Apostila de Lógica

Prof. Mário Benevides

mario@cos.ufrj.br

19 de Março de 2015

UFRJ

Motivação Prática

- Álgebra de Boole
- Programação em lógica (PROLOG)
- Sistemas especialistas
- Especificação de programas
- Verificação de programas
- Banco de dados:
  - BD's dedutivos
  - Hipótese de mundo fechado
  - Default / prioridades
  - Ontologias
- Sistemas distribuídos:
  - Tempo
  - Conhecimento e crença
- Lei (Lógica deôntica)
- Linguagens de programação

Livros

- Lógica para a Computação - Thomson Learning - Flávio Soares Corrêa da Silva, Marcelo Finger e Ana Cristina Vieira de Melo
- Lógica para a Ciência da Computação - Ed. Campus - João Nunes de Souza
- A Mathematical Introduction to Logic - Enderton
- Introduction to Mathematical Logic - Mendelson
- Introdução à Lógica Modal Aplicada a Computação - Marcos Mota Costa
- Programação em Lógica e a Linguagem PROLOG - Casanova
- Lógica - John Nolt e Linnes Rohatyn

Sumário

Capítulo 1

Introdução

LÓGICA é o estudo do raciocínio dedutivo.

Histórico

- Aristóteles → leis do discurso;
- Idade Média → lógica filosófica;
- Boole (1815-1864) → álgebra booleana;
- Peano (c.1865) → axiomatização da aritmética;
- Frege (1874)
  → investigar fundamentos da matemática
  → lógica moderna;
- Russel-Whitehead (1910)
  → Princípia Matemática
  → lógica moderna;
- Hilbert (1925) →
  → formalização da noção de prova
  → mecanização da matemática;
- Gentzen (1935) → teoria da prova;
- Godel (1931-1935) →
  → completude da lógica
  → incompletude da aritmética;
- Investigar Fundamentos da Computação.

O que é lógica?

- É o estudo do raciocínio dedutivo. (informalmente);
- É um sistema formal (formalmente)

Lógica:

LINGUAGEM

+

REGRAS DE DEDUÇÃO / INFERÊNCIA

+

SEMÂNTICA

Linguagem

É usada para descrever o conhecimento que se deseja representar.

Regras de Dedução

Servem para tirar conclusões a partir do conhecimento representado na linguagem.

Semântica

Serve para dar significado aos objetos descritos na linguagem.

Tipos mais comuns de lógica:

- Lógica Clássica Proposicional
- Lógica Clássica de 1^a Ordem
- Lógicas Modais: tempo, conhecimento, ações, pogramas e etc.
- Lógicas Paraconsistente;
- Lógicas Relevantes;
- Lógicas Difusas;
- Lógicas Probabilistícas.

Hipóteses da Lógica Clássica:

- proposicões atômicas;
- proposições mais complexas podem ser (construídas decompostas) de (em) proposições atômicas;
- proposições atômicas são verdadeiras ou falsas (2 valores);
- o valor verdade de uma proposição mais complexa somente depende dos valores das proposições atômicas que a compõe.

Ex1: João ama Maria. (V ou F)

Ex2: João é estudante e João é alto.

o valor verdade só depende de sabermos se:

João é estudante. (V ou F?)

João é alto. (V ou F?)

Capítulo 2

Lógica Clássica Proposicional

Neste capítulo nós apresentaremos a Lógica Clássica Proposicional. Na seção 2.1 nós definimos a linguagem. Na seção 2.2 nós apresentamos a semântica e definimos a importante noção de conseguência lógica. Na seção 2.3 apresentamos algoritmos para verificar conseguência lógica e satisfabilidade e discutimos a complexidades destes problenas. Na seção 2.4 são apresentados alguns sistemas dedutivos. Finalmente, na seção 2.4.6, enunciamos e provamos os teoremas de Correção e Completude da Lógica Clássica Proposicional.

2.1 Linguagem da Lógica Clássica Proposicional

A linguagem proposicional é uma linguagem formal cujo objetivo é representar trechos de discurso de uma maneira precisa e sem ambigüidades. Os seguintes operadores serão usados para formar proposições mais complexas a partir de proposições mais simples:

e	$\wedge$
ou	$\vee$
se <condição> então <conclusão>	$\rightarrow$
não	$\neg$

- Para representar proposições atômicas usaremos letras maiúsculas, por exemplo: A, B, C,...

Exemplos $A \wedge B, A \vee B, A \rightarrow B, \neg A$

Sócrates é um homem.
Se Sócrates é um homem então Sócrates é mortal.

$A-$Sócrates é um homem.

$B-$Sócrates é mortal.

A
$A \rightarrow B$

Definição

Um alfabeto proposicional é composto por três conjuntos de símbolos:

- **Conectivos/operadores lógicos:** $\wedge, \vee, \rightarrow, \neg$
- **Símbolos auxiliares:** (e)
- **Símbolos proposicionais:** qualquer letra maiúscula é um símbolo proposicional (ex: A, B,..., Z).

  Podemos acrescentar um subscrito numérico a letras maiúsculas $(A_1, A_2, ...)$. Denotamos este conjunto por $\mathcal{P}$.

Apresentaremos a seguir uma gramática para definirmos quais são as fórmulas bem formadas da linguagem:

Definição A noção de fórmula bem formada, ou simplesmente fórmula, é definida, indutivamente, pelas seguintes condições:

- Qualquer símbolo proposicional é uma fórmula.
- Se α e β são fórmulas então $(\alpha \wedge \beta)$, $(\alpha \vee \beta)$, $\neg\alpha$, $(\alpha \rightarrow \beta)$ também o são;
- Nada mais é fórmula

De uma forma alternativa podemos definir a linguagem proposicional por meio de uma notação BNF.

$\alpha ::= P \mid (\alpha_1 \wedge \alpha_2) \mid (\alpha_1 \vee \alpha_2) \mid (\alpha_1 \rightarrow \alpha_2) \mid \neg\alpha$
onde P é um símbolo proposicional.

