Matemática Computacional

Aula 4

Professor Luís Gonzaga de Paulo

Conversa inicial

Olá, aluno!! Preparado para mais uma aula de Matemática Computacional?!

Hoje abordaremos os aspectos principais de elementos e estruturas matemáticas de grande importância para a computação, tanto no que se refere a tratamento de dados quanto no estabelecimento de processos e fluxo de atividades para a solução de problemas.

Vamos começar?

Veja o que o professor Luís Gonzaga tem a dizer sobre a aula de hoje?
Acesse o material on-line e assista ao vídeo!

Contextualizando

Existem inúmeros tipos de aplicações de computador em nosso cotidiano que fazem uso maciço da matemática, além dos aspectos de cálculos e da lógica que já estudamos.

Por exemplo, você já se perguntou como é possível orientar-se matematicamente em um mapa?
Como o tempo pode ser calculado em um cronograma de projeto?
Como o computador consegue listar palavras em ordem alfabética?
E o que acontece quando procuro um dado qualquer em uma base de dados?
Em que é baseado o programa que comanda uma máquina ou um Robô?

Diversas respostas possíveis passam pelo conhecimento dos elementos matemáticos que estudaremos nessa aula.

Pode nos ajudar, professor Luís Gonzaga?
Veja o que ele responde no material on-line!

Tema 01: Grafos

O que é um Grafo?

Grafo é uma estrutura matemática de representação gráfica, utilizado para o estudo de relações entre os objetos ou elementos de um determinado conjunto. Os grafos são representados pela equação G (V,A) compostos de dois elementos:

- Os vértices - V, pontos ou círculos que representam os elementos, componentes ou objetos do conjunto em questão;
- As arestas - A, segmentos de retas ou linhas que representam as ligações, dependências ou caminhos entre os vértices.

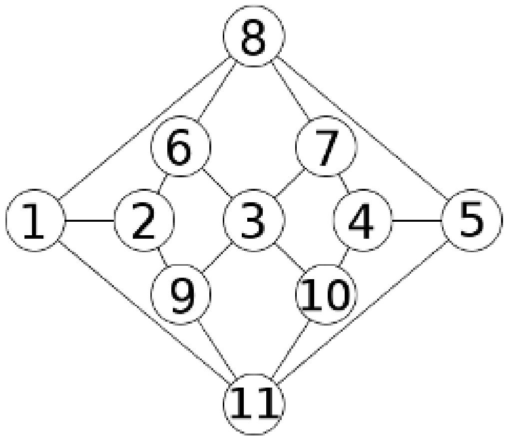


Figura de um grafo

Devido à sua aplicação, os grafos podem ter características distintas, tais como direção para as arestas, que passam a ser representada por uma →, caracterizando assim um **grafo orientado**, também denominado **grafo dirigido**, **grafo direcionado** ou **dígrafo**).

É possível também que um vértice seja ligado a ele próprio, formando um laço ou *loop.* As arestas podem receber rótulos para sua identificação, ou também ter um peso ou valor numérico associado.

