TEG

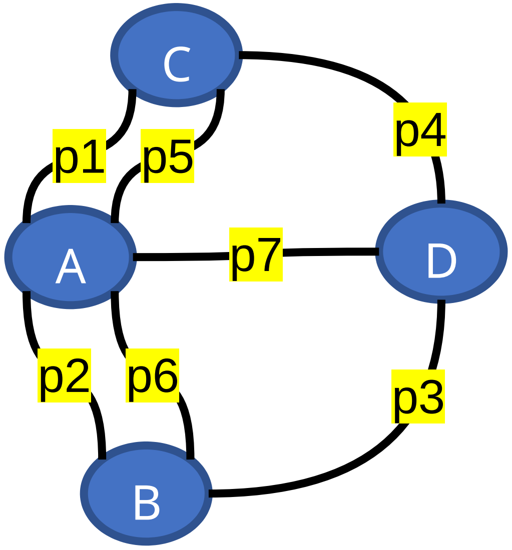
Gilmário B. Santos

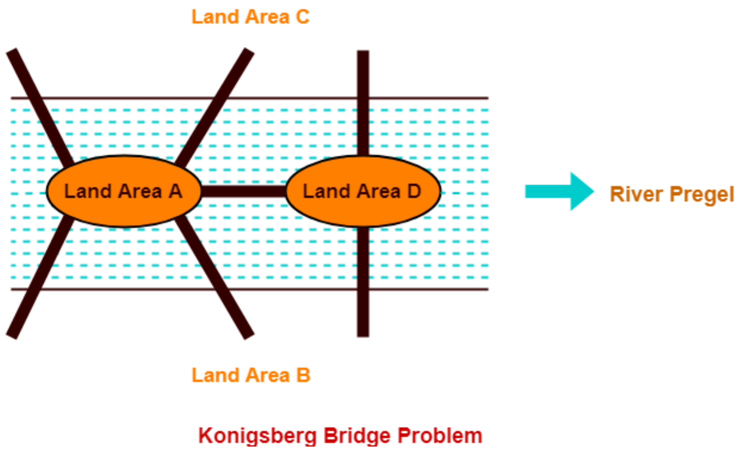
gilmario.santos@udesc.br

http://www.joinville.udesc.br/portal/pagina/gilmario

Grafos Eulerianos

Analisando problema das pontes de Konigsberg "começando uma caminhada por qualquer uma das partes de terra firme A, B, C, D, é possível cruzar cada uma das sete pontes exatamente uma vez e retornar ao ponto de partida sem ter que nadar pelo rio Pregel?"





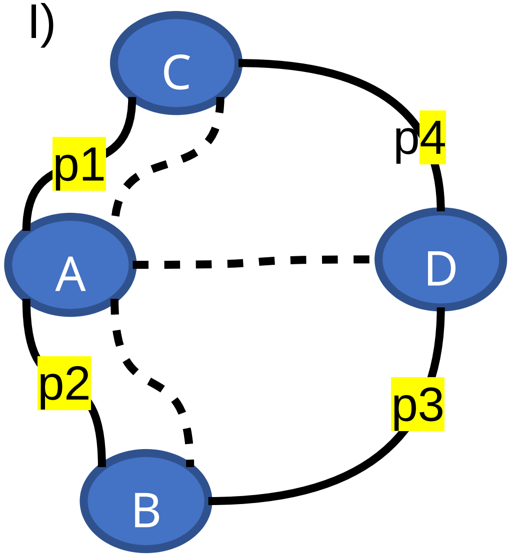
Grafos Eulerianos

Euler determinou as condições para tal travessia desejada partindo de qualquer das áreas de terra:

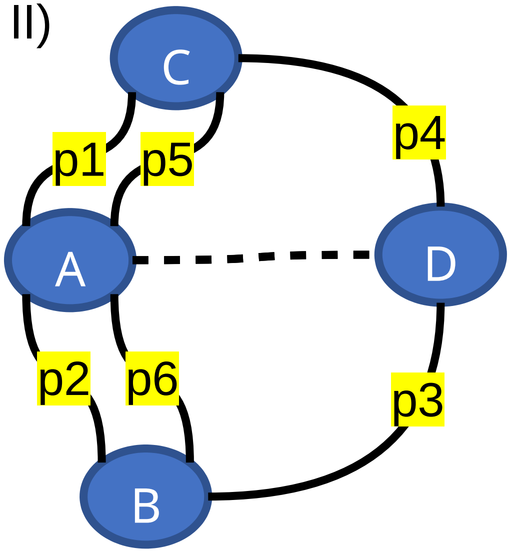
I. Ausência de vértices de grau ímpar (ou seja, um vértice de saída poderá ser o vértice de chegada do trajeto);

II. Ou a ocorrência de exatamente dois vértices de graus ímpares, nesse caso haveria uma trilha não fechada onde um vértice de grau ímpar seria a saída e outro seria a chegada

I)



II)



Nenhuma das duas condições estava atendida no caso real das pontes (há 4 vértices de grau ímpar).

Euler concluiu pela impossibilidade da travessia desejada.

Grafos Eulerianos

O estudo de Euler é considerado precursor na teoria de grafos

O estudo de Euler sobre o problema das pontes de Konigsberg fundou o conceito de Grafo Euleriano.

Sabemos que, se todos os vértices de um grafo G apresentam grau >= 2, então G contém um ciclo.

Se G(V,E) é um grafo Euleriano ele possui um passeio fechado contendo todas as arestas de G sem repeti-las, ou seja, G possui uma trilha fechada.

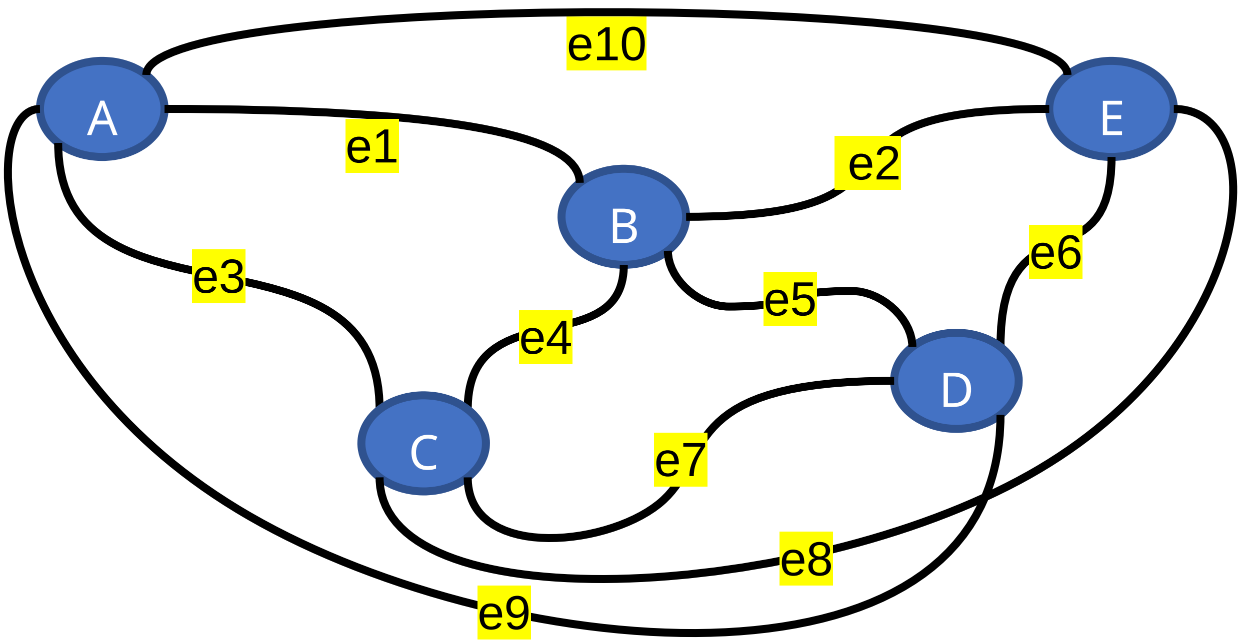
Grafos Eulerianos

Considerando G um grafo conexo não orientado, G é Euleriano:

