MC558 — Análise de Algoritmos II

Cid C. de Souza   Cândida N. da Silva   Orlando Lee

27 de março de 2023

Antes de mais nada...

- Uma versão anterior deste conjunto de slides foi preparada por Cid Carvalho de Souza e Cândida Nunes da Silva para uma instância anterior desta disciplina.
- O que vocês tem em mãos é uma versão modificada preparada para atender a meus gostos.
- Nunca é demais enfatizar que o material é apenas um guia e não deve ser usado como única fonte de estudo. Para isso consultem a bibliografia (em especial o CLR ou CLRS).

Orlando Lee

Agradecimentos (Cid e Cândida)

- Várias pessoas contribuíram direta ou indiretamente com a preparação deste material.
- Algumas destas pessoas cederam gentilmente seus arquivos digitais enquanto outras cederam gentilmente o seu tempo fazendo correções e dando sugestões.
- Uma lista destes "colaboradores" (em ordem alfabética) é dada abaixo:
  - Célia Picinin de Mello
  - José Coelho de Pina
  - Orlando Lee
  - Paulo Feofiloff
  - Pedro Rezende
  - Ricardo Dahab
  - Zanoni Dias

Problema das Sete Pontes de Königsberg

O Problema das Sete Pontes de Königsberg é considerado historicamente o problema que originou a Teoria dos Grafos.

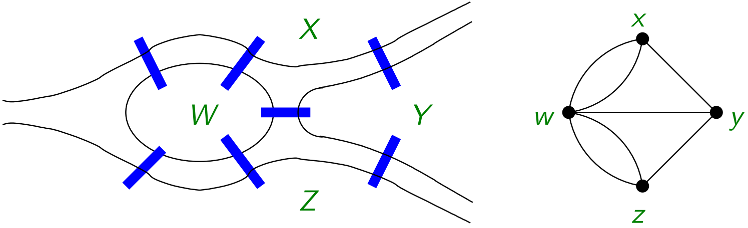


Figura: As sete pontes de Königsberg e um grafo modelo.

A cidade de Königsberg ficava localizada à beira do rio Pregel na Prússia (atualmente Kaliningrado, Rússia). A cidade ocupava uma ilha central e certas áreas à margem do rio. As regiões eram ligadas por sete pontes.

Problema das Sete Pontes de Königsberg

Segundo o folclore, os habitantes da cidade se perguntavam se seria possível sair de casa, cruzar cada ponte exatamente uma vez e voltar para casa.

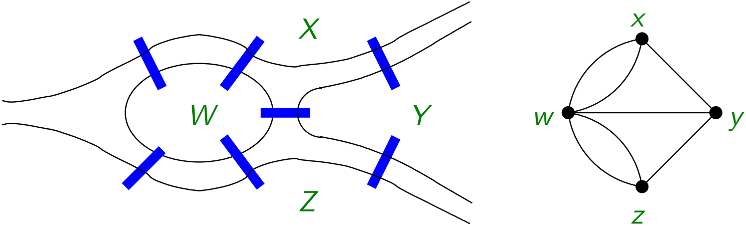


Figura: As sete pontes de Königsberg e um grafo modelo.

Podemos modelar cada região por um vértice e cada ponte por uma aresta ligando vértices correspondentes às regiões que a ponte liga obtendo o grafo da figura.

O problema então se reduz a determinar se o grafo possui uma trilha fechada passando exatamente uma vez por cada aresta.

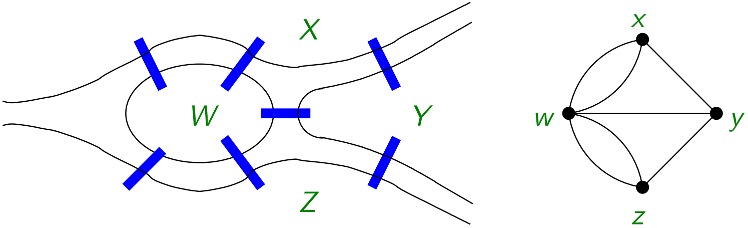


Figura: As sete pontes de Königsberg e um grafo modelo.

