

DIAGRAMAS DE DECISÃO BINÁRIOS (DDB'S)

Luiz Carlos Vieira

22 de Setembro de 2015

Instituto de Matemática e Estatística da Universidade de São Paulo

REPRESENTAÇÃO DE FUNÇÕES BOOLEANAS

funções booleanas

- Formalismo descritivo importante para sistemas de *hardware* e de *software*
 - tais como circuitos síncronos e assíncronos
 - sistemas reativos
 - e programas de estados finitos
- Representação computacional eficiente
 - e que auxilia na verificação de sistemas

definição: variáveis booleanas

Definição 6.1(a)

Uma variável booleana x é uma variável que só pode assumir os valores 0 e 1. Denotamos variáveis booleanas por x_1, x_2, \dots , e x, y e z, \dots

definição: funções booleanas

Definição 6.1(b)

As seguintes funções são definidas no conjunto $\{0, 1\}$:

- $\overline{0} \stackrel{\text{def}}{=} 1$ e $\overline{1} \stackrel{\text{def}}{=} 0$;
- $x \cdot y \stackrel{\text{def}}{=} 1$ se x e y têm valor 1; caso contrário, $x \cdot y \stackrel{\text{def}}{=} 0$;
- $x + y \stackrel{\text{def}}{=} 0$ se x e y têm valor 0; caso contrário, $x + y \stackrel{\text{def}}{=} 1$;
- $x \oplus y \stackrel{\text{def}}{=} 1$ se exatamente um entre x e y é igual a 1; caso contrário, $x \oplus y \stackrel{\text{def}}{=} 0$.

variáveis e funções booleanas

Ou seja:

Uma função booleana f com n variáveis é uma função de $\{0, 1\}^n$ para $\{0, 1\}$. Escrevemos $f(x_1, x_2, \dots, x_n)$ ou $f(\mathcal{V})$ para indicar que uma representação sintática de f só depende das variáveis booleanas em \mathcal{V} .

Note que \cdot , $+$ e \oplus são funções booleanas com duas variáveis, enquanto que \neg é uma função booleana com uma única variável. As funções binárias \cdot , $+$ e \oplus são escritas em notação infixa, isto é, escrevemos $x + y$ em vez de $+(x, y)$, etc.

alguns exemplos de funções booleanas

$$1. f(x, y) \stackrel{\text{def}}{=} x \cdot (y + \overline{x})$$

$$2. g(x, y) \stackrel{\text{def}}{=} x \cdot y + (1 \oplus \overline{x})$$

$$3. h(x, y, z) \stackrel{\text{def}}{=} x + y \cdot (x \oplus \overline{y})$$

$$4. k() \stackrel{\text{def}}{=} 1 \oplus (0 \cdot \overline{1})$$

representação de funções booleanas

O que já se estudou até então são duas formas de se representar funções booleanas.

- **fórmulas proposicionais:**

- \wedge denota \cdot
- \vee denota $+$
- \neg denota $\bar{}$
- e \top e \perp denotam, respectivamente, 1 e 0

- **tabelas-verdade:** representam funções booleanas de maneira óbvia

tabelas-verdade de funções booleanas

Tabela 1: Tabela-verdade da função booleana

$$f(x, y) \stackrel{\text{def}}{=} \overline{x + y}$$

x	y	$f(x, y)$
1	1	0
0	1	0
1	0	0
0	0	1

Tabela 2: Tabela-verdade da fórmula proposicional

$$\phi : \neg(p \vee q)$$

p	q	ϕ
V	V	F
F	V	F
V	F	F
F	F	V

vantagens e desvantagens

Tabela 3: Vantagens e desvantagens das tabelas-verdade e das fórmulas proposicionais ao representar funções booleanas

	Tabelas-Verdade	Fórmulas Proposicionais
Vantagens	<ul style="list-style-type: none">• operações¹ simples	<ul style="list-style-type: none">• representação compacta
Desvantagens	<ul style="list-style-type: none">• ineficientes em espaço• computacionalmente intratável	<ul style="list-style-type: none">• operações¹ difíceis• computacionalmente custoso

¹verificação de satisfação e validade, e comparação de duas funções booleanas

comparação geral

Tabela 4: Comparação geral das formas de representação de funções booleanas

Representação de funções booleanas	compacta?	teste de		operações booleanas		
		satisfação	validade	·	+	-
fórmulas proposicionais	muitas vezes	difícil	difícil	fácil	fácil	fácil
fórmulas FND	algumas vezes	fácil	difícil	difícil	fácil	difícil
fórmulas FNC	algumas vezes	difícil	fácil	fácil	difícil	difícil
tabelas-verdade ordenadas	nunca	difícil	difícil	difícil	difícil	difícil
DDBO's reduzidos	muitas vezes	fácil	fácil	+/-	+/-	fácil

definição: árvore de decisão binária finita

Definição 6.3

Seja T uma árvore de decisão binária finita. Então T determina *uma única* função booleana das variáveis nos nós não-terminais da seguinte maneira:

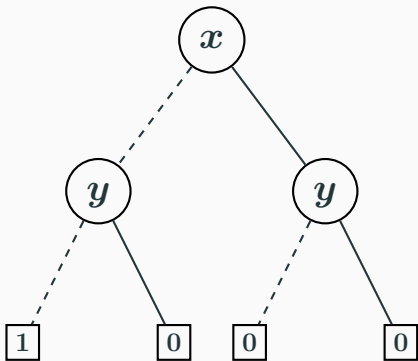
Dada uma atribuição de 0's e 1's às variáveis booleanas que ocorrem em T , começamos pela raiz de T e pegamos a linha tracejada sempre que o valor da variável no nó atual é 0; caso contrário, percorremos a linha sólida. O valor da função é o valor do nó terminal atingido.

por exemplo

- Árvore da função:

$$f(x, y) \stackrel{\text{def}}{=} \overline{x + y}$$

- Para encontrar $f(0, 1)$:



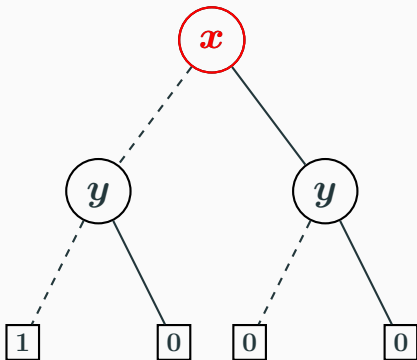
por exemplo

- Árvore da função:

$$f(x, y) \stackrel{\text{def}}{=} \overline{x + y}$$

- Para encontrar $f(0, 1)$:

1. inicia-se pela raiz



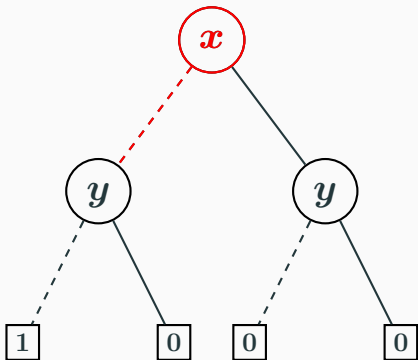
por exemplo

- Árvore da função:

$$f(x, y) \stackrel{\text{def}}{=} \overline{x + y}$$

- Para encontrar $f(0, 1)$:

1. inicia-se pela raiz
2. como x é 0, segue-se pela linha pontilhada



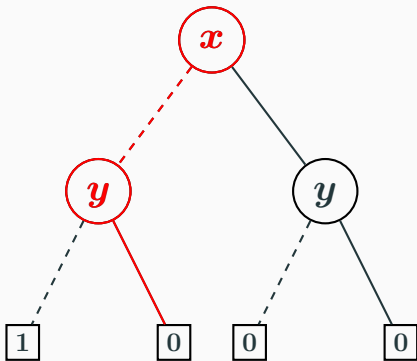
por exemplo

- Árvore da função:

$$f(x, y) \stackrel{\text{def}}{=} \overline{x + y}$$

- Para encontrar $f(0, 1)$:

1. inicia-se pela raiz
2. como x é 0, segue-se pela linha pontilhada
3. como y é 1, segue-se pela linha sólida



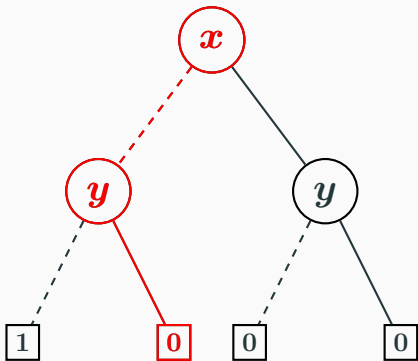
por exemplo

- Árvore da função:

$$f(x, y) \stackrel{\text{def}}{=} \overline{x + y}$$

- Para encontrar $f(0, 1)$:

1. inicia-se pela raiz
2. como x é 0, segue-se pela linha pontilhada
3. como y é 1, segue-se pela linha sólida
4. chega-se à folha 0; logo $f(0, 1) = 0$