1. Se e somente se todos os seus vértices possuem graus pares;

   Exemplo:

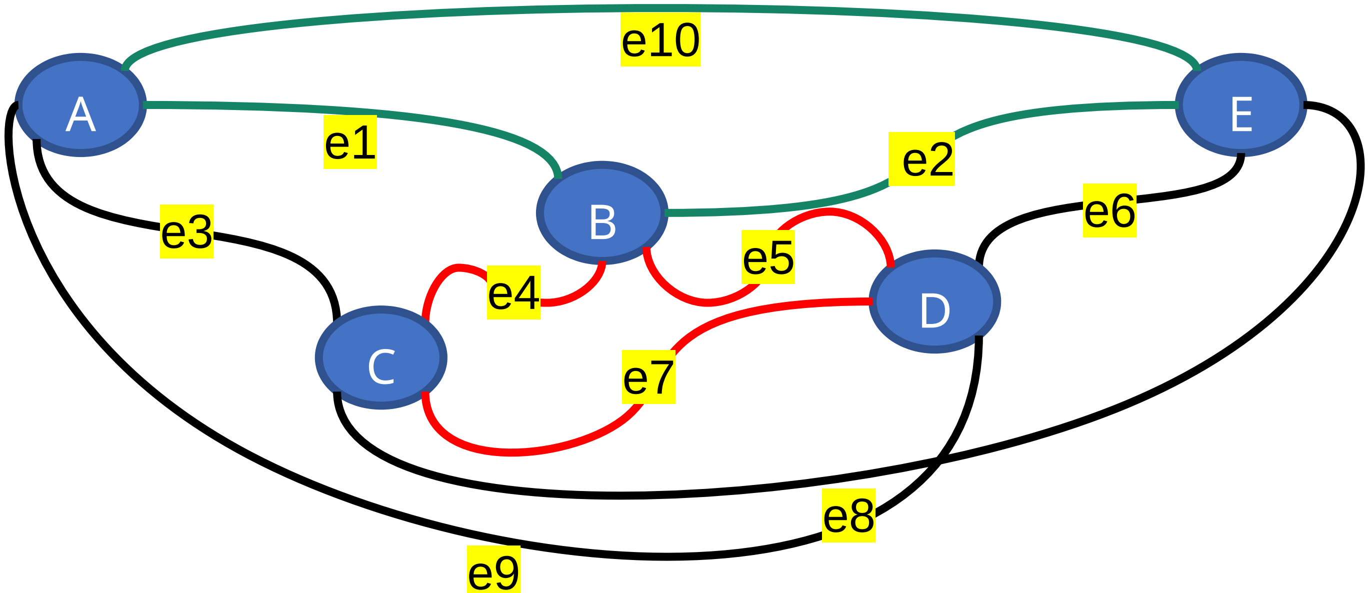
   A e1 B e2 E e6 D e5 B e4 C e7 D e9 A e3 C e8 E A



Grafos Eulerianos

Considerando G um grafo conexo não orientado, G é Euleriano:

1. Se e somente se todos os seus vértices possuem graus pares;
2. Se e somente se o conjunto E puder se redividido em ciclos disjuntos (não compartilham arestas).

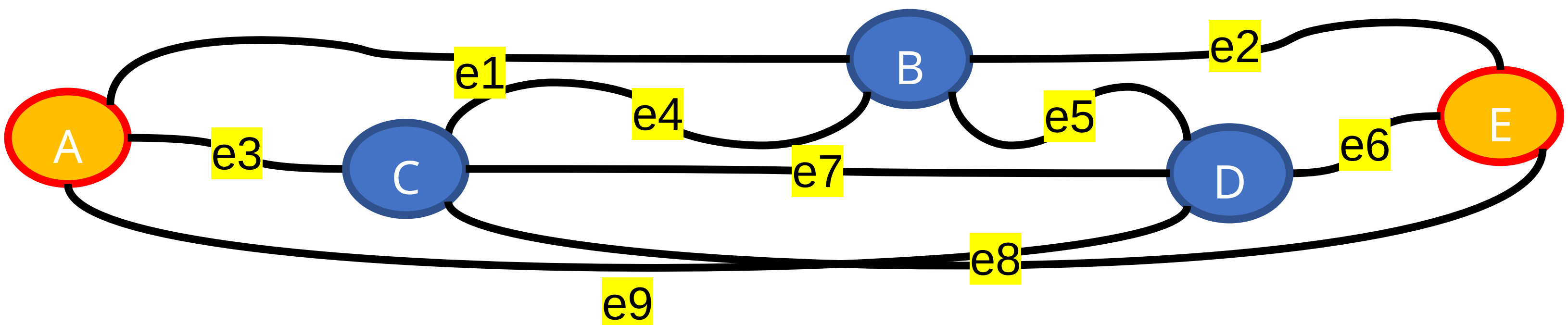


Grafos Eulerianos

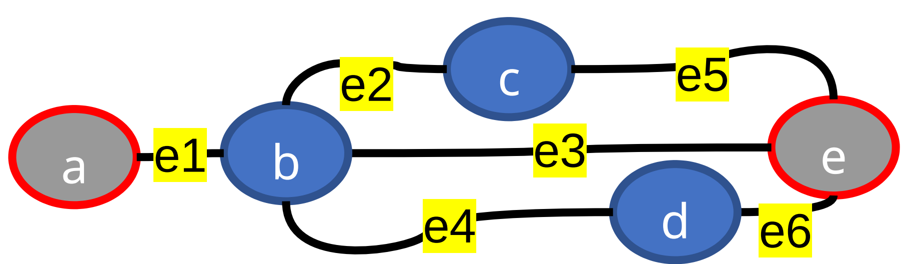
Considerando G um grafo conexo não orientado, G é semieuleriano (trilha não fechada) se e somente se houver exatamente dois vértices com graus ímpares.

Nesse caso, a trilha se inicia em um dos vértices de grau ímpar e se encerra no outro vértice de grau ímpar:

A e1 B e2 E e6 D e5 B e4 C e7 D e9 A e3 C e8 E



a e1 b e2 c e5 e e3 b e4 d e6 e



Grafos Eulerianos

Um algoritmo simples que determina se o grafo é Euleriano ou semieuleriano apenas verifica os graus dos vértices do grafo:

Se todos os graus forem pares→ é Euleriano

Se exatamente dois graus forem ímpares→ semieuleriano

Perceba que um grafo pode não ser nem Euleriano nem semieuleriano.

Sendo um grafo Euleriano ou semieuleriano, o algoritmo de Fleury determina a trilha fechada ou a trilha aberta Euleriana.

Grafos Eulerianos

Algoritmo de Fleury (1883)

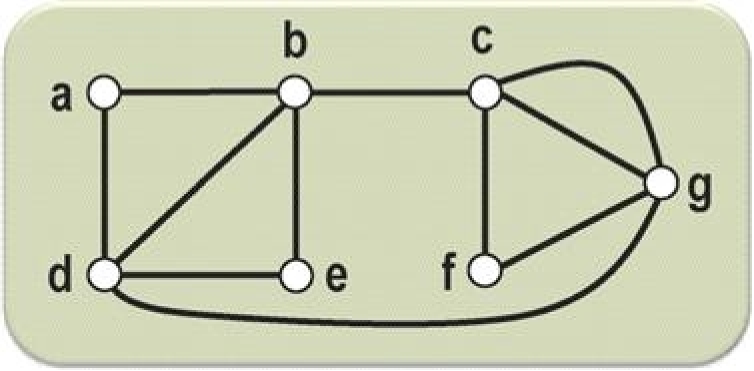
Determina uma trilha Euleriana ou semieuleriana, a diferença é que no segundo caso, os vértices de graus ímpares ocorrem necessariamente nas extremidades da trilha (aberta).

