ como sendo:

$$G(x) = \begin{bmatrix} h_j^i(x) \\ g_i(x) \end{bmatrix} \in \mathbb{R}^n$$

Considere agora o problema de complementaridade não linear:

$$\begin{gathered} h_j^i(x)F_j^i = 0, j \in J^i, i \in I \\ h_j^i \geq 0, j \in J^i, i \in I \\ g_i(x)u_i = 0, i \in I \\ g_i(x) \geq 0 \\ x \geq 0 \end{gathered} \tag{2.6}$$

Observe que qualquer solução $\overline{x} = [\overline{F}\ \overline{u}]^T$ apresentado na Seção 2.1 satisfaz $g_i(x) = 0$, para todo $i \in I$, logo essa solução também é solução para o problema de complementaridade (2.6)

Suponha agora, que para todo $j \in J^i$, que $C_j^i : \mathbb{R}_+^{n_1} \to \mathbb{R}_+$ é uma função positiva. Suponha também, que para todo $i \in I$, que $d_i : \mathbb{R}_+^{n_2} \to \mathbb{R}_+$ é uma função não negativa. Vamos mostrar que qualquer solução de (2.6) é também solução do problema de equilíbrio de tráfego apresentado na Seção 2.1. Suponha que não, ou seja, existe um $x = [F\ u]^T$ tal que

$$g_i(x) = \sum_{j \in J^i} F_j^i - d_i(u) > 0$$

para algum $i \in I$.

Então $g_i(x)u_i = 0$, implica $u_i = 0$. Como d_i é não negativa, então

$$\sum_j F_j^i > d_i(u) \geq 0$$

o que implica que $F_j^i > 0$ para algum $j \in J^i$ e algum $i \in I$. Mas para estes j e i particulares, a equação $h_j^i(x)F_j^i = 0$ implica que $h_j^i = C_j^i(F) - u_i = 0$ ou ainda, $C_j^i(F) = u_i$.

Como $u_i = 0$, então $C_j^i(F) = 0$, contradizendo a hipótese que $C_j^i(F) > 0$.

Portanto, toda solução do problema de complementaridade não linear é solução do problema de equilíbrio de tráfego, ou seja, ambos são equivalentes. □

Todo problema de complementaridade não linear tem solução única (OLIVEIRA, 1989), ou seja, que os fluxos nos arcos é único. O fluxo nos caminhos não precisa

ser único, pois pode existir várias combinações de fluxos nos caminhos que correspondem ao mesmo conjunto de fluxo nos arcos (OLIVEIRA, 1989).

3 Elementos da Teoria das Filas

Neste capítulo será feito uma fundamentação teórica sobre a Teoria das Filas, objetivando caracterizar os pesos não negativos dos arcos de uma rede e assim utilizar os resultados obtidos sobre o Problema do Caminho Mínimo.

3.1 Teoria das Filas

Neste capítulo será feito um embasamento teórico sobre Teoria das Filas e Fluxo em Redes, com o intuito de uma melhor compreensão para o problema que será apresentado no próximo capítulo.

A Teoria das Filas é um setor da Pesquisa Operacional que utiliza conceitos básicos de processos estocásticos e de matemática aplicada para analisar o fenômeno de formação de filas e de suas características. Vem sendo desenvolvida com a finalidade de prever o comportamento das filas de modo a permitir o dimensionamento adequado das instalações, equipamentos e sua infra-estrutura.

Os principais resultados sobre esses temas são apresentados juntamente as definições necessárias para o entendimento dos mesmos.

3.1.1 Definições sobre Teoria das Filas

As filas são uma constante na nossa vida cotidiana. Nós a enfrentamos com bom ou mau humor, ou até com indiferença. O certo é que, no dia-a-dia, elas constituem algo desagradável, que nos conformamos em aceitar. Já o analista de transporte enfrenta problemas em que as filas surgem com implicações econômicas sérias, exigindo um tratamento racional do fenômeno.

Navios no porto esperando atracação, trens de carga aguardando linha, veículos em postos de pedágio são alguns dos exemplos de filas com implicações sérias de ordem operacional e econômica.

É necessário deixar claro de início que os modelos de fila nem sempre con-

seguem representar as situações reais com grande precisão. Muitas vezes as premissas necessárias ao desenvolvimento matemático dos modelos envolvem simplificações substanciais. Mesmo assim, há vantagem em desenvolver tais modelos, principalmente porque eles levam a um melhor entendimento das principais condicionantes do processo.

Uma fila é caracterizada por um processo de chegadas (pessoas, carros, navios, etc.) a um sistema de atendimento formado por uma ou mais unidades de serviço (boxes de pedágios, berços de atracação de navios, etc.).

No modelo estudado neste trabalho, a disciplina da fila obedece à ordem de chegada, não se considerando outros tipos de prioridade. Não há também desistências, pois os problemas de transportes aqui estudados constituem sistemas fechados, sem alternativas. As unidades podem ser atendidas individualmente (pedágio, porto, etc.) ou em grupos (pessoas no elevador, veículos num semáforo, etc.).

