

# Implementação de uma biblioteca da Lógica de Inconsistência Formal LFI1 em Coq

Helena Vargas Tannuri

Universidade do Estado de Santa Catarina  
helenavargastannuri@gmail.com

Orientadora: Dra Karina Girardi Roggia

Coorientador: Miguel Alfredo Nunes

19/11/2024

# Sumário

- 1 Introdução
- 2 Objetivos
- 3 Trabalhos Relacionados
- 4 Lógica Modal
  - Lógicas Multimodais
- 5 Fusão de Lógicas Modais
- 6 Coq
- 7 Implementação
- 8 Conclusões
- 9 Referências

- Lógicas paraconsistentes são sistemas não-clássicos que separam a trivialidade da contradição;
  - Usualmente, lógicas ortodoxas assumem que toda teoria contraditória é uma teoria trivial, ou seja, uma teoria com todas as fórmulas.
- Lógica não-clássica é qualquer lógica que quebra algum dos princípios da lógica clássica;
  - As lógicas paraconsistentes quebram o princípio da explosão (definido como  $\alpha \rightarrow (\neg\alpha \rightarrow \beta)$ ) (CARNIELLI; CONIGLIO; MARCOS, 2007).
- Lógicas de inconsistência formal (**LFIs**) são lógicas paraconsistentes que resgatam de maneira *controlada* o princípio da explosão, internalizando o conceito de inconsistência dentro da sua linguagem (CARNIELLI; CONIGLIO, 2016).

- Lógicas paraconsistentes possuem diversas aplicações em diferentes campos do conhecimento;

- Assistentes de provas são softwares para desenvolvimento de provas formais;
- Coq é um assistente de provas com grande disponibilidade de materiais didáticos e diversas ferramentas que facilitam no desenvolvimento de provas (??);
- Fusões de lógicas é um tópico complexo, porém as dificuldades referentes a isso podem ser amenizadas com o uso de assistentes de provas como o Coq.
- Continuação do que foi desenvolvido em (SILVEIRA, 2020) e (??).

Modelar, no assistente Coq, sistemas de lógicas multimodais resultantes da fusão de lógicas modais mais simples, preservando propriedades das lógicas combinadas.

# Objetivos Específicos

- 1 Estudar os principais conceitos de combinações de lógicas, em especial, a fusão;
- 2 Realizar um estudo de caso de fusões de lógicas modais no Coq;
- 3 Modelar, de forma paramétrica, sistemas de lógicas multimodais resultantes de fusão de lógicas modais em Coq.

- 1 ?? (??) - apresenta fusões de lógicas modais em Cálculo- $\lambda$  Simplesmente Tipado e modelagem de lógicas combinadas em diversos assistentes de provas;
- 2 ?? (??) - descrevem lógicas modais resultantes de combinação em Isabelle;
- 3 ?? (??) - descrevem uma lógica modal resultante de fusão em Coq;
- 4 ?? (??) - modela sistemas lógicos e combinações de lógicas em uma linguagem baseada em teoria de tipos chamada  $M_{MT}$ .



Menor conjunto LM que respeita:

- ❶  $\top, \perp \in \text{LM}$
- ❷  $\mathbb{P} \subseteq \text{LM}$
- ❸ Se  $\varphi \in \text{LM}$ , então  $\circ \varphi \in \text{LM}$ , sendo  $\circ \in \{\Box, \Diamond, \neg\}$
- ❹ Se  $\varphi, \psi \in \text{LM}$ , então  $\varphi \circ \psi \in \text{LM}$ , sendo  $\circ \in \{\wedge, \vee, \rightarrow\}$

## Frames

Um frame é um par  $\mathcal{F} = \langle \mathcal{W}, \mathcal{R} \rangle$ , onde  $\mathcal{W} \neq \emptyset$  e  $\mathcal{R} \subseteq \mathcal{W} \times \mathcal{W}$ .

