IntroProof

lec01
2025-08-18

O aluno João não é um string; nem é um nome; é uma pessoa mesmo!

João : Person "João" : String João : Name

João tem 3 filhos : ~~Bool~~ Prop

João tem Brasil filhos

Type error, type-check

verdadeiro ou falso? Nenhum dos dois! É tão errado (type error) que nem chega perto de ter chances de ser falso!

João tem ____ filhos : ~~Int~~ Nat → Prop

Nat: domínio; Prop: codomínio

Int : Type Prop : Type

2 + 4 : Int (+: operação binária no Int)

true OR true : Bool (OR: operação binária no Bool)

Int : Type Nat → Prop : Type

↳ inferimos essa tipagem sem calcular "o valor" de 2+4.

PALAVRINHAS

Objeto vs Prop

Bool vs Prop

Static vs Dynamic

Dados vs Alvo

Usar vs Atacar

Comando vs Comentário

Atômico vs Composto

Prop

Como saber se algo é uma Prop?

Dica: veja se faz sentido dizer:

Suponha (que) P ...

Eu acho que P ...

Eu vou demonstrar P ...

Quantos tipos?

ou qualquer outro tipo atômico que apareceu até agora: Person, String, ..

Por causa do (→) e do Nat já é garantido que temos uma infinidade de tipos:

Nat , Nat → Nat , Nat → (Nat → Nat) , Nat → (Nat → (Nat → Nat)) , ...

hw: descreva os habitantes desses tipos

Construindo Props (conectivos lógicos)

	-elim Como usar	-intro Como atacar
_ & _	?	Split.
_ ou _	?	Esc-L Esc-R.
_ ⇒ _	?	Suponha P.
_ ⇔ _	?	?
¬ _	?	?
_ = _	?	?
∀ ...	?	?
∃ ...	?	?
⋮		

