



SMAC03 – Grafos

1. Introdução a Grafos

Rafael Frinhani

frinhani@unifei.edu.br

2º semestre de 2025

Introduzir os conceitos básicos sobre grafos, apresentar sua história e possibilidades de aplicação na solução de problemas atuais, complexidade de algoritmos no contexto de grafos.

AGENDA

1. Introdução a Grafos

1.1 Contextualização

Contextualização, processo de modelagem, exemplos de modelos, definição de grafo.

1.2 História

Estudos de Euler, Kirchoff, Guthrie, Hamilton, Cayley, Borůvka.

1.3 Aplicações

Comunicação, Circuitos, Hidráulico, Financeiro, Transporte, Agendamento, Software, Internet, Redes Sociais, Comércio, Redes Complexas, Fenômenos de Espalhamento.

1.4 Revisão de Complexidade de Algoritmos

Complexidade de Espaço e de Tempo, conceitos de complexidade de algoritmos, Notação Big-O, classes de complexidade, complexidade nos problemas de otimização e de algoritmos em grafos.



1. Introdução

1.1. Contextualização

A complexidade do mundo real dificulta o processo de tomada de decisão, exigindo o uso de **modelos para abstrair os elementos essenciais** a serem considerados na solução de um problema.

A representação da realidade é uma necessidade da sociedade moderna, seja:

- pela impossibilidade de se lidar diretamente com a realidade;
- por aspectos econômicos;
- pela complexidade do problema;
- para possibilitar o trabalho computacional.

1.1. Contextualização

A complexidade do mundo real dificulta o processo de tomada de decisão, exigindo o uso de modelos para abstrair os elementos essenciais a serem considerados na solução de um problema.

A representação da realidade é uma necessidade da sociedade moderna, seja:

- pela impossibilidade de se lidar diretamente com a realidade;
- por aspectos econômicos;
- pela complexidade do problema;
- para possibilitar o trabalho computacional.

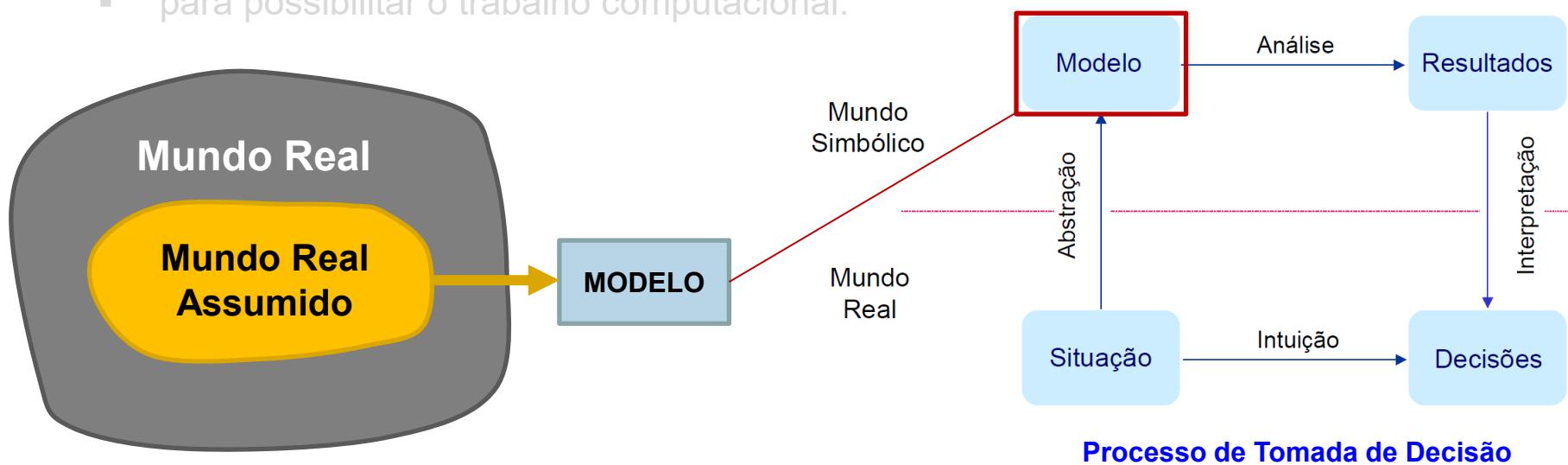


1.1. Contextualização

A complexidade do mundo real dificulta o processo de tomada de decisão, exigindo o uso de modelos para abstrair os elementos essenciais a serem considerados na solução de um problema.

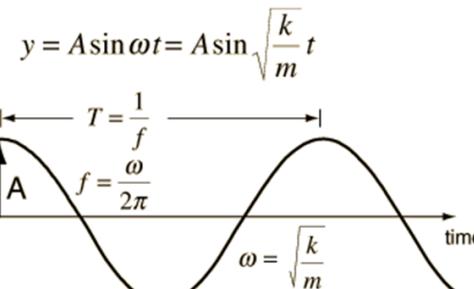
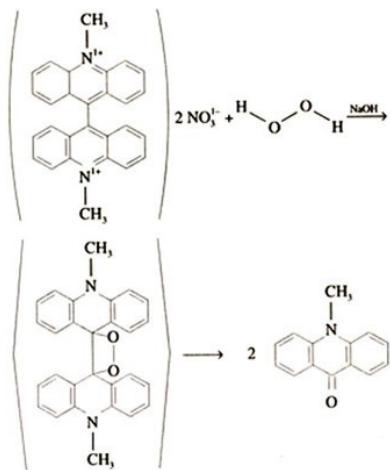
A representação da realidade é uma necessidade da sociedade moderna, seja:

- pela impossibilidade de se lidar diretamente com a realidade;
- por aspectos econômicos;
- pela complexidade do problema;
- para possibilitar o trabalho computacional.



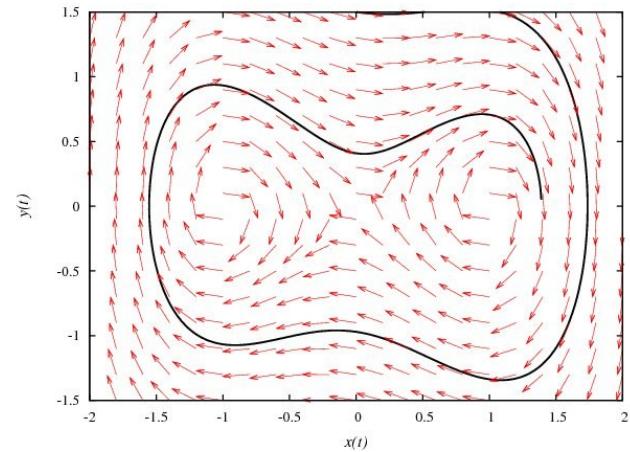
Exemplos de Modelos

Reação Química

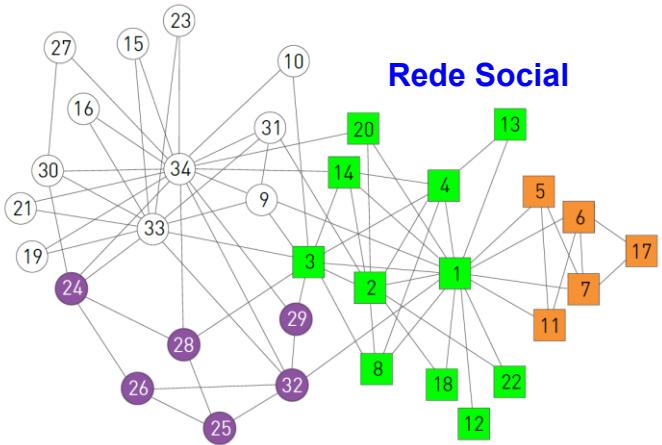


Movimento Oscilatório

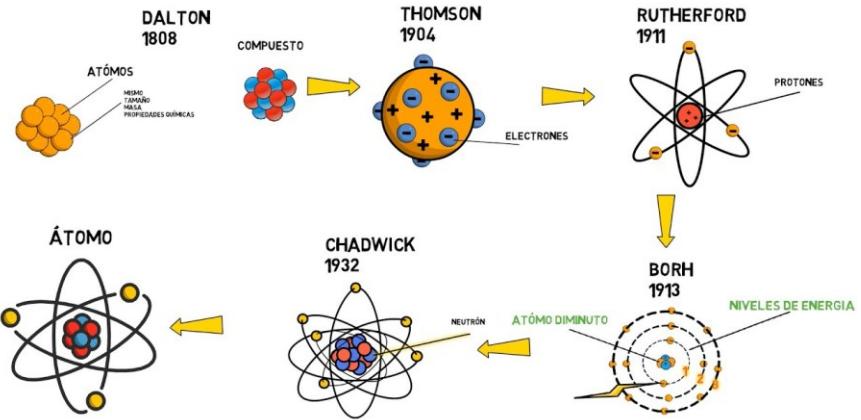
Movimento dos Flúidos



Rede Social



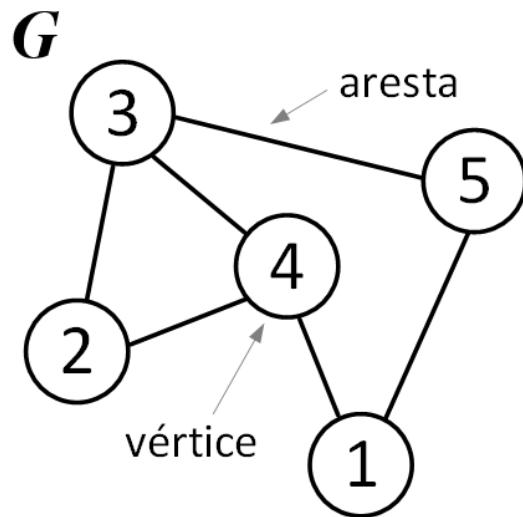
MODELOS ATÓMICOS



Grafo – Definição

É uma representação abstrata de um conjunto de objetos e das relações existentes entre eles.

Um grafo consiste de um conjunto finito de **vértices** (objetos) e um conjunto de **arestas** (relações). Arestas conectam pares de vértices.

