UNIVERSIDADE DO ESTADO DE SANTA CATARINA – UDESC
CENTRO DE CIÊNCIAS TECNOLÓGICAS – CCT
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO – BCC

HELENA VARGAS TANNURI

IMPLEMENTAÇÃO DE UMA BIBLIOTECA DA LÓGICA DE INCONSISTÊNCIA FORMAL LFI1 EM ROCQ

JOINVILLE
2025

HELENA VARGAS TANNURI

IMPLEMENTAÇÃO DE UMA BIBLIOTECA DA LÓGICA DE INCONSISTÊNCIA FORMAL LFI1 EM ROCQ

Trabalho de conclusão de curso submetido à Universidade do Estado de Santa Catarina como parte dos requisitos para a obtenção do grau de Bacharel em Ciência da Computação

Orientadora: Karina Girardi Roggia
Coorientador: Miguel Alfredo Nunes

JOINVILLE
2025

HELENA VARGAS TANNURI

IMPLEMENTAÇÃO DE UMA BIBLIOTECA DA LÓGICA DE INCONSISTÊNCIA FORMAL LFI1 EM ROCQ

Trabalho de conclusão de curso submetido à Universidade do Estado de Santa Catarina como parte dos requisitos para a obtenção do grau de Bacharel em Ciência da Computação

Orientadora: Karina Girardi Roggia
Coorientador: Miguel Alfredo Nunes

BANCA EXAMINADORA:

Orientadora:

Dra. Karina Girardi Roggia
UDESC

Coorientador:

Miguel Alfredo Nunes
UNICAMP

Membros:

Dr. Cristiano Damiani Vasconcellos
UDESC

Me. Paulo Henrique Torrens
University of Kent

Joinville, Junho de 2025

"Different conclusions are reached when one fact is viewed from two separate points of view. When that happens, there is no immediate way to judge which point of view is the correct one. There is no way to conclude one's own conclusion is the correct one. But for that exact reason, it is also premature to decide one's own conclusion is wrong."

(Senjougahara Hitagi — Bakemonogatari, [2009])

RESUMO

Na medida em que sistemas de computação modernos escalam, a existência de informações contraditórias torna-se inevitável. As lógicas ortodoxas não são capazes de tratar este tipo de informação sem que o princípio da explosão tome lugar. Com isso, sistemas paraconsistentes — sistemas nos quais a explosividade é cuidadosamente separada da contraditoriedade — são uma alternativa vantajosa quando comparados às lógicas ortodoxas. Neste contexto, as lógicas de inconsistência formal, sobretudo a **LFI1**, usufruem de propriedades interessantes que as garantem aplicações em diversos campos do conhecimento, como, por exemplo, no desenvolvimento de sistemas de gerenciamento de bancos de dados. Com isso, a prova de metateoremas sobre estas lógicas evidencia características das diferentes abordagens possíveis no estudo destes sistemas. Ademais, assistentes de provas, como o Rocq, proporcionam aos teoremas neles desenvolvidos uma garantia de correção dificilmente encontrada em provas manuais. Este trabalho propõe explanar e definir a lógica de inconsistência formal **LFI1**, bem como desenvolver metateoremas para este sistema no assistente de provas Rocq.

Palavras-chave: Rocq, lógica paraconsistente, **LFI1**, lógica de inconsistência formal, lógica trivalorada.

ABSTRACT

As modern computer systems scale, the existence of contradictory information becomes inevitable. Most orthodox logics are not able to cope with this kind of information without the principle of explosion taking place. Thus, establishing paraconsistent systems – systems in which explosiveness is carefully separated from contradictoriness – is an advantageous alternative compared to orthodox logics. From this perspective, the logics of formal inconsistency, specially **LFI1**, enjoy some interesting properties which allow them to be used in many areas, for example in the development of database management systems. In this light, proving metatheorems about these logics highlights characteristics of different approaches when studying these systems. Furthermore, proof assistants, such as Rocq, guarantee the theorems proved inside them a degree of certainty about their correctness hardly ever found in manual proofs. The present work explores and defines the logic of formal inconsistency **LFI1**, as well as proves metatheorems for this system inside the Rocq proof assistant.

Keywords: Rocq, paraconsistent logic, **LFI1**, logics of formal inconsistency, three-valued logic.

LISTA DE TABELAS

Tabela 1 – Valorações possíveis para φ, $\neg\varphi$ e $\circ\varphi$, considerando (*vNeg*), (*vCon*) e (*vCi*). 27
Tabela 2 – Cronograma Proposto para o TCC2 73

LISTA DE ABREVIATURAS E SIGLAS

MP	*Modus ponens*
MTD	Metateorema da dedução
sse	Se e somente se
CIC	*Calculus of inductive constructions*

LISTA DE SÍMBOLOS

$p, q, r \ldots$	Variáveis atômicas.
$A, B, C, \ldots$	Conjuntos quaisquer.
$\alpha, \beta, \gamma, \ldots$	Fórmulas quaisquer.
Γ, Δ	Conjuntos de fórmulas.
Σ, Θ	Assinaturas de linguagens.
$\Vdash$	Relação de consequência qualquer (sintática ou semântica).
$\vdash$	Relação de consequência sintática.
$\models$	Relação de consequência semântica.

SUMÁRIO

1 INTRODUÇÃO 10
1.1 OBJETIVO GERAL 12
1.2 OBJETIVOS ESPECÍFICOS 12
1.3 TRABALHOS RELACIONADOS 12
1.4 METODOLOGIA 13
1.5 ESTRUTURA DO TRABALHO 13
2 LÓGICAS DE INCONSISTÊNCIA FORMAL 14
2.1 PARACONSISTÊNCIA 15
2.2 INCONSISTÊNCIA FORMAL 17
3 A LÓGICA DE INCONSISTÊNCIA FORMAL LFI1 20
3.1 LINGUAGEM 21
3.2 AXIOMATIZAÇÃO 21
3.3 SEMÂNTICA 24
3.3.1 Semântica Matricial 25
3.3.2 Bivalorações 27
3.4 METATEOREMAS 28
3.4.1 Metateorema da dedução 28
3.4.2 Correção 34
3.4.3 Completude em relação às bivalorações 49
3.4.4 Equivalência entre as semânticas matricial e de valorações 55
3.4.5 Completude em relação à semântica matricial 56
4 IMPLEMENTAÇÃO EM ROCQ 57
4.1 ASSISTENTE DE PROVAS ROCQ 57
4.2 BIBLIOTECA EM ROCQ 58
4.2.1 Utils.v 58
4.2.2 Language.v 59
4.2.3 Syntax.v 59
4.2.4 Semantics.v 60
4.2.5 Deduction_metatheorem.v 64
4.2.6 Soundness.v 65
4.2.7 Cardinality.v 67
4.2.8 Completeness.v 69
5 CONCLUSÕES PARCIAIS 73
REFERÊNCIAS 74

1 INTRODUÇÃO

As lógicas paraconsistentes são uma família de lógicas na qual a presença de contradições não implica trivialidade, ou seja, são sistemas lógicos que possuem uma negação que não respeita o princípio da explosão, definido como $\alpha \rightarrow (\neg\alpha \rightarrow \beta)$ (CARNIELLI; CONIGLIO; MARCOS, 2007). Tradicionalmente, em lógicas ortodoxas, qualquer teoria que seja inconsistente — e, portanto, possua duas sentenças $\{\alpha, \neg\alpha\}$ — será uma teoria trivial (uma teoria que contém todas as sentenças). Deste modo, as lógicas paraconsistentes surgem como uma ferramenta que permite tratar contradições sem trivializar o sistema lógico (CARNIELLI; CONIGLIO, 2016).

De acordo com Priest, Tanaka e Weber (2022), as motivações para o estudo de lógicas paraconsistentes podem ser observadas em diversos campos do conhecimento. Nas ciências naturais, por exemplo, teorias inconsistentes e não-triviais são comuns, como é o caso da teoria do átomo de Bohr, que, segundo Brown e Priest (2015), deve possuir um mecanismo de inferência paraconsistente[1]. No campo da linguística, inconsistências não-triviais também são possíveis, como a preservação da noção espacial da palavra "próximo" mesmo tratando-se de objetos impossíveis[2]. Ademais, no contexto da computação, uma aplicação da paraconsistência é o uso de lógicas de inconsistência formal para a modelagem e o desenvolvimento de bancos de dados evolucionários (CARNIELLI; MARCOS; AMO, 2000).

As lógicas de inconsistência formal (**LFI**s) são lógicas paraconsistentes que introduzem na sua linguagem os conceitos de consistência e inconsistência como formas de representar o excesso de informações (por exemplo, evidência para α e evidência para $\neg\alpha$), para resgatar a capacidade de se obter a trivialidade em alguns casos (CARNIELLI; CONIGLIO; MARCOS, 2007). Ao explicitamente representar a consistência dentro da sua linguagem, é possível estudar teorias inconsistentes sem necessariamente assumir que elas são triviais, porém possibilitando a trivialidade em situações específicas. A ideia por trás das **LFI**s é que deve-se respeitar as noções da lógica clássica o máximo possível, desviando desta somente na presença de contradições (PRIEST; TANAKA; WEBER, 2022). Segundo Carnielli e Coniglio; Barrio e Carnielli (2016, 2019), na lógica **LFI1**, uma lógica paraconsistente e trivalorada, os conceitos de inconsistência e consistência são introduzidos à linguagem por meio do operador $\bullet$ para a inconsistência ou $\circ$ para a consistência, sendo que qualquer um destes pode ser usado para definir a linguagem da **LFI1**. Desta forma, como veremos ao longo do presente trabalho, e possível resgatar a trivialidade através do princípio da explosão Gentil, definido, no caso da **LFI1**, como $\circ\alpha \rightarrow (\alpha \rightarrow (\neg\alpha \rightarrow \beta))$ (CARNIELLI; CONIGLIO; MARCOS, 2007). Este princípio diz que a trivialidade é obtida a partir da contradição de uma informação consistente.

Um sistema lógico capaz de lidar com informações inconsistentes é de grande interesse

[1] De acordo com a teoria, um elétron orbita o núcelo do átomo sem radiar energia. Porém, de acordo com as equações de Maxwell, que compõem parte da teoria de Bohr, um elétron que está acelerando em órbita deve radiar energia. Estes fatos são inconsistentes entre si, entretanto, não é possível inferir *tudo* sobre o comportamento dos elétrons a partir disso. Portanto, o mecanismo de inferência deve se tratar de um mecanismo paraconsistente.

[2] Por exemplo, na sentença "Adam está próximo de um cubo esférico.", a noção espacial entre Adam e um objeto impossível é preservada (MCGINNIS, 2013).

no campo da computação, sobretudo no gerenciamento de bancos de dados (CARNIELLI; MARCOS; AMO, 2000). Um banco de dados pode ser definido como um conjunto estruturado de relações finitas que armazena informações. Estas informações precisam satisfazer condições conhecidas como restrições de integridade antes de serem inseridas no banco (CODD, 1970). As restrições são definidas pelo projetista do banco de dados no momento da implementação e podem ser formalizadas como sentenças de primeira ordem fixas (CARNIELLI; MARCOS; AMO, 2000). Conforme o banco de dados evolui, é preciso atualizar as informações contidas para refletir a realidade. Contudo, como informações contraditórias não são permitidas pelas restrições de integridade, isso torna o processo de atualização difícil e trabalhoso. Ademais, a existência de bancos de dados que possam alterar suas restrições de integridade com o passar do tempo (conhecidos como bancos de dados evolucionários) é outro ponto de interesse que pode ser explorado com o uso das **LFI**s.

Concomitante aos estudos das lógicas paraconsistentes, avanços nas áreas da computação e da matemática — como a definição de teoria de tipos por Russell (1903, 1908), a formulação desta teoria com base na sintaxe do Cálculo-λ por Church (1940) e o descobrimento da correspondência de Curry-Howard por Curry e Feys (1958) e Howard (1980) — possibilitaram o desenvolvimento de assistentes de provas (HARRISON; URBAN; WIEDIJK, 2014). Assistentes de provas são ferramentas da área de verificação formal, que buscam garantir que um programa está correto de acordo com uma especificação formal. Isto é feito a partir de provas desenvolvidas utilizando métodos matemáticos para a correção de propriedades de um *software* (CHLIPALA, 2019). Tradicionalmente, a verificação da validade de provas é feita manualmente por avaliadores, que seguem o raciocínio do autor e dão um veredito baseado no quão convincente a prova é. Os assistentes de provas surgem como alternativas à verificação manual, possibilitando ao matemático — ou programador — verificar provas na medida em que elas são desenvolvidas, tornando este processo mais fácil e confiável (PAULIN-MOHRING, 2015).

Assistentes de provas como Rocq, Lean e Isabelle permitem ao usuário definir e provar propriedades sobre objetos matemáticos com valor computacional (GEUVERS, 2009). No presente trabalho será utilizado o Rocq (anteriormente conhecido como Coq), que utiliza o cálculo de construções indutivas (CIC) como formalismo para o desenvolvimento de provas (The Coq Development Team, 2024). O Rocq ganhou notoriedade como ferramenta de verificação formal após seu uso na prova de correção de diversos teoremas e sistemas computacionais complexos, como a prova do teorema das quatro cores (GEUVERS, 2009), a certificação de um compilador para a linguagem de programação C (LEROY, 2021) e a prova da correção do algoritmo união-busca (CONCHON; FILLIÂTRE, 2007).

A proposta deste trabalho é desenvolver uma biblioteca da lógica de inconsistência formal **LFI1** em Rocq, de maneira análoga como foi feito para a lógica modal por Silveira (2020). Após a implementação da biblioteca, propõe-se que sejam provados metateoremas relevantes para a **LFI1** utilizando o Rocq.

1.1 OBJETIVO GERAL

O objetivo geral deste trabalho é implementar uma biblioteca de **LFI1** em Rocq, assim como desenvolver provas dos metateoremas da dedução, correção e completude dentro da biblioteca.

1.2 OBJETIVOS ESPECÍFICOS

- Estudar conceitos relevantes sobre lógicas paraconsistentes, em especial a **LFI1**;
- Estudar e revisar as provas manuais para os metateoremas da dedução, correção e completude da **LFI1**;
- Realizar um levantamento do estado da arte do desenvolvimento de lógicas paraconsistentes em assistentes de provas;
- Desenvolver uma biblioteca da **LFI1** em Rocq, baseada na semântica e sintaxe previamente definidas;
- Desenvolver e verificar formalmente as provas para os metateoremas da dedução, correção e completude em Rocq.

1.3 TRABALHOS RELACIONADOS

A partir de um levantamento acerca do estado da arte do desenvolvimento de lógicas não-clássicas em computação na literatura, foram encontrados alguns trabalhos relacionados.

Em (VILLADSEN; SCHLICHTKRULL, 2017), os autores implementam uma biblioteca de uma lógica paraconsistente utilizando assistente de provas Isabelle. A lógica em questão possui uma quantidade infinita contável de valores verdades não-clássicos, sendo uma generalização da lógica trivalorada proposta por Łukasiewicz, como definida por Simons (2023). Além de implementar a biblioteca, são provados metateoremas sobre esta lógica, como o número mínimo de valores verdades a serem analisados para determinar o valor verdade de uma fórmula e os metateoremas da redução e da dedução. Em outro trabalho, Schlichtkrull (2019) prova alguns outros metateoremas para esta mesma lógica dentro do assistente, expandindo ainda mais a biblioteca já implementada.

Amo e Pais (2007) especificam uma linguagem de consulta a banco de dados baseada na lógica de inconsistência formal **LFI1**. A linguagem em questão, chamada de P-Datalog, é uma extensão paraconsistente de outra linguagem chamada Datalog$^{\neg}$, esta, por sua vez, se trata de uma linguagem dedutiva[3] de consulta baseada na lógica de primeira ordem clássica. No trabalho

[3] Uma linguagem dedutiva de consulta é capaz de concluir fatos adicionais a partir do que está armazenado num determinado banco de dados (ABITEBOUL; HULL; VIANU, 1995).

é apresentada uma semântica bem-fundada para P-Datalog, bem como um método de avaliação *bottom-up*, baseado numa computação de ponto fixo alternante (descrito em Gelder (1989)).

Em (ÁVILA; ABE; PRADO, 1997), os autores descrevem uma extensão da linguagem de programação ParaLog, chamada de ParaLog_*e*, que se propõe a mesclar conceitos de programação lógica clássica com conceitos de inconsistência, a fim de ampliar o escopo de programação lógica em ambientes com informações contraditórias. A linguagem ParaLog_*e* utiliza como base a lógica evidencial[4], uma lógica paraconsistente com uma quantidade infinita (não-contável) de valores verdades, membros de $[0,1] \times [0,1]$. No trabalho, os autores definem a sintaxe e a semântica da linguagem e apresentam exemplos de programas escritos nela.

1.4 METODOLOGIA

O desenvolvimento do presente trabalhou se iniciou com uma pesquisa bibliográfica envolvendo os fundamentos teóricos acerca da lógica tratada e uma análise das provas dos metateoremas da **LFI1** existentes na literatura. Depois disso, foi feito um levantamento do estado da arte de implementações de lógicas paraconsistentes em assistentes de provas diversos, bem como aplicações da paraconsistência no contexto da computação.

Com a finalização desta etapa, foi iniciada a escrita do texto e o desenvolvimento das provas manuais dos metateoremas da **LFI1** com base na literatura pesquisada.

1.5 ESTRUTURA DO TRABALHO

O trabalho está organizado como segue: no Capítulo 2, as lógicas de inconsistência formal são definidas e caracterizadas em relação a suas propriedades e motivações para seu desenvolvimento. No Capítulo 3, a lógica **LFI1** é apresentada e sua linguagem, sintaxe e semântica são definidas formalmente, além disso, alguns metateoremas relevantes são provados. Por fim, tem-se o capítulo de considerações parciais.

[4] Uma revisão sobre esta lógica pode ser encontrada em Junior et al. (2024).

2 LÓGICAS DE INCONSISTÊNCIA FORMAL

No estudo de lógicas clássicas, uma contradição é considerada inseparável da trivialidade, ou seja, se uma teoria possuir um subconjunto $\{\alpha, \neg\alpha\}$ de fórmulas, pode-se derivar qualquer sentença. Esta propriedade é chamada de *explosividade*. Desta forma, as lógicas clássicas (e certas lógicas não-clássicas, como a lógica intuicionista), expressam sua *explosividade* como representada pela seguinte equação:

Contradições = Trivialidade

As *lógicas de inconsistência formal* são lógicas paraconsistentes que se propõem a questionar a noção apresentada anteriormente sem abrir mão completamente da trivialidade. Isto é feito estabelecendo uma nova propriedade, chamada de *explosividade gentil*, que resgata a trivialidade introduzindo o conceito de consistência na sua linguagem (CARNIELLI; CONIGLIO; MARCOS, 2007). A consistência é expressa na *explosividade gentil* da seguinte forma:

Contradições + Consistência = Trivialidade

Definir uma lógica que consiga superar o tabu da *explosividade* e, ao mesmo tempo, representar uma ferramenta legítima capaz de formalizar o raciocínio e separar inferências aceitáveis de inferências equivocadas é um dos objetivos dos lógicos que se descrevem como *paraconsistentistas*. As lógicas de inconsistência formal cumprem este objetivo de maneira elegante, servindo um propósito importante no estudo de lógicas não-clássicas.

Neste capítulo, são apresentadas algumas definições necessárias para caracterizar as lógicas de inconsistência formal, baseadas em Carnielli e Coniglio (2016) e em Carnielli, Coniglio e Marcos (2007). Antes de definir as **LFI**s é preciso apresentar alguns conceitos básicos acerca de sistemas lógicos paraconsistentes.

Ademais, o presente trabalho segue o mesmo caminho de Carnielli e Coniglio (2016), baseando-se na teoria geral de relações de consequências para definir *lógicas Tarskianas*. Neste sentido, como a lógica **LFI1** se trata de uma *lógica Tarskiana*, o presente trabalho se restringe a trabalhar somente neste escopo.

Nas Seções 2.1 e 2.2, serão apresentadas definições necessárias para identificar formalmente as lógicas paraconsistentes e lógicas de inconsistência formal. Estas definições se aplicam tanto a relações de consequência semântica quanto a relações de consequência sintática, denotadas genericamente pelo operador $\Vdash$. Este trabalho não se aprofunda em detalhes sobre conceitos envolvendo relações e operações de consequência, suas propriedades ou as diferenças entre abordagens prova-teóricas e modelo-teóricas. O leitor interessado em tais assuntos pode consultar os trabalhos de Wójcicki (1984, 1988a, 1988b) e Carnielli et al. (2008).

2.1 PARACONSISTÊNCIA

Um sistema lógico que se atreve a romper com o princípio da explosão — o qual afirma que a partir de uma teoria contraditória, qualquer conclusão segue — é dito paraconsistente. As justificativas para questionar tal princípio existem em diversos campos do conhecimento, como na linguística (MCGINNIS, 2013), na computação (CARNIELLI; MARCOS; AMO, 2000) e até mesmo nas ciências naturais (BROWN; PRIEST, 2015). Outra justificativa para o desenvolvimento de sistemas lógicos paraconsistentes é um descontentamento com o caráter explosivo das lógicas ortodoxas. Por exemplo, uma discordância de que a partir da evidência de que "Choveu na tarde de ontem." e da evidência de que "Não choveu na tarde de ontem." pode-se concluir que "Um triângulo tem quatro lados." [1]. Nesta seção, a paraconsistência será definida formalmente, partindo de definições básicas sobre lógica e classificando diferentes sistemas de acordo com propriedades acerca de sua *relação de consequência*.

Uma lógica $\mathscr{L}$ será representada como uma dupla $\mathscr{L} = \langle \mathcal{L}, \Vdash \rangle$, onde $\mathcal{L}$ é sua linguagem (seu conjunto de fórmulas bem formadas) e $\Vdash$ é uma relação de consequência de conclusão única, definida como $\Vdash \subseteq \wp(\mathcal{L}) \times \mathcal{L}$, sendo $\wp(\mathcal{L})$ o conjunto das partes de $\mathcal{L}$. Em uma consequência do tipo $\Gamma \Vdash \alpha$ (lida como "α é uma consequência de Γ") diz-se que o conjunto Γ é o conjunto de premissas e α é a conclusão. A fim de facilitar a escrita e leitura, a seguinte notação será utilizada ao longo do texto:

Notação 1. Sejam Γ, Δ conjuntos de fórmulas e φ, ψ fórmulas, então $\Gamma, \Delta, \varphi \Vdash \psi$ denota $\Gamma \cup \Delta \cup \{\varphi\} \Vdash \psi$.

Definição 1 (Assinatura proposicional)**.** Uma assinatura proposicional Θ é um conjunto de conectivos lógicos, cada um contendo a informação sobre sua aridade. ■

Por exemplo, a assinatura proposicional para a lógica proposicional clássica pode ser definida como $\Theta_{LPC} = \{\wedge^2, \vee^2, \neg^1, \rightarrow^2\}$, onde o operador $\wedge^2$ representa uma conjunção, $\vee^2$ representa uma disjunção, $\neg^1$ representa uma negação e $\rightarrow^2$ representa uma implicação. No restante do texto, as aridades destes conectivos será omitida e a aridade de novos conectivos será apresentada somente na sua definição.

Uma assinatura proposicional juntamente com um conjunto enumerável de átomos são base para a definição de uma linguagem proposicional, que por sua vez é utilizada para definir uma lógica proposicional, como é o caso da **LFI1**.

Definição 2 (Lógica proposicional)**.** Um sistema lógico $\mathscr{L}$, definido sobre uma linguagem $\mathcal{L}_\Theta$ é dito proposicional caso $\mathcal{L}_\Theta$ seja definida a partir de um conjunto enumerável de átomos $\mathcal{P} = \{p_i \mid i \in \mathbb{N}\}$ e uma assinatura proposicional Θ. A linguagem $\mathcal{L}_\Theta$ é chamada de linguagem proposicional. Escreveremos $\mathcal{L}$ caso a assinatura da linguagem seja irrelevante ou possa ser inferida sem ambiguidade a partir do contexto. ■

[1] Esta forma de explosividade é um exemplo de uma contradição estudada pelas lógicas de relevância, que tratam da conexão entre as premissas e a conclusão de uma inferência (MARES, 2024).

