Lógica Computacional

Construção da Tabela Verdade

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

Nesta webaula, você compreenderá o conceito de tabela verdade.

Verificação da satisfatibilidade e validade de fórmulas

Tanto o hardware como software computacional são baseados na lógica computacional, que, por sua vez, é baseada na lógica formal. Tudo é possível porque a lógica computacional produz somente resultados binários para suas fórmulas, ou seja, Verdadeiro ou Falso.

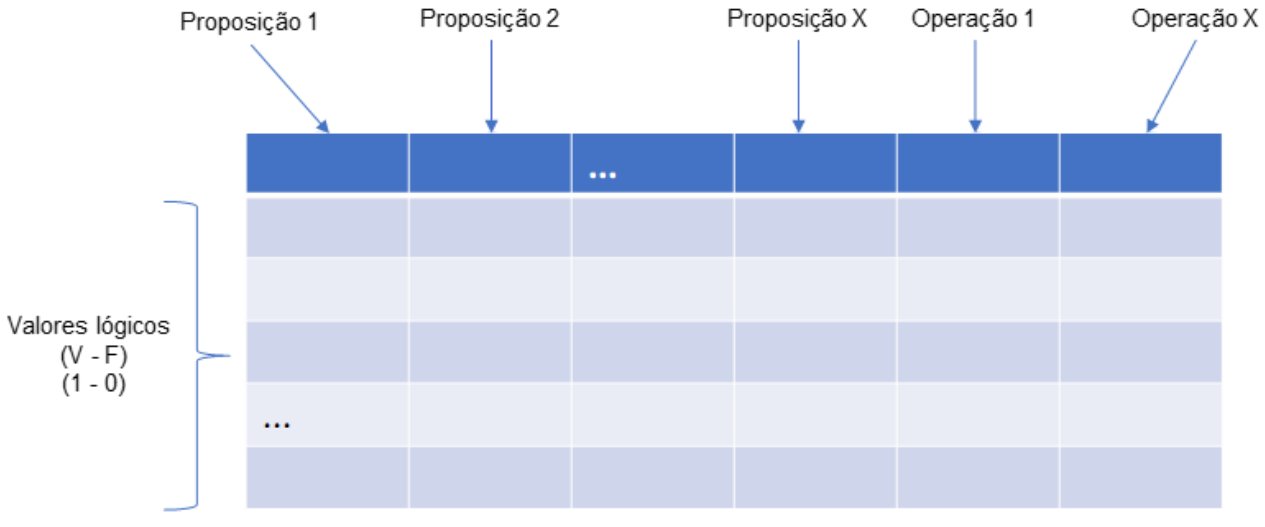
> As fórmulas podem ora ser valoradas em 0, em cujo caso a valoração falsifica a fórmula, ora ser valoradas em 1, em cujo caso a valoração satisfaz a fórmula. Estes fatos motivam a classificação das fórmulas de acordo com o seu comportamento diante de todas as valorações possíveis de seus átomos. Um dos grandes desafios da computação é encontrar métodos eficientes para decidir se uma fórmula é satisfazível/insatisfazível, ou se é válida/falsificável. Um dos primeiros métodos propostos na literatura para a verificação da satisfatibilidade e validade de fórmulas é o método da Tabela da Verdade.
>
> — SILVA; FINGER; MELO, 2017, p. 13

Tabela verdade

A Tabela Verdade é um mecanismo que permite valorar fórmulas de forma genérica a partir de entradas binárias e conectores lógicos.

A Figura 1 mostra o esquema geral para uma Tabela Verdade. Nas primeiras colunas coloca-se as proposições (quantas forem necessárias testar), em seguida, coloca-se as operações lógicas que se deseja valorar. Nas linhas são colocadas os valores lógicos (V – F) tanto para as proposições quanto para os resultados das fórmulas.

Figura 1 | Esquema geral da Tabela Verdade



Fonte: elaborado pela autora.

A quantidade de linhas (combinações) aumenta exponencialmente com a quantidade de proposições seguindo a regra 2^n, em que *n* é o número de proposições. Portanto, para duas proposições tem-se $2^2 = 4$ linhas; para três proposições tem-se $2^3 = 8$ linhas; para quatro proposições tem-se $2^4 = 16$ linhas; e assim por diante.