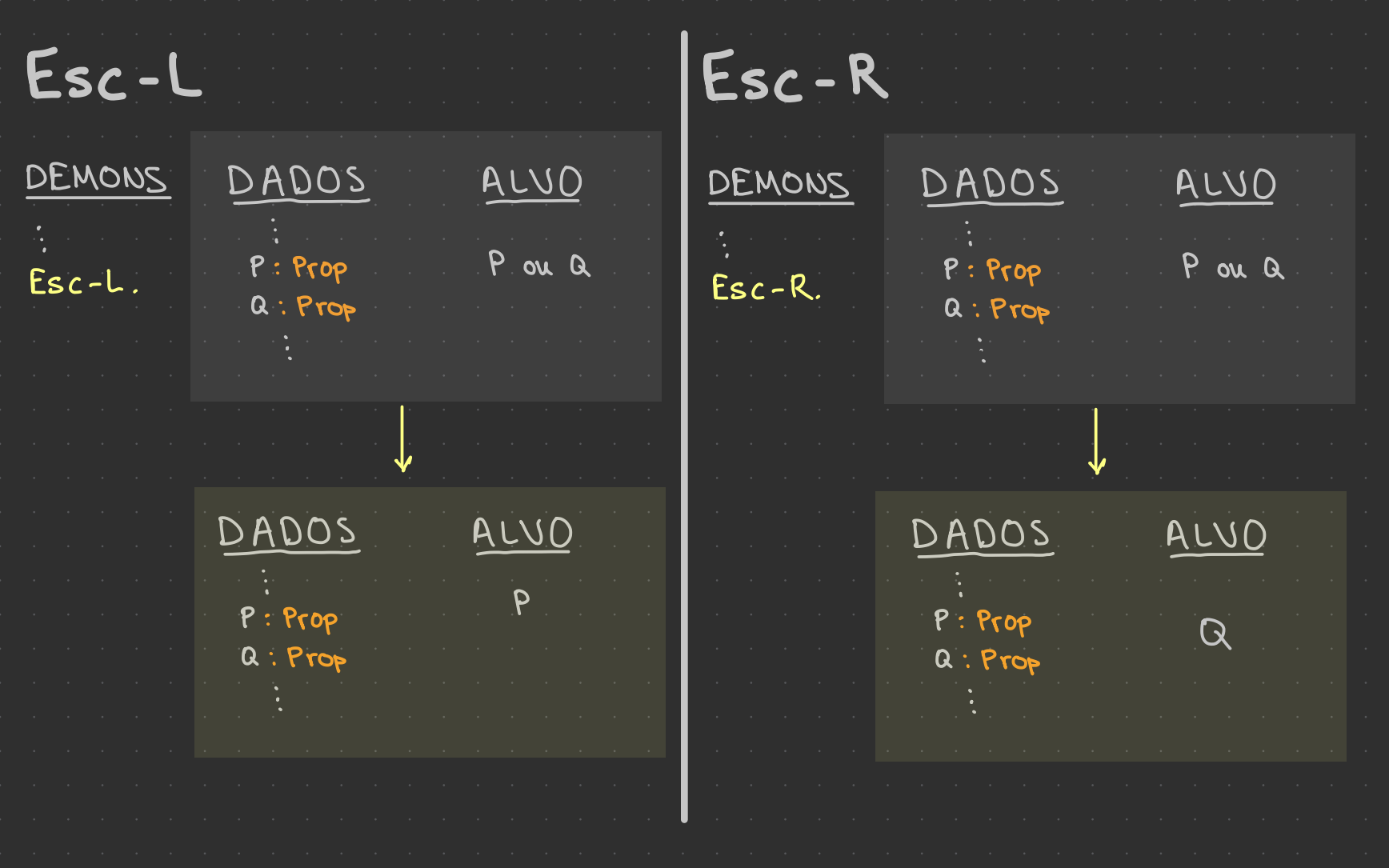
comandos

Para cada comando precisamos saber:

- quando é executável/permitido
- qual o seu efeito no tabuleiro DADOS-ALVO

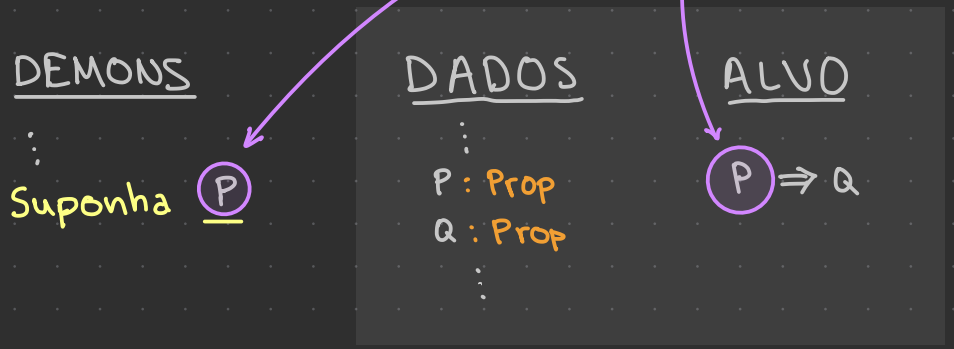
Split

DEMONS
Split.
DADOS
P : Prop
Q : Prop
ALVO
P & Q
DADOS
P : Prop
Q : Prop
ALVO
P
DADOS
P : Prop
Q : Prop
ALVO
Q
precisamos "matar" ambos


Esc-L
DEMONS
Esc-L.
DADOS
ALVO
P : Prop
Q : Prop
P ou Q
DADOS
ALVO
P : Prop
Q : Prop
P
Esc-R
DEMONS
Esc-R.
DADOS
ALVO
P : Prop
Q : Prop
P ou Q
DADOS
ALVO
P : Prop
Q : Prop
Q

Suponha _ .

a única coisa que posso supor é a premissa da implicação do alvo

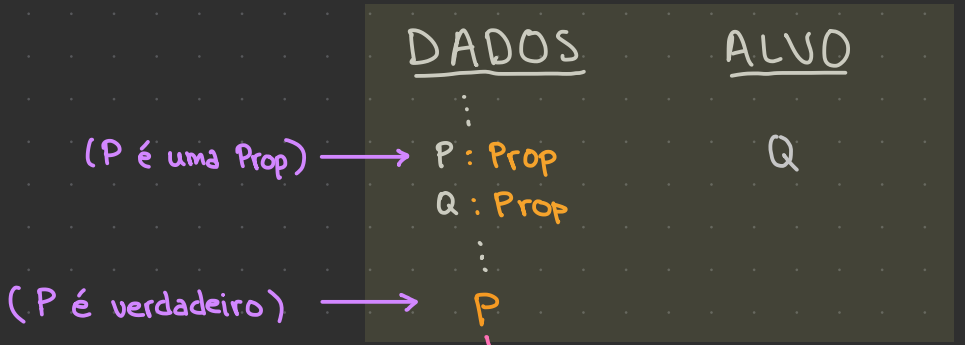


Concelho perigoso:

Dá a impressão que tem uma coisa proibida que não pode supor.