Fleury trabalha com um grafo reduzido induzido pelas arestas ainda não marcadas pelo algoritmo.

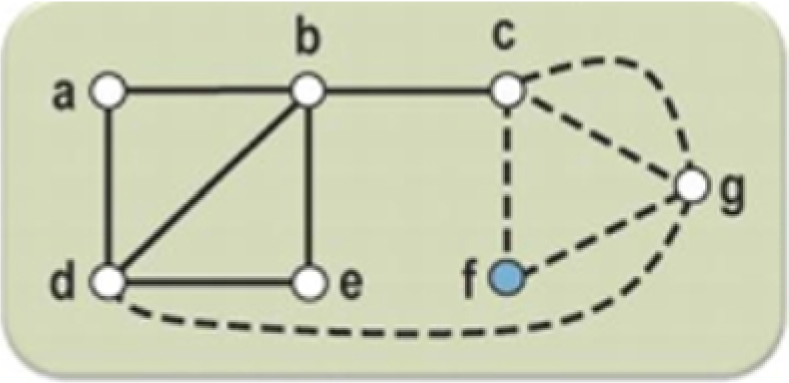
> Inicialmente um vértice *v* do grafo é escolhido (no caso semieuleriano este vértice é um dos dois de grau ímpar).
>
> O vértice é adicionado à lista C.
>
> A cada iteração, uma aresta (v,w) é escolhida para ser percorrida. Neste ponto, a **regra da ponte** deve ser observada:
>
> > Se uma aresta (v,w) é uma ponte no grafo induzido pelas arestas não marcadas, então (v,w) só deve ser escolhida caso não haja qualquer outra opção.

Grafos Eulerianos

Relembrando:
Subgrafo induzido por arestas:
se G = (V,E) é um grafo e D $\subseteq$ E, então o subgrafo de G induzido por D é o grafo G[D] = (U,D), onde U é o conjunto dos vértices que são extremidades das arestas em D.



G=(V,E)



G[D]=(U,D)
U={a,b,c,d,e}
D=E-{dg, gc, cf, fg, cg}

Grafos Eulerianos

Se o grafo G(V,E) for Euleriano, Fleury encontra uma trilha Euleriana válida

função Fleury(G = (V,E): grafo) : trilha

G' := G {G' = (V', E')}

v_0 := um vértice de G'

C := [v_0] {Inicialmente, o circuito contém apenas v_0}

Enquanto E' não vazio

v := cópia do último vértice de *C* {*C* é de fato uma pilha}

Se ***v*** tem só uma aresta incidente; {aresta de ponte é evitada ao máximo}

a := aresta que incidente ao vértice ***v*** em *G'*

Senão

a := aresta que não é ponte e é incidente ao vértice ***v*** em *G'*

Retirar aresta ***a*** do grafo *G'*

Acrescentar ***a*** no final de *C*

w := vértice ligado ao vértice *v* pela aresta ***a***

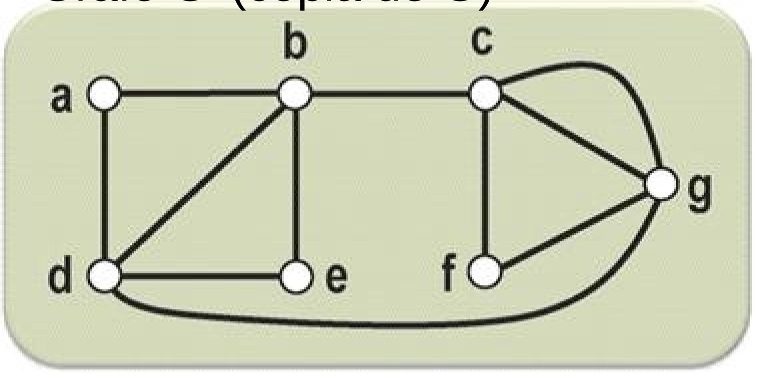
Acrescentar ***w*** no final de *C*

Retornar *C*

Grafos Eulerianos

Exemplo de um grafo Euleriano sendo explorado por Fleury, retirado de Goldbarg (2012) :

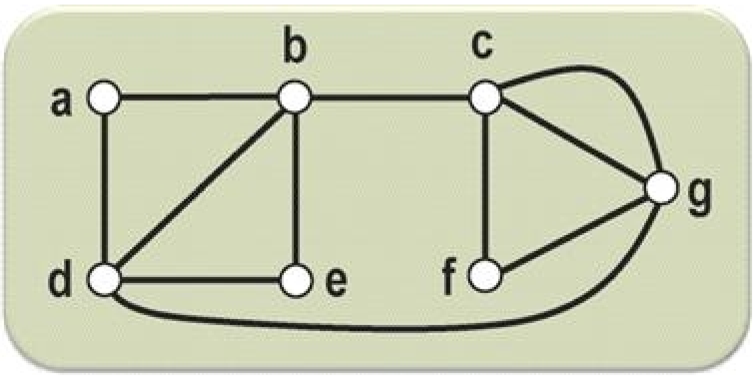
Grafo G’ (cópia de G)

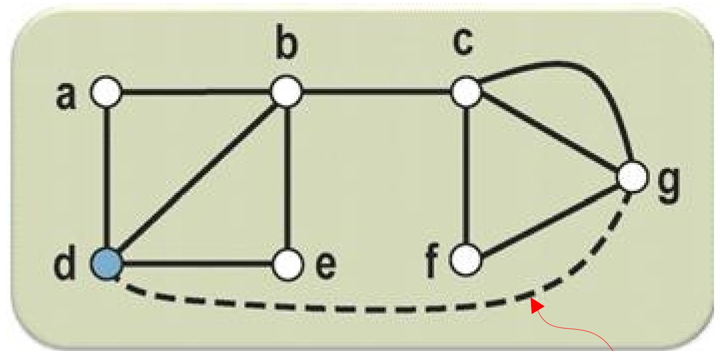


v_0=d
Empilha v_0=d
C={d}
⌂

GOLDBARG, Marco. Grafos. Rio de Janeiro: GEN LTC, 2012. E-book.

Grafos Eulerianos





(1) $(v, w) = (d, g)$

C={d}

^

v=d (copia em *v* o topo da pilha *C*)

Escolhe aresta que não seja ponte e incida sobre o vértice v=d

a= $\overline{dg}$

w=g

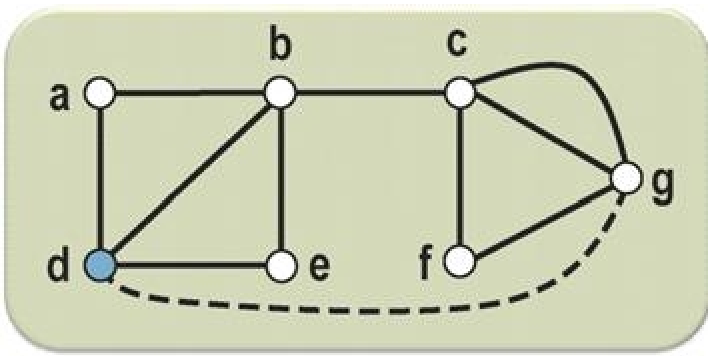
Retira aresta $\overline{dg}$ do grafo *G'*