No exemplo da lógica proposicional clássica, sua linguagem pode ser definida como segue:

Definição 3 (Linguagem da LPC)**.** A linguagem $\mathcal{L}_{\Theta_{LPC}}$ da LPC é definida indutivamente como o menor conjunto a que respeita as seguintes regras:

1. $\mathcal{P} \subseteq \mathcal{L}_{\Theta_{LPC}}$
2. Se $\varphi \in \mathcal{L}_{\Theta_{LPC}}$, então $\neg\varphi \in \mathcal{L}_{\Theta_{LPC}}$
3. Se $\varphi, \psi \in \mathcal{L}_{\Theta_{LPC}}$, então $\varphi \otimes \psi \in \mathcal{L}_{\Theta_{LPC}}$, com $\otimes \in \{\wedge, \vee, \rightarrow\}$ ■

No desenvolvimento de metateoremas sobre propriedades de uma determinada lógica, a indução na complexidade de uma fórmula é um método comum de prova. Para isso, dada uma lógica $\mathscr{L}$ sobre uma linguagem $\mathcal{L}$, define-se uma função recursiva $C(\varphi) : \mathcal{L} \rightarrow \mathbb{N}$ que retorna, para uma dada fórmula, um número natural representando sua complexidade, baseada na quantidade de operadores e átomos:

Definição 4 (Complexidade de fórmulas para a lógica proposicional clássica)**.** Dada uma fórmula $\varphi \in \mathcal{L}_{LPC}$, a complexidade $C(\varphi)$ é definida recursivamente da seguinte forma:

1. Se $\varphi = p$, onde $p \in \mathcal{P}$, então $C(\varphi) = 1$;
2. Se $\varphi = \neg\psi$, então $C(\varphi) = C(\psi) + 1$;
3. Se $\varphi = \psi \otimes \gamma$, onde $\otimes \in \{\wedge, \vee, \rightarrow\}$, então $C(\varphi) = C(\psi) + C(\gamma) + 1$. ■

Definição 5 (Substituição)**.** Uma substituição σ de todas as ocorrências de uma variável p_i por uma fórmula ψ em uma fórmula φ, é denotada por $\sigma(\varphi) = \varphi\{p_i \mapsto \psi\}$ (SILVA; FINGER; MELO, 2006). A substituição $\varphi\{p_i \mapsto \psi\}$ é definida indutivamente como (considerando $\triangle$, $\otimes$ conectivos quaisquer de aridade 1 e 2 respectivamente):

1. Se $\varphi = p_i$ então, $\varphi\{p_i \mapsto \psi\} = \psi$;
2. Se $\varphi = p_j$ e $j \neq i$ então, $\varphi\{p_i \mapsto \psi\} = \varphi$;
3. Se $\varphi = \triangle\gamma$ então, $\varphi\{p_i \mapsto \psi\} = \triangle(\gamma\{p_i \mapsto \psi\})$;
4. Se $\varphi = \varphi_0 \otimes \varphi_1$ então, $\varphi\{p_i \mapsto \psi\} = \varphi_0\{p_i \mapsto \psi\} \otimes \varphi_1\{p_i \mapsto \psi\}$.

Uma fórmula α é dita *instância de substituição* de uma fórmula β caso exista uma substituição σ tal que $\alpha = \sigma(\beta)$. ■

Uma lógica Tarksiana é uma lógica que goza das propriedades da reflexividade, monotonicidade e corte, definida formalmente como segue:

Definição 6 (Lógica Tarskiana)**.** Uma lógica $\mathscr{L}$, definida sobre uma linguagem $\mathcal{L}$ e munida com uma relação de consequência $\Vdash$ é dita *Tarskiana* caso satisfaça as seguintes propriedades para todo $\Gamma \cup \Delta \cup \{\alpha\} \subseteq \mathcal{L}$:

(i) Se $\alpha \in \Gamma$ então $\Gamma \Vdash \alpha$; (reflexividade)

(ii) Se $\Delta \Vdash \alpha$ e $\Delta \subseteq \Gamma$ então $\Gamma \Vdash \alpha$; (monotonicidade)

(iii) Se $\Delta \Vdash \alpha$ e $\Gamma \Vdash \delta$ para todo $\delta \in \Delta$ então $\Gamma \Vdash \alpha$. (corte)

■

A seguir, definiremos uma notação para a aplicação de uma função sobre um conjunto, similar a uma função *map* em programação funcional.

Notação 2. Dada uma função $f : A \to B$ e um conjunto $A' \subseteq A$, $f[A']$ denota o conjunto $\{f(a) \mid a \in A'\}$.

Com a noção de substituição para lógicas proposicionais apresentada na Definição 5, é possível definir lógica *padrão* como sendo uma lógica Tarskiana, *finitária* e *estrutural*:

Definição 7 (Lógica padrão)**.** Uma lógica proposicional $\mathscr{L}$ definida sobre uma linguagem proposicional $\mathcal{L}$ é dita *estrutural* caso respeite a seguinte condição para todo $\Gamma \cup \Delta \cup \{\alpha\} \subseteq \mathcal{L}$:

(i) Se $\Gamma \Vdash \alpha$, então $\sigma[\Gamma] \Vdash \sigma(\alpha)$, para toda substituição σ de variável por fórmula.

Uma lógica $\mathscr{L}$ é dita *finitária* caso satisfaça a seguinte condição:

(ii) Se $\Gamma \Vdash \alpha$, então existe conjunto finito $\Gamma_0 \subseteq \Gamma$ tal que $\Gamma_0 \Vdash \alpha$.

Por fim, uma lógica proposicional $\mathscr{L}$ é dita *padrão* caso ela seja Tarskiana, finitária e estrutural. ■

Com isto, definiremos formalmente o conceito de *paraconsistência* para lógicas Tarskianas.

Definição 8 (Lógica Tarskiana paraconsistente)**.** Uma lógica Tarskiana $\mathscr{L}$, definida sobre uma linguagem $\mathcal{L}$, é dita *paraconsistente* se ela possuir uma negação[2] $\neg$ e existirem fórmulas $\alpha, \beta \in \mathcal{L}$ tal que $\alpha, \neg\alpha \nVdash \beta$. ■

Caso a linguagem de $\mathscr{L}$ possua uma implicação $\to$ que respeite o metateorema da dedução[3], então $\mathscr{L}$ é paraconsistente se e somente se a fórmula $\alpha \to (\neg\alpha \to \beta)$ não for válida. Ou seja, caso o princípio da explosão é inválido (em relação a $\neg$), o conectivo $\neg$ é considerado uma negação *não explosiva*.

2.2 INCONSISTÊNCIA FORMAL

A motivação para o desenvolvimento das **LFI**s é a existência sistemas lógicos paraconsistentes nos quais é possível resgatar, de maneira *controlada*, o princípio da explosão. Ao internalizar o conceito de consistência, as **LFI**s propõem a noção de que uma contradição que

[2] Esta negação pode ser primitiva (pertencente à assinatura da linguagem) ou definida a partir de outras fórmulas.
[3] Definido como $\Gamma, \alpha \Vdash \beta \iff \Gamma \Vdash \alpha \to \beta$.

é reconhecidamente inconsistente numa dada teoria é inofensiva e é somente fruto do excesso de informação. O resgate *controlado* da explosividade é feito definindo um conjunto $\bigcirc(p)$ de fórmulas dependentes somente de uma variável proposicional p. Caso uma lógica $\mathscr{L}$ seja explosiva ao unir-se um conjunto $\bigcirc(\alpha)$ com uma contradição $\{\alpha, \neg\alpha\}$ (ou seja, se o seguinte for válido para todo α e β pertencentes à sua linguagem: $\bigcirc(\alpha), \alpha, \neg\alpha \Vdash \beta$ e $\bigcirc(\alpha), \alpha \nVdash \beta$ e $\bigcirc(\alpha), \neg\alpha \nVdash \beta$) então dizemos que $\mathscr{L}$ é *gentilmente explosiva*. Esta é uma das formas de definir as lógicas de inconsistência formal. Como será mostrado nesta seção, existem ao menos outras três definições diferentes para as **LFI**s e sua *explosividade gentil*.

Notação 3. Dado um átomo p, define-se $\bigcirc(p)$ como um conjunto não-vazio de fórmulas dependentes somente de p. Com base neste conjunto, define-se a notação $\bigcirc(\varphi)$ para representar o conjunto obtido pela substituição de todas as ocorrências de p por φ em todos os elementos de $\bigcirc(p)$, ou seja, para uma fórmula φ qualquer, $\bigcirc(\varphi) = \{\psi\{p \mapsto \varphi\} \mid \psi \in \bigcirc(p)\}$.

A definição a seguir foi proposta por Carnielli e Coniglio (2016) e se encontra no meio de outras duas definições, uma mais generalista (apresentada na Definição 10) e outra mais restrita (apresentada na Definição 11). Ela é utilizada para definir o que é informalmente descrito como **LFI** no início desta seção.

Definição 9 (Lógica de Inconsistência Formal)**.** Seja $\mathscr{L} = \langle \mathcal{L}_\Theta, \Vdash \rangle$ uma lógica padrão, de forma que sua assinatura proposicional Θ possua uma negação $\neg$. Seja $\bigcirc(p)$ um conjunto não-vazio de fórmulas dependentes somente da variável proposicional p. Então $\mathscr{L}$ será uma lógica de inconsistência formal (**LFI**) (em relação a $\bigcirc(p)$ e $\neg$) caso respeite as seguintes condições:

(i) Existem $\gamma, \delta \in \mathcal{L}_\Theta$, de modo que $\gamma, \neg\gamma \nVdash \delta$;

(ii) Existem $\alpha, \beta \in \mathcal{L}_\Theta$, de modo que:

(ii.a) $\bigcirc(\alpha), \alpha \nVdash \beta$;

(ii.b) $\bigcirc(\alpha), \neg\alpha \nVdash \beta$;

(iii) Para todo $\varphi, \psi \in \mathcal{L}_\Theta$, tem-se $\bigcirc(\varphi), \varphi, \neg\varphi \Vdash \psi$. ■

A condição (i) diz que toda **LFI** é *não-explosiva* (em relação a $\neg$) e as condições (ii) e (iii) dizem que toda **LFI** é *gentilmente explosiva* (em relação a $\bigcirc(p)$ e $\neg$).

Na literatura, existem outras duas definições para as lógicas de inconsistência formal, que relaxam a condição (ii) para obter uma definição mais uniforme. Carnielli e Coniglio (2016), definem **LFI**s *fracas* da seguinte forma:

Definição 10 (**LFI** Fraca)**.** Seja $\mathscr{L} = \langle \mathcal{L}_\Theta, \Vdash \rangle$ uma lógica padrão, de forma que sua assinatura proposicional Θ possua uma negação $\neg$. Seja $\bigcirc(p)$ um conjunto não-vazio de fórmulas dependentes somente na variável proposicional p. Então $\mathscr{L}$ será uma **LFI** *fraca* (em relação a $\bigcirc(p)$ e $\neg$) caso ela respeite as seguintes condições:

(i) Existem $\varphi, \psi \in \mathcal{L}_\Theta$, de modo que $\varphi, \neg\varphi \nVdash \psi$;

(ii) Existem $\varphi, \psi \in \mathcal{L}_\Theta$, de modo que $\bigcirc(\varphi), \varphi \nvdash \psi$;

(iii) Existem $\varphi, \psi \in \mathcal{L}_\Theta$, de modo que $\bigcirc(\varphi), \neg\varphi \nvdash \psi$;

(iv) Para todo $\varphi, \psi \in \mathcal{L}_\Theta$, tem-se $\bigcirc(\varphi), \varphi, \neg\varphi \vdash \psi$. ■

Como é possível observar pelas duas definições acima, toda **LFI** é uma **LFI** fraca (já que a condição (ii) da Definição 9 satisfaz as condições (ii) e (iii) da Definição 10), mas o inverso não é necessariamente verdade (observe que as váriaveis φ e ψ das condições (ii) e (iii) não precisam ser iguais). Ademais, é possível estabelecer outra definição (também mais uniforme do que a Definição 9) que introduz o conceito de **LFI**s *fortes* como feito abaixo:

Definição 11 (**LFI** Forte)**.** Seja $\mathscr{L} = \langle \mathcal{L}_\Theta, \vdash \rangle$ uma lógica padrão, de forma que sua assinatura proposicional Θ possua uma negação $\neg$. Seja $\bigcirc(p)$ um conjunto não-vazio de fórmulas dependentes somente na variável proposicional p. Então $\mathscr{L}$ será uma **LFI** *forte* (em relação a $\bigcirc(p)$ e $\neg$) caso ela respeite as seguintes condições:

(i) Existem $\alpha, \beta \in \mathcal{L}_\Theta$, de modo que:

(i.a) $\alpha, \neg\alpha \nvdash \beta$;

(i.b) $\bigcirc(\alpha), \alpha \nvdash \beta$;

(i.c) $\bigcirc(\alpha), \neg\alpha \nvdash \beta$;

(ii) Para todo $\varphi, \psi \in \mathcal{L}_\Theta$, tem-se $\bigcirc(\varphi), \varphi, \neg\varphi \vdash \psi$. ■

É imediato perceber que toda **LFI** forte é uma **LFI** (já que a condição (i) da Definição 11 satisfaz as condições (i) e (ii) da Definição 9), mas o inverso não é necessariamente verdade. Ademais, no escopo das lógicas proposicionais, é possível estabelecer uma forma mais simples de provar que uma dada lógica proposicional é uma **LFI** forte, tomando α e β como dois átomos p e q quaisquer nas condições (i.a), (i.b) e (i.c) da definição acima.

Definição 12 (**LFI** forte para lógicas proposicionais)**.** Seja $\mathscr{L} = \langle \mathcal{L}_\Theta, \vdash \rangle$ uma lógica padrão, de forma que sua assinatura proposicional Θ possua uma negação $\neg$ e sua linguagem $\mathcal{L}_\Theta$ seja definida sobre um conjunto enumerável de átomos $\mathcal{P} = \{p_0, \ldots, p_n\}$. Seja $\bigcirc(p)$ um conjunto não-vazio de fórmulas dependentes somente na variável proposicional p. Então $\mathscr{L}$ será uma **LFI** *forte* (em relação a $\bigcirc(p)$ e $\neg$) caso ela respeite as seguintes condições:

(i) Existem $p, q \in \mathcal{P}$, de modo que:

(i.a) $p, \neg p \nvdash q$;

(i.b) $\bigcirc(p), p \nvdash q$;

(i.c) $\bigcirc(p), \neg p \nvdash q$;

(ii) Para todo $\varphi, \psi \in \mathcal{L}_\Theta$, tem-se $\bigcirc(\varphi), \varphi, \neg\varphi \vdash \psi$. ■

Esta definição será utilizada para provar que a lógica proposicional **LFI1** se trata de uma lógica de inconsistência formal forte.

3 A LÓGICA DE INCONSISTÊNCIA FORMAL LFI1

Com os avanços da internet no contexto do gerenciamento de bancos de dados, informações passaram a ser coletadas a partir de diferentes fontes que frequentemente se contradizem. Dada a existência das restrições de integridade — que impedem contradições — a atualização e manutenção de bancos de dados se torna um processo difícil (CARNIELLI; MARCOS; AMO, 2000). Portanto, uma lógica capaz de lidar com informações inconsistentes sem necessariamente sofrer com a trivialidade é de grande interesse. A lógica de inconsistência formal **LFI1** é capaz de lidar com contradições ao introduzir na sua assinatura o operador $\circ$ para representar a consistência, internalizando este conceito em sua linguagem. Uma informação é dita consistente caso ela e sua negação não sejam simultaneamente verdadeiras, ou seja, dada uma informação α, sua consistência $\circ\alpha$ será verdade quando α for falsa ou $\neg\alpha$ for falsa. Com a introdução deste novo operador, é possível lidar com a inconsistência de informações sem que trivialidade ocorra, já que — caso uma informação seja conhecidamente *inconsistente*, ou seja, $\neg\circ\alpha$ — então ela se trata de uma contradição inofensiva, fruto do excesso de informações numa dada teoria. Com isso, na **LFI1**, o conjunto $\bigcirc(p)$ de fórmulas dependentes somente na variável p (descrito na Definição 9) assume forma $\{\circ p\}$.

No trabalho de Carnielli, Marcos e Amo (2000), a lógica **LFI1*** é definida como uma extensão de primeira ordem da lógica proposicional **LFI1**. A motivação para definir-se uma lógica de inconsistência formal de primeira ordem vem da natureza das informações contidas em bancos de dados, estas que podem ser compreendidas como sentenças de primeira ordem fixas (CODD, 1970), entretanto, o presente trabalho trata somente da lógica proposicional **LFI1**. Ademais, Carnielli, Marcos e Amo (2000) tomam o operador de *inconsistência* (denotado por $\bullet$) como primitivo. Isto foi feito pois o foco era explorar a **LFI1** como uma ferramenta para lidar com inconsistências em bancos de dados, portanto tomar a inconsistência como primitiva era de grande interesse. Entretanto, no presente trabalho, será utilizada a definição apresentada em Carnielli e Coniglio (2016), onde a linguagem é definida utilizando o operador $\circ$ como primitivo. Isto salienta algumas propriedades interessantes da negação $\neg$, como a presença das leis de De Morgan, axiomatizadas na Seção 3.2.

Este capítulo é dividido da seguinte forma: na Seção 3.1, é apresentada a linguagem da lógica proposicional **LFI1** bem como definições necessárias para desenvolver as provas de metateoremas. A Seção 3.2 contém uma breve explicação sobre sistemas de prova sintáticos e um cálculo de Hilbert para a **LFI1** é definido. Na Seção 3.3, a semântica da **LFI1** é definida a partir de matrizes lógicas e de uma semântica de valorações não determinística. Finalmente, na Seção 3.4, são provados os metateoremas da dedução, correção, completude e a equivalência entre os diferentes sistemas semânticos apresentados.

3.1 LINGUAGEM

A lógica proposicional **LFI1** aqui apresentada é definida com base em Carnielli e Coniglio (2016) sobre a linguagem $\mathcal{L}_\Sigma$, que por sua vez é definida sobre um conjunto enumerável de átomos $\mathcal{P} = \{p_n \mid n \in \mathbb{N}\}$ e uma assinatura proposicional $\Sigma = \{\wedge^2, \vee^2, \rightarrow^2, \neg^1, \circ^1\}$. Como de costume, o conectivo $\wedge$ representa uma conjunção, $\vee$ representa uma disjunção, $\rightarrow$ representa uma implicação, $\neg$ representa uma negação e $\circ$ é o conectivo de consistência, definido de forma primitiva. No restante do texto a aridade destes conectivos será omitida. A linguagem $\mathcal{L}_\Sigma$ da **LFI1** é definida da seguinte forma:

Definição 13 (Linguagem da **LFI1**)**.** A linguagem $\mathcal{L}_\Sigma$ da **LFI1** é definida indutivamente como o menor conjunto a que respeita as seguintes regras:

1. $\mathcal{P} \subseteq \mathcal{L}_\Sigma$
2. Se $\varphi \in \mathcal{L}_\Sigma$, então $\triangle\varphi \in \mathcal{L}_\Sigma$, com $\triangle \in \{\neg, \circ\}$
3. Se $\varphi, \psi \in \mathcal{L}_\Sigma$, então $\varphi \otimes \psi \in \mathcal{L}_\Sigma$, com $\otimes \in \{\wedge, \vee, \rightarrow\}$ ■

A precedência dos conectivos é dada de maneira costumeira, com a adição do operador $\circ$ de consistência, seguindo a ordem (da maior precedência para a menor): $\circ$, $\neg$, $\wedge$, $\vee$, $\rightarrow$. Os conectivos binários $\wedge$ e $\vee$ são associativos à esquerda, ou seja, uma expressão do tipo $\alpha \wedge \beta \wedge \gamma$ é lida como $((\alpha \wedge \beta) \wedge \gamma)$, e o conectivo $\rightarrow$ é associativo à direita, ou seja, uma expressão do tipo $\alpha \rightarrow \beta \rightarrow \gamma$ é lida como $(\alpha \rightarrow (\beta \rightarrow \gamma))$.

A linguagem da **LFI1** pode ser definida de maneira equivalente utilizando-se o operador de inconsistência (representado por $\bullet$), definido como $\bullet\alpha \stackrel{\text{def}}{=} \neg \circ \alpha$, como feito por Carnielli, Marcos e Amo (2000).

Na Definição 4, a função C da complexidade de uma fórmula na lógica proposicional clássica foi recursivamente definida. É possível estendê-la para identificar a complexidade de uma fórmula na **LFI1** adicionando-se uma condição para o operador $\circ$:

Definição 14 (Complexidade de uma fórmula na **LFI1**)**.** Seja $\varphi \in \mathcal{L}_\Sigma$ uma fórmula bem formada, a complexidade $C(\varphi)$ é dada adicionando-se a seguinte condição à Definição 4:

Se $\varphi = \circ\psi$, então $C(\varphi) = C(\psi) + 2$. ■

Note que a complexidade de uma fórmula do tipo $\circ\alpha$ é estritamente maior que a complexidade de α e $\neg\alpha$. Isto se dá pois, como será evidenciado pela semântica de valorações na Definição 21, existe uma dependência de $\circ\alpha$ em $\{\alpha, \neg\alpha\}$, como apresentada por Carnielli e Coniglio (2016).

3.2 AXIOMATIZAÇÃO

A teoria das provas é uma das abordagens para o estudo das relações de consequência, onde a validade de uma inferência é atestada caso haja uma *prova* das conclusões a partir das

premissas. Uma prova consiste em uma sequência de passos bem definidos aplicados sobre conjuntos (ocasionalmente unitários) de fórmulas, com base nos princípios de um determinado sistema de provas. A teoria das provas é sintática[1] por natureza, ou seja, numa inferência $A \vdash B$, é relevante apenas a estrutura das fórmulas presentes em A e B, não sua interpretação ou valor-verdade. Essa estrutura é manipulada a fim de obter-se uma sequência de passos que — além de atestar sua validade — serve como argumento para tal (BEALL; RESTALL; SAGI, 2024). Desta forma, pode-se definir um sistema de provas sintático para servir como relação de consequência para uma determinada lógica.

No contexto da **LFI1**, existem dois sistemas de prova sintáticos estabelecidos até o momento: um cálculo de Hilbert, descrito pelos autores Carnielli, Marcos e Amo; Carnielli e Coniglio (2000, 2016) e um sistema de *Tableau*, descrito por Carnielli e Marcos (2001). No presente trabalho, foi escolhido o cálculo de Hilbert para definir a sintaxe da **LFI1**, dada a maior facilidade para desenvolver metateoremas em contraste ao sistema de *Tableau*.

O cálculo de Hilbert (também conhecido como sistema de Hilbert ou axiomatização de Hilbert) é um sistema composto por um conjunto de fórmulas, chamadas de *axiomas* e um conjunto de *regras de inferência*. Uma regra de inferência é formada por uma lista de fórmulas chamadas de premissas da regra e uma fórmula chamada de conclusão da regra (RESTALL, 1999). Uma prova (também chamada de derivação ou dedução) do tipo $\Gamma \vdash \varphi$ consiste em uma sequência finita de fórmulas $\psi_0, \ldots, \psi_n$, onde $\psi_n = \varphi$, e cada ψ_i $(0 \leq i \leq n)$ é um axioma, um elemento do conjunto de premissas Γ ou o resultado da aplicação de uma regra de inferência em fórmulas anteriores. Usualmente, o cálculo de Hilbert possui apenas uma regra de inferência, esta sendo o *modus ponens*. Este também é o caso para o cálculo de Hilbert que será utilizado para a **LFI1**.

O cálculo de Hilbert definido a seguir foi apresentado por Carnielli e Coniglio (2016) como alternativa ao que havia sido definido em trabalhos anteriores (CARNIELLI; MARCOS; AMO, 2000; CARNIELLI; CONIGLIO; MARCOS, 2007). Segundo os autores, esta definição evidencia algumas propriedades interessantes da negação $\neg$ como as leis de De Morgan.

Definição 15 (Cálculo de Hilbert para **LFI1**)**.** A lógica **LFI1** é definida a partir da relação de consequência sintática $\vdash_{\mathbf{LFI1}}$ sobre a linguagem $\mathcal{L}_\Sigma$ através do seguinte cálculo de Hilbert:
Axiomas:

$$\alpha \to (\beta \to \alpha) \tag{Ax1}$$
$$(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma)) \tag{Ax2}$$
$$\alpha \to (\beta \to (\alpha \wedge \beta)) \tag{Ax3}$$
$$(\alpha \wedge \beta) \to \alpha \tag{Ax4}$$
$$(\alpha \wedge \beta) \to \beta \tag{Ax5}$$

[1] Vale notar que a separação *prova — sintaxe — semântica — modelo* não é tão bem definida, algo que é explorado em Prawitz (2005).

$\alpha \rightarrow (\alpha \vee \beta)$ **(Ax6)**

$\beta \rightarrow (\alpha \vee \beta)$ **(Ax7)**

$(\alpha \rightarrow \gamma) \rightarrow ((\beta \rightarrow \gamma) \rightarrow ((\alpha \vee \beta) \rightarrow \gamma))$ **(Ax8)**

$(\alpha \rightarrow \beta) \vee \alpha$ **(Ax9)**

$\alpha \vee \neg\alpha$ **(Ax10)**

$\circ\alpha \rightarrow (\alpha \rightarrow (\neg\alpha \rightarrow \beta))$ **(bc1)**

$\neg\neg\alpha \rightarrow \alpha$ **(cf)**

$\alpha \rightarrow \neg\neg\alpha$ **(ce)**

$\neg\circ\alpha \rightarrow (\alpha \wedge \neg\alpha)$ **(ci)**

$\neg(\alpha \vee \beta) \rightarrow (\neg\alpha \wedge \neg\beta)$ **(neg$\vee_1$)**

$(\neg\alpha \wedge \neg\beta) \rightarrow \neg(\alpha \vee \beta)$ **(neg$\vee_2$)**

$\neg(\alpha \wedge \beta) \rightarrow (\neg\alpha \vee \neg\beta)$ **(neg$\wedge_1$)**

$(\neg\alpha \vee \neg\beta) \rightarrow \neg(\alpha \wedge \beta)$ **(neg$\wedge_2$)**

$\neg(\alpha \rightarrow \beta) \rightarrow (\alpha \wedge \neg\beta)$ **(neg$\rightarrow_1$)**

$(\alpha \wedge \neg\beta) \rightarrow \neg(\alpha \rightarrow \beta)$ **(neg$\rightarrow_2$)**

Regra de inferência:

$$\frac{\alpha \qquad \alpha \rightarrow \beta}{\beta}\ \text{MP}$$

Desta forma, a lógica **LFI1** é definida como $\mathbf{LFI1} = \langle \mathcal{L}_\Sigma, \vdash_{\mathbf{LFI1}} \rangle$. ■

Os axiomas (**Ax1**) — (**Ax10**) e a regra de inferência **MP** (*modus ponens*) são importados da lógica proposicional clássica. O axioma (**bc1**) é chamado de *princípio da explosão gentil*. Os axiomas (**neg**$\vee_1$) — (**neg**$\rightarrow_2$) expressam as leis de De Morgan em relação a negação paraconsistente $\neg$. Os axiomas (**ce**) e (**cf**) expressam, respectivamente, a introdução e a eliminação da dupla negação. O axioma (**ci**) representa a inconsistência de uma informação.