Algumas vezes utilizamos o conectivo *se e somente se* $\leftrightarrow$ que é definido como:

$(\alpha \leftrightarrow \beta) \equiv ((\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha))$

Exemplo e fórmulas bem formadas:

- $A \to B$ (não é fórmula)
- $(A) \to \neg(B)$ (não é fórmula)
- $(\neg A \vee B) \wedge (B \vee C) \to D$ (não é fórmula)
- $((A \to (B \to \neg A)) \to (A \vee B))$ (é fórmula)
- $(A \to (B \wedge C))$ (é fórmula)

Exercícios:

Questão 1. Represente as seguintes proposições utilizando a linguagem da lógica clássica proposicional. Utilize os símbolos proposicionais C (está chovendo) e N (está nevando).

(a) Está chovendo, mas não está nevando.

(b) Não é o caso que está chovendo ou nevando.

(c) Se não está chovendo, então está nevando.

(d) Não é o caso que se está chovendo então está nevando.

(e) Está chovendo se e somente se está nevando.

(f) Se está nevando e chovendo, então está nevando.

(g) Se não está chovendo, então não é o caso que está nevando e chovendo.

Nos exercícios seguintes, represente o texto na linguagem da lógica proposicional, especificando significado dos símbolos proposicionais utilizados.

Questão 2. Ela não está em casa ou não está atendendo ao telefone. Mas se ela não está em casa, então ela foi seqüestrada. E se ela não está atendendo ao telefone, ela está correndo algum outro perigo. Ou ela foi seqüestrada ou ela está correndo um outro perigo.

Questão 3. Hoje é fim-de-semana se e somente se hoje é sábado ou domingo. Hoje não é sábado. Hoje não é domingo. Portanto, hoje não é um fim-de-semana.

Questão 4. A proposta de auxílio está no correio. Se os juízes a receberem até sexta-feira, eles a analisarão. Portanto, eles a analisarão porque se a proposta estiver no correio, eles a receberão até sexta-feira.

Observação:

- Convenções sobre omissão de parênteses:
  $\neg > \wedge > \vee > \rightarrow$

- Parênteses mais externos podem ser omitidos:
  $A \rightarrow (B \rightarrow C) \equiv (A \rightarrow (B \rightarrow C))$

2.2 Semântica da Lógica Clássica Proposicional

- A semântica da lógica clássica proposicional consiste na atribuição de significado às fórmulas da linguagem.

- Isto é feito através da atribuição de valor verdade.

- Para cada fórmula é atribuído um valor verdadeiro ou falso.
  valores-verdade:
  V - verdadeiro
  F - falso

- O valor verdade de uma fórmula depende unicamente dos valores verdade atribuídos aos seus símbolos proposicionais.

Tabela Verdade

Conjunção:

A	B	$A \wedge B$
V	V	V
V	F	F
F	V	F
F	F	F

Hoje tem aula **e** hoje é quinta-feira.

Disjunção (não-exclusiva):

A	B	$A \vee B$
V	V	V
V	F	V
F	V	V
F	F	F

Hoje tem aula **ou** hoje é quinta-feira.

Negação:

A	$\neg A$
V	F
F	V

Hoje **não** tem aula.

Implicação:

Exemplo: considere o anúncio abaixo:

Para pagamento à vista, nós damos 25% de desconto na compra de qualquer TV.

Re-escrevendo:"Se pagamento à vista então 25% de desconto."

Suponha agora que D. Maria deseja verificar se este anúncio é honesto ou não.

Possibilidade	Pagamento à vista	25% de desconto	Anúncio
1	V	V	V
2	V	F	F
3	F	V	V
4	F	F	?

A	B	$A \to B$
V	V	V
V	F	F
F	V	V
F	F	V(?)

Existem lógicas que discordam da linha 4 Ex: 3-valores, intuicionista, relevante...

Exercício:

Construa a tabela verdade de: $(\neg A \vee B) \to C$

A	B	C	$\neg A$	$(\neg A \vee B)$	$(\neg A \vee B) \to C$
V	V	V			
V	V	F			
V	F	V			
V	F	F			
F	V	V			
F	V	F			
F	F	V			
F	F	F			

Função de Atribuição de Valor Verdade

A cada símbolo proposicional nós queremos atribuir um valor verdadeiro ou falso. Isto é feito através de uma função **v** de atribuição de valor verdade. $\mathbf{v}$:$\mathcal{P} \mapsto \{V, F\}$, onde $\mathcal{P}$ é conjunto dos símbolos proposicionais

Exemplos:$\mathbf{v}(A) = F$, $\mathbf{v}(B) = V$, $\mathbf{v}(C) = V$

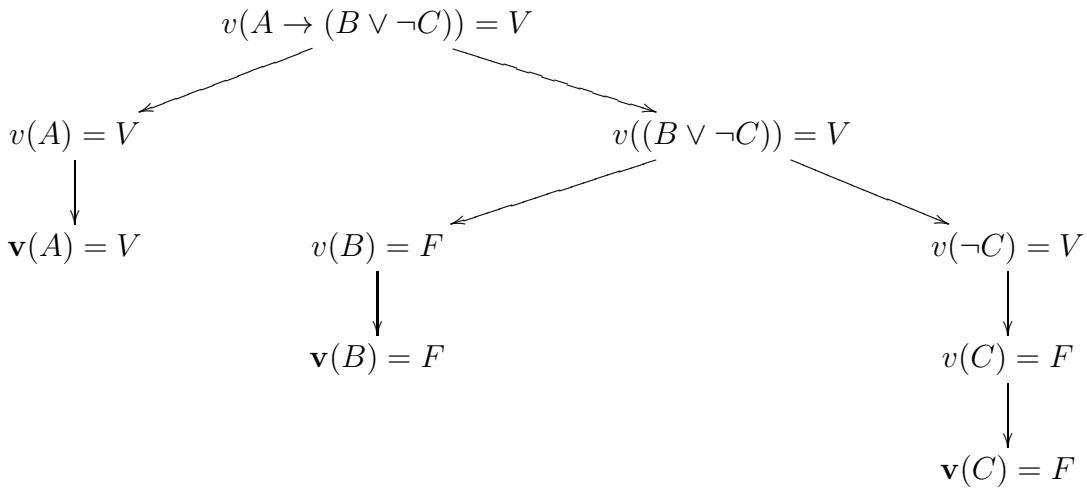
Uma vez atribuído valor verdade a cada símbolo proposicional em $\mathcal{P}$, queremos estender esta atribuição para o conjunto de todas as fórmulas da linguagem proposicional, que denotaremos por W. Na definição a seguir α e β denotam fórmulas e A denota um símbolo proposicional, isto é, $\alpha,\ \beta \in W$ e $A \in \mathcal{P}$.