Empilha $\overline{dg}$ e empilha w=g

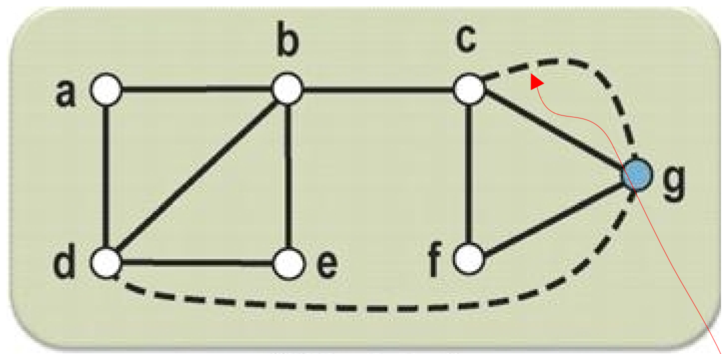
C={d,$\overline{dg}$,g}

^

Grafos Eulerianos



(1) $(v, w) = (d, g)$



(2) $(v, w) = (g, c)$

C={d,$\overline{dg}$,g}
^

v=g (copia em *v* o topo da pilha *C*)

Escolhe aresta que não seja ponte e incida sobre o vértice v=d

a= $\overline{gc}$

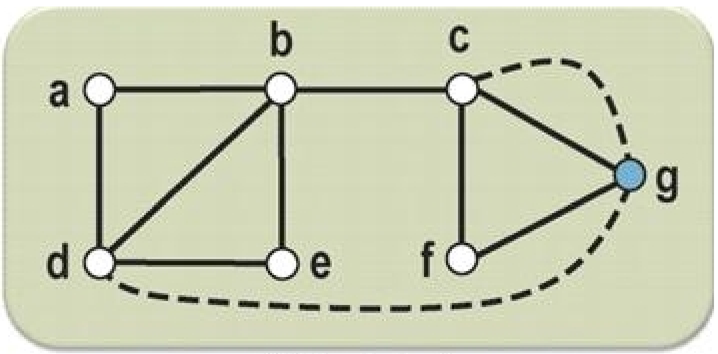
w=c

Retira aresta $\overline{gc}$ do grafo *G′*

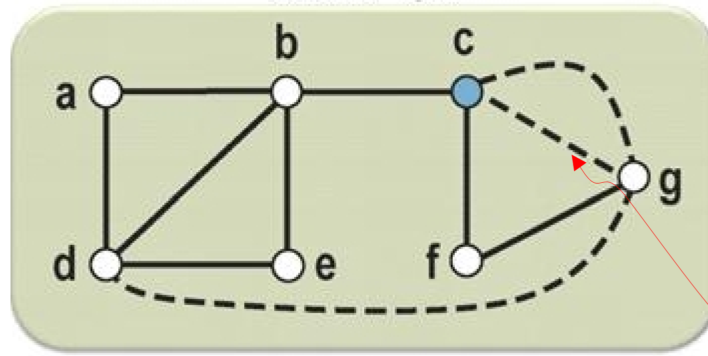
Empilha $\overline{gc}$ e empilha w=c

C={d,$\overline{dg}$,g,$\overline{gc}$,c}
^

Grafos Eulerianos

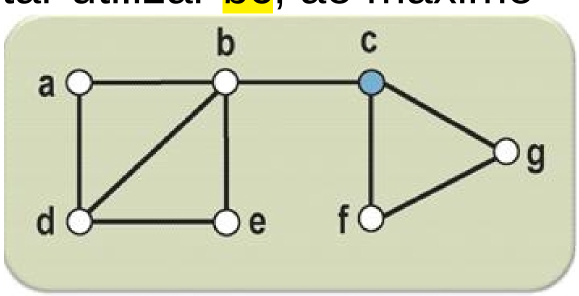


(2) (v, w) = (g, c)



(4) (v, w) = (c, g)

A aresta $\overline{bc}$ é uma ponte no subgrafo induzido pelas arestas ainda não marcadas.
Pela "regra da ponte", deve-se evitar utilizar $\overline{bc}$, ao máximo



(3) Grafo induzido na 3ª iteração

$C=\{d,\overline{dg},g,\overline{gc},c\}$

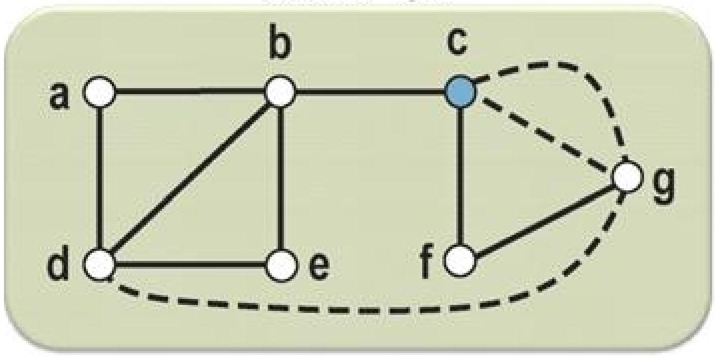
^

v=c (topo da pilha *C*)

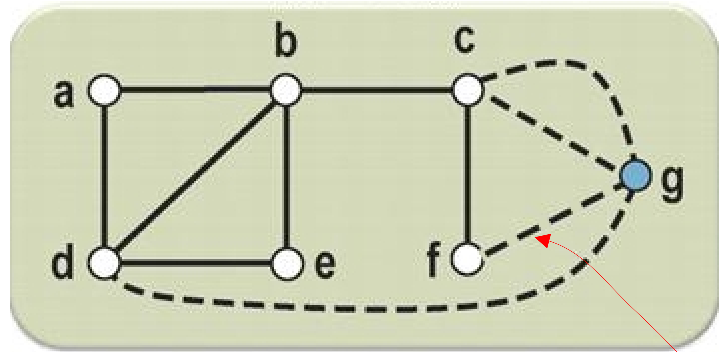
Escolhe aresta que não seja ponte e incida sobre o vértice *v=c*
a= $\overline{cg}$
w=g
Retira aresta $\overline{cg}$ do grafo *G′*
Empilha $\overline{cg}$ e empilha w=g
$C=\{d,\overline{dg},g,\overline{gc},c,\overline{cg},g\}$

^

Grafos Eulerianos



(4) $(v, w) = (c, g)$



(6) $(v, w) = (g, f)$

$C=\{d,\overline{dg},g,\overline{gc},c,\overline{cg},g\}$

^

v=g (cópia do topo da pilha *C*)

Escolhe aresta que não seja ponte e incida sobre o vértice v=g

a= $\overline{gf}$

w=f

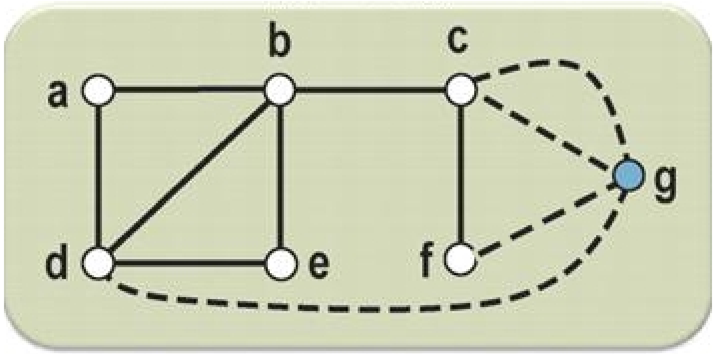
Retira aresta $\overline{gf}$ do grafo *G'*

Empilha $\overline{gf}$ e empilha w=f

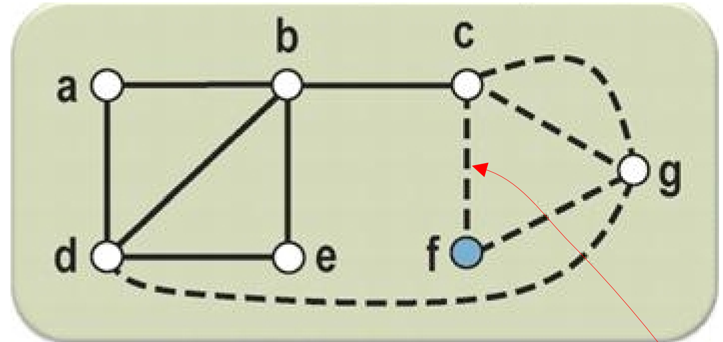
$C=\{d,\overline{dg},g,\overline{gc},c,\overline{cg},g,\overline{gf},f\}$

