Capítulo 7
Teoria dos Grafos

Edmilson Marmo Moreira
Universidade Federal de Itajubá - UNIFEI
Programa de Pós-graduação em Ciência e Tecnologia da Computação - POSCOMP

"Os homens são como os vinhos: a idade azeda os maus e apura os bons".

Cícero

1 Considerações Iniciais

No século XVIII, os cidadãos da cidade de Königsberg (uma cidade antiga da Prússia, mais tarde chamada de Kaliningrado, na Rússia) tinham um pequeno problema. O rio que passava pela cidade formava uma ilha. Diversas pontes atravessavam o rio, como mostra a figura 1. O problema era determinar se uma pessoa poderia passear pela cidade passando apenas uma vez por cada ponte.

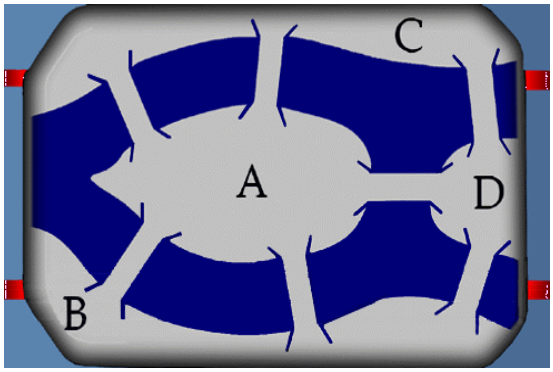


Figura 1: Ilustração das pontes de Königsberg

O matemático Leonhard Euler (1707-1783) ficou curioso com esta situação e resolveu o problema geral apresentando a solução em um artigo publicado em 1736. A idéia de Euler foi simples. Primeiro, ele representou o mapa de Königsberg através de um diagrama como o da figura 2. Os pontos b e c representam as duas *bandas* do rio, os pontos a e d representam as duas ilhas e os 7 arcos conectando estes pontos representam as 7 pontes.

Euler pôde então caracterizar o **problema das pontes de Königsberg**, como é conhecido hoje, da seguinte forma: é possível, partindo de um dos quatro pontos a, b, c ou d, percorrer todos os arcos do diagrama uma única vez e retornar ao ponto de partida?

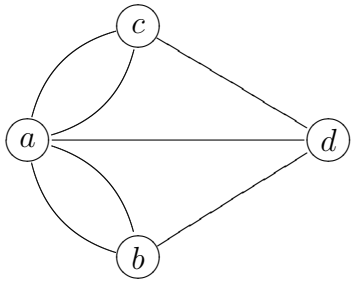


Figura 2: O problema das pontes de Königsberg representado por um Grafo

Euler mostrou que a resposta para esta questão é: "**não**" pela razão que se segue. Suponha que é possível realizar a tarefa descrita pelo problema. Então, qualquer pessoa que percorrer o caminho deve entrar e sair de cada um dos pontos a, b, c e d um número par de vezes.

Se a viagem começa no ponto a, por exemplo, então cada vez que a pessoa entrar em um dos pontos b, c e d, ela deve também deixá-lo. Desta forma, o número de vezes que ela entra e sai de cada ponto b, c e d deve ser par. O mesmo é verdade para o ponto de partida a, mas, neste caso, deve-se também incluir o ponto de partida original de a e o ponto final em a. Por razões idênticas, a situação se repetirá utilizando os pontos b, c e d como ponto de partida.

Portanto, uma vez que a pessoa deve entrar e sair de cada um destes quatro pontos percorrendo arcos diferentes e deve utilizar todas as conexões entre os pontos para finalizar a tarefa, deve existir um número par de arcos conectados em cada um dos pontos a, b, c e d. Como pode ser observado na figura 2, esta situação não ocorre.

O diagrama da figura 2 é chamado de **Grafo** e o artigo de Euler foi o marco inicial de um novo ramo da matemática conhecido como **Teoria dos Grafos**. A teoria dos grafos é uma das áreas de maior aplicação da matemática, sendo usada na computação, ciência, engenharia, química, biologia, economia, pesquisa operacional, lingüística, etc.

2 Conceitos Básicos

Intuitivamente, um grafo é uma coleção de pontos e um conjunto de arcos que conectam pares destes pontos. De uma forma geral, permite-se mais de um arco conectando os mesmos pares de pontos, conforme ilustração da figura 3.

A forma como o grafo é representado não é importante. O que é significativo são os pontos do grafo e o número de arcos entre cada par de pontos (incluindo *loops*). Por exemplo, os diagramas das figuras 4 e 5 representam exatamente o mesmo grafo.

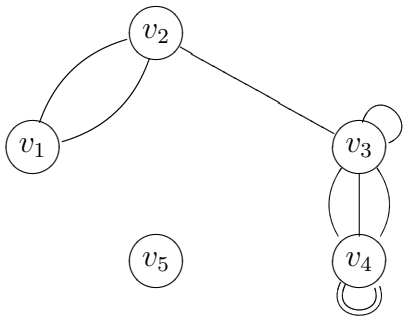


Figura 3: Exemplo de grafo com vários arcos conectando os mesmo pontos

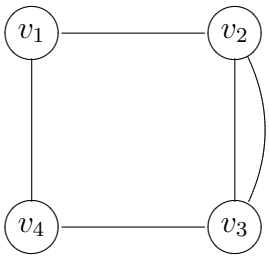


Figura 4: Exemplo de grafo

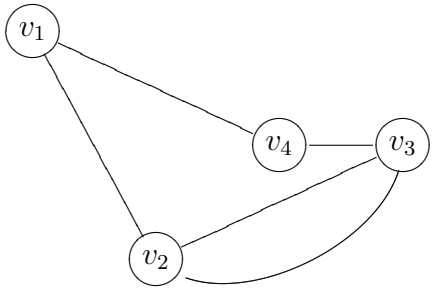


Figura 5: O mesmo grafo da Fig. 4

Por causa disto, as características importantes do grafo da figura 3 podem ser descritas através do conjunto

$$V = \{v_1, v_2, \ldots, v_5\}$$

de seus pontos, e do conjunto

$$E = \{\{v_1, v_2\}, \{v_1, v_2\}, \{v_2, v_3\}, \{v_3\}, \{v_3, v_4\}, \{v_3, v_4\}, \{v_3, v_4\}, \{v_4\}, \{v_4\}\}$$

de seus arcos.

Os elementos do conjunto V são chamados de **vértices** ou **nós** do grafo e os elementos do conjunto E são chamados de **arestas** do grafo. Um arco da forma $\{v\}$ é chamado de *loop* de $\{v\}$.

Definição 2.1 (Grafo) *Um grafo G = (V, A, g) é uma tripla ordenada onde:*

V = *um conjunto não-vazio de* ***vértices (nós);***

A = *um conjunto de* ***arestas (arcos);***

g = *uma função que associa cada aresta a um par não ordenado x-y de vértices chamados de* ***extremos*** *de a.*

Sendo e uma aresta e v, w dois vértices, escreve-se $e = \{v, w\}$ ou $e = \{w, v\}$ dizendo-se, então, que e é uma aresta entre v e w ou que a aresta e liga os vértices v e w que, por este fato, se dizem **adjacentes**.

2.1 Terminologia

Duas arestas que tenham os mesmos extremos são chamadas de **arestas paralelas** ou **arestas múltiplas**. As arestas a_1 e a_2 da Fig. 6 são paralelas.

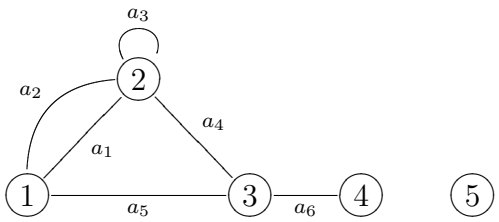


Figura 6: Exemplo de grafo com arestas rotuladas

Se entre dois vértices existir mais que uma aresta então, se for necessário distinções, o grafo correspondente toma o nome de **multigrafo**. Grafos que não possuem *loops* nem arestas paralelas são denominados **grafos simples**. Um **vértice isolado** não é adjacente a qualquer outro vértice; na Fig. 6 o nó 5 é um vértice isolado. O **grau** de um vértice é o número de arestas que o tem como ponto extremo. Na Fig. 6, os vértices 1 e 3 possuem grau 3, o vértice 2 tem grau 5, o vértice 4 tem grau 1 e o vértice 5 possui grau 0. Um grafo G é **regular de grau** k ou *k-regular* se todo vértice tem grau k. Em outras palavras, um grafo é regular se todo vértice tem o mesmo grau. O grafo conexo 0-regular é o grafo trivial com um vértice e nenhuma aresta. O grafo conexo 1-regular é o grafo com dois vértices e uma aresta que os conecta, e assim por diante. A ordem de um grafo G é dada pela cardinalidade do conjunto de vértices, ou seja, pelo número de vértices de G.

Como a função g, que relaciona cada aresta a seus extremos, é entendida como uma função propriamente dita, cada aresta tem um único par de pontos extremos. Se g for uma função injetiva, então haverá apenas uma aresta associada a cada par de vértices e o grafo não terá arestas paralelas.