Com isso, uma derivação em **LFI1** pode ser definida:

Definição 16 (Derivação em **LFI1**)**.** Seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas, uma derivação de φ a partir de Γ em **LFI1**, denotada como $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, é uma sequência finita de fórmulas $\varphi_0, \ldots, \varphi_n$ onde, para cada $1 \leq i \leq n$, alguma das seguintes condições é satisfeita:

(i) φ_i é um axioma;

(ii) $\varphi_i \in \Gamma$;

(iii) existem $j, k < i$, de modo que φ_i é o resultado da aplicação de MP em φ_j e φ_k. ■

Para ilustrar, provaremos um exemplo de derivação no cálculo de Hilbert apresentado:

Exemplo 1. *A derivação* $\circ\psi, \alpha \to (\psi \wedge \neg\psi), \neg\alpha \to (\psi \wedge \neg\psi) \vdash_{\mathbf{LFI1}} \varphi$ *é válida.*

Prova do Exemplo 1. A seguinte derivação completa a prova:

1. $\circ\psi$ (Premissa)
2. $\alpha \to (\psi \wedge \neg\psi)$ (Premissa)
3. $\neg\alpha \to (\psi \wedge \neg\psi)$ (Premissa)
4. $\alpha \vee \neg\alpha$ (**Ax10**)
5. $(\alpha \to (\psi \wedge \neg\psi)) \to ((\neg\alpha \to (\psi \wedge \neg\psi)) \to ((\alpha \vee \neg\alpha) \to (\psi \wedge \neg\psi)))$ (**Ax8**)
6. $(\neg\alpha \to (\psi \wedge \neg\psi)) \to ((\alpha \vee \neg\alpha) \to (\psi \wedge \neg\psi))$ (MP 2, 5)
7. $(\alpha \vee \neg\alpha) \to (\psi \wedge \neg\psi)$ (MP 3, 6)
8. $\psi \wedge \neg\psi$ (MP 4, 7)
9. $(\psi \wedge \neg\psi) \to \psi$ (**Ax4**)
10. $(\psi \wedge \neg\psi) \to \neg\psi$ (**Ax5**)
11. ψ (MP 8, 9)
12. $\neg\psi$ (MP 8, 10)
13. $\circ\psi \to (\psi \to (\neg\psi \to \varphi))$ (**bc1**)
14. $\psi \to (\neg\psi \to \varphi)$ (MP 1, 13)
15. $\neg\psi \to \varphi$ (MP 11, 14)
16. φ (MP, 12, 15)

■

3.3 SEMÂNTICA

De forma geral, a semântica é o estudo de como um sistema de símbolos (uma linguagem) internaliza informações, ou seja, é o estudo de como interpretar os símbolos de uma linguagem (BROWN, 2005). Num sistema lógico, *matrizes lógicas* são comumente usadas para estabelecer o comportamento esperado dos conectivos lógicos de sua assinatura. Outra forma de compreender a semântica de um sistema lógico é definir condições que caracterizam funções conhecidas como valorações, que define uma *semântica de valorações*. Nesta seção, a semântica da **LFI1** será definida de duas formas distintas: a partir de uma *matriz lógica* e a partir de uma *semântica de valorações*. As definições e notações para os conceitos de *álgebra* definidos aqui baseiam-se nos trabalhos de Carnielli e Coniglio (2016), Sikorski (1966) e Rasiowa (1963).

3.3.1 Semântica Matricial

Uma das formas de definir a semântica de uma lógica proposicional é definir uma *matriz lógica* (também chamada de tabela-verdade) para os conectivos de sua assinatura proposicional. Para isso, é necessário definir o conceito de *álgebra* para assinaturas proposicionais:

Definição 17 (Álgebra para assinaturas proposicionais)**.** Uma álgebra para uma assinatura proposicional Θ é uma dupla $\mathcal{A} = \langle A, O \rangle$, onde A é um conjunto não vazio (chamado de *domínio* da álgebra) e O é uma função de interpretação que associa cada conectivo n-ário $c \in \Theta$ à uma operação $c^{\mathcal{A}}$: $A^n \to A$ em A. ■

Quando não for confuso, o mesmo símbolo será utilizado para representar um conectivo c e sua interpretação $O(c) = c^{\mathcal{A}}$. Além disso o símbolo utilizado para se referir a uma álgebra $\langle A, O \rangle$ será simplesmente o símbolo para seu domínio A. Ademais, caso Θ seja finita, a função O será substituída pela lista de conectivos de Θ. Por exemplo, uma álgebra para a assinatura Σ da **LFI1** é escrita como $\mathcal{A} = \langle A, \wedge, \vee, \rightarrow, \neg, \circ \rangle$.

Observação 1. Uma linguagem definida sobre uma assinatura proposicional $\Theta = \{c_1, \ldots, c_n\}$ e um conjunto enumerável de átomos $\mathcal{P} = \{p_n \mid n \in \mathbb{N}\}$ pode ser compreendida como uma álgebra da forma $\langle \mathcal{L}_\Theta, c_1 \ldots, c_n \rangle$, onde $\mathcal{L}_\Theta$ é um conjunto de fórmulas bem formadas a partir dos conectivos em Θ e dos átomos em $\mathcal{P}$. Denotamos esta álgebra simplesmente por $\mathcal{L}_\Theta$ (SIKORSKI, 1966; WÓJCICKI, 1984).

Duas álgebras $\langle A, o_1, \ldots, o_n \rangle$ e $\langle B, o'_1, \ldots, o'_n \rangle$ definidas sobre uma mesma assinatura proposicional são ditas *similares* caso, para todo $1 \leq j \leq n$, as operações o_j e o'_j tenham mesma aridade. Uma função (mapeamento) entre duas estruturas algébricas similares que preserva sua estrutura é chamada de *homomorfismo*. Ou seja, dadas duas álgebras similares $\langle A, O \rangle$, $\langle B, O' \rangle$ definidas sobre uma assinatura proposicional Θ, um mapeamento $h : A \to B$ será um *homomorfismo* caso, para todo conectivo $c \in \Theta$ de aridade n, e para todo $a_0, \ldots, a_n \in A$, tem-se $h(c(a_0, \ldots, a_n)) = c(h(a_0), \ldots, h(a_n))$.

Definição 18 (Matriz Lógica)**.** Seja Θ uma assinatura proposicional. Uma *matriz lógica* $\mathcal{M}$ definida sobre Θ é uma tripla $\mathcal{M} = \langle A, D, O \rangle$, tal que o par $\langle A, O \rangle$ é uma álgebra para Θ e D é um subconjunto de A cujos elementos são ditos *designados* (estes são elementos de A considerados como verdadeiros). ■

Com isso, uma matriz lógica $\mathcal{M} = \langle A, D, O \rangle$ induz uma relação de consequência semântica para uma lógica Tarskiana $\mathscr{L}$, definida sobre uma linguagem $\mathcal{L}_\Theta$, da seguinte forma: sendo $\Gamma \cup \{\alpha\} \in \mathcal{L}_\Theta$, tem-se $\Gamma \models_{\mathcal{M}} \alpha$ sse, para todo homomorfismo $h : \mathcal{L}_\Theta \to A$, se $h[\Gamma] \subseteq D$, então $h(\alpha) \in D$. Em particular, α é uma tautologia em $\mathscr{L}$ sse $h(\alpha) \in D$ para todo homomorfismo h. Perceba que o homomorfismo h nada mais é do que uma função que associa fórmulas da linguagem $\mathcal{L}_\Theta$ a valores do domínio A da matriz $\mathcal{M}$. Desta forma, h é dito uma *valoração* sobre $\mathcal{M}$.

Uma lógica que apresenta três elementos no domínio de sua matriz lógica é dita *trivalorada*. Algumas lógicas trivaloradas introduzem um terceiro valor além da verdade e falsidade para representar uma informação desconhecida, como é o caso da lógica de Kleene (GOTTWALD, 2022). A **LFI1**, por sua vez, introduz o valor $1/2$ além dos valores clássicos 0 e 1 para representar uma informação inconsistente. Ou seja, caso α tenha o valor $1/2$, então $\neg\alpha$ também terá o valor $1/2$.

Definição 19 (Matriz lógica da **LFI1**)**.** A matriz lógica $\mathcal{M}_{\mathbf{LFI1}} = \langle M, D, O\rangle$ com domínio $M = \{1, 1/2, 0\}$ e um conjunto de valores designados $D = \{1, 1/2\}$ é definida da seguinte forma:

$\rightarrow$	1	$1/2$	0
1	1	$1/2$	0
$1/2$	1	$1/2$	0
0	1	1	1

$\wedge$	1	$1/2$	0
1	1	$1/2$	0
$1/2$	$1/2$	$1/2$	0
0	0	0	0

$\vee$	1	$1/2$	0
1	1	1	1
$1/2$	1	$1/2$	$1/2$
0	1	$1/2$	0

	$\neg$
1	0
$1/2$	$1/2$
0	1

	$\circ$
1	1
$1/2$	0
0	1

■

A partir disso, definimos a relação de consequência semântica $\vDash_{\mathcal{M}_{\mathbf{LFI1}}}$ da seguinte forma:

Definição 20 (Relação de consequência semântica $\vDash_{\mathcal{M}_{\mathbf{LFI1}}}$)**.** Dado um conjunto de fórmulas $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$, tem-se $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi$ sse, para toda valoração $h : \mathcal{L}_\Sigma \rightarrow M$ de $\mathcal{M}_{\mathbf{LFI1}}$, se $h[\Gamma] \subseteq \{1, 1/2\}$ então $h(\varphi) \in \{1, 1/2\}$. ■

Para ilustrar, provaremos um exemplo de inferência na semântica matricial apresentada:

Exemplo 2. *A inferência* $\vDash_{\mathcal{M}_{\mathbf{LFI1}}} \circ\circ\alpha$ *é válida.*

Prova do Exemplo 2. Vamos mostrar que, para toda valoração $h : \mathcal{L}_\Sigma \rightarrow \{1, 1/2, 0\}$ de $\mathcal{M}_{\mathbf{LFI1}}$, $h(\circ\circ\alpha) \in \{1, 1/2\}$. Para isso, construiremos uma tabela com todas as valorações possíveis para $\circ\circ\alpha$:

α	$\circ\alpha$	$\circ\circ\alpha$
1	1	1
$1/2$	0	1
0	1	1

Como é possível observar, a coluna da fórmula $\circ\circ\alpha$ é composta somente por elementos pertencentes ao conjunto $\{1, 1/2\}$, portanto, para toda valoração h de $\mathcal{M}_{\mathbf{LFI1}}$, $h(\circ\circ\alpha) \in \{1, 1/2\}$. Logo $\vDash_{\mathcal{M}_{\mathbf{LFI1}}} \circ\circ\alpha$. ■

3.3.2 Bivalorações

Outra forma de definir a relação de consequência semântica de uma lógica é descrever uma semântica de valorações, esta que define um conjunto de cláusulas condicionais sobre funções (COSTA; ALVES, 1977). Quando uma função respeita estas cláusulas, ela é chamada de valoração para a lógica.

Definição 21 (Semântica de valorações para **LFI1**)**.** Uma função $v : \mathcal{L}_\Sigma \to \{1,0\}$ é uma valoração para a lógica **LFI1** caso ela satisfaça as seguintes cláusulas:

$$v(\alpha \wedge \beta) = 1 \iff v(\alpha) = 1 \text{ e } v(\beta) = 1 \qquad (vAnd)$$
$$v(\alpha \vee \beta) = 1 \iff v(\alpha) = 1 \text{ ou } v(\beta) = 1 \qquad (vOr)$$
$$v(\alpha \to \beta) = 1 \iff v(\alpha) = 0 \text{ ou } v(\beta) = 1 \qquad (vImp)$$
$$v(\neg\alpha) = 0 \Longrightarrow v(\alpha) = 1 \qquad (vNeg)$$
$$v(\circ\alpha) = 1 \Longrightarrow v(\alpha) = 0 \text{ ou } v(\neg\alpha) = 0 \qquad (vCon)$$
$$v(\neg\circ\alpha) = 1 \Longrightarrow v(\alpha) = 1 \text{ e } v(\neg\alpha) = 1 \qquad (vCi)$$
$$v(\neg\neg\alpha) = 1 \iff v(\alpha) = 1 \qquad (vDNE)$$
$$v(\neg(\alpha \wedge \beta)) = 1 \iff v(\neg\alpha) = 1 \text{ ou } v(\neg\beta) = 1 \qquad (vDM_\wedge)$$
$$v(\neg(\alpha \vee \beta)) = 1 \iff v(\neg\alpha) = 1 \text{ e } v(\neg\beta) = 1 \qquad (vDM_\vee)$$
$$v(\neg(\alpha \to \beta)) = 1 \iff v(\alpha) = 1 \text{ e } v(\neg\beta) = 1 \qquad (vCip_\to)$$

O conjunto de todas as valorações para a lógica **LFI1** será denotado por $V^{\mathbf{LFI1}}$. ■

Com isso, definimos a relação de consequência semântica $\vDash_{\mathbf{LFI1}}$ da seguinte forma:

Definição 22 (Relação de consequência semântica $\vDash_{\mathbf{LFI1}}$)**.** Dado um conjunto de fórmulas $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$, tem-se $\Gamma \vDash_{\mathbf{LFI1}} \varphi$ sse, para todo $v \in V^{\mathbf{LFI1}}$, se $v[\Gamma] \subseteq \{1\}$ então $v(\varphi) = 1$. ■

Note que, por conta de (*vNeg*) e (*vCon*), o valor $v(\triangle\alpha)$ — para $\triangle \in \{\neg, \circ\}$ — não é determinado pelo valor $v(\alpha)$ da subfórmula α. Ou seja, os conectivos $\neg$ e $\circ$ apresentam um comportamento não determinístico em relação a esta semântica de valorações. Por exemplo, caso $v(\alpha)$ seja 1, pode-se ter $v(\neg\alpha)$ tanto 0 como 1 (mas não ambos). Isto pode ser observado na seguinte tabela de possíveis valorações para $\neg\varphi$ e $\circ\varphi$, dada uma fórmula φ:

φ	$\neg\varphi$	$\circ\varphi$	
0	1	1	v_1
1	0	1	v_2
	1	0	v_3

Tabela 1 – Valorações possíveis para φ, $\neg\varphi$ e $\circ\varphi$, considerando (*vNeg*), (*vCon*) e (*vCi*).

Uma semântica de valorações que define funções que mapeiam para conjuntos com somente dois elementos é chamada de *bivaloração*. Este tipo de semântica possui características

interessantes que a distinguem da semântica matricial. A semântica de matrizes nos permite mostrar a validade de uma inferência com facilidade, contudo, alguns lógicos e filósofos demonstram descontentamento com a existência de múltiplos valores-verdade designados, argumentando que uma distinção deve ser feita entre "valorações algébricas" — correspondente às matrizes lógicas — e sua "definição genuína", em termos de bivalorações (SUSZKO, 1975). Além disso, as bivalorações servem um papel importante na teoria dos modelos, como na comparação de sistemas lógicos e na associação de outros sistemas semânticos a uma lógica já estabelecida (CALEIRO; MARCOS, 2012).

3.4 METATEOREMAS

A metalógica é uma área da matemática que estuda sistemas lógicos, desenvolvendo metateoremas (JACQUETTE, 2002). Um metateorema é uma prova sobre propriedades de um sistema formal, sobretudo sobre suas relações de consequência, utilizando uma metalinguagem (TARSKI, 1956; RASIOWA, 1963; BARILE, 2024). Estas propriedades são chamadas de *metapropriedades*.

Algumas metapropriedades importantes de serem provadas sobre um sistema lógico munido com uma relação de consequência sintática $\vdash$ e uma relação de consequência semântica $\models$ (como é o caso da **LFI1**) são a correção, que afirma que tudo que é derivável em $\vdash$ é válido em $\models$, e a completude, que afirma que tudo que é válido em $\models$ pode ser provado em $\vdash$. Outra metapropriedade relevante é a equivalência entre a semântica matricial e as bivalorações da **LFI1**, apresentados na Seção 3.3, já que ambos sistemas possuem suas particularidades e vantagens. Ademais, o metateorema da dedução (definido como $\Gamma, \alpha \vdash \beta \iff \Gamma \vdash \alpha \rightarrow \beta$) é uma propriedade interessante que possibilita a obtenção de outros resultados pertinentes, como será evidenciado ao longo desta seção. Caracterizaremos também a lógica **LFI1** como sendo uma lógica Tarskiana, finitária e **LFI** forte.

Os lemas e teoremas desenvolvidos nesta seção seguem um caminho baseado no que foi apresentado por Carnielli e Coniglio (2016) e por Costa e Alves (1977), com modificações para se adequarem ao presente trabalho.

3.4.1 Metateorema da dedução

O metateorema da dedução é uma propriedade conveniente de ser provada, já que nos fornece corolários importantes a fim de desenvolver provas de outros metateoremas.

Primeiro, algumas proposições sobre a **LFI1** que tornam o desenvolvimento do teorema da dedução mais simples serão provadas.

Proposição 1. *A lógica* $\mathbf{LFI1} = \langle \mathcal{L}_\Sigma, \vdash_{\mathbf{LFI1}} \rangle$ *é Tarskiana (Definição 6).*

Prova da Proposição 1. Queremos mostrar que a **LFI1** satisfaz as seguintes condições para todo $\Gamma \cup \Delta \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$:

(i) Se $\varphi \in \Gamma$, então $\Gamma \vdash_{\mathbf{LFI1}} \varphi$. (reflexividade)

(ii) Se $\Delta \vdash_{\mathbf{LFI1}} \varphi$ e $\Delta \subseteq \Gamma$, então $\Gamma \vdash_{\mathbf{LFI1}} \varphi$. (monotonicidade)

(iii) Se $\Delta \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma \vdash_{\mathbf{LFI1}} \delta$ para todo $\delta \in \Delta$, então $\Gamma \vdash_{\mathbf{LFI1}} \varphi$. (corte)

Prova do item (i). Vamos supor $\varphi \in \Gamma$. Então basta aplicar a premissa φ para mostrar $\Gamma \vdash_{\mathbf{LFI1}} \varphi$.

Prova do item (ii). Vamos supor $\Delta \vdash_{\mathbf{LFI1}} \varphi$ e $\Delta \subseteq \Gamma$. Então, existe uma prova $\varphi_1, \ldots, \varphi_n$ de $\varphi_n = \varphi$ a partir de Δ. Provaremos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$ por indução no tamanho n da prova.

BASE. $n = 1$. Analisando a obtenção de $\varphi = \varphi_1$, existem dois casos:

CASO 1. φ_1 é um axioma.

Neste caso, basta aplicar o mesmo axioma para mostrar $\Gamma \vdash_{\mathbf{LFI1}} \varphi_1$.

CASO 2. $\varphi_1 \in \Delta$.

Como $\Delta \subseteq \Gamma$, então temos $\varphi_1 \in \Gamma$ e, portanto, $\Gamma \vdash_{\mathbf{LFI1}} \varphi_1$ aplicando a premissa φ_1.

PASSO.

HIPÓTESE DE INDUÇÃO (HI): A propriedade segue para todo $i < n$.

Analisando a obtenção de φ_n, existem três casos:

CASO 1. φ_n é um axioma. Análogo ao caso da base.

CASO 2. $\varphi_n \in \Delta$. Análogo ao caso da base.

CASO 3. φ_n é resultado de MP em duas fórmulas φ_j, φ_k com $j, k < n$.

Neste caso, $\varphi_j = \varphi_k \to \varphi_n$ (ou $\varphi_k = \varphi_j \to \varphi_n$, a prova para este caso é análoga). Então, temos $\Delta \vdash_{\mathbf{LFI1}} \varphi_k$ e $\Delta \vdash_{\mathbf{LFI1}} \varphi_k \to \varphi_n$. Pela (HI), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi_k$ e $\Gamma \vdash_{\mathbf{LFI1}} \varphi_k \to \varphi_n$. Portanto, basta aplicar MP para obter $\Gamma \vdash_{\mathbf{LFI1}} \varphi_n$.

Prova do item (iii). Vamos supor $\Delta \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma \vdash_{\mathbf{LFI1}} \delta$ para todo $\delta \in \Delta$. Então, existe uma prova $\varphi_1, \ldots, \varphi_n$ de $\varphi_n = \varphi$ a partir de Δ. Vamos fazer uma indução no tamanho n desta prova.

BASE. $n = 1$. Analisando a obtenção de $\varphi = \varphi_1$, existem dois casos:

CASO 1. φ_1 é um axioma.

Neste caso, basta aplicar o mesmo axioma para mostrar $\Gamma \vdash_{\mathbf{LFI1}} \varphi_1$.

CASO 2. $\varphi_1 \in \Delta$.

Neste caso, $\Gamma \vdash_{\mathbf{LFI1}} \varphi_1$ por suposição.

PASSO.

HIPÓTESE DE INDUÇÃO (HI): A propriedade segue para todo $i < n$.

Analisando a obtenção de $\varphi = \varphi_n$, existem três casos:

CASO 1. φ_n é um axioma. Análogo ao caso da base.

CASO 2. $\varphi_n \in \Delta$. Análogo ao caso da base.

CASO 3. φ_n é resultado de MP em duas fórmulas φ_j, φ_k com $j,k < n$.

Então, $\varphi_j = \varphi_k \to \varphi_n$ (ou $\varphi_k = \varphi_j \to \varphi_n$, a prova para este caso é análoga). Além disso, temos $\Delta \vdash_{\mathbf{LFI1}} \varphi_k$ e $\Delta \vdash_{\mathbf{LFI1}} \varphi_k \to \varphi_n$. Logo, por (HI), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi_k$ e $\Gamma \vdash_{\mathbf{LFI1}} \varphi_k \to \varphi_n$. Então, basta aplicar MP para obter φ_n. Portanto, $\Gamma \vdash_{\mathbf{LFI1}} \varphi_n$.

Então, a **LFI1** é uma lógica Tarskiana. ■

Proposição 2. *A lógica* $\mathbf{LFI1} = \langle \mathcal{L}_\Sigma, \vdash_{\mathbf{LFI1}} \rangle$ *é finitária, ou seja, para todo* $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$,

Se $\Gamma \vdash_{\mathbf{LFI1}} \alpha$, *então existe conjunto finito* $\Gamma_0 \subseteq \Gamma$ *tal que* $\Gamma_0 \vdash_{\mathbf{LFI1}} \alpha$.

Prova da Proposição 2. Seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto qualquer de fórmulas com $\Gamma \vdash_{\mathbf{LFI1}} \varphi$. Então, existe uma sequência finita de derivação $\varphi_1, \ldots, \varphi_n$, a partir de Γ, onde $\varphi_n = \varphi$.

Vamos definir o conjunto Γ_0 como sendo o conjunto formado somente pelas premissas desta derivação (logo, Γ_0 é finito). Como todo elemento de Γ_0 é uma premissa de Γ, é evidente que $\Gamma_0 \subseteq \Gamma$. Vamos provar $\Gamma_0 \vdash_{\mathbf{LFI1}} \varphi$ por indução no tamanho n da sequência de derivação.

BASE. $n = 1$. Analisando a obtenção de φ_1 em $\Gamma \vdash_{\mathbf{LFI1}} \varphi_1$, existem dois casos:

CASO 1. φ_1 é um axioma.

Neste caso, basta aplicar o mesmo axioma para mostrar $\Gamma_0 \vdash_{\mathbf{LFI1}} \varphi_1$.

CASO 2. $\varphi_1 \in \Gamma$.

Neste caso, $\Gamma_0 \vdash_{\mathbf{LFI1}} \varphi_1$ pela construção de Γ_0.

PASSO.

HIPÓTESE DE INDUÇÃO (HI): A propriedade segue para todo $i < n$.

Analisando a obtenção de φ_n em $\Gamma \vdash_{\mathbf{LFI1}} \varphi_n$, existem três casos:

CASO 1. φ_n é um axioma. Análogo ao caso da base.

CASO 2. $\varphi_n \in \Gamma$. Análogo ao caso da base.

CASO 3. φ_n é resultado de MP em duas fórmulas φ_j, φ_k com $j,k < n$.

Então, $\varphi_j = \varphi_k \to \varphi_n$ (ou $\varphi_k = \varphi_j \to \varphi_n$, a prova para este caso é análoga). Além disso, temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi_k$ e $\Gamma \vdash_{\mathbf{LFI1}} \varphi_k \to \varphi_n$. Logo, por (HI), temos $\Gamma_0 \vdash_{\mathbf{LFI1}} \varphi_k$ e $\Gamma_0 \vdash_{\mathbf{LFI1}} \varphi_k \to \varphi_n$. Então, basta aplicar MP para obter φ_n. Portanto, $\Gamma_0 \vdash_{\mathbf{LFI1}} \varphi_n$.

Então, a **LFI1** é finitária. ■

Teorema 1 (Metateorema da dedução para $\vdash_{\mathbf{LFI1}}$)**.** *Para todo conjunto de fórmulas* $\Gamma \cup \{\alpha, \beta\} \subseteq \mathcal{L}_\Sigma$*:*

$$\Gamma, \alpha \vdash_{\mathbf{LFI1}} \beta \iff \Gamma \vdash_{\mathbf{LFI1}} \alpha \to \beta.$$

O seguinte lema torna a prova do teorema mais imediata:

Lema 1. *A derivação* $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \alpha$ *é válida para todo* $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$.