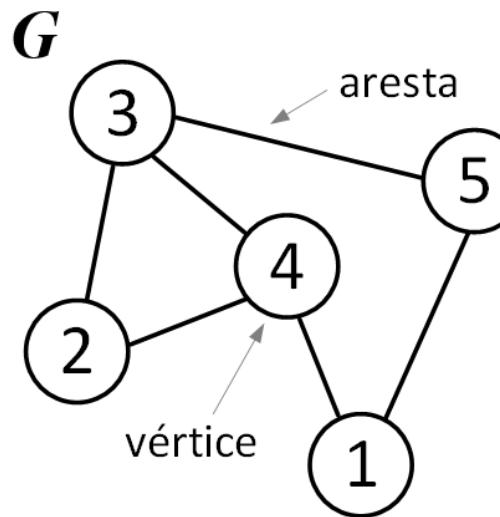


Grafo Simples

Grafo – Definição

É uma representação abstrata de um conjunto de objetos e das relações existentes entre eles.

Um grafo consiste de um conjunto finito de vértices (objetos) e um conjunto de arestas (relações). Arestas conectam pares de vértices.



Formalmente: Um grafo G é definido pelos conjuntos de vértices V e de arestas E .

$$G = (V, E)$$

Exemplo: O grafo ao lado possui 5 vértices e 6 arestas. Pode ser representado por:

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

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

Surgimento do grafo

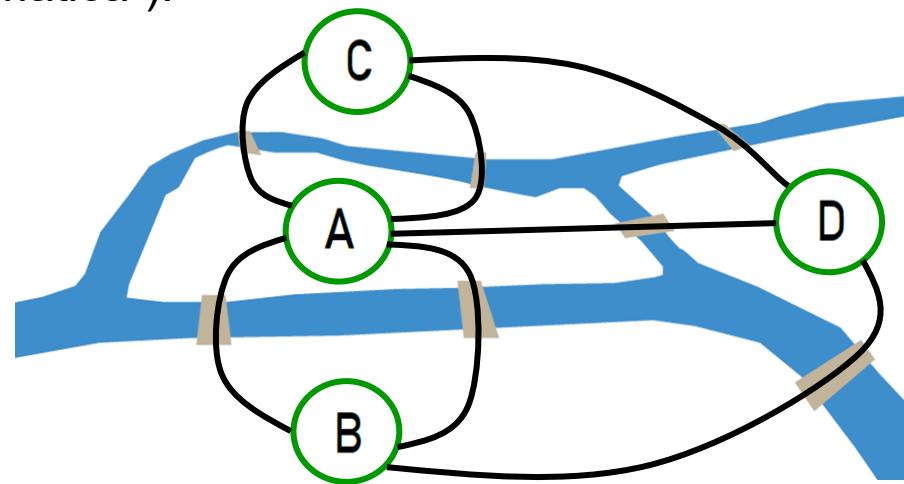


Leonhard Paul Euler
1707 - 1783

Euler provou matematicamente que este caminho não existe para as pontes Königsberg. Um **grafo Euleriano** é aquele que atendeu o requisito.

A origem dos estudos sobre grafos é creditada ao matemático suíço Leonhard Euler, que em 1735 abordou o problema das pontes de Königsberg: “É possível atravessar todas as 7 pontes sem cruzar uma ponte mais de uma vez?”

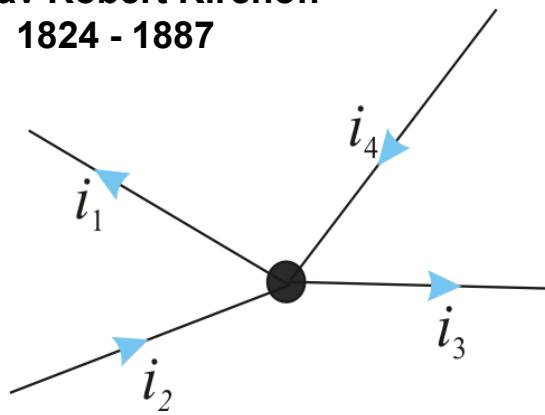
As partes de terra foram representadas por vértices (A, B, C, D) e as pontes pelas arestas, obtendo assim um grafo (inicialmente chamado “árvore matemática”).



Árvores Matemáticas

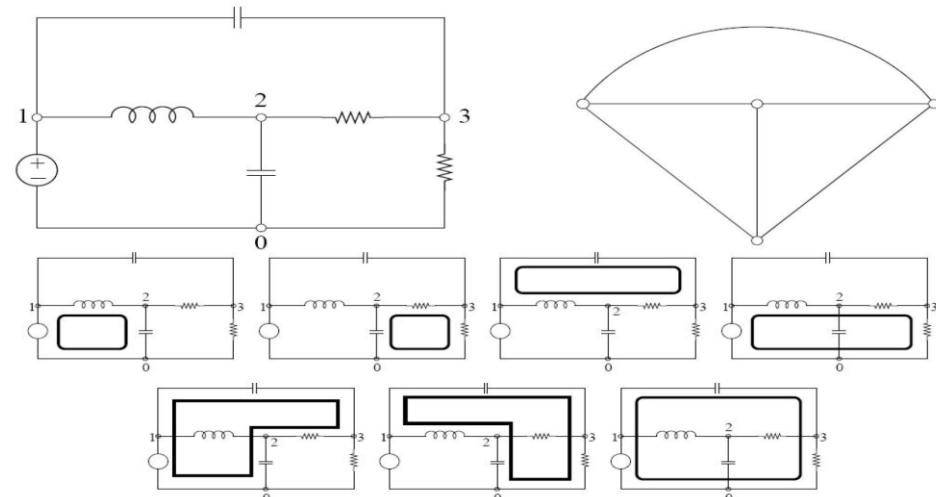


Gustav Robert Kirchoff
1824 - 1887



O físico alemão Gustav Kirchoff em 1847 analisou o comportamento das “árvore matemáticas” com a investigação de circuitos elétricos.

Provou o Teorema da Matriz-Árvore, que permite calcular a quantidade de árvores geradoras em grafos e multigrafos não-orientados.

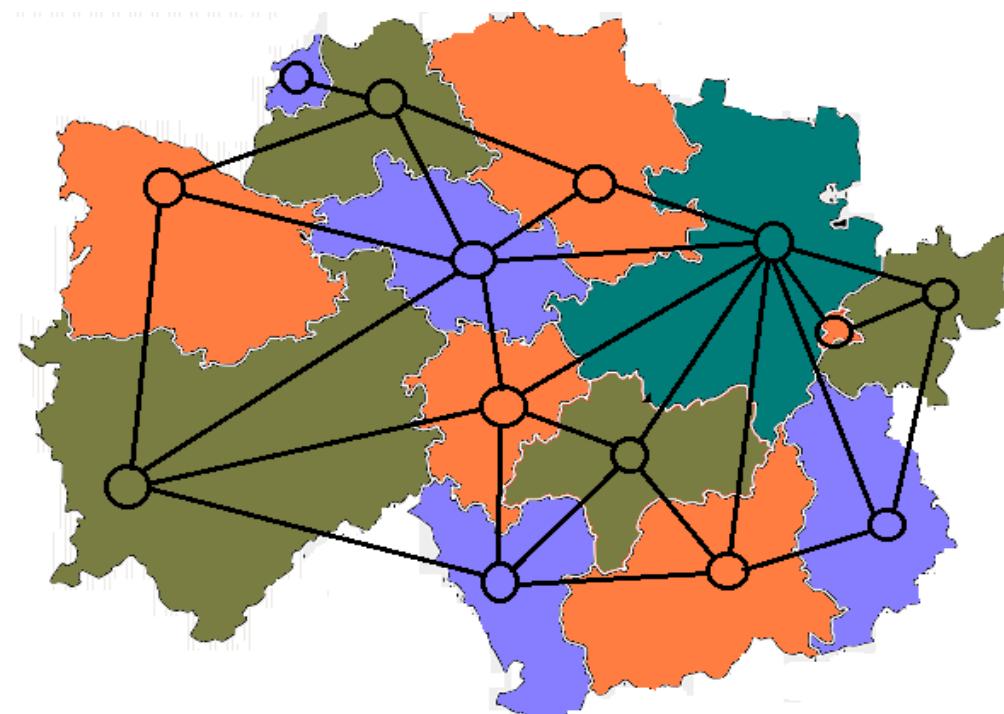


Coloração de Mapas

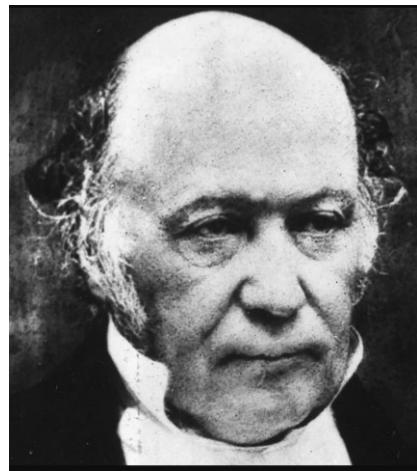


Francis Guthrie
1831 - 1899

O matemático e botânico sul-africano Francis Guthrie propôs em 1852 a solução para o problema das quatro cores: Qualquer mapa político pode ser colorido com no máximo quatro cores?



Rota Eficiente



William Rowan Hamilton
1805 - 1865

Hamilton foi um matemático, físico e astrónomo irlandês, que em 1859 inventou um jogo chamado “A Voyage Around the World”.

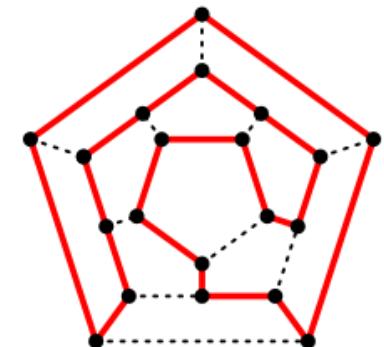
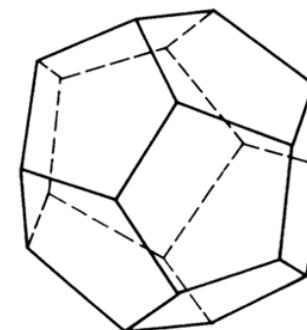
O jogo consistia de um dodecaedro no qual cada vértice representa uma cidade e as arestas as estradas. Objetivo: Existe um roteiro de viagem que passa por cada cidade exatamente uma vez?