Um **grafo completo** é aquele no qual todos os vértices distintos são adjacentes. Neste caso, g é quase uma função sobrejetora – todo par $x - y$ de vértices distintos está no conjunto imagem de g –, mas não há um laço em cada vértice, de forma que pares do tipo $x - x$ não devem ter imagem. O grafo completo com n vértices é denotado por K_n.

Teorema 2.2 (Número de arestas de um grafo K_n) *O número de arestas em um grafo completo é* $n(n-1)/2$.

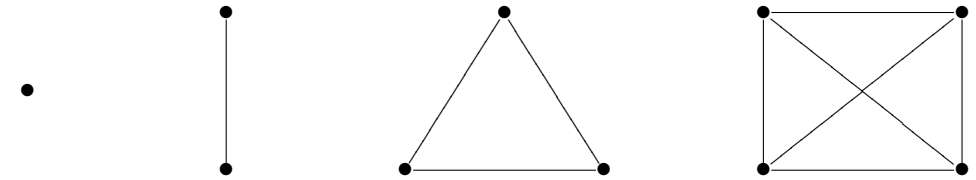
Demonstração: A prova é realizada por indução matemática. Supondo G_n um grafo que contém n vértices. O caso base é G_1. Neste caso, como existe somente um vértice, não existem arestas, pois os grafos completos são grafos simples e, portanto, não possuem laços. Desta forma, verifica-se que $n(n-1)/2 = 0$.

Supondo que a hipótese é verdadeira para G_n, onde $n \geq 1$. Seja, agora, o grafo G_{n+1}. É necessário mostrar que o número de vértices neste grafo é $n(n+1)/2$. Fazendo v_{n+1} o vértice adicional que se encontra em G_{n+1} e não em G_n. O número máximo de arestas no grafo G_{n+1} é igual ao número máximo no grafo G_n mais todas as ligações possíveis entre v_{n+1} e cada vértice de G_n. Como esse número de ligações é igual ao número de vértices em G_n, tem-se:

$$\text{Número máximo} = \frac{n(n-1)}{2} + n = \frac{n(n-1)+2n}{2} = \frac{n^2+n}{2} = \frac{n(n+1)}{2}$$

Isso prova o teorema.

A figura 7 apresenta alguns exemplos de grafos completos.



K_1 K_2 K_3 K_4

Figura 7: Exemplos de Grafos Completos

Um **subgrafo** de um grafo consiste em um conjunto de vértices e um conjunto de arestas que são subconjuntos dos conjuntos de vértices e arestas originais, respectivamente, nos quais os extremos de qualquer aresta precisam ser os mesmos que no grafo original. Em outras palavras, é um grafo obtido apagando-se parte do grafo original e deixando o restante sem alterações. Um tipo especial de subgrafo é denominado **clique**. Um clique é um subgrafo que é completo. As figuras 8 e 9 mostram subgrafos da figura 6. O grafo na figura 8 é simples e também completo.

Um **caminho** de um vértice n_0 a um vértice n_k é uma seqüência:

$$n_0, a_0, n_1, a_1, \ldots, n_{k-1}, ak - a, n_k$$

de vértices e arestas onde, para cada i, os extremos da aresta a_i são $n_i - n_{i+1}$. No grafo da figura 6, um caminho do vértice 2 ao vértice 4 consiste na seqüência $2, a_1, 1, a_2, 2, a_4, 3, a_6, 4$. O **comprimento** de um caminho é o número de arestas que ele contém. Se uma aresta

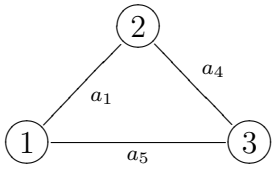


Figura 8: Um subgrafo do grafo da figura 6

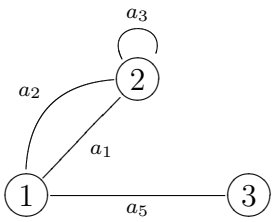


Figura 9: Outro subgrafo do grafo da figura 6

for usada mais de uma vez, ela deve ser contada tantas vezes quantas for usada. O **comprimento** do caminho que acabou de ser descrito entre os nós 2 e 4 é 4. Um grafo é dito conexo se houver um caminho entre quaisquer dois vértices. Os grafos das figuras 8 e 9 são conexos, mas o grafo da figura 6 não é. Um **ciclo** em um grafo é um caminho de comprimento igual ou maior que 3 de algum vértice n_0 até n_0 de novo de forma que nenhum vértice ocorra mais de uma vez no caminho, sendo n_0 o único vértice que ocorre mais de uma vez e este ocorre apenas nos extremos do caminho. No grafo da figura 8, $1, a_1, 2, a_4, 3, a_5, 1$ é um ciclo. Um grafo sem ciclos é dito **acíclico** e um ciclo de comprimento k é chamado de *k-ciclo*.

Um **caminho simples** é um caminho em que todos os vértices são distintos. Um caminho em que todas as arestas são distintas é chamado **trilha**.

Um grafo G é dito um **grafo rotulado** se estão associados dados de algum tipo às suas arestas e/ou vértices. Em particular, G é **grafo ponderado** se a cada aresta e de G está associado um número não negativo $w(e)$ dito **peso** ou **comprimento** de e. A figura 10 mostra um grafo ponderado onde o comprimento de cada aresta está descrito de maneira óbvia. O peso ou comprimento de um caminho em grafo ponderado G é definido como sendo a soma dos pesos das arestas no caminho. Um problema importante na teoria dos grafos é encontrar o **menor caminho**, isto é, um caminho de peso (comprimento) mínimo entre quaisquer dois vértices. O comprimento do caminho mínimo entre P e Q na figura 10 é 14; um tal caminho é

$$(P, A_1, A_2, A_5, A_3, A_6, Q).$$

Um grafo G é dito **biparticionado** ou **bipartido** se o seu conjunto de vértices V pode ser particionado em dois subconjuntos M e N tais que cada aresta de G conecta um vértice de M a um vértice de N. Um grafo é completo e biparticionado se cada vértice de M é conectado a cada vértice de N. Esse tipo de grafo é denotado por $K_{m,n}$, onde m é o número de vértices em M, e n é o número de vértices em N. A figura 11 mostra o grafo $K_{2,3}$.

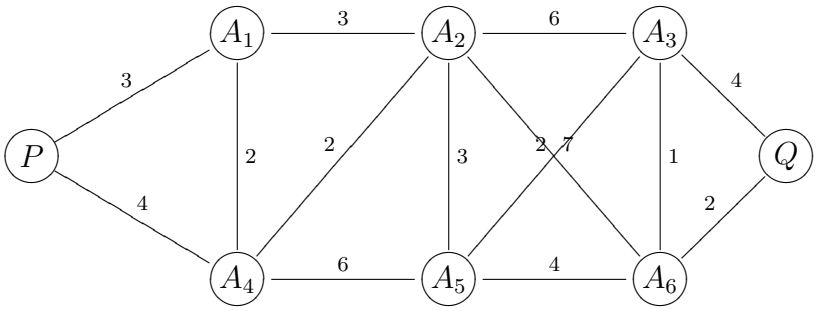


Figura 10: Exemplo de grafo ponderado

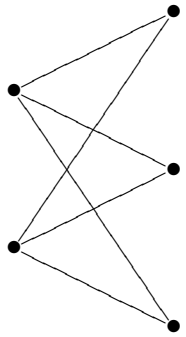


Figura 11: Grafo $K_{2,3}$

Existe um teorema interessante em relação aos grafos bipartidos.

Teorema 2.3 (Grafo bipartido) *Um grafo G é um grafo bipartido se, e somente se, todo ciclo de G possuir caminho par.*

Demonstração: *Ida:* Seja X e Y as duas partições de G. Todo caminho em G alterna um vértice X com um vértice Y. Isso é a consequência da definição de grafo bipartido. Supondo que um ciclo contém um vértice v_i em uma das duas partições. Para voltar a esse vértice, é preciso ir na outra partição e voltar um número par de vezes.

Volta: Seja G um grafo onde todo ciclo é de comprimento par. Seja um vértice v_i de G. Coloca-se num conjunto X o vértice v_i e todos os outros vértices que estão a uma distância par de v_i. Os outros vértices formam o conjunto Y. Se não tivesse nenhuma aresta ligando dois vértices de X ou dois vértices de Y, seria respeitado as condições para que o grafo fosse bipartido. Supondo agora que existe uma outra aresta entre dois vértices a e b de X (ou Y). Já existe um caminho par entre a e b. Acrescentando a nova aresta, obtêm-se um ciclo de comprimento ímpar, o que contradiz a hipótese. Portanto, não pode existir outra aresta entre quaisquer par de vértices que já está em X (igualmente para Y) e o grafo é bipartido.

Isso prova o teorema.