## Modelos

Modelos são pares de frames e funções de valoração da forma  $\mathcal{M} = \langle \mathcal{F}, \mathcal{V} \rangle$ , onde  $\mathcal{V}$  é uma função total binária definida por  $\mathcal{V} : \mathbb{P} \rightarrow 2^{\mathcal{W}}$

Axiomatização de Hilbert, 10 axiomas proposicionais:

$$p_0 \rightarrow (p_1 \rightarrow p_0) \quad (\text{Ax1})$$

$$(p_0 \rightarrow (p_1 \rightarrow p_2)) \rightarrow ((p_0 \rightarrow p_1) \rightarrow (p_0 \rightarrow p_2)) \quad (\text{Ax2})$$

$$(\neg p_1 \rightarrow \neg p_0) \rightarrow (p_0 \rightarrow p_1) \quad (\text{Ax3})$$

$$p_0 \rightarrow (p_1 \rightarrow (p_0 \wedge p_1)) \quad (\text{Ax4})$$

$$(p_0 \wedge p_1) \rightarrow p_0 \quad (\text{Ax5})$$

$$(p_0 \wedge p_1) \rightarrow p_1 \quad (\text{Ax6})$$

$$p_0 \rightarrow (p_0 \vee p_1) \quad (\text{Ax7})$$

$$p_1 \rightarrow (p_0 \vee p_1) \quad (\text{Ax8})$$

$$(p_0 \rightarrow p_2) \rightarrow ((p_1 \rightarrow p_2) \rightarrow (p_0 \vee p_1) \rightarrow p_2) \quad (\text{Ax9})$$

$$\neg\neg p_0 \rightarrow p_0 \quad (\text{Ax10})$$

Axiomatização de Hilbert, 2 axiomas modais:

$$\Box(p_0 \rightarrow p_1) \rightarrow (\Box p_0 \rightarrow \Box p_1) \quad (\text{K})$$

$$\Diamond(p_0 \vee p_1) \rightarrow (\Diamond p_0 \vee \Diamond p_1) \quad (\text{Possibilidade})$$

E regras de Necessitação e Modus Ponens:

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \Box \varphi} \text{ Nec}$$

$$\frac{\Gamma \vdash \varphi \rightarrow \psi \quad \Gamma \vdash \varphi}{\Gamma \vdash \psi} \text{ MP}$$

## Corretude Forte

Se há uma derivação sintática para uma fórmula a partir de uma teoria, então há uma derivação semântica em algum frame/modelo a partir da mesma teoria:  $\Gamma \vdash \varphi \Rightarrow \Gamma \models \varphi$

## Completude Forte

Se há uma derivação semântica para uma fórmula em algum frame/modelo a partir de uma teoria, então há uma derivação sintática a partir da mesma teoria:  $\Gamma \models \varphi \Rightarrow \Gamma \vdash \varphi$

Extensão do conceito de lógica modais com apenas uma (ou um par de) modalidade(s) que contém diversas modalidades. A linguagem de uma lógica multimodal é o menor conjunto  $LM_n$  que respeita:

- 1  $\top, \perp \in LM_n$
- 2  $\mathbb{P} \subseteq LM_n$
- 3 Se  $\varphi \in LM_n$ , então  $\circ \varphi \in LM_n$ , sendo  $\circ \in \{\Box_1, \dots, \Box_n, \Diamond_1, \dots, \Diamond_n, \neg\}$
- 4 Se  $\varphi, \psi \in LM_n$ , então  $\varphi \circ \psi \in LM_n$ , sendo  $\circ \in \{\wedge, \vee, \rightarrow\}$

## n-frames

Um n-frame é uma tupla  $\mathcal{F}_n = \langle \mathcal{W}, \mathcal{R}_1, \dots, \mathcal{R}_n \rangle$ , onde  $\mathcal{W} \neq \emptyset$  e  $\mathcal{R}_i \subseteq \mathcal{W} \times \mathcal{W}, 1 \leq i \leq n$ .

Para cada  $\Box_i/\Diamond_i$  tem-se:

$$\Box_i(p_0 \rightarrow p_1) \rightarrow (\Box_i p_0 \rightarrow \Box_i p_1)$$

$$\Diamond_i(p_0 \vee p_1) \rightarrow (\Diamond_i p_0 \vee \Diamond_i p_1)$$

Regra de Necessitação é substituída por:

$$\frac{\Gamma \vdash \varphi}{\Gamma \vdash \Box_i \varphi} \text{Nec}_i$$

# Fusão de Lógicas Modais

- Método para combinar sistemas lógicos distintos original desenvolvido por (??);
- Possibilita especificar sistemas lógicos com múltiplas modalidades distintas, dados dois sistemas lógicos mais simples;
- A fusão de lógicas preserva corretude e completude - sendo  $\mathcal{L}_1$  e  $\mathcal{L}_2$  lógicas corretas/completas uma lógica  $\mathcal{L}_3$  resultante da sua fusão também será correta/completa;
- Preservação de corretude e completude foi originalmente demonstrada por (??), prova se resume a construir modelos para  $\mathcal{L}_3$  a partir de certos modelos para  $\mathcal{L}_1$  e  $\mathcal{L}_2$ .