Tabela verdade da conjunção (AND – E)

O conectivo lógico de conjunção (AND - E) é utilizado para realizar uma operação binária entre duas proposições quando se deseja obter um resultado verdadeiro se, e somente se, as duas proposições forem verdadeiras. Confira a Figura 2 a seguir:

Figura 2 | Tabela Verdade da conjunção

A	B	$A \wedge B$
V	V	V
V	F	F
F	V	F
F	F	F

Fonte: elaborado pela autora.

Tabela Verdade da disjunção (OR - OU)

O conectivo lógico de disjunção (OR - OU) é utilizado para realizar uma operação binária entre duas proposições quando se deseja obter um resultado falso se, e somente se, as duas proposições forem falsas. Confira a Figura 3 a seguir:

Figura 3 | Tabela Verdade da disjunção

A	B	$A \vee B$
V	V	V
V	F	V
F	V	V
F	F	F

Fonte: elaborado pela autora.

Tabela Verdade para negação

O operador lógico de negação tem a função de inverter seja uma entrada ou o resultado de uma operação. Confira a Figura 4 a seguir:

Figura 4 | Negação de uma proposição

A	$\neg A$
V	F
F	V

Fonte: elaborado pela autora.

A Figura 5 mostra a negação do resultado de uma fórmula que possui a conjunção. Os parênteses na fórmula devem ser usados com consciência, pois influenciam o resultado. Na Figura 5, os parênteses indicam que a negação fora deles deve inverter o resultado de tudo o que está dentro.

Figura 5 | Negação do resultado de uma fórmula

A	B	$A \wedge B$	$\neg (A \wedge B)$
V	V	V	F
V	F	F	V
F	V	F	V
F	F	F	V

Fonte: elaborado pela autora.

Para aprimorar seu conhecimento, recomendamos a leitura das 5 primeiras páginas da seguinte obra:

GERSTING, J. L. **Fundamentos matemáticos para a ciência da computação**: matemática discreta e suas aplicações. 7. ed. Rio de Janeiro: LTC, 2017. (Disponível na Biblioteca Virtual.)

Também recomendamos a leitura da página 9 a 15 da obra:

SILVA, F. S. C. da; FINGER, M.; MELO, A. C. V. de. **Lógica para computação**. 2. ed. São Paulo: Cengage Learning, 2017. (Também disponível na Biblioteca Virtual.)

Lógica Computacional

Resultados na Tabela Verdade

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

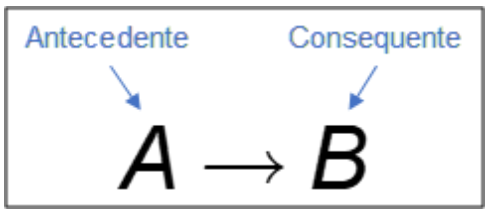
Construímos uma tabela verdade para testarmos todos os resultados possíveis para uma combinação de entradas em determinada fórmula. Considerando seus conhecimentos em tabela verdade, nesta webaula estudaremos a implicação lógica.

Implicação lógica

Uma fórmula é composta por proposições e operadores lógicos, por exemplo a negação (NOT), a conjunção (AND) e a disjunção (OR). Além desses conectores, as proposições podem ser combinadas na forma "**se** proposição 1, **então** proposição 2". O conectivo lógico dessa combinação é o **condicional**, representado por $\rightarrow$, e significa que a verdade da proposição 1 implica a verdade da proposição 2 (GERSTING, 2017). Em outras palavras, podemos dizer que, dada uma sequência de proposições, a partir da operação condicional é possível chegar a uma conclusão (um resultado), que é uma nova proposição.

Observe na Figura 1 que a primeira parte, antes do conector, é chamada de antecedente, e a segunda parte é chamada de consequente.