«não pode supor teu alvo»

A verdade: Só pode supor algo quando teu alvo é uma implicação $P \Rightarrow Q$ e até nesse caso a única coisa que pode supor é o P.



$P \Rightarrow Q$ é como uma promessa:

«Prometo Q se P.»

«Dado P, consigo Q.»

É uma promessa conditional:
eu não prometi Q se não tiver P

Igualdade vs Equivalência

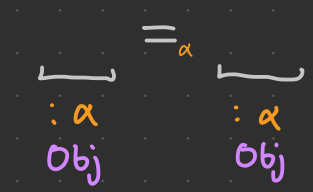
PALAVRINHAS

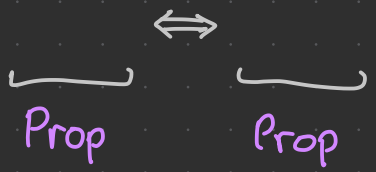
igualdade vs equivalência

intensão vs extensão

proposição vs comando

$\equiv$ / $\Leftrightarrow$ vs $\overset{\text{def}}{\equiv}$ / $\overset{\text{def}}{\Leftrightarrow}$





Exemplos:

"hi" $=_{String}$ "oi" ← uma proposição falsa → "hi" palíndromo ⇔ "oi" substring de "boi"

$42 =_{Int} 7 \cdot 6$ ← uma proposição verdadeira → 7 par ⇔ "hi" palindromo

"hi" $=_?$ 42 ← TYPE ERROR! 3 primo ⇔ 7

Intensão vs Extensão

Ismênia $=_{Person}$ a diretora do IMD

Ismênia $\not\equiv$ a diretora do IMD

↳ outra coisa ser a Ismênia, outra coisa ser a diretora do IMD!

$\{2\} =_{Set\ Int} \{$ os primos pares $\}$

$\{2\} \not\equiv \{$ os primos pares $\}$

O que significa $x \in \{2\}$?

Significa: $x = 2$

O que significa $x \in \{$ os primos pares $\}$?

Significa: x primo & x par.

Definições

Def. __ par : Int → Prop

Seja n : Int. : Cmd

~~Seja k : Int.~~

não!

n par ⇔ (∀k : Int)[n = 2k]

DADOS

n : Int

~~k : Int~~

Tem como uma definição de predicado ser errada?

Sim! Se discorda do nosso ♡:

- aceita algo que não deveria;
- não aceita algo que deveria.

3 par ⇔ (∀k : Int)[3 = 2k]

(concordou com o ♡ (tome k := 1 para refutar)

4 par ⇔ (∀k : Int)[4 = 2k]

(discordou (tome k := 1 para refutar)

Finalmente corrigimos assim:

$$n \text{ par} \overset{\text{def}}{\iff} (\exists k : \text{Int})[\, n = 2k \,] \quad : \text{Cmd}$$

agora sim!

Mais uma definição:

$$_^2 : \text{Int} \to \text{Int}$$

$$x^2 \overset{\text{def}}{\equiv} x \; x$$

Assim temos: $3^2 \equiv 3 \cdot 3 = 9$

Tem como uma def ser errada?

Sim! Novamente: se discorda do ♡

Desafios:

Seja x : __ . : Type → Cmd

Seja _ : Int . : Var → Cmd

Teoremas e demonstrações

Θ. Seja $x : \text{Int}$.

Suponha x par.

Demonstre: $(\forall y : \text{Int})[\ y \text{ par} \Rightarrow x \cdot y + 2 \text{ par}\ ]$

DEMONS.	DADOS	ALVO
Seja $w : \text{Int}$.	$x : \text{Int}$	~~$(\forall y : \text{Int})[\ \cdots\ ]$~~
Suponha w par.	$w : \text{Int}$	~~w par $\Rightarrow x \cdot w + 2$ par~~
⋮	x par	$x \cdot w + 2$ par
?	w par	$\Updownarrow$ $(\exists k : \text{Int})[\ x \cdot w + 2 = 2k\ ]$

Conectivos lógicos

	Como usar (-elim)	Como atacar (-intro)
$P \,\&\, Q$	?	Split.
P ou Q	?	Esc-L Esc-R.
$P \Rightarrow Q$	?	Suponha P.
$P \Leftrightarrow Q$	?	?
$\neg P$	?	?
$u =_\alpha v$	?	?
$(\forall x : \alpha)[\varphi(x)]$	?	Seja $_ : \alpha$.
$(\exists x : \alpha)[\varphi(x)]$	?	?
⋮		

ilusão de opção/escolha

de fato precisamos escolher (um nome válido e bom!)