Definimos uma função v de atribuição de valor verdade a fórmulas da linguagem como uma extensão da função **v** tal que:

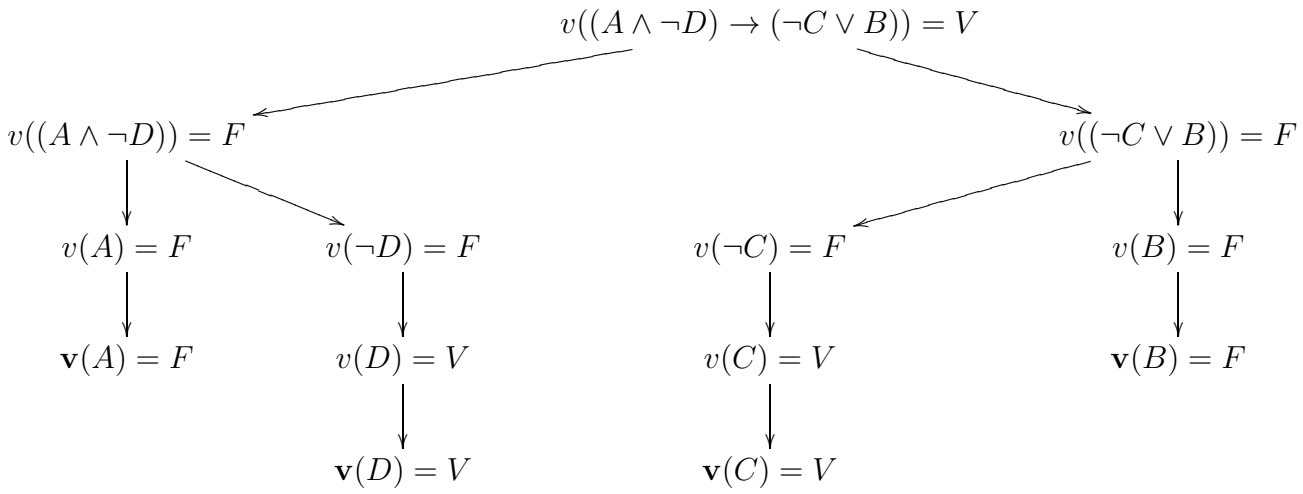
$v : W \mapsto \{V, F\}$, onde v deve satisfazer as seguintes condições:

1. $v(A) = \mathbf{v}(A), \text{se} A \in \mathcal{P}$
2. $v(\neg\alpha) = \begin{cases} V & \text{se} \quad v(\alpha) = F \\ F & \text{se} \quad v(\alpha) = V \end{cases}$
3. $v(\alpha \wedge \beta) = \begin{cases} V & \text{se} \quad v(\alpha) = v(\beta) = V \\ F & \text{caso contrário} \end{cases}$
4. $v(\alpha \vee \beta) = \begin{cases} F & \text{se} \quad v(\alpha) = v(\beta) = F \\ V & \text{caso contrário} \end{cases}$
5. $v(\alpha \to \beta) = \begin{cases} F & \text{se} \quad v(\alpha) = V \text{ e } v(\beta) = F \\ V & \text{se} \quad \text{caso contrário} \end{cases}$

Exemplo: Ache o valor verdade da seguinte fórmula para a valoração
$\mathbf{v}(A) = V,\ \mathbf{v}(B) = F,\ \mathbf{v}(C) = F$:



Exemplo: Ache o valor verdade da seguinte fórmula para a valoração
$\mathbf{v}(A) = F,\ \mathbf{v}(B) = F, \mathbf{v}(C) = V,\ \mathbf{v}(D) = V$:



Algoritmo para Construir Tabela Verdade

Quantas linhas possui uma tabela verdade para $(A \wedge \neg D) \rightarrow (\neg C \vee B)$?

Cada linha corresponde a uma possível atribuição de valores verdade aos símbolos proposicionais que compõe a fórmula. Como esta fórmula possui 4 símbolos proposicionais (A,B,C e D), sua tabela verdade deve ter $2^4 = 16$ linhas.

Tabela Verdade computa o valor verdade de uma fórmula para todas as possíveis atribuições **v** a seus símbolos proposicionais.

Logo, o problema de se saber todos os valores verdades de uma fórmula na lógica clássica proposicional, para todas as atribuições **v** a seus símbolos proposicionais, é decidível; o algoritmo é o seguinte:

passo 1: conte o número de símbolos proposicionais;
passo2: monte uma tabela com 2^n linhas e com quantas colunas for o número de subfórmulas da fómula;
passo 3: preencha as colunas dos símbolos proposicionais com V ou F alternando de cima para baixo VFVF para a 1a coluna, VVFF... para a 2a, VVVVFFFF para a 3a e assim por diante, nas potências de 2.
passo 4: compute o valor verdade das outras colunas usando as tabelas básicas fornecidas.