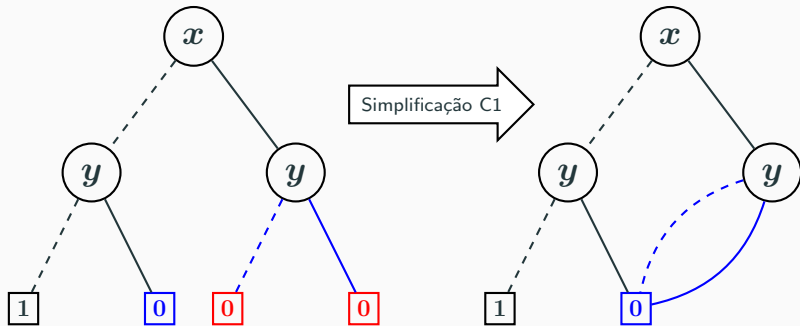


semelhanças com tabelas-verdade

- Árvores de Decisão Binárias são semelhantes às tabelas-verdade em relação ao tamanho
 - se f depender de n variáveis booleanas, a árvore correspondente terá pelo menos $2^{n+1} - 1$ nós (contra as 2^n linhas da tabela verdade)
- Mas muitas vezes elas contêm redundâncias que podem ser exploradas

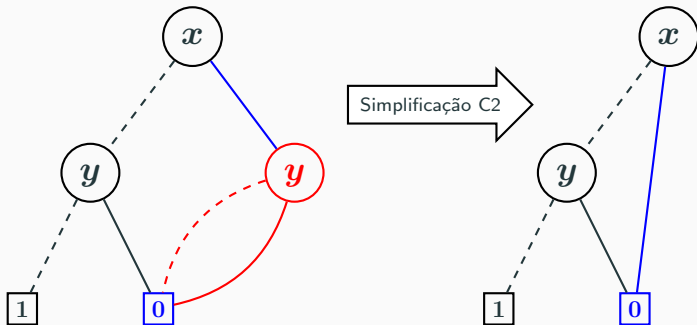
A exploração de redundâncias em Árvores de Decisão Binárias faz com que deixem de ser árvores e se tornem grafos. Assim, passam a ser chamados de Diagramas de Decisão Binários (BDDs).

remoção de nós terminais duplicados



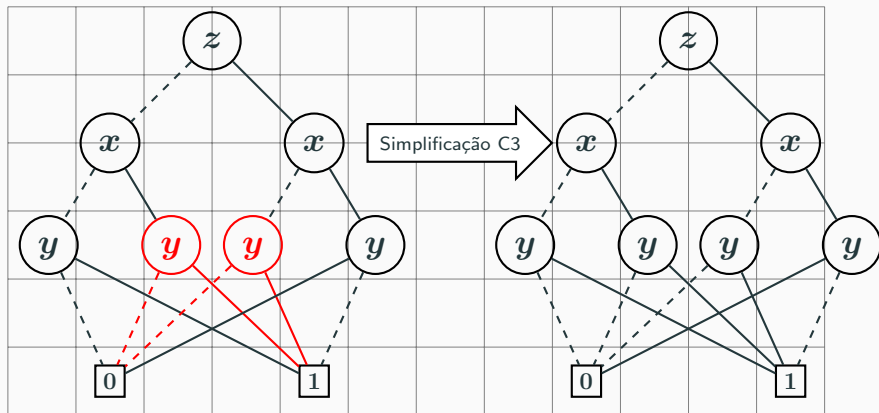
Se um DDB contém mais de um nó terminal 0, redirecionam-se todas as arestas que apontam para tais nós para apenas um deles. Repete-se o mesmo processo para os nós terminais com 1

remoção de testes redundantes

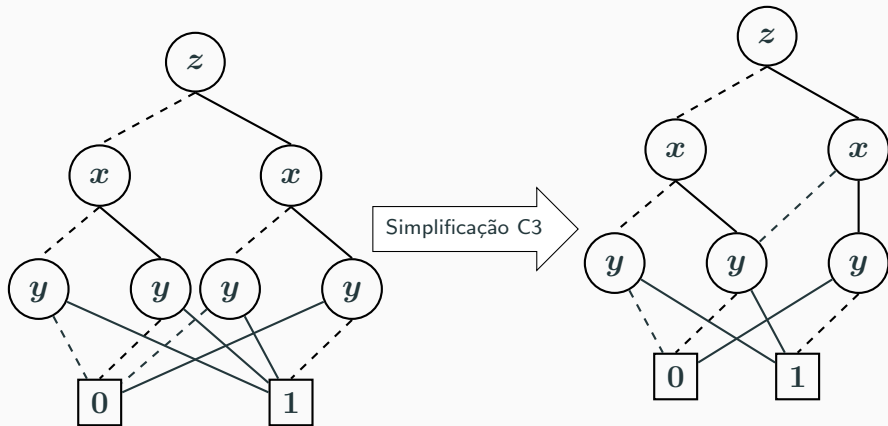


Se ambas as arestas de um nó n apontam para o mesmo nó m , elimina-se o nó n , enviando todas as arestas que nele chegavam para m .

teste



remoção de nós não-terminais duplicados



Se dois nós distintos n e m são raízes de sub-DDBs idênticas, pode-se eliminar um deles redirecionando todas as arestas que

mais simplificações