Prova do Lema 1. A seguinte sequência de derivação demonstra o lema:

1. $(\alpha \to ((\alpha \to \alpha) \to \alpha)) \to ((\alpha \to (\alpha \to \alpha)) \to (\alpha \to \alpha))$ (**Ax2**)
2. $\alpha \to ((\alpha \to \alpha) \to \alpha)$ (**Ax1**)
3. $\alpha \to (\alpha \to \alpha)$ (**Ax1**)
4. $(\alpha \to (\alpha \to \alpha)) \to (\alpha \to \alpha)$ (MP 1,2)
5. $\alpha \to \alpha$ (MP 3,4)

■

Prova do Teorema 1. A prova será dividida em duas partes:
($\Longleftarrow$) Supondo $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \beta$, então existe uma sequência de derivação $\varphi_1 \ldots \varphi_n$ onde $\varphi_n = \alpha \to \beta$. Temos $\Gamma \subseteq \Gamma \cup \{\alpha\}$, logo, pela monotonicidade da **LFI1**, $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \alpha \to \beta$.

A seguinte sequência de derivação completa a prova de $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \beta$:

1. $\ldots$
$\vdots$ $\ddots$
n. $\alpha \to \beta$ (Resultado da suposição)
$n+1$. α (Premissa)
$n+2$. β (MP $n, n+1$)

($\Longrightarrow$) Supondo $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \beta$, então existe uma sequência de derivação $\varphi_1 \ldots \varphi_n$ onde $\varphi_n = \beta$ a partir do conjunto de premissas $\Gamma \cup \{\alpha\}$. A prova de $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \beta$ é feita pela indução no tamanho n da sequência de derivação:

BASE. $n = 1$. A sequência contém somente uma fórmula $\varphi_1 = \beta$. Portanto, existem duas possibilidades:

1. φ_1 é um axioma.
2. $\varphi_1 \in \Gamma \cup \{\alpha\}$.

CASO 1. φ_1 é um axioma.

A seguinte derivação mostra $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \varphi_1$:

1. φ_1 (Axioma)
2. $\varphi_1 \to (\alpha \to \varphi_1)$ (**Ax1**)
3. $\alpha \to \varphi_1$ (MP 1,2)

CASO 2. $\varphi_1 \in \Gamma \cup \{\alpha\}$.

Existem dois casos a serem considerados:

2.1 $\varphi_1 = \alpha$

2.2 $\varphi_1 \in \Gamma$

SUBCASO 2.1. $\varphi_1 = \alpha$.

É necessário mostrar $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \alpha$, o que foi provado pelo Lema 1.

SUBCASO 2.2. $\varphi_1 \in \Gamma$.

Então $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \varphi_1$ é provado pela seguinte sequência de derivações:

1. φ_1 (Premissa)
2. $\varphi_1 \to (\alpha \to \varphi_1)$ (**Ax1**)
3. $\alpha \to \varphi_1$ (MP 1, 2)

Portanto, $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \varphi_1$ segue para o caso base.

PASSO.

HIPÓTESE DE INDUÇÃO (HI): Para qualquer sequência de derivação $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \beta$ de tamanho i, com $i < n$, tem-se $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \beta$.

É preciso mostrar que $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \beta$ segue caso a dedução $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \beta$ seja de tamanho n. Então, vamos supor $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi_n$ e mostrar $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \varphi_n$.

Ao analisar a obtenção de φ_n na sequência de derivação de $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi_n$, existem três casos a se considerar:

1. φ_n é um axioma.
2. $\varphi_n \in \Gamma \cup \{\alpha\}$.
3. φ_n é obtido por *modus ponens* em duas fórmulas φ_j e φ_k com $j, k < n$.

Os casos 1 e 2 são análogos aos casos provados na base.

CASO 3. φ_n é obtido por *modus ponens* em duas fórmulas φ_j e φ_k com $j, k < n$.

Então, $\varphi_k = \varphi_j \to \varphi_n$ (ou $\varphi_j = \varphi_k \to \varphi_n$, a prova para este caso é análoga).

Dada a nossa suposição de $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi_n$, temos $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi_j$ e $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi_j \to \varphi_n$ sequências de dedução anteriores à aplicação da regra do *modus ponens* na linha n.

Então, pela (HI), temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \varphi_j$ e $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to (\varphi_j \to \varphi_n)$.

A seguinte sequência de derivação mostra $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \varphi_n$:

1. $\ldots$
$\vdots$ $\ddots$
j. $\alpha \to \varphi_j$ (HI sobre φ_j)
$\vdots$ $\ddots$
$j+k$. $\alpha \to (\varphi_j \to \varphi_n)$ (HI sobre φ_k)

$j+k+1.\ (\alpha \to (\varphi_j \to \varphi_n)) \to ((\alpha \to \varphi_j) \to (\alpha \to \varphi_n))$ **(Ax2)**

$j+k+2.\ (\alpha \to \varphi_j) \to (\alpha \to \varphi_n)$ (MP $j+k, j+k+1$)

$j+k+3.\ \alpha \to \varphi_n$ (MP $j, j+k+2$)

Portanto, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \beta$ e a prova está finalizada. ■

Com a prova do metateorema da dedução para o cálculo de Hilbert da **LFI1**, os seguintes corolários são imediatos:

Corolário 1. *Para todo conjunto de fórmulas* $\Gamma \cup \{\alpha, \beta, \varphi\} \subseteq \mathcal{L}_\Sigma$*:*

Se $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma, \beta \vdash_{\mathbf{LFI1}} \varphi$ então $\Gamma, \alpha \vee \beta \vdash_{\mathbf{LFI1}} \varphi$.

Prova do Corolário 1. Seja $\Gamma \cup \{\alpha, \beta, \varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas qualquer.

Suponha $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma, \beta \vdash_{\mathbf{LFI1}} \varphi$. Então, por MTD (Teorema 1), temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \varphi$ e $\Gamma \vdash_{\mathbf{LFI1}} \beta \to \varphi$.

A seguinte sequência de derivação mostra $\Gamma, \alpha \vee \beta \vdash_{\mathbf{LFI1}} \varphi$:

1. $\alpha \to \varphi$ (MTD aplicado à suposição)
2. $\beta \to \varphi$ (MTD aplicado à suposição)
3. $\alpha \vee \beta$ (Premissa)
4. $(\alpha \to \varphi) \to ((\beta \to \varphi) \to ((\alpha \vee \beta) \to \varphi))$ **(Ax8)**
5. $(\beta \to \varphi) \to ((\alpha \vee \beta) \to \varphi)$ (MP 1, 4)
6. $(\alpha \vee \beta) \to \varphi$ (MP 2, 5)
7. φ (MP 3, 6)

■

Corolário 2. *Para todo conjunto de fórmulas* $\Gamma \cup \{\alpha, \beta, \varphi\} \subseteq \mathcal{L}_\Sigma$*:*

Se $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma, \neg\alpha \vdash_{\mathbf{LFI1}} \varphi$ então $\Gamma \vdash_{\mathbf{LFI1}} \varphi$.

Prova do Corolário 2. Seja $\Gamma \cup \{\alpha, \beta, \varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas qualquer.

Suponha $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma, \neg\alpha \vdash_{\mathbf{LFI1}} \varphi$. Pelo Corolário 1, temos $\Gamma, \alpha \vee \neg\alpha \vdash_{\mathbf{LFI1}} \varphi$. Por MTD (Teorema 1), temos $\Gamma \vdash_{\mathbf{LFI1}} (\alpha \vee \neg\alpha) \to \varphi$.

A seguinte sequência de derivação mostra $\Gamma \vdash_{\mathbf{LFI1}} \varphi$:

1. $(\alpha \vee \neg\alpha) \to \varphi$ (MTD aplicado à suposição)
2. $\alpha \vee \neg\alpha$ **(Ax10)**
3. φ (MP 1,2)

■

3.4.2 Correção

A metapropriedade da correção garante que o sistema dedutivo de uma determinada lógica prova somente sentenças válidas de acordo com sua semântica. Esta é uma característica fundamental para qualquer sistema lógico que se preste.

Antes de provar a correção, todavia, provaremos primeiro o seguinte lema que facilitará a obtenção dos resultados pertinentes:

Lema 2. *Seja v uma valoração em $V^{\mathbf{LFI1}}$, então existe uma valoração h sobre $\mathcal{M}_{\mathbf{LFI1}}$ tal que, para todo $\varphi \in \mathcal{L}_\Sigma$, $v(\varphi) = 1$ sse $h(\varphi) \in \{1, 1/2\}$.*

Prova do Lema 2. Seja v uma valoração em $V^{\mathbf{LFI1}}$. Então, considere a valoração h sobre $\mathcal{M}_{\mathbf{LFI1}}$ tal que para toda variável proposicional p:

$$h(p) = \begin{cases} 1 & \text{sse } v(p) = 1 \text{ e } v(\neg p) = 0 \\ 1/2 & \text{sse } v(p) = 1 \text{ e } v(\neg p) = 1 \\ 0 & \text{sse } v(p) = 0 \end{cases}$$

Então, por indução na complexidade de uma fórmula $\varphi \in \mathcal{L}_\Sigma$, será provado o seguinte:

$$h(\varphi) = \begin{cases} 1 & \text{sse } v(\varphi) = 1 \text{ e } v(\neg\varphi) = 0 \\ 1/2 & \text{sse } v(\varphi) = 1 \text{ e } v(\neg\varphi) = 1 \\ 0 & \text{sse } v(\varphi) = 0 \end{cases}$$

Notação 4. Antes de desenvolver o lema, convém estabelecer uma notação para as justificativas de cada passo da prova:

- (Matriz de $\triangle$) refere-se à matriz de um conectivo $\triangle \in \Sigma$ pertencente a $\mathcal{M}_{\mathbf{LFI1}}$;
- (HI) refere-se à aplicação da hipótese de indução;
- (Dist. $\otimes, \oplus$) refere-se à aplicação da distributividade de $\otimes$ sobre $\oplus$, ou seja, a aplicação do teorema $a \otimes (b \oplus c) \Longleftrightarrow (a \otimes b) \oplus (a \otimes c)$;
- (Idemp. $\otimes$) refere-se à aplicação da idempotência de $\otimes$, ou seja, a aplicação do teorema $a \otimes a \Longrightarrow a$.

BASE. $C(\varphi) = 1$.

Como $C(\varphi) = 1$, temos $\varphi \in \mathcal{P}$ pela Definição 14. Com isso, a construção de h prova o caso base.

PASSO.

HIPÓTESE DE INDUÇÃO (HI): A propriedade vale para toda fórmula φ tal que $C(\varphi) = k$, com $k < n$.

Vamos mostrar que a propriedade vale caso $C(\varphi) = n$.

CASO 1. $\varphi = \neg\alpha$.

SUBCASO 1.1. $h(\varphi) = 1$ sse $v(\varphi) = 1$ e $v(\neg\varphi) = 0$.

($\Longrightarrow$) Supondo $h(\varphi) = h(\neg\alpha) = \neg h(\alpha) = 1$, vamos provar $v(\varphi) = 1$ e $v(\neg\varphi) = 0$.

Pela matriz de $\neg$ temos $h(\alpha) = 0$. Então, por (HI), $v(\alpha) = 0$.

Por (*vNeg*) temos $v(\neg\alpha) = v(\varphi) = 1$. Enfim, por (*vDNE*), $v(\neg\varphi) = v(\neg\neg\alpha) = v(\alpha) = 0$

($\Longleftarrow$) Supondo $v(\varphi) = v(\neg\alpha) = 1$ e $v(\neg\varphi) = v(\neg\neg\alpha) = 0$, vamos provar $h(\varphi) = h(\neg\alpha) = 1$.

Por (*vDNE*) temos $v(\neg\neg\alpha) = v(\alpha) = 0$.

Por (HI) temos $h(\alpha) = 0$. Então, $h(\varphi) = h(\neg\alpha) = \neg h(\alpha) = 1$, pela matriz de $\neg$.

SUBCASO 1.2. $h(\varphi) = 1/2$ sse $v(\varphi) = 1$ e $v(\neg\varphi) = 1$.

($\Longrightarrow$) Supondo $h(\varphi) = h(\neg\alpha) = 1/2$, vamos provar $v(\varphi) = 1$ e $v(\neg\varphi) = 1$.

Pela matriz de $\neg$ temos $h(\alpha) = 1/2$. Então, por (HI), $v(\alpha) = 1$ e $v(\neg\alpha) = v(\varphi) = 1$.

Enfim, por (*vDNE*) temos $v(\neg\varphi) = v(\neg\neg\alpha) = v(\alpha) = 1$.

($\Longleftarrow$) Supondo $v(\varphi) = v(\neg\alpha) = 1$ e $v(\neg\varphi) = v(\neg\neg\alpha) = 1$, vamos provar $h(\varphi) = 1/2$.

Por (*vDNE*) temos $v(\alpha) = v(\neg\neg\alpha) = 1$.

Por (HI), temos $h(\alpha) = 1/2$. Então, pela matriz de $\neg$, temos $h(\varphi) = h(\neg\alpha) = \neg h(\alpha) = 1/2$.

SUBCASO 1.3. $h(\varphi) = 0$ sse $v(\varphi) = 0$.

($\Longrightarrow$) Supondo $h(\varphi) = h(\neg\alpha) = \neg h(\alpha) = 0$, vamos provar $v(\varphi) = v(\neg\alpha) = 0$.

Pela matriz de $\neg$ temos $h(\alpha) = 1$. Então, por (HI), $v(\alpha) = 1$ e $v(\varphi) = v(\neg\alpha) = 0$.

($\Longleftarrow$) Supondo $v(\varphi) = v(\neg\alpha) = 0$, vamos provar $h(\varphi) = h(\neg\alpha) = 0$.

Por (*vNeg*) temos $v(\alpha) = 1$. Logo, por (HI), temos $h(\alpha) = 1$.

Pela matriz de $\neg$ temos $h(\varphi) = h(\neg\alpha) = \neg h(\alpha) = 0$.

CASO 2. $\varphi = \circ\alpha$.

SUBCASO 2.1. $h(\varphi) = 1$ sse $v(\varphi) = 1$ e $v(\neg\varphi) = 0$.

($\Longrightarrow$) Supondo $h(\varphi) = h(\circ\alpha) = \circ h(\alpha) = 1$, vamos provar $v(\varphi) = 1$ e $v(\neg\varphi) = 0$.

Pela matriz de $\circ$, temos $h(\alpha) = 1$ ou $h(\alpha) = 0$.

SUBSUBCASO 2.1.1. $h(\alpha) = 1$.

Por (HI) temos $v(\alpha) = 1$ e $v(\neg\alpha) = 0$.

Por (*vCi*) temos $v(\neg\varphi) = v(\neg\circ\alpha) = 0$. Então, por (*vNeg*), temos $v(\varphi) = v(\circ\alpha) = 1$.

SUBSUBCASO 2.1.2. $h(\alpha) = 0$.

Por (HI) temos $v(\alpha) = 0$.

Por (*vCi*), temos $v(\neg\varphi) = v(\neg \circ \alpha) = 0$. Então, por (*vNeg*) temos $v(\varphi) = 1$.

($\Longleftarrow$) Supondo $v(\varphi) = v(\circ\alpha) = 1$ e $v(\neg\varphi) = v(\neg \circ \alpha) = 0$, vamos provar $h(\varphi) = h(\circ\alpha) = 1$.

Por (*vCon*) temos $v(\alpha) = 0$ ou $v(\neg\alpha) = 0$.

SUBSUBCASO 2.1.3. $v(\alpha) = 0$.

Por (HI) temos $h(\alpha) = 0$. Então, pela matriz de $\circ$, temos $h(\varphi) = h(\circ\alpha) = \circ h(\alpha) = 1$.

SUBSUBCASO 2.1.4. $v(\neg\alpha) = 0$.

Por (*vNeg*) temos $v(\alpha) = 1$. Então, por (HI), temos $h(\alpha) = 1$.

Finalmente, pela matriz de $\circ$, temos $h(\varphi) = h(\circ\alpha) = \circ h(\alpha) = 1$.

SUBCASO 2.2. $h(\varphi) = 1/2$ sse $v(\varphi) = 1$ e $v(\neg\varphi) = 1$.

($\Longrightarrow$) Pela matriz lógica de $\circ$, nunca temos $h(\varphi) = h(\circ\alpha) = 1/2$.

($\Longleftarrow$) Supondo $v(\varphi) = 1$ e $v(\neg\varphi) = 1$, então, por (*vCon*) aplicado a $v(\varphi) = v(\alpha) = 1$, temos $v(\alpha) = 0$ ou $v(\neg\alpha) = 0$.

Entretanto, por (*vCi*) aplicado a $v(\neg\varphi) = v(\neg\alpha) = 1$, temos $v(\alpha) = 1$ e $v(\neg\alpha) = 1$.

Portanto, nunca temos $v(\varphi) = 1$ e $v(\neg\varphi) = 1$.

SUBCASO 2.3. $h(\varphi) = 0$ sse $v(\varphi) = 0$.

($\Longrightarrow$) Supondo $h(\varphi) = h(\circ\alpha) = \circ h(\alpha) = 0$, vamos provar $v(\varphi) = v(\circ\alpha) = 0$.

Pela matriz de $\circ$, temos $h(\alpha) = 1/2$. Então, por (HI), temos $v(\alpha) = 1$ e $v(\neg\alpha) = 1$.

Finalmente, por (*vCon*), temos $v(\varphi) = v(\circ\alpha) = 0$.

($\Longleftarrow$) Supondo $v(\varphi) = v(\circ\alpha) = 0$, vamos provar $h(\varphi) = h(\circ\alpha) = 0$.

Por (*vDNE*) temos $v(\neg\neg \circ \alpha) = 0$. Então, por (*vNeg*), temos $v(\neg \circ \alpha) = 1$.

Por (*vCi*) temos $v(\alpha) = 1$ e $v(\neg\alpha) = 1$. Logo, por (HI), temos $h(\alpha) = 1/2$.

Finalmente, pela matriz de $\circ$, temos $h(\varphi) = h(\circ\alpha) = \circ h(\alpha) = 0$.

CASO 3. $\varphi = \alpha \wedge \beta$.

SUBCASO 3.1. $h(\varphi) = h(\alpha \wedge \beta) = h(\alpha) \wedge h(\beta) = 1 \Longleftrightarrow v(\varphi) = v(\alpha \wedge \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \wedge \beta)) = 0$.

$$
\begin{aligned}
&\text{Temos } h(\varphi) = h(\alpha \wedge \beta) = h(\alpha) \wedge h(\beta) = 1 \\
&\Longleftrightarrow h(\alpha) = 1 \text{ e } h(\beta) = 1 && \text{(Matriz de } \wedge\text{)} \\
&\Longleftrightarrow v(\alpha) = 1, v(\neg\alpha) = 0, v(\beta) = 1 \text{ e } v(\neg\beta) = 0 && \text{(HI)} \\
&\Longleftrightarrow v(\varphi) = v(\alpha \wedge \beta) = 1 \text{ e } v(\neg\varphi) = v(\neg(\alpha \wedge \beta)) = 0 && \text{((}vAnd\text{) e (}vDM_{\wedge}\text{))}
\end{aligned}
$$

SUBCASO 3.2. $h(\varphi) = h(\alpha \wedge \beta) = h(\alpha) \wedge h(\beta) = 1/2 \iff v(\varphi) = v(\alpha \wedge \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \wedge \beta)) = 1$.

Temos $h(\varphi) = h(\alpha \wedge \beta) = h(\alpha) \wedge h(\beta) = 1/2$
$\iff (h(\alpha), h(\beta) \in \{1, 1/2\})$ e $(h(\alpha) = 1/2$ ou $h(\beta) = 1/2)$ (Matriz de $\wedge$)
$\iff (v(\alpha) = 1$ e $v(\beta) = 1)$ e
$(v(\alpha) = v(\neg\alpha) = 1$ ou $v(\beta) = v(\neg\beta) = 1)$ (HI)
$\iff (v(\alpha) = 1, v(\neg\alpha) = 1$ e $v(\beta) = 1)$ ou
$(v(\alpha) = 1, v(\neg\beta) = 1$ e $v(\beta) = 1)$ (Dist. $\wedge$, $\vee$ e Idemp. $\wedge$)
$\iff (v(\alpha) = 1$ e $v(\beta) = 1)$ e
$(v(\neg\alpha) = 1$ ou $v(\neg\beta) = 1)$ (Dist. $\wedge$, $\vee$ e Idemp. $\wedge$)
$\iff (v(\varphi) = v(\alpha \wedge \beta) = 1)$ e
$(v(\neg\varphi) = v(\neg(\alpha \wedge \beta)) = 1)$ ($(vAnd)$ e $(vDM_{\wedge})$)

SUBCASO 3.3. $h(\varphi) = h(\alpha \wedge \beta) = h(\alpha) \wedge h(\beta) = 0 \iff v(\varphi) = v(\alpha \wedge \beta) = 0$.

Temos $h(\varphi) = h(\alpha \wedge \beta) = h(\alpha) \wedge h(\beta) = 0$
$\iff h(\alpha) = 0$ ou $h(\beta) = 0$ (Matriz de $\wedge$)
$\iff v(\alpha) = 0$ ou $v(\beta) = 0$ (HI)
$\iff v(\varphi) = v(\alpha \wedge \beta) = 0$ ($vAnd$)

CASO 4. $\varphi = \alpha \vee \beta$.

SUBCASO 4.1. $h(\varphi) = h(\alpha \vee \beta) = h(\alpha) \vee h(\beta) = 1 \iff v(\varphi) = v(\alpha \vee \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \vee \beta)) = 0$.

Temos $h(\varphi) = h(\alpha \vee \beta) = h(\alpha) \vee h(\beta) = 1$
$\iff h(\alpha) = 1$ ou $h(\beta) = 1$ (Matriz de $\vee$)
$\iff (v(\alpha) = 1$ e $v(\neg\alpha) = 0)$ ou $(v(\beta) = 1$ e $v(\neg\beta) = 0)$ (HI)
$\iff (((v(\alpha) = 1$ e $v(\neg\alpha) = 0))$ ou $v(\beta) = 1)$ e
$(((v(\alpha) = 1$ e $v(\neg\alpha) = 0))$ ou $v(\neg\beta) = 0)$ (Dist. $\vee$, $\wedge$)
$\iff ((v(\alpha) = 1$ ou $v(\beta) = 1)$ e $(v(\neg\alpha) = 0$ ou $v(\beta) = 1))$ e
$((v(\alpha) = 1$ ou $v(\neg\beta) = 0)$ e $(v(\neg\alpha) = 0$ ou $v(\neg\beta) = 0))$ (Dist. $\vee$, $\wedge$)
$\iff ((v(\alpha) = 1$ ou $v(\beta) = 1)$ e $(v(\alpha) = 1$ ou $v(\beta) = 1))$ e
$((v(\alpha) = 1$ ou $v(\beta) = 1)$ e $(v(\neg\alpha) = 0$ ou $v(\neg\beta) = 0))$ ($vNeg$)
$\iff ((v(\alpha) = 1$ ou $v(\beta) = 1)$ e $(v(\neg\alpha) = 0$ ou $v(\neg\beta) = 0))$ (Idemp. $\wedge$)
$\iff v(\varphi) = v(\alpha \vee \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \vee \beta)) = 0$. ($(vOr)$ e $(vDM_{\vee})$)

SUBCASO 4.2. $h(\varphi) = h(\alpha \vee \beta) = h(\alpha) \vee h(\beta) = 1/2 \iff v(\varphi) = v(\alpha \vee \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \vee \beta)) = 1$.

Temos $h(\varphi) = h(\alpha \vee \beta) = h(\alpha) \vee h(\beta) = 1/2$
$\iff (h(\alpha) = h(\beta) = 1/2)$ ou $(h(\alpha) = 1/2$ e $h(\beta) = 0)$ ou
$(h(\alpha) = 0$ e $h(\beta) = 1/2)$. (Matriz de $\vee$)
$\iff (v(\alpha) = 1, v(\neg\alpha) = 1, v(\beta) = 1$ e $v(\neg\beta) = 1)$ ou
$(v(\alpha) = 1, v(\neg\alpha) = 1$ e $v(\beta) = 0)$ ou
$(v(\alpha) = 0, v(\beta) = 1$ e $v(\neg\beta) = 1)$ (HI)
$\iff (v(\alpha) = 1, v(\neg\alpha) = 1, v(\beta) = 1$ e $v(\neg\beta) = 1)$ ou
$(v(\alpha) = 1, v(\neg\alpha) = 1, v(\beta) = 0$ e $v(\neg\beta) = 1)$ ou
$(v(\alpha) = 0, v(\neg\alpha) = 1, v(\beta) = 1, v(\neg\beta) = 1)$ ($vNeg$)
$\iff (v(\neg\alpha) = 1$ e $v(\neg\beta) = 1)$ e $((v(\alpha) = 1$ e $v(\beta) = 1)$ ou
$(v(\alpha) = 1$ e $v(\beta) = 0)$ ou $(v(\alpha) = 0$ e $v(\beta) = 1))$. (Dist. $\wedge$, $\vee$)
$\iff (v(\neg\alpha) = 1$ e $v(\neg\beta) = 1)$ e
$(v(\alpha \vee \beta) = 1$ ou $v(\alpha \vee \beta) = 1$ ou $v(\alpha \vee \beta) = 1)$ (vOr)
$\iff v(\varphi) = v(\alpha \vee \beta) = 1$ e
$v(\neg\varphi) = v(\neg(\alpha \vee \beta)) = 1$. (($vDM_{\vee}$) e (Idemp. $\vee$))

SUBCASO 4.3. $h(\varphi) = h(\alpha \vee \beta) = h(\alpha) \vee h(\beta) = 0$ sse $v(\varphi) = v(\alpha \vee \beta) = 0$.