A definição de grafo como uma tripla ordenada, constituído por um conjunto de vértices, um conjunto de arestas e uma função de mapeamento, permite que dois grafos que se parecem muito diferentes em suas representações gráficas, sejam, ainda assim, o mesmo grafo. Isto foi observado nos grafos representados pelas figuras 4 e 5, pois eles possuem os mesmos vértices, as mesmas arestas e a mesma função de associação entre arestas e vértices. Desta forma, dois grafos $G(V_1, A_1, g_1)$ e $G(V_2, A_2, g_2)$ são **isomorfos** se existirem bijeções $f_1 : V_1 \rightarrow V_2$ e $f_2 : A_1 \rightarrow A_2$ tais que para cada aresta $a \in A_1, g_1(a) = x - y$ se, e somente se, $g_2[f_2(a)] = f_1(x) - f_1(y)$. Em outras palavras, tem-se $\mid V_1 \mid = \mid V_2 \mid$ e existe uma função unívoca: $f : V_1 \rightarrow V_2$, tal que (i, j) é elemento de A_1 se, e somente se, $(f(i), f(j))$ é elemento de A_2. A figura 12 ilustra o exemplo de dois grafos isomorfos entre si. Pode-se observar que:

$$f(a) = 1, f(b) = 2, f(c) = 3, f(d) = 8, f(e) = 5, f(f) = 6, f(g) = 7, f(h) = 4.$$

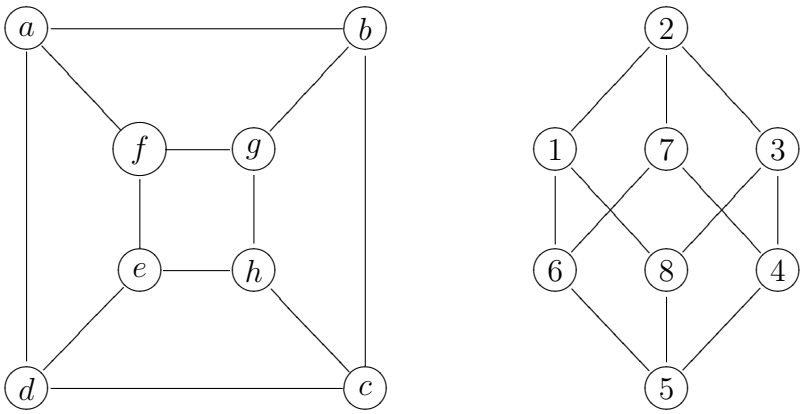


Figura 12: Grafos isomorfos

2.2 Grafos Planares

Um grafo ou multigrafo que pode ser desenhado no plano de tal modo que suas arestas não se cortem é dito **planar**. O grafo da figura 4 é um exemplo de grafo planar. Uma representação particular de um multigrafo planar finito é chamado **mapa**.

Um fato sobre grafos planares foi descoberto pelo matemático Leonhard Euler. Um grafo simples, conexo e planar divide o plano em um número de regiões, incluindo as regiões totalmente fechadas e uma região infinita exterior. Euler observou uma relação entre o número v de vértices, o número a de arestas e o número r de regiões neste tipo de grafos. Esta relação é denominada **fórmula de Euler**.

Teorema 2.4 (Fórmula de Euler) $v - a + r = 2$

Demonstração: Suponha que o mapa M consiste de um único vértice v. Então, $v = 1$, $a = 0$ e $r = 1$. Logo, $v - a + r = 2$. Caso contrário, M pode ser montado a partir de um vértice isolado usando as seguintes duas construções:

1. Acrescente um novo vértice q_2 e conecte-o a um vértice existente q_1 por uma aresta que não corte nenhuma aresta existente, como na figura 13.
2. Conecte dois vértices existentes q_1 e q_2 por uma aresta a que não cruze nenhuma aresta existente, como na figura 14.

Nenhuma das operações muda o valor de $v - a + r$. Logo, M tem o mesmo valor para $v - a + r$ do que no mapa com um único vértice, isto é, $v - a + r = 2$. Isso prova o teorema.

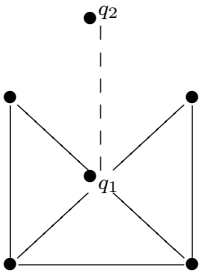


Figura 13: Grafo auxiliar 01

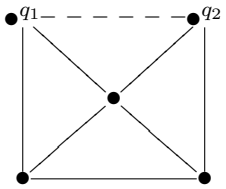


Figura 14: Grafo auxiliar 02

Existem duas conseqüências da fórmula de Euler, se for incluído mais restrições no grafo. Supondo que seja preciso que o grafo não seja apenas simples, conexo e planar, mas tenha, pelo menos, três vértices. Em uma representação planar deste grafo, pode-se contar o número de arestas que são adjacentes a cada região, incluindo a região exterior. Arestas que estão totalmente dentro de uma região contribuem com duas arestas para esta região. Arestas que separam duas regiões contribuem com uma aresta para cada região. Portanto, se houver a arestas no grafo, o número de arestas de regiões é $2a$.

Não existem regiões com apenas uma aresta adjacente, porque não há laços no grafo. Não há regiões com exatamente duas arestas adjacentes porque não existem arestas paralelas e o grafo consistindo apenas em uma aresta ligando dois vértices (que tem duas arestas adjacentes à região exterior) foi excluído. Portanto, em cada região existem pelo menos três arestas adjacentes, de forma que $3r$ é o número mínimo de arestas em cada região. Por isso,

$$2a \geq 3r$$

ou pelo teorema 2.4:

$$2a \geq 3(2 - v + a) = 6 - 3v + 3a$$

ou seja,

$$a \leq 3v - 6.$$

Se for imposta uma última restrição de que não haja ciclos de comprimento 3, cada região terá pelo menos quatro arestas adjacentes, portanto $4r$ será o número mínimo de arestas de região. Isto leva à desigualdade

$$2a \geq 4r$$

que pode ser escrita como

$$a \leq 2v - 4.$$

2.3 Coloração de Grafos

Considerando um grafo G, uma **coloração de vértices**, ou simplesmente, uma **coloração** de G é uma atribuição de cores aos vértices de G de tal forma que vértices adjacentes tenham cores distintas. Diz-se que G é n-colorável se existe uma coloração de G que usa n cores. O número mínimo de cores necessárias para pintar G é dito **número cromático** de G e é denotado por $\chi(G)$.

A seguir será apresentado o algoritmo de Welch e Powell para a coloração de um grafo G. Este algoritmo nem sempre fornece a coloração minimal de G.

Algoritmo de Welch-Powell – A entrada é um grafo G.

Passo 1: Ordene os vértices de G em ordem decrescente de grau.

Passo 2: Atribua a primeira cor, C_1, ao primeiro vértice e, então, seqüencialmente, atribua C_1 a cada vértice que não é adjacente a algum vértice que o antecedeu e ao qual foi atribuída a cor C_1.

Passo 3: Repita o **Passo 2** com a segunda cor C_2 e os vértices subseqüentes não coloridos.

Passo 4: Repita o **Passo 3** com a terceira cor C_3, depois com a quarta cor C_4, e assim sucessivamente, até que todos os vértices estejam coloridos.

Passo 5: Saia.

Teorema 2.5 (Teorema das Quatro Cores) *Se as regiões de qualquer mapa M são coloridas de forma que regiões adjacentes tenham cores distintas, então não mais do que quatro cores são necessárias.*

A demonstração do teorema 2.5 utiliza computadores, essencialmente. Especificamente, Appel e Haken mostraram que, se o teorema das quatro cores fosse falso, deveria existir um contra-exemplo em um conjunto de aproximadamente 2000 grafos planares. Mostraram depois, usando o computador, que nenhum destes tipos de grafos planares era um contra-exemplo.

2.4 Árvores

Um grafo A é dito uma **árvore** se A é conexo e não tem ciclos. A figura 15 apresenta um exemplo de árvore. Uma **floresta** é um grafo sem ciclos; logo, as componentes conexas de uma floresta são árvores. Um grafo sem ciclo é dito **acíclico**. A árvore que consiste em um único vértice e nenhuma aresta é dita **árvore degenerada**.

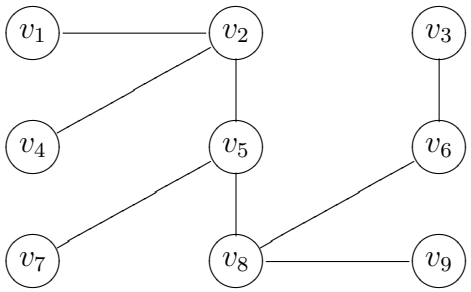


Figura 15: Exemplo de árvore

Considerando uma árvore A, claramente, existe apenas um caminho simples entre dois vértices de A; caso contrário, os dois caminhos formariam um ciclo. Além disso:

1. Supondo que não existe uma aresta $\{u, v\}$ em A e se for adicionada a aresta $a = \{u, v\}$ a A. Então, o caminho simples de u até v em A mais o vértice a formará um ciclo; neste caso, A deixará de ser uma árvore.

2. Por outro lado, supondo que existe uma aresta $a = \{u, v\}$ em A e se esta aresta for deletada de A, então, A não é mais um grafo conexo (já que não existe um caminho entre u e v); neste caso, A deixa de ser uma árvore.