- Sendo  $\mathcal{L}_1$  e  $\mathcal{L}_2$  lógicas modais:
  - Linguagens são  $LM_1$  e  $LM_2$ ;
  - Correspondentes as classes  $\mathfrak{F}_1$  e  $\mathfrak{F}_2$
  - Axiomatizadas pelos conjuntos de axiomas  $\Sigma_1$  e  $\Sigma_2$  com regras  $Nec_1$  e  $Nec_2$  respectivamente.
- A lógica  $\mathcal{L}_3 = \mathcal{L}_1 \odot \mathcal{L}_2$ :
  - Terá linguagem  $LM_3 = LM_1 \cup LM_2$ ;
  - Será correspondente à classe  $\mathfrak{F}_3$  de 2-frames da forma  $\langle \mathcal{W}, \mathcal{R}, \mathcal{S} \rangle$ , onde  $\langle \mathcal{W}, \mathcal{R} \rangle \in \mathfrak{F}_1$  e  $\langle \mathcal{W}, \mathcal{S} \rangle \in \mathfrak{F}_2$
  - Será axiomatizada por  $\Sigma_3 = \Sigma_1 \oplus \Sigma_2$  com regras  $Nec_1$  e  $Nec_2$ .

- Assistente de provas para lógica de alta ordem, capaz de descrever e raciocinar sobre objetos matemáticos (GEUVERS, 2009);
- O Coq é baseado em teoria de tipos, devido a Correspondência de Curry-Howard é capaz de expressar sistemas lógicos sofisticados;
- Coq tem uma grande quantidade de ferramentas de automação de provas, e também permite que seus usuários desenvolvam suas próprias ferramentas;
- Essas características tornam Coq uma boa ferramenta para representar sistemas lógicos complexos e operar sobre eles.

- A biblioteca de lógica modal desenvolvida em (SILVEIRA, 2020) e (??) serviu de base para a implementação;
- Foi provada a corretude do sistema  $\mathbf{KT} \odot \mathbf{K4}$ ;
- Foi modelada, de forma paramétrica, a fusão de sistemas sintáticos de lógicas modais;
- Foi modelada a semântica de sistemas multimodais.

```
Definition relative_soundness (A: axiom -> Prop) (R: Frame -> Prop) :=  
  forall  $\Gamma$   $\varphi$ , (A;  $\Gamma \Vdash \varphi$ ) -> forall F V, R F -> entails (Build_Model F V)  $\Gamma \varphi$ .  
  
Definition relative_KT4soundness (A: KT4axiom -> Prop) (R: KT4Frame -> Prop) :=  
  forall  $\Gamma$   $\varphi$ , (A;  $\Gamma \Vdash_{t4} \varphi$ ) -> forall F V, R F -> KT4entails (Build_KT4Model F V)  $\Gamma \varphi$ .  
  
(* Proving that this definition of soundness is correct*)  
Lemma relative_soundness_correct:  
  relative_soundness K anyFrame <-> (forall (G: theory) ( $\varphi$ : formula), (K; G  $\Vdash \varphi$ ) -> (G  $\models \varphi$ )).
```

```
Theorem KT4_soundness:  
  relative_KT4soundness KT4Ax anyKT4Frame.
```

Figura: Corretude de  $\mathbf{KT} \odot \mathbf{K4}$

```
Inductive MMformula: Set :=  
  | MMLit      : nat          -> MMformula  
  | MMNeg      : MMformula    -> MMformula  
  | MMBox      : nat (*index*) -> MMformula -> MMformula  
  | MMDia      : nat (*index*) -> MMformula -> MMformula  
  | MMAnd      : MMformula    -> MMformula -> MMformula  
  | MMOr       : MMformula    -> MMformula -> MMformula  
  | MMImplies : MMformula    -> MMformula -> MMformula.
```

