Aula 06 – fotos (quadro negro)

Hoje: aula 07.

____ x ____

L é uma lista em Python

```
if L != []:
```

```
if len(L) > 0:
```

if L:

Recap: pela nossa def, um contexto é uma atribuição de V ou F para todos os símbolos proposicionais

Teorema da Concordância v1:

O valor de uma fórmula φ da LC em um contexto depende apenas dos valores neste contexto dos símbolos proposicionais que ocorrem em φ.

Teorema da Concordância v2:

Para quaisquer

- φ fórmula da LC
- c_0, c_1 contextos

se c_0 & c_1 concordam sobre os valores dos símbolos proposicionais que ocorrem em φ, então também concordam sobre o valor de φ.

Prova: Indução em φ

B) φ é atômico

Então $c_0(\varphi) = c_1(\varphi)$ por hipótese.

R¬) φ é $(\neg\psi)$.

(HI): Para quaisquer c_0, c_1 contextos se c_0 & c_1 concordam sobre os valores dos símbolos proposicionais que ocorrem em ψ, então também concordam sobre o valor de ψ.

Quero provar:

Para quaisquer c_0, c_1 contextos.

se c_0 & c_1 concordam sobre os valores dos símbolos proposicionais que ocorrem em φ, então também concordam sobre o valor de φ.

Sejam c_0, c_1 contextos e suponha que concordem sobre os símb. prop. de φ.

Logo, também concordam sobre os de ψ.
Assim pela HI temos $c_0(\psi) \underset{*}{=} c_1(\psi)$.

Mas
$$c_0(\varphi) = \begin{cases} V & \text{se } c_0(\psi) = F \\ F & \text{se } c_0(\psi) = V \end{cases}$$
$$\underset{*}{=} \begin{cases} V & \text{se } c_1(\psi) = F \\ F & \text{se } c_1(\psi) = V \end{cases}$$
$$= c_1(\varphi).$$

Rv) φ é $(\psi \vee \beta)$

(HIψ) Para quaisquer c_0, c_1 contextos se c_0 & c_1 concordam sobre os valores dos símbolos proposicionais que ocorrem em ψ, então também concordam sobre o valor de ψ.

(HIβ) Para quaisquer c_0, c_1 contextos se c_0 & c_1 concordam sobre os valores dos símbolos proposicionais que ocorrem em β, então também concordam sobre o valor de β.

Sejam c_0, c_1 contextos e suponha que concordem sobre os símb. prop. de φ.

Logo, como os símbs props de ψ & de β são também símbs props de φ, temos por HI $c_0(\psi) \underset{\star}{=} c_1(\psi)$ & $c_0(\beta) \underset{\diamond}{=} c_1(\beta)$

Logo $c_0(\varphi) = \begin{cases} V, & \text{se } c_0(\psi) = V \\ c_0(\beta), & \text{c.c.} \end{cases}$

$$\underset{\star,\diamond}{=} \begin{cases} V, & \text{se } c_1(\psi) = V \\ c_1(\beta), & \text{c.c.} \end{cases}$$

$$= c_1(\varphi).$$

$R\rightarrow$, $R\wedge$, $R\leftrightarrow$) Exercício. $\blacksquare$

Tabelas de Verdade

Pelo teorema da concordância, se φ tem n símbolos proposicionais distintos ocorrendo em si, os contextos (que são potencialmente infinitos em quantidade) podem ser particionados em 2^n categorias, de acordo

Com os valores V/F atribuídos a esses símbs. props; dentro de cada categoria, todos os contextos atribuem o mesmo valor V/F a φ.

Assim, podemos listar essas categorias como linhas de uma tabela; nas colunas podemos listar todas as subfórmulas de φ, ordenadas de acordo com a organização da recursão (menos complexas à esquerda). Cada célula da tabela tem V/F de acordo com o valor que os contextos naquela "classe" atribuem à fórmula daquela coluna.

Ex: Uma tabela de verdade para

$\varphi = ((p \vee q) \to p)$

subfórmulas: p, q, $(p \vee q)$, φ

$2^2 = 4$ linhas

p	q	$p \vee q$	$(p \vee q) \to p$
V	V	V	V
V	F	V	V
F	V	V	F
F	F	F	V

Representação compacta da Tabela:
em vez de fazer uma coluna para cada subfórmula de φ, podemos fazer o seguinte:

- escrever φ "espaçadamente" no cabeçalho da tabela
- separar cada símbolo proposicional e conectivo na sua própria coluna

- daí preencher as linhas, na ordem:

1º) símbolos proposicionais (apenas a 1ª ocorrência de cada)

2º) cada subfórmula composta tem seu conectivo "principal" (a raiz da sua árvore); o valor da subfórmula é dado na célula que está na coluna deste conectivo.