Para um maior desafio, as cinco primeiras cidades eram determinadas aleatoriamente.

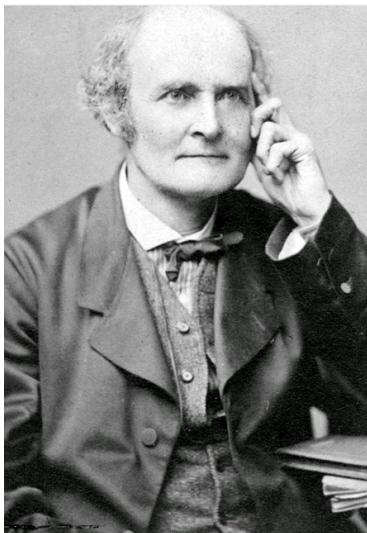
Um grafo que atenda esse requisito é chamado de **grafo hamiltoniano**.



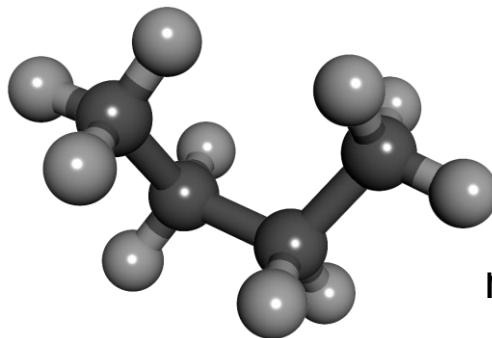
Dodecahedron



Estruturas Moleculares

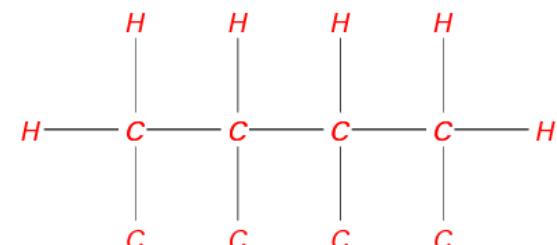


Arthur Cayley
1821 - 1895

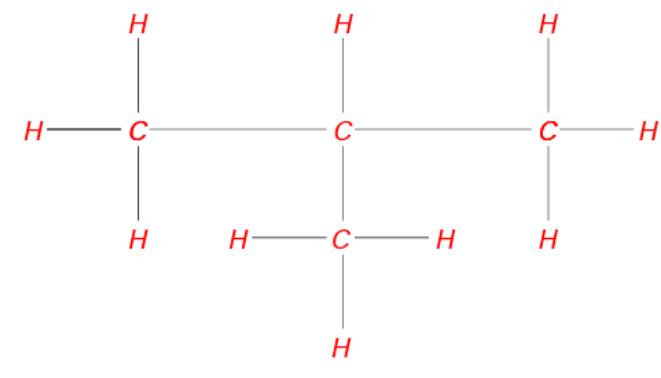


Modelo 3D de uma molécula de Isobutano

Arthur Cayley, matemático inglês que em 1857 usou “árvore matemática” na química orgânica para enumerar todos os isômeros para certos hidrocarbonetos.



Butano



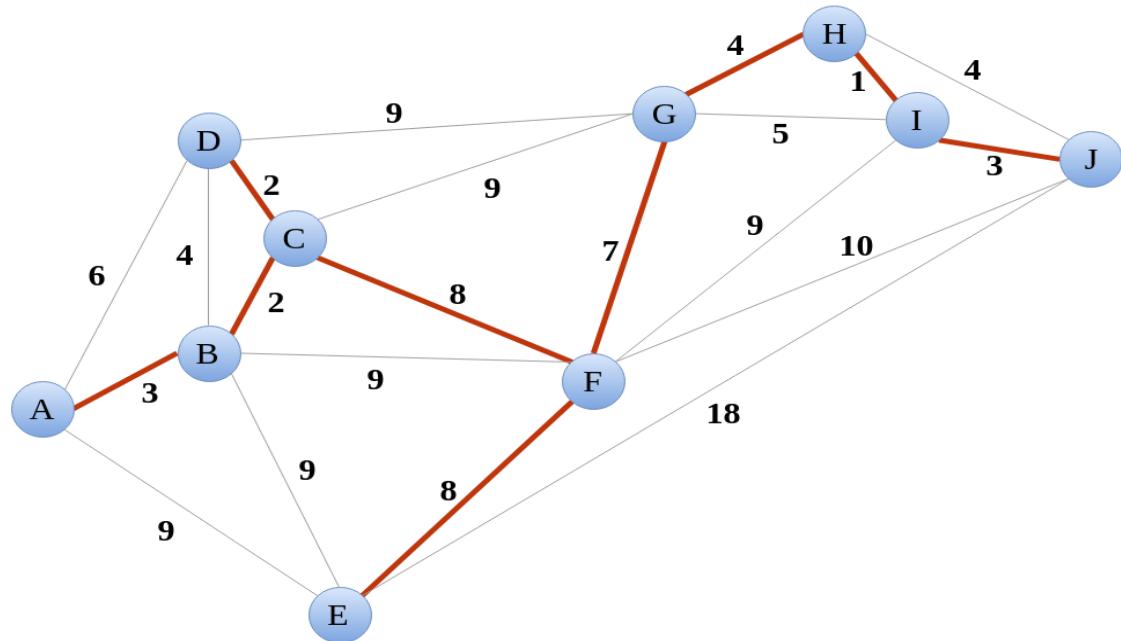
Isobutano

Pesquisa Operacional



Otakar Borůvka
1899 - 1995

Otakar Borůvka, matemático tcheco que solucionou em 1926 o problema de se projetar redes de distribuição elétrica eficientes através de árvore geradora mínima.



BORUVKA, O. *A Contribution to the Solution of a Problem on the Economical Construction of Power Networks*, Elecktronický obzor, 15, 1926. p.153-154.



1.3. Aplicações

Grafos podem ser usados em qualquer área nas quais existam relações entre os objetos do estudo.

O grafo possibilita a **representação do problema**. Um **algoritmo** em grafo é usado para encontrar a **solução para o problema**.

Exemplos de Aplicações e sua representação em Grafo

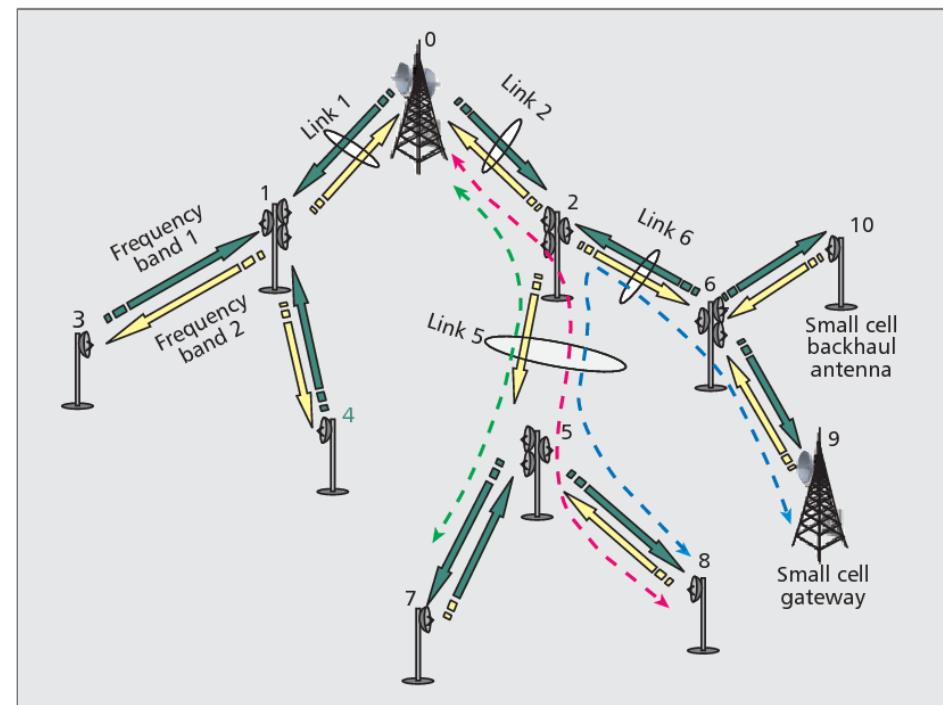
APLICAÇÃO	VÉRTICE	ARESTA
Comunicação	Centrais telefônicas, Redes de Computadores, Satélites	Cabos, Fibra Óptica, Enlaces
Circuitos	Portas Lógicas, registradores, processadores	Filamentos
Hidráulico	Reservatórios, estações de bombeamento	Rios, Tubulações
Financeiro	Produto Financeiro, Moeda	Transações
Transporte	Cidades, Aeroportos	Rodovias, Rotas Aereas
Escalonamento	Tarefas, Horários,	Restrições de Precedência
Software	Módulos	Interações
Internet	Páginas Web, Provedor	Links
Redes Sociais	Pessoas, Atores, Consumidores	Amizade, Filmes, Preferências
Comércio	Clientes, Fornecedores	Preferência de consumo

Redes de Comunicação

Inclui redes de computadores, telefonia, sensores e outras nas quais equipamentos interconectados enviam, encaminham e recebem mensagens de vários tipos. O interesse é entender a estrutura de conexão visando projetar redes mais eficientes.

No trabalho de Collings et al. (2015), os autores realizam a gestão online da rede definida por *software*.

Utilizam cliques para capturar o desequilíbrio de tráfego e planejar de maneira otimizada as frequências, infraestruturas e estrutura de rede a qualquer instante.