?

Seja _ : _ .

quais nome são válidos?

DEMONS

Seja w : α .

DADOS

φ · ?

ALVO

$(\forall x : \alpha)[\varphi(x)]$

DADOS

φ · ?

w : α

ALVO

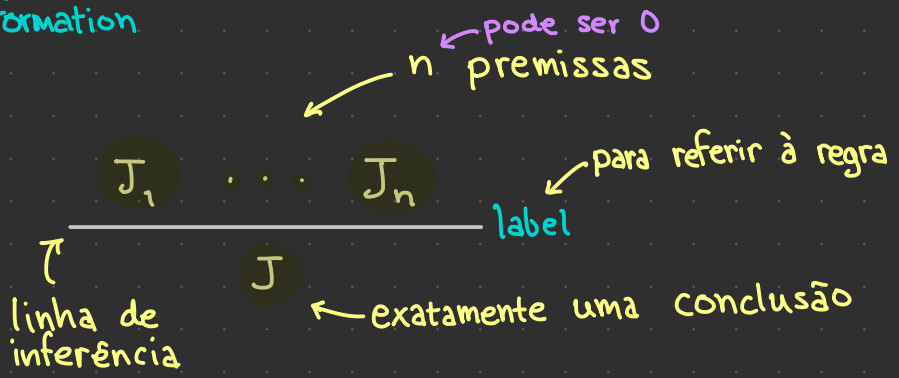
$\varphi(w)$

Regras de inferência

lec03
2025-08-22

$$\frac{\alpha : \text{Type} \qquad \beta : \text{Type}}{\alpha \to \beta : \text{Type}}\ (\to)\text{-Formation}$$

Forma geral das regras:



$$\frac{}{\text{Person} : \text{Type}}\ \text{Person-F} \qquad \frac{}{\text{Nat} : \text{Type}}\ \text{Nat-F} \qquad \ldots$$

Tuplas

$$\frac{\alpha : \text{Type} \qquad \beta : \text{Type}}{\alpha \times \beta \cdot \text{Type}}\ (\times)\text{-F}$$

Como **construo** habitantes (elementos) do $\alpha \times \beta$?

$$\frac{a : \alpha \qquad b : \beta}{\langle a, b \rangle : \alpha \times \beta}\ (\times)\text{-I}$$

Introdução

$\langle a, b \rangle$: tupla

Como **uso** habitantes do $\alpha \times \beta$?

Notações alternativas:

$w \cdot l$	outl w	π_1 w	proj_1 w	fst w
$w \cdot r$	outr w	π_2 w	proj_2 w	snd w

$$\frac{w : \alpha \times \beta}{w \cdot l : \alpha}\ (\times)\text{-E}_1$$

Eliminação

$w \cdot l$: projecção

E_1: Left

$$\frac{w : \alpha \times \beta}{w \cdot r : \beta}\ (\times)\text{-E}_2$$

E_2: Right

Exemplo

$$\dfrac{\dfrac{}{\text{Person} : \text{Type}}\,\text{Person-F} \qquad \dfrac{\dfrac{}{\text{Nat} : \text{Type}}\,\text{Nat-F} \qquad \dfrac{}{\text{String} : \text{Type}}\,\text{Str-F}}{\text{Nat} \times \text{String} : \text{Type}}\,(\times)\text{-F}}{\text{Person} \to (\text{Nat} \times \text{String}) : \text{Type}}\,(\to)\text{-F}$$

Abstraindo com buracos

João tem 3 filhos : Prop

João tem _ filhos : Nat → Prop

___ tem 3 filhos : Person → Prop

___ tem _ filhos : Person → (Nat → Prop)

ou.? (Person × Nat) → Prop

ou.? Nat → (Person → Prop)

Precedência sintáctica

O que significa $\alpha \times \beta \to \gamma$? Nada!