Uma **folha** de um grafo é um vértice de grau 1. As folhas são chamadas também de **vértices terminais** ou **vértices pendentes**.

Teorema 2.6 *Toda árvore com ao menos dois vértices tem uma folha.*

Demonstração: Seja T uma árvore com ao menos dois vértices e seja P o caminho mais longo em T. Como T é conexo e contém ao menos dois vértices, P tem dois ou mais

vértices. Supondo, $P = v_0 \sim v_1 \sim \ldots \sim v_l$, onde $l \geq 1$. Afirma-se que o primeiro e o último vértices de P (isto é, v_0 e v_l) são folhas de T.

Supondo, por contradição, que v_0 não seja folha. Como v_0 tem ao menos um vizinho (v_1), tem-se que $d(v_0) \geq 2$. Seja x outro vizinho de v_0 (isto é, $x \neq v_1$). Nota-se que x não é um vértice em P, pois, em caso contrário, existiria um ciclo:

$$v_0 \sim v_1 \sim \ldots \sim x \sim v_0.$$

Pode-se, assim, antepor x ao caminho P para formar o caminho Q:

$$Q = x \sim v_0 \sim v_1 \sim \ldots \sim v_l.$$

Todavia, Q é um caminho em T que é mais longo do que P. Isso é uma contradição! Portanto, v_0 é uma folha. Da mesma forma, v_l é uma folha. Portanto, T tem ao menos duas folhas.

Teorema 2.7 *Seja G um grafo com $v > 1$ vértices. Então, as seguintes afirmações são equivalentes:*

1. *G é uma árvore.*
2. *G é um grafo acíclico e tem $v - 1$ arestas.*
3. *G é conexo e tem $v - 1$ arestas.*

Demonstração: A demonstração é por indução sobre v. O teorema certamente é verdade para o grafo que possui apenas um vértice e, portanto, nenhuma aresta. Isto é, o teorema vale para $v = 1$.

Assumindo que $v > 1$ e que o teorema vale para grafos com menos do que v vértices.

(1) implica (2): Supondo que G é uma árvore. Então, G é acíclico, e precisa-se mostrar apenas que G tem $v - 1$ arestas. Conforme demonstrado no teorema 5.1, G tem pelo menos dois vértices de grau 1. Deletando-se um destes vértices e sua respectiva aresta, obtém-se uma árvore A que tem $v - 1$ vértices. O teorema vale para A, portanto, A tem $v - 2$ arestas. Logo, G tem $v - 1$ arestas.

(2) implica (3): Supondo que G é acíclico e tem $v - 1$ arestas. Precisa-se mostrar apenas que G é conexo. Supondo que G é desconexo e tem k componentes $A_1, A_2, \ldots, A_k$, que são árvores, uma vez que cada uma é conexa e acíclica.

Supondo que A_i tem v_i vértices. Note que $v_i < v$. Portanto, o teorema vale para A_i, e logo A_i tem v_i arestas. Portanto, $v = v_1 + v_2 + v_3 + \ldots + v_k$ e

$$v - 1 = (v_1 - 1) + (v_2 - 1) + \ldots + (v_k - 1) = v_1 + v_2 + \ldots + v_k - k = v - k$$

Assim, $k = 1$. Mas isso contradiz a hipótese de que G é desconexo e tem $k > 1$ componentes. Logo, G é conexo.

(3) implica (1): Supondo que G é conexo e tem $v-1$ arestas. É necessário mostrar apenas que G é acíclico. Supondo que G tem um ciclo contendo um aresta a. Deletando a, obtem-se o grafo $H = G - a$, que também é conexo. Mas H tem v vértices e $n - 2$ arestas, e isto é uma contradição. Logo, G é acíclico e, portanto, é uma árvore.

Um subgrafo A de um grafo conexo é uma **árvore geradora** de G se A é uma árvore e A inclui todos os vértices de G. A figura 16 mostra um grafo G e as árvores geradores A_1, A_2 e A_3 de G.

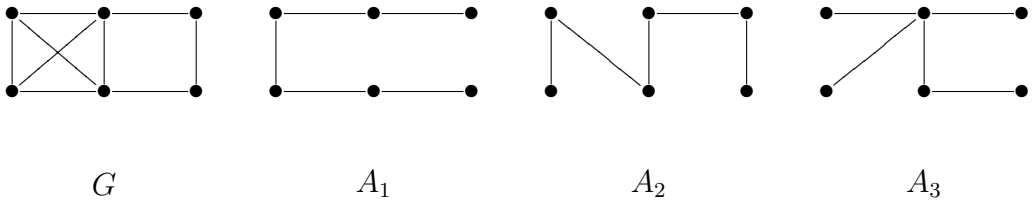


Figura 16: Árvores Geradoras

Supondo que G é um grafo conexo ponderado, ou seja, cada aresta de G está associada a um número não negativo (*peso*). Então, qualquer árvore geradora A de G está associada a um peso total obtido pela soma dos pesos das arestas em A. Uma **árvore minimal geradora** de G é uma árvore geradora cujo peso total é o menor possível. Os algoritmos a seguir permitem encontrar a árvore minimal geradora A de um grafo conexo ponderado G, onde G tem v vértices.

Algoritmo Árvore Geradora Mínima – A entrada é um grafo conexo ponderado G com v vértices.

Passo 1: Ordene as arestas de G em ordem decrescente de peso.

Passo 2: Seqüencialmente, delete cada aresta que não desconecta o grafo até que restem $v - 1$ arestas.

Passo 3: Saia.

Algoritmo Árvore Geradora Mínima (Kruskal) – A entrada é um grafo conexo ponderado G com v vértices.

Passo 1: Ordene as arestas de G em ordem crescente de peso.

Passo 2: Começando apenas com vértices de G e procedendo seqüencialmente, adicione cada aresta que não gere um ciclo até que $v - 1$ arestas sejam adicionadas.

Passo 3: Saia.

O peso de uma árvore minimal geradora é único, mas a árvore, propriamente dita, não é. Árvores geradoras minimais distintas podem ocorrer quando duas ou mais arestas têm o mesmo peso. Neste caso, a ordenação das arestas no Passo 1 dos dois algoritmos não é única e pode, portanto, resultar em diferentes árvores geradoras minimais.

2.5 Grafo Orientado

Um grafo também pode possuir arestas dirigidas, como no exemplo da figura 17. Neste caso, o grafo é denominado **grafo orientado** ou **dígrafo**.

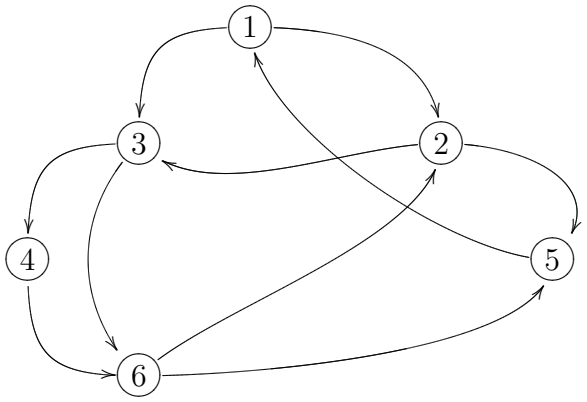


Figura 17: Exemplo de um grafo orientado – dígrafo

Definição 2.8 (Dígrafo) *Um dígrafo G = (V, A, g) é uma tripla ordenada onde:*

V = *um conjunto não-vazio de* ***vértices*** *(**nós**);*

A = *um conjunto de* ***arestas*** *(**arcos**);*

g = *uma função que associa cada aresta a um par ordenado (x,y) de vértices.*

Num dígrafo escreve-se $a \equiv (v, w)$ para significar que a é um arco que liga v a w orientado de v para w. Neste caso, diz-se que v é **adjacente** ao vértice w e que o arco a é **incidente** sobre w e **emergente** de v.

Em grafos dirigidos há pequenas diferenças nas definições de grau, caminhos e conectividade. Suponha que G é um grafo orientado. O **grau de saída** de um vértice v de G é o número de arestas começando em v, e o **grau de entrada** é o númeo de arestas terminando em v. Como cada aresta começa e termina em um vértice, a soma dos graus de saída dos vértices de um grafo orientado G é igual à soma dos graus de entrada dos vértices, que é igual ao número de arestas em G.

Um vértice v com grau de entrada zero é dito uma **fonte**, e um vértice v com grau de saída zero é dito um **sumidouro**.

Os conceitos de caminho, caminho simples, trilha e ciclo são os mesmos dos grafos não orientados, exceto pelo fato de que a direção da aresta deve coincidir com a direção do caminho. Especificamente, seja G um grafo orientado, então:

1. Um **caminho** orientado P em G é uma seqüência alternada de vértices e arestas orientadas, por exemplo,

   $$P = (v_0, a_1, v_1, a_2, v_2, \ldots, a_n, v_n)$$

   tal que cada aresta a_i começa em v_{i-1} e termina em v_i. Quando não existem ambigüidades, denota-se P por sua seqüência de vértices ou por sua seqüência de arestas.