O Problema das Sete Pontes de Königsberg tem solução? Por quê?

Problema das Sete Pontes de Königsberg

- O grafo indicado não admite uma tal trilha pois tem um vértice x de grau ímpar. Se uma trilha existisse, ela deveria passar por x um número par de vezes (metade chegando em x e a outra metade saindo de x). Isto não é possível pois não podemos repetir arestas.
- Outra situação em que poderia não existir solução ocorre quando existem arestas em componentes distintos do grafo. Note que componentes triviais são irrelevantes para o problema.

Podemos pensar no mesmo problema para um grafo arbitrário G. Para que G admita uma trilha fechada que passa exatamente uma vez por cada aresta, as seguintes condições são necessárias:

1. todo vértice de G deve ter grau par e
2. G tem no máximo um componente não-trivial.

Em 1741, o famoso matemático suíco Leohnard Euler afirmou que tais condições também são suficientes. Em homenagem a sua contribuição (e devido a seu prestígio), seu nome ficou associado a tais grafos.

Somente em 1873, Hierholzer apresentou uma prova completa da suficiência.

Grafos Eulerianos

Um grafo G é Euleriano se admite uma trilha fechada T que passa uma vez em cada aresta. Dizemos que T é uma trilha Euleriana ou trilha de Euler.

Um grafo é par se todo vértice de G tem grau par.

Nosso objetivo é mostrar que um grafo G é Euleriano se, e somente se, G é par e tem no máximo um componente não-trivial.

(Ou seja, G é conexo *a menos dos vértices isolados.*)

Prova do Teorema de Euler

Apresentamos uma prova usando a ideia de escolher de trilha máxima (maximal aqui não funciona).

Lema. Toda trilha maximal em um grafo par G é fechada.

Prova. Seja T uma trilha maximal e seja u seu início. Todo vértice interno de T é incidente a um número par de arestas em $E(T)$. Se o término v de T não for igual a u, então v seria incidente a um número ímpar de arestas em $E(T)$. Como G é par, isto implica que T pode ser estendida, uma contradição com a escolha de T. Portanto, T é fechada. ∎

Teorema. Um grafo G é Euleriano se, e somente se,

1. G é par e
2. G tem no máximo um componente não trivial.

Prova.

($\Leftarrow$) Seja T uma trilha de **comprimento máximo** de G. Pelo Lema, T é fechada. Suponha que T não seja Euleriana, i.e., $E(G) - E(T) \neq \emptyset$.

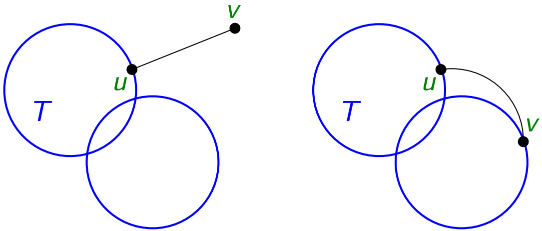
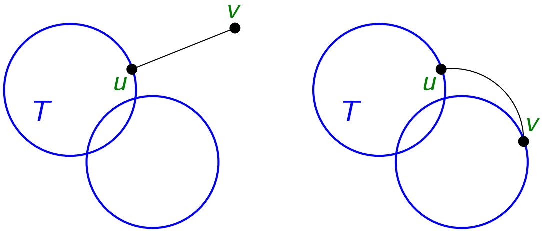


Figura: "Trilha máxima" T e possibilidades da aresta $e = uv$.

Como G tem no máximo um componente não trivial, existe uma aresta $e = uv \in E(G) - E(T)$ com pelo menos um de seus extremos, digamos u, em $V(T)$.



Seja $G' = G - E(T)$. Note que G' é par (Por quê?).

Seja T' uma trilha maximal em G' começando em u. Seja s o início de T.

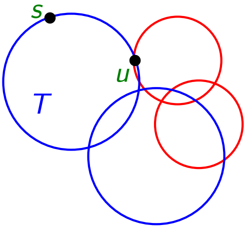


Figura: "Trilha máxima" T e trilha maximal T' de G' .