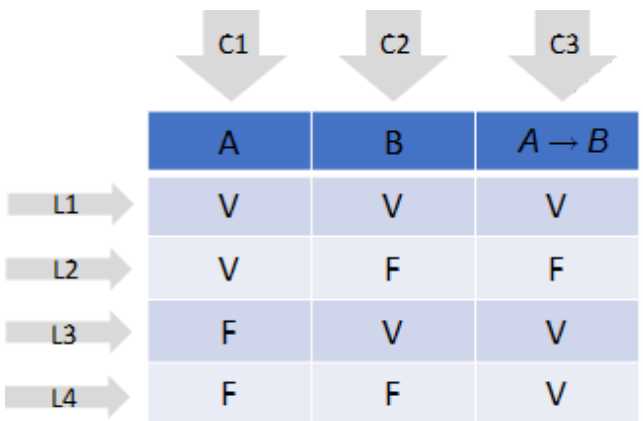
Figura 1 | Implicação lógica



Fonte: elaborado pela autora.

A tabela verdade para o condicional está apresentada na Figura 2:

Figura 2 | Tabela verdade para o condicional



Fonte: elaborado pela autora.

Construímos uma tabela verdade para testarmos todos os resultados possíveis para uma combinação de entradas em determinada fórmula. Considerando seus conhecimentos em tabela verdade, nesta webaula estudaremos a implicação lógica.

Resultados da implicação lógica

Os resultados da implicação não são tão óbvios. Para entendermos, vamos utilizar o exemplo usado em uma nota de aula do professor Chibeni, da Unicamp (CHIBENI, 2019). Considere as seguintes proposições:

A: soltar a pedra.
B: a queda da pedra.

A fórmula $A \rightarrow B$ deve ser lida como "Se a pedra for solta, então a pedra cairá".

Agora, vamos avaliar todas as respostas possíveis com base na tabela verdade da Figura 2:

Figura 2 | Tabela verdade para o condicional

	C1	C2	C3
	A	B	$A \rightarrow B$
L1	V	V	V
L2	V	F	F
L3	F	V	V
L4	F	F	V

Fonte: elaborado pela autora.

- **Linha 1:** na primeira linha, temos como entrada a verdade das proposições A e B. Traduzindo para nosso exemplo, quer dizer que a pedra foi solta e caiu, portanto a condição era verdadeira e o resultado é V (coluna 3).
- **Linha 2:** na linha dois, temos como entrada a verdade para A e a falsidade para B. No nosso exemplo, quer dizer que a pedra foi solta, mas não caiu. Nesse caso, a condição não é verdadeira e o resultado é F (coluna 3).
- **Linhas 3 e 4:** nas terceira e quarta linhas, temos como entrada a falsidade para A (que é o antecedente); nesse caso, não há como avaliar a condicional, e o resultado é tomado como verdadeiro.

Tautologia, contradição e contingência

A tabela verdade deve ser usada como um método exaustivo de extração de resultados. Quando o resultado de uma fórmula obtém somente V como resposta, a fórmula é denominada **tautologia**; por outro lado, quando o resultado de uma fórmula obtém somente F como resposta, a fórmula é denominada **contradição**. Quando uma tabela verdade não é uma tautologia e não é uma contradição, ela é uma **contingência**.

Os resultados de tautologia, contradição e contingência são úteis para testar se duas fórmulas são equivalentes (se são iguais). A equivalência é representada pelo operador $\Leftrightarrow$.

Para saber se duas fórmulas são equivalentes, é necessário construir a tabela verdade e verificar se a equivalência é uma tautologia. Um importante resultado da equivalência são as leis De Morgan. Leia mais sobre esse importante tema acessando as páginas 1 a 10 da obra:

GERSTING, J. L. **Fundamentos matemáticos para a ciência da computação:** matemática discreta e suas aplicações. 7. ed. Rio de Janeiro: LTC, 2017. (Disponível na Biblioteca Virtual.)

Lógica Computacional

Aplicações da tabela verdade

Você sabia que seu material didático é interativo e multimídia? Isso significa que você pode interagir com o conteúdo de diversas formas, a qualquer hora e lugar. Na versão impressa, porém, alguns conteúdos interativos ficam desabilitados. Por essa razão, fique atento: sempre que possível, opte pela versão digital. Bons estudos!

Nesta webaula, vamos estudar a ordem de precedência dos conectivos lógicos.