2. O **comprimento** do caminho P é n, seu número de arestas.

3. Um **caminho simples** é um caminho com vértices distintos. Uma **trilha** é um caminho com arestas distintas.

4. Um **caminho fechado** tem os vértices primeiro e último iguais.

5. Um **caminho gerador** contém todos os vértices de G.

6. Um **ciclo** ou **circuito** é um caminho fechado com vértices distintos (exceto o primeiro e o último).

7. Um **semicaminho** é o mesmo que um caminho, a não ser pelo fato de que a aresta a_i pode iniciar em v_{i-1} ou v_i e terminar no outro vértice. **Semitrilhas** e caminhos **semi-simples** são definidos de maneira análoga.

Um vértice v é **alcançável** a partir de um vértice u se existir um caminho de u para v. Se v é alcançável a partir de u, então (eliminando as arestas redundantes) existe um caminho simples de u para v.

Existem três tipos de conectividade em um grafo orientado G:

1. G é **fortemente conexo** ou **forte** se, para qualquer par de vértices u e v em G, existe um caminho de u para v e um caminho de v para u, isto é, se cada um deles é alcançável a partir do outro.
2. G é **unilateralmente conexo** ou **unilateral** se, para qualquer par de vértices u e v em G, existe um caminho de u para v ou um caminho de v para u, isto é, se algum deles é alcançável a partir do outro.
3. G é **fracamente conexo** ou **fraco** se existe um semicaminho entre quaisquer dois vértices u e v em G.

Seja G' um grafo (não orientado), obtido do grafo orientado G considerando todas as arestas de G como não orientadas. Claramente, G é fracamente conexo se, e somente se, o grafo G' é conexo.

É importante notar que conectividade forte implica conectividade unilateral, e que conectividade unilateral implica em conectividade fraca. Diz-se que G é **estritamente unilateral** se é unilateral mas não forte, e é **estritamente fraco** se é fraco mas não unilateral.

A matriz de adjacências M de um grafo direcionado G com v vértices e sem arestas paralelas terá um 1 na posição i, j se houver uma aresta do vértice v_i para o vértice v_j. Este é um caminho de tamanho 1 de v_i para v_j. A matriz de adjacências dá informações relativas a uma forma limitada de alcançabilidade, as que são obtidas através de caminhos de comprimento 1. No entanto, será multiplicado $M \times M$. Denota-se este produto booleano por $M^{(2)}$ a fim de distinguir de M^2, o resultado da multiplicação de $M \times M$ ordinária de matrizes. A multiplicação de matrizes booleanas possuem a seguinte adaptação:

$$M^{(2)}[i,j] = \bigvee_{k=1}^{n} (m_{ik} \wedge m_{kj})$$

Se um termo como $m_{i2} \wedge m_{2j}$ nesta soma valer 0, então ou $m_{i2} = 0$ ou $m_{2j} = 0$ (ou ambos), e não há caminho de comprimento 1 de v_i a v_2 ou não há caminho de tamanho 1 de v_2 a v_j (ou ambos). Portanto, não existe caminho de comprimento 2 de v_i para v_j passando por v_2. Se, por outro lado, $m_{i2} \wedge m_{2j}$ não for 0, então tanto $m_{i2} = 1$ e $m_{2j} = 1$. Neste caso, existe um caminho de comprimento 1 de v_i para v_2 e um caminho de comprimento 1 de v_2 para v_j, de forma que há um caminho de comprimento 2 de v_i para v_j passando por v_2. Portanto, as entradas em $M^{(2)}$ indica a alcançabilidade através de caminhos de comprimento 2.

Teorema 2.9 (Matrizes Booleanas de Ajacências e Alcançabilidade) *Se M é a matriz booleana de adjacências para um grafo direcionado G com n vértices e sem arestas paralelas, então $M^{(m)} = [i, j] = 1$ se, e somente se, houver um caminho de comprimento m do vértice v_i ao vértice v_j.*

Demonstração: Realiza-se uma prova por indução em m. Já foi visto que o resultado é verdadeiro para $m = 1$ (e $m = 2$). Supondo que $M^{(p)} = [i, j] = 1$ se, e somente se, existir um caminho de comprimento p de v_i até v_j. Sabe-se que

$$M^{(p+1)}[i, j] = \bigvee_{k=1}^{n} (M^{(p)}[i, k] \wedge a_{kj})$$

Esta expressão valerá 1 se, e somente se, pelo menos uma parcela valer 1, supondo que $M^{(p)}[i, q] \wedge a_{qj} = 1$, ou $M^{(p)}[i, q] = 1$ e $a_{qj} = 1$. Isto será verdade se, e somente se, existir um caminho de comprimento p de v_i até v_q (pela hipótese de indução) e existir um caminho de comprimento 1 de v_q a v_j, que significa que existe um caminho de comprimento $p + 1$ de v_i a v_j. Isto prova o teorema!

Se o vértice v_j é alcançável a partir do vértice v_i, ele o será através de um caminho de algum tamanho. Este caminho será evidenciado por um 1 na posição i, j de M, $M^{(2)}$, $M^{(3)}$, etc., mas, quantas multiplicações serão necessárias para se identificar todos os caminhos de qualquer tamanho entre dois vértices? Se houver v vértices no grafo, então qualquer caminho com $v+1$ ou mais vértices deve ter pelo menos um vértice repetido. Isto decorre do Princípio da Casa do Pombo – existem v "caixas" (vértices distintos) nas quais se está colocando mais de v objetos (os vértices de um caminho com v ou mais arestas). A seção de um caminho que se encontre entre os vértices repetidos é um ciclo. Se $v_i \neq v_j$, então o ciclo pode ser eliminado para formar um caminho mais curto; assim, se existe um caminho entre dois vértices v_i e v_j, este caminho tem comprimento máximo de $v - 1$. Conseqüentemente, seja $v_i = v_j$ ou $v_i \neq v_j$, não é necessário procurar por um caminho de v_i até v_j com comprimento maior que v. Portanto, para determinar a alcançabilidade, deve-se apenas considerar os elementos i, j em $M, M^{(2)}, \ldots, M^{(v)}$. Desta forma, pode-se definir a **matriz de alcançabilidade R** como

$$R = M \vee M^{(2)} \vee \ldots \vee M^{(v)}$$

Então, v_j é alcançável a partir de v_i se, e somente se, o elemento i, j em R for 1.

3 Representação Computacional de Grafos

Uma das vantagens dos grafos é a sua representação visual das informações. Entretanto, para o armazenamento e manipulação de grafos por computador, as informações necessitam ser representadas de outras maneiras. Uma opção é usar a definição formal de grafos; ou seja, pode-se armazenar um conjunto de vértices, um conjunto de arestas e uma função que associe a cada aresta um conjunto com seus dois vértices extremos.

Exemplo:

1. Para representar o grafo da figura 6 usando a definição formal seria necessário armazenar os conjuntos:

$$\{1, 2, 3, 4, 5\}$$

$$\{a_1, a_2, a_3, a_4, a_5, a_6\}$$

e

$$\{[a_1, 1, 2], [a_2, 1, 2], [a_3, 2, 2], [a_4, 2, 3], [a_5, 1, 3], [a_6, 3, 4]\}$$

onde, neste conjunto, a relação funcional é representada por vetores de três elementos.

Esta representação pode ser mais eficiente se for restrita a grafos simples e conexos, pois, neste caso, não há necessidade de explicitar os nomes das arestas ou fornecer o conjunto de vértices.

Este tipo de representação não é adequado para a realização de diversas operações em grafos. Por exemplo, supondo que se deseja encontrar todos os vértices adjacentes a um determinado vértice. Deve-se comparar a ocorrência do respectivo vértice em cada elemento do conjunto que representa a função g, o que é uma tarefa com alto custo computacional. Por conseguinte, as representações de grafos com conjuntos não são utilizadas. A solução é utilizar estruturas de dados tais como **matrizes de adjacências** ou **listas de adjacências**.

3.1 Matriz de Adjacências

Supondo que um grafo tem n vértices numerados $n_1, n_2, \ldots, n_n$. Esta numeração define uma ordenação arbitrária no conjunto de vértices. No entanto, não há qualquer relevância no fato de um vértice aparecer antes de outro na ordenação. De posse dos vértices ordenados, pode-se formar uma matriz $n \times n$ onde o elemento i, j é o número de

arestas entre o vértice n_i e n_j. Esta matriz é chamada de matriz de adjacências A do grafo com relação à ordenação.

Desta forma, $a_{ij} = p$ onde existem p arestas entre n_i e n_j.