Então $T'' := sTu \bullet T' \bullet uTs$ é uma trilha fechada de comprimento maior que T, contradição. Portanto, T é Euleriana. ∎

(Esta prova foi adaptada da prova apresentada no livro do West.)

Um algoritmo linear para encontrar uma trilha Euleriana

Descreveremos um algoritmo linear para o seguinte problema.

Problema da trilha Euleriana

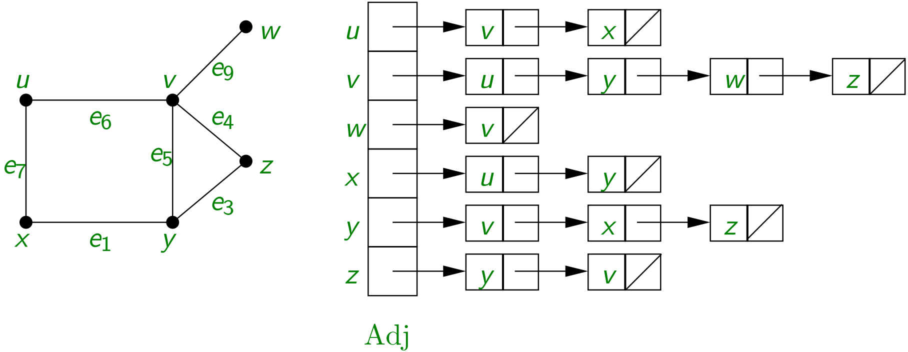
Entrada: um grafo G dado na forma de **listas de adjacências**.

Saída: uma trilha Euleriana de G, se existir, caso contrário, imprima **Não é Euleriano!**.

Por algoritmo linear queremos dizer de complexidade $O(V + E)$.

Listas de adjacências (usada em Algoritmos em Grafos)

Seja $G = (V, E)$ um grafo **simples**. A representação por listas de adjacências de G consiste em um vetor $\mathrm{Adj}[\ ]$ indexado por V tal que para cada $u \in V$, $\mathrm{Adj}[u]$ aponta para uma lista ligada contendo os vizinhos de u.



Espaço: $O(V + E)$.

Um algoritmo linear para encontrar uma trilha de Euler

Problema da trilha Euleriana

Entrada: um grafo G dado na forma de **listas de adjacências**.

Saída: uma trilha Euleriana de G, se existir, caso contrário, imprima **Não é Euleriano!**.

- Pelo Teorema de Euler, G é Euleriano se, e somente se, G é par e tem no máximo um componente não-trivial.
- É fácil verificar em tempo $O(V + E)$ se G tem algum vértice de grau ímpar. Se tiver, então imprima **Não é Euleriano!**.
- Veremos depois que é possível testar em tempo $O(V + E)$ se G tem no máximo um componente não-trivial. Se tiver mais de um, então imprima **Não é Euleriano!**.
- Assim, podemos supor que G é par e possui no máximo um componente não-trivial. Na verdade, podemos supor que G é conexo. (Por quê?)

Extensão da representação por listas de adjacências

- Um nó da lista $\mathrm{Adj}[u]$ contendo um vértice v corresponde a uma aresta uv em G.
- Suporemos que em cada nó temos um valor booleano que indica se a aresta uv foi visitada (usada) na trilha.
- Um detalhe importante é que se marcarmos uv como visitada no nó que contém v na lista $\mathrm{Adj}[u]$, é necessário marcar vu como visitada no nó que contém u na lista $\mathrm{Adj}[v]$.
- Poderíamos simplesmente percorrer a lista $\mathrm{Adj}[v]$, entretanto isto **não** resultaria em um algoritmo linear.
- Em vez disto, introduziremos uma redundância na representação para lidar com este detalhe (próximo slide).