No caso de filas com atendimento individual é conveniente a adoção da simbologia sugerida por Erlang em 1909. Um modelo de fila desse tipo é representado por um símbolo, do tipo $X/Y/C$, onde:

a) a letra X representa o tipo do processo de chegadas. Para processo de Poisson coloca-se um M; para chegadas com intervalos formando um processo de renovações qualquer, utiliza-se o símbolo GI;

b) a letra Y indica o tipo de distribuição dos tempos de atendimento. M representa distribuição exponencial, E_k representa distribuição de Erlang de ordem k, D indica tempo constante de serviço (caso determinístico), G representa uma distribuição genérica;

c) a letra C representa o número de canais de atendimento em paralelo.

A fila $M/E_2/3$, por exemplo, corresponde a uma configuração com chegadas regidas por um processo de Poisson, tempos de atendimento regidos por uma distribuição de Erlang de ordem dois e com três posições de atendimento em paralelo.

Faz-se a distinção na análise de uma fila, entre o **sistema**, que engloba as unidades sendo atendidas e as que estão esperando, e a **fila**, que corresponde apenas às unidades que estão esperando atendimento. Assim é comum indicar por W o tempo gasto no sistema e por W_q o tempo de espera gasto na fila.

Uma relação genérica permite calcular o tempo médio de espera na fila, uma vez calculado o seu comprimento médio.

Essa expressão é dada da seguinte maneira:

$$\lambda \overline{W}_q = \overline{L}_q \tag{3.1}$$

onde:

$\overline{W}_q :=$ tempo médio de espera na fila;

$\overline{L}_q :=$ comprimento médio da fila;

$\lambda :=$ fluxo médio de chegadas (unidades por unidade de tempo).

A equação (3.1) é válida para a fase de regime, quando essa fase existir. A **fase de regime** corresponde à situação limite, atingida após decorrido um tempo suficientemente longo, em que a distribuição probabilística dos parâmetros da fila passa a ser constante e, portanto, independente do tempo. A fase inicial, que decorre desde o início do processo até a situação de regime, é denominada **fase transiente**.

Definição 3.1.1. O índice de congestionamento do sistema é dado pela relação entre a demanda média num certo intervalo e a capacidade média de atendimento do sistema nesse intervalo. Denotaremos este índice por ρ.

Por exemplo, se chegam em média 1800 veículos por hora num posto de pedágio, dotado de quatro boxes, durando o atendimento de cada veículo 6 segundos em média, tem-se:

$$\rho = \frac{1800}{4(3600/6)} = 0,75.$$

3.1.2 Distribuição de Poisson

Nesta seção será analisada uma distribuição bastante utilizada em Teoria das Filas, a distribuição de Poisson.

A probabilidade de ocorrência de uma variável discreta n, quando regida por uma distribuição de Poisson é dada por:

$$P(n) = \frac{\lambda^n}{n!} e^{-\lambda} \tag{3.2}$$

onde $n = 0, 1, 2, \ldots$ e $\lambda > 0$.

Para calcular a esperança e a variância da distribuição de Poisson é conveniente determinar a sua transformada Z:

$$P(Z) = \sum_{n=0}^{\infty} P(n)Z^n = \sum_{n=0}^{\infty} \frac{(\lambda Z)^n}{n!} e^{-\lambda} = e^{\lambda(Z-1)} \tag{3.3}$$

Aplicando-se as propriedades da transformada Z tem-se:

$$E[n] = \left.\frac{d}{dZ}\right|_{Z=1} = \lambda e^{\lambda(Z-1)}\big|_{Z=1} = \lambda \tag{3.4}$$

e usando também o fato de que:

$$Var[n] = \left.\frac{d^2P(Z)}{dZ^2}\right|_{Z=1} + E[n] - E^2[n] \tag{3.5}$$

ou seja,

$$Var[n] = \lambda^2 e^{\lambda(Z-1)}\big|_{Z=1} + \lambda - \lambda^2 = \lambda \tag{3.6}$$

Conclui-se, portanto, que o parâmetro λ representa a esperança e a variância da distribuição.

3.1.3 Distribuição Exponencial

É uma distribuição de variável aleatória contínua com função de densidade de probabilidade dada por

$$f(x) = \lambda e^{-\lambda x} \tag{3.7}$$

com $\lambda > 0$ e $x \geq 0$. A transformada de Laplace da distribuição exponencial é obtida da seguinte maneira:

$$F(p) = \int_0^{\infty} e^{-px} f(x)dx = \lambda \int_0^{\infty} e^{-(\lambda+p)x} dx = \frac{\lambda}{\lambda + p} \tag{3.8}$$

Através da transformada de Laplace podem ser obtidas a expectância e a variância:

$$E[x] = -\left.\frac{d}{dp}F(p)\right|_{p=0} = -\left.\frac{-\lambda}{(\lambda+p)^2}\right|_{p=0} = \frac{1}{\lambda} \tag{3.9}$$

$$Var[x] = \left.\frac{d^2}{dp^2}F(p)\right|_{p=0} - E^2[x] = \left.\frac{2\lambda}{(\lambda+p)^3}\right|_{p=0} - \frac{1}{\lambda^2} = \frac{1}{\lambda^2} \tag{3.10}$$

3.1.4 Características do processo de chegada e atendimento

As chegadas dos indivíduos podem ser estudadas de duas maneiras diferentes:

- analisando os intervalos entre as chegadas sucessivas;
- analisando o número de chegadas durante intervalos de tempo com duração pré determinada.