Exemplo:

1. Considerando o grafo da figura 18.

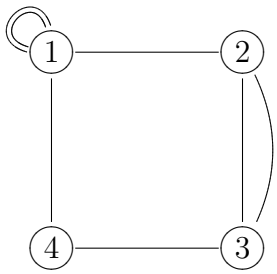


Figura 18: Exemplo de grafo para representação em Matriz de Adjacências

A matriz de adjacências para este grafo com respeito à ordenação 1, 2, 3, 4 é uma matriz 4×4. O elemento $a_{1,1}$ é 2 devido ao fato de haver dois laços no vértice 1. Todos os demais elementos da diagonal são 0. O elemento $a_{2,1}$ é 1 porque existe apenas uma aresta entre os vértices 2 e 1, o que também indica que o elemento $a_{1,2}$ vale 1. E, assim por diante. A matriz resultante é:

$$M = \begin{bmatrix} 2 & 1 & 0 & 1 \\ 1 & 0 & 2 & 0 \\ 0 & 2 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Para todo grafo não-direcionado, a matriz de adjacências será uma matriz simétrica. Neste caso, somente os elementos da diagonal principal e os elementos abaixo dela necessitam ser armazenados.

Em um grafo direcionado, a matriz de adjacências A reflete a direção das arestas. Em uma matriz direcionada $a_{ij} = p$ onde p é o número de arestas do vértice n_i para o vértice n_j. Neste caso, a matriz não é necessariamente simétrica, uma vez que uma aresta do vértice n_i para o vértice n_j não implica uma aresta do vértice n_j para o vértice n_i.

Exemplo:

1. Considerando o grafo da figura 19.

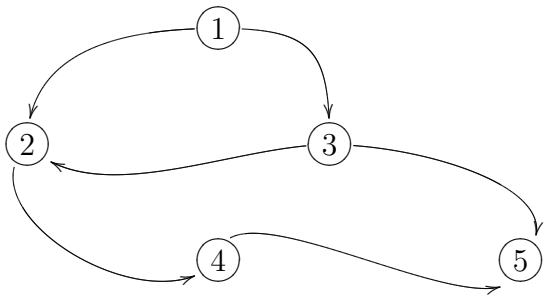


Figura 19: Exemplo de dígrafo para representação em Matriz de Adjacências

A matriz de adjacências para este grafo será:

$$M = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

3.2 Lista de Adjacências

Diversos grafos, ao contrário de serem completos, possuem relativamente poucas arestas. Esses grafos têm matrizes de adjacências ditas esparsas; isto é, suas matrizes de adjacências contém muitos zeros. No caso de o grafo ter n vértices, serão necessários n^2 elementos para representarem sua matriz de adjacências, ainda que a maior parte desses elementos seja zero. Qualquer algoritmo que precise que todas as arestas do grafo sejam verificadas realizará comparações com todos os n^2 elementos da matriz, uma vez que não há outro meio de determinar quais os elementos que não são zero além de examiná-los. Para encontrar todos os vértices adjacentes ao vértice n_i, será preciso varrer toda a i-ésima linha da matriz de adjacências, que tem um total de n elementos.

Um grafo com poucas arestas pode ser representado mais eficientemente, se for armazenado apenas os elementos não-nulos de sua matriz de adjacências. Esta representação consiste em uma lista para cada vértice de todos os vértices adjacentes a ele. Desta forma, tem-se um vetor de n ponteiros, um para cada vértice, para obter o início de cada lista. Esta representação na forma de **lista encadeada**, apesar de precisar de mais memória para os ponteiros, pode ser mais eficiente que a matriz adjacências. Para se encontrar todos os vértices adjacentes a n_i será preciso percorrer a lista referente a n_i, que deve ter menos elementos que os n que seria necessário examinar na matriz de adjacências. Entretanto, para se determinar se um vértice n_j em particular é adjacente ao vértice n_i, será necessário percorrer toda a lista encadeada de n_i.

A figura 20 apresenta um exemplo de lista de adjacências.

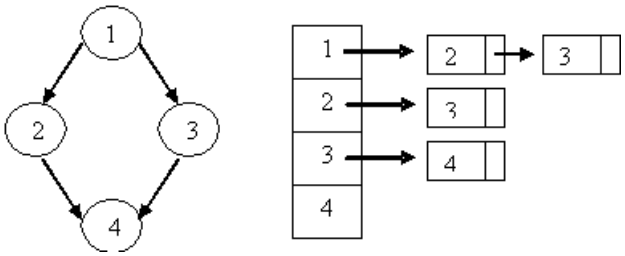


Figura 20: Exemplo de Grafo e Lista de Adjacência correspondente

3.3 Representação de Grafos em Prolog

Há várias formas de se representar grafos na linguagem Prolog. Por exemplo, o grafo da figura 21 poderia ser representado através de uma sequência de fatos, cada um representando uma ligação, ou seja, uma aresta do grafo.

```
Predicates
  Aresta(integer, integer).
Clauses
  Aresta(1,2).
  Aresta(1,3).
  Aresta(1,4).
  Aresta(2,4).
  Aresta(3,4).
```

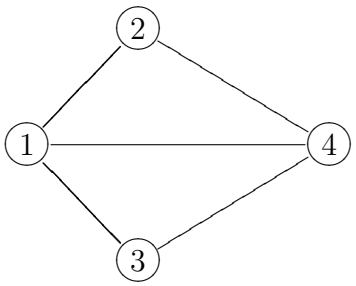


Figura 21: Exemplo de Grafo para ilustrar representaçao de fatos em Prolog

Se fosse necessário representar o fato de que as ligações são bi-direcionais, ou seja, que o grafo não é orientado, poderia ser utilizada a seguinte regra:

```
Predicates
  TemAresta(integer, integer).
Clauses
  TemAresta(X,Y) :- Aresta(X,Y); Aresta(Y,X).
```

Outra representação possível seria a que consiste em uma estrutura formada pela lista de nós adjacentes. O código a seguir apresenta uma lista cujos elementos são estruturas do tipo `no` que define o elemento e a lista de nós adjacentes a ele. Neste código estão presentes as regras para se montar um nó a partir dos fatos que representam as ligações entre os nós (`Aresta(X,Y)`) e uma regra para se criar um lista de nós (`grafo`) a partir da ordem do grafo.

```
Domains
  listaInt=integer*
  no=no(integer, listaInt)
  grafo=no*
Predicates
  Adjacencias(integer, listaInt).
  MontaNo(integer, no).
  MontaGrafo(integer, grafo).
Clauses
  Adjacencias(X, Lista) :-
    FindAll(Z, TemAresta(X,Z) Lista).
  MontaNo(N, no(N, Lista)) :-
    Adjacencias(N, Lista).
  MontaGrafo(0, []) :- !.
  MontaGrafo(N, [Cab|Cauda]) :-
    MontaNo (N, Cab), N1 = N - 1,
    MontaGrafo (N1, Cauda).
```

4 Algoritmos de Grafos

Este seção apresenta alguns dos principais algoritmos relacionados a busca em grafos e identificação de caminhos mínimos.

4.1 Algoritmo de Warshall

Seja G um grafo orientado com m vértices $v_1, v_2, \ldots v_m$. Suponha que se deseja encontrar a matriz de alcançabilidade do grafo G. Warshall propôs um algoritmo que é muito mais eficiente do que calcular as potências da matriz de adjacências M.

Primeiro defini-se as matrizes booleanas quadradas $m \times m$ $P_0, P_1, \ldots P_m$ como a

seguir. Seja $P_k[i,j]$ o elemento i,j da matriz P_k.Então:

$$P_k[i,j] = \begin{cases} 1 & \text{se existe um caminho simples de } v_i \text{ para } v_j \\ & \text{que não usa nenhum outro vértice exceto possivelmente } v_1, v_2, \ldots, v_k \\ 0 & \text{caso contrário} \end{cases}$$

Isto é,

$P_0[i,j] = 1$ se existe aresta de v_i para v_j

$P_1[i,j] = 1$ se existe um caminho simples de v_i para v_j que não usa nenhum outro vértice exceto possivelmente v_1

$P_2[i,j] = 1$ se existe um caminho simples de v_i para v_j que não usa nenhum outro vértice exceto possivelmente v_1 e v_2

E assim sucessivamente.

É importante observar que a matriz $P_0 = M$, a matriz de adjacências de G. Além disso, como G tem apenas m vértices, a última matriz $P_m = R$, a matriz de alcançabilidade de G.

Warshall observou que $P_k[i,j] = 1$ pode ocorrer apenas se um dos seguintes dois casos acontecer:

1. Existe um caminho simples de v_i para v_j que não usa nenhum outro vértice exceto possivelmente $v_1, v_2, \ldots, v_{k-1}$; logo, $P_{k-1}[i,j] = 1$.

2. Existe um caminho simples de v_i para v_k e um caminho simples de v_k para v_j onde cada caminho simples não usa nenhum outro vértice exceto possivelmente $v_1, v_2, \ldots, v_{k-1}$; logo, $P_{k-1}[i,k] = 1$ e $P_{k-1}[k,j] = 1$.

Conseqüentemente, os elementos de P_k podem ser obtidos por:

$$P_K[i,j] = P_{k-1} \vee (P_{k-1}[i,k] \wedge P_{k-1}[k,j])$$

A seguir o algoritmo de Warshall.

Algoritmo de Warshall – Um grafo orientado G com m vértices é representado na memória pela sua matriz de adjacências M. O algoritmo determina a matriz de alcançabilidade R do grafo G.