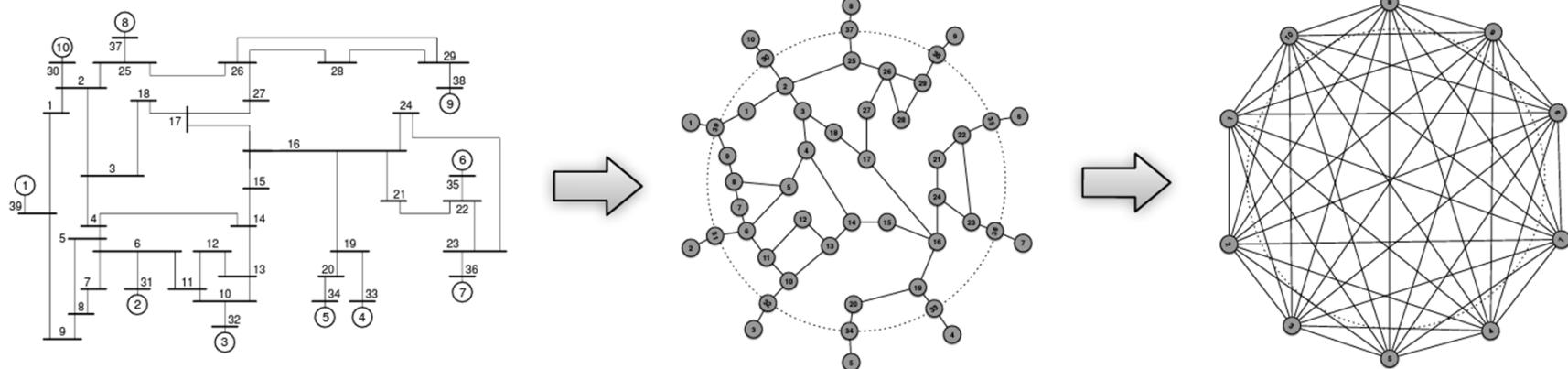


Ni, W., Collings, I.B., Lipman, J., Wang, X., Tao, M., & Abolhasan, M. (2015). *Graph theory and its applications to future network planning: software-defined online small cell management*. IEEE Wireless Communications, 22, 52-60.

Circuitos

Um circuito elétrico compreende dispositivos como transistores, resistores e capacitores que estão intrinsecamente ligados. Também inclui os sistemas de alta potência. O interesse está desde verificar possíveis problemas, até a otimização de circuitos.

Dörfler et al. (2018) fornecem uma visão geral das conexões da teoria dos grafos e do projeto e análise de circuitos elétricos, desde os circuitos integrados até grandes redes de distribuição.

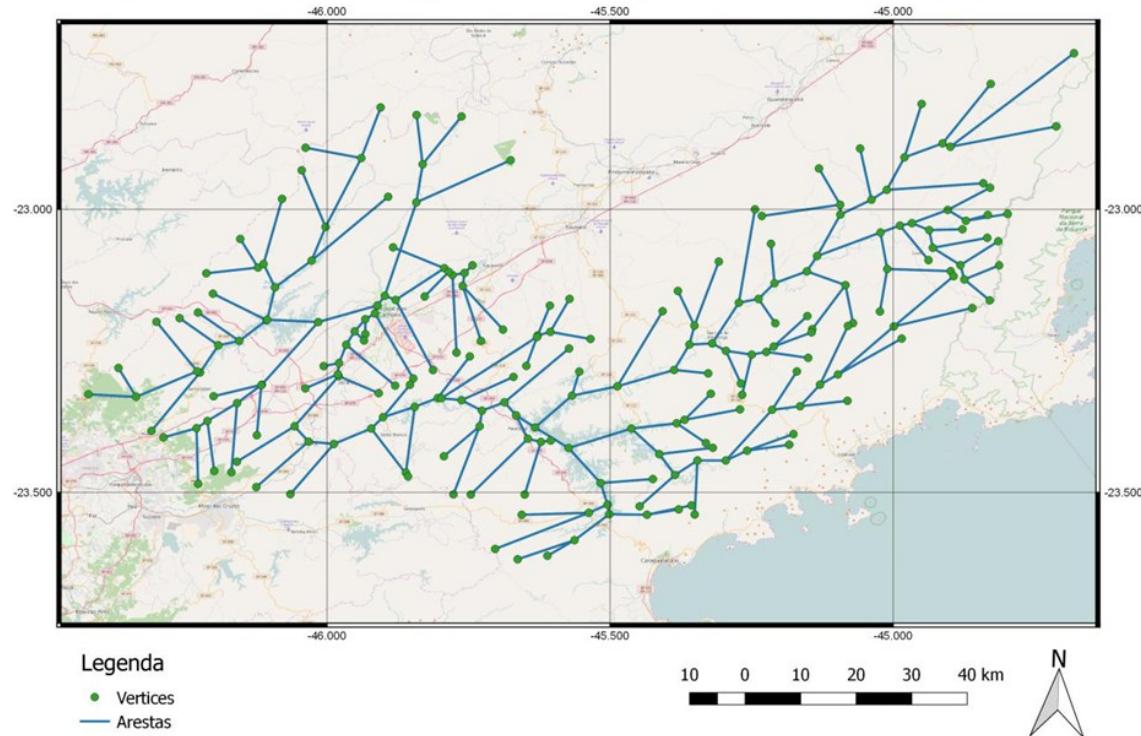


F. Dörfler, J. W. Simpson-Porco and F. Bullo, "Electrical Networks and Algebraic Graph Theory: Models, Properties, and Applications" in *Proceedings of the IEEE*, vol. 106, no. 5, pp. 977-1005, May 2018, DOI 10.1109/JPROC.2018.2821924.

Hidrografia

Grafos são usados nos estudos hidrográficos para análises de escoamento. Os vértices representam pontos de confluência, cabeceiras e foz dos rios, as arestas as redes de drenagem responsáveis pela ligação de um trecho do rio a outro.

Mapa da rede de drenagem do município São José dos Campos - SP



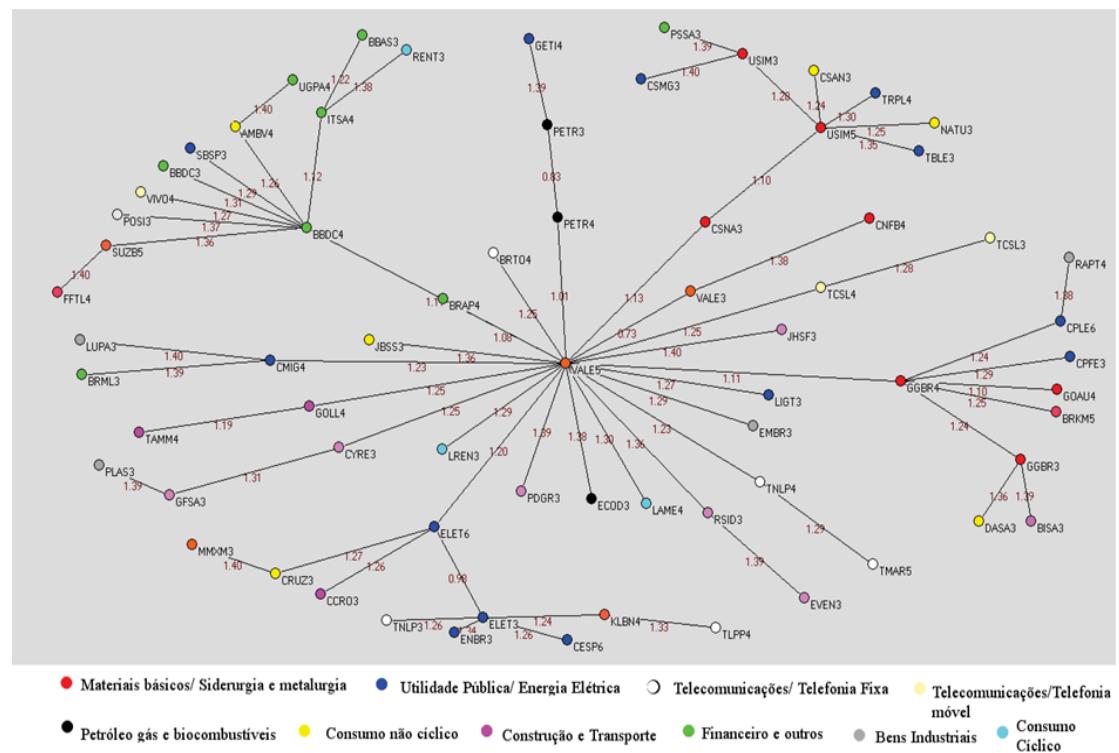
Oliveira, L. V.; de Carvalho, S.V.
& Santos, L.B.L. (2016)
"Desenvolvimento da biblioteca
HydroC - Estudos na
delimitação estocástica de
bacias hidrográficas. Relatório
Final de Projeto de Iniciação
Científica (PIBIC / CNPq / INPE)."

Financeiro

Grafos podem auxiliar no entendimento das correlações existentes em transações financeiras, negociações de commodities e produtos do mercado de ações.

Penalva (2011) reproduziu e sintetizou a estrutura de correlações no mercado financeiro dos produtos negociados na Bovespa & BMF.