A análise dos intervalos fornece mais informações sobre o processo de chegadas e por isso é recomendável. O segundo método pode ser aplicado quando existem indícios claros de que o processo de chegadas é de Poisson.

Vamos admitir que se mediram n intervalos sucessivos, assumindo os valores $t_1, t_2, \ldots, t_n$. Seja t o tempo total, dado por:

$$t = \sum_{i=1}^{n} t_i$$

A origem do intervalo t_1 (e consequentemente de t) pode ou não coincidir com uma chegada. A razão média de chegadas λ é calculada dividindo-se o número de chegadas n pelo intervalo total:

$$\lambda = \frac{n}{t}$$

Dizemos que o processo de chegadas é um processo de renovações quando os intervalos entre as chegadas sucessivas forem variáveis aleatórias independentes e identicamente distribuídas.

Vamos considerar apenas processos de renovações. Para garantir as propriedades dos processos de renovações é preciso escolher períodos de tempo durante os quais

a razão média de chegadas não varie significativamente, isto é, o comportamento médio do sistema durante o período considerado deve permanecer praticamente inalterado.

Vamos considerar agora a distribuição que rege os intervalos entre as chegadas, dada pela função densidade de probabilidade $f(t)$. A probabilidade de que a próxima chegada se dê no intervalo $(t, t+dt)$ após a chegada anterior é dada por $f(t)dt$. Se considerarmos um instante t, cuja origem é independente de qualquer chegada, a probabilidade de ocorrência de uma chegada é dada por λdt.

Vamos definir agora a probabilidade $Q_n(t)$ de haver n chegadas durante um intervalo de tempo t, medido a partir de uma chegada. Se o início do intervalo t não coincidir necessariamente com uma chegada, tem-se então a probabilidade $P_n(t)$.

A expressão $Q_n(t)$ pode ser obtida através da fórmula de recorrência (NOVAES, 1975):

$$Q_n(t) = \int_0^t f(\tau) Q_{n-1}(t-\tau) d\tau$$

que corresponde a considerar $n-1$ chegadas durante o tempo $t-\tau$ e uma chegada no fim do intervalo t.

Para determinar $Q_0(t)$ basta lembrar que a probabilidade de não ocorrência de chegadas durante um intervalo t, medido após a última chegada, é igual ao complemento da probabilidade de haver chegadas naquele intervalo:

$$Q_0(t) = \int_t^{+\infty} f(\tau) d\tau = 1 - \int_0^t f(\tau) d\tau$$

No caso de não haver vinculação da origem do intervalo t com uma chegada, temos então:

$$P_n(t) = \int_0^{t-x} f(\tau) Q_{n-1}(t-x-\tau) d\tau \int_0^t \lambda dx \tag{3.11}$$

Esta expressão é formada pelo produto de três probabilidades:

- uma chegada no instante x com probabilidade λx;
- uma chegada no instante τ, após o instante x, com probabilidade $f(\tau)d\tau$;
- $n-1$ chegadas no intervalo $t-x-\tau$, com probabilidade $Q_{n-1}(t-x-\tau)$.

O produto de probabilidade é integrado em relação a τ e a x. A equação 2.11 pode ser reduzida à seguinte:

$$P_n(t) = \lambda \int_0^t Q_0(x) Q_{n-1}(t-x) dx \tag{3.12}$$

Nos casos gerais, a probabilidade $Q_n(t)$ é diferente de $P_n(t)$, mas para processos de Poisson elas são iguais, o que nos leva a uma série de propriedades bastante úteis na análise de modelo de filas.

Para que exista um processo de Poisson é preciso que sejam satisfeitas as seguintes premissas:

- As chegadas devem ser independentes e as características probabilísticas do sistema não devem se alterar com o tempo. Em particular, λ deve permanecer constante;
- A probabilidade de mais de uma chegada no intervalo infinitesimal dt é desprezível. Assim:

$$P_1(dt) = \lambda dt \tag{3.13}$$

$$P_n(dt) = 0, n \geq 2 \tag{3.14}$$

E consequentemente, como

$$\sum_{n=0}^{1} P_n(dt) = 1$$

tem-se

$$P_0(dt) = 1 - P_1(dt) = 1 - \lambda dt \tag{3.15}$$

A partir dessas premissas pode-se deduzir as equações que regem o processo de Poisson. Vamos admitir que chegaram n indivíduos no intervalo $t + dt$, cuja probabilidade é $P_n(t + dt)$. A probabilidade de haver chegado n indivíduos no intervalo $t + dt$ é igual à soma de duas probabilidades:

- terem chegado n indivíduos no período t e nenhum no período dt;
- terem chegado $n-1$ indivíduos no período t e um elemento no intervalo dt.

Assim sendo:

$$P_n(t+dt) = P_n(t)(1-\lambda dt) + P_{n-1}(t)(\lambda dt) \tag{3.16}$$

válida para $n \geq 1$.