Begin

/* Iniciar R como imagem de M */

for(k=0; k<m; k++)

for(i=0; i<m; i++)
  for(j=0; j<m; j++)
    $R[i,j] = (R[i,j] \vee (R[i,k] \wedge R[k,j])$
End.

4.2 Algoritmo para o Caminho Mínimo

Seja G um grafo orientado com m vértices $v_1, v_2, \ldots v_m$. Supondo que G é ponderado, isto é, possui um peso associado a cada aresta. Então, G pode ser mantido na memória pela sua **matriz de pesos** $W = (w_{ij})$ definida por

$$w_{ij} = \begin{cases} w(a) & \text{se existe uma aresta } a \text{ de } v_i \text{ para } v_j \\ 0 & \text{caso contrário} \end{cases}$$

A matriz de caminhos R diz se existem ou não caminhos entre os vértices. Agora, deseja-se determinar a matriz Q, que apresenta os comprimentos dos caminhos mínimos entres os vértices ou, mais precisamente, a matriz $Q = (q_{ij})$, onde q_{ij} é o comprimento do menor caminho de v_i para v_j.

Descreve-se a seguir uma modificação no algoritmo de Warshall para determinar a matriz Q de maneira eficiente.

Defini-se um seqüência de matrizes $Q_0, Q_1, \ldots, Q_m$ (análogas às matrizes anteriormente definidas $P_0, P_1, \ldots P_m$) onde $Q_k[i,j]$ é definido como a seguir:

$Q_k[i,j] =$ menor valor entre comprimento do caminho procedente de v_i para v_j ou a soma dos comprimentos dos caminhos precedentes de v_i para v_k e de v_k para v_j.

Mais exatamente,

$$Q_k[i,j] = min(Q_{k-1}[i,j], Q_{k-1}[i,k] + Q_{k-1}[k,j])$$

A matriz inicial Q_0 é a mesma que a matriz de pesos W, exceto pelo fato de que cada 0 em W é substituído por ∞ (ou o maior número do tipo representado pela matriz). A matriz final Q_m será a matriz procurada Q.

4.3 Busca em Profundidade

A idéia geral de uma busca por profundidade começada pelo vértice x é descrita a seguir. Primeiramente, processa-se o vértice inicial x. Depois processa-se cada vértice v

ao longo de um caminho P que inicia em x; isto é, processa-se um vizinho de x, depois um vizinho de um vizinho de x, e assim por diante. Depois de ser atingido um vértice que não possui vizinhos ainda não processados, retrocede-se então no caminho P até que se possa continuar ao longo de outro caminho P, e assim por diante. O retrocesso é realizado utilizando-se uma estrutura PILHA contendo os vértices iniciais de novos possíveis caminhos. Também é necessário um campo, STATUS, que diz o estado corrente de qualquer vértice de tal forma que nenhum vértice seja processado mais de uma vez.

A seguir o algoritmo de busca em profundidade.

Algoritmo de Busca em Profundidade – Este algoritmo executa uma busca em profundidade em um grafo orientado G a partir de um vértice x.

Passo 1: Inicialize todos os vértices para o estado prontidão (STATUS = 1).

Passo 2: Insira o vértice de partida x na PILHA e mude seu estado para o estado de espera (STATUS=2).

Passo 3: Repita os Passos 4 e 5 até que a PILHA esteja vazia.

Passo 4: Retire o vértice v do topo da PILHA. Processe v, faça STATUS(v)=3, estado processado.

Passo 5: Examine cada vizinhança j de v:

1. Se STATUS(J)=1 (prontidão), insira j na PILHA e faça STATUS(J)=2 (estado de espera).
2. Se STATUS(J)=2 (espera), delete o j anterior da PILHA e insira o j corrente na pilha.
3. Se STATUS(J)=3 (processado), ignore o vértice j.

Passo 6: Saia.

A estrutura PILHA neste algoritmo não é tecnicamente uma pilha, uma vez que no Passo 5(2), permite-se que um vértice j seja deletado e posteriormente inserido no topo da pilha (embora seja o mesmo vértice j, representa uma aresta diferente na estrutura de adjacências). Sem a remoção do j no Passo 5(2), obtém-se uma forma alternativa para o algoritmo.

4.4 Busca em Largura

A idéia geral de uma busca em largura começada pelo vértice x é descrita a seguir. Primeiramente, processa-se o vértice inicial x. Depois processa-se todos os vizinhos de x. A seguir os vizinhos dos vizinhos de x, e assim sucessivamente. Naturalmente, necessita-se ter o controle dos vizinhos de um vértice, e necessita-se garantir também que nenhum vértice seja processado mais de uma vez. Isso é realizado utilizando uma estrutura FILA para conhecer os vértices aguardando processamento.

A seguir o algoritmo de busca em largura.

Algoritmo de Busca em Largura – Este algoritmo executa uma busca em largura em um grafo orientado G a partir de um vértice x.

Passo 1: Inicialize todos os vértices para o estado prontidão (STATUS = 1).

Passo 2: Insira o vértice de partida x na FILA e mude seu estado para o estado de espera (STATUS=2).

Passo 3: Repita os Passos 4 e 5 até que a FILA esteja vazia.

Passo 4: Retire o vértice v no início da FILA. Processe v, faça STATUS(v)=3, estado processado.

Passo 5: Examine cada vizinhança j de v:

1. Se STATUS(J)=1 (prontidão), insira j no final da FILA e faça STATUS(J)=2 (estado de espera).
2. Se STATUS(J)=2 (espera) ou STATUS(J)=3 (processado), ignore o vértice j.

Passo 6: Saia.

4.5 Ordenação Topológica

Seja S um grafo orientado tal que:

1. Cada vértice v_i de S representa uma tarefa;
2. Cada aresta (orientada) (u, v) de S significa que a tarefa u deve ser completada antes do início da tarefa v.

Supondo que um tal grafo S contém um ciclo, por exemplo $P = (u, v, w, u)$, isso significa que será necessário concluir a tarefa u antes de se iniciar a tarefa v, e será preciso terminar a tarefa v antes de iniciar w e, finalmente, será preciso completar a tarefa w antes de se iniciar a tarefa u. Logo, não é possível começar nenhuma das três tarefas no ciclo. Conseqüentemente, um grafo S deste tipo, representando tarefas relacionadas por pré-requisitos, não pode ter ciclos.

Uma operação fundamental em um grafo orientado acíclico S é o processamento dos vértices, um após o outro, de tal forma que o vértice u é sempre processado antes do vértice v se (u, v) é uma aresta. Esta ordenação linear T dos vértices de S, que pode não ser única, é dita **ordenação topológica**.

O algoritmo seguinte irá determinar uma ordenação topológica. A idéia central do algoritmo é de que qualquer vértice v com grau de entrada zero pode ser escolhido como primeiro elemento na ordem T. Essencialmente, o algoritmo repete os dois passos seguintes até que S esteja vazio:

1. Ache um vértice v com grau de entrada zero.
2. Delete v e suas arestas do grafo S.

Utiliza-se uma estrutura Fila auxiliar para guardar temporariamente todos os vértices com grau zero.

A seguir o algoritmo de ordenação topológica.

Algoritmo de Ordenação Topológica – Este algoritmo determina uma ordenação topológica T de um grafo orientado acíclico S.

Passo 1: Encontre todos os graus de entrada $deg^+(v)$ de cada vértice v de S

Passo 2: Insira todos os vértices de grau zero na FILA.

Passo 3: Repita os Passos 4 e 5 até que a FILA esteja vazia.

Passo 4: Remova e processe o primeiro vértice v da FILA.

Passo 5: Repita para cada vizinhança j de v:

1. Faça $deg^+(j) = deg^+(j) - 1$.
   /* Delete a aresta de v para j */
2. Se $deg^+(j) = 0$, então adicione j na FILA.

Passo 6: Saia.

5 Caminhos Eulerianos e Hamiltonianos

Os caminhos eulerianos são assim designados pela relação com o problema das pontes de Königsberg. Um **caminho euleriano** é um caminho de um grafo que contém cada aresta uma e uma só vez. Um caminho euleriano que seja fechado é designado por **circuito euleriano**.

Teorema 5.1 (Euler) *Um grafo (ou multigrafo) conexo possui um caminho euleriano se e somente se tiver um número de vértices de grau ímpar igual a 0 ou 2. O caminho euleriano é um circuito euleriano se aquele numero for 0; caso contrário, o caminho euleriano vai de um dos vértices de grau ímpar ao outro vértice também de grau ímpar.*

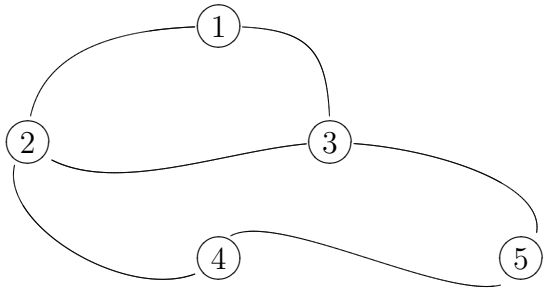
Um problema relacionado com o anterior, mas consideravelmente de maior dificuldade de resolução foi colocado pelo matemático irlandês W. Hamilton. Diz-se que um **caminho é hamiltoniano** se passar uma e uma só vez por cada um dos vértices do grafo.