Tanto $(\alpha \times \beta) \to \gamma$ quanto $\alpha \times (\beta \to \gamma)$ tem significado.

aquela função que recebendo... retorna...

$$(\ w \mapsto \text{age}\ (w.l) + \text{length}\ (w.r)) : (\text{Person} \times \text{String}) \to \text{Nat}$$

$$\langle\ \text{João}\ ,\ \text{length}\ \rangle : \text{Person} \times (\text{String} \to \text{Nat})$$

Atribuimos ao $_ \times _$ precedência sintáctica maior que a do $_ \to _$.

A partir desta linha, o $\alpha \times \beta \to \gamma$ ganhou significado:

$$\alpha \times \beta \to \gamma \equiv (\alpha \times \beta) \to \gamma$$

Conectivos lógicos

lec04
2025-08-29

	Como usar (elim)	Como atacar (intro)
$P \,\&\, Q$	Ext-L P&Q. Ext-R P&Q.	Split.
P ou Q	Separe em casos a partir de P ou Q.	Esc-L Esc-R.
$P \Rightarrow Q$	App P⇒Q no P para obter Q.	Suponha P.
$\neg P$	?	?
$u =_\alpha v$	. . .	. . .
$(\forall x : \alpha)[\varphi(x)]$	App $(\forall x:\alpha)[\varphi(x)]$ no $a : \alpha$ para obter $\varphi(a)$.	Seja _ : α.
$(\exists x : \alpha)[\varphi(x)]$	Seja $a : \alpha$ tal que $\varphi(a)$.	Escolho _ : α. --ALVO: $\varphi(a)$

dualidade

inferível
$\varphi : \alpha \to \text{Prop}$

Conectivos definidos (ou açucar sintáctico)

$P \Leftrightarrow Q \overset{\text{def}}{\Longleftrightarrow} (P \Rightarrow Q) \;\&\; (Q \Rightarrow P)$

$P \text{ xou } Q \overset{\text{def}}{\Longleftrightarrow} (P \;\&\; (\neg Q)) \text{ ou } (Q \;\&\; (\neg P))$

$? \Uparrow \Downarrow ?$

$(P \text{ ou } Q) \;\&\; \neg(P \& Q)$

$P \Leftarrow Q \overset{\text{def}}{\Longleftrightarrow} Q \Rightarrow P$

$(\forall x \in A)[\, \varphi(x) \,]$

$(\forall x : \alpha)[\, x \in A \Rightarrow \varphi(x) \,]$

$(\exists x \in A)[\, \varphi(x) \,]$

$(\exists x)[\, x \in A \;\&\; \varphi(x) \,]$

$\exists_{\geq 1}$

$\exists_{\leq 1}$

$(\exists! x : \alpha)[\, \varphi(x) \,] \overset{\text{def}}{\Longleftrightarrow} (\exists x : \alpha)[\varphi(x)] \;\&\; (\forall u, v : \alpha)[\varphi(u) \& \varphi(v) \Rightarrow u = v]$

↑ existe único $x : \alpha$ tal que $\varphi(x)$

$(\forall x : \alpha \;\; \psi)[\, \varphi(x) \,] \overset{\text{def}}{\Longleftrightarrow} (\forall x : \alpha)[\, \psi(x) \Rightarrow \varphi(x) \,]$

↑ todo x legal çia

$(\exists x : \alpha \;\; \psi)[\, \varphi(x) \,] \overset{\text{def}}{\Longleftrightarrow} (\exists x : \alpha)[\, \psi(x) \;\&\; \varphi(x) \,]$

↑ existe x legal que çia

$\varphi : \alpha \to \text{Prop}$

$\psi : \alpha \to \text{Prop}$

Regras de inferência : F, I, E

$$\frac{P \Rightarrow Q \ \text{true} \qquad P \ \text{true}}{Q \ \text{true}} \ (\Rightarrow)\text{-E}$$

como usar

$$\frac{P \ \text{true} \qquad Q \ \text{true}}{P \,\&\, Q \ \text{true}} \ (\&)\text{-I}$$

como atacar

$$\frac{f : \alpha \to \beta \qquad a : \alpha}{f\ a : \beta} \ ?$$

$$\frac{P \ \text{prop} \qquad Q \ \text{prop}}{P \,\&\, Q \ \text{prop}} \ (\&)\text{-F}$$

como formar

$$\frac{P \ \text{prop} \qquad Q \ \text{prop}}{P \Rightarrow Q \ \text{prop}} \ (\Rightarrow)\text{-F}$$

. . . e os outros?