Ex: $\varphi = ((p \vee q) \to p)$

$(p$	$\vee$	$q)$	$\to$	p
V	V	V	V	V
V	V	F	V	V
F	V	V	F	F
F	F	F	V	F

"resposta"

Um outro "defeito" das tabelas de verdade é a quantidade de linhas, mas esse é mais complicado... (voltaremos a esse ponto mais tarde).

—×—

Podemos categorizar as fórmulas da LC de acordo com seus "comportamentos semânticos":

def: Seja φ fórmula da LC. Dizemos que φ é

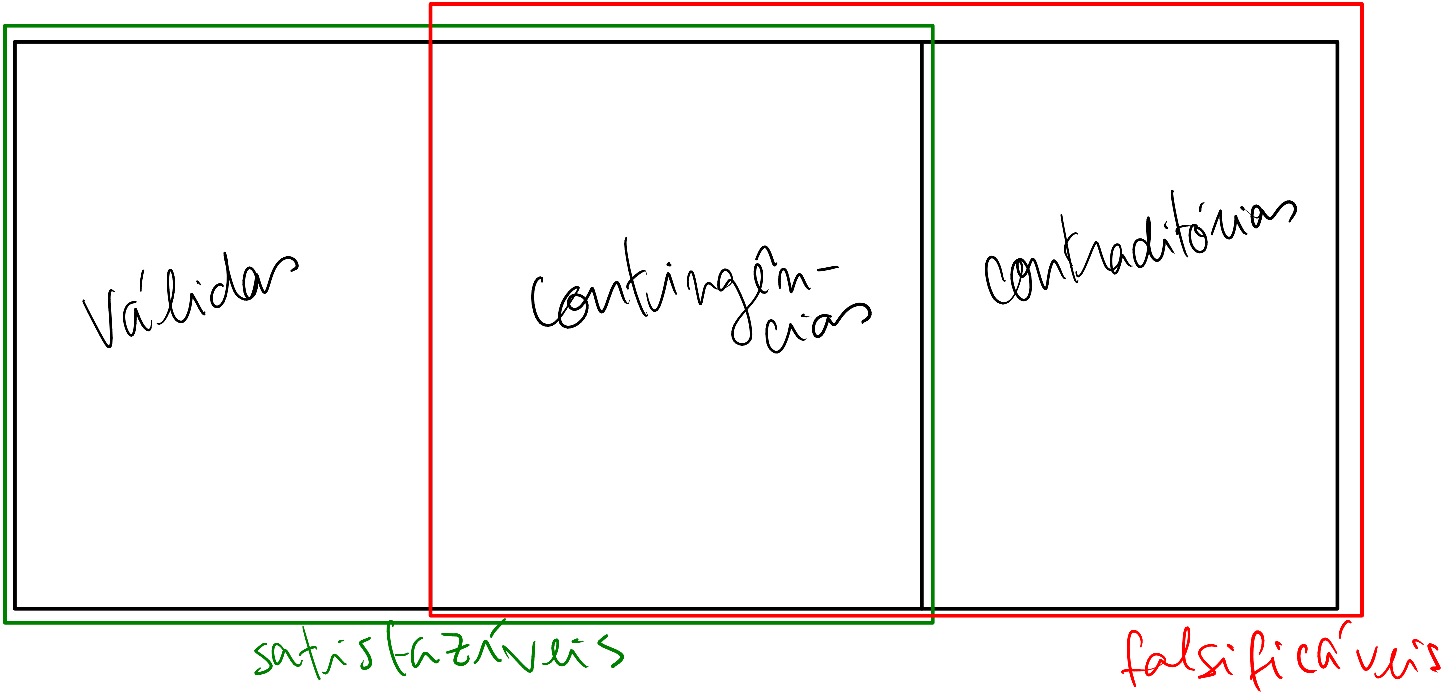
1) tautologia ou validade se φ é V em qualquer contexto

2) contradição ou insatisfazível se φ é F em qualquer contexto

3) satisfazível se φ é V em algum contexto

4) <u>falsificável</u> se φ é F em algum contexto

5) <u>contingência</u> se φ é V em algum contexto e F em algum contexto


válidas
contingências
contraditórias
satisfazíveis
falsificáveis

Em certo sentido, um problema central/motor da lógica é

Problema da Classificação

Entrada: uma fórmula φ

Saída: dizer em quais das 5 classes acima φ se encontra.

Mais específicos:

Para cada uma das 5 classes, temos o problema: dada φ, dizer se φ está naquela classe.

Para a classe dos satisfatíveis, o problema é conhecido como SAT. Não sabemos se existe uma forma eficiente (polinomial) de resolver SAT

Essa é a pergunta P vs NP!
SAT foi o primeiro exemplo de problema NP-completo: todo problema da classe NP pode ser reduzido (polinomialmente) a SAT (de forma que resolver SAT em tempo polinomial implica resolver o problema dado tb em

tempo polinomial) : Teorema de Cook -Levin.