Extensão da representação por listas de adjacências

- Em cada nó de uma lista $\mathrm{Adj}[u]$ contendo um vértice, digamos v, temos um apontador para o nó da lista $\mathrm{Adj}[v]$ que contém u.
- Desta forma, podemos marcar vu como visitada neste nó em tempo $O(1)$.
- Se na entrada fosse dada a lista das arestas de G, bastaria percorrer esta lista e construir as listas de adjacências com esses apontadores. Isto consome tempo $O(V + E)$.

Construindo uma trilha maximal

TRILHA_MAXIMAL(r, Adj)

1. $u \leftarrow r$, $T \leftarrow (u)$
2. **enquanto** $\mathrm{Adj}[u]$ não chegou ao fim **faça**
3. seja v o vértice atual em $\mathrm{Adj}[u]$ $\triangleright$ aresta uv **não** visitada
4. $T \leftarrow T \bullet (v)$
5. $u \leftarrow v$

- No pseudocódigo supomos que mantemos para cada vértice u um apontador para uma aresta **não** visitada da lista $\mathrm{Adj}[u]$ (ou NIL); se em algum momento, quando acessamos o apontador, este referencia uma aresta visitada, simplesmente avançamos na lista.
- Percorremos cada lista de adjacência **apenas uma vez**; assim gastamos tempo $O(1)$ para visitar cada aresta.
- Para simplificar a apresentação, omitimos esses detalhes no pseudocódigo.

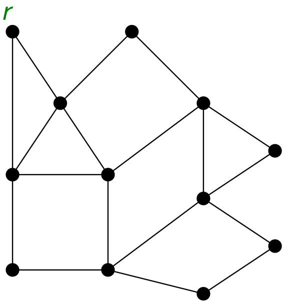
Construindo uma trilha maximal

$\textsc{Trilha\_Maximal}(r, \mathrm{Adj})$

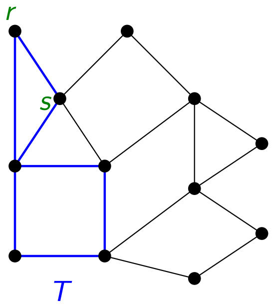
1. $u \leftarrow r$, $T \leftarrow (u)$
2. **enquanto** $\mathrm{Adj}[u]$ não chegou ao fim **faça**
3. seja v o vértice atual em $\mathrm{Adj}[u]$ ▷ aresta uv **não** visitada
4. $T \leftarrow T \bullet (v)$
5. $u \leftarrow v$

- O algoritmo só para quando $u = r$ e não há arestas incidentes a u que não foram visitadas. (Por quê?)
- $\textsc{Trilha\_Maximal}$ tem complexidade $O(E(T))$.
- A ideia do algoritmo é repetir este processo construindo trilhas maximais (disjuntas nas arestas) e combinar essas trilhas para obter uma trilha de Euler.

Construindo uma trilha de Euler



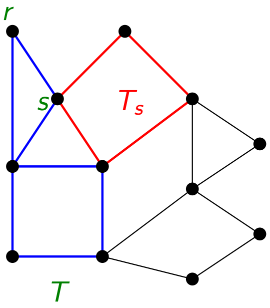
Escolha um vértice r qualquer.



Construa uma trilha maximal T a partir de r.

T não é Euleriana. Encontre o próximo vértice s em T que seja incidente a alguma aresta de $E(G) - E(T)$.

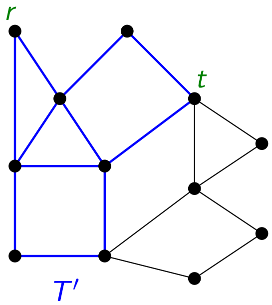
Construindo uma trilha de Euler



Construa uma trilha maximal T_s a partir de s que não usa arestas de T.

Seja $T' = rTs \bullet T_s \bullet sTr$.

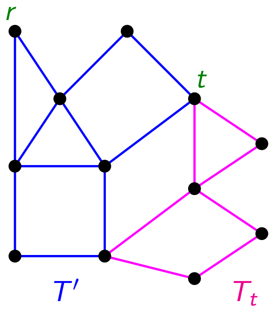
Construindo uma trilha de Euler



Seja $T' = rTs \bullet T_s \bullet sTr$.

