Engenharia de Computação

ECM253 – Linguagens Formais, Autômatos e Compiladores

Métodos de prova em lógica proposicional

Slides da disciplina ECM253 – Linguagens Formais, Autômatos e Compiladores
Curso de Engenharia de Computação
Instituto Mauá de Tecnologia
Prof. Marco Antonio Furlan de Souza

- Um **argumento** é uma **implicação** em que o **antecedente** é uma **conjunção** de **proposições** e o **consequente** também é uma **proposição**. **Formulação geral**:

$$P_1 \wedge P_2 \wedge P_3 \wedge \ldots \wedge P_n \rightarrow Q$$

- Nesta formulação, $P_1, P_2, P_3, \ldots, P_n$ são proposições denominadas de **hipóteses do argumento** e Q é uma proposição denominada de **conclusão do argumento**;
- Tanto os P_i quanto Q devem ser **fbfs**;
- **Lê-se** um argumento assim: *Q é uma **conclusão lógica** de $P_1, P_2, P_3, \ldots, P_n$ sempre que a **verdade** de $P_1, P_2, P_3, \ldots, P_n$ **levar à verdade** de Q*.
- Um **argumento** é dito **válido** se e somente se ele for uma **tautologia**.

Argumentos válidos

- **Por exemplo**, considerar a argumentação a seguir: *Se George Washington foi o primeiro presidente dos Estados Unidos, então John Adams foi o primeiro vice-presidente. George Washington foi o primeiro presidente dos Estados Unidos. Portanto, John Adams foi o primeiro vice-presidente.*
- Pode-se **identificar duas proposições** nesta argumentação:
  - $A =$ *George Washington foi o primeiro presidente dos Estados Unidos*
  - $B =$ *John Adams foi o primeiro vice-presidente dos Estados Unidos*.
- Então, esta argumentação pode ser **formalizada em lógica proposicional** assim:

$$(A \rightarrow B) \wedge A \rightarrow B$$

- **Pergunta**: Será que ela é válida?

Métodos para verificar tautologias

Tabela verdade

- Pode-se **provar** que um **argumento é valido** criando uma **tabela verdade** para este argumento;
- Para o argumento $(A \rightarrow B) \wedge A \rightarrow B$, tem-se a seguinte tabela verdade:

A	B	A → B	(A → B) ∧ A	(A → B) ∧ A → B
V	V	V	V	V
V	F	F	F	V
F	V	V	F	V
F	F	V	F	V

- Então o argumento $(A \rightarrow B) \wedge A \rightarrow B$ é uma tautologia! Logo, trata-se de um **argumento válido**.

- O problema do **método com tabela verdade** é **a quantidade de linhas geradas** quando se tem muitos símbolos proposicionais – o uso do **algoritmo facilita a verificação** de uma **tautologia** (mas só funciona com a lógica proposicional);
- Para entender o algoritmo, primeiramente deve-se **lembrar que**, quando se tem uma **implicação como** $P \rightarrow Q$, esta será **falsa** apenas se $P = \mathsf{V}$ e $Q = \mathsf{F}$;
- O **algoritmo para testar tautologia** utiliza tal fato e utiliza a técnica de **prova por refutação**: para provar que $P_1 \wedge P_2 \wedge, \ldots, \wedge P_n \rightarrow Q$ **é uma tautologia**, basta **não conseguir provar** que $P_1 \wedge P_2 \wedge, \ldots, \wedge P_n \rightarrow \neg Q$ ou seja, **inicialmente hipotetiza-se que**:
  - $\neg Q$ é **verdadeiro**;
  - $P_1 \wedge P_2 \wedge, \ldots, \wedge P_n$ é **falso**;

Métodos para verificar tautologias

Algoritmo para testar tautologia

- Nessas condições,**utilizando os valores verdade iniciais**, **descobre-se por várias iterações** os **valores verdade** de **todas as proposições envolvidas** até que uma das seguintes condições ocorra:
  - **Se um mesmo símbolo** (basta um) **apresentar dois valores verdade** (V e F), então tem-se uma **inconsistência** – neste caso **não foi possível provar** $P_1 \wedge P_2 \wedge, \dots, \wedge P_n \rightarrow \neg Q$. Neste caso **as hipóteses iniciais foram refutadas**, logo $P_1 \wedge P_2 \wedge, \dots, \wedge P_n \rightarrow Q$ **é uma tautologia**;
  - **Senão**, **todos os símbolos** possuem um único valor verdade (V ou F). Então foi possível provar que $P_1 \wedge P_2 \wedge, \dots, \wedge P_n \rightarrow \neg Q$ é verdadeiro – logo $P_1 \wedge P_2 \wedge, \dots, \wedge P_n \rightarrow Q$ **não é uma tautologia**.