^

Grafos Eulerianos



(6) $(v, w) = (g, f)$



(7) $(v, w) = (f, c)$

$C=\{d,\overline{dg},g,\overline{gc},c,\overline{cg},g,\overline{gf},f\}$

^

v=f (cópia do topo da pilha *C*)

Escolhe aresta que não seja ponte e incida sobre o vértice v=f

a= $\overline{fc}$

w=c

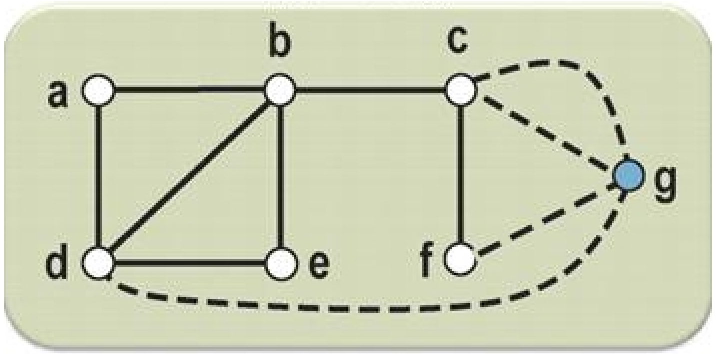
Retira aresta $\overline{fc}$ do grafo *G'*

Empilha $\overline{fc}$ e empilha w=c

$C=\{d,\overline{dg},g,\overline{gc},c,\overline{cg},g,\overline{gf},f,\overline{fc},c\}$

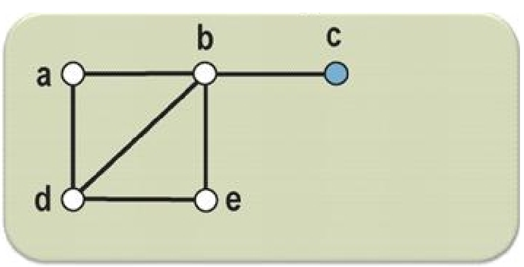
^

Grafos Eulerianos

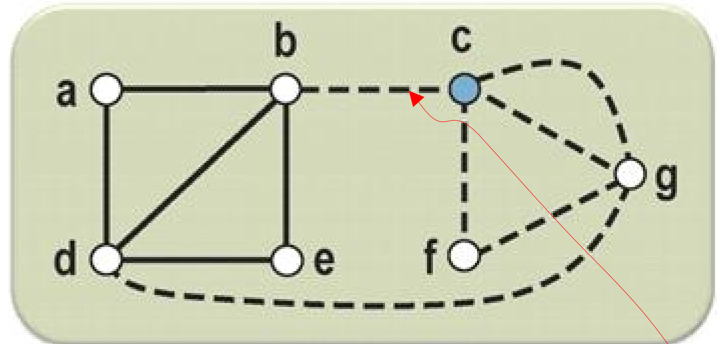


(6) (v, w) = (g, f)

Apesar da aresta $\overline{bc}$ ser uma ponte, até agora evitada, ela é a última aresta que incide em C ainda não marcada.



(8) Grafo induzido na 6ª iteração



(9) (v, w) = (c, b)

$C=\{d,\overline{dg},g,\overline{gc},c,\overline{cg},g,\overline{gf},f,\overline{fc},c\}$

^

v=c (cópia do topo da pilha *C*)

Escolhe última aresta incidente a *v=c* e ainda não marcada, mesmo que seja uma ponte.

a= $\overline{cb}$

w=b

Retira aresta $\overline{cb}$ do grafo *G'*

Empilha $\overline{cb}$ e empilha w=b

$C=\{d,\overline{dg},g,\overline{gc},c,\overline{cg},g,\overline{gf},f,\overline{fc},c,\overline{cb},b\}$

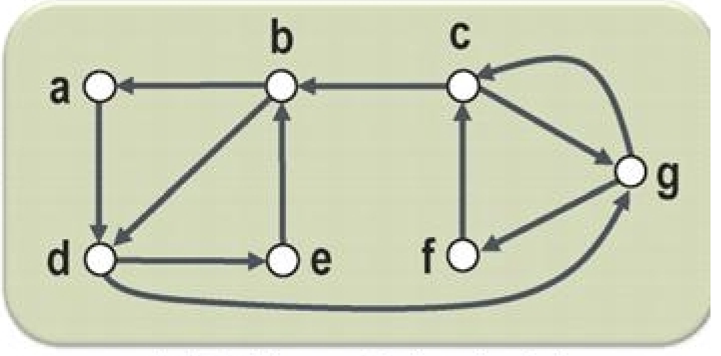
^

Grafos Eulerianos

O processo se repete até que

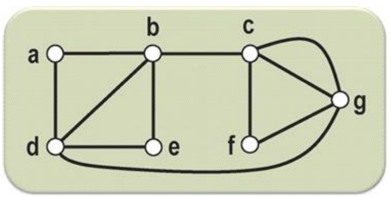
Todas as arestas tenham sido removidas de *G'* e acrescentadas à *C*

O Laço "Enquanto E' não vazio" é interrompido e *C* apresenta a trilha Euleriana procurada

