Capítulo 5: Raciocínio Clássico

Raciocínio Clássico introduz uma quantidade de princípios na lógica, que facilitam a maneira como pensamos e construímos nossas provas".

- Se tomarmos todas as regras da lógica proposicional que vimos até agora e excluímos reductio ad absurdum, temos o que é conhecido como lógica intuicionista.
- Na lógica intuicionista, é possível visualizar provas em termos computacionais: uma prova de $A \wedge B$ é uma prova de A junto com uma prova de B. Uma prova de $A \rightarrow B$ é um procedimento que transforma a evidência de A em evidência de B , e uma prova de $A \vee B$ é uma prova de um ou outro, indicando qual dos dois termos é o caso.
- A regra ex falsum só "faz sentido" porque esperamos que não exista uma prova do falso.

- **5.1. Prova por Absurdo**
- Na dedução natural, a prova por contradição é expressa da seguinte maneira:

$$\dfrac{\dfrac{\overline{\neg A}^{\,1}}{\vdots}}{\dfrac{\bot}{A}\,1}$$

- Em LEAN, esta inferência recebe o nome de by_contradiction e, como faz parte das regras clássicas, devemos utilizar o commando open classical para torná-lo utilizável.

```
open classical

variable (A : Prop)

example : A :=
by_contradiction
  (assume h : ¬ A,
    show false, from sorry)
```

- Uma das consequências mais importantes da prova por absurdo é a **Lei do Terceiro Excluído**. Para resolvermos certos provas, precisaremos dividí-la em dois casos, assumindo primeiro A e depois ¬A. Isso equivale ao usarmos as regras de eliminação da disjunção em A v ¬ A para obtê-los. Aqui está uma prova de como obtemos A v ¬ A, em dedução natural, através da prova por contradição:

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\overline{\neg(A \lor \neg A)}^{\,2} \qquad \dfrac{\overline{A}^{\,1}}{A \lor \neg A}}{\bot}}{\neg A}{}^{1}}{A \lor \neg A} \qquad \overline{\neg(A \lor \neg A)}^{\,1}}{\bot}}{A \lor \neg A}{}^{1}$$

- Aqui está a mesma prova, montada no LEAN:

```
open classical

variable (A : Prop)

example : A ∨ ¬ A :=
by_contradiction
  (assume h1 : ¬ (A ∨ ¬ A),
    have h2 : ¬ A, from
      assume h3 : A,
      have h4 : A ∨ ¬ A, from or.inl h3,
      show false, from h1 h4,
    have h5 : A ∨ ¬ A, from or.inr h2,
    show false, from h1 h5)
```

- O princípio é conhecido como Lei do Terceiro Excluído porque diz que uma proposição A é verdadeira ou falsa; Não há um possível terceiro estado. Este teorema é chamado de **em** na biblioteca Lean. Para qualquer proposição A, "em A" denota uma prova de A v ¬ A.

```
open classical
example (A : Prop) : A ∨ ¬ A :=
em A

variable A : Prop
#check em A
```

```
[Lean] em A : A ∨ ¬A (6, 1)
```

- A prova por contradição também é equivalente ao princípio ¬¬A↔A. A implicação da direita para a esquerda é intuitiva; A outra implicação é clássica, e é conhecida como eliminação de dupla negação. Aqui está uma prova em dedução natural:

$$\dfrac{\begin{array}{c}\overline{A}^{\,1}\\ \vdots \\ B\end{array} \qquad \begin{array}{c}\overline{B}^{\,1}\\ \vdots \\ A\end{array}}{A \leftrightarrow B}\;1\ \leftrightarrow\mathrm{I}$$

$$\dfrac{\dfrac{\dfrac{\overline{\neg\neg A}^{\,2} \qquad \overline{\neg A}^{\,1}}{\bot}}{A}\,1 \qquad \dfrac{\dfrac{\overline{\neg A}^{\,1} \qquad \overline{A}^{\,2}}{\bot}}{\neg\neg A}\,1}{\neg\neg A \leftrightarrow A}\,2$$

- E aqui está a prova em LEAN:

```
open classical

example (A : Prop) : ¬ ¬ A ↔ A :=
iff.intro
  (assume h1 : ¬ ¬ A,
    show A, from by_contradiction
      (assume h2 : ¬ A,
        show false, from h1 h2))
  (assume h1 : A,
    show ¬ ¬ A, from assume h2 : ¬ A, h2 h1)
```

- Às vezes, a prova por contradição é necessária, mas, nem sempre é a única maneira de se montar uma prova. Ela pode ser menos informativa do que a prova direta.
- Suponhamos, por exemplo, que queremos provar $A \wedge B \wedge C \rightarrow D$. Em uma prova direta, assumimos A, B e C, e tentar chegar a D. Ao longo do caminho, derivaremos outras conseqüências de A, B e C, e isso pode ser útil em outros contextos. Se usarmos a prova por contradição, por outro lado, assumimos A, B, C e $\neg D$, e tentamos provar $\perp$.
- Nesse caso, quaisquer resultados auxiliares que possamos obter dessa maneira não serão utilizados fora da prova por absurdo pelo fato de que, no final das contas, $\perp$ é a nossa conclusão de todas as hipóteses A, B, C e $\neg D$.