Temos $h(\varphi) = h(\alpha \vee \beta) = h(\alpha) \vee h(\beta) = 0$
$\iff h(\alpha) = 0$ e $h(\beta) = 0$ (Matriz de $\vee$)
$\iff v(\alpha) = 0$ e $v(\beta) = 0$ (HI)
$\iff v(\varphi) = v(\alpha \vee \beta) = 0$ (vOr)

CASO 5. $\varphi = \alpha \to \beta$.

SUBCASO 5.1. $h(\varphi) = h(\alpha \to \beta) = h(\alpha) \to h(\beta) = 1 \iff v(\varphi) = v(\alpha \to \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \to \beta)) = 0$.

Temos $h(\varphi) = h(\alpha \to \beta) = h(\alpha) \to h(\beta) = 1$
$\iff h(\alpha) = 0$ ou $h(\beta) = 1$ (Matriz de $\to$)
$\iff v(\alpha) = 0$ ou $(v(\beta) = 1$ e $v(\neg\beta) = 0)$ (HI)
$\iff (v(\alpha \to \beta) = 1$ e $v(\neg(\alpha \to \beta)) = 0)$ ou
$(v(\alpha \to \beta) = 1$ e $v(\neg(\alpha \to \beta)) = 0)$ (($vImp$) e ($vCip_{\to}$))
$\iff v(\varphi) = v(\alpha \to \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \to \beta)) = 0$ (Idemp. $\vee$)

SUBCASO 5.2. $h(\varphi) = h(\alpha \to \beta) = h(\alpha) \to h(\beta) = 1/2 \iff v(\varphi) = v(\alpha \to \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \to \beta)) = 1$.

Temos $h(\varphi) = h(\alpha \to \beta) = h(\alpha) \to h(\beta) = 1/2$
$\iff (h(\alpha) = 1$ e $h(\beta) = 1/2)$ ou
$(h(\alpha) = 1/2$ e $h(\beta) = 1/2)$ (Matriz de $\to$)
$\iff (v(\alpha) = v(\beta) = v(\neg\beta) = 1$ e $v(\neg\alpha) = 0)$ ou
$(v(\alpha) = v(\neg\alpha) = v(\beta) = v(\neg\beta) = 1)$ (HI)
$\iff v(\alpha \to \beta) = v(\neg(\alpha \to \beta)) = 1$ e
$v(\alpha \to \beta) = v(\neg(\alpha \to \beta)) = 1$ ($(vImp)$ e $(vCip_{\to})$)
$\iff v(\varphi) = v(\alpha \to \beta) = 1$ e $v(\neg\varphi) = v(\neg(\alpha \to \beta)) = 1$ (Idemp. $\vee$)

SUBCASO 5.3. $h(\varphi) = h(\alpha \to \beta) = h(\alpha) \to h(\beta) = 0$ sse $v(\varphi) = v(\alpha \to \beta) = 0$.

Temos $h(\varphi) = h(\alpha \to \beta) = h(\alpha) \to h(\beta) = 0$
$\iff (h(\alpha) = 1$ e $h(\beta) = 0)$ ou $(h(\alpha) = 1/2$ e $h(\beta) = 0)$ (Matriz de $\to$)
$\iff (v(\alpha) = 1$ e $v(\neg\alpha) = v(\beta) = 0)$ ou
$(v(\alpha) = v(\neg\alpha) = 1$ e $v(\beta) = 0)$ (HI)
$\iff v(\alpha \to \beta) = 0$ ou $v(\alpha \to \beta) = 0$ $(vImp)$
$\iff v(\alpha \to \beta) = 0.$ (Idemp. $\vee$)

■

Corolário 3. *Para todo conjunto de fórmulas* $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_{\Sigma}$*:*

$$\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi \Longrightarrow \Gamma \vDash_{\mathbf{LFI1}} \varphi.$$

Prova do Corolário 3. Seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_{\Sigma}$ um conjunto de fórmulas e seja $v \in V^{\mathbf{LFI1}}$ uma valoração. Vamos supor $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi$.

Pelo Lema 2, então existe uma valoração h tal que, para todo $\psi \in \mathcal{L}_{\Sigma}$, $v(\psi) = 1$ sse $h(\psi) \subseteq \{1, 1/2\}$. Supondo $v[\Gamma] = 1$, então temos $h[\Gamma] \subseteq \{1, 1/2\}$. Pela nossa suposição de $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi$, temos $h(\varphi) \in \{1, 1/2\}$. Novamente, pelo Lema 2, temos $v(\varphi) = 1$. Portanto, segue $\Gamma \vDash_{\mathbf{LFI1}} \varphi$.

■

A ideia da prova da correção para a **LFI1** é, como de costume, provar que todo axioma do cálculo de Hilbert $\vdash_{\mathbf{LFI1}}$ é válido na semântica matricial $\vDash_{\mathcal{M}_{\mathbf{LFI1}}}$ e mostrar que a regra de inferência *modus ponens* preserva sua validade. No desenvolvimento da prova, as igualdades do tipo $h(\triangle\varphi) = \triangle h(\varphi)$ e $h(\varphi \otimes \psi) = h(\varphi) \otimes h(\psi)$, onde $\triangle, \otimes$ são conectivos quaisquer, ficam implícitas.

Teorema 2 (Correção em relação à semântica matricial)**.** *A lógica* **LFI1** *é correta em relação a sua semântica matricial, ou seja, para todo conjunto de fórmulas* $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_{\Sigma}$*:*

$$\Gamma \vdash_{\mathbf{LFI1}} \alpha \Longrightarrow \Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \alpha.$$

Prova do Teorema 2. Seja $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas. Supondo $\Gamma \vdash_{\mathbf{LFI1}} \alpha$, existe uma sequência de derivação $\varphi_1 \ldots \varphi_n$ onde $\varphi_n = \alpha$. A prova de $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \alpha$ é obtida por indução no tamanho n da sequência de derivação:

BASE. $n = 1$. A sequência contém somente uma fórmula $\varphi_1 = \alpha$. Portanto, existem duas possibilidades:

1. φ_1 é um axioma.
2. $\varphi_1 \in \Gamma$.

CASO 1. φ_1 é um axioma. Então vamos mostrar que para toda valoração $h : \mathcal{L}_\Sigma \to M$ sobre $\mathcal{M}_{\mathbf{LFI1}}$, se $h[\Gamma] \subseteq \{1, 1/2\}$, então $h(\varphi_1) \in \{1, 1/2\}$. Como φ_1 é um axioma, basta analisar todos os casos possíveis:

SUBCASO 1.1. $\varphi_1 = \alpha \to (\beta \to \alpha)$

α	β	$\alpha \to (\beta \to \alpha)$
1	1	1
1	$1/2$	1
1	0	1
$1/2$	1	$1/2$
$1/2$	$1/2$	$1/2$
$1/2$	0	1
0	1	1
0	$1/2$	1
0	0	1

SUBCASO 1.2. $\varphi_1 = (\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))$.

α	β	γ	$(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma))$
1	1	1	1
1	1	$1/2$	$1/2$
1	1	0	1
1	$1/2$	1	1
1	$1/2$	$1/2$	$1/2$
1	$1/2$	0	$1/2$
1	0	1	1
1	0	$1/2$	1

1	0	0	1
1/2	1	1	1
1/2	1	1/2	1/2
1/2	1	0	1/2
1/2	1/2	1	1
1/2	1/2	1/2	1/2
1/2	1/2	0	1/2
1/2	0	1	1
1/2	0	1/2	1/2
1/2	0	0	1/2
0	1	1	1
0	1	1/2	1
0	1	0	1
0	1/2	1	1
0	1/2	1/2	1
0	1/2	0	1
0	0	1	1
0	0	1/2	1
0	0	0	1

SUBCASO 1.3. $\varphi_1 = \alpha \to (\beta \to (\alpha \wedge \beta))$.

α	β	$\alpha \to (\beta \to (\alpha \wedge \beta))$
1	1	1
1	1/2	1/2
1	0	1
1/2	1	1/2
1/2	1/2	1/2
1/2	0	1
0	1	1
0	1/2	1
0	0	1

SUBCASO 1.4. $\varphi_1 = (\alpha \wedge \beta) \to \alpha$.

α	β	$(\alpha \wedge \beta) \to \alpha$
1	1	1
1	$1/2$	1
1	0	1
$1/2$	1	$1/2$
$1/2$	$1/2$	$1/2$
$1/2$	0	1
0	1	1
0	$1/2$	1
0	0	1

SUBCASO 1.5. $\varphi_1 = (\alpha \wedge \beta) \to \beta$.

α	β	$(\alpha \wedge \beta) \to \beta$
1	1	1
1	$1/2$	$1/2$
1	0	1
$1/2$	1	1
$1/2$	$1/2$	$1/2$
$1/2$	0	1
0	1	1
0	$1/2$	1
0	0	1

SUBCASO 1.6. $\varphi_1 = \alpha \to (\alpha \vee \beta)$.

α	β	$\alpha \to (\alpha \vee \beta)$
1	1	1
1	$1/2$	1
1	0	1
$1/2$	1	1
$1/2$	$1/2$	$1/2$
$1/2$	0	$1/2$
0	1	1
0	$1/2$	1

0	0	1

SUBCASO 1.7. $\varphi_1 = \beta \to (\alpha \vee \beta)$.

α	β	$\beta \to (\alpha \vee \beta)$
1	1	1
1	$1/2$	1
1	0	1
$1/2$	1	1
$1/2$	$1/2$	$1/2$
$1/2$	0	1
0	1	1
0	$1/2$	$1/2$
0	0	1

SUBCASO 1.8. $\varphi_1 = (\alpha \to \gamma) \to ((\beta \to \gamma) \to ((\alpha \vee \beta) \to \gamma))$.

α	β	γ	$(\alpha \to \gamma) \to ((\beta \to \gamma) \to ((\alpha \vee \beta) \to \gamma))$
1	1	1	1
1	1	$1/2$	$1/2$
1	1	0	1
1	$1/2$	1	1
1	$1/2$	$1/2$	$1/2$
1	$1/2$	0	1
1	0	1	1
1	0	$1/2$	$1/2$
1	0	0	1
$1/2$	1	1	1
$1/2$	1	$1/2$	$1/2$
$1/2$	1	0	1
$1/2$	$1/2$	1	1
$1/2$	$1/2$	$1/2$	$1/2$
$1/2$	$1/2$	0	$1/2$
$1/2$	0	1	1
$1/2$	0	$1/2$	$1/2$
$1/2$	0	0	$1/2$

0	1	1	1
0	1	$1/2$	$1/2$
0	1	0	1
0	$1/2$	1	1
0	$1/2$	$1/2$	$1/2$
0	$1/2$	0	$1/2$
0	0	1	1
0	0	$1/2$	1
0	0	0	1

SUBCASO 1.9. $\varphi_1 = (\alpha \to \beta) \vee \alpha$.

α	β	$(\alpha \to \beta) \vee \alpha$
1	1	1
1	$1/2$	1
1	0	1
$1/2$	1	1
$1/2$	$1/2$	$1/2$
$1/2$	0	$1/2$
0	1	1
0	$1/2$	1
0	0	1

SUBCASO 1.10. $\varphi_1 = \alpha \vee \neg\alpha$.

α	$\neg\alpha$	$\alpha \vee \neg\alpha$
1	0	1
$1/2$	$1/2$	$1/2$
0	1	1

SUBCASO 1.11. $\varphi_1 = \circ\alpha \to (\alpha \to (\neg\alpha \to \beta))$.

α	$\neg\alpha$	$\circ\alpha$	β	$\circ\alpha \to (\alpha \to (\neg\alpha \to \beta))$
1	0	1	1	1
1	0	1	$1/2$	1

1	0	1	0	1
$1/2$	$1/2$	0	1	1
$1/2$	$1/2$	0	$1/2$	1
$1/2$	$1/2$	0	0	1
0	1	1	1	1
0	1	1	$1/2$	1
0	1	1	0	1

SUBCASO 1.12. $\varphi_1 = \neg\neg\alpha \to \alpha$.

α	$\neg\neg\alpha$	$\neg\neg\alpha \to \alpha$
1	1	1
$1/2$	$1/2$	$1/2$
0	0	1

SUBCASO 1.13. $\varphi_1 = \alpha \to \neg\neg\alpha$.

α	$\neg\neg\alpha$	$\alpha \to \neg\neg\alpha$
1	1	1
$1/2$	$1/2$	$1/2$
0	0	1

SUBCASO 1.14. $\varphi_1 = \neg \circ \alpha \to (\alpha \wedge \neg\alpha)$.

α	$\neg\alpha$	$\neg \circ \alpha$	$\neg \circ \alpha \to (\alpha \wedge \neg\alpha)$
1	0	0	1
$1/2$	$1/2$	1	$1/2$
0	1	0	1

SUBCASO 1.15. $\varphi_1 = \neg(\alpha \vee \beta) \to (\neg\alpha \wedge \neg\beta)$.

α	β	$\neg(\alpha \vee \beta) \rightarrow (\neg\alpha \wedge \neg\beta)$
1	1	1
1	$1/2$	1
1	0	1
$1/2$	1	1
$1/2$	$1/2$	$1/2$
$1/2$	0	$1/2$
0	1	1
0	$1/2$	$1/2$
0	0	1

SUBCASO 1.16. $\varphi_1 = (\neg\alpha \wedge \neg\beta) \rightarrow \neg(\alpha \vee \beta)$.

α	β	$(\neg\alpha \wedge \neg\beta) \rightarrow \neg(\alpha \vee \beta)$
1	1	1
1	$1/2$	1
1	0	1
$1/2$	1	1
$1/2$	$1/2$	$1/2$
$1/2$	0	$1/2$
0	1	1
0	$1/2$	$1/2$
0	0	1

SUBCASO 1.17. $\varphi_1 = \neg(\alpha \wedge \beta) \rightarrow (\neg\alpha \vee \neg\beta)$.

α	β	$\neg(\alpha \wedge \beta) \rightarrow (\neg\alpha \vee \neg\beta)$
1	1	1
1	$1/2$	$1/2$
1	0	1
$1/2$	1	$1/2$
$1/2$	$1/2$	$1/2$
$1/2$	0	1
0	1	1
0	$1/2$	1

0	0	1

SUBCASO 1.18. $\varphi_1 = (\neg\alpha \vee \neg\beta) \to \neg(\alpha \wedge \beta)$.

α	β	$(\neg\alpha \vee \neg\beta) \to \neg(\alpha \wedge \beta)$
1	1	1
1	$1/2$	$1/2$
1	0	1
$1/2$	1	$1/2$
$1/2$	$1/2$	$1/2$
$1/2$	0	1
0	1	1
0	$1/2$	1
0	0	1

SUBCASO 1.19. $\varphi_1 = \neg(\alpha \to \beta) \to (\alpha \wedge \neg\beta)$.

α	β	$\neg(\alpha \to \beta) \to (\alpha \wedge \neg\beta)$
1	1	1
1	$1/2$	$1/2$
1	0	1
$1/2$	1	1
$1/2$	$1/2$	$1/2$
$1/2$	0	$1/2$
0	1	1
0	$1/2$	1
0	0	1

SUBCASO 1.20. $\varphi_1 = (\alpha \wedge \neg\beta) \to \neg(\alpha \to \beta)$.

α	β	$(\alpha \wedge \neg\beta) \to \neg(\alpha \to \beta)$
1	1	1
1	$1/2$	$1/2$
1	0	1
$1/2$	1	1

$1/2$	$1/2$	$1/2$
$1/2$	0	$1/2$
0	1	1
0	$1/2$	1
0	0	1

Com isso, o caso 1 está provado e $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi_1$ segue caso φ_1 seja um axioma.

CASO 2. $\varphi_1 \in \Gamma$. Logo, dada uma valoração $h : \mathcal{L}_\Sigma \to M$ de $\mathcal{M}_{\mathbf{LFI1}}$ se $h[\Gamma] \subseteq \{1, 1/2\}$, temos $h(\varphi_1) \in \{1, 1/2\}$. Portanto, $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi_1$.

PASSO.

HIPÓTESE DE INDUÇÃO (HI): Para qualquer sequência da derivação de $\Gamma \vdash_{\mathbf{LFI1}} \alpha$ de tamanho $k < n$, tem-se $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \alpha$.

Portanto, é preciso mostrar que $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \alpha$ segue caso a sequência de derivação de $\Gamma \vdash_{\mathbf{LFI1}} \alpha$ tenha tamanho n. Então, vamos supor $\Gamma \vdash_{\mathbf{LFI1}} \varphi_n$.

Ao analisar a obtenção de φ_n em $\Gamma \vdash_{\mathbf{LFI1}} \varphi_n$, existem três casos a se considerar:

1. φ_n é um axioma.
2. $\varphi_n \in \Gamma$.
3. φ_n é obtido por *modus ponens* em duas fórmulas φ_j e φ_k com $j, k < n$.

Os casos 1 e 2 são análogos aos casos provados na base.

CASO 3. φ_n é obtido por *modus ponens* em duas fórmulas φ_j e φ_k com $j, k < n$.

Logo, $\varphi_k = \varphi_j \to \varphi_n$ (ou $\varphi_j = \varphi_k \to \varphi_n$, a prova para este caso é análoga).

Dada nossa suposição de $\Gamma \vdash_{\mathbf{LFI1}} \varphi_n$, então $\Gamma \vdash_{\mathbf{LFI1}} \varphi_j$ e $\Gamma \vdash_{\mathbf{LFI1}} \varphi_j \to \varphi_n$.

Pela (HI), temos $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi_j$ e $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi_j \to \varphi_n$.

Seja h uma valoração qualquer sobre $\mathcal{M}_{\mathbf{LFI1}}$. Então, vamos supor $h[\Gamma] \subseteq \{1, 1/2\}$.

Temos $h(\varphi_j) \in \{1, 1/2\}$ e $h(\varphi_j \to \varphi_n) \in \{1, 1/2\}$.

Pela matriz de $\to$, temos $h(\varphi_j) = 0$ ou $h(\varphi_n) \in \{1, 1/2\}$.

Isto, unido ao fato de termos $h(\varphi_j) \in \{1, 1/2\}$, nos permite concluir $h(\varphi_n) \in \{1, 1/2\}$.

Portanto $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi_n$.

Com isso, provamos $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \alpha$ e a prova está finalizada. ■

Com a correção em relação à semântica matricial, o seguinte resultado é imediato:

Corolário 4 (Correção em relação à semântica de valorações)**.** *A lógica* **LFI1** *é correta em relação a sua semântica de valorações, ou seja, para todo conjunto de fórmulas* $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$*:*

$$\Gamma \vdash_{\mathbf{LFI1}} \alpha \Longrightarrow \Gamma \vDash_{\mathbf{LFI1}} \alpha.$$

Prova do Corolário 4. Seja $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas. Então, supondo $\Gamma \vdash_{\mathbf{LFI1}} \alpha$, temos, pelo Teorema 2, $\Gamma \models_{\mathcal{M}_{\mathbf{LFI1}}} \alpha$. Finalmente, pelo Corolário 3, temos $\Gamma \models_{\mathbf{LFI1}} \alpha$. ■

Com o resultado anterior, a prova de que a lógica **LFI1** se trata de uma **LFI** forte, como apresentado na Definição 12, segue naturalmente.

Corolário 5. *Seja p uma variável proposicional qualquer e $\bigcirc(p) = \{\circ p\}$ um conjunto de fórmulas dependente somente de p. A lógica* **LFI1** *é uma* **LFI** *forte em relação a $\neg$ e $\bigcirc(p)$.*

Prova do Corolário 5. Sejam $a, b \in \mathcal{P}$ variáveis proposicionais distintas e sejam $\varphi, \psi \in \mathcal{L}_\Sigma$ fórmulas quaisquer.

Prova da condição (i): A prova das condições será feita com base nas valorações v_1, v_2 e v_3 presentes na Tabela 1.

(i.a) Vamos provar $a, \neg a \nvdash_{\mathbf{LFI1}} b$. Tomando a valoração v_3, com $v_3(b) = 0$, então temos $a, \neg a \nvDash_{\mathbf{LFI1}} b$. Pela contraposta do Corolário 4, temos $a, \neg a \nvdash_{\mathbf{LFI1}} b$.

(i.b) Vamos provar $\circ a, a \nvdash_{\mathbf{LFI1}} b$. Tomando a valoração v_2, com $v_2(b) = 0$, então temos $a, \neg a \nvDash_{\mathbf{LFI1}} b$. Pela contraposta do Corolário 4, temos $\circ a, a \nvdash_{\mathbf{LFI1}} b$.

(i.c) Vamos provar $\circ a, \neg a \nvdash_{\mathbf{LFI1}} b$. Tomando a valoração v_1, com $v_1(b) = 0$, então temos $\circ a, \neg a \nvDash_{\mathbf{LFI1}} b$. Pela contraposta do Corolário 4, temos $\circ a, \neg a \nvdash_{\mathbf{LFI1}} b$.

Prova da condição (ii): A sequência de derivação abaixo mostra $\varphi, \neg\varphi, \circ\varphi \vdash_{\mathbf{LFI1}} \psi$:

1. φ	(Premissa)
2. $\neg\varphi$	(Premissa)
3. $\circ\varphi$	(Premissa)
4. $\circ\varphi \rightarrow (\varphi \rightarrow (\neg\varphi \rightarrow \psi))$	(**bc1**)
5. $\varphi \rightarrow (\neg\varphi \rightarrow \psi)$	(MP 3, 4)
6. $\neg\varphi \rightarrow \psi$	(MP 1, 5)
7. ψ	(MP 2, 6)

Portanto, a **LFI1** se trata de uma **LFI** forte. ■

3.4.3 Completude em relação às bivalorações

A completude é uma metapropriedade que garante que tudo aquilo que é válido segundo um sistema semântico de uma lógica pode ser provado pelo seu sistema dedutivo, ou seja, se uma fórmula não pode ser derivada no sistema dedutivo, então ela é inválida.

A prova da completude para a lógica **LFI1** depende das seguintes definiçõespara ser desenvolvida:

Definição 23 (Conjunto de fórmulas maximal não-trivial)**.** Seja $\mathscr{L}$ uma lógica Tarskiana definida sobre uma linguagem $\mathcal{L}$ e sejam $\Gamma, \{\varphi\}$ conjuntos de fórmulas de modo que $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}$. O

conjunto Γ é dito maximal não-trivial em relação a φ em $\mathscr{L}$ se $\Gamma \nvdash_{\mathscr{L}} \varphi$ mas $\Gamma, \psi \vdash_{\mathscr{L}} \varphi$ para qualquer $\psi \notin \Gamma$. ■

Definição 24 (Teoria fechada)**.** Seja $\mathscr{L}$ uma lógica Tarskiana. Um conjunto de fórmulas Γ é dito fechado em $\mathscr{L}$ (ou dito uma *teoria fechada* em $\mathscr{L}$) se, para toda fórmula φ, tem-se $\Gamma \vdash_{\mathscr{L}} \varphi$ sse $\varphi \in \Gamma$. ■

Lema 3. *Todo conjunto de fórmulas maximal não-trivial em relação a φ em $\mathscr{L}$ é fechado em $\mathscr{L}$.*

Prova do Lema 3. Seja Γ uma conjunto de fórmulas maximal não-trivial em relação a φ em $\mathscr{L}$.
($\Longrightarrow$) Se $\psi \in \Gamma$, então $\Gamma \vdash_{\mathscr{L}} \psi$, já que $\mathscr{L}$ é Tarskiana.
($\Longleftarrow$) Se $\Gamma \vdash_{\mathscr{L}} \psi$, então supondo $\psi \notin \Gamma$, temos $\Gamma, \psi \vdash_{\mathscr{L}} \varphi$, pela Definição 23. Pela propriedade do corte, segue que $\Gamma \vdash_{\mathscr{L}} \varphi$, o que contradiz o fato de Γ ser maximal não-trivial em relação a φ em $\mathscr{L}$. Portanto, $\psi \in \Gamma$.

Então Γ é fechado em $\mathscr{L}$. ■

Provaremos agora o lema de Lindenbaum, proposto originalmente por Adolf Lindenbaum (de acordo com Tarski (1956)) e adaptado por Łoś; Wójcicki (1955, 1984). A versão apresentada por Carnielli e Coniglio (2016) utiliza o princípio da boa ordem e aplica uma recursão transfinita para que o lema siga verdadeiro mesmo quando se tratando de sistemas definidos sobre linguagens não enumeráveis. Para o propósito do presente trabalho, isso não se mostra necessário, tendo em vista que a linguagem da **LFI1** (Definição 13) é enumerável.

Lema 4 (Lindenbaum-Łoś para **LFI1**)**.** *Dada a lógica* $\mathbf{LFI1} = \langle \mathcal{L}_\Sigma, \vdash_{\mathbf{LFI1}} \rangle$ *então temos, para qualquer conjunto de fórmulas* $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$*, se* $\Gamma \nvdash_{\mathbf{LFI1}} \varphi$ *então existe um conjunto de fórmulas* Δ*, com* $\Gamma \subseteq \Delta \subseteq \mathcal{L}_\Sigma$*, tal que* Δ *é um conjunto maximal não-trivial em relação a* φ *em* **LFI1**.