Exemplo: $(\neg A \rightarrow B) \vee C$

$2^3 = 8$

A	B	C	$\neg A$	$(\neg A \rightarrow B)$	$(\neg A \rightarrow B) \vee C$
V	V	V	F	V	V
V	V	F	F	V	V
V	F	V	F	V	V
V	F	F	F	V	V
F	V	V	V	V	V
F	V	F	V	V	V
F	F	V	V	F	V
F	F	F	V	F	F

Tautologias, Contradições, Fórmula Equivalentes

Existem fórmulas onde todas as linhas da Tabela Verdade dão verdade. Elas são verdadeiras não importando os valores verdade que atribuímos aos seus símbolos proposicionais. Estas fórmulas são chamadas **tautologias**. Da mesma forma, existem fórmulas que são sempre falsas, independente dos valores verdade atribuídos aos seus símbolos proposicionais. Estas são chamadas **contradições**. Além disso, existem fórmulas que, embora diferentes, têm tabelas verdade que coincidem linha a linha. Tais fórmulas são ditas **equivalentes**.

Exemplos:

A	$A \rightarrow A$
V	V
F	V

$A \rightarrow A$ é uma tautologia.

A	B	$B \to A$	$A \to (B \to A)$
V	V	V	V
V	F	V	V
F	V	F	V
F	F	V	V

A	B	$(B \vee A)$	$\neg(A \vee B)$	$A \wedge \neg(A \vee B)$
V	V	V	F	F
V	F	V	F	F
F	V	V	F	F
F	F	F	V	F

$A \wedge \neg(A \vee B)$ é uma contradição.

A	B	$B \wedge A$	$\neg(A \wedge B)$
V	V	V	F
V	F	F	V
F	V	F	V
F	F	F	V

A	B	$\neg A$	$\neg B$	$\neg A \vee \neg B$
V	V	F	F	F
V	F	F	V	V
F	V	V	F	V
F	F	V	V	V

$\neg(A \wedge B)$ é equivalente a $\neg A \vee \neg B$.

Definição 1 *Tautologia e Contradição:*

- *Uma fórmula α é uma tautologia se e somente se, para toda atribuição $\boldsymbol{v}$, $v(\alpha) = V$.*
- *Uma fórmula α é uma contradição se e somente se, para toda atribuição $\boldsymbol{v}$, $v(\alpha) = F$.*

Exemplos de tautologias "famosas":

- $A \vee \neg A$
- $A \rightarrow A$
- $(A \rightarrow ((A \rightarrow B) \rightarrow B)$
- $A \wedge B \rightarrow A$
- $A \wedge B \rightarrow B$
- $\neg\neg A \rightarrow A$
- $A \rightarrow A \vee B$
- $B \rightarrow A \vee B$
- $((A \rightarrow B) \wedge \neg B) \rightarrow \neg A$

Exemplos de contradições:

- $A \wedge \neg A$
- $\neg(A \rightarrow A)$
- $A \wedge (A \rightarrow B) \wedge \neg B$

Exercício

Verificar se estas fórmulas são realmente tautologias e contradições.

Definição 1 *Equivalência entre Fórmulas:*

Duas fórmulas α e β são ditas equivalentes, $\alpha \equiv \beta$, se e somente se, para toda atribuição $\boldsymbol{v}$, $v(\alpha) = v(\beta)$.

Intuitivamente, duas fórmulas são equivalentes se, linha a linha, elas tem a mesma tabela verdade.

Exemplos de equivalências:

$\neg\neg A \equiv A$

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

Exercício: Verificar se as seguintes fórmulas são equivalentes:

1. $\neg(A \wedge B) \equiv \neg A \vee \neg B$
2. $\neg(P \rightarrow Q) \equiv (P \wedge \neg Q)$
3. $P \wedge (Q \vee R) \equiv (P \wedge Q) \vee (P \wedge R)$
4. $P \rightarrow Q \equiv \neg Q \rightarrow \neg P$
5. $P \vee (Q \wedge R) \equiv (P \vee Q) \wedge (P \vee R)$

Observação:

Utilizando a noção de equivalência, é possível definir alguns dos conectivos a partir de outros. Por exemplo, utilizando a negação ($\neg$) e mais um conectivo qualquer ($\wedge$, $\vee$ ou $\rightarrow$) podemos definir todos os outros. Assim:

Definimos $\rightarrow$ e $\wedge$ usando $\neg$ e $\vee$

$P \rightarrow Q \equiv \neg P \vee Q$

$P \wedge Q \equiv \neg(\neg P \vee \neg Q)$

Definimos $\rightarrow$ e $\vee$ usando $\neg$ e $\wedge$

$P \rightarrow Q \equiv \neg(P \wedge \neg Q)$

$P \vee Q \equiv \neg(\neg P \wedge \neg Q)$

Definimos $\wedge$ e $\vee$ usando $\neg$ e $\rightarrow$

$P \wedge Q \equiv \neg(P \rightarrow \neg Q)$

$P \vee Q \equiv \neg P \rightarrow Q$

Exercício: Verificar as equivalências acima.

Na verdade todos os conectivos podem ser definido a partir de um único novo conectivo chamado. Isto é o que vamos ver no exercício seguinte.

Exercício: (Sheffer Stroke P/Q)

Suponha P/Q é uma fórmula com a seguinte Tabela Verdade

P	Q	P/Q
V	V	F
V	F	F
F	V	F
F	F	V

Defina $\wedge, \vee, \neg, \rightarrow$ usando "/" Dica: $\neg P \equiv$(P/P)

P	$\neg P$	P/P
V	F	F
F	V	V

Dada uma tabela verdade, como saber a fórmula correspondente?

P	Q	R	S	α
V	V	V	V	V
F	V	V	V	V
V	F	V	V	V
F	F	V	V	F
V	V	F	V	F
F	V	F	V	F
V	F	F	V	F
F	F	F	V	F
V	V	V	F	F
F	V	V	F	F
V	F	V	F	F
F	F	V	F	F
V	V	F	F	F
F	V	F	F	F
V	F	F	F	F
F	F	F	F	F

Linhas em que α é verdadeira:

P	Q	R	S	α
V	V	V	V	V
F	V	V	V	V
V	F	V	V	V

Logo:

- $P \wedge Q \wedge R \wedge S$
- $\neg P \wedge Q \wedge R \wedge S$
- $P \wedge \neg Q \wedge R \wedge S$

$$(P \wedge Q \wedge R \wedge S) \vee (\neg P \wedge Q \wedge R \wedge S) \vee (P \wedge \neg Q \wedge R \wedge S)$$

Formalizando: Para achar a fórmula correspondente a uma tabela verdade, procedemos da seguinte maneira:

1. Para cada linha da tabela verdade em que a fórmula α é verdadeira, escrevemos uma fórmula correspondendo à conjunção (E) dos símbolos proposicionais que forem verdadeiros e das negações daqueles que forem falsos na linha considerada.