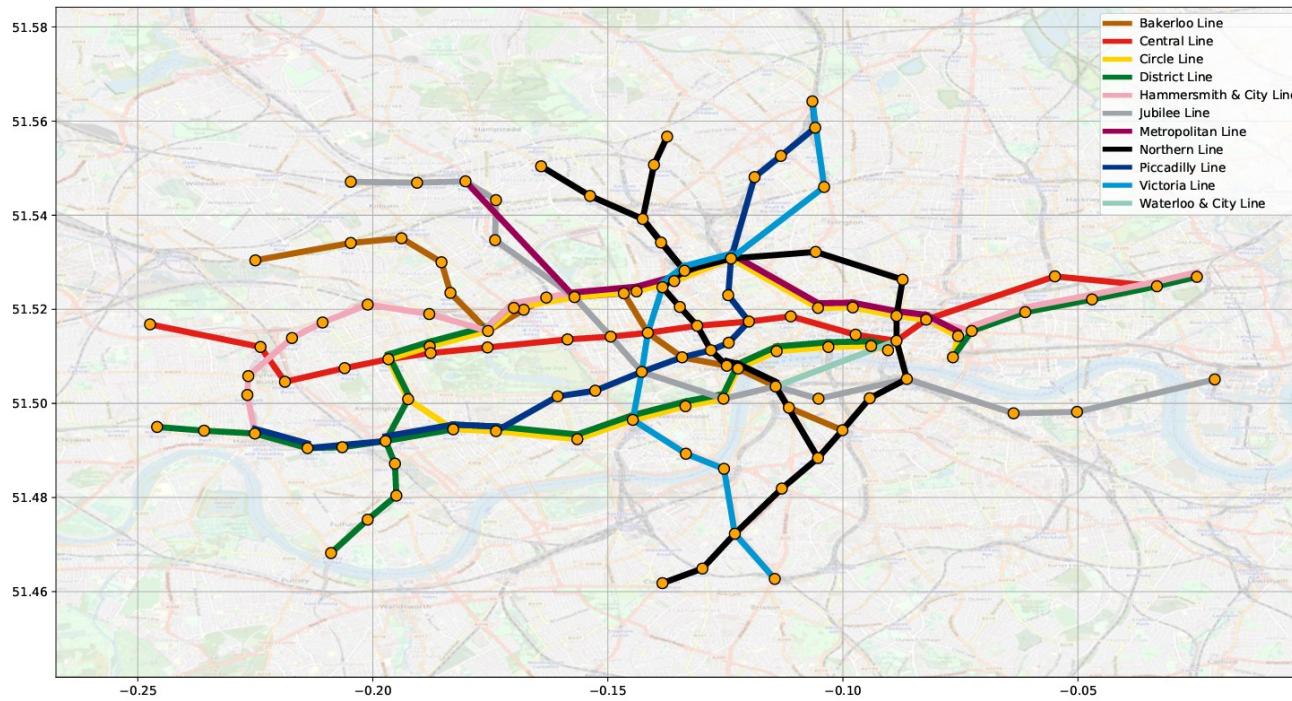
Além das commodities, considerou outros recursos naturais, preços das ações de petrolíferas e fatores como notícias e estoque de petróleo bruto.



Penalva, D. (2011) "Dinâmica de Correlações no Mercado Financeiro Bovespa & BMF". Dissertação de Mestrado. Instituto de Física Teórica. Universidade do Estadual Paulista.

Mapas e Transporte

Grafos são úteis para representação de mapas, auxiliando em sistemas de localização, no planejamento logístico de produtos e pessoas, bem como ocupação do espaço urbano e mobilidade.

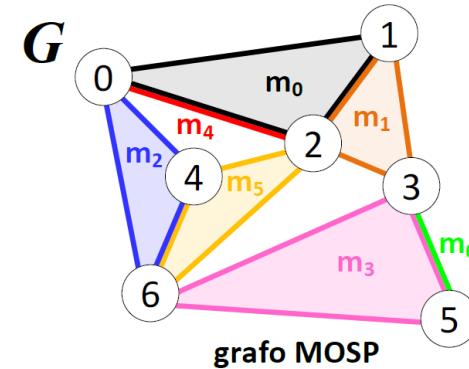
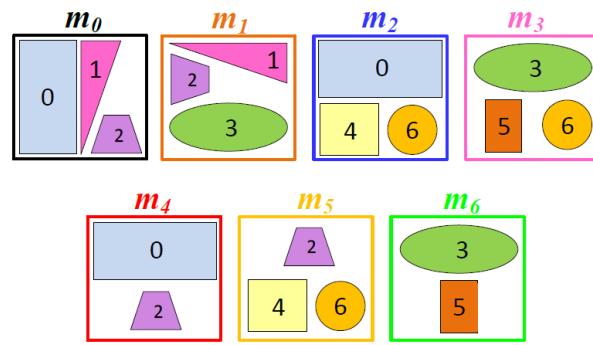


Dees, B. S., Xu, Y. L., Constantinides, A. G., & Mandic, D. P. (2021, July). Graph Theory for Metro Traffic Modelling. In *2021 International Joint Conference on Neural Networks (IJCNN)* (pp. 1-5). IEEE.

Agendamento e Produção

Um processo de fabricação requer a execução de uma variedade de trabalhos sob um conjunto de restrições que precedem a execução de certas tarefas. Grafos auxiliam na tomada de decisão em problemas de escalonamento de tarefas e otimização de processos produtivos.

Frinhani et al. (2018), a partir de uma modelagem em grafo desenvolveu a heurística *PieceRank* para solução do problema de minimização de pilhas abertas, observado em ambientes produtivos.



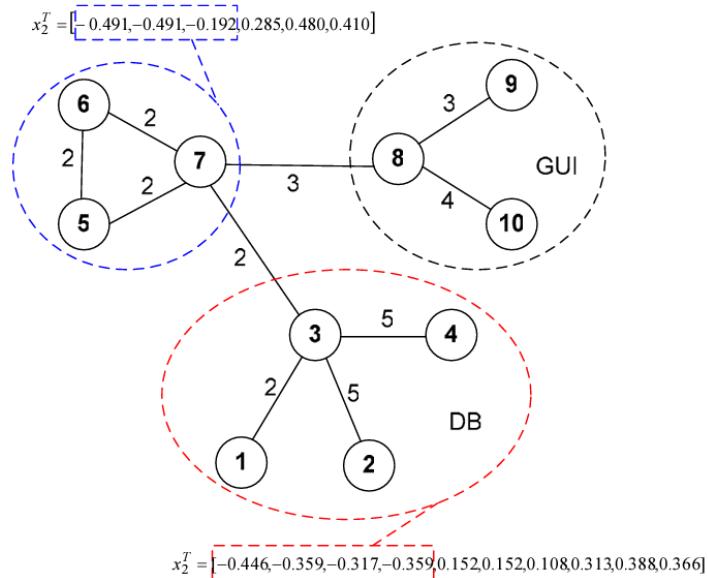
Frinhani RdMD, Carvalho MAMd, Soma NY (2018) A *PageRank*-based heuristic for the minimization of open stacks problem. PLoS ONE 13(8): e0203076. <https://doi.org/10.1371/journal.pone.0203076>

Softwares

Um compilador recorre a grafos para representar os relacionamentos entre módulos de um sistema. Os itens são as classes e as conexões à possibilidade de que um método em uma classe pode chamar outro. Bancos de dados orientados a grafos.

A análise por grafo permite determinar a melhor forma de alojar recursos para melhor eficiência do programa.

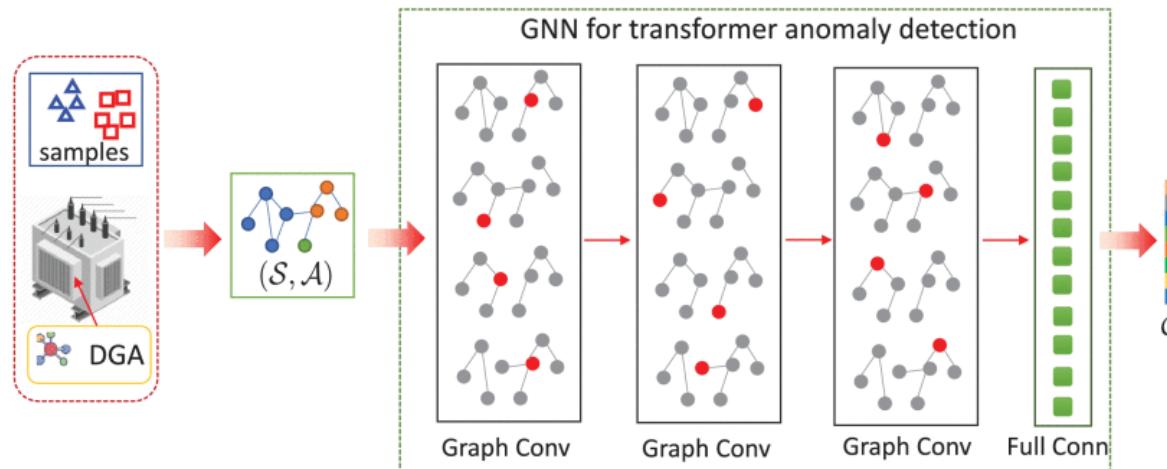
Chatzigeorgiou et al. (2006) utilizou agrupamento em grafos visando partitionar um sistema de *software* em subconjuntos de módulos ou componentes que compartilhem características comuns.



Chatzigeorgiou, A.; Tsantalis, N. & Stephanides, G. (2006). *Application of graph theory to OO software engineering*. In Proceedings of the 2006 International Workshop on Interdisciplinary Software Engineering Research (WISER '06). Association for Computing Machinery, New York, NY, USA, 29–36. <https://doi.org/10.1145/1137661.1137669>

Internet

A web inteira é um grafo, onde os itens são páginas e as conexões os links. Algoritmos em grafos são componentes essenciais dos motores de busca que auxiliam na localização de informações na web.



Wu & Tang (2022) estudaram Redes Neurais em Grafos (Graph Neural Network, GNN) para detecção de anomalias em transporte inteligente habilitado para *Industrial Internet of Things* (IIoT) e *smart factory* (fábrica inteligente).