Se no período $t+dt$ não houver chegado nenhum elemento, a probabilidade correspondente será igual ao produto da probabilidade de não haver chegado nenhum elemento no período t, pela probabilidade de não haver chegado nenhuma unidade no intervalo dt, ou seja,

$$P_0(t+dt) = P_0(t)(1-\lambda dt) \tag{3.17}$$

Fazendo as operações algébricas necessárias, as expressões (3.16) e (3.17) resultam em:

$$\frac{P_n(t+dt) - P_n(t)}{dt} = -\lambda P_n(t) + \lambda P_{n-1}(t) \tag{3.18}$$

e

$$\frac{P_0(t+dt) - P_0(t)}{dt} = -\lambda P_0(t) \tag{3.19}$$

Aplicando o limite quando $dt \to 0$ os primeiros membros das expressões (3.18) e (3.19) se transformam nas derivadas de $P_n(t)$ e $P_0(t)$ em relação ao tempo, ou seja,

$$P_n^{'}(t) = -\lambda P_n(t) + \lambda P_{n-1}(t) \tag{3.20}$$

$$P_0^{'}(t) = -\lambda P_0(t) \tag{3.21}$$

Consideremos agora a transformada "Z" de $P_n(t)$ (NOVAES, 1975):

$$P(Z,t) = \sum_{n=0}^{\infty} P_n(t)Z^n = P_0(t) + P_1(t)Z + P_2(t)Z^2 + \ldots + P_n(t)Z^n + \ldots \tag{3.22}$$

Derivando a equação (3.22) em relação a t, obtemos:

$$\frac{\partial P(Z,t)}{\partial t} = \sum_{n=0}^{\infty} P_n^{'}(t) Z^n \tag{3.23}$$

Reescrevendo as equações (2.20) e (2.21) de forma a evidenciar diversas equações:

$$\begin{aligned} P_0^{'}(t) &= -\lambda P_0(t) \\ P_1^{'}(t) &= -\lambda P_1(t) + \lambda P_0(t) \\ P_2^{'}(t) &= -\lambda P_2(t) + \lambda P_1(t) \\ &\vdots \\ P_n^{'}(t) &= -\lambda P_n(t) + \lambda P_{n-1}(t) \end{aligned} \tag{3.24}$$

Multiplicando, membro a membro, as equações acima por $Z^0, Z^1, Z^2, \ldots$ obtemos:

$$\begin{aligned} P_0^{'}(t) Z^0 &= -\lambda P_0(t) Z^0 \\ P_1^{'}(t) Z^1 &= -\lambda P_1(t) Z^1 + \lambda P_0(t) Z^1 \\ P_2^{'}(t) Z^2 &= -\lambda P_2(t) Z^2 + \lambda P_1(t) Z^2 \\ &\vdots \\ P_n^{'}(t) Z^n &= -\lambda P_n(t) Z^n + \lambda P_{n-1}(t) Z^{n-1} \end{aligned} \tag{3.25}$$

Somando, membro a membro, e aplicando o limite quando $n \to \infty$, tem-se:

$$\frac{\partial P(Z,t)}{\partial t} = -\lambda P(Z,t) + \lambda Z P(Z,t) \tag{3.26}$$

ou

$$\frac{\partial P(Z,t)}{\partial t} - \lambda (Z-1) P(Z,t) = 0 \tag{3.27}$$

Essa equação diferencial tem solução geral do tipo:

$$P(Z,t) = a e^{\lambda (Z-1) t} \tag{3.28}$$

onde a é uma constante. A transformada "Z" tem a propriedade:

$$P(Z,t)|_{Z=1} = 1 \tag{3.29}$$

válida para distribuições de probabilidade (NOVAES, 1975). Aplicando $Z = 1$ em (3.28) obtemos $a = 1$, donde concluímos que

$$P(Z,t) = e^{\lambda(Z-1)t} \tag{3.30}$$

Para se obter as probabilidades $P_0(t), P_1(t), \ldots$ deriva-se sucessivamente a transformada $P(Z,t)$ em relação a Z e utilizando a expressão

$$P_n(t) = \frac{1}{n!} \left. \frac{\partial^{(n)} P(Z,t)}{\partial Z^n} \right|_{Z=0} \tag{3.31}$$

Assim sendo:

$$\begin{aligned} P_0(t) &= e^{-\lambda t} \\ P_1(t) &= \lambda e^{-\lambda t} \\ &\vdots \\ P_n(t) &= \frac{(\lambda t)^n e^{-\lambda t}}{n!} \end{aligned} \tag{3.32}$$

Conforme foi visto na subseção 3.1.2 a expressão acima representa uma distribuição de Poisson quando se fixa um valor de t, pois pode-se fazer $\lambda' = \lambda t$, obtendo-se

$$P_n(t) = \frac{(\lambda')^n e^{-\lambda'}}{n!} \tag{3.33}$$

Uma característica importante no processo de Poisson é a distribuição dos intervalos entre chegadas sucessivas. Para determinar essa distribuição vamos considerar uma chegada e marcar, a partir deste instante, um intervalo t qualquer.