Conectivos definidos

lec05
2025-09-01

$$P \Leftrightarrow Q \overset{\text{def}}{\Longleftrightarrow} (P \Rightarrow Q) \;\&\; (Q \Rightarrow P)$$

$$P \text{ xou } Q \overset{\text{def}}{\Longleftrightarrow} (P \;\&\; (\neg Q)) \text{ ou } (Q \;\&\; (\neg P))$$

$$P \Leftarrow Q \overset{\text{def}}{\Longleftrightarrow} Q \Rightarrow P$$

$$(\exists! x:\alpha)[\varphi(x)] \overset{\text{def}}{\Longleftrightarrow} \underbrace{(\exists x:\alpha)[\varphi(x)]}_{\exists_{\geq 1}\ \text{(existência)}} \;\&\; \underbrace{(\forall u,v:\alpha)[\varphi(u)\;\&\;\varphi(v) \Rightarrow u=v]}_{\exists_{\leq 1}\ \text{(unicidade)}}$$

↑ existe único $x:\alpha$ tal que $\varphi(x)$

$\varphi : \alpha \to \text{Prop}$
$\psi : \alpha \to \text{Prop}$

$$(\forall x:\alpha\ \psi)[\varphi(x)] \overset{\text{def}}{\Longleftrightarrow} (\forall x:\alpha)[\psi(x) \Rightarrow \varphi(x)]$$

↑ todo x legal ção

$$(\exists x:\alpha\ \psi)[\varphi(x)] \overset{\text{def}}{\Longleftrightarrow} (\exists x:\alpha)[\psi(x)\;\&\;\varphi(x)]$$

↑ existe x legal que ção

Casos especiais onde $\psi(x) :\Leftrightarrow x \in A$

$$(\forall x \in A)[\varphi(x)] \overset{\text{Sug}}{\Leftrightarrow} (\forall x : \alpha)[x \in A \Rightarrow \varphi(x)]$$

$$(\exists x \in A)[\varphi(x)] \overset{\text{Sug}}{\Leftrightarrow} (\exists x : \alpha)[x \in A \;\&\; \varphi(x)]$$

$$x \not R y \overset{\text{Sug}}{\Leftrightarrow} \neg(x R y)$$

$$(\forall x : \alpha\ \psi)[\varphi(x)] \overset{\text{def}}{\Leftrightarrow} (\forall x : \alpha)[\psi(x) \Rightarrow \varphi(x)]$$

todo x legal φia

Seja $x : \alpha$ t.q. $\psi(x)$. { Seja $x : \alpha$. Suponha $\psi(x)$.

ambíguo: é um $\forall, \Rightarrow$ - ataque na $(\forall x : \alpha\ \psi)[\varphi(x)]$

ou um $\exists$ - uso de $(\exists x : \alpha)[\psi(x)]$?

Conectivos lógicos

	Como usar (elim)	Como atacar (intro)
$P \,\&\, Q$	Ext-L $P\&Q$. Ext-R $P\&Q$.	Split.
P ou Q	Separe em casos a partir de P ou Q.	Esc-L Esc-R.
$P \Rightarrow Q$	App $P \Rightarrow Q$ no P para obter Q.	Suponha P.
$\bot$ ← bottom	Contradição. ↖ EX FALSO QUODLIBET	(nada)
$u =_\alpha v$	. . .	. . .
$(\forall x : \alpha)[\,\varphi(x)\,]$	App $(\forall x:\alpha)[\,\varphi(x)\,]$ no $a : \alpha$ para obter $\varphi(a)$.	Seja $_ : \alpha$.
$(\exists x : \alpha)[\,\varphi(x)\,]$	Seja $a : \alpha$ tal que $\varphi(a)$.	Escolho $_ : \alpha$. --ALVO: $\varphi(a)$

dualidade

inferível

$\varphi : \alpha \to \mathrm{Prop}$

Negação

$\neg P \overset{\text{def}}{\Longleftrightarrow} P \Rightarrow \bot$

« refutar P » significa « demonstrar $\neg P$ »

"leis de lógica": comutatividade de (&)

Sejam P, Q : Prop.
Demonstre: P & Q ⇒ Q & P.