2. A fórmula α é a disjunção (OU) das fórmulas escritas no passo 1.

Definição 1 *Átomo e Literal*

- *Uma fórmula atômica ou átomo é qualquer símbolo proposicional. Ex:* A, *e* C.

- *Um literal é um átomo ou sua negação. Ex:* $A, \neg A, B, \neg C$.

Forma Normal Disjuntiva(FND):

Uma fórmula α está na forma normal disjuntiva se e somente se α tem a seguinte forma:

$\alpha = C_1 \vee C_2 \vee ... \vee C_n$

onde cada $C_i = (Q_1 \wedge Q_2 \wedge ... \wedge Q_m),\ 1 \leq i \leq n$, e Q_j, $1 \leq j \leq m$, são literais, isto é, cada C_i é uma conjunção de literais. E α é um disjunção de conjunções de literais.

Forma Normal Conjuntiva(FNC):

Uma fórmula α está na forma normal conjuntiva se e somente se α tem a seguinte forma:

$\alpha = D_1 \wedge D_2 \wedge ... \wedge D_n$

onde cada $D_i = (Q_1 \vee Q_2 \vee ... \vee Q_m),\ 1 \leq i \leq n$ e Q_j, $1 \leq j \leq m$, são literais, isto é, cada D_i é uma disjunção de literais. E α é um conjunção de disjunções de literais.

Algoritmo: Converter fórmulas para a FNC:

passo1: elimine o conectivo $\rightarrow$ usando:

$\alpha \rightarrow \beta \equiv (\neg\alpha \vee \beta)$

$\neg(\alpha \rightarrow \beta) \equiv (\alpha \wedge \neg\beta)$

passo 2: mova a negação ($\neg$) para o interior da fórmula, usando as seguintes regras:

$\neg\neg\alpha \equiv \alpha$

$\neg(\alpha \wedge \beta) \equiv (\neg\alpha \vee \neg\beta)$

$\neg(\alpha \vee \beta) \equiv (\neg\alpha \wedge \neg\beta)$

passo3: mova as conjunções para o exterior da fórmula usando:

$\alpha \vee (\beta \wedge \gamma) \equiv (\alpha \vee \beta) \wedge (\alpha \vee \gamma)$

$(\alpha \wedge \beta) \vee \gamma \equiv (\alpha \vee \gamma) \wedge (\beta \vee \gamma)$

Exemplo:

$(A \vee B) \rightarrow C \Rightarrow\ passo1\ \Rightarrow \neg(A \vee B) \vee C \Rightarrow\ passo2\ \Rightarrow$

$(\neg A \wedge \neg B) \vee C \Rightarrow\ passo3\ \Rightarrow (\neg A \vee C) \wedge (\neg B \vee C)$ FNC

Exercício:

Fazer um algoritmo para converter fórmulas para a forma normal disjuntiva.

Definição:

Seja α uma fórmula e Γ um conjunto de fórmulas:

1. Uma atribuição de valor verdade $\mathbf{v}$:$\mathcal{P} \mapsto \{V, F\}$ satisfaz α se e somente se $v(\alpha) = V$. E $\mathbf{v}$ satisfaz Γ se e somente se v satisfaz cada membro de Γ.

2. Γ é satisfatível se e somente se existe uma atribuição $\mathbf{v}$ que satisfaz Γ. Caso contrário, Γ é insatisfatível.

Definição:

Uma fórmula β é dita uma ***consequência logica*** de um conjunto de fórmulas $\Gamma = \{\alpha_1, \alpha_2, ..., \alpha_n\}$, ou β é uma ***implicação logica*** de Γ, $\alpha_1, \alpha_2, ..., \alpha_n \models \beta$, se somente se para toda valoração $\mathbf{v}$ se $v(\alpha_1 \wedge \alpha_2 \wedge ... \wedge \alpha_n) = V$, então $v(\beta) = V$.

Teorema:

Uma fórmula β é dita uma consequência logica de um conjunto de fórmulas $\Gamma = \{\alpha_1, \alpha_2, ..., \alpha_n\}$, ou seja, $\alpha_1, \alpha_2, ..., \alpha_n \models \beta$ se somente se $(\alpha_1 \wedge \alpha_2 \wedge ... \wedge \alpha_n) \rightarrow \beta$ é uma tautologia.

Exemplo:

$(C \vee T) \wedge \neg T \rightarrow C$ é tautologia $\Rightarrow (C \vee T) \wedge \neg T \models C$

2.3 Complexidade

Nesta seção gostariamos de investigar dois problemas distintos:

Problema 1: Dada uma fórmula φ com comprimento n e uma valoração v para os símbolos proposicionais. Qual a complexidade de se calcular o valor de $v(\varphi)$ para a atribuição $\mathbf{v}$? Calcular $\nu(\varphi, \mathbf{v})$.

Onde o comprimento de uma fórmula é o número de símbolos da fórmula, i.e., núumero de símbolos proposicionais + número de conectivos lógicos.

A seguir especificamos uma função $\nu(\varphi,\mathbf{v})$ que implementa a função $v(\varphi)$ para a atribuição $\mathbf{v}$.