T' não é Euleriana. Encontre o próximo vértice t em T' que seja incidente a alguma aresta de $E(G) - E(T')$.

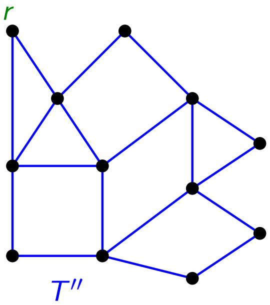
Construindo uma trilha de Euler



Construa uma trilha maximal T_t a partir de t que não usa arestas de T'.

Seja $T'' = rT't \bullet T_t \bullet tTr$.

Construindo uma trilha de Euler



Seja $T'' = rT't \bullet T_t \bullet tTr$.

T'' é Euleriana. Devolva T''.

Algoritmo

$\text{TRILHA\_DE\_EULER}(\text{Adj}[\,])$ $\triangleright$ G é par e conexo

1. Seja $r \in V(G)$
2. $T \leftarrow \text{TRILHA\_MAXIMAL}(r, \text{Adj})$
3. **enquanto** T **não** é Euleriana **faça**
4. encontre o próximo vértice s em T que é incidente a uma aresta em $E(G) - E(T)$
5. $T' \leftarrow \text{TRILHA\_MAXIMAL}(s, \text{Adj})$
6. $T \leftarrow rTs \bullet T' \bullet sWr$
7. **devolva** T

- A entrada é na forma de listas de adjacências (com os apontadores adicionais mencionados).
- Para cada lista $\text{Adj}[u]$, mantemos um apontador para a posição atual (aresta não visitada).
- Qual é a complexidade do algoritmo TRILHA_DE_EULER?

Análise de complexidade

$\text{TRILHA\_DE\_EULER}(\text{Adj}[\,])$ ▷ G é par e conexo

1. Seja $r \in V(G)$
2. $T \leftarrow \text{TRILHA\_MAXIMAL}(r, \text{Adj})$
3. **enquanto** T **não** é Euleriana **faça**
4. encontre o próximo vértice s em T que é incidente a uma aresta em $E(G) - E(T)$
5. $T' \leftarrow \text{TRILHA\_MAXIMAL}(s, \text{Adj})$
6. $T \leftarrow rTs \bullet T' \bullet sWr$
7. **devolva** T

Percorremos as listas de adjacências apenas uma vez. Lembre-se que ao escolher a aresta uv, o vértice u fica marcado como visitado em $\text{Adj}[v]$ (omitido no pseudocódigo). Lembre-se que TRILHA_MAXIMAL tem complexidade $O(E(T'))$ – linhas 2 e 5.

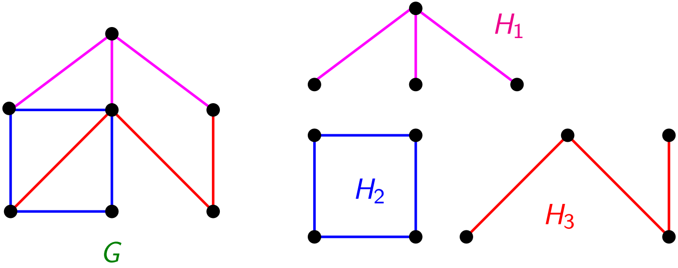
Podemos guardar o último vértice w usado para expandir a trilha.

Nenhum vértice entre r e w é incidente a arestas fora da trilha atual T.

Assim, para encontrar s basta começar a procurar a partir de w em T.

Decomposição de grafos

Considere o grafo G da figura. Observe que G é a união disjunta de um subgrafo H_1 isomorfo a $K_{1,3}$, de um subgrafo H_2 isomorfo a C_4 e de um subgrafo H_3 isomorfo a P_4.



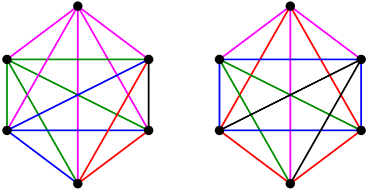
Dizemos que a coleção $\mathcal{F} := \{H_1, H_2, H_3\}$ é uma **decomposição** de G.