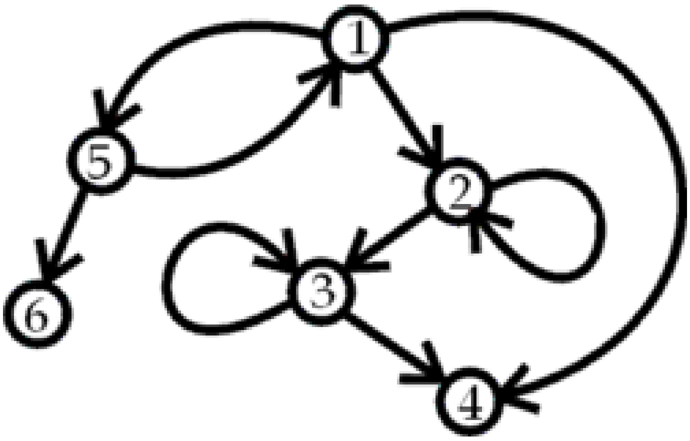


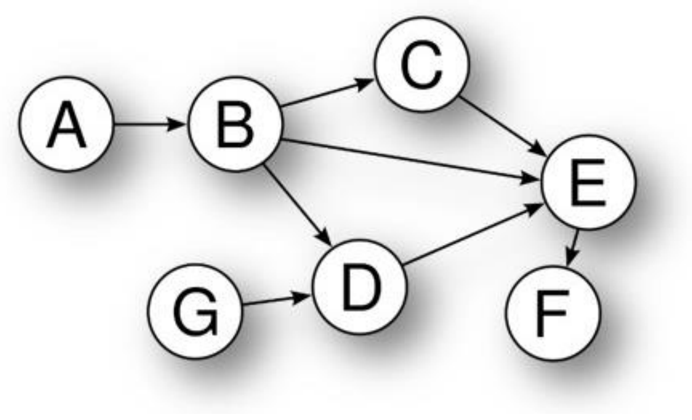
Figura de um dígrafo

G é denominado um grafo dirigido quando é formado por (V, A) tal que:

V = {A, B, C, D, E, F, G}

A = {(A,B), (B,C), (B,D), (B,E), (C,E), (D,E), (E,F), (G,D)}

Exemplo:

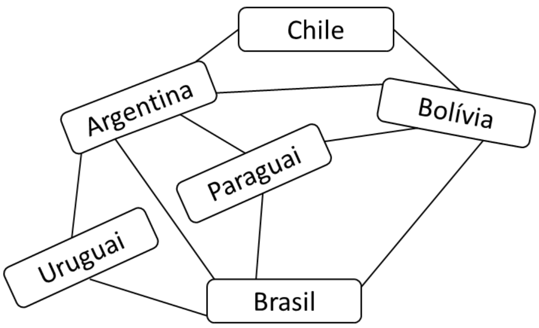


Exemplo de um grafo dirigido

Neste exemplo é possível identificar **rotas** ou **fluxos** entre os vértices, traçando caminhos entre eles. Em outro exemplo, G é um grafo não dirigido formado por (V, A), representando as fronteiras entre alguns países da América do Sul, onde:

V = {Argentina, Bolívia, Brasil, Chile, Paraguai, Uruguai}

A = {(Argentina, Bolívia), (Argentina, Brasil), (Argentina, Chile), (Argentina, Paraguai), (Argentina, Uruguai), (Bolívia, Brasil), (Bolívia, Chile), (Bolívia, Paraguai), (Brasil, Paraguai), (Brasil, Uruguai)}

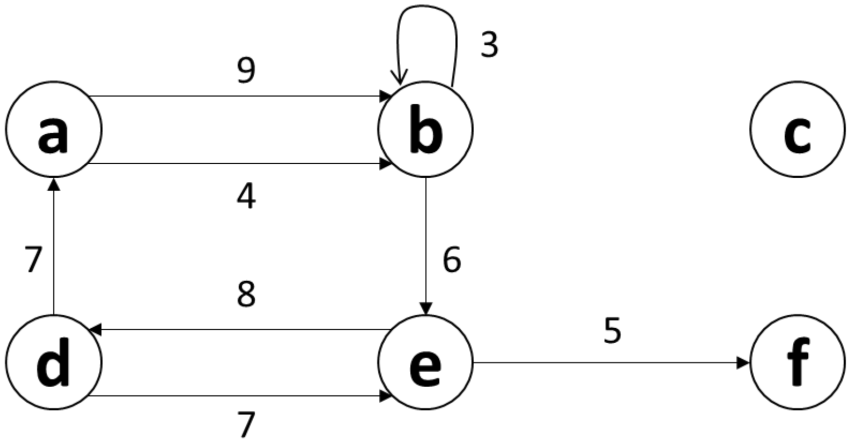


Exemplo de um Grafo não dirigido

Em mais um exemplo, G é um grafo dirigido rotulado formado por (V, N, A) tal que:

V = {a, b, c, d, e, f};

A = {(a,9,b), (a,4,b), (b,3,b), (b,6,e), (d,7,a), (d,7,e), (e,8,d), (e,5,f)}



Um grafo dirigido rotulado

Agora confira os detalhes

1) Note que foram acrescentados valores que identificam as arestas, e que há um vértice c para o qual não há nenhuma ligação, além de um *loop* no vértice b.

2) Uma aresta conecta dois vértices, denominados incidentes à esta aresta. O número de arestas incidentes a um vértice determina a valência (ou grau) de um vértice, sendo que os *loops* são contados duas vezes. Assim, como mostrado no último exemplo, o vértice a tem valência ou grau 3, o vértice b tem valência ou grau 5, o vértice f tem valência 1 e o vértice c tem valência 0 (zero).