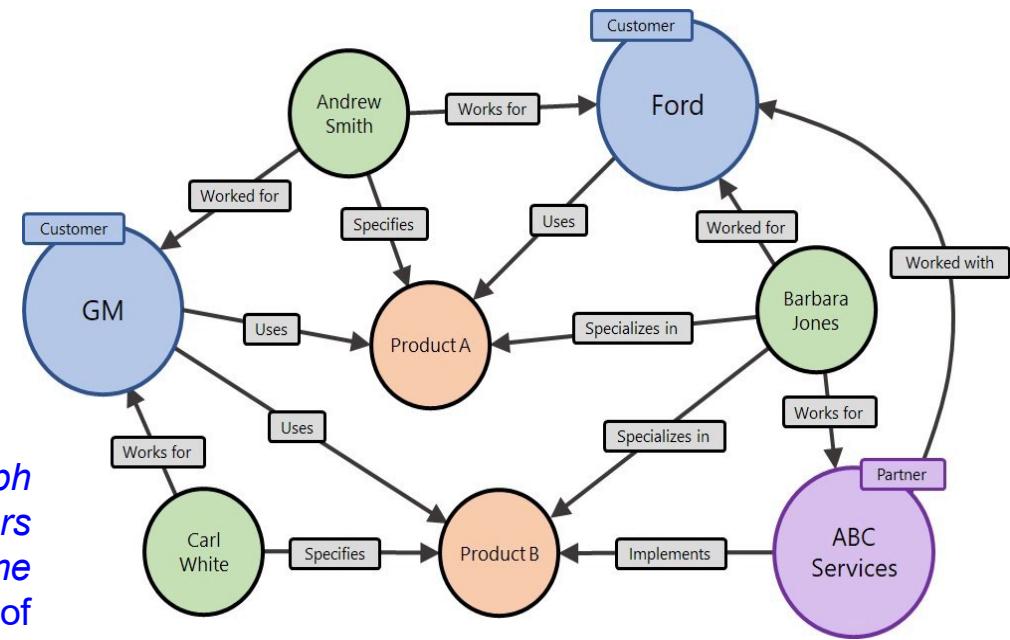
Wu, Y.; Dai, H.-N. & Tang, H. (2022) "Graph Neural Networks for Anomaly Detection in Industrial Internet of Things" in IEEE Internet of Things Journal, vol. 9, no. 12, pp. 9214-9231, DOI 10.1109/JIOT.2021.3094295.

Comércio

Varejistas e instituições financeiras rastreiam ordens de compra ou venda em um mercado. Uma conexão nesta situação representa a transferência de dinheiro e mercadorias entre uma instituição e um cliente. O conhecimento da natureza da estrutura de conexão neste caso pode melhorar a compreensão das preferências de compra de consumidores.

Teegan & Hartley (2015) propõem um método conceitual para calcular o valor financeiro do boca-a-boca compartilhado entre clientes pagantes e não pagantes em uma rede social.

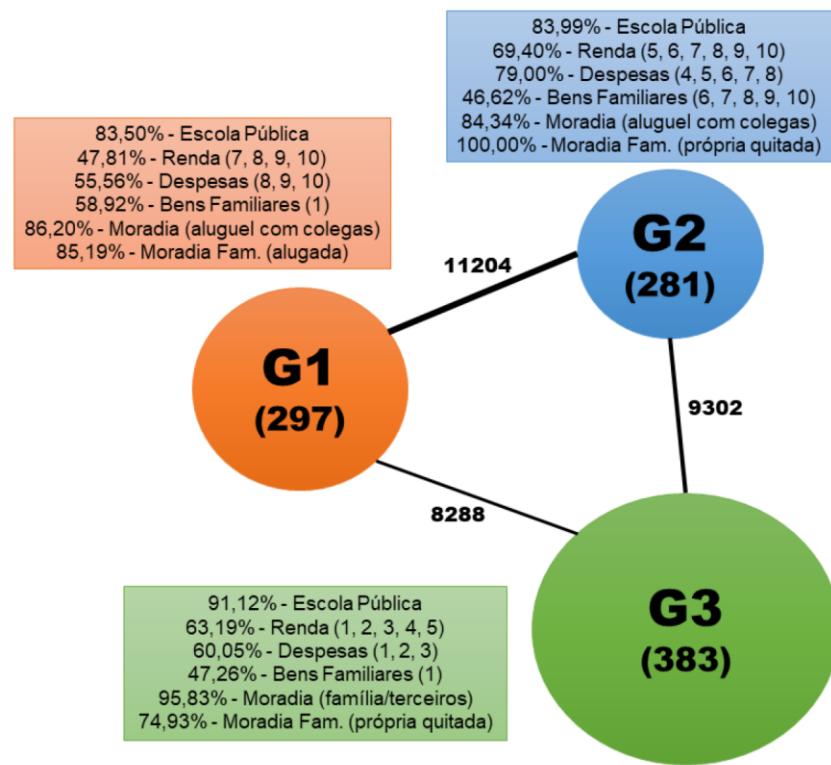
Teegan, C. C. G. & Hartley, N. (2015) *Using Graph Theory to Value Paying and Nonpaying Customers in a Social Network: Linking Customer Lifetime Value to Word-of-Mouth Social Value*, Journal of Relationship Marketing, 14:4, 301-320, DOI 10.1080/15332667.2015.1095008



Redes Sociais

Os objetos correspondem a pessoas e as conexões as amizades ou seguidores. Compreender as propriedades dessas redes requer alto processamento, sendo um tema de grande interesse não apenas para empresas, mas também na política, diplomacia, entretenimento, educação, marketing e outros domínios.

Porto & Frinhani (2021) utilizam grafos e métodos de detecção de comunidades em redes complexas para a identificação automática de grupos de solicitantes de auxílio estudantil com similaridades socio-econômicas.



Porto, Rodrigo & Frinhani, Rafael. (2021). Descoberta de Conhecimento no Apoio à Tomada de Decisão na Gestão Acadêmica em uma Instituição Federal de Ensino Superior. DOI 10.13140/RG.2.2.25696.76807.

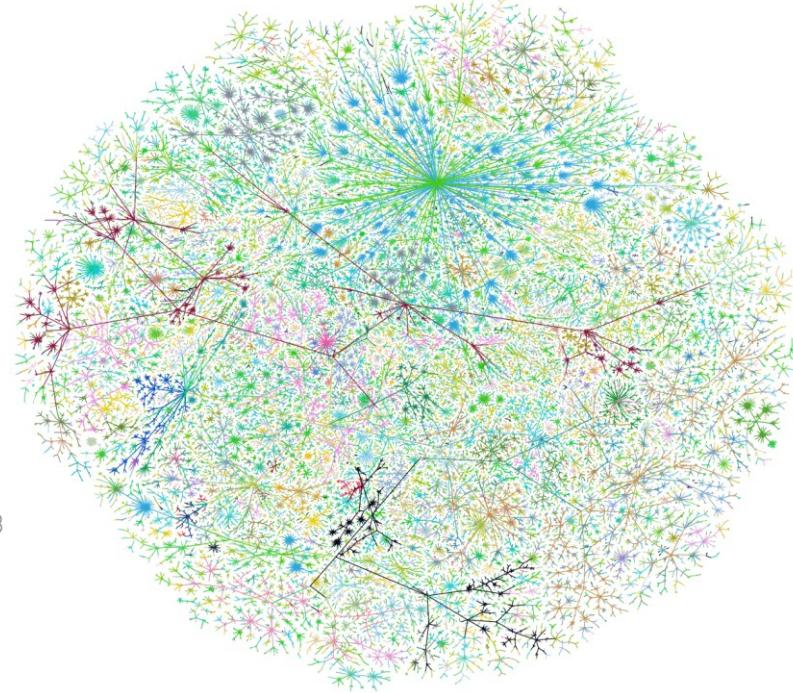
Ciência de Redes

Grafos são a base fundamental da Ciência de Redes, que é um campo acadêmico que estuda redes complexas (ex. telecomunicações, computadores, biológicas, redes sociais, redes cognitivas e semânticas). Entre outras áreas inclui estatística, física, mineração de dados e visualização da informação, modelagem inferencial.

Exemplos de Redes Complexas

N = Qtd. Nodes (vértices) / L = Qtd. Links (arestas)

NETWORK	NODES	LINKS	DIRECTED UNDIRECTED	N	L
Internet	Routers	Internet connections	Undirected	192,244	609,066
WWW	Webpages	Links	Directed	325,729	1,497,134
Power Grid	Power plants, transformers	Cables	Undirected	4,941	6,594
Mobile Phone Calls	Subscribers	Calls	Directed	36,595	91,826
Email	Email addresses	Emails	Directed	57,194	103,731
Science Collaboration	Scientists	Co-authorship	Undirected	23,133	93,439
Actor Network	Actors	Co-acting	Undirected	702,388	29,397,908
Citation Network	Paper	Citations	Directed	449,673	4,689,479
E. Coli Metabolism	Metabolites	Chemical reactions	Directed	1,039	5,802
Protein Interactions	Proteins	Binding interactions	Undirected	2,018	2,930



Barabási, A. L. "Network Science - Chapter 2 - Graph Theory". Disponível em: <http://networksciencebook.com/>, Acessado em: 22/08/2022.

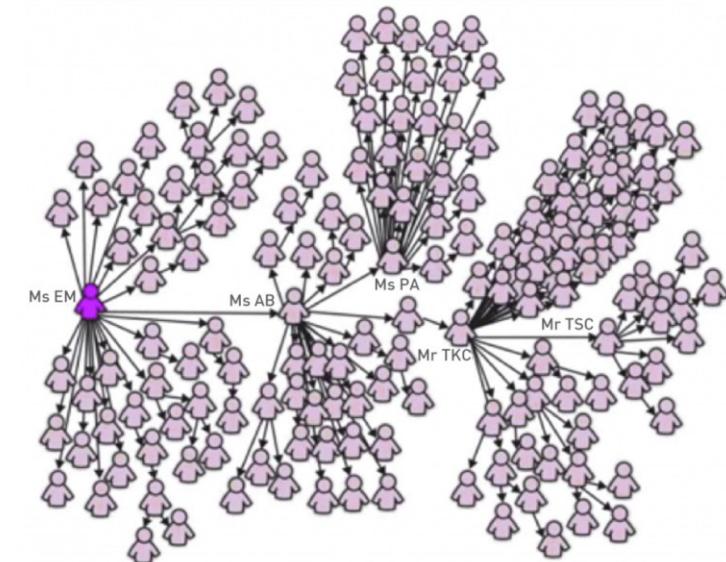
Fenômenos de Espalhamento

No estudo de redes complexas é possível identificar os *super-spreaders* (super-disseminadores), além de modelar fenômenos de espalhamento de vírus biológicos (epidemias) ou digitais, bem como o envio de mensagens em redes sociais.