Função $\nu(\varphi,\mathbf{v})$: **bool**

caso φ

$= P$ onde P é um símbolo proposicional, **retorna** $\mathbf{v}(P)$;

$= \neg\varphi_1$, **retornar NOT** $\nu(\varphi_1,\mathbf{v})$;

$= \varphi_1 \wedge \varphi_2$, **retornar** $\nu(\varphi_1,\mathbf{v})$ **AND** $\nu(\varphi_2,\mathbf{v})$;

$= \varphi_1 \vee \varphi_2$, **retornar** $\nu(\varphi_1,\mathbf{v})$ **OR** $\nu(\varphi_2,\mathbf{v})$;

$= \varphi_1 \rightarrow \varphi_2$, **retornar NOT** $\nu(\varphi_1,\mathbf{v})$ **OR** $\nu(\varphi_2,\mathbf{v})$;

Complexidade da função $\nu(\varphi,\mathbf{v})$ é O(n), pois a função é chamada uma vez para cada símbolo proposicional e uma vez para cada conectivo lógico.

Problema 2: Dada uma fórmula φ com comprimento n e m símbolos proposicionais. Verificar se existe alguma valoração que satisfaz φ.

Função SAT(φ): **bool**

para cada valoração **v faça**

se $\nu(\varphi,\mathbf{v})$ **então retorna verdadeiro**

retorna falso

Complexidade da função SAT(φ)

Complexidade da função SAT $\approx$ número de valorações diferentes $\times$ complexidade de $\nu(\varphi,\mathbf{v})$

Complexidade da função SAT $\approx O(2^m) \times O(n) \approx O(2^m.n)$

Obs.:

1) problema 1 é polinomial (linear) no comprimento da fórmula;

2) problema 2 é NP completo.

2.4 Sistemas Dedutivos

Nas seções anteriores apresentamos a linguagem e a semântica da Lógica Clássica Proposicional. Voltaremos agora a problema central deste curso. Dado Um banco de dados (conjunto de fórmulas), BD = $\{\alpha_1, \cdots, \alpha_n\}$, e uma pergunta (fórmula), α com saber se o banco de dados implica logicamente na pergunta.

Nós já temos um algoritmo para responder BD $\models \alpha$ montando a tabela verdade para $\alpha_1 \wedge \alpha_2 \wedge ... \wedge \alpha_n \rightarrow \alpha$. Se for uma tautologia responde SIM, senão responde NãO.

A complexidade deste algoritmo é da ordem de 2^n, onde n é o número de símbolos proposicionais em $\alpha_1 \wedge \alpha_2 \wedge ... \wedge \alpha_n \rightarrow \alpha$.

2.4.1 Dedução Natural

- Jakowski(1924) e Gentzen(1935)
- Somente regras de inferência
- Para cada conectivo lógico temos 2 regras:

1. Regra de Introdução
2. Regra de Eliminação
3. Regra de Introdução: como uma fórmula contendo o conectivo pode ser inferida
4. Regra de Eliminação: que consequências podemos tirar de uma fórmula contendo o conectivo

Queremos escrever regras de inferência que sejam válidas, isto é, que as premissas impliquem logicamente nas conclusões.

Regras de inferência:

$$\frac{P^1, P^2, ..., P^n}{C}$$

Se todas as premissas $P^1, P^2, ..., P^n$ forem verdadeiras, então a conclusão C é verdadeira.

$P^1, P^2, ..., P^n \vdash C$

Exemplo: A seguintes regras são válidas?

$$\frac{\alpha}{\alpha \wedge \neg\alpha}$$

Não, pois α não implica logicamente $\alpha \wedge \neg\alpha$, Pois, $\alpha \wedge \neg\alpha$ é sempre falso (contradição).

$$\frac{\neg\alpha \quad \alpha \vee \beta}{\beta}$$

É válida, pois se $v(\neg\alpha) = V$ e $v(\alpha \vee \beta) = V$, então $v(\beta) = V$ porque $v(\alpha) = F$ e para $v(\alpha \vee \beta) = V$, $v(\beta) = V$.

Regras de Inferência:

Primeiro apresentaremos as regras que não utilizam o conceito de suposição para em seguida introdizir este conceito e apresentar as regras restantes.

Conjunção $\wedge$

$\wedge$**-I**

$$\frac{\alpha \ \beta}{\alpha \wedge \beta}$$

Se BD $\vdash \alpha$ e BD $\vdash \beta$, então responda sim para $\alpha \wedge \beta$. Isto é BD $\vdash \alpha \wedge \beta$.

$\wedge$-I é válida? Sim, pois se $v(\alpha) = v(\beta) = V$ então $v(\alpha \wedge \beta) = V$

$\wedge$-E

$$\frac{\alpha \wedge \beta}{\alpha} \qquad \frac{\alpha \wedge \beta}{\beta}$$

$\wedge$-E é válida? Sim, pois se $v(\alpha \wedge \beta) = V$ então $v(\alpha) = v(\beta) = V$

Disjunção $\vee$

$\vee$-I

$$\frac{\alpha}{\alpha \vee \beta} \qquad \frac{\beta}{\alpha \vee \beta}$$

É válida? Sim, pois se $v(\alpha) = V$ então $v(\alpha \vee \beta) = V$

$\vee$ **-E** Esta regra envolve o conceito de suposição que veremos a seguir. Iremos apresentá-la e seguir.

Implicação $\rightarrow$

$\rightarrow$ E

$$\frac{\alpha, \alpha \rightarrow \beta}{\beta}$$

Se $v(\alpha) = V$ e $v(\alpha \rightarrow \beta) = V$ então $v(\beta) = V$

Exemplo:

quinta-feira

se quinta-feira então aula de lógica

aula de lógica

Exemplo 1:

1. $A \wedge B$

2.$A \rightarrow C$

3.$C \rightarrow D$

$BD \vdash D$

4. $A \wedge \wedge E(1)$

5. C $\rightarrow$ E(2,4)

6. D $\rightarrow$ E(3,5)

$\rightarrow$ **I** Esta regra envolve o conceito de suposição que veremos a seguir. Iremos apresentá-la em seguida.

Negação: $\neg$

$\neg - I$

$$\frac{\alpha}{\neg\neg\alpha}$$

$\neg - E_1$

$$\frac{\neg\neg\alpha}{\alpha}$$

Redução ao Absurdo: Esta regra envolve o conceito de suposição que veremos a seguir. Iremos apresentá-la em seguida.