A probabilidade de que a próxima chegada se dê após o intervalo t é igual à probabilidade de haver zero chegadas no intervalo t, ou seja,

$$P_0(t) = e^{-\lambda t} \tag{3.34}$$

Denominando de τ o intervalo entre as chegadas sucessivas, a probabilidade dada pela equação (3.34) corresponde a probabilidade de se ter $\tau > t$, ou seja,

$$F(t) = p(\tau \leq t) = 1 - p(\tau > t) = 1 - e^{-\lambda t} \tag{3.35}$$

Derivando a equação (3.35) em relação à t obtemos

$$f(t) = \lambda e^{-\lambda t} \tag{3.36}$$

Vê-se portanto, que os intervalos de tempo entre as chegadas sucessivas num processo de Poisson são regidos por uma distribuição exponencial.

3.1.5 A Fila $M/M/1$

Vamos inicialmente apresentar a solução através de equações diferenciais e posteriormente como resultante de um processo de geração e extinção. As seguintes premissas são admitidas:

a) As chegadas são regidas por um processo de Poisson, com razão média constante igual a λ;

b) O tempo de atendimento é regido por distribuição Exponencial de parâmetro μ;

c) As unidades são atendidas na ordem de chegada, não havendo desistências.

Seja $P_n(t)$ a probabilidade de haver n unidades no sistema no instante t. Admitindo-se inicialmente $n \neq 0$, pode-se analisar a evolução da fila dentro de um intervalo infinitesimal dt. Suponhamos que existem n elementos no sistema no instante $t + dt$. Considerando as várias combinações de eventos, podemos dizer que a ocorrência desse evento (n elementos no instante $t + dt$) é formada pela união dos seguintes eventos mutuamente excludentes:

i) Haver n elementos no sistema no instante t, não haver chegado nenhum elemento no intervalo dt e não ter sido liberado nenhum elemento nesse intervalo;

ii) Haver $n-1$ elementos no sistema no instante t, ter chegado um elemento no intervalo dt e não ter sido liberado nenhum;

iii) Haver $n + 1$ elementos no sistema no instante t, não ter chegado nenhum elemento no intervalo dt e ter sido liberado um elemento nesse intervalo;

iv) Combinações restantes não incluídas nos itens anteriores: essas combinações apresentam probabilidade de ocorrência desprezível porque implicam em mais de uma chegada e/ou saída dentro do intervalo infinitesimal dt, no caso de processos de Poisson esses eventos apresentam probabilidade nula.

Dessa forma, as três probabilidades de transição nulas, indicadas acima são apresentadas como segue:

i) Nenhuma chegada e nenhuma liberação no intervalo dt:

$$(1 - \lambda dt)(1 - \mu dt) = 1 - (\lambda + \mu)dt + \lambda\mu dt^2 \cong 1 - (\lambda + \mu)dt \tag{3.37}$$

ii) Uma chegada no intervalo dt e nenhuma liberação nesse intervalo:

$$\lambda dt(1 - \mu dt) = \lambda dt - \lambda\mu dt^2 \cong \lambda dt \tag{3.38}$$

iii) Nenhuma chegada no intervalo dt e uma liberação nesse intervalo:

$$(1 - \lambda dt)\mu dt = \mu dt - \lambda\mu dt^2 \cong \mu dt \tag{3.39}$$

Combinando as probabilidades acima, pode-se escrever:

$$P_n(t + dt) = P_n(t)[1 - (\lambda + \mu)dt] + P_{n-1}(t)\lambda dt + P_{n+1}(t)\mu dt \tag{3.40}$$

válida para $n \geq 1$.

Quando $n = 0$ a equação acima se altera, sendo necessária deduzi-la à parte. A probabilidade de não se registrar nenhuma unidade no sistema no instante $t + dt$ é formada pela soma das seguintes probabilidades:

i) Não haver nenhum elemento no instante t e não haver chegado nenhum no intervalo dt. Note que não cabe considerar a probabilidade ou não de liberação, visto que o sistema está ocioso. A probabilidade de não haver chegado nenhum elemento no intervalo dt é dada simplesmente por $(1 - \lambda dt)$;

ii) Haver um elemento no instante t e ter sido liberado dentro do intervalo dt. A probabilidade de liberação é dada por μdt.

Somando-se as duas probabilidades, obtemos:

$$P_0(t+dt) = P_0 dt[1-\lambda dt] + P_1(t)\mu dt \tag{3.41}$$

O sistema de equações definido por (2.40) e (2.41), após transposição do termo $P_n(t)$ e divisão por dt resulta em:

$$\frac{P_n(t+dt) - P_n(t)}{dt} = -(\lambda+\mu)P_n(t) + \lambda P_{n-1}(t) + \mu P_{n+1}(t) \tag{3.42}$$

e

$$\frac{P_0(t+dt) - P_0}{dt} = -\lambda P_0(t) + \mu P_1(t) \tag{3.43}$$

Aplicando o limite com $dt \to 0$, o primeiro membro das equações (3.42) e (3.43) se torna nas derivas de $P_n(t)$ e $P_0(t)$ em relação ao tempo:

$$P_n^{'}(t) = -(\lambda+\mu)P_n(t) + \lambda P_{n-1}(t) + \mu P_{n+1}(t) \tag{3.44}$$