Embora o problema da existência de ciclos hamiltonianos possa parecer semelhante ao problema da determinação de circuitos eulerianos de um grafo, a verdade é que não é nada fácil dizer se um grafo é ou não hamiltoniano em geral. Há alguns resultados parciais, mas não há resultados gerais.

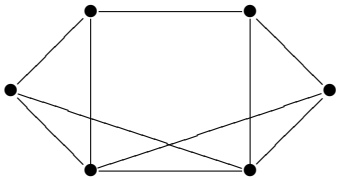
6 Exercícios

1. Implemente em linguagem C/C++ todos os algorimtos descritos nesta nota de aula. considere as representações de grafos por matriz de adjacências e lista de adjacências.
2. Desenho todos os grafos simples que é possível construir com 1, 2, 3 e 4 vértices.
3. Encontre o número cromático de cada grafo abaixo:

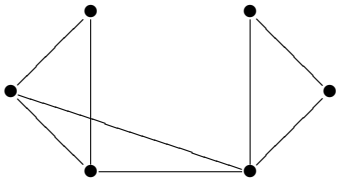
   (a)



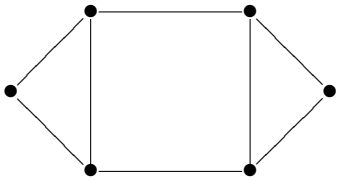
(b)



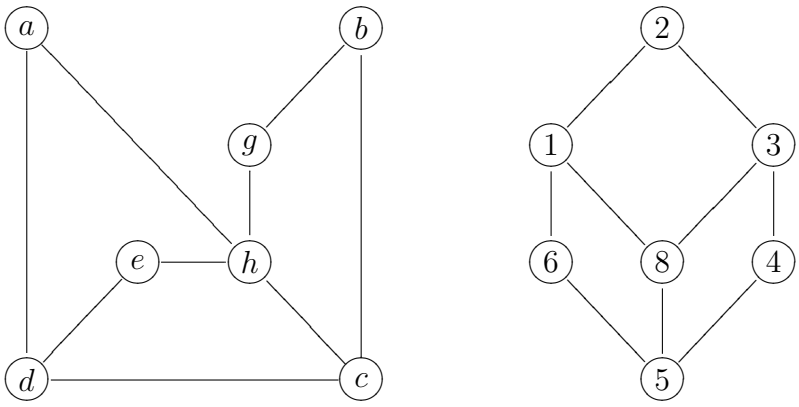
(c)



(d)



4. O complemento de um grafo simples $G = (V, A, g)$ é o grafo simples $G' = (V, A', g')$ no qual existe uma aresta entre dois vértices se, e somente se, não existe uma aresta entre os mesmos vértices em G. Desenhe o complemento dos seguintes grafos:

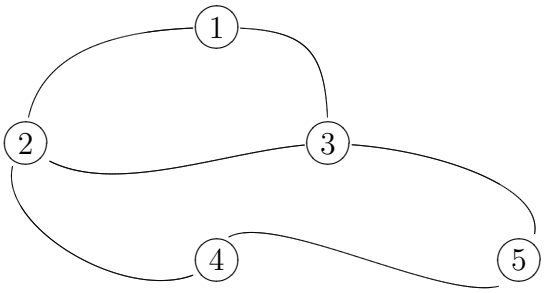


5. Mostre que $G \cup G'$ é um grafo completo.

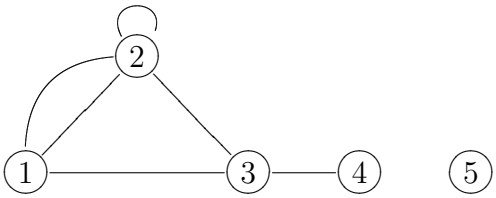
6. Seja G um grafo com v vértices e a arestas. Quantas arestas contém o grafo G'?

7. Escreva as matrizes de adjacências e as matrizes de incidências que representam os grafos seguintes:

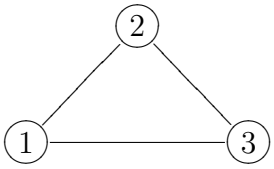
(a)



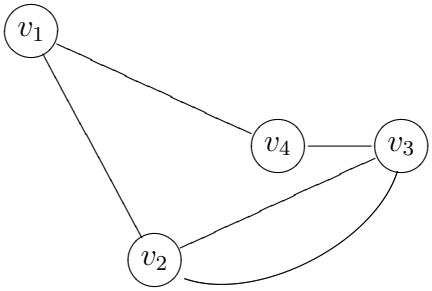
(b)



(c)



(d)



8. Descreva o grafo cuja matriz de adjacências é I_n a matriz identidade $n \times n$

9. A matriz de adjacências para um grafo não-direcionado é dada em sua forma triangular inferior por:

$$\begin{bmatrix} 2 & & & \\ 1 & 0 & & \\ 0 & 1 & 1 & \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

   Desenhe o grafo que ela representa.

10. A matriz de adjacências para um grafo direcionado é dada por:

$$\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 2 & 0 \end{bmatrix}$$

Desenhe o grafo que ela representa.

11. Descreva o grafo direcionado cuja matriz de adjacências tem todos os 1s na linha 1 e na coluna 1 e nas demais posições tem 0s.

12. Descreva o grafo direcionado cuja matriz de adjacências tem 1s nas posições $(i, i+1)$ para $0 \leq i < n-1$, e um 1 na posição $(n-1, 0)$ e 0s nas demais posições.

13. Seja A a matriz:

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

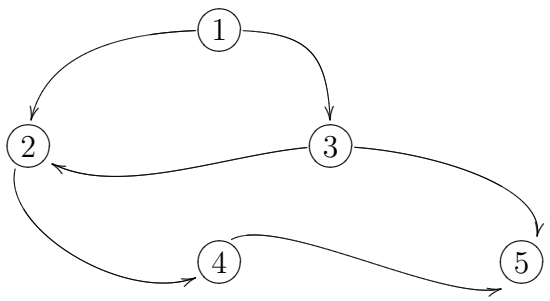
Encontre $A^{(2)}$ e $A^{(3)}$.

14. Seja G um dígrafo representado pela matriz abaixo:

$$\begin{bmatrix} 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Encontre a matriz de alcançabilidade.

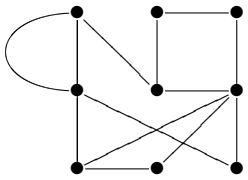
15. Use o algoritmo de busca em profundidade para realizar uma ordenação topológica no grafo da figura a seguir:



16. Um grafo com quatro vértices ímpares pode ser conexo.

17. Um grafo cúbico é um grafo simples regular de grau 3. Construa 2 grafos cúbicos não-isomorfos.

18. Determinar um circuito euleriano no seguinte grafo:



19. Faça uma função para verificar se um grafo orientado é conexo.

20. Desenhar um grafo cuja matriz de adjacência é tal que:

$$A^{(2)} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 3 & 1 & 1 \\ 1 & 1 & 2 & 1 \\ 1 & 1 & 1 & 2 \end{bmatrix}$$

e

$$A^{(3)} = \begin{bmatrix} 0 & 3 & 1 & 1 \\ 3 & 2 & 4 & 4 \\ 1 & 4 & 2 & 3 \\ 1 & 4 & 3 & 2 \end{bmatrix}$$

21. Faça uma função que determine o grau de cada vértice de um grafo, devolvendo o resultado em um vetor de inteiros.

22. O problema do cavalo, ou passeio do cavalo, é um problema matemático envolvendo o movimento da peça do cavalo no tabuleiro de xadrez. O cavalo é colocado no tabuleiro vazio e, seguindo as regras do jogo, precisa passar por todas as casas exatamente uma vez. Existem diversas soluções para o problema, dentre elas 26.534.728.821.064 terminam numa casa da qual ele ataca a casa na qual iniciou o seu movimento.

    Durante séculos muitas variações desse problema foram estudadas por matemáticos, incluindo Euler que em 1759 foi o primeiro a estudar cientificamente esse problema. As variações do problema são:

    - tamanhos diferentes de tabuleiro
    - número de jogadores
    - maneira com que o cavalo se move.

Usando a chamada "notação algébrica" do xadrzez, pede-se modelar o problema de levar o cavalo de **g1** para **b1** sem cair em nenhuma casa sombreada da figura 22.

Figura 22: O Problema do Cavalo

Referências

GERSTING, J. L. *Fundamentos Matemáticos Para a Ciência da Computação*. Rio de Janeiro: Livros Técnicos e Científicos Editora S.A., 1995. 518 p.

LIPSCHUTZ, S.; LIPSON, M. *Teoria e Problemas de Matemática Discreta*. Porto Alegre: Bookman, 2004. 511 p.

PINTO, J. S. *Tópicos de Matemática Discreta*. [S.l.], 2005.

ROSEN, K. H. *Discrete Mathematics and Its Applications*. 5th. ed. New York: McGraw Hill, 2005.