O algoritmo

```
procedimento TestarTautologia(fbf P; fbf Q)
//Dados fbfs P e Q, decidir se a fbf P → Q é uma tautologia
início
    //Assumir que P → Q NÃO é uma tautologia
    P = verdadeiro    //atribuir V para P
    Q = falso   //atribuir F para Q
    repita
        para cada fbf composta que já tenha um valor verdade
        atribuído, atribua valores verdade a seus componentes
    até que todas as ocorrências de símbolos tenham valores
        verdade atribuídos
    se algum símbolo possui dois valores verdade
        então // Há uma contradição – é falso que não é tautologia
            escreva("P → Q É uma tautologia")
        senão //Provou-se que é verdade que não é uma tautologia
            escreva("P → Q NÃO É uma tautologia")
    fim se
fim TestarTautologia
```

Exemplo

- Aplicar o algoritmo para verificar se $(A \rightarrow B) \wedge A \rightarrow B$ é uma tautologia ou não:
  - Atribuir V à $(A \rightarrow B) \wedge A$;
  - Atribuir F à B;
  - Na **primeira iteração** para que o valor de $(A \rightarrow B) \wedge A$ seja V, então deve-se atribuir V à $(A \rightarrow B)$ e V à $\wedge A$ (conjunção);
  - Na **segunda repetição**, como o valor de $A \rightarrow B$ é V então é o caso que A deva ser V ou F; mas B não pode ser F, então atribui-se V à B. Porém, como A é V e B é F (do **início** e da **primeira repetição**), então B possui agora dois valores verdade: V e F!
  - Como B ficou com dois valores – a prova que não é tautologia falhou – logo, tem-se uma tautologia!

- É uma **sequência de fbfs** na qual **cada fbf ou** é uma **hipótese ou** é o **resultado** da **aplicação** de uma das **regras de derivação** (inferência e/ou equivalência) de um sistema formal **às fbfs** existentes **na sequência**;
- Esta técnica permite a **verificação de tautologias**. Forma geral:

P_1	(hipótese)
P_2	(hipótese)
$\vdots$	
P_n	(hipótese)
fbf_1	(obtida pela aplicação de regra de derivação)
fbf_2	(obtida pela aplicação de regra de derivação)
$\vdots$	
Q	(obtida pela aplicação de regra de derivação)

Métodos para verificar tautologias

Sequência de prova

- **Regras de derivação**
  - **Devem ser escolhidas** de modo que o **sistema formal seja correto** (somente argumentos válidos podem ser provados) e **completo** (todo argumento válido possui prova);
  - **Duas categorias** para regras de derivação:
    - **Regras de equivalência**: permite **reescrever fbfs** individuais de outra forma, porém equivalente;
    - **Regras de inferência**: permite que **novas fbfs** sejam **derivadas** (deduzidas) a partir de **fbfs presentes** na sequência de prova.
  - A **cada passo**, deve-se **escolher** apenas **uma regra** a ser **aplicada por vez** – é um **algoritmo**!

Regras de equivalência

- Estabelece que certos **pares de fbfs** são **equivalentes**.

Regras de equivalência		
Expressão	**Equivalente à**	**Nome/abreviação**
$P \vee Q$ $P \wedge Q$	$Q \vee P$ $Q \wedge P$	Comutativa/com
$(P \vee Q) \vee R$ $(P \wedge Q) \wedge R$	$P \vee (Q \vee R)$ $P \wedge (Q \wedge R)$	Associativa/ass
$\neg(P \vee Q)$ $\neg(P \wedge Q)$	$\neg P \wedge \neg Q$ $\neg P \vee \neg Q$	Leis de DeMorgan/dm
$P \rightarrow Q$	$\neg P \vee Q$	Implicação/imp
P	$\neg\neg P$	Negação dupla/dn
$P \leftrightarrow Q$	$(P \rightarrow Q) \wedge (Q \rightarrow P)$	Definição de bicondicional/bc

Métodos para verificar tautologias

Sequência de prova

- **Regras de equivalência**
  - Estabelece que certos **pares de fbfs** são **equivalentes** (cont.).