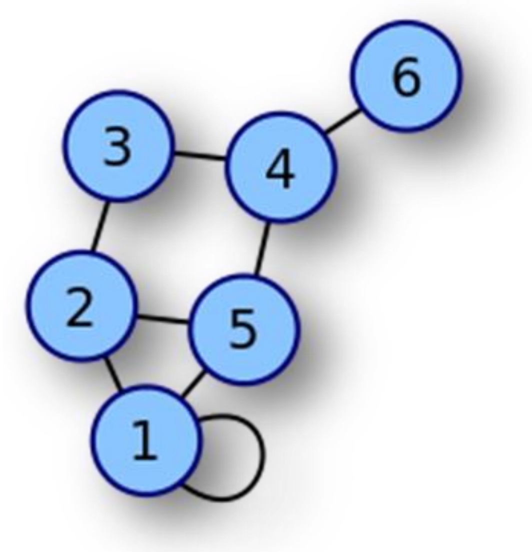
3) Já os vértices adjacentes são aqueles ligados entre si por uma aresta, como os vértices a e b, a e d, b e d e também d e e, e e f.

Um grafo finito com ***n*** vértices pode ser matematicamente representado por sua **matriz de adjacência**: uma matriz **n-por-n** cujo valor na linha **i** e coluna **j** fornece o número de arestas que conectam o i-ésimo ao j-ésimo vértices. Vejamos no exemplo:

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

Matriz de adjacência

O grafo finito em questão, do qual se trata a matriz de adjacência apresentada, é o seguinte:



Grafo finito para a matriz de adjacência apresentada.

Importante

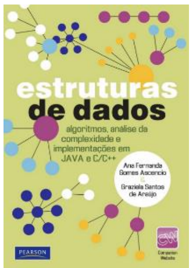
Um dígrafo contempla o grau de saída (o número de arestas saindo de um vértice) e o grau de entrada (o número de arestas entrando em um vértice)

No dígrafo, então, o grau de um vértice é igual à soma dos graus de saída e de entrada.

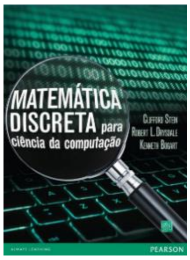
Veja mais detalhes com o professor Luís Gonzaga! Lá no vídeo do material on-line!

Dica de Leitura

Leia o capítulo 8 do livro da Ana Ascêncio: **"Estrutura de Dados: algoritmos, análise da complexidade e implementações em Java e C/C++".**



Confira também o Item 6.1 do livro do Stein, **"Matemática discreta para ciência da computação".**



Os dois livros estão na biblioteca virtual, que você acessa pelo portal ÚNICO no material on-line!

Tema 02: Árvores e árvores binárias

Em matemática computacional, um elemento é denominado **árvore** quando é um **grafo conexo**, isto é, sempre existe uma aresta – caminho - entre dois quaisquer de seus vértices, e também **acíclico**, ou seja, não permite ciclos ou *loops*. Em função disto, são consideradas verdadeiras as seguintes afirmações acerca de uma árvore:

- Toda árvore é um grafo, mas nem todo grafo é uma árvore
- Toda árvore é um grafo bipartido – ou seja, possui pelo menos dois níveis
- Toda árvore é um grafo planar, representado em um plano – ou em duas dimensões apenas
- Todo grafo conexo possui pelo menos uma árvore de extensão associada, composta de todos os seus vértices e algumas de suas arestas

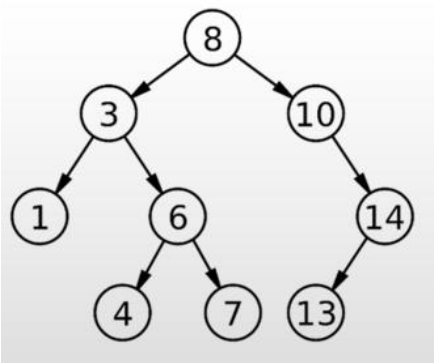
Dado um determinando grafo **G** com ***n*** vértices, podemos considerá-lo como uma árvore avaliando se **G** é um grafo:

Conexo, e há exatamente um caminho entre dois vértices quaisquer

Acíclico, e um simples ciclo é formado se uma nova aresta for incluída

Conexo, e deixará de ser conexo se qualquer aresta for removida de G

Conexo, acíclico e tem n − 1 arestas.