Figura: Linguagem do Sistema Multimodal

```
Section MultiModal.
```

```
Variable Modalities: nat.
```

```
Hypothesis minimum_modalities: Modalities > 1.
```

Figura: Definição da Variável do Número de Modalidades no Sistema

# Implementação

```
Fixpoint deducible_formula ( $\varphi$ : MMformula): Prop :=
  match  $\varphi$  with
  | MMLit _           => True
  | MMNeg  $\psi$          => deducible_formula  $\psi$ 
  | MMBox  $x \ \psi$  | MMDia  $x \ \psi$       =>  $x < \text{Modalities}$  /\ deducible_formula  $\psi$ 
  | MMAnd  $\psi \ \gamma$  | MMOr  $\psi \ \gamma$  | MMImplies  $\psi \ \gamma$  => deducible_formula  $\psi$  /\ deducible_formula  $\gamma$ 
  end.

Fixpoint deducible_theory ( $\Gamma$ : MMtheory): Prop :=
  match  $\Gamma$  with
  | [ ]      => True
  |  $h :: t$  => (deducible_formula  $h$ ) /\ (deducible_theory  $t$ )
  end.
```

Figura: Definição de Dedutibilidade de Fórmulas e Teorias

```
Record nFrame: Type := {  
  nW      : Set;  
  nR      : list (nW -> nW -> Prop);  
  rel_cond: (length nR) = Modalities;  
}.  
  
Record nModel: Type := {  
  nF: nFrame;  
  nV: nat -> (nW nF) -> Prop  
}.
```

Figura: Frames e Modelos Multimodais



```
Definition split_frame (F: nFrame): list Frame:=  
  match F with  
  | Build_nFrame nW nR _ => map (Build_Frame nW) nR  
  end.
```

```
Definition nFrame_to_Frame (F: nFrame) (index: nat): Frame :=  
  match F with  
  | Build_nFrame nW nR _ => Build_Frame nW (get_rel nW nR index)  
  end.
```

Figura: Função de Tradução de Frames Multimodais para Frames Monomodais

```
Definition nModel_to_Model (M: nModel) (index: nat): Model :=  
  match M with  
  | Build_nModel (Build_nFrame nW nR _) V => Build_Model  
    [Build_Frame nW (get_rel nW nR index)] V  
  end.
```

Figura: Função de Tradução de Modelos Multimodais para Modelos Monomodais

```
Theorem MMvaluation_generalization: forall modalities W lR RC V  $\varphi$  n,  
  let NF := Build_nFrame modalities W lR RC in  
  let NM := Build_nModel modalities NF V in  
  let M := (nModel_to_Model modalities NM n) in  
  deducible_formula modalities (formula_to_MMformula  $\varphi$  n) ->  
    forall w, fun_validation M w  $\varphi$  <->  
      valuation modalities NM w (formula_to_MMformula  $\varphi$  n).
```

Figura: Extensão da Valoração

# Implementação

```
Inductive Kn (index: list nat): MMaxiom -> Prop :=
| Kn_ax1: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax1  $\varphi \psi$ ))) -> Kn index (MMax1  $\varphi \psi$ )
| Kn_ax2: forall  $\varphi \psi \gamma$ , (deducible_formula (MMinstance (MMax2  $\varphi \psi \gamma$ ))) -> Kn index (MMax2  $\varphi \psi \gamma$ )
| Kn_ax3: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax3  $\varphi \psi$ ))) -> Kn index (MMax3  $\varphi \psi$ )
| Kn_ax4: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax4  $\varphi \psi$ ))) -> Kn index (MMax4  $\varphi \psi$ )
| Kn_ax5: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax5  $\varphi \psi$ ))) -> Kn index (MMax5  $\varphi \psi$ )
| Kn_ax6: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax6  $\varphi \psi$ ))) -> Kn index (MMax6  $\varphi \psi$ )
| Kn_ax7: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax7  $\varphi \psi$ ))) -> Kn index (MMax7  $\varphi \psi$ )
| Kn_ax8: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax8  $\varphi \psi$ ))) -> Kn index (MMax8  $\varphi \psi$ )
| Kn_ax9: forall  $\varphi \psi \gamma$ , (deducible_formula (MMinstance (MMax9  $\varphi \psi \gamma$ ))) -> Kn index (MMax9  $\varphi \psi \gamma$ )
| Kn_ax10: forall  $\varphi \psi$ , (deducible_formula (MMinstance (MMax10  $\varphi \psi$ ))) -> Kn index (MMax10  $\varphi \psi$ )
| Kn_axK: forall i  $\varphi \psi$ , In i index ->
    (deducible_formula (MMinstance (MMaxK i  $\varphi \psi$ ))) -> Kn index (MMaxK i  $\varphi \psi$ )
| Kn_axPos: forall i  $\varphi \psi$ , In i index ->
    (deducible_formula (MMinstance (MMaxPos i  $\varphi \psi$ ))) -> Kn index (MMaxPos i  $\varphi \psi$ )
```