Exemplos em Redes Complexas

PHENOMENA	AGENT	NETWORK
Venereal Disease	Pathogens	Sexual Network
Rumor Spreading	Information, Memes	Communication Network
Diffusion of Innovations	Ideas, Knowledge	Communication Network
Computer Viruses	Malwares, Digital viruses	Internet
Mobile Phone Virus	Mobile Viruses	Social Network/Proximity Network
Bedbugs	Parasitic Insects	Hotel - Traveler Network
Malaria	Plasmodium	Mosquito - Human network

Barabási, A. L. "Network Science - Chapter 10 – Spreading Phenomena". Disponível em: <http://networksciencebook.com/>, Acessado em: 22/08/2022.



A figura ilustra 144 dos 206 pacientes diagnosticados com SARS em Cingapura, cujos contágios foram atribuídos a cinco indivíduos *super-spreaders*. Foi identificado como Paciente Zero o médico da província de Guangdong, na China.

Contextualização

Para resolver um problema um algoritmo precisa de dois recursos importantes: **espaço** e **tempo**.

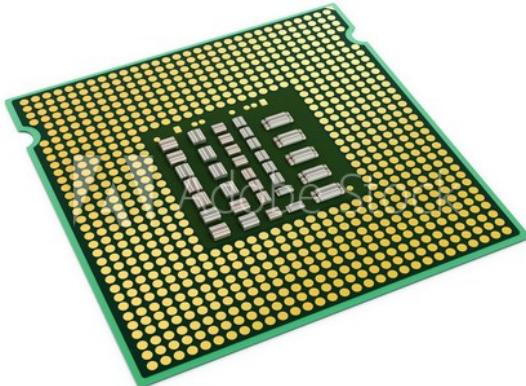
A **complexidade de espaço** relaciona o tamanho da entrada de dados com o espaço de armazenamento requerido (ex. memória).



Contextualização

Para resolver um problema um algoritmo precisa de dois recursos importantes: **espaço** e **tempo**.

A **complexidade de espaço** relaciona o tamanho da entrada de dados com o espaço de armazenamento requerido (ex. memória).



A **complexidade de tempo** de um algoritmo é o número de etapas que ele executa para resolver um problema de tamanho n .

Contextualização

A análise de complexidade é uma medida imparcial, no sentido que analisa a eficiência do algoritmo com base nos passos que ele executa para solucionar o problema (não depende de máquina).

Contextualização

A análise de complexidade é uma medida imparcial, no sentido que analisa a eficiência do algoritmo com base nos passos que ele executa para solucionar o problema (não depende de máquina).

A **análise do tempo de execução** é influenciada pelo desempenho do **processador**. A comparação de métodos pelo tempo de execução é robusta quando estes forem executados no mesmo equipamento.

Contextualização

A análise de complexidade é uma medida imparcial, no sentido que analisa a eficiência do algoritmo com base nos passos que ele executa para solucionar o problema (não depende de máquina).

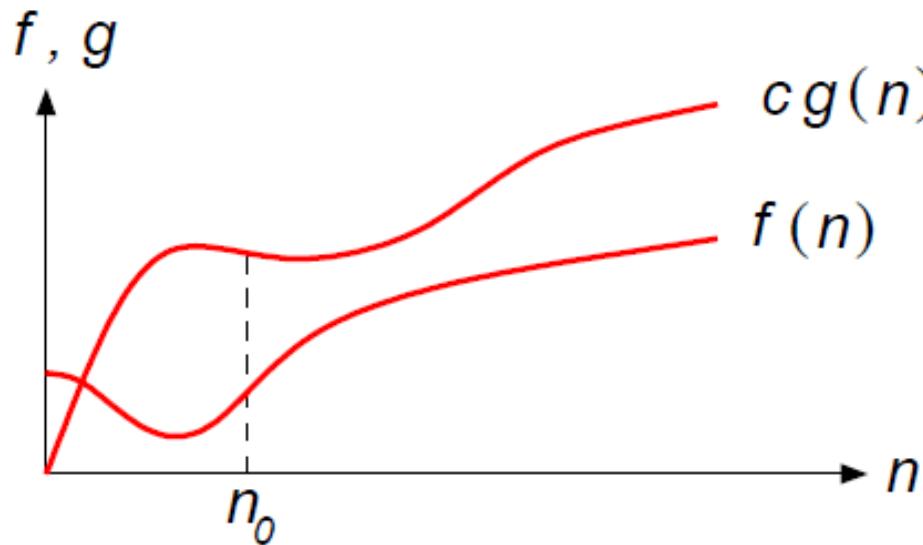
A análise do tempo de execução é influenciada pelo desempenho do processador. A comparação de métodos pelo tempo de execução é robusta quando estes forem executados no mesmo equipamento.

O objetivo na determinação da complexidade computacional de um algoritmo é obter um **limite assintótico** na contagem de passos (quando a entrada n assume valores grandes).

Notação Big-O

A notação O é usada para dar um **limite superior** sobre uma função de custo (ou função de complexidade) dentro de um fator constante.

Quando a notação O é usada para expressar o tempo de execução de um algoritmo no pior caso, define-se também o **limite (superior) do tempo de execução** desse algoritmo para todas as entradas.



Exemplo

O algoritmo de ordenação por inserção é $O(n^2)$ no pior caso (vetor inversamente ordenado).

Classes de Complexidade

Constante: $f(n) = O(1)$

- O uso do algoritmo independe do tamanho de n
- Instruções são executadas um número fixo de vezes (ex. acesso em tabela *Hash*)

Logarítmica: $f(n) = O(\log n)$

- Ocorre tipicamente em algoritmos que resolvem um problema transformando-o em problemas menores (ex. Pesquisa Binária)
- O tempo de execução pode ser considerado como sendo menor que uma constante grande (ex. Para $n = 1000$, $\log_2 \approx 10$)

Classes de Complexidade

Constante: $f(n) = O(1)$

- O uso do algoritmo independe do tamanho de n
- Instruções são executadas um número fixo de vezes (ex. acesso em tabela *Hash*)

Logarítmica: $f(n) = O(\log n)$

- Ocorre tipicamente em algoritmos que resolvem um problema transformando-o em problemas menores (ex. Pesquisa Binária)
- O tempo de execução pode ser considerado como sendo menor que uma constante grande (ex. Para $n = 1000$, $\log_2 \approx 10$)

Linear: $f(n) = O(n)$

- Em geral, um pequeno trabalho é realizado em cada elemento
- Quando n dobra de tamanho, o tempo de execução também dobra;

Linear Logarítmica: $f(n) = O(n \log n)$

- Algoritmos que usam decomposição, resolvem cada subproblema de forma independente e depois agrupam as soluções (paradigma Divisão e Conquista)
- Exemplo, Para $n = 1.000.000$, $\log_2 \approx 20.000.000$ (MergeSort)

Classes de Complexidade (cont.)

Quadrática: $f(n) = O(n^2)$

- Quando os itens de dados são processados aos pares, loop de iteração aninhado
- Para $n = 1.000$, o número de operações é da ordem de 1.000.000 (Ex. Algoritmos de ordenação simples como Seleção e Inserção)

Cúbica: $f(n) = O(n^3)$

- Algoritmos úteis apenas para solucionar problemas pequenos
- Exemplo, Para $n = 100$, operações 1.000.000 (ex. multiplicação de matrizes)

Classes de Complexidade (cont.)

Quadrática: $f(n) = O(n^2)$

- Quando os itens de dados são processados aos pares, loop de iteração aninhado
- Para $n = 1.000$, o número de operações é da ordem de 1.000.000 (Ex. Algoritmos de ordenação simples como Seleção e Inserção)

Cúbica: $f(n) = O(n^3)$

- Algoritmos úteis apenas para solucionar problemas pequenos
- Exemplo, Para $n = 100$, operações 1.000.000 (ex. multiplicação de matrizes)

Exponencial: $f(n) = O(2^n)$

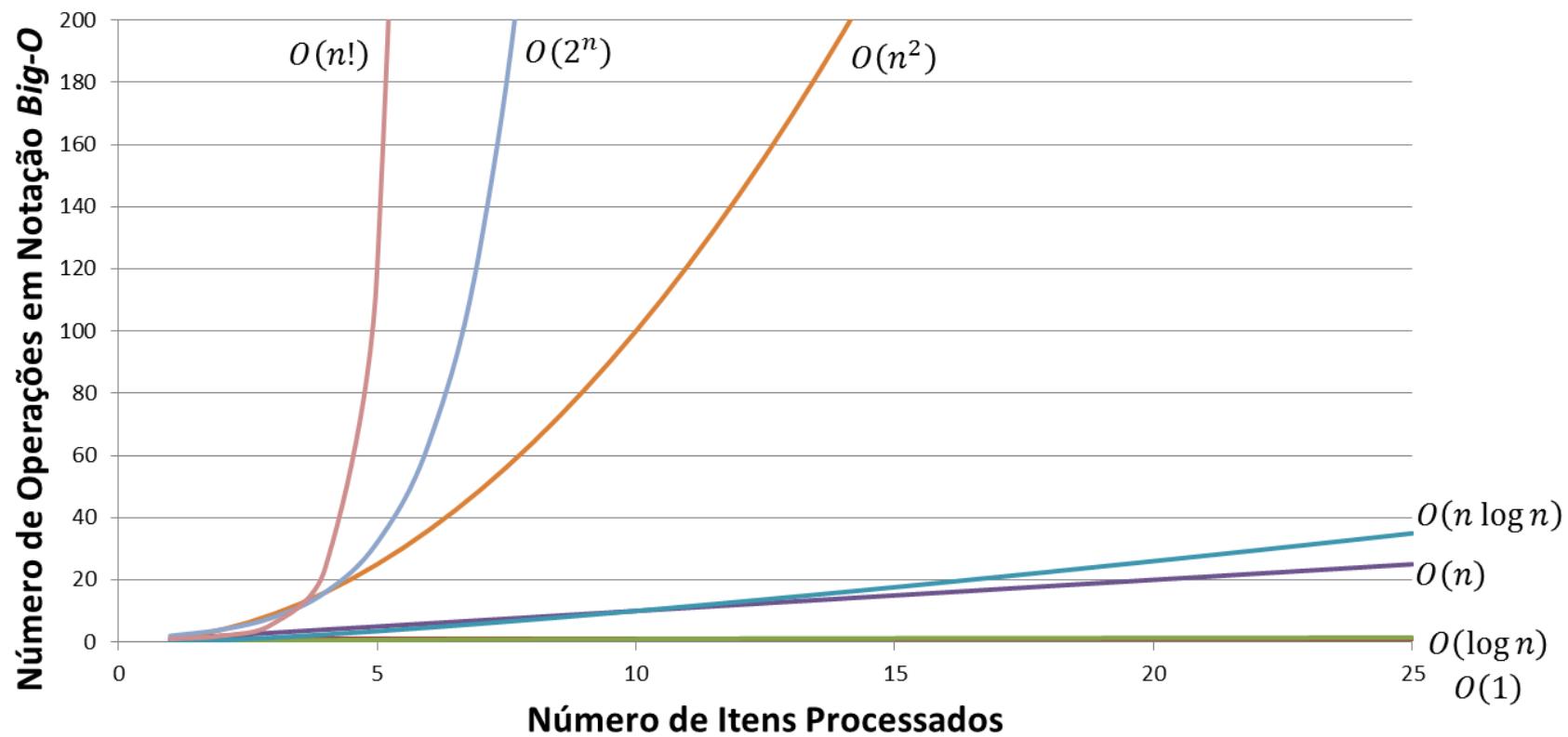
- Algoritmos dessa classe não são úteis sob o ponto de vista prático
- Solução de problemas quando se usa a força bruta
- Para $n = 20$, o número de operações é 1.000.000