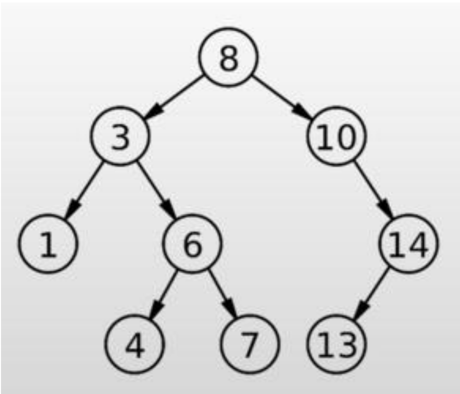
Uma árvore binária

Uma árvore é denominada enraizada se um vértice é escolhido como especial. Neste caso, este vértice escolhido passa a ser denominado Raiz, e uma árvore que não possui raiz é denominada livre.

Como já vimos, o número máximo de arestas – ou ramos - em um vértice – ou elemento - é chamado de Ordem ou Grau, e a máxima sequência de vértices de grau um ou maior define a quantidade de níveis de uma árvore.

Árvore binária

Uma **árvore binária** é definida como um grafo acíclico, conexo, dirigido e que cada nó não tem grau (ou ordem) maior que 2. Só existe um caminho entre dois nós distintos, e cada ramo da árvore é uma aresta dirigida, sem peso, que parte do antecessor (pai) e vai ao sucessor (filho), formando uma hierarquia.



Uma árvore binária.

Em uma árvore binária a **profundidade** de um nó é a distância deste nó até a raiz. E um conjunto de nós com a mesma profundidade é denominado **nível** da árvore. O nó de maior profundidade define a altura da árvore. Os nós de uma árvore binária possuem graus zero, um ou dois, isto é, têm nenhum vértice, um vértice ou dois vértices, respectivamente. E um nó de grau zero, ou seja, que não está conectado a nenhum outro, é denominado folha.

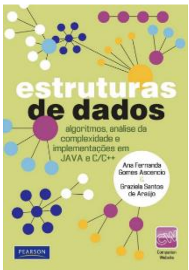
Em programação de computadores, a construção e o acesso aos elementos de uma árvore binária é geralmente implementada por uma operação ou função recursiva. De fato, a maioria das operações realizadas com o uso das árvores binárias utilizam a recursão.

A árvore binária é o tipo de árvore mais utilizado na computação, e uma de suas principais aplicações é a construção de listas ordenadas para permitir a busca binária.

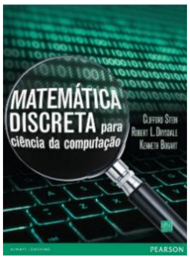
Professor Luís Gonzaga, ajuda a gente?! Pode!
Lá no vídeo do material on-line!

Dica de Leitura

Leia o capítulo 7 do livro da Ana Ascêncio: **"Estrutura de Dados: algoritmos, análise da complexidade e implementações em Java e C/C++".**



Confira também o Item 6.2 do livro do Stein, **"Matemática discreta para ciência da computação".**



Os dois livros estão na biblioteca virtual, que você acessa pelo portal ÚNICO no material on-line!

Saiba Mais

Acesse também a página "Árvore binária", vai ilustrar ainda mais o que estamos estudando!

Clique aqui: http://www.ft.unicamp.br/liag/siteEd/definicao/arvore-binaria.php

As **máquinas de estado** são abstrações de processos ou equipamentos reais (mecânicos, eletrônicos, software, etc), isto é, um modelo matemático de sistema que tem entradas e saídas discretas que representa, em determinado momento, um estado pré-estabelecido.

As máquinas de estado também são conhecidas como Autômatos, ou *FSM - Finite State Machine.* Este tipo de abstração permite a modelagem de grande número de problemas.

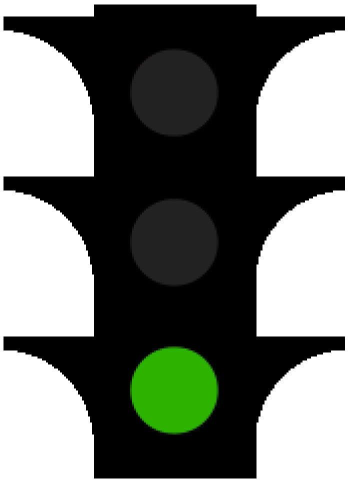
A principal característica da máquina de estados é que, apesar de possuir a capacidade de representar vários estados, só pode estar em um estado por vez, denominado **Estado Atual**, que representa a informação relativa às entradas passadas, a qual é necessária para determinar o comportamento do sistema para as próximas entradas.