DEMONS

Suponha pq : P & Q. (x) pq

Ext-R (x) pq para obter Q. (z) q

q : Q

Split

Parte-L:

Imediato (z) q.

DADOS	ALVO
P : Prop	Q
Q : Prop	
pq : P & Q	
q : Q	

Parte-R:

Ext-L (x) pq para obter P. (3) P

Imediato (3) P.

□

DADOS	ALVO
P : Prop	P
Q : Prop	
pq : P & Q	
q : Q	
p : P	

DADOS	ALVO
P : Prop	~~(P & Q)~~ ⇒ (Q & P)
Q : Prop	~~Q & P~~
pq : P & Q	
q : Q	

Dupla negação

lec06
2025-09-03

$P : Prop \vdash P \Rightarrow \neg\neg P$

DEMONS

Suponha P.

Suponha $\neg P$.

App $\neg P$ no P para obter $\bot$.
$P \Rightarrow \bot$

Contradição. ∎

DADOS

P

$\neg P \Leftrightarrow P \Rightarrow \bot$

$\bot$

ALVO

~~$P \Rightarrow \neg\neg P$~~

$\neg\neg P$

$\Updownarrow$

$\neg P \Rightarrow \bot$

$\Updownarrow$

$(P \Rightarrow \bot) \Rightarrow \bot$

$\bot$

$\vdash P \Leftarrow \neg\neg P$

DEMONS

Suponha $\neg\neg P$.

Vou demonstrar: $\neg P$

DADOS	ALVO
$\neg\neg P$	~~$\neg P$~~
P	$\bot$

Sup P.

(Devemos desistir!)

$\neg P \Rightarrow \bot$

App $\neg\neg P$ no $\neg P$.

Contradição.

DADOS	ALVO
$\neg\neg P$	$P \Leftarrow \neg\neg P$
$\Updownarrow$	P
$\neg P \Rightarrow \bot$	
$\neg P$	

LEM ↶ magia ✡

LEM : $(\forall P : \mathrm{Prop})[\, P \vee \neg P \,]$

LEM[P] : $P \vee \neg P$

Com palavras:

« Para qualquer proposição P, temos $P \vee \neg P$. »

Como esquema de regra de inferência:

$$\frac{}{P \vee \neg P}\ \mathrm{LEM}[P]$$

Como comando:

✡ LEM P . -- DADOS: ..., $P \vee \neg P$

✡ - contaminação contagiosa!

$\vdash \neg\neg(P \vee \neg P)$ A irrefutabilidade do LEM[P]

DEMONS

Sup $\neg(P \vee \neg P)$. -- ALVO: $\bot$

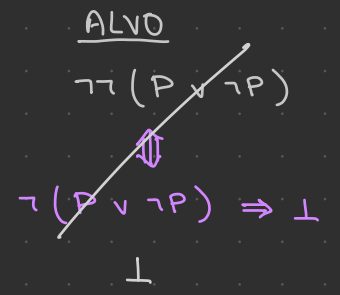
Vou demons: $P \vee \neg P$

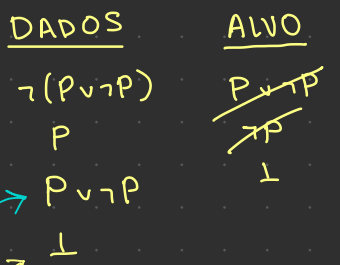
| Esc-R.
| Sup P.
| Vou demons: $P \vee \neg P$
| | Esc-L.
| | Imediato.
| -- DADOS: ..., $P \vee \neg P$
| App $\neg(P \vee \neg P)$ no $P \vee \neg P$.
| Contradição.

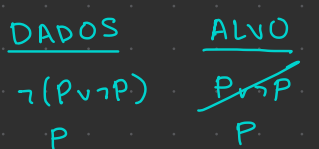
-- DADOS: ..., $P \vee \neg P$

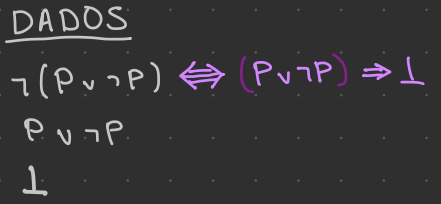
App $\neg(P \vee \neg P)$ no $P \vee \neg P$.

Contradição. ∎









Primeiro contato com Lean. (Veja gravação.)

lec07

2025-09-15