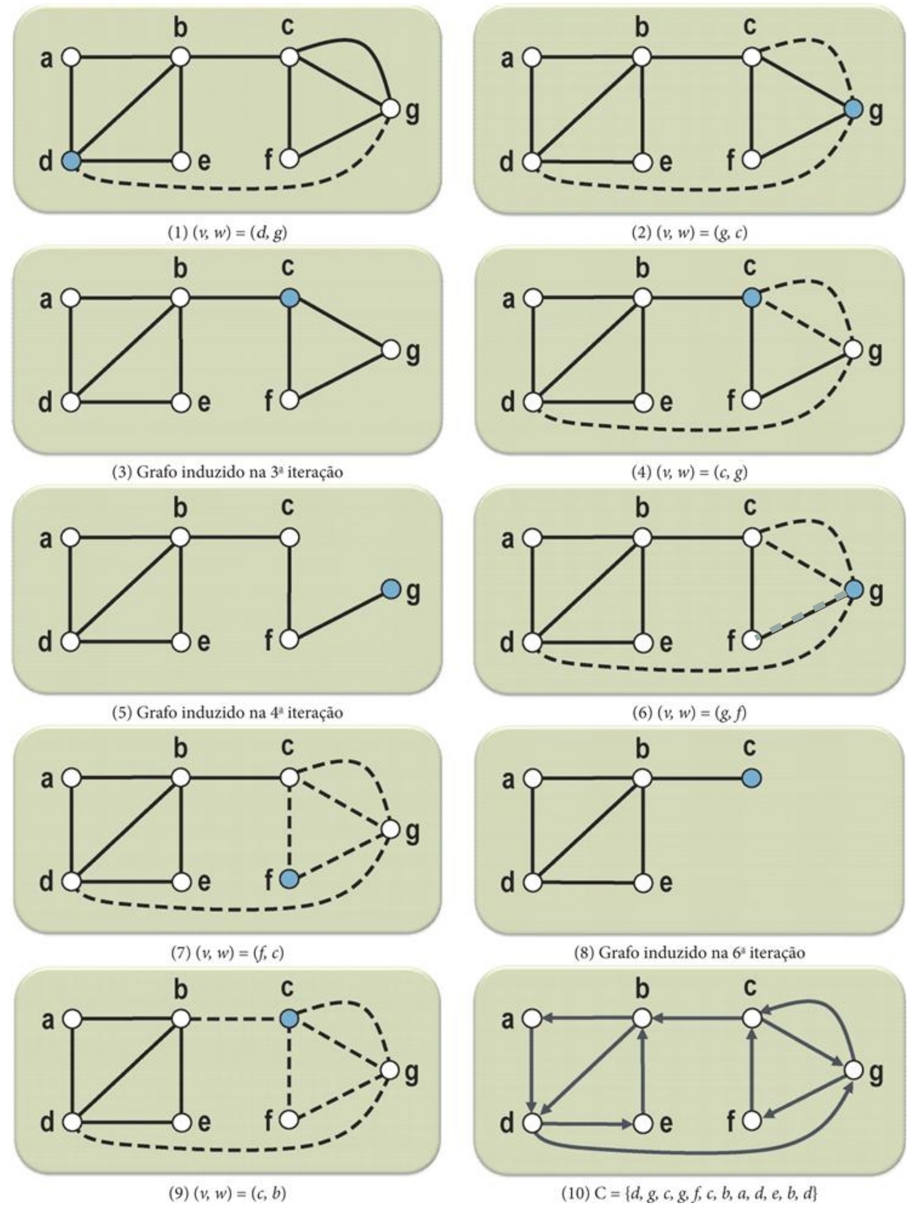


(10) C = {*d*, *g*, *c*, *g*, *f*, *c*, *b*, *a*, *d*, *e*, *b*, *d*}

Grafos Eulerianos

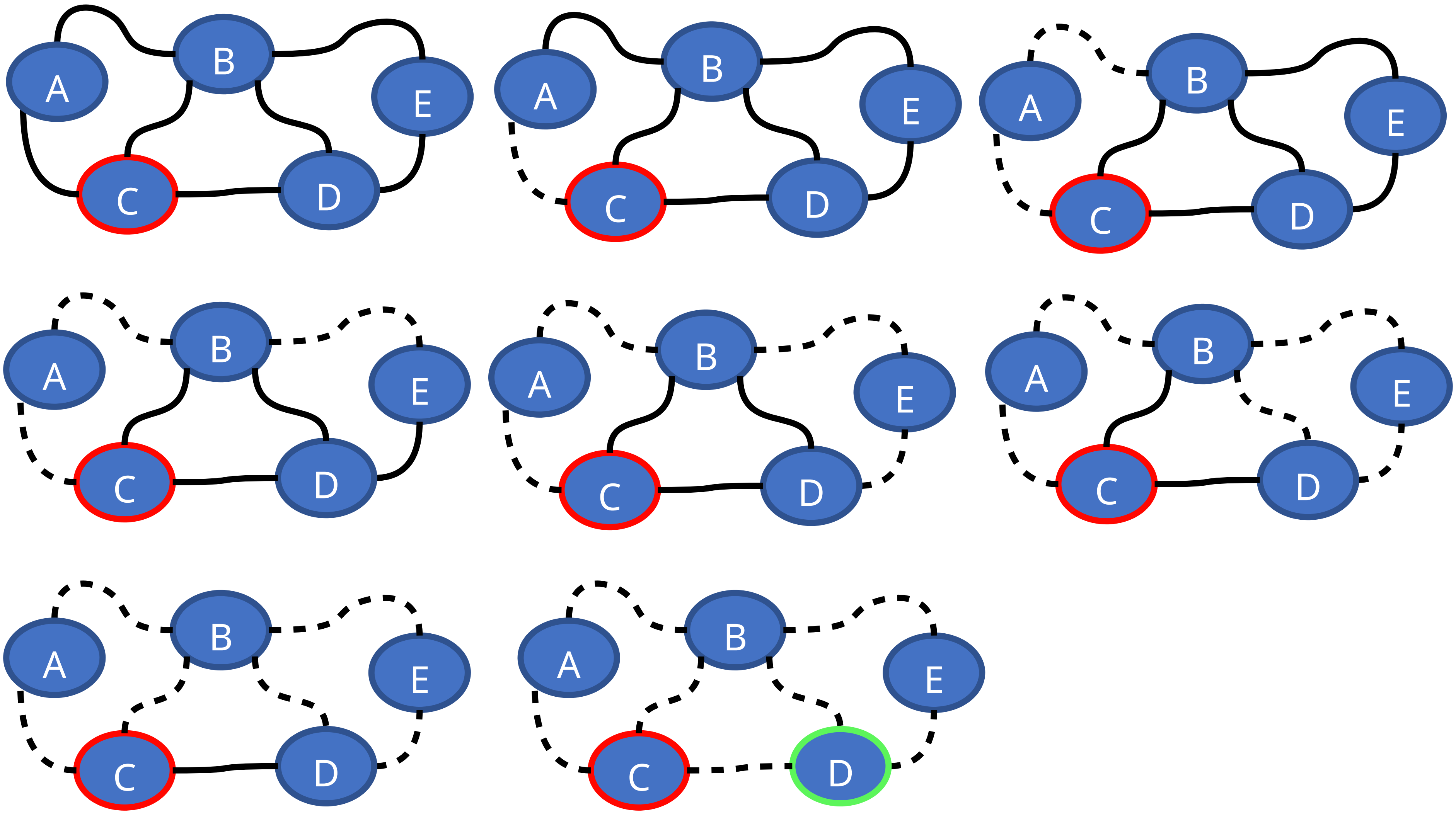


Ainda que haja a necessidade de testar a condição de ponte da aresta escolhida (o que pode ser feito com um algoritmo de percurso em grafos), o algoritmo de Fleury para G(V,E) terá complexidade O(|E|^2).
(Goldbarg,2012)



Grafos Eulerianos

Para um grafo semieuleriano, bastaria iniciar Fleury com um dos vértices de grau ímpar e se teria a desejada trilha Euleriana (aberta).



Grafos Eulerianos

Problema do carteiro chinês

Neste problema, discutido pelo matemático chinês Mei-Ku Kwan (Wilson), um carteiro deseja entregar suas cartas percorrendo a menor distância total possível e retornando ao seu ponto de partida.

Obviamente, o carteiro deve percorrer cada estrada em sua rota pelo menos uma vez, mas deve evitar repetir estradas.

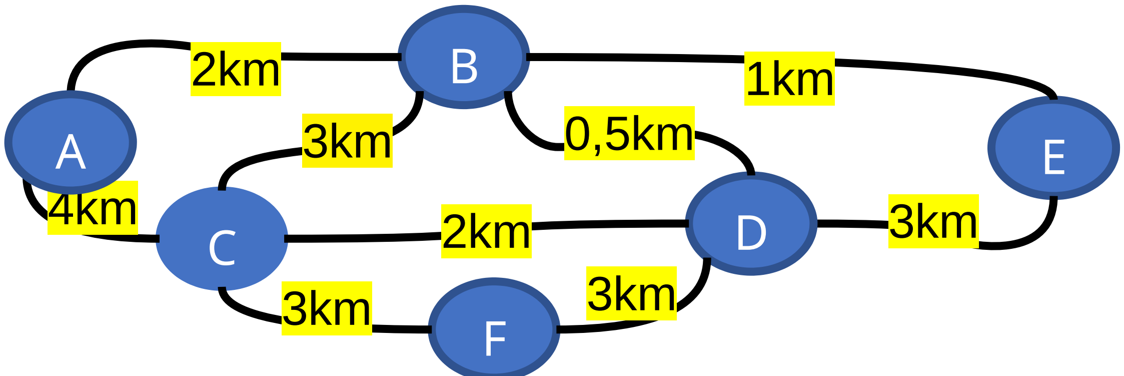
Grafos Eulerianos

Problema do carteiro chinês

O requisito é encontrar um trilha fechada de peso total mínimo que inclua cada aresta uma única vez (ou pelo menos evitar ao máximo percorrer a mesma aresta mais de uma vez).

Se o grafo é Euleriano, então qualquer trilha Euleriana é uma caminhada fechada do tipo requerido (afinal o peso total do trilha Euleriano é a soma dos pesos de todas as arestas do grafo, sem repetições).