e

$$P_0^{'}(t) = -\lambda P_0(t) + \mu P_1(t) \tag{3.45}$$

A solução do sistema de equações acima, para o caso transiente, é bastante complexa e de aplicação pouco prática. Na análise do modelo de filas considera-se, na maioria dos casos, a fase de regime. Admitindo que o processo seja ergódico, a distribuição de probabilidade $P_n(t)$ passa a ser constante e, portanto independente do tempo, quando $t \to \infty$. Dessa forma, a sua derivada em relação ao tempo se anula. O sistema de equações obtidos em (3.44) e (3.45) se reduz a

$$-(\lambda+\mu)\pi_n + \lambda\pi_{n-1} + \mu\pi_{n+1} = 0 \tag{3.46}$$

$$-\lambda\pi_0 + \mu\pi_1 = 0 \tag{3.47}$$

onde

$$\pi_n = \lim_{t\to\infty} P_n(t)$$

A resolução do sistema acima nos conduz (NOVAES, 1975):

$$\pi_n = \left(\frac{\lambda}{\mu}\right)^n \pi_0 \tag{3.48}$$

A probabilidade π_0 pode ser calculada através da equação:

$$\pi_0 + \pi_1 + \pi_2 + \ldots + \pi_n + \ldots = 1 \tag{3.49}$$

o que leva a (NOVAES, 1975)

$$\pi_0 = \frac{1}{1 + \left(\frac{\lambda}{\mu}\right) + \left(\frac{\lambda}{\mu}\right)^2 + \ldots} \tag{3.50}$$

Se $\frac{\lambda}{\mu} < 1$ a progressão geométrica do denominador de (3.50) apresenta soma finita e igual a $\frac{1}{1-\frac{\lambda}{\mu}}$. Assim sendo:

$$\pi_0 = 1 - \frac{\lambda}{\mu} \tag{3.51}$$

A distribuição de π_n e, portanto a probabilidade de haver n elementos no sistema é dado por:

$$\pi_n = \left(\frac{\lambda}{\mu}\right)^n \left(1 - \frac{\lambda}{\mu}\right) \tag{3.52}$$

desde que $\frac{\lambda}{\mu} < 1$, ou seja, desde que o índice de congestionamento do sistema, de acordo com a Definição 3.1.1 seja menor do que 1.

A distribuição definida por (3.52) representa uma distribuição Geométrica, que possui média e variância definidas como sendo

$$E[n] = \frac{\rho}{1-\rho} = \frac{\lambda}{\mu - \lambda} \tag{3.53}$$

$$Var[n] = \frac{\rho}{(1-\rho)^2} \tag{3.54}$$

A média apresentada em (3.53) representa o número médio de clientes no sistema (NS). Dividindo esta equação pelo valor de μ, obtemos o tempo médio na fila gasto por cada elemento (TF), o tempo que leva em média para que cada elemento seja atendido:

$$TF = \frac{\lambda}{\mu(\mu-\lambda)} \tag{3.55}$$

O número médio de elementos na fila (NF) pode ser calculado lembrando do fato de que só existe fila quando existe mais do que um elemento no sistema. Quando isso ocorre, o número de elementos na fila é igual ao número de elementos no sistema menos um, pois este está sendo atendido. Assim sendo:

$$NF = \sum_{n=2}^{\infty}(n-1)\pi_n = \frac{\rho^2}{1-\rho} = \frac{\lambda^2}{\mu(\mu-\lambda)} \tag{3.56}$$

A probabilidade do sistema estar ocioso é π_0, ou seja, $\frac{\mu-\lambda}{\lambda}$ e a do sistema estar ocupado é dada por $\frac{\lambda}{\mu}$.

O tempo médio que um cliente gasta no sistema (TS) é dado por:

$$TS = \frac{1}{\mu-\lambda} \tag{3.57}$$

4 Caminhos Mínimos em um Sistema de Tráfego Urbano Semaforizado

Conforme foi visto na Subseção 3.1.5, dada uma fila na qual a razão média de chegadas é regida por um processo de Poisson com taxa λ, sabemos que o tempo de atendimento é dado por uma distribuição Exponencial com parâmetro μ. Conhecido o λ, podemos estimar um μ tal que $\rho = \frac{\lambda}{\mu} < 1$ e assim utilizar os resultados obtidos, como por exemplo o cálculo do tempo médio do cliente no sistema, dada pela equação (3.57).

Em uma rede (N, A, C) conexa, podemos sem perda de generalidade considerar que cada nó é um ponto de atendimento. Considere um arco (i, j). Se um cliente está no nó i e deseja ir para o nó j, o custo do deslocamento pode ser considerado como o tempo que o cliente levará até terminar de ser atendido, ou seja, o tempo médio em que o usuário permanece no sistema e "chega" ao nó j onde o cliente pode permanecer ou não, dependendo de onde for o seu destino.

Assim sendo, conhecida a média de chegadas λ_0 em um nó j, podemos calcular μ_0 tal que $\frac{\lambda_0}{\mu_0} < 1$. Uma vez obtido este μ_0, podemos considerar os apresentados e discutidos na Seção 3.1 para atribuir o valor do arco como sendo o tempo médio do cliente no sistema.

Como $0 < \lambda < \mu$ temos que $\frac{1}{\mu - \lambda} > 0$, ou seja, os custos dos arcos serão não negativos. Assim sendo, dado um grafo conexo e as taxas médias de chegadas em cada nó, uma maneira de determinar o caminho mínimo dada uma origem e um destino, é calcular para cada λ_j um μ_j tal que $\frac{\lambda_j}{\mu_j} < 1$ e assim atribuir aos arcos os custos como sendo os calculados pela equação (3.57). Note que isso pode resultar em um multigrafo, mas que ainda é conexo.