Figura: Axiomatização Multimodal do Sistema K

# Implementação

```
Inductive join (S1 S2: axiom -> Prop) (index1 index2: nat): list nat -> MMaxiom -> Prop :=  
| derivable_S1: forall a b, S1 a -> index1 <> index2 -> axiom_to_MMaxiom index1 a b ->  
  deducible_formula (MMinstance b) -> join S1 S2 index1 index2 (index1 :: index2 :: nil) b  
| derivable_S2: forall a b, S2 a -> index1 <> index2 -> axiom_to_MMaxiom index2 a b ->  
  deducible_formula (MMinstance b) -> join S1 S2 index1 index2 (index1 :: index2 :: nil) b.
```

```
Inductive join_one (S1: axiom -> Prop) (S2: list nat -> MMaxiom -> Prop) (index1: nat)·  
  (index2: list nat): list nat -> MMaxiom -> Prop :=  
| derivable_S1_one: forall a b, S1 a -> ~ In index1 index2 -> axiom_to_MMaxiom index1 a b ->  
  deducible_formula (MMinstance b) -> join_one S1 S2 index1 index2 (index1 :: index2) b  
| derivable_S2_one: forall a, S2 index2 a -> ~ In index1 index2 ->  
  join_one S1 S2 index1 index2 (index1 :: index2) a.
```

```
Inductive join_two (S1 S2: list nat -> MMaxiom -> Prop) (index1 index2: list nat):·  
  list nat -> MMaxiom -> Prop :=  
| derivable_S1_two: forall a, S1 index1 a -> ~ share_element index1 index2 ->·  
  join_two S1 S2 index1 index2 (index1 ++ index2) a  
| derivable_S2_two: forall a, S2 index2 a -> ~ share_element index1 index2 ->·  
  join_two S1 S2 index1 index2 (index1 ++ index2) a.
```

Figura: Definição de Fusão de Sistemas Sintáticos

# Implementação

```
Reserved Notation "A ; G |--M p" (at level 110, no associativity).
Inductive MMdeduction (A: MMaxiom -> Prop): MMtheory -> MMformula -> Prop :=
  (* Premise. *)
  | MMPrem: forall (Γ: MMtheory) (φ: MMformula) (i: nat),
    (deducible_theory Γ) -> (nth_error Γ i = Some φ) -> A; Γ |--M φ
  (* Axiom. *)
  | MMAx: forall (Γ: MMtheory) (a: MMaxiom) (φ: MMformula),
    A a -> (deducible_formula φ) -> MMinstance a = φ ->
    (deducible_theory Γ) -> A; Γ |--M φ
  (* Modus Ponens. *)
  | MMMp: forall (Γ: MMtheory) (φ ψ: MMformula),
    (deducible_formula <!φ -> ψ!>) -> (deducible_theory Γ) ->
    (A; Γ |--M (<!φ -> ψ!>)) -> (A; Γ |--M φ) -> (A; Γ |--M ψ)
  (* Necessitation. *)
  | MMNec: forall (Γ: MMtheory) (φ: MMformula) (i: nat),
    i < Modalities -> (deducible_formula φ) -> (deducible_theory Γ) ->
    (A; Γ |--M φ) -> (A; Γ |--M <![i]φ!>)
where "A ; G |--M p" := (MMdeduction A G p).
```

Figura: Regras de Derivação do Sistema de Hilbert

```
Theorem MMdeduction_generalization_Kn: forall modalities indexes n  $\Gamma$   $\varphi$ ,  
  let K_to_Kn := formula_to_MMformula in  
  n < modalities ->  
  In n indexes ->  
  deduction K  $\Gamma$   $\varphi$  ->  
  MMdeduction modalities (Kn modalities indexes) (theory_to_MMtheory  $\Gamma$  n) (K_to_Kn  $\varphi$  n).
```