Regras de equivalência		
Expressão	**Equivalente à**	**Nome/abreviação**
$P \lor (Q \land R)$	$(P \lor Q) \land (P \lor R)$	Leis Distributivas/dis
$P \land (Q \lor R)$	$(P \land Q) \lor (P \land R)$	
$P \lor P$	P	Leis Idempotentes/idem
$P \land P$	P	
$P \lor \mathsf{F}$	P	Leis de Identidade/id
$P \land \mathsf{V}$	P	(V = verdadeiro; F = falso)
$P \lor \neg P$	V	Leis de Inverso/inv
$P \land \neg P$	F	(V = verdadeiro; F = falso)
$P \lor \mathsf{V}$	V	Leis de Dominação/dom
$P \land \mathsf{F}$	F	(V = verdadeiro; F = falso)
$P \lor (P \land Q)$	P	Leis de Absorção/abs
$P \land (P \lor Q)$	P	

- **Regras de equivalência**
  - **Exemplo de aplicação.** Provar que $(P \to (Q \wedge R)) \to ((P \to Q) \wedge (P \to R))$. Neste caso, a **hipótese** é $(P \to (Q \wedge R))$ e a **conclusão** é $((P \to Q) \wedge (P \to R))$.

1. $P \to (Q \wedge R)$	(hipótese)
2. $\neg P \vee (Q \wedge R)$	1, imp
3. $(\neg P \vee Q) \wedge (\neg P \vee R)$	2, dis
4. $(P \to Q) \wedge (\neg P \vee R)$	3, imp
5. $(P \to Q) \wedge (P \to R)$ $\square$	4, imp

Regras de inferência

- Uma **regra de inferência** estabelece que **se uma ou mais fbfs** existentes **correspondem** com a **primeira parte de um padrão de regra**, **então pode-se adicionar à sequência uma nova fbf produzida** pela **segunda parte do padrão** que foi correspondido.

Regras de inferência		
De	**Pode derivar**	**Nome/Abreviação**
P, $P \to Q$	Q	*Modus ponens*/mp
$P \to Q$, $\neg Q$	$\neg P$	*Modus tollens*/mt
P, Q	$P \wedge Q$	Conjunção/con
$P \wedge Q$	P, Q	Simplificação/sim
P	$P \vee Q$	Adição /add

Regras de derivação

- **Exemplo de aplicação.** Provar que $((\neg P \vee \neg Q) \to (R \wedge S)) \wedge (R \to T) \wedge \neg T \to P$ é um argumento válido. Aqui se utilizam regras de equivalência e de inferência.

1. $(\neg P \vee \neg Q) \to (R \wedge S)$	(hipótese)
2. $(R \to T)$	(hipótese)
3. $\neg T$	(hipótese)
4. $\neg R$	2,3, mt
5. $(\neg R \vee \neg S)$	4, add
6. $\neg(R \wedge S)$	5, dm
7. $\neg(\neg P \vee \neg Q)$	6,1, mt
8. $P \wedge Q$	7, dm
9. P $\square$	8, sim

Teste seus conhecimentos

- Utilizar o método de **sequência de prova** nos exercícios a seguir:
  1) **Provar** que $(A \rightarrow B \rightarrow C) \rightarrow (A \wedge B \rightarrow C)$.
  2) Utilizando a identidade apresentada no enunciado do exercício anterior, **provar a lei do silogismo**: $(A \rightarrow B) \wedge (B \rightarrow C) \rightarrow (A \rightarrow C)$.

Referências bibliográficas

[1] GERSTING, J.L. **Fundamentos matemáticos para a ciência da computação**. 4.ed. Rio de Janeiro, RJ: LTC, 2001. 538 p. ISBN 85-216-1263-X.