Considerando que um indivíduo está em um nó (origem) e deseja ir para outro nó (destino), podemos considerar o problema de qual caminho ele deve utilizar para que o custo total do caminho seja o mínimo possível. Neste caso, o custo seria o tempo médio total que seria a soma de cada tempo médio calculado sobre cada arco percorrido.

Após essas considerações temos uma rede que é conexa e os arcos possuem

pesos não negativos, caindo nas hipóteses do Algoritmo de Dijkstra. Após aplicarmos o algoritmo, obtemos uma árvore ótima que nos diz qual caminho o indivíduo deve percorrer para chegar ao seu destino de modo que o tempo médio total seja o mínimo possível. Além disso, dado qualquer nó k da rede diferente da origem e do destino, sabemos que o caminho de arestas que estão na árvore obtida pelo algoritmo cujas extremidades são dadas pela origem e o nó k tem o menor custo possível, pois no final das iterações obtemos o caminho de menor custo da origem para todos os outros nós da rede.

Pensando neste modelo, foi implementado o seguinte algoritmo:

Algoritmo

Passo 1

Fornecer os seguintes dados:

$n \leftarrow$ *número de nós da rede;*

$S \leftarrow$ *origem do caminho;*

$T \leftarrow$ *destino do caminho* ($T \neq S$)*;*

$\rho \leftarrow$ *índice de congestionamento;*

$A_{n\times n} \leftarrow$ *matriz de adjacências* ($a_{ij} = 1$ *se* $(i,j) \in A$, $adj_{ij} = 0$ *caso contrário*)*;*

$R_{n\times n} \leftarrow$ *matriz com as taxas de chegadas* (R_{ij} = *razão média de chegadas por minuto ao nó* j *do nó* i)*;*

$D_{n\times n} \leftarrow$ *matriz com os tempos de atendimentos* (d_{ij} = *tempo de atendimento no nó* j *dos clientes que vem do nó* i) *;*

Passo 2

Calcular a matriz de custos utilizando os resultados da Teoria das Filas;

$F_{n\times n} \leftarrow$ matriz com os pesos das arestas;

Passo 3

Dijkstra(Matriz de Custos, origem, destino);

Fim do Algoritmo

Este algoritmo foi implementado em um computador com 2 GB de memória, processador Intel Atom CPU N450 @ 1.66 $GHz \times 2$ em um sistema de 32 bits, linguagem C e sistema operacional *ubuntu 14.04*.

Suponha que uma cidade tenha 700 semáforos e que sejam dadas as matrizes de adjacências $A_{700\times700}$ e de chegadas $R_{700\times700}$. O gestor do sistema de tráfego dessa cidade deseja ajustar os ciclos semafóricos de modo que a capacidade de atendimento satisfaça o índice de congestionamento na qual será determinado pelo mesmo. Por exemplo, em um semáforo onde chegam 30 veículos por minuto com um índice de congestionamento de 0.75, a capacidade de atendimento tem que ser de 40 veículos por minuto.

O algoritmo apresentado irá gerar uma matriz $D_{700\times700}$ que no elemento d_{ij} terá a capacidade de atendimento do nó j dado que o veículo veio do nó i. Em seguida, o algoritmo calcula a matriz de custos $F_{700\times700}$.

O algoritmo apresentado leva aproximadamente 4 minutos para resolver o problema do caminho mínimo de uma origem para todos os outros nós da rede, em particular para um destino especificado. Com base neste tempo e nas configurações do computador na qual o mesmo foi implementado, para se obter o caminho mínimo entre um nó origem e um nó destino e armazenar esses resultados em um banco de dados seriam necessárias aproximadamente $46,6$ horas, ou seja, aproximadamente 2 dias. Uma maneira de armazenar os caminhos poderia ser utilizando o código de Prüfer, apresentado no Capítulo 1.

Com o armazenamento dos resultados em um banco de dados, conjectura-se a possibilidade de um aplicativo para smartphones com a seguinte função: o usuário fornece a origem e o destino e o aplicativo lhe retorna o(s) caminho(s) com o menor tempo médio de percurso possível. Este aplicativo não precisaria executar o algoritmo; acessaria um banco de dados na web que por sua vez contêm os resultados do algoritmo armazenados.

Na Tabela 4.1 são apresentados os tempos de processamento do algoritmo em função do número de nós. A quantidade de arcos foi calculada com base no resultado do Teorema 1.1.2, mas como a matriz de custos não é necessariamente simétrica e para cada par de nós temos dois arcos que com extremidades nestes nós, temos então que a quantidade de arcos é $n(n-1)$. A Figura 4.1 ilustra um gráfico com a variação de tempo (tempo de processamento) do algoritmo em função da quantidade de nós.