Figura: Prova de Generalização da Dedução para  $K_n$

```
Theorem join_preserves_deduction: forall modalities S1 n1  $\Gamma$   $\varphi$ ,  
  let K_to_Kn := formula_to_MMformula in  
  n1 < modalities -> deduction S1  $\Gamma$   $\varphi$  ->  
  forall S2 n2 b, (forall a, axiom_to_MMaxiom n1 a b) ->  
  join modalities S1 S2 n1 n2 (n1 :: n2 :: nil) b ->  
  MMdeduction modalities (join modalities S1 S2 n1 n2 (n1 :: n2 :: nil))  
    [theory_to_MMtheory  $\Gamma$  n1] (K_to_Kn  $\varphi$  n1).  
  
Theorem join_two_preserves_deduction: forall modalities S1 S2 l1 l2  $\Gamma$   $\varphi$  a,  
  ~ share_element l1 l2 ->  
  MMdeduction modalities (S1 l1)  $\Gamma$   $\varphi$  -> join_two S1 S2 l1 l2 (l1 ++ l2) a ->  
  MMdeduction modalities (join_two S1 S2 l1 l2 (l1 ++ l2))  $\Gamma$   $\varphi$ .
```

Figura: Prova da Preservação de Deduções pela Fusão



```
Theorem join_one_preserves_deduction_left: forall modalities S1 n  $\Gamma$   $\varphi$ ,  
  let K_to_Kn := formula_to_MMformula in  
  n < modalities -> deduction S1  $\Gamma$   $\varphi$  ->  
  forall S2 l b, (forall a, axiom_to_MMaxiom n a b) ->  
  join_one modalities S1 S2 n l (n :: l) b ->  
  MMdeduction modalities (join_one modalities S1 S2 n l (n :: l))  
    [theory_to_MMtheory  $\Gamma$  n] (K_to_Kn  $\varphi$  n).
```





```
Theorem join_one_preserves_deduction_right: forall modalities S2 l n  $\Gamma$   $\varphi$ ,  
  let K_to_Kn := formula_to_MMformula in  
  n < modalities -> ~ In n l -> MMdeduction modalities (S2 l)  $\Gamma$   $\varphi$  ->  
  forall S1 b, join_one modalities S1 S2 n l (n :: l) b ->  
  MMdeduction modalities (join_one modalities S1 S2 n l (n :: l))  $\Gamma$   $\varphi$ .
```

Figura: Prova da Preservação de Deduções pela Fusão

# Dificuldades na Implementação

- Representação da linguagem do sistema multimodal - Não representa união de linguagens;
- Definição de frames multimodais - Não foi possível representar fusão de frames e uso da função  $nth$ ;
- Tradução de modelos multimodais para modelos monomodais;
- Corretude do Sistema  $\mathbf{KT} \odot \mathbf{K4}$  não foi provada pelo método de transferência.

- Uma implementação paramétrica de fusão de sistemas sintáticos semelhante a desenvolvida nesse trabalho não foi encontrado nos trabalhos relacionados;
- Não foi possível terminar a implementação da fusão de sistemas semânticos devido a escolhas de implementação da biblioteca base;
- Não foi possível demonstrar transferência de propriedades no Coq.

-  CARNIELLI, W.; CONIGLIO, M.; MARCOS, J. Logics of Formal Inconsistency. In: \_\_\_\_\_. *Handbook of Philosophical Logic*. [S.l.]: Springer, 2007. p. 1–93. ISBN 978-1-4020-6323-7.
-  CARNIELLI, W.; CONIGLIO, M. E. *Paraconsistent logic: Consistency, contradiction and negation*. [S.l.]: Springer International Publishing, 2016.
-  GEUVERS, H. Proof assistants: History, ideas and future. *Sadhana*, Springer, v. 34, p. 3–25, 2009.
-  SILVEIRA, A. A. da. *Implementação de uma biblioteca de lógica modal em Coq*. Dissertação (Projeto de Diplomação) — Bacharelado em Ciência da Computação—Centro de Ciências Tecnológicas, UDESC, Joinville, 2020.