Uma caminhada pode ser encontrada pelo algoritmo de Fleury: B – A - C - F - D - E - B - C - D – B, ao óbvio custo de custo=21,5km.



Grafos Eulerianos

Problema do carteiro chinês

Ainda sobre o problema do carteiro chinês, suponha que o grafo seja semieuleriano, nesse caso a trilha fechada dependerá de repetição de aresta(s).

Ainda assim, pode ser encontrada uma solução que apesar de não ser a ideal, em termos de custo mínimo, pois certamente ocorrerá repetição de aresta, apresentará um resultado prático útil.

A ideia é perseguir uma solução que seja "boa o suficiente". Para entender isso, pense no conceito de otimização:

> Otimização matemática: seleção de um melhor elemento, com relação a algum critério, de algum conjunto de alternativas disponíveis

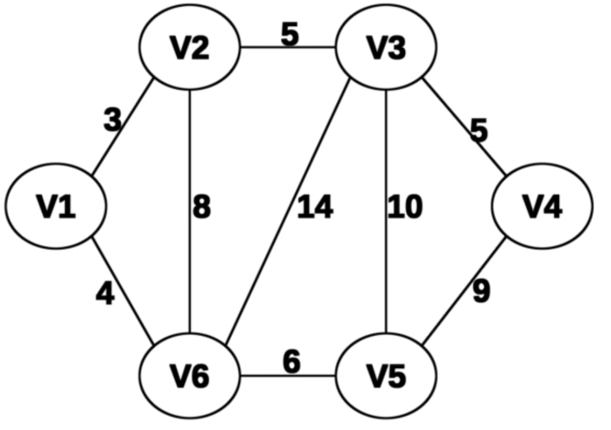
Grafos Eulerianos

Se o grafo é Euleriano, aplicamos Fleury. Porém, nem sempre a modelagem determina um grafo desse tipo.

Se o grafo não é Euleriano, embora exista algoritmo eficiente para sua solução, o problema se torna mais difícil. É necessário "Eulerizar" o grafo. Discussões sobre esse processo são encontradas nos livros:

Wilson, R.J. Introduction to Graph Theory. John Wiley & Sons Inc., 1985
Boaventura, Paulo. Grafos: introdução e prática. Blucher.
Bondy, A., Murty, USR. Graph theory with applications

A título de ilustração, a seguir discutiremos um caso especial, em que exatamente dois vértices têm grau ímpar e uma solução "suficientemente boa" é encontrada.



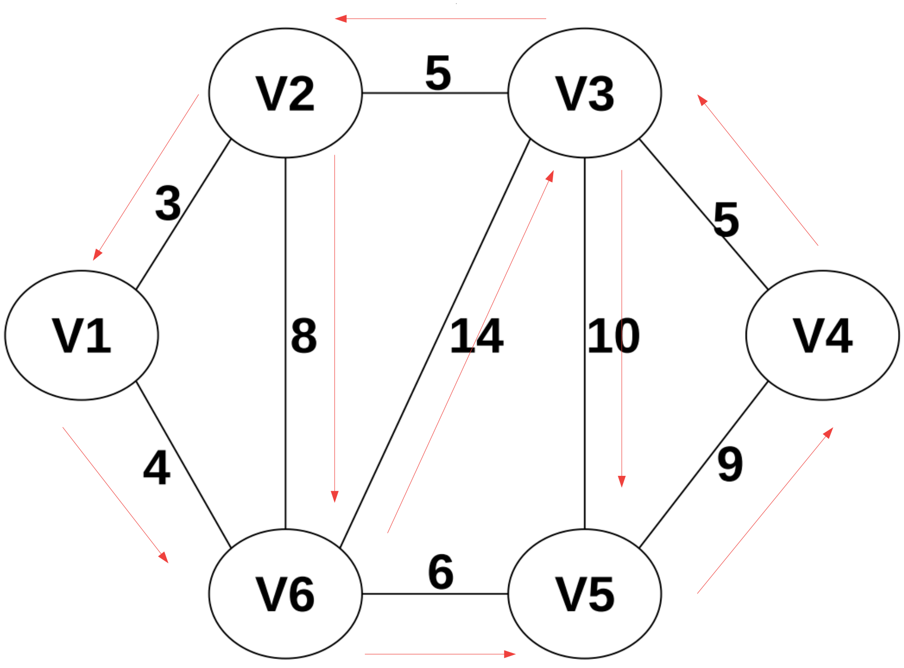
Grafos Eulerianos

Buscaremos um percurso que seja o mais curto, repetindo o mínimo possível de arestas. Inicialmente aplicamos Fleury para o caso original semieuleriano:

O grafo é semieuleriano: dois vértices de grau ímpar v2 e v5

Trilha semieuleriana: v2-v1-v6-v5-v4-v3-v2-v6-v3-v5

Custo total= 64

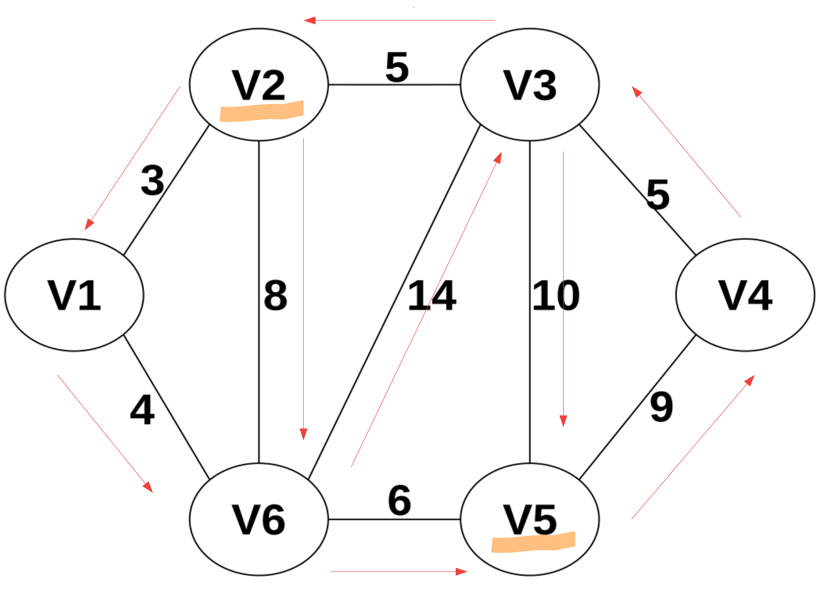


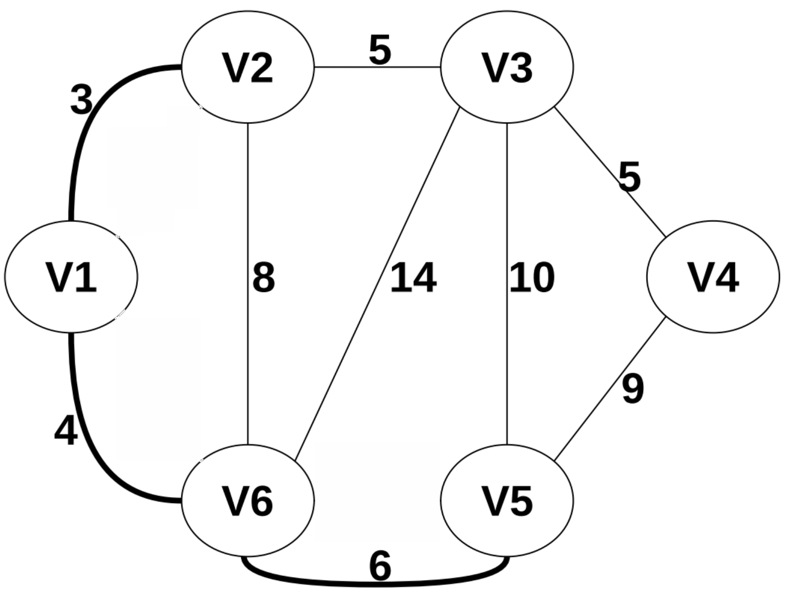
Grafos Eulerianos

Trilha semieuleriana: v2-v1-v6-v5-v4-v3-v2-v6-v3-v5   Custo total= 64

Para fechar a trilha (retorno de v5 a v2) percorrendo a menor distância possível, devemos determinar o caminho mais curto de v5 a v2 usando, por exemplo, o algoritmo do tipo Dijkstra.