A representação de uma máquina de estados.

O funcionamento de uma FSM considera que um **estado** armazena informações sobre o passado, e reflete as mudanças desde a entrada neste estado até o momento presente.

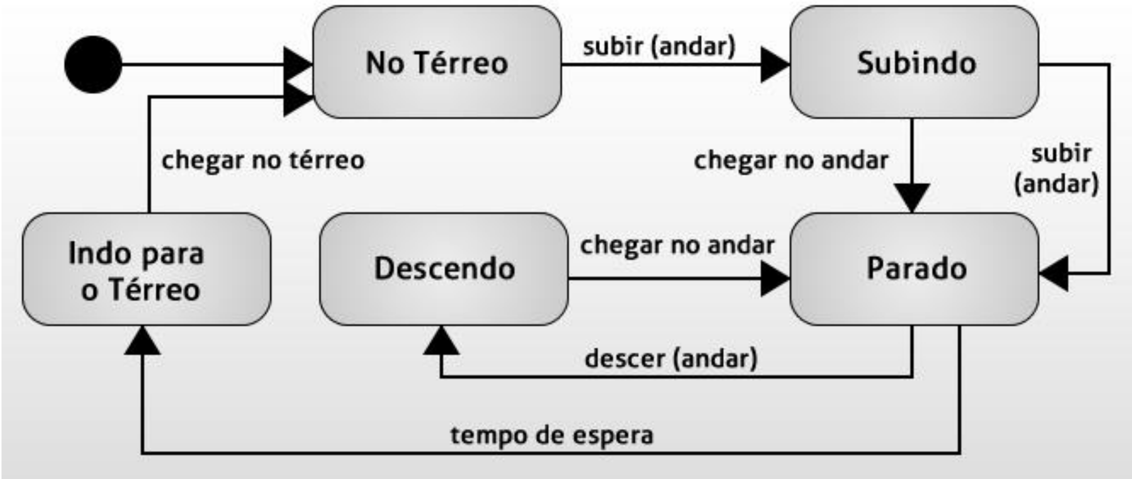
Uma **transição** indica uma mudança de estado - geralmente descrita ou especificada por uma condição que precisa ser realizada para que a transição ocorra.

Já uma **ação** é a descrição de uma atividade que deve ser realizada para que ocorra uma transição ou que ocorre em função de uma transição.



Um semáforo (farol ou sinal): exemplo de uma FSM

O funcionamento de uma máquina de estado pode ser representado pelo **Diagramas de Estado:**



Um diagrama de estado de FSM de um elevador.

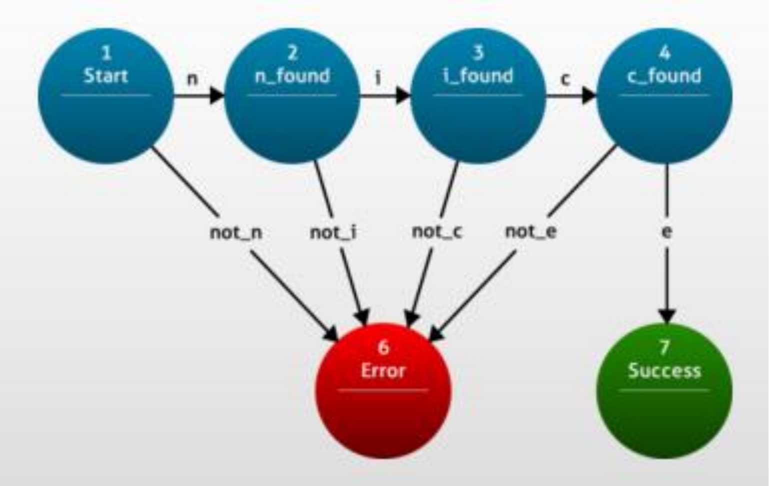
As máquinas de estado podem ser representadas também pelas **Tabelas de Transição de Estado**.

Essas tabelas podem representar uma máquina de estados finita, e contém as informações completas sobre as ações e os estados para cada um dos estados e todas as transições registradas ou previstas.