Prova do Lema 4. Dada $\mathbf{LFI1} = \langle \mathcal{L}_\Sigma, \vdash_{\mathbf{LFI1}} \rangle$ e seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas com $\Gamma \nvdash_{\mathbf{LFI1}} \varphi$. Arranje as fórmulas de $\mathcal{L}_\Sigma$ numa sequência $\mathcal{C} = \varphi_1, \varphi_2, \ldots, \varphi_i, \ldots$ e defina Γ_i recursivamente da seguinte forma:

$$\text{(i)} \quad \Gamma_0 = \Gamma$$

$$\text{(ii)} \quad \Gamma_i = \begin{cases} \Gamma_{i-1} & \text{sse } \Gamma_{i-1}, \varphi_i \vdash_{\mathbf{LFI1}} \varphi \\ \Gamma_{i-1} \cup \{\varphi_i\} & \text{sse } \Gamma_{i-1}, \varphi_i \nvdash_{\mathbf{LFI1}} \varphi \end{cases}$$

Então, defina um conjunto de fórmulas $\Delta = \bigcup_{i=0}^{\infty} \Gamma_i$. Dividiremos a prova em algumas partes:

(1) $\Gamma \subseteq \Delta \subseteq \mathcal{L}_\Sigma$.

Pela construção de Γ_i, temos $\Gamma_0 = \Gamma \subseteq \Gamma_i$ e $\Gamma_i \subseteq \mathcal{L}_\Sigma$ para todo $i \in \mathbb{N}$. Como Δ é formado a partir da união de todos os conjuntos Γ_i, temos $\Gamma_0 \subseteq \Delta$ e $\Delta \subseteq \mathcal{L}_\Sigma$.

(2) $\Gamma_k \subseteq \Gamma_i$, onde $0 \leq k \leq i$.

Dados dois números $i,k \in \mathbb{N}$ com $k \leq i$, o conjunto Γ_i é construído pela união de todos os seus predecessores, portanto, $\Gamma_k \subseteq \Gamma_i$.

(3) $\Gamma_i \nvdash_{\mathbf{LFI1}} \varphi$ para todo $i \geq 0$.

Provaremos por indução em i:

BASE. $i = 0$.

Temos $\Gamma_0 \nvdash_{\mathbf{LFI1}} \varphi$ pela nossa hipótese de $\Gamma \nvdash_{\mathbf{LFI1}} \varphi$.

PASSO.

Hipótese de indução (HI): Para qualquer $k < i$, temos $\Gamma_k \nvdash_{\mathbf{LFI1}} \varphi$.

Como caso particular da Hipótese de Indução, temos que $\Gamma_{i-1} \nvdash_{\mathbf{LFI1}} \varphi$. Portanto, pela construção de Γ_i, temos $\Gamma_i \nvdash_{\mathbf{LFI1}} \varphi$.

(4) Para todo $i \geq 1$, se $\varphi_i \in \Delta$, então $\varphi_i \in \Gamma_i$.

Vamos supor $\varphi_i \in \Delta$ e $\varphi_i \notin \Gamma_i$. Temos, por (3), $\Gamma_{i-1} \nvdash_{\mathbf{LFI1}} \varphi$. Pela definição de Γ_i, temos $\Gamma_{i-1}, \varphi_i \vdash_{\mathbf{LFI1}} \varphi$. Portanto, por (2) e já que a **LFI1** é Tarskiana (Proposição 1), temos $\Gamma_k, \varphi_i \vdash_{\mathbf{LFI1}} \varphi$, para qualquer $k \geq i$. Então, por (3), $\varphi_i \notin \Gamma_k$ para qualquer $k > i$. Portanto, $\varphi_i \notin \Delta$, o que contradiz a nossa suposição inicial. Portanto, $\varphi_i \in \Gamma_i$.

(5) Para todo $\Delta' \subseteq \Delta$ finito, existe um Γ_i na sequência com $\Delta' \subseteq \Gamma_i$.

Seja $\Delta' \subseteq \Delta$ um subconjunto finito. Como Δ é formado a partir da sequência $\mathcal{C} = \varphi_1, \varphi_2, \ldots, \varphi_i, \ldots$, então existe um $\varphi_{\max} \in \Delta'$ tal que $\max \geq i$ para qualquer $\varphi_i \in \Delta'$. Logo, $\varphi_{\max} \in \Delta$ e, por (4), temos $\varphi_{\max} \in \Gamma_{\max}$. Para qualquer outro elemento $\varphi_k \in \Delta'$, com $\max \geq k$, temos, por (4), $\varphi_k \in \Gamma_k$, e, por (2), $\Gamma_k \subseteq \Gamma_{\max}$. Portanto, $\Delta' \subseteq \Gamma_{\max}$.

(6) $\Delta \nvdash_{\mathbf{LFI1}} \varphi$.

Vamos supor $\Delta \vdash_{\mathbf{LFI1}} \varphi$. Portanto, como a **LFI1** é finitária (Proposição 2), existe um $\Delta' \subseteq \Delta$ finito com $\Delta' \vdash_{\mathbf{LFI1}} \varphi$. Logo, por (3), existe um Γ_i na sequência tal que $\Delta' \subseteq \Gamma_i$. Então, como a **LFI1** é Tarskiana (Proposição 1), temos $\Gamma_i \vdash_{\mathbf{LFI1}} \varphi$, o que é uma contradição (por (2)). Portanto, $\Delta \nvdash_{\mathbf{LFI1}} \varphi$.

(7) Δ é maximal não-trivial em relação a φ em **LFI1**.

Seja ψ uma fórmula qualquer com $\psi \notin \Delta$. Então $\psi = \varphi_i$ para algum $i \in \mathbb{N}$. Supondo $\varphi_i \in \Gamma_i$, chegamos numa contradição (já que, por construção de Δ, teríamos $\varphi_i \in \Delta$), portanto, temos $\varphi_i \notin \Gamma_i$. Logo, pela construção de Γ_i, temos $\Gamma_{i-1}, \varphi_i \vdash_{\mathbf{LFI1}} \varphi$. Então, já que a **LFI1** é Tarskiana, temos $\Delta, \varphi_i \vdash_{\mathbf{LFI1}} \varphi$, ou seja, $\Delta, \psi \vdash_{\mathbf{LFI1}} \varphi$. Portanto, Δ é maximal não-trivial em relação a φ em **LFI1**.

■

Lema 5. *Seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto qualquer de fórmulas com Γ sendo um conjunto maximal não-trivial em relação a φ em* **LFI1**. *A função $v : \mathcal{L}_\Sigma \to \{1,0\}$ definida, para todo $\psi \in \mathcal{L}_\Sigma$, como:*

$$v(\psi) = 1 \iff \psi \in \Gamma$$

é uma valoração para **LFI1**.

Prova do Lema 5. Queremos provar que v satisfaz todas as cláusulas da Definição 21.

CASO 1. Cláusula (*vAnd*)

($\Longrightarrow$) Vamos supor $v(\alpha \wedge \beta) = 1$. Logo $(\alpha \wedge \beta) \in \Gamma$.

Supondo $\alpha \notin \Gamma$, temos $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$. Pelo axioma (**Ax4**) e MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$. Logo, já que a **LFI1** é Tarskiana (Proposição 6), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que é uma contradição. Portanto $\alpha \in \Gamma$.

Supondo $\beta \notin \Gamma$, temos $\Gamma, \beta \vdash_{\mathbf{LFI1}} \varphi$. Pelo axioma (**Ax5**) e MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \beta$. Logo, já que a **LFI1** é Tarskiana (Proposição 6), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que é uma contradição. Portanto $\beta \in \Gamma$.

Logo, temos $v(\alpha) = 1$ e $v(\beta) = 1$.

($\Longleftarrow$) Vamos supor $v(\alpha) = 1$ e $v(\beta) = 1$. Logo, $\alpha \in \Gamma$ e $\beta \in \Gamma$. Pelo axioma (**Ax3**) e MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \wedge \beta$. Pelo Lema 3, temos $\alpha \wedge \beta \in \Gamma$.

Logo, temos $v(\alpha \wedge \beta) = 1$

CASO 2. Cláusula (*vOr*)

($\Longrightarrow$) Vamos supor $v(\alpha \vee \beta) = 1$. Logo, $\alpha \vee \beta \in \Gamma$.

Supondo $\alpha \notin \Gamma$ e $\beta \notin \Gamma$. Temos $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma, \beta \vdash_{\mathbf{LFI1}} \varphi$. Logo, pelo Corolário 1, temos $\Gamma, \alpha \vee \beta \vdash_{\mathbf{LFI1}} \varphi$. Logo, já que a **LFI1** é Tarskiana (Proposição 6), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que é uma contradição. Portanto $\alpha \in \Gamma$ ou $\beta \in \Gamma$.

Logo, temos $v(\alpha) = 1$ ou $v(\beta) = 1$.

($\Longleftarrow$) Vamos supor $v(\alpha) = 1$ ou $v(\beta) = 1$. Logo, temos $\alpha \in \Gamma$ ou $\beta \in \Gamma$.

Caso tenhamos $\alpha \in \Gamma$, então, pelo axioma (**Ax6**), temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \rightarrow (\alpha \vee \beta)$. Pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$. Então, por MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \vee \beta$. Finalmente, pelo Lema 3, temos $\alpha \vee \beta \in \Gamma$ e, consequentemente, $v(\alpha \vee \beta) = 1$.

Caso tenhamos $\beta \in \Gamma$, a prova é análoga.

Portanto, temos $v(\alpha \vee \beta) = 1$.

CASO 3. Cláusula (*vImp*)

($\Longrightarrow$) Vamos supor $v(\alpha \rightarrow \beta) = 1$. Logo, $\alpha \rightarrow \beta \in \Gamma$.

Supondo $v(\alpha) = 1$ e $v(\beta) = 0$, temos $\alpha \in \Gamma$ e $\beta \notin \Gamma$. Então, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$, $\Gamma \nvdash_{\mathbf{LFI1}} \beta$ e $\Gamma \vdash_{\mathbf{LFI1}} \alpha \rightarrow \beta$. Logo, por MP, $\Gamma \vdash_{\mathbf{LFI1}} \beta$, o que é uma contradição. Portanto, $v(\alpha) = 0$ ou $v(\beta) = 1$.

($\Longleftarrow$) Vamos supor $v(\alpha) = 0$ ou $v(\beta) = 1$.

Caso $v(\alpha) = 0$, então temos $\alpha \notin \Gamma$. Supondo $\alpha \to \beta \notin \Gamma$, temos, pelo Lema 3, $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma, \alpha \to \beta \vdash_{\mathbf{LFI1}} \varphi$. Logo, pelo Corolário 1, temos $\Gamma, (\alpha \to \beta) \vee \alpha \vdash_{\mathbf{LFI1}} \varphi$. Portanto, pelo axioma (**Ax9**) e já que a **LFI1** é Tarskiana, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$, o que é uma contradição. Portanto, $\alpha \to \beta \in \Gamma$ e, então, $v(\alpha \to \beta) = 1$.

Caso $v(\beta) = 1$, então temos $\beta \in \Gamma$. Pelo axioma (**Ax1**), temos $\Gamma \vdash_{\mathbf{LFI1}} \beta \to (\alpha \to \beta)$. Portanto, por MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \beta$. Logo $\alpha \to \beta \in \Gamma$ e, então, $v(\alpha \to \beta) = 1$.

Logo, $v(\alpha \to \beta) = 1$.

CASO 4. Cláusula (*vNeg*)

Supondo $v(\neg\alpha) = 0$, temos $\neg\alpha \notin \Gamma$. Então, caso $v(\alpha) = 0$, temos $\alpha \notin \Gamma$.

Portanto, $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma\neg\alpha \vdash_{\mathbf{LFI1}} \varphi$. Pelo Corolário 2, temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que é uma contradição. Logo, $v(\alpha) = 1$.

CASO 5. Cláusula (*vCon*)

Supondo $v(\circ\alpha) = 1$, temos $\circ\alpha \in \Gamma$. Logo, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \circ\alpha$.

Vamos supor $v(\alpha) = 1$ e $v(\neg\alpha) = 1$. Portanto, temos $\alpha \in \Gamma$ e $\neg\alpha \in \Gamma$ e, pelo Lema 3, $\Gamma \vdash_{\mathbf{LFI1}} \alpha$ e $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha$.

Pelo axioma (**bc1**), temos $\Gamma \vdash_{\mathbf{LFI1}} \circ\alpha \to (\alpha \to (\neg\alpha \to \varphi))$. Portanto, aplicando MP três vezes, temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que é uma contradição. Logo, $v(\alpha) = 0$ ou $v(\neg\alpha) = 0$

CASO 6. Cláusula (*vCi*)

Supondo $v(\neg \circ \alpha) = 1$, temos $\neg \circ \alpha \in \Gamma$. Então, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg \circ \alpha$.

Pelo axioma (**ci**), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg \circ \alpha \to (\alpha \wedge \neg\alpha)$. Portanto, por MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \wedge \neg\alpha$.

Pelo axioma (**Ax4**), temos $\Gamma \vdash_{\mathbf{LFI1}} (\alpha \wedge \neg\alpha) \to \alpha$. Aplicando MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$.

Pelo axioma (**Ax5**), temos $\Gamma \vdash_{\mathbf{LFI1}} (\alpha \wedge \neg\alpha) \to \neg\alpha$. Aplicando MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha$.

Pelo Lema 3, temos $\alpha \in \Gamma$ e $\neg\alpha \in \Gamma$. Logo, $v(\alpha) = 1$ e $v(\neg\alpha) = 1$.

CASO 7. Cláusula (*vDNE*)

($\Longrightarrow$) Supondo $v(\neg\neg\alpha) = 1$, temos $\neg\neg\alpha \in \Gamma$. Então, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\neg\alpha$. Pelo axioma (**cf**), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\neg\alpha \to \alpha$. Logo, por MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$. Finalmente, pelo Lema 3, temos $\alpha \in \Gamma$ e, portanto, $v(\alpha) = 1$.

($\Longleftarrow$) Supondo $v(\alpha) = 1$, temos $\alpha \in \Gamma$. Então, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$. Pelo axioma (**ce**), temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to \neg\neg\alpha$. Logo, por MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\neg\alpha$. Finalmente, pelo Lema 3, temos $\neg\neg\alpha \in \Gamma$ e, portanto, $v(\neg\neg\alpha) = 1$.

CASO 8. Cláusula ($vDM_{\wedge}$)

($\Longrightarrow$) Supondo $v(\neg(\alpha \wedge \beta)) = 1$, temos $\neg(\alpha \wedge \beta) \in \Gamma$. Então, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \wedge \beta)$. Pelo axioma (**neg**$\wedge_1$), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \wedge \beta) \rightarrow (\neg\alpha \vee \neg\beta)$. Logo, por MP, temos $\Gamma \vdash_{\mathbf{LFI1}} (\neg\alpha \vee \neg\beta)$.

Vamos então supor $\neg\alpha \notin \Gamma$ e $\neg\beta \notin \Gamma$. Então, temos $\Gamma, \neg\alpha \vdash_{\mathbf{LFI1}} \varphi$ e $\Gamma, \neg\beta \vdash_{\mathbf{LFI1}} \varphi$. Pelo Corolário 1, temos $\Gamma, \neg\alpha \vee \neg\beta \vdash_{\mathbf{LFI1}} \varphi$. Já que a **LFI1** é Tarskiana (Proposição 1), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que resulta numa contradição. Portanto, temos $\neg\alpha \in \Gamma$ ou $\neg\beta \in \Gamma$ e, consequentemente, $v(\neg\alpha) = 1$ ou $v(\neg\beta) = 1$.

($\Longleftarrow$) Supondo $v(\neg\alpha) = 1$ ou $v(\neg\beta) = 1$, temos $\neg\alpha \in \Gamma$ ou $\neg\beta \in \Gamma$.

Caso tenhamos $\neg\alpha \in \Gamma$, então, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha$. Logo, pelo axioma (**Ax6**), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha \rightarrow (\neg\alpha \vee \neg\beta)$. Aplicando MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha \vee \neg\beta$. Pelo axioma (**neg**$\wedge_2$), temos $\Gamma \vdash_{\mathbf{LFI1}} (\neg\alpha \vee \neg\beta) \rightarrow \neg(\alpha \wedge \beta)$. Aplicando MP novamente, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \wedge \beta)$. Pelo Lema 3, temos $\neg(\alpha \wedge \beta) \in \Gamma$ e, consequentemente, $v(\neg(\alpha \wedge \beta)) = 1$.

Caso tenhamos $\neg\beta \in \Gamma$, então, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\beta$. Logo, pelo axioma (**Ax7**), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\beta \rightarrow (\neg\alpha \vee \neg\beta)$. Aplicando MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha \vee \neg\beta$. Pelo axioma (**neg**$\wedge_2$), temos $\Gamma \vdash_{\mathbf{LFI1}} (\neg\alpha \vee \neg\beta) \rightarrow \neg(\alpha \wedge \beta)$. Aplicando MP novamente, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \wedge \beta)$. Pelo Lema 3, temos $\neg(\alpha \wedge \beta) \in \Gamma$ e, consequentemente, $v(\neg(\alpha \wedge \beta)) = 1$.

Portanto, temos $v(\neg(\alpha \wedge \beta)) = 1$.

CASO 9. Cláusula ($vDM_{\vee}$)

($\Longrightarrow$) Supondo $v(\neg(\alpha \vee \beta)) = 1$, temos $\neg(\alpha \vee \beta) \in \Gamma$. Logo, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \vee \beta)$. Pelo axioma (**neg**$\vee_1$), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \vee \beta) \rightarrow (\neg\alpha \wedge \neg\beta)$. Logo, por MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha \wedge \neg\beta$.

Vamos supor $\neg\alpha \notin \Gamma$ ou $\neg\beta \notin \Gamma$. Então, temos $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ ou $\Gamma, \beta \vdash_{\mathbf{LFI1}} \varphi$. Pelos axiomas (**Ax4**) e (**Ax5**), temos $\Gamma \vdash_{\mathbf{LFI1}} (\neg\alpha \wedge \neg\beta) \rightarrow \alpha$ e $\Gamma \vdash_{\mathbf{LFI1}} (\neg\alpha \wedge \neg\beta) \rightarrow \beta$. Aplicando MP duas vezes temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$ e $\Gamma \vdash_{\mathbf{LFI1}} \beta$. Já que a **LFI1** é Tarskiana (Proposição 1), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que é resulta numa contradição. Portanto, $\neg\alpha \in \Gamma$ e $\neg\beta \in \Gamma$ e, consequentemente, temos $v(\neg\alpha) = 1$ e $v(\neg\beta) = 1$.

($\Longleftarrow$) Supondo $v(\neg\alpha) = 1$ e $v(\neg\beta) = 1$, temos $\neg\alpha \in \Gamma$ e $\neg\beta \in \Gamma$. Pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha$ e $\Gamma \vdash_{\mathbf{LFI1}} \neg\beta$. Pelo axioma (**Ax3**), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha \rightarrow (\neg\beta \rightarrow (\neg\alpha \wedge \neg\beta))$. Aplicando MP duas vezes, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg\alpha \wedge \neg\beta$. Pelo axioma (**neg**$\vee_2$), temos $\Gamma \vdash_{\mathbf{LFI1}} (\neg\alpha \wedge \neg\beta) \rightarrow \neg(\alpha \vee \beta)$. Aplicando MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \vee \beta)$. Pelo Lema 3, temos $\neg(\alpha \vee \beta) \in \Gamma$ e, consequentemente, $v(\neg(\alpha \vee \beta)) = 1$.

CASO 10. Cláusula ($vCip_{\rightarrow}$)

($\Longrightarrow$) Supondo $v(\neg(\alpha \to \beta)) = 1$, temos $\neg(\alpha \to \beta) \in \Gamma$. Logo, pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \to \beta)$. Pelo axioma (**neg**$\to_1$), temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \to \beta) \to (\alpha \wedge \neg\beta)$. Aplicando MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \wedge \neg\beta$. Pelos axiomas (**Ax4**) e (**Ax5**), temos $\Gamma \vdash_{\mathbf{LFI1}} (\alpha \wedge \neg\beta) \to \alpha$ e $\Gamma \vdash_{\mathbf{LFI1}} (\alpha \wedge \neg\beta) \to \beta$. Aplicando MP duas vezes, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$ e $\Gamma \vdash_{\mathbf{LFI1}} \neg\beta$.

Vamos supor $v(\alpha) = 0$ ou $v(\neg\beta) = 0$. Logo, temos $\alpha \notin \Gamma$ ou $\neg\beta \notin \Gamma$. Logo, temos $\Gamma, \alpha \vdash_{\mathbf{LFI1}} \varphi$ ou $\Gamma, \neg\beta \vdash_{\mathbf{LFI1}} \varphi$. Já que a **LFI1** é Tarskiana (Proposição 1), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$, o que resulta numa contradição. Portanto, temos $v(\alpha) = 1$ e $v(\neg\beta) = 1$.

($\Longleftarrow$) Supondo $v(\alpha) = 1$ e $v(\neg\beta) = 1$, temos $\alpha \in \Gamma$ e $\neg\beta \in \Gamma$. Pelo Lema 3, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha$ e $\Gamma \vdash_{\mathbf{LFI1}} \neg\beta$. Pelo axioma (**Ax3**), temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \to (\neg\beta \to (\alpha \wedge \neg\beta))$. Aplicando MP duas vezes, temos $\Gamma \vdash_{\mathbf{LFI1}} \alpha \wedge \neg\beta$. Pelo axioma (**neg**$\to_2$), temos $\Gamma \vdash_{\mathbf{LFI1}} (\alpha \wedge \neg\beta) \to \neg(\alpha \to \beta)$. Aplicando MP, temos $\Gamma \vdash_{\mathbf{LFI1}} \neg(\alpha \to \beta)$. Pelo Lema 3, temos $\neg(\alpha \to \beta) \in \Gamma$ e, consequentemente, $v(\neg(\alpha \to \beta)) = 1$. ■

Teorema 3 (Completude em relação à semântica de valorações)**.** *A lógica* **LFI1** *é completa em relação a sua semântica de valorações, ou seja, para todo conjunto de fórmulas* $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$*:*

$$\Gamma \vDash_{\mathbf{LFI1}} \alpha \Longrightarrow \Gamma \vdash_{\mathbf{LFI1}} \alpha.$$

Prova do Teorema 3. Seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas com $\Gamma \nvdash_{\mathbf{LFI1}} \varphi$. Então, pelo Lema 4, existe um conjunto de fórmulas Δ com $\Gamma \subseteq \Delta \subseteq \mathcal{L}_\Sigma$ com Δ sendo um conjunto maximal não-trivial em relação a φ em **LFI1**. Temos $\Delta \nvdash_{\mathbf{LFI1}} \varphi$ e, portanto, pelo Lema 3, temos $\varphi \notin \Delta$. Pelo Lema 5, existe uma valoração v tal que, para todo $\psi \in \mathcal{L}_\Sigma$, $v(\psi) = 1 \Longleftrightarrow \psi \in \Delta$. Logo, $v[\Delta] \subseteq \{1\}$ e $v(\varphi) = 0$. Como $\Gamma \subseteq \Delta$, temos $v[\Gamma] \subseteq \{1\}$. Portanto, pela Definição 22, $\Gamma \nvDash_{\mathbf{LFI1}} \varphi$. Então, o Teorema segue por contraposição. ■

3.4.4 Equivalência entre as semânticas matricial e de valorações

Com a prova da completude em relação a semântica de valorações, os seguintes resultados são imediatos:

Corolário 6 (Equivalência entre a semântica matricial e a semântica de valorações)**.** *A semântica matricial e a semântica de valorações para a lógica* **LFI1** *são equivalentes, ou seja, para todo conjunto de fórmulas* $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$*:*

$$\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \alpha \Longleftrightarrow \Gamma \vDash_{\mathbf{LFI1}} \alpha.$$

Prova do Corolário 6. Seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas quaisquer.
($\Longrightarrow$) Supondo $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi$, então $\Gamma \vDash_{\mathbf{LFI1}} \varphi$ segue pelo Corolário 3.
($\Longleftarrow$) Supondo $\Gamma \vDash_{\mathbf{LFI1}} \varphi$, então, pela completude (Teorema 3), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$. Logo, pela correção (Teorema 2), temos $\Gamma \vDash_{\mathcal{M}_{\mathbf{LFI1}}} \varphi$. ■

3.4.5 Completude em relação à semântica matricial

Com esta equivalência, a prova da completude da **LFI1** em relação à semântica matricial é obtida como um corolário simples.

Corolário 7 (Completude em relação à semântica matricial)**.** *A lógica* **LFI1** *é completa em relação a sua semântica matricial, ou seja, para todo conjunto de fórmulas* $\Gamma \cup \{\alpha\} \subseteq \mathcal{L}_\Sigma$*:*

$$\Gamma \models_{\mathcal{M}_{\mathbf{LFI1}}} \alpha \Longrightarrow \Gamma \vdash_{\mathbf{LFI1}} \alpha.$$

Prova do Corolário 7. Seja $\Gamma \cup \{\varphi\} \subseteq \mathcal{L}_\Sigma$ um conjunto de fórmulas com $\Gamma \models_{\mathcal{M}_{\mathbf{LFI1}}} \varphi$. Pela equivalência entre a semântica matricial e as bivalorações (Corolário 6), temos $\Gamma \models_{\mathbf{LFI1}} \varphi$. Logo, pela completude em relação às bivalorações (Teorema 3), temos $\Gamma \vdash_{\mathbf{LFI1}} \varphi$. ■

4 IMPLEMENTAÇÃO EM ROCQ

Neste capítulo será descrita a implementação da biblioteca da lógica de inconsistência formal **LFI1** no assistente de provas Rocq (anteriormente conhecido como Coq), bem como o desenvolvimento de alguns metateoremas apresentados na Seção 3.4 dentro da biblioteca. A implementação será análoga àquela feita por Silveira (2020), que implementou uma biblioteca de lógica modal. Antes de tratar especificamente da implementação, o Rocq será brevemente apresentado e caracterizado.