A solução: tomar o caminho mais curto v5 —> v6 —> v1 —> v2 juntamente com a trilha semieuleriana inicial, o que fornece uma distância total de 13 + 64 = 77. Este é o menor trajeto, repetindo o mínimo de arestas.





Grafos Eulerianos

Trilha semieuleriana por Fleury: v2-v1-v6-v5-v4-v3-v2-v6-v3-v5, Custo = 64

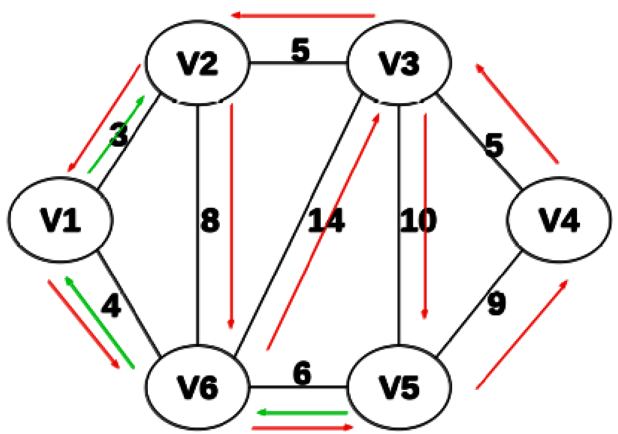
Para fechar a trilha (de v5 a v2), Dijkstra: v5 —> v6 —> v1 —> v2, custo =13

Total=13 + 64 = 77.

Este é o menor trajeto/custo repetindo o mínimo de arestas.

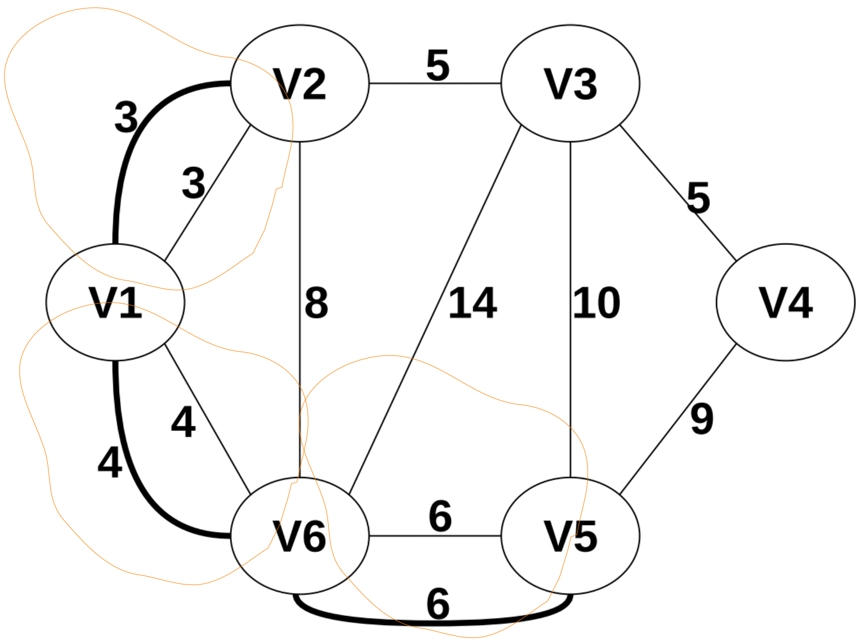
v2→ v1→ v6→ v5→ v4→ v3→ v2→ v6→ v3→ v5 —> v6 —> v1 —> v2

Uma solução “boa o suficientemente”!



Grafos Eulerianos

Ao conceber o caminho mais curto e a trilha semieuleriana, obtemos um grafo “Eulerizado” como se houvesse arestas múltiplas de mesmo peso, como na figura abaixo:



Grafos Eulerianos

Quais dos seguintes grafos são Eulerianos ou semieulerianos?
(i) Grafo completo K_5;
(ii) Grafo completo bipartido $K_{2,3}$;
(iii) Grafo cube;
(iv) Grafo octaedro;
(v) Grafo de Petersen.

Responda
(i) Para quais valores de n o K_n é Euleriano?
(ii) Quais são os grafos completos bipartidos que são Eulerianos?
(iii) Para quais valores de n o grafo roda W_n é Euleriano?
(iv) Para quais valores de k, o k-cubo (Q_k) é Euleriano?
(v) Discuta uma abordagem que determine uma solução razoável (minimize as possíveis repetições de arestas no trajeto do carteiro) sobre o problema do carteiro chinês no caso do grafo não ser nem Euleriano, nem semieuleriano.

TEG

Bibliografia

Básica

LUCCHESI, C. L. et alli. Aspectos Teóricos da Computação, Parte C: Teoria dos Grafos, projeto Euclides, 1979.

SANTOS, J. P. O. et alli. Introdução à Análise Combinatória. UNICAMP; 1995.

SZWARCFITER, J. L. Grafos e Algoritmos Computacionais. Campus, 1986.

GERSTING, Judith L. Fundamentos Matemáticos para a Ciência da Computação. Rio de Janeiro. 3a Ed. Editora.

Complementar:

1.) CORMEN, T. Introduction to Algorithms, third edition, MIT press, 2009

2.) ROSEN, K. Discrete Mathematics and its applications, seventh edition, McGraw Hill, 2011.

3.) WEST, Douglas, B. Introduction to Graph Theory, second edition, Pearson, 2001.

4.) BONDY, J.A., MURTY, U.S.R., Graph Theory with applications , Springer, 1984.

5.) SEDGEWICK, R. Algorithms in C - part 5 - Graph Algorithms, third edition, 2002, Addison-Wesley.

6.) GOLDBARG, M., GOLDBARG E., Grafos: Conceitos, algoritmos e aplicações. Editora Elsevier, 2012.

7.) BONDY, J.A., MURTY, U.S.R., Graph Theory with applications , Springer, 1984

8.) FEOFILOFF, P., KOHAYAKAWA, Y., WAKABAYASHI, Y., uma introdução sucinta à teoria dos grafos. 2011. (www.ime.usp.br/~pf/teoriadosgrafos)

9.) DIESTEL, R. Graph Theory, second edition, springer, 2000

10.) FURTADO, A. L. Teoria de grafos. Rio de janeiro. Editora LTC. 1973.

11.) WILSON, R.J. Introduction to Graph Theory. John Wiley & Sons Inc., 1985

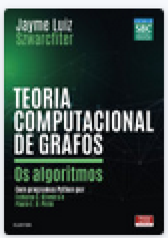
12.) BOAVENTURA NETTO , P. O. Grafos: Teoria, Modelos, Algoritmos. Edgard Blucher, SP, quinta edição

Tutoriais, artigos, notas de aula...

Vários livros podem ser acessados np formato eletrônico (e-book) via

https://www.udesc.br/bu/acervos/ebook

Exemplos:

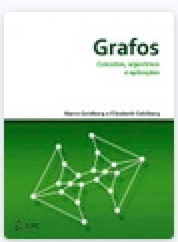


Teoria Computacional de Grafos - Os Algoritmos

Jayme Luiz Szwarcfiter

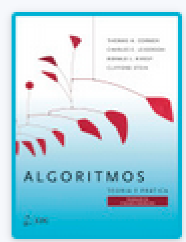
Fundamentos Matemáticos para a Ciência da Computação

Judith L. Gersting



Grafos

Marco Goldbarg



Algoritmos - Teoria e Prática

Thomas Cormen