Tabela 4.1: Simulações do algoritmo

Experimento	Quantidade de nós	Quantidade de arcos	Tempo de processamento (s)
1	10	90	0
2	20	380	0
3	30	870	0
4	40	1560	1
5	50	2450	1.5
6	60	3540	1.75
7	70	4830	2.5
8	80	6320	3
9	90	8010	4.25
10	100	9900	5
11	150	22350	10.5
12	200	39800	19.75
13	250	62250	30.5
14	300	89700	43.75
15	350	122150	59.5
16	400	159600	77.75
17	450	202050	99
18	500	249500	121.25
19	600	359400	175.5
20	700	489300	238.25

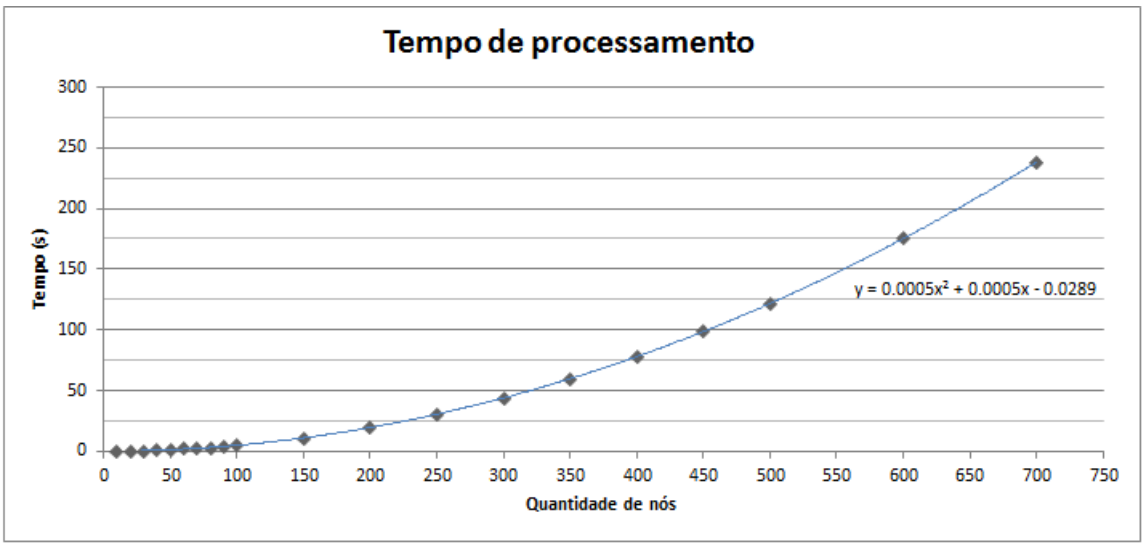

Figura 4.1: Tempo de processamento do algoritmo em função dos nós

Na Figura 4.1 percebemos que o tempo de processamento cresce de forma polinomial e que um polinômio de grau 2 ajusta bem a representação dos pontos.

Conclusão

Através dos estudos desenvolvidos, as metas previstas para o trabalho foram todas alcançadas. Inicialmente foi feito o desenvolvimento teórico para alguns resultados relacionados à Teoria dos Grafos. Este estudo foi de grande importância, visto que vários problemas reais podem ser modelados pela teoria dos grafos, podendo ser resolvidos utilizando algoritmos matemáticos, como o Algoritmo de Dijkstra para o problema do caminho mínimo por exemplo. Em seguida foi feito um embasamento teórico em equilíbrio de tráfego, culminando com um importante resultado relacionado à equivalência entre o problema da complementaridade não linear e o problema equilíbrio.

Este resultado faz uma caracterização para o problema do equilíbrio de tráfego e também torna mais acessível solucionar o problema, uma vez que já existem métodos de resolução clássicos para o problema da complementaridade não linear.

E finalmente, com o intuito de melhor caracterizar o problema abordado foi feito um estudo em uma ferramenta da Pesquisa Operacional, a Teoria das Filas. Através dos estudos de modelos probabilísticos, vários resultados importantes foram obtidos, como por exemplo a obtenção de expressões para o cálculo do tempo médio de um cliente em um sistema ou em uma fila. Através destes resultados, podemos assim dar uma interpretação diferente para valores de arestas de uma rede.

Depois de todos esses estudos, o Algoritmo de Dijkstra foi implementado visando resolver o problema do caminho mínimo com os pesos das arestas sendo calculados através dos resultados obtidos com os estudos em Teoria das Filas. Dada uma rede completa (dado um par de nós, existe uma aresta ligando-os) com arestas com valores positivos, um nó como sendo origem e outro nó como sendo destino, o índice de congestionamento estimado para o sistema e as taxas de chegadas em cada nó, o algoritmo implementado encontra o caminho com o menor custo entre a origem e o destino. Várias simulações foram realizadas e o algoritmo resolve problemas com até 700 nós.

Referências Bibliográficas

[1] LOVÁSZ, Lászlo et al. *Matemática Discreta*. 2 ed. Rio de Janeiro: Sociedade Brasileira de Matemática, 2013.

[2] NOVAES, Antônio. *Pesquisa Operacional e Transportes: modelos probabilísticos*. 1 ed. São Paulo: Editora da Universidade de São Paulo, 1975.

[3] OLIVEIRA, Regina. *Equilíbrio de Tráfego sob duas abordagens: Complementaridade e Otimização*. Campinas: Dissertação de Mestrado IMECC - UNICAMP, 1989.