4.1 ASSISTENTE DE PROVAS ROCQ

Os assistentes de provas são ferramentas de *software* que auxiliam o usuário no desenvolvimento de teoremas, permitindo que provas sejam verificadas na medida em que são escritas (GEUVERS, 2009), conferindo a estes programas uma importância significativa na verificação e especificação formal de *software*. Atualmente, existem diversos assistentes de provas como: Agda, Isabelle, Rocq, Lean, Idris e Twelf. Cada um destes tem suas particularidades e diferenças em relação ao formalismo matemático utilizado como base.

O Rocq é um assistente de provas baseado no Cálculo de Construções Indutivas (CCI) que possui aplicações em diferentes áreas da matemática e da computação como (mas não limitado a) lógica, linguagens formais, linguística computacional e desenvolvimento de programas seguros (BERTOT; CASTÉRAN, 2004). Sob a ótica da Correspondência de Curry-Howard, o Rocq é tanto uma linguagem de programação funcional quanto uma linguagem de prova, podendo ser dividido em quatro partes (SILVA, 2019):

- A linguagem de programação e especificação *Gallina*, que goza da propriedade da normalização forte[1], a qual garante que todo programa termina.
- A linguagem de comandos *Vernacular*, que permite interação com o assistente.
- O conjunto de táticas (*tactics*) utilizadas para manipular elementos durante o desenvolvimento de uma prova.
- A linguagem $\mathcal{L}$tac, utilizada para implementar novas táticas e automatizar provas.

No restante desse trabalho, conceitos básicos sobre o funcionamento do assistente de provas Rocq e sobre seu uso no desenvolvimento e verificação de provas não serão apresentados em grandes detalhes. Ao leitor interessado em tais assuntos, é recomendada uma consulta aos trabalhos de Pierce et al. (2017) e Silveira (2020).

[1] Um termo-λ é fortemente normalizável caso toda sequência de reescrita acabe numa forma normal (num termo irredutível). Um sistema no qual todos os termos-λ são fortemente normalizáveis possui a propriedade da normalização forte (NIPKOW, 2006).

4.2 BIBLIOTECA EM ROCQ

Nesta seção será apresentada a biblioteca para a lógica **LFI1**, com os principais artefatos semânticos e sintáticos apresentados ao longo do trabalho, bem como provas para os metateoremas que foram provados manualmente. A biblioteca foi desenvolvida utilizando a versão 9.0.0 do Rocq e está disponível em <https://github.com/dipled/LFI1_Library>. Cada subseção representa um arquivo da biblioteca e as partes mais importantes de cada um destes será comentada ao longo do texto.

4.2.1 Utils.v

Este arquivo contém algumas notações, definições e propriedades úteis que foram estabelecidas para facilitar a implementação.

```
Require Import Arith Constructive_sets.
    ⋮
Notation " a ∈ A " := (In A a) (at level 10).
Notation " a ∉ A " := (~In A a) (at level 10).
Notation " B ∪ C " := (Union B C) (at level 48, left associativity).
Notation " [ a ] " := (Singleton a) (at level 0, right associativity).
Notation " A ⊆ B " := (Included A B) (at level 71).
Notation " ∅ "  := (Empty_set).
```

As bibliotecas `Arith` e `Constructive_sets` fornecem alguns teoremas e definições fundamentais para trabalhar com conjuntos em Rocq, na forma de estruturas chamadas de `Ensembles`, que são mapeamentos de um tipo qualquer `U` para `Prop`. A fim de facilitar a leitura do código, foram estabelecidas notações para conceitos de pertinência, união e contingência de conjuntos e conjuntos vazio e unitário.

```
Theorem iff_neg : forall A B : Prop, (A <-> B) -> (~A <-> ~B).
    ⋮
Theorem contra : forall A B : Prop, (A -> B) -> (~B -> ~A).
    ⋮
Ltac destruct_conjunction H :=
match type of H with
| _ /\ _ =>
  let L := fresh "L" in
  let R := fresh "R" in
  destruct H as [L R]; destruct_conjunction L; destruct_conjunction R
| _ => idtac
end.
```

A tática `destruct_conjunction`, implementada em $\mathcal{L}$tac, é uma tática que recebe como argumento uma hipótese `H` e caso esta hipótese seja uma conjunção, ela é destruída em duas novas hipóteses `L` e `R` e a tática é repetida recursivamente para cada uma destas hipóteses.

4.2.2 Language.v

Este arquivo contém a definição da linguagem da **LFI1**, como apresentada na Seção 3.1.

```
Definition Atom := nat.

Inductive Formula : Set :=
  | Lit    : Atom -> Formula
  | Neg    : Formula -> Formula
  | And    : Formula -> Formula -> Formula
  | Or     : Formula -> Formula -> Formula
  | Imp    : Formula -> Formula -> Formula
  | Consistency : Formula -> Formula.

Notation " x → y " :=
(Imp x y) (at level 8, right associativity).
    ⋮
Notation " # x " :=
(Lit x) (at level 2, no associativity, x constr at level 1, format "# x").
```

Note que `Lit` é construído a partir de um `Atom`, que por sua vez é só um *alias* para o tipo `nat` (um número natural), portanto o tipo indutivo `Formula` é enumerável.

4.2.3 Syntax.v

Este arquivo contém a definição da sintaxe da **LFI1**, através do cáclulo de Hilbert definido na Seção 3.2.

```
Inductive Ax : Set :=
  | Ax1      : Formula -> Formula -> Ax
  | Ax2      : Formula -> Formula -> Formula -> Ax
    ⋮
  | bc1      : Formula -> Formula -> Ax
  | cf       : Formula -> Ax
  | ce       : Formula -> Ax
  | ci       : Formula -> Ax
    ⋮
Definition instantiate (a : Ax) : Formula :=
  match a with
```

```
  | Ax1 α β     => α → (β → α)
  | Ax2 α β γ => (α → (β → γ)) → ((α → β) → (α → γ))
    ⋮
  | bc1 α β     => ∘α → (α → (¬α → β))
  | cf α          => ¬¬α → α
  | ce α          => α → ¬¬α
  | ci α          => ¬∘α → (α ∧ ¬ α)
    ⋮
  end.
```

Primeiro, o tipo `Ax` foi definido como um tipo indutivo (também chamado de tipo algébrico), com construtores que recebem como argumentos fórmulas correspondentes ao número de fórmulas dos axiomas apresentados na Definição 15.

Depois, a função `instantiate` é definida. Ela recebe como argumento um axioma (que por sua vez foi construído a partir de fórmulas) e retorna uma fórmula correspondente àquela representada pelo axioma, por exemplo, o axioma **cf** representa a eliminação da dupla negação, portanto, a função `instantiate` receberia um dado `cf` α do tipo `Ax`, onde α é uma fórmula qualquer da linguagem, e retornaria um dado $\neg\neg\alpha \rightarrow \alpha$ do tipo `Formula`.

```
Inductive deduction : Ensemble Formula -> Formula -> Prop :=
  | Premisse : forall (Γ : Ensemble Formula) (φ : Formula), φ ∈ Γ -> deduction Γ φ
  | AxiomInstance : forall (Γ : Ensemble Formula) (a : Ax), deduction Γ (instantiate a)
  | MP : forall (Γ : Ensemble Formula) (φ ψ : Formula), (deduction Γ (φ → ψ)) ->
    (deduction Γ φ) -> deduction Γ ψ.

Notation " Γ ⊢ φ " := (deduction Γ φ) (at level 50, no associativity).
```

O tipo `deduction` é definido indutivamente a partir de um conjunto de fórmulas (premissas) e uma fórmula (conclusão) e retornando um dado do tipo `Prop` referente a possibilidade de construir uma derivação da conclusão a partir do conjunto de premissas, aplicando uma sequência de construtores, que representam os três casos de derivação descritos na Definição 16. O construtor `Premisse` afirma que se uma fórmula qualquer pertence ao conjunto de premissas, então ela é derivável a partir deste conjunto. O construtor `AxiomInstance` afirma que um axioma é sempre derivável a partir de qualquer conjunto de premissas. O construtor `MP` afirma que, dadas duas fórmulas φ e ψ quaisquer, se for possível derivar $\varphi \rightarrow \psi$ e φ a partir de um conjunto de fórmulas, então podemos derivar ψ a partir deste mesmo conjunto. Por último, uma notação com o operador infixo $\vdash$ é estabelecida para facilitar a leitura de derivações.

4.2.4 Semantics.v

Este arquivo contém a definição dos dois sistemas semânticos apresentados na Seção 3.3. O primeiro sistema definido foi a semântica matricial, conforme as Definições 19 e 20.

```
Inductive MatrixDomain : Set :=
  | One
  | Half
  | Zero.

Definition designatedValue (a : MatrixDomain) : Prop :=
  match a with
  | Zero => False
  | _ => True
  end.

Definition andM (a b : MatrixDomain) : MatrixDomain :=
  match a, b with
  | Zero, _  => Zero
  | _, Zero  => Zero
  | One, One => One
  | _, _     => Half
  end.
    ⋮
Notation " x ∧m y " :=
(andM x y) (at level 20, left associativity).
    ⋮
```

Note que `designatedValue` é um predicado sobre o domínio da matriz lógica, e retorna um `Prop` referente à pertinência de um dado valor ao conjunto $\{1, 1/2\}$.

Após definir a matriz lógica, é preciso estabelecer a noção de valorações sobre essa matriz, conforme apresentada na Definição 18, que define valoração como um homomorfismo entre linguagem e a álgebra da matriz, preservando a estrutura dos seus operadores.

```
Definition preserveAnd (v : Formula -> MatrixDomain) : Prop :=
forall φ ψ: Formula, (v (φ ∧ ψ)) = (v φ) ∧m (v ψ).
    ⋮
Definition valuation (v : Formula -> MatrixDomain) : Prop :=
preserveOr v /\ preserveTo v /\ preserveAnd v /\ preserveNeg v /\ preserveCirc v.

Definition matrixEntails (Γ : Ensemble Formula) (φ : Formula) :=
forall v : (Formula -> MatrixDomain),
valuation v ->
(forall (ψ: Formula),
  ψ ∈ Γ -> designatedValue (v ψ)) ->
    designatedValue (v φ).
```

```
Notation " Γ ⊨m φ " := (matrixEntails Γ φ) (at level 50, no associativity).
```

A definição `valuation` representa um predicado sobre qualquer mapeamento de `Formula` para `MatrixDomain`, que retorna `True` caso este mapeamento seja um homomorfismo sobre a linguagem e o domínio da matriz e retorna `False` caso contrário. A relação de consequência semântica é definida em `matrixEntails` (conforme a Definição 20), que diz que, dado um conjunto de fórmulas (premissas) e uma fórmula (conclusão), esta fórmula será uma consequência semântica do conjunto de premissas caso toda valoração que satisfaz todas as fórmulas deste conjunto também satisfizer a conclusão.

Depois disso, a semântica de bivalorações foi implementada, com base nas Definições 21 e 22.

```
Inductive BivaluationDomain : Set :=
  | Bot
  | Top.

Notation " ⊥ " := Bot.
Notation " ⊤ " := Top.

Definition vAnd (v : Formula -> BivaluationDomain) : Prop :=
  forall φ ψ : Formula, (v (φ ∧ ψ)) = ⊤ <-> (v φ = ⊤) /\ (v ψ = ⊤).
    ⋮
Definition vNeg (v : Formula -> BivaluationDomain) : Prop :=
  forall φ : Formula, (v ¬φ) = ⊥ -> v φ = ⊤.

Definition vCon (v : Formula -> BivaluationDomain) : Prop :=
  forall φ : Formula, (v ∘φ) = ⊤ -> (v φ = ⊥) \/ (v ¬φ = ⊥).

Definition vCi (v : Formula -> BivaluationDomain) : Prop :=
  forall φ : Formula, (v ¬∘φ) = ⊤ -> (v φ = ⊤) /\ (v ¬φ = ⊤).
    ⋮
Definition vCip (v : Formula -> BivaluationDomain) : Prop :=
  forall φ ψ : Formula, (v ¬(φ →ψ)) = ⊤ <-> (v φ = ⊤) /\ (v ¬ψ = ⊤).
Definition bivaluation (v : Formula -> BivaluationDomain) : Prop :=
  vAnd v /\ vOr v /\ vImp v /\ vNeg v /\ vCon v /\ vCi v /\
  vDne v /\ vDmAND v /\ vDmOR v /\ vCip v.
```

A bivaloração é definida sobre um tipo indutivo com construtores $\top$ e $\bot$, representando respectivamente os valores para verdade e falsidade do domínio. As cláusulas `vAnd` até `vCip` definem as condições que um mapeamento entre `Formula` e `BivaluationDomain` precisam respeitar para serem consideradas bivalorações, caracterizadas pelo predicado `bivaluation`.

Estes mapeamentos apresentam um comportamento não-determinístico em algumas cláusulas, como na cláusula vNeg.

Após definir estas cláusulas, algumas propriedades úteis sobre a semântica de bivalorações (como o terceiro excluído em bivaluation_lem) são provadas.

```
Lemma bivaluation_lem : forall (v : Formula -> BivaluationDomain) (φ : Formula),
(v φ = ⊤) ∨ (v φ = ⊥).
Proof. ... Qed.

Lemma bivaluation_dec1 : forall (v : Formula -> BivaluationDomain) (φ : Formula),
v φ = ⊤ <-> ~ v φ = ⊥.
Proof. ... Qed.

Lemma bivaluation_dec2 : forall (v : Formula -> BivaluationDomain) (φ : Formula),
v φ = ⊥ <-> ~ v φ = ⊤.
Proof. ... Qed.

Definition vAndf (v : Formula -> BivaluationDomain) : Prop :=
forall φ ψ : Formula, (v (φ ∧ ψ)) = ⊥ <-> (v φ = ⊥) ∨ (v ψ = ⊥).
    ⋮
Definition vNegf (v : Formula -> BivaluationDomain) : Prop :=
forall φ : Formula, (v φ) = ⊥ -> (v ¬φ) = ⊤.

Definition vConf (v : Formula -> BivaluationDomain) : Prop :=
forall φ : Formula, (v φ = ⊤) ∧ (v ¬φ = ⊤) -> (v ∘φ) = ⊥.

Definition vCif (v : Formula -> BivaluationDomain) : Prop :=
forall φ : Formula, (v φ = ⊥) ∨ (v ¬φ = ⊥) -> (v ¬∘φ) = ⊥.
    ⋮
Lemma bivaluation_additional :
forall (v :  Formula -> BivaluationDomain),
bivaluation v ->
vAndf v ∧  vOrf v ∧  vImpf v ∧  vNegf v ∧  vConf v ∧  vCif v ∧
vDnef v ∧  vDmANDf v ∧  vDmORf v ∧  vCipf v.
Proof. ... Qed.
```

As provas de bivaluation_lem, bivaluation_dec1 e bivaluation_dec2 seguem por análise de caso em v(φ). O lema bivaluation_additional demonstra cláusulas úteis na prova da completude em relação a semântica de bivalorações, que são derivadas das cláusulas estabelecidas originalmente. Nas provas manuais estas cláusulas foram consideradas triviais e, portanto, não foram demonstradas. Este lema segue por análise de caso em v(φ) e v(ψ).

4.2.5 Deduction_metatheorem.v

Este arquivo contém a prova do metateorema da dedução dentro da biblioteca, análoga à prova manual do Teorema 1. Antes de provar este teorema, alguns lemas e proposições auxiliares foram demonstrados.

```
Proposition lfi1_reflexivity :
forall (Γ : Ensemble Formula) (φ : Formula),
  φ ∈ Γ -> Γ ⊢ φ.
Proof. ... Qed.

Proposition lfi1_monotonicity :
forall (Γ Δ : Ensemble Formula) (φ : Formula),
  Δ ⊢ φ ∧ Δ ⊆ Γ -> Γ ⊢ φ.
Proof. ... Qed.

Proposition lfi1_cut :
forall (Γ Δ : Ensemble Formula) (φ : Formula),
  Δ ⊢ φ ∧ (forall (δ : Formula), δ ∈ Δ -> Γ ⊢ δ) -> Γ ⊢ φ.
Proof. ... Qed.

Lemma id : forall (Γ : Ensemble Formula) (φ : Formula), Γ ⊢ φ → φ.
Proof. ... Qed.
```

Em particular, a prova da monotonicidade em `lfi1_monotonicity` é de suma importância no desenvolvimento das demonstrações das metapropriedades mais elaboradas. A prova desta propriedade se dá por indução estrutural na derivação $\Delta \vdash \varphi$.

Depois disso, o teorema da dedução é finalmente provado.

```
Theorem deduction_metatheorem : forall (Γ : Ensemble Formula) (α β : Formula),
((Γ ∪ [α]) ⊢ β) <-> (Γ ⊢ α → β).
Proof.
  ⋮
Qed.

Corollary proof_by_cases : forall (Γ : Ensemble Formula) (α β φ : Formula),
(Γ ∪ [α] ⊢ φ) ∧ (Γ ∪ [β] ⊢ φ) -> (Γ ∪ [α ∨ β] ⊢ φ).
Proof. ... Qed.

Corollary proof_by_cases_neg : forall (Γ : Ensemble Formula) (α φ : Formula),
(Γ ∪ [α] ⊢ φ) ∧ (Γ ∪ [¬α] ⊢ φ) -> (Γ ⊢ φ).
Proof. ... Qed.
```

A prova segue por indução estrutural na derivação $\Gamma \cup [\alpha] \vdash \beta$ na ida e na derivação $\Gamma \vdash \alpha \to \beta$ na volta. O teorema nos cede dois corolários (`proof_by_cases` e `proof_by_cases_neg`) que facilitam a prova da completude em relação à semântica de bivalorações. Ambos corolários seguem por derivações no cálculo de Hilbert utilizando o recém-provado teorema da dedução.

4.2.6 Soundness.v

Este arquivo implementa as provas desenvolvidas ao longo da Seção 3.4.2, necessárias para desenvolver o metateorema da correção.

```
Definition h_formula (α : Formula) (v : Formula -> BivaluationDomain) : MatrixDomain :=
  match (v α), (v ¬α) with
  | ⊤, ⊥ => One
  | ⊤, ⊤ => Half
  | ⊥, _ => Zero
  end.

Lemma h_valuation : forall (v : Formula -> BivaluationDomain),
bivaluation v -> valuation (fun x => h_formula x v).
Proof. ... Qed.
```

A primeira definição feita é a da função `h_formula` que constrói uma valoração sobre as matrizes a partir de uma bivaloração recebida como argumento, similar a função apresentada no Lema 2. Na prova manual deste lema, o fato desta função ser uma valoração sobre as matrizes – um homomorfismo entre a linguagem e as matrizes – foi tomado como trivial. Para o Rocq, entretando, este fato não é imediato e deve ser demonstrado, o que é feito pelo lema `h_valuation`, que segue por análise de caso nos valores de $v(\alpha)$ e $v(\neg\alpha)$ da função `h_formula`.

Depois disso, a prova do Lema 2 é desenvolvida.

```
Lemma bivaluation_matrix_lemma : forall (v : Formula -> BivaluationDomain),
bivaluation v ->
(exists (h :  Formula -> MatrixDomain),
  (forall φ : Formula, v φ = ⊤ <-> designatedValue (h φ)) ∧ valuation h).
Proof.
  ⋮
Qed.

Corollary matrix_bivaluation_imp : forall (Γ : Ensemble Formula) (α : Formula),
(Γ ⊨m α) -> (Γ ⊨ α).
Proof. ... Qed.
```

O lema `bivaluation_matrix_lemma` segue por análise de caso em $v(\varphi)$ e $v(\neg\varphi)$, utilizando a valoração obtida ao aplicar a função `h_formula` numa bivaloração v qualquer e o

resultado obtido no lema `h_valuation`. Com este resultado, a prova da ida da equivalência entre os sistemas semânticos é desenvolvida em `matrix_bivaluation_imp`, utilizando as definições das relações de consequência semântica matricial e de bivalorações, como feito no Corolário 3.

Finalmente, o Teorema 2 e o Corolário 4 são desenvolvidos.

```
Theorem soundness_matrix : forall (Γ : Ensemble Formula) (α : Formula),
(Γ ⊢ α) -> (Γ ⊨m α).
Proof. ... Qed.

Corollary soundness_bivaluations : forall (Γ : Ensemble Formula) (α : Formula),
(Γ ⊢ α) -> (Γ ⊨ α).
Proof. ... Qed.
```

Em `soundness_matrix`, a correção em relação à semântica matricial é provada por indução na derivação $\Gamma \vdash \alpha$, mostrando a validade de todos os axiomas e da regra *modus ponens*. Esta análise de caso é facilitada pelas táticas `repeat` e `try` aplicadas em todos os *subgoals* de prova simultaneamente, evitando repetições desnecessárias de casos de prova análogos entre si.

O corolário `soundness_bivaluations` segue pela aplicação de `soundness_matrix` na hipótese $\Gamma \vdash \alpha$, o que resulta em $\Gamma \vDash_{\mathrm{m}} \alpha$. Depois disso, a aplicação de `matrix_bivaluation_imp` em $\Gamma \vDash_{\mathrm{m}} \alpha$ nos dá $\Gamma \vDash \alpha$, completando a prova.

Com estes resultados, a prova de que a lógica **LFI1** se trata, de fato, de uma **LFI** forte (em relação a $\neg$ e $\circ$) é desenvolvida, analogamente ao que foi feito no Corolário 5.

```
(** LFI1 is an LFI w.r.t ¬ and ∘, i.e.
1) exists (α β : Formula), ∼(α, ¬α ⊢ β)
2) exists (α β : Formula), ∼(α, ∘α ⊢ β) /\ ∼(¬α, ∘α ⊢ β)
3) forall (α β : Formula), ∘α, α, ¬α ⊢ β
*)
Fixpoint valuation_condition_1 (x : Formula) : MatrixDomain :=
match x with
| #0 => Half
| a ∧ b => (valuation_condition_1 a) ∧m (valuation_condition_1 b)
| a ∨ b => (valuation_condition_1 a) ∨m (valuation_condition_1 b)
| a → b => (valuation_condition_1 a) →m (valuation_condition_1 b)
| ¬a => ¬m(valuation_condition_1 a)
| ∘a => ∘m(valuation_condition_1 a)
| _ => Zero
end.

Fixpoint valuation_condition_2_1 (x : Formula) : MatrixDomain :=
match x with
| #0 => One
| a ∧ b => (valuation_condition_2_1 a) ∧m (valuation_condition_2_1 b)
```

```
| a ∨ b => (valuation_condition_2_l a) ∨_m (valuation_condition_2_l b)
| a → b => (valuation_condition_2_l a) →_m (valuation_condition_2_l b)
| ¬a => ¬_m(valuation_condition_2_l a)
| ∘a => ∘_m(valuation_condition_2_l a)
| _ => Zero
end.

Fixpoint valuation_condition_2_r (x : Formula) : MatrixDomain :=
match x with
| #0 => Zero
| ¬#0 => One
| a ∧ b => (valuation_condition_2_r a) ∧_m (valuation_condition_2_r b)
| a ∨ b => (valuation_condition_2_r a) ∨_m (valuation_condition_2_r b)
| a → b => (valuation_condition_2_r a) →_m (valuation_condition_2_r b)
| ¬a => ¬_m(valuation_condition_2_r a)
| ∘a => ∘_m(valuation_condition_2_r a)
| _ => Zero
end.

Proposition lfi1_is_lfi :
(exists (α β : Formula), ~([α] ∪ [¬α] ⊢ β)) ∧
(exists (α β : Formula), ~([∘α] ∪ [α] ⊢ β) ∧ ~([∘α] ∪ [¬α] ⊢ β)) ∧
(forall (α β : Formula), ([∘α] ∪ [α] ∪ [¬α]) ⊢ β).
Proof. ... Qed.
```

A prova das condições 1 e 2 dependem da definição de valorações específicas que servem de contraexemplo para a validade da derivação, utilizando a contrapositiva da recém-provada correção (`soundness_matrix`). Estas valorações são definidas recursivamente em `valuation_condition_1`, `valuation_condition_2_l` e `valuation_condition_2_r`. A prova da condição 3 é feita por derivação no cálculo de Hilbert, utilizando do teorema da dedução (`deduction_metatheorem`) para facilitar o desenvolvimento.

4.2.7 Cardinality.v

Este arquivo implementa alguns conceitos relacionados à cardinalidade de tipos indutivos, utilizados para provar que um dado tipo possui o mesmo número de elementos que outro. O módulo `IndefiniteDescription` contém um axioma chamado `constructive_indefinite_description`, que torna possível a definição de funções de escolha, necessárias para a prova do lema de Lindenbaum (Lema 4). Este módulo também importa arquivos que implementam o axioma da escolha. Já o módulo `Classical_sets` contém o princípio do terceiro excluído como axioma.

```
From Stdlib Require Import Classical_sets IndefiniteDescription.

Theorem strong_lem : forall P : Prop, {P} + {~P}.
Proof. ... Qed.

Theorem contraposition : forall A B : Prop, (A -> B) <-> (~B -> ~A).
Proof. ... Qed.

Inductive image_set {A B : Type} (f : A -> B) (M: Ensemble A) : Ensemble B :=
image_intro : forall a, a ∈ M -> (f a) ∈ (image_set f M).

Definition function_injective {A B : Type} (f: A -> B): Prop :=
  forall a1 a2, f a1 = f a2 -> a1 = a2.

Definition function_surjective {A B : Type} (f: A -> B): Prop :=
  forall b, exists a, f a = b.

Definition inverse_function {A B : Type} (f : A -> B) (g : B -> A) : Prop :=
  forall x : A, g (f x) = x.

Record injection (A B: Type): Type := Build_injection {
  inj_f :> A -> B;
  in_inj : function_injective inj_f
}.

Record surjection (A B: Type): Type := Build_surjection {
  sur_f :> A -> B;
  su_surj : function_surjective sur_f
}.
```

O teorema `strong_lem` representa uma versão forte do princípio do terceiro excluído, provado utilizando a versão normal do terceiro excluído juntamente com o axioma `constructive_indefinite_description`, que possibilita a definição do conjunto de Lindenbaum.