Exemplo:

BD = $\{A \wedge B\}$ BD $\vdash A \vee B$?

1. $A \wedge B$

2. $A \wedge$E1(1)

3. $A \vee B \vee$I(2)

Vamos introduzir agora o impotante conceito de **suposição**. Informalmente, uma suposição é uma fórmula que supomos ser verdadeira para concluirmos algo que depende desta suposição. Representamos suposições como fórmulas entre [...], por exemplo, $[A \wedge B]$.

Vamos começar com dois exemplos.

No meu banco de dados eu tenho uma axiomatização das leis da Mecanica Clássica e que eu gostaria de estabelecer a seguinta proposição condicional:

"**Se** corpo solto no ar **então** corpo cai"

BD = leis da Mecanica Clássica

Suponho que o corpo está solto no ar.

Provo, usando esta suposição e as leis do BD que

o corpo cai

Se corpo solto no ar **então** corpo cai

Porém, a suposição que o corpo está solto no ar deve ser descartada (descarregada) após a proposição condicional ter sido estabelecida.

Um segundo exemplo seria:

No meu banco de dados eu tenho uma axiomatização da Aritmética e eu gostaria de estabelecer a seguinta proposição condicional:

"**Se** n é impar **então** sucessor de n é par"

BD = axiomatização da Aritmética

Suponho que n é impar.

Provo, usando esta suposição e as leis do BD que

sucesor de n 'e par

Se n é impar **então** sucessor de n é par

De novo, temos que descarrgar a suposição que n é impar após a proposição condicional ter sido estabelecida.

Introdução da $\rightarrow$

$\rightarrow$ **I**

$$\frac{\begin{array}{c}[\alpha]^i \\ \vdots \\ \beta\end{array}}{\alpha \rightarrow \beta^i}$$

Onde $[\alpha]$ é uma suposição

Exemplo:

BD $= \{A \rightarrow C,\ C \rightarrow D\}$

Pergunta: $A \rightarrow D$

BD $\vdash A \rightarrow D$

1.	$A \rightarrow C$		BD
2.	$C \rightarrow$ D		BD
3.	$[A]^1$		Suposição
	3.1	C	$\rightarrow$ E(3,1)
	3.2	D	$\rightarrow$ E(3.1,2)
4.	A$\rightarrow D^1$		$\rightarrow$ I(3,3.2)

Eliminacao da $\vee$

$\vee$ **-E**

$$\frac{\alpha \vee \beta \qquad \begin{array}{c}[\alpha]^i \\ \vdots \\ \gamma\end{array} \qquad \begin{array}{c}[\beta]^j \\ \vdots \\ \gamma\end{array}}{\gamma^{ij}}$$

Exemplo:

Hoje é terça-feira $\vee$ Hoje é quinta-feira.

Se Hoje é terça-feira *então* aula de lógica.

$$\frac{\textit{Se}\text{ Hoje é quinta-feira }\textit{então}\text{ aula de lógica.}}{\text{aula de lógica}}$$

Exemplo:

BD $= \{A \vee B,\ A \to C,\ B \to C\}$

Pergunta: C

BD $\vdash C$

1.	$A \vee B$		BD
2.	$A \to C$		BD
3.	$B \to C$		BD
4.	$[A]^1$		Suposição
	4.1	C	$\to$ E(4,2)
5.	$[B]^2$		Suposição
	5.1	C	$\to$ E(5,3)
6.	$C^{1,2}$		$\vee$ E(1,4.1,5.1)

Redução ao Absurdo:

A seguir apresentamos as duas últimas regras de Dedução Natural. A regra ABS introduz o absurdo a patir de uma contradição qualquer $\alpha \wedge \neg\alpha$. A regra de Rudução ao Absudo, RAA, é uma regra fundamental de Dedução Natural, sua intuição é que se supusermos a negação do que queremos provar e chegarmos a um absurdo então podemos concluir nossa prova dizendo sim para a pergunta.

¬-RAA

$$\frac{\begin{array}{c}[\neg\alpha] \\ \vdots \\ \bot\end{array}}{\alpha}$$

¬-ABS

$$\frac{\alpha \wedge \neg\alpha}{\bot}$$

Exemplo:

BD $= \{A \to C\}$

Pergunta: $\neg(A \wedge \neg C)$

BD $\vdash \neg(A \wedge \neg C)$

1.	$A \to C$		BD
2.	$[\neg\neg(A \wedge \neg C)]^1$		Suposição
	2.1	$(A \wedge \neg C)$	$\neg$ E(2)
	2.2	A	$\wedge$ E(2.1)
	2.3	$\neg C$	$\wedge$ E(2.1)
	2.4	C	$\to$ E(2.2,1)
	2.5	$C \wedge \neg C$	$\wedge$ I(2.3,2.4)
	2.6	$\perp$	ABS(2.5)
3.	$\neg(A \wedge \neg C)^1$		RAA I(2,2.6)

A seguir apresentamos todas as regras de dedução Natural.

Dedução Natural para a Lógica Proposicional Clássica

$\wedge$-I

$$\frac{\alpha \quad \beta}{\alpha \wedge \beta}$$

$\wedge$-E

$$\frac{\alpha \wedge \beta}{\alpha} \qquad \frac{\alpha \wedge \beta}{\beta}$$

$\vee$-I

$$\frac{\alpha}{\alpha \vee \beta} \qquad \frac{\beta}{\alpha \vee \beta}$$

$\vee$-E

$$\frac{\alpha \vee \beta \qquad \begin{array}{c}[\alpha]^i \\ \vdots \\ \gamma\end{array} \qquad \begin{array}{c}[\beta]^j \\ \vdots \\ \gamma\end{array}}{\gamma^{ij}}$$

$\rightarrow$-I

$$\frac{\begin{array}{c}[\alpha]^i \\ \vdots \\ \beta\end{array}}{\alpha \rightarrow \beta^i}$$