Fatorial: $f(n) = O(n!)$

- Solução de problemas quando se usa força bruta
- Para $n = 20$, tem-se $20! = 2.432.902.008.176.640.000$ (ex. Caixeiro Viajante)

Classes de Complexidade (cont.)

Ilustração das Complexidades Mais Comuns - Notação *Big-O*



Taxas de Crescimento

$$1 < \log n < n, n^2, n^3 \dots n^k < 2^n, n!$$

constante
logarítmica
polinomial
exponencial

Tempos de execução de uma máquina que executa 10^9 passos por segundo (~1 Ghz), como uma função do custo do algoritmo e o tamanho da entrada n :

Size	$\log_2 n$	n	$n \log_2 n$	n^2	n^3	2^n
10	3.322 ns	10 ns	33 ns	100 ns	1 μ s	1 μ s
20	4.322 ns	20 ns	86 ns	400 ns	8 μ s	1 ms
30	4.907 ns	30 ns	147 ns	900 ns	27 μ s	1 s
40	5.322 ns	40 ns	213 ns	2 μ s	64 μ s	18.3 min
50	5.644 ns	50 ns	282 ns	3 μ s	125 μ s	13 days
100	6.644 ns	100 ns	664 ns	10 μ s	1 ms	$40 \cdot 10^{12}$ years
1000	10 ns	1 μ s	10 μ s	1 ms	1 s	
10000	13 ns	10 μ s	133 μ s	100 ms	16.7 min	
100000	17 ns	100 μ s	2 ms	10 s	11.6 days	
1000000	20 ns	1 ms	20 ms	16.7 min	31.7 years	

Solução de Problemas de Otimização

Uma das maneiras de se solucionar problemas com a garantia de encontrar a solução ótima seria gerar todas as $n!$ permutações dos n elementos da entrada (força bruta, busca exaustiva, enumeração explícita).

Algoritmos para gerar permutações:

- Steinhaus-Johnson-Trotter
- Shimon Evens
- Plain Change Transition
- Heap
- Permute by Sorting
- Random in Place

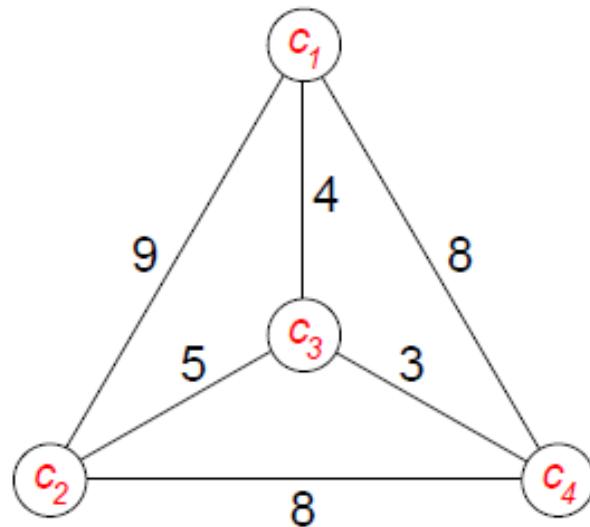
N	number of perms	million/sec	billion/sec	trillion/sec
10	3628800			
11	39916800	seconds		insignificant
12	479001600	minutes		
13	6227020800	hours	seconds	
14	87178291200	day	minute	
15	1307674368000	weeks	minutes	
16	20922789888000	months	hours	seconds
17	355687428096000	years	days	minutes
18	6402373705728000		months	hours
19	121645100408832000		years	days
20	2432902008176640000			month

O custo computacional aumenta significativamente com aumento de n

O problema do Caixeiro Viajante (Exponencial)

Um caixeiro viajante deseja visitar n cidades com início e término em uma mesma cidade, e cada cidade deve ser visitada uma única vez.

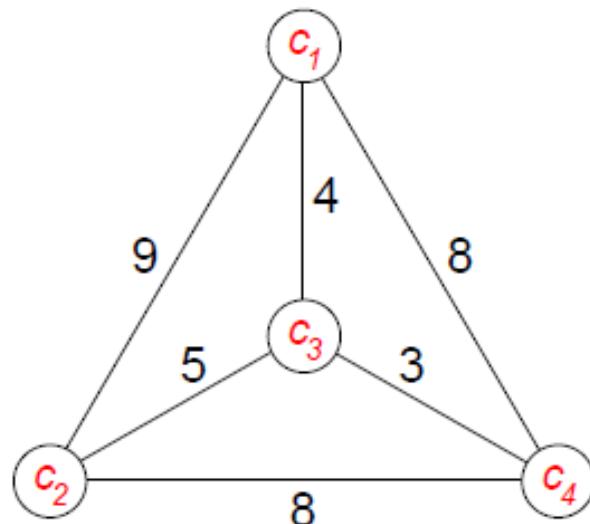
Supondo que sempre há uma estrada entre duas cidades quaisquer, o problema é encontrar a menor rota para a viagem.



O problema do Caixeiro Viajante (Exponencial)

Um caixeiro viajante deseja visitar n cidades com início e término em uma mesma cidade, e cada cidade deve ser visitada uma única vez.

Supondo que sempre há uma estrada entre duas cidades quaisquer, o problema é encontrar a menor rota para a viagem.



Por força bruta, a solução seria gerar todas as possibilidades de percurso.

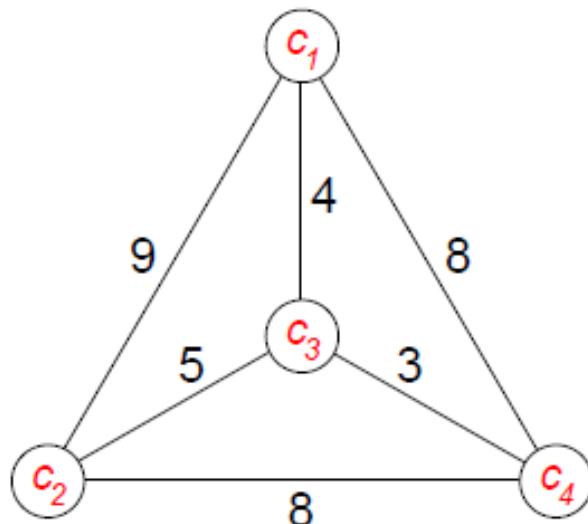
Para $n = 4$, $n! = 24$ percursos possíveis

1234	1324	3124	3214	2314	2134
1243	1342	3142	3241	2341	2143
1423	1432	3412	3421	2431	2413
4123	4132	4312	4321	4231	4213

O problema do Caixeiro Viajante (Exponencial)

Um caixeiro viajante deseja visitar n cidades com início e término em uma mesma cidade, e cada cidade deve ser visitada uma única vez.

Supondo que sempre há uma estrada entre duas cidades quaisquer, o problema é encontrar a menor rota para a viagem.



Crescimento da quantidade de percursos possíveis:

Para $n = 4$, $n! = 24$

Para $n = 10$, $n! = 3.628.800$

Para $n = 20$, $n! = 2.432.902.008.176.640.000$

1.4. Complexidade de Algoritmos

Exemplos de Complexidade em Grafos

Graph Data Structure Operations

Data Structure	Time Complexity					
	Storage	Add Vertex	Add Edge	Remove Vertex	Remove Edge	Query
Adjacency list	$O(V + E)$	$O(1)$	$O(1)$	$O(V + E)$	$O(E)$	$O(V)$
Incidence list	$O(V + E)$	$O(1)$	$O(1)$	$O(E)$	$O(E)$	$O(E)$
Adjacency matrix	$O(V ^2)$	$O(V ^2)$	$O(1)$	$O(V ^2)$	$O(1)$	$O(1)$
Incidence matrix	$O(V \cdot E)$	$O(E)$				

$|V|$ = quantidade de vértices

$|E|$ = quantidade de arestas

Algumas notações assintóticas consideram apenas V e E , ficando implícito que indicam a quantidade

Graph Algorithms

Algorithm	Time Complexity		Space Complexity
	Average	Worst	Worst
Dijkstra's algorithm	$O(E \log V)$	$O(V ^2)$	$O(V + E)$
A^* search algorithm	$O(E)$	$O(b^d)$	$O(b^d)$
Prim's algorithm	$O(E \log V)$	$O(V ^2)$	$O(V + E)$
Bellman–Ford algorithm	$O(E \cdot V)$	$O(E \cdot V)$	$O(V)$
Floyd-Warshall algorithm	$O(V ^3)$	$O(V ^3)$	$O(V ^2)$
Topological sort	$O(V + E)$	$O(V + E)$	$O(V + E)$

Perguntas? Sugestões?