O teorema `contraposition` define a versão forte da prova por contraposição, permitindo provas de $A \to B$ a partir de provas do tipo $\sim B \to \sim A$.

Os `Records injection` e `surjection` facilitam o uso de funções injetoras e sobrejetoras, pois elementos destes tipos carregam consigo predicados sobre suas funções, que caracterizam-nas de acordo com sua injetividade e sobrejetividade.

4.2.8 Completeness.v

Este arquivo implementa as provas desenvolvidas ao longo das Seções 3.4.3, 3.4.4 e 3.4.5, referentes aos metateoremas da completude (tanto em relação às matrizes, quanto às bivalorações) e da equivalência entre os sistemas semânticos.

```
Proposition In_lem {U : Type} : forall (A : Ensemble U) (x : U),
  x ∈ A ∨ x ∉ A.
Proof. intros. apply classic. Qed.

Definition maximal_nontrivial (Γ : Ensemble Formula) (φ : Formula) : Prop :=
  ~ Γ ⊢ φ ∧ (forall (ψ : Formula), ψ ∉ Γ -> (Γ ∪ [ψ] ⊢ φ)).

Definition closed_theory (Γ : Ensemble Formula) : Prop :=
forall φ : Formula, Γ ⊢ φ <-> φ ∈ Γ.

Lemma maximal_nontrivial_is_closed : forall (Γ : Ensemble Formula) (φ : Formula),
  maximal_nontrivial Γ φ -> closed_theory Γ.
Proof. ... Qed.
```

O teorema `In_lem` representa o princípio do terceiro excluído aplicado à pertinência de um dado elemento a um conjunto. Os predicados `maximal_nontrivial` e `closed_theory` implementam, respectivamente, as Definições 23 e 24 sobre conjuntos de fórmulas bem formadas. O lema `maximal_nontrivial_is_closed` prova que todo conjunto maximal não trivial de fórmulas é uma teoria fechada (Lema 3). Isto é feito por análise de caso em $\gamma \in \Gamma$, onde γ é uma fórmula qualquer.

Depois destas definições e lemas, é preciso estabelecer uma bivaloração específica que permite o desenvolvimento da prova de completude.

```
Definition completeness_valuation (Γ : Ensemble Formula) :
Formula -> BivaluationDomain :=
fun x =>
  match (strong_lem (x ∈ Γ)) with
  | left _ => ⊤
  | right _ => ⊥
  end.

Lemma completeness_valuation_is_bivaluation :
forall (Γ : Ensemble Formula) (φ : Formula),
  (maximal_nontrivial Γ φ) -> bivaluation (completeness_valuation Γ).
Proof. ... Qed.
```

A função `completeness_valuation` é provada como sendo uma bivaloração pelo lema `complete-`

ness_valuation_is_bivaluation, por análise de caso, mostrando que a função respeita todas as cláusulas necessárias (Definição 21), como feito no Lema 5.

Depois disso, uma seção para a prova do lema de Lindenbaum é criada. Esta seção contém duas variáveis genéricas Γ e φ e uma hipótese de que $\Gamma \nvdash \varphi$.

```
Section Lindenbaum.

Variable (Γ : Ensemble Formula)
         (φ : Formula).
Hypothesis Gamma_does_not_derive_phi : ~Γ ⊢ φ.
Fixpoint Γ_i
  (i : nat) (f: nat -> Formula) : Ensemble Formula :=
  match i with
  | 0   => Γ
  | S n => match (strong_lem (((Γ_i n f) ∪ [f n]) ⊢ φ)) with
           | left _  => (Γ_i n f)
           | right _ => (Γ_i n f) ∪ [f n]
           end
  end.
Definition Δ
  (f: nat -> Formula) : Ensemble Formula :=
fun (ψ : Formula) => exists n : nat, ψ ∈ (Γ_i n f).
```

A função recursiva Γ_i representa a função de escolha Γ_i definida no início do Lema 4 e define um conjunto (Ensemble) de fórmulas. A função f recebida como argumento representa a organização das fórmulas da linguagem numa sequência. Já a função Δ representa a união generalizada de todos os conjuntos definidos por Γ_i ($\Delta = \bigcup_{i=0}^{\infty} \Gamma_i$).

Então, alguns fatos sobre estes conjuntos são provados.

```
Fact Γ_i_included_Δ :
forall (i : nat) (f : nat -> Formula),
  (Γ_i i f) ⊆ (Δ f).
Proof. ... Qed.

Fact Γ_i_does_not_derive_φ :
forall (i : nat) (f : nat -> Formula),
  ~((Γ_i i f) ⊢ φ).
Proof. ... Qed.

Fact Γ_i_m_included_Γ_i_n :
forall (f : nat -> Formula) (m : nat) (n : nat),
m <= n -> (Γ_i m f) ⊆ (Γ_i n f).
Proof. ... Qed.
```

Os fatos Γ_i_included_Δ e Γ_i_does_not_derive_φ são provados por indução em i.

O fato Γ_i_m_included_Γ_i_n é provado utilizando a definição da função Γ_i.

```
Fact Δ_Γi_con :
  forall (f :  nat -> Formula) (δ : Formula),
(Δ f) ⊢ δ -> (exists n : nat, (Γi n f) ⊢ δ).
Proof. ... Qed.

Fact Δ_does_not_derive_φ :
  forall (f :  nat -> Formula),
  ~ (Δ f) ⊢ φ.
Proof. ... Qed.

Fact not_in_Δ_Γi : forall (ψ : Formula) (f : nat -> Formula),
  ψ ∉ (Δ f) -> forall n : nat, ψ ∉ (Γi n f).
Proof. ... Qed.
```

O fato Δ_Γ_i_con segue por indução na hipótese Δ f $\vdash$ δ.

O fato Δ_does_not_derive_φ segue utilizando as provas de Γ_i_does_not_derive_φ e Δ_Γ_i_con para mostrar que, supondo Δ f $\vdash$ φ chega-se numa contradição.

O fato not_in_Δ_Γ_i segue pela definição de Δ.

```
Fact not_in_Γi_derives_φ : forall (f : nat -> Formula) (i : nat),
  (f i) ∉ (Γi (S i) f) -> (Γi i f) ∪ [f i] ⊢ φ.
Proof. ... Qed.

Lemma Δ_maximal_nontrivial : forall (f : surjection nat Formula),
  maximal_nontrivial (Δ f) φ.
Proof. ... Qed.

End .
```

O fato not_in_Γ_i_derives_φ segue por análise de caso em Γ_i i f $\cup$ [f (i)] $\vdash$ φ, utilizando o princípio do terceiro excluído.

O lema Δ_maximal_nontrivial segue utilizando a definição de sobrejeção (visto que, neste caso, considera-se a função que representa organização das fórmulas numa sequência como sendo uma sobrejeção), a definição maximal_nontrivial e os fatos Δ_does_not_derive_φ, not_in_Γ_i_derives_φ e lfi1_monotonicity.

Com isto, a completude da lógica **LFI1** pode enfim ser provada.

```
Axiom surjection_nat_formula : surjection nat Formula.

Theorem completeness_bivaluations : forall (Γ : Ensemble Formula) (α : Formula),
(Γ ⊨ α) -> (Γ ⊢ α).
```

```
Proof. ... Qed.

Corollary completeness_matrix : forall (Γ : Ensemble Formula) (α : Formula),
(Γ ⊨m α) -> (Γ ⊢ α).
Proof. ... Qed.

Corollary matrix_bivaluations_eq : forall (Γ : Ensemble Formula) (α : Formula),
(Γ ⊨m α) <-> (Γ ⊨ α).
Proof. ... Qed.
```

O axioma `surjection_nat_formula` estabelece a existência de uma sobrejeção entre `nat` e `Formula`, um fato que pode ser provado utilizando uma numeração de Gödel (RAATIKAINEN, 2025), mas que para os propósitos do presente trabalho foi tomado como um axioma.

O teorema `completeness_bivaluations` (relativo ao Teorema 3) segue pela contrapositiva[2] ($\Gamma \nvdash \alpha \Longrightarrow \Gamma \nvDash \alpha$) e utiliza os fatos provados no lema de Lindenbaum juntamente com a definição da bivaloração `completeness_valuation`. Este teorema nos cede dois corolários `completeness_matrix` e `matrix_bivaluations_eq`, que representam, respectivamente, a completude da **LFI1** em relação à semântica matricial (Corolário 7) e a equivalência entre os sistemas semânticos (Corolário 6). Estes corolários seguem aplicando os resultados obtidos em `completeness_bivaluations`, `soundness_matrix` e `matrix_bivaluation_imp`.

[2] Utilizando o teorema `contraposition` (implementado na Seção 4.2.1), que só pôde ser demonstrado visto que estamos assumindo o princípio do terceiro excluído como válido neste módulo.

5 CONCLUSÕES PARCIAIS

O estudo de lógicas paraconsistentes mostra-se relevante para o desenvolvimento de *softwares* capazes de lidar com informações contraditórias. Dentro desta família de lógicas, os sistemas de inconsistência formal destacam-se no contexto de bases de dados — sobretudo bases evolucionárias — já que internalizam o conceito de contraditoriedade dentro da sua linguagem. A **LFI1** é uma lógica de inconsistência formal com propriedades que facilitam o desenvolvimento de sistemas de gerenciamento de bancos de dados, por exemplo, mesmo na presença de inconsistência na base.

Assistentes de provas são ferramentas de *software* que permitem ao usuário provar teoremas sobre objetos expressos dentro de si, sem que a verificação destas provas dependa de um julgamento humano para garantir sua validade. O Coq é um assistente de provas, bem como uma linguagem de programação, robusto e com um núcleo axiomático enxuto, que possui aplicações em diferentes áreas da matemática, como lógica, linguagens formais, linguística computacional e desenvolvimento de programas seguros (BERTOT; CASTÉRAN, 2004).

O presente trabalho define a linguagem, sintaxe e semântica da **LFI1**, além de revisar e desenvolver manualmente metateoremas que evidenciam características deste sistema, como a correção, completude e o metateorema da dedução, servindo como base para o prosseguimento do TCC2, no qual propõe-se implementar uma biblioteca da lógica **LFI1** em Coq e provar metapropriedades desta lógica dentro do assistente.

Sendo assim, no que segue, é apresentada uma lista de itens que propõem-se serem explorados no TCC2, juntamente com um cronograma para a execução de cada item.

1. Definir a linguagem da **LFI1** na biblioteca;
2. Implementar a sintaxe (cálculo de Hilbert) da **LFI1**;
3. Implementar os sistemas semânticos (matricial e bivaloração) da **LFI1**;
4. Desenvolver metateoremas da **LFI1** na biblioteca.

Item	2024/1	2024/2					
	Dez	Jan	Fev	Mar	Abr	Maio	Jun
1							
2							
3							
4							

Tabela 2 – Cronograma Proposto para o TCC2

REFERÊNCIAS

ABITEBOUL, Serge; HULL, Richard; VIANU, Victor. Foundations of databases. In: ____. [S.l.: s.n.], 1995. ISBN 0-201-53771-0. Citado na página 12.

AMO, Sandra de; PAIS, Mônica Sakuray. A paraconsistent logic programming approach for querying inconsistent databases. **International Journal of Approximate Reasoning**, v. 46, n. 2, p. 366–386, 2007. ISSN 0888-613X. Special Track on Uncertain Reasoning of the 18th International Florida Artificial Intelligence Research Symposium (FLAIRS 2005). Disponível em: <https://www.sciencedirect.com/science/article/pii/S0888613X06001307>. Citado na página 12.

ÁVILA, Bráulio Coelho; ABE, Jair Minoro; PRADO, José Pacheco de Almeida. Paralog_*e*: a paraconsistent evidential logic programming language. In: **Proceedings 17th International Conference of the Chilean Computer Science Society**. [S.l.: s.n.], 1997. p. 2–8. Citado na página 13.

BARILE, Margherita. **Metatheorem**. 2024. Disponível em: <https://mathworld.wolfram.com/Metatheorem.html>. Citado na página 28.

BARRIO, Eduardo Alejandro; CARNIELLI, Walter. Volume II: New advances in Logics of Formal Inconsistency. **Logic Journal of the IGPL**, v. 28, n. 5, p. 845–850, 01 2019. ISSN 1367-0751. Disponível em: <https://doi.org/10.1093/jigpal/jzy063>. Citado na página 10.

BEALL, J.C.; RESTALL, Greg; SAGI, Gil. Logical Consequence. In: ZALTA, Edward N.; NODELMAN, Uri (Ed.). **The Stanford Encyclopedia of Philosophy**. Summer 2024. [S.l.]: Metaphysics Research Lab, Stanford University, 2024. Citado na página 22.

BERTOT, Yves; CASTÉRAN, Pierre. **Interactive theorem proving and program development. Coq'Art: The Calculus of inductive constructions.** [S.l.: s.n.], 2004. ISBN 3540208542. Citado 2 vezes nas páginas 57 e 73.

BROWN, K. **Encyclopedia of Language and Linguistics**. Elsevier Science, 2005. ISBN 9780080547848. Disponível em: <https://books.google.com.br/books?id=cxYGQfiD_1oC>. Citado na página 24.

BROWN, M. Bryson; PRIEST, Graham. Chunk and Permeate II: Bohr's Hydrogen Atom. **European Journal for Philosophy of Science**, Springer Verlag, v. 5, n. 3, p. 297–314, 2015. Citado 2 vezes nas páginas 10 e 15.

CALEIRO, Carlos; MARCOS, João. Many-valuedness Meets Bivalence: Using Logical Values in an Effective Way. **J. Mult.-Valued Log. Soft Comput.**, v. 19, p. 51–70, 01 2012. Citado na página 28.

CARNIELLI, Walter et al. **Analysis and Synthesis of Logics**: How to cut and paste reasoning systems. [S.l.]: Springer, 2008. v. 35. (Applied Logics Series, v. 35). Citado na página 14.

CARNIELLI, Walter; CONIGLIO, Marcelo; MARCOS, João. Logics of Formal Inconsistency. In: ____. **Handbook of Philosophical Logic**. [S.l.]: Springer, 2007. p. 1–93. ISBN 978-1-4020-6323-7. Citado 3 vezes nas páginas 10, 14 e 22.

CARNIELLI, Walter; CONIGLIO, Marcelo Esteban. **Paraconsistent logic: Consistency, contradiction and negation**. [S.l.]: Springer International Publishing, 2016. Citado 9 vezes nas páginas 10, 14, 18, 20, 21, 22, 24, 28 e 50.

CARNIELLI, Walter; MARCOS, João. Tableau systems for logics of formal inconsistency. In: . [S.l.: s.n.], 2001. v. 2, p. 848–852. Citado na página 22.

CARNIELLI, Walter; MARCOS, João; AMO, Sandra De. Formal inconsistency and evolutionary databases. **Logic and logical philosophy**, p. 115–152, 2000. Citado 6 vezes nas páginas 10, 11, 15, 20, 21 e 22.

CHLIPALA, Adam. **Certified programming with dependent types: A pragmatic introduction to the coq proof assistant**. [S.l.]: The MIT Press, 2019. Citado na página 11.

CHURCH, Alonzo. A formulation of the simple theory of types. **The journal of symbolic logic**, Cambridge University Press, v. 5, n. 2, p. 56–68, 1940. Citado na página 11.

CODD, E. F. A relational model of data for large shared data banks. **Commun. ACM**, Association for Computing Machinery, New York, NY, USA, v. 13, n. 6, p. 377–387, jun 1970. ISSN 0001-0782. Disponível em: <https://doi.org/10.1145/362384.362685>. Citado 2 vezes nas páginas 11 e 20.

CONCHON, Sylvain; FILLIÂTRE, Jean-Christophe. A persistent union-find data structure. In: **Proceedings of the 2007 Workshop on Workshop on ML**. New York, NY, USA: Association for Computing Machinery, 2007. (ML '07), p. 37–46. ISBN 9781595936769. Disponível em: <https://doi.org/10.1145/1292535.1292541>. Citado na página 11.

COSTA, Newton C. A. Da; ALVES, E. H. A semantical analysis of the calculi C_n. **Notre Dame Journal of Formal Logic**, Duke University Press, v. 18, n. 4, p. 621–630, 1977. Citado 2 vezes nas páginas 27 e 28.

CURRY, Haskell Brooks; FEYS, Robert. **Combinatory logic**. Amsterdam: North-Holland Amsterdam, 1958. v. 1. Citado na página 11.

GELDER, Allen Van. The alternating fixpoint of logic programs with negation. In: **Proceedings of the Eighth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems**. New York, NY, USA: Association for Computing Machinery, 1989. (PODS '89), p. 1–10. ISBN 0897913086. Disponível em: <https://doi.org/10.1145/73721.73722>. Citado na página 13.

GEUVERS, Herman. Proof assistants: History, ideas and future. **Sadhana**, Springer, v. 34, p. 3–25, 2009. Citado 2 vezes nas páginas 11 e 57.

GOTTWALD, Siegfried. Many-Valued Logic. In: ZALTA, Edward N. (Ed.). **The Stanford Encyclopedia of Philosophy**. Summer 2022. [S.l.]: Metaphysics Research Lab, Stanford University, 2022. Citado na página 26.

HARRISON, John; URBAN, Josef; WIEDIJK, Freek. History of interactive theorem proving. In: **Computational Logic**. Amsterdam: [s.n.], 2014. v. 9, p. 135–214. Citado na página 11.

HOWARD, William Alvin. The formulae-as-types notion of construction. In: CURRY, Haskell et al. (Ed.). **To H. B. Curry: Essays on Combinatory Logic, Lambda Calculus, and Formalism**. Chicago: Academic Press, 1980. Citado na página 11.

JACQUETTE, Dale (Ed.). **A Companion to Philosophical Logic**. Malden, MA, USA: Wiley-Blackwell, 2002. Citado na página 28.

JUNIOR, Arnaldo de Carvalho et al. A comprehensive review on paraconsistent annotated evidential logic: Algorithms, applications, and perspectives. **Engineering Applications of Artificial Intelligence**, v. 127, p. 107342, 2024. ISSN 0952-1976. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0952197623015269>. Citado na página 13.

LEROY, Xavier. **The CompCert C verified compiler: Documentation and user's manual**. Tese (Doutorado) — Inria, 2021. Citado na página 11.

ŁOŚ, Jerzy. The algebraic treatment of the methodology of elementary deductive systems. **Studia Logica: An International Journal for Symbolic Logic**, Springer, v. 2, p. 151–212, 1955. ISSN 00393215, 15728730. Disponível em: <http://www.jstor.org/stable/20013531>. Citado na página 50.

MARES, Edwin. Relevance Logic. In: ZALTA, Edward N.; NODELMAN, Uri (Ed.). **The Stanford Encyclopedia of Philosophy**. Summer 2024. [S.l.]: Metaphysics Research Lab, Stanford University, 2024. Citado na página 15.

MCGINNIS, Nicholas D. The unexpected applicability of paraconsistent logic: A Chomskyan route to dialetheism. **Foundations of Science**, Springer Verlag, v. 18, n. 4, p. 625–640, 2013. Citado 2 vezes nas páginas 10 e 15.

NIPKOW, T. **Rewriting Techniques and Applications: 9th International Conference, RTA-98, Tsukuba, Japan, March 30 - April 1, 1998, Proceedings**. Springer Berlin Heidelberg, 2006. (Lecture Notes in Computer Science). ISBN 9783540697213. Disponível em: <https://books.google.com.br/books?id=ESr6CAAAQBAJ>. Citado na página 57.

PAULIN-MOHRING, Christine. Introduction to the calculus of inductive constructions. In: PALEO, Bruno Woltzenlogel; DELAHAYE, David (Ed.). **All about Proofs, Proofs for All**. College Publications, 2015, (Studies in Logic (Mathematical logic and foundations), v. 55). Disponível em: <https://inria.hal.science/hal-01094195>. Citado na página 11.

PIERCE, Benjamin C. et al. **Logical Foundations**. 5.3. ed. Software Foundations, 2017. v. 1. Disponível em: <https://softwarefoundations.cis.upenn.edu/lf-current>. Citado na página 57.

PRAWITZ, Dag. Logical consequence: A constructivist view. In: SHAPIRO, Stewart (Ed.). **Oxford Handbook of Philosophy of Mathematics and Logic**. [S.l.]: Oxford University Press, 2005. Citado na página 22.

PRIEST, Graham; TANAKA, Koji; WEBER, Zach. Paraconsistent Logic. In: ZALTA, Edward N. (Ed.). **The Stanford Encyclopedia of Philosophy**. Spring 2022. [S.l.]: Metaphysics Research Lab, Stanford University, 2022. Citado na página 10.

RAATIKAINEN, Panu. Gödel's Incompleteness Theorems. In: ZALTA, Edward N.; NODELMAN, Uri (Ed.). **The Stanford Encyclopedia of Philosophy**. Summer 2025. [S.l.]: Metaphysics Research Lab, Stanford University, 2025. Citado na página 72.

RASIOWA, Helena. **The Mathematics of Metamathematics**. Warszawa: Państwowe Wydawn. Naukowe, 1963. Citado 2 vezes nas páginas 24 e 28.

RESTALL, Greg. **An Introduction to Substructural Logics**. New York: Routledge, 1999. Citado na página 22.

RUSSELL, Bertrand. **Principles of Mathematics**. Cambridge: Cambridge University Press, 1903. Citado na página 11.

RUSSELL, Bertrand. Mathematical logic as based on the theory of types. **American Journal of Mathematics**, Association for Symbolic Logic, v. 30, n. 3, p. 222–262, 1908. Citado na página 11.

SCHLICHTKRULL, Anders. New Formalized Results on the Meta-Theory of a Paraconsistent Logic. In: DYBJER, Peter; SANTO, José Espírito; PINTO, Luís (Ed.). **24th International Conference on Types for Proofs and Programs (TYPES 2018)**. Dagstuhl, Germany: Schloss Dagstuhl – Leibniz-Zentrum für Informatik, 2019. (Leibniz International Proceedings in Informatics (LIPIcs), v. 130), p. 5:1–5:15. ISBN 978-3-95977-106-1. ISSN 1868-8969. Disponível em: <https://drops.dagstuhl.de/entities/document/10.4230/LIPIcs.TYPES.2018.5>. Citado na página 12.

SIKORSKI, Roman. Algebra of formalized languages. **Journal of Symbolic Logic**, Association for Symbolic Logic, v. 31, n. 3, p. 1–31, 1966. Citado 2 vezes nas páginas 24 e 25.

SILVA, Flávio Soares Corrêa da; FINGER, Marcelo; MELO, Ana Cristina Vieira de. **Lógica para Computação**. [S.l.]: Cengage Learning, 2006. v. 1. Citado na página 16.

SILVA, Rafael Castro Goncalves. **Uma certificação em Coq do algoritmo W monádico. 2019. 78 p**. Dissertação (Mestrado) — Universidade do Estado de Santa Catarina, Programa de Pós Graduação em Computação Aplicada, 2019. Citado na página 57.

SILVEIRA, Ariel Agne 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. Citado 2 vezes nas páginas 11 e 57.

SIMONS, Peter. Jan Łukasiewicz. In: ZALTA, Edward N.; NODELMAN, Uri (Ed.). **The Stanford Encyclopedia of Philosophy**. Spring 2023. [S.l.]: Metaphysics Research Lab, Stanford University, 2023. Citado na página 12.

SUSZKO, Roman. Remarks on Łukasiewicz's three-valued logic. **Bulletin of the Section of Logic**, Department of Logic, University of Lodz, v. 4, n. 3, p. 87–90, 1975. Citado na página 28.

TARSKI, Alfred. **Logic, Semantics, Metamathematics**. Oxford,: Clarendon Press, 1956. Citado 2 vezes nas páginas 28 e 50.

The Coq Development Team. **The Coq Reference Manual**. France, 2024. Citado na página 11.

VILLADSEN, Jørgen; SCHLICHTKRULL, Anders. Formalizing a paraconsistent logic in the Isabelle proof assistant. In: ____. **Transactions on Large-Scale Data- and Knowledge-Centered Systems XXXIV: Special Issue on Consistency and Inconsistency in Data-Centric Applications**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2017. p. 92–122. ISBN 978-3-662-55947-5. Disponível em: <https://doi.org/10.1007/978-3-662-55947-5_5>. Citado na página 12.

WÓJCICKI, Ryszard. **Lectures on Propositional Calculi**. Ossolineum [Poland]: Pub. House of the Polish Academy of Sciences, 1984. Citado 3 vezes nas páginas 14, 25 e 50.

WÓJCICKI, Ryszard. An axiomatic treatment of non-monotonic arguments. **Bulletin of the Section of Logic**, Department of Logic, University of Lodz, v. 17, n. 2, p. 56–61, 1988. Citado na página 14.

WÓJCICKI, Ryszard. **Theory of Logical Calculi: Basic Theory of Consequence Operations**. Dordrecht, Boston and London: Kluwer Academic Publishers, 1988. Citado na página 14.