Ordem de precedência dos conectivos lógicos

A matemática faz parte do cotidiano desde os primeiros anos escolares. As fórmulas começam simples, com as quatro operações básicas e depois vão ganhando complexidade. Assim como as fórmulas matemáticas, é possível construir expressões lógicas mais complexas a partir da combinação das proposições, dos conectivos e dos parênteses. As expressões lógicas podem ser simples, como $A \rightarrow B$, ou podem combinar diversos operadores lógicos, em expressões mais complexas, por exemplo $(A \wedge B) \vee (B \rightarrow C)$.

Para resolver uma expressão lógica que combina várias preposições com conectores lógicos é preciso obedecer a seguinte **regra de precedência:**

1. Para expressões que possuem parênteses, primeiro são efetuadas as operações lógicas dentro dos parênteses mais internos.
2. $\neg$ (Negação)
3. $\wedge, \vee$ (Conjunção e disjunção)
4. $\rightarrow$ (Implicação)
5. $\leftrightarrow$ (Bicondicional)

Ao seguir rigorosamente a ordem de precedência dos operadores, o uso de parênteses pode ser omitido nos casos adequados. Por exemplo, a fórmula $A \vee (\neg B)$ pode simplesmente ser escrita como $A \vee \neg B$, pois, por causa da ordem de precedência, a negação será realizada primeiro, mesmo sem parênteses.

Dada uma fórmula com várias proposições, conectores e parênteses dentro de parênteses, a **resolução deve começar pelos parênteses mais internos**. Por exemplo, a fórmula $((A \vee B) \rightarrow C) \wedge A$ deve ter a seguinte ordem de resolução:

1. $A \vee B$ (parênteses mais internos).
2. $((A \vee B) \rightarrow C)$ (parênteses mais externos).
3. $((A \vee B) \rightarrow C) \wedge A$ (operação fora dos parênteses).

A Figura 1, a seguir, mostra o resultado para essa fórmula.

Figura 1 | Tabela verdade para a fórmula $((A \vee B) \rightarrow C) \wedge A$

			P	Q	
A	B	C	$A \vee B$	$P \rightarrow C$	$Q \wedge A$
V	V	V	V	V	V
V	V	F	V	F	F
V	F	V	V	V	V
V	F	F	V	F	F
F	V	V	V	V	F
F	V	F	V	F	F
F	F	V	F	V	F
F	F	F	F	V	F

Fonte: elaborado pelo autor.

Veja que foram usadas proposições intermediárias para nomear os resultados. Primeiro, obtém-se P que é o resultado 1, depois utiliza-o para obter Q, que é o resultado 2 e, por fim, o Q é usado para obter o resultado final da fórmula.

Operações e regras lógicas na computação

Uma dúvida que pode surgir é como aplicar todas essas operações e regras lógicas no universo da programação. A resposta é simples: elas são utilizadas para construir uma sequência de instruções, chamada de **algoritmo**, que soluciona algum problema. Mais precisamente, as operações lógicas são usadas em estruturas condicionais (ou estruturas de decisão) e têm o objetivo de realizar testes alterando o fluxo de execução de um programa, de acordo com a resposta obtida.

Por exemplo, em um site de aluguel de imóveis, quando o cliente seleciona opções como: imóvel do tipo apartamento com 1 dormitório; 1 banheiro; sem vaga de garagem, essa seleção é transformada em uma expressão lógica do tipo: Apartamento **E** 1 quarto **E** 1 banheiro **E** sem garagem.

Na programação, a implicação $A \rightarrow B$ é traduzida para **se A... então B**... e significa que, se A for uma proposição verdadeira, então B acontecerá.

A estrutura condicional faz parte do arsenal de técnicas de programação que permitem alterar o fluxo de execução do programa em detrimento de decisões que são tomadas. Para entender como esse recurso é utilizado no mundo da computação, recomenda-se o vídeo **Estruturas Condicionais 1 - Curso de Algoritmos #07**, do professor Gustavo Guanabara.

ESTRUTURAS Condicionais I – Curso de Algoritmos #07 – Gustavo Guanabara. Curso em Vídeo. YouTube, [s.d.].

Para visualizar o vídeo, acesse seu material digital.