Uma decomposição de um grafo G é uma família $\mathcal{F}$ de subgrafos aresta-disjuntos de G tal que

$$\bigcup_{F \in \mathcal{F}} E(F) = E(G).$$

Se todo membro de $\mathcal{F}$ é isomorfo a algum grafo de uma famíla particular $\mathcal{H}$ de grafos, então dizemos que $\mathcal{F}$ é uma decomposição em grafos de $\mathcal{H}$. Por exemplo, se todo membro de $\mathcal{F}$ é um caminho (ciclo), então dizemos que $\mathcal{F}$ é uma decomposição em caminhos (ciclos) de G.

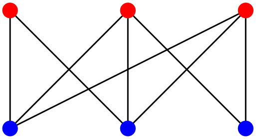
Exemplo 1. Um grafo completo com n vértices pode ser decomposto em $n-1$ grafos bipartidos $K_{1,i}$, $1 \leq i \leq n-1$. De fato, em geral pode haver várias maneiras de decompor um grafo completo em grafos bipartidos completos.

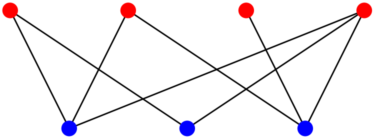


Graham e Pollak (1971) mostraram que qualquer decomposição de K_n em grafos bipartidos completos requer pelo menos $n-1$ grafos. O leitor interessado pode ver uma prova deste resultado no livro de BM06.

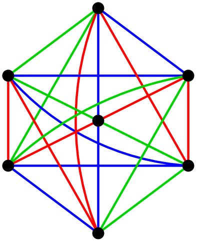
Decomposição de grafos

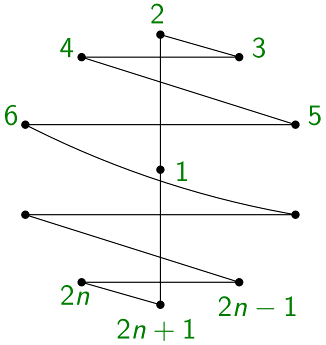
Exemplo 2. Um grafo bipartido pode ser decomposto em estrelas. Uma estrela é um grafo isomorfo a $K_{1,r}$ para algum $r > 0$.





Exemplo 3. Para $n \geq 1$, o grafo completo K_{2n+1} pode ser decomposto em n ciclos Hamiltonianos, i.e., um ciclo contendo todos os vértices. Veja a figura. Rotacionando a figura n vezes obtemos n ciclos Hamiltonianos disjuntos nas arestas de K_{2n+1}.





Decomposição em ciclos

Qual seria uma condição necessária óbvia para um grafo G admitir uma decomposição em ciclos?

Suponha que G admite uma decomposição em ciclos, digamos $\mathcal{C}$. Seja v um vértice de G. Cada ciclo de $\mathcal{C}$ que contém v contribui com dois para o grau de v. Como os ciclos são aresta-disjuntos e contém todas as arestas de G, segue que o grau de v é par.

Todo grafo que admite uma decomposição em ciclos tem que ser par.

Esta condição também é suficiente? SIM!

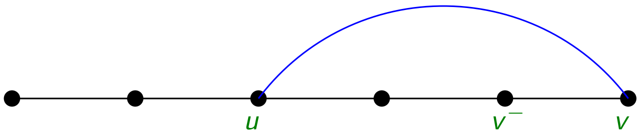
Grau mínimo e ciclos

Precisamos de um resultado auxiliar (já visto).

Lema. Se G é um grafo com $\delta(G) \geq 2$, então G contém um ciclo.

Prova. Se G contém um laço ou arestas múltiplas, então o resultado segue.