- **5.2. Alguns Princípios Clássicos**

- Seja A → B uma implicação qualquer. Chamamos ¬B → ¬A de **contrapositiva** de A → B. Toda implicação implica sua contrapositiva, e a outra direção é classicamente verdadeira. Aqui está a prova:

$$\dfrac{\dfrac{\dfrac{\dfrac{\neg B \to \neg A \qquad \dfrac{}{\neg B}\,{}^{1}}{\neg A} \qquad \dfrac{}{A}\,{}^{2}}{\bot}}{B}\,{}^{1}}{A \to B}\,{}^{2}$$

- Afirmar que "se A então B" é equivalente a dizer que não pode ser o caso que A é verdadeiro e B é falso.

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{\neg(A\land\neg B)}{}^{3} \qquad \dfrac{\dfrac{}{A}{}^{2} \qquad \dfrac{}{\neg B}{}^{1}}{A\land\neg B}}{\bot}}{B}{}^{1}}{A\to B}{}^{2}}{\neg(A\land\neg B)\to(A\to B)}{}^{3}$$

- Aqui está a demonstração das provas em LEAN dos exercícios anteriores:

```
open classical

variables (A B : Prop)

example (h : ¬ B → ¬ A) : A → B :=
assume h1 : A,
show B, from
  by_contradiction
    (assume h2 : ¬ B,
      have h3 : ¬ A, from h h2,
      show false, from h3 h1)

example (h : ¬ (A ∧ ¬ B)) : A → B :=
assume : A,
show B, from
  by_contradiction
    (assume : ¬ B,
      have A ∧ ¬ B, from and.intro ‹A› this,
      show false, from h this)
```

- Usando o princípios clássicos, podemos reescrever a implicação em termos da disjunção e da negação:

$$(A \to B) \leftrightarrow \neg A \vee B$$

- As seguintes equivalências são conhecidas como leis de De Morgan:

$$\neg(A \vee B) \leftrightarrow \neg A \wedge \neg B$$
$$\neg(A \wedge B) \leftrightarrow \neg A \vee \neg B$$

- A direção direta do segundo deles requer um raciocínio clássico(?).

- Uando essas identidades, sempre desmembrar as nossas fórmulas, tornando as mais “simples” colocando as negações juntas das proposições. Por exemplo:

$$\begin{aligned}\neg(\neg A \wedge B \to C) &\leftrightarrow \neg(\neg(\neg A \wedge B) \vee C)\\ &\leftrightarrow \neg\neg(\neg A \wedge B) \wedge \neg C\\ &\leftrightarrow \neg A \wedge B \wedge \neg C\end{aligned}$$

- Uma fórmula construída a partir de ∧, ∨ e ¬ em que as negações só ocorrem em variáveis é dito estar na **forma negativa normal**.

- Usando as leis de distribuição, pode-se garantir que todas as disjunções sejam externas, de modo que a fórmula sejam um grande "or" de "ands" de variáveis proposicionais e variáveis proposicionais negadas. Uma fórmula escrita desta maneira é dita estar em **forma normal disjuntiva**.

- Alternativamente, todos os "ands" podem ser trazidos para fora. Essa fórmula é dita estar em **forma conjuntiva normal**.

- Escrever nossas fórmulas utilizando uma dessas representações pode se tornar muito extensivo. Veja só:
- Colocar $(A \vee B) \wedge (C \vee D) \wedge (E \vee F)$ em forma normal disjuntiva.

Off-Topic – Circuitos Digitais

- Existem duas formas de escrever expressões booleanas:

  – **Soma de produtos** $\Rightarrow$ $\boldsymbol{A.B + \overline{A}.C + B.\overline{C}}$

  – Produto de somas $\Rightarrow$ $(M+N).(\overline{P+Q}).(R+\overline{W})$

- Construa a tabela verdade e projete o circuito:

  $$x = \overline{A}.B + A\overline{B}$$ Soma de Produtos

A	B	X	
0	0	0	
0	1	1	$\overline{A}.B$
1	0	1	$A.\overline{B}$
1	1	0