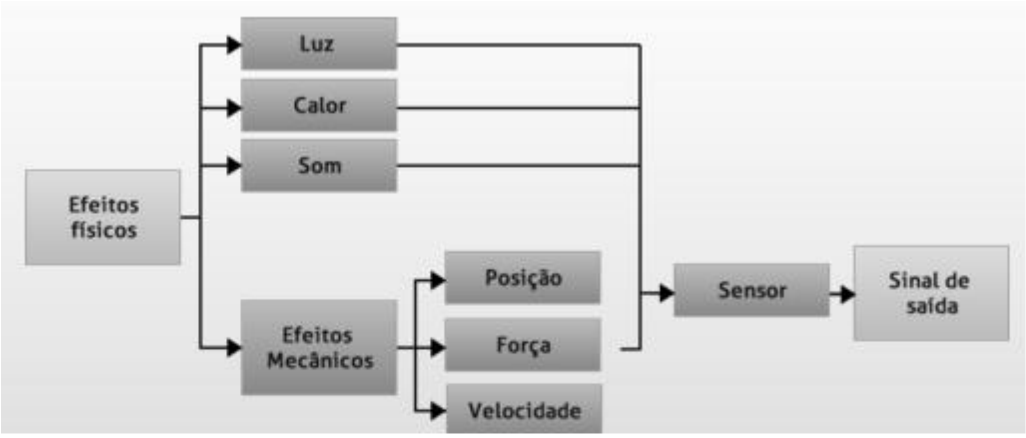
Tabela de Transição de Estados			
Condição	Estado		
	Estado A	Estado B	Estado C
1	...	...	...
2	...	Estado C	...
3	...	...	...

Tabela de estados de uma FSM.

Existem basicamente dois tipos de Máquinas de estado: os tipos **Aceitadores** (ou **Reconhecedores**), que produzem uma saída binária, dizendo **sim** ou **não** para responder se a entrada é aceita pela máquina ou não, e os tipos **Transdutores**, que geram uma saída baseada em uma entrada e/ou um estado utilizando ações, e são utilizados para aplicações de controle.



FSM do tipo aceitadores



FSM do tipo Transdutores

As Máquinas de Estado são amplamente utilizadas na modelagem do comportamento de um software, de um sistema ou de um aplicativo, no projeto de sistemas digitais (*hardware* e *software*), na Engenharia de software, em projetos de compiladores e em protocolos de rede, e no estudo da computação e das linguagens.

"Máquinas de Estados"?

O professor Luís Gonzaga sabe tudo! Veja no vídeo do material on-line!

Trocando ideias

Procure na bibliografia da disciplina, no material complementar e na internet, usos práticos dos elementos e estruturas apresentados nesta aula. Identifique os benefícios e a complexidade destas aplicações, como evoluíram em função do tempo e também as possíveis alternativas ao seu uso.

Acesse o fórum "Elementos e estruturas" no UNIVIRTUS e apresente suas descobertas, e discuta com seus colegas a respeito do que vocês encontraram e concluíram.

Na Prática

Acesse as listas de exercícios disponibilizadas no UNIVIRTUS e procure resolvê-las, se necessário com o apoio de seus colegas, trocando informações pelo fórum "Elementos e estruturas" no UNIVIRTUS.

Na bibliografia e no material complementar são apresentados diversos exemplos de máquinas de estado, alguns de forma gráfica e outros por tabelas de estado.

Complemente a definição das FSM apresentadas, criando as tabelas de estado para aquelas que estão somente em formato gráfico, e criando os diagramas para os que apresentam somente a tabela de estados.

Agora é a vez do professor Luís Gonzaga...

Veja no vídeo do material on-line!

Síntese

Nesta aula foram apresentados os grafos, um modelo matemático de grande importância para a computação.

Vimos também os conceitos, as características e operações realizadas por meio das árvores binárias, um elemento muito utilizado em armazenamento e recuperação de informações.

E finalmente foram apresentadas as máquinas de estado, uma formulação matemática para a simulação das máquinas e processos reais.

Para encerrar...

O professor Luís Gonzaga está volta! Assista ao vídeo do material on-line!

Referências

Ascêncio, Ana F. G.; Araújo, Graziela S. de. **Estrutura de Dados**: algoritmos, análise da complexidade e implementações em Java e C/C++. São Paulo: Pearson Prentice Hall, 2010.

Stein, C.; Drysdale, R. L. e Bogart, K. **Matemática discreta para ciência da computação**. São Paulo:Pearson Education do Brasil, 2013.