Assim, podemos supor que G é simples. Seja P um caminho maximal de G e seja v seu término. Seja v^- o vértice que precede v em P. Como $d(v) \geq 2$, v é adjacente a algum vértice u distinto de v^-. Como P é maximal, u tem que pertencer a P. Então $uPv \bullet (v, e, u)$ é um ciclo de G. ■

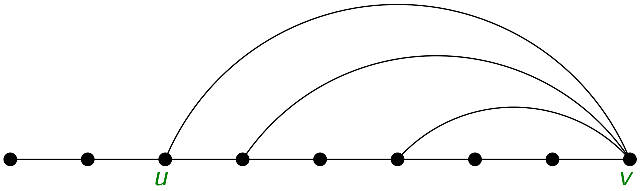


Grau mínimo e ciclos longos

Podemos refinar o teorema anterior quando G é simples.

Teorema. Se G é um grafo simples com $\delta(G) \geq 2$, então G contém um ciclo de comprimento pelo menos $\delta(G) + 1$.

Prova. Seja P um caminho maximal de G e seja v seu término. Pela maximalidade de P, todos os vizinhos de v tem que pertencer a P. Seja u o vizinho de v que está mais próximo em P do seu início. Assim, v e seus vizinhos estão em uPv. Isto implica que o comprimento de uPv é pelo menos $d(v) \geq \delta(G)$. Portanto, $uPv \bullet (v, u)$ é um ciclo de comprimento pelo menos $\delta(G) + 1$. ■



Teorema (Veblen, 1912). Um grafo G admite uma decomposição em ciclos se, e somente se, G é par.

Prova. ($\Rightarrow$) Já vimos esta implicação.

($\Leftarrow$) Seja G um grafo par. Provaremos por indução em $m := m(G)$ que G admite uma decomposição em ciclos.

Base: $m = 0$. O resultado é trivialmente verdade.

Hipótese de indução: suponha que para todo grafo par G' com $m(G') < m$ admite uma decomposição em ciclos.

Seja G um grafo par com m arestas. Podemos supor que G não tem vértices isolados. Logo, $\delta(G) \geq 2$. Pelo Lema, G contém um ciclo C.

Seja $G' := G - E(C)$. Por HI, G' admite uma decomposição em ciclos $\mathcal{C}'$. Logo, $\mathcal{C} := \mathcal{C}' \cup \{C\}$ é uma decomposição em ciclos de G. ∎

Decomposição em ciclos

Teorema (Veblen, 1912). Um grafo G admite uma decomposição em ciclos se, e somente se, G é par.

Usando o Teorema de Veblen, é fácil ver que um grafo par não tem aresta-de-corte, pois toda aresta pertence a algum ciclo.

Todo grafo par admite uma decomposição em ciclos. Uma pergunta natural é se podemos limitar o número de ciclos em uma decomposição.

Conjectura (Hajős, 1968). Todo grafo simples Euleriano admite uma decomposição em no máximo $\lfloor (n-1)/2 \rfloor$ ciclos.

Este é o melhor limitante possível em função de n. Considere o grafo completo K_{2n+1}. Seu número de arestas é $(2n+1)(2n)/2 = n(2n+1)$. Um ciclo Hamiltoniano tem comprimento $2n+1$ e K_{2n+1} pode ser decomposto em $n = \lfloor (2n+1-1)/2 \rfloor$ ciclos Hamiltonianos.

Problemas em aberto

Obviamente, todo grafo admite uma decomposição em caminhos. O problema é mais interessante e difícil se limitarmos o número de caminhos.

Conjectura (Gallai, 1966). Todo grafo simples admite uma decomposição em no máximo $\lfloor (n+1)/2 \rfloor$ caminhos.

Este é o melhor limitante possível em função de n. Considere o grafo completo K_{2n}. Seu número de arestas é $(2n)(2n-1)/2 = n(2n-1)$. Um caminho Hamiltoniano (i.e., que contém todos os vértices) tem comprimento $2n-1$ e K_{2n} pode ser decomposto em $n = \lfloor (2n+1)/2 \rfloor$ caminhos Hamiltonianos.

Exercício. Mostre que K_{2n} pode ser decomposto em n caminhos Hamiltonianos. **Dica:** adicione um vértice.

Referências

Veja Seção Eulerian Circuits do West96 (há outra prova da caracterização de grafos Eulerianos). Veja também a Seção 3.3 Euler Tours de BM08.