$\rightarrow$-E

$$\frac{\alpha \quad \alpha \rightarrow \beta}{\beta}$$

$\neg$-I

$$\frac{\alpha}{\neg\neg\alpha}$$

$\neg$-E

$$\frac{\neg\neg\alpha}{\alpha}$$

$\neg$-RAA

$$\frac{\begin{array}{c}[\neg\alpha]^i \\ \vdots \\ \bot\end{array}}{\alpha^i}$$

$\neg$-ABS

$$\frac{\alpha \wedge \neg\alpha}{\bot}$$

Definição:

1. Uma ***prova*** em dedução natural de uma formula φ a partir de um conjunto de fórmulas BD=$\{\varphi 1, \varphi 2, ..., \varphi$ k$\}$ é uma sequência de fórmulas rotuladas da seguinte forma:

i. as fórmulas do BD formam o prefixo da sequência e são rotuladas 1: $\varphi 1$, 2: $\varphi 2$, ..., k: φk;

ii. se a última fórmula da sequência é ρ.r: φi (onde ρ pode ser a sequência vazia), então a próxima fórmula rotulada será:

ii.1 ρ.r.1: φ j se φi é uma suposição;

ii.2 ρ.r+1: φ j se φ j foi obtida pela aplicação de regras do grupo I a fórmulas no escopo igual superior a σ, onde $\rho = \sigma$.r;

ii.3 σ.s+1: φ j se φ j foi obtida pela aplicação das regras do grupo II e $\rho = \sigma$.s;

iv. n: φ é a última fórmula da sequência.

2. A fórmula φ é dita um ***teorema do conjunto de fórmulas*** BD, BD$\vdash \varphi$.

3.A fórmula φ é dita um teorema lógico se BD é vazio.

OBS: Uma prova pode ser chamada algumas vezes de derivação.

Definição:

1. Um conjunto de fórmulas $\Gamma = \{\alpha 1, \alpha 2, ..., \alpha$ n$\}$ é **in*consistente*** se somente se $\Gamma \vdash \beta \wedge \neg\beta$ para alguma fórmula β.

2. Um conjunto de fórmulas $\Gamma = \{\alpha 1, \alpha 2, ..., \alpha$n$\}$ é ***consistente*** se somente se ele não é inconsistente.

Notação:

Nós abreviaremos:

$\Gamma \cup \alpha \vdash \beta$ por $\Gamma, \alpha \vdash \beta$

$\emptyset \vdash \alpha$ por $\vdash \alpha$

Exercícios:

Prove por dedução natural as seguintes afirmações:

1. BD $= \{A \to B \vee C, A \to \neg B, C \to \neg D\}$; BD $\vdash A \to \neg D$?
2. $\vdash (P \wedge Q) \to (Q \wedge P)$?
3. $Q \vdash P \to Q$?
4. $P \to Q \wedge R \vdash (P \to Q) \wedge (P \to R)$?
5. $P \to (Q \wedge R) \vdash (P \wedge Q) \to R$?
6. $\neg P \wedge \neg Q \vdash \neg(P \vee Q)$?
7. P $\vee Q \to R \vdash P \to (Q \to R)$?
8. $B, R \vee S \to A, R \vee S, A \wedge R \to C, B \wedge S \to C \vdash C$?
9. $(P \to Q) \wedge (Q \to R) \vdash P \to R$?
10. $P \to R, Q \to S \vdash P \wedge Q \to R \wedge S$?
11. $A \wedge B, (A \vee B) \to (R \to S), (P \to Q) \to R, A \wedge P \to Q \vdash S$?
12. $P \to Q, \neg Q \vdash \neg P$?
13. $(P \to Q) \to Q, Q \to P \vdash \neg P$?
14. $\vdash \neg(P \vee Q) \to \neg P \wedge \neg Q$?

Árvores de Prova

A forma mais usual de se representar provas em Dedução Natural é usando-se *árvores de prova*. Informalmente, uma árvore de prova é uma árvore finita na qual na raiz temos a fórmula que queremos provar α e nas folhas temos as suposições e fórmulas do banco Γ. Cada suposição tem um número, e se este número aparece na aplicação de uma regra de inferência significa que a suposição foi descarregada. Cada nó intermediário é obtido, como conclusão, pela aplicação de uma regra de inferência aos seus filhos, que são as premissas da regra[1].

Nós dizemos que uma fórmula α segue de um banco de dados (conjunto de fórmulas) Γ, $\Gamma \vdash \alpha$, se existe uma árvore de prova com α na raiz e todas as fórmulas de Γ são exatamente a únicas siposições que não foram descarregadas.

Exemplo 2.4.1 *:* $\vdash (A \to (B \wedge C)) \to (A \to B)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{[(A \to (B \wedge C))]^1 \quad [A]^2}{(B \wedge C)}}{B}}{(A \to B)^2}}{((A \to (B \wedge C)) \to (A \to B))^1}$$

Exemplo 2.4.2 $(A \to B) \vdash \neg(A \wedge \neg B)$.

$$\dfrac{\dfrac{\dfrac{\dfrac{[\neg\neg(A \wedge \neg B)]^3}{A \wedge \neg B}}{\neg B} \quad \dfrac{[A \to B]^2 \quad \dfrac{[\neg\neg(A \wedge \neg B)]^1}{A \wedge \neg B}}{A}}{B}}{\neg B \wedge B}}{\bot}}{\neg(A \wedge \neg B)^{1,3}}$$

Exercícios: Faça todos os exercícios do final da seção anterior colocando as provas em forma de árvores de prova.

[1] É importante observar que a árvore de prova é desenhada com a raiz embaixo e as folhas estão em cima. Isto não é usual em Computação

2.4.2 Método de Tableaux

As deduções são feitas por refutação, i.e., se queremos deduzir α a partir de um banco de fórmulas BD, BD $\vdash \alpha$, partimos da negação de α e tentamos chegar no absurdo. As deduções têm forma de árvore.

A seguir apresentamos todas as regras de de Tableaux.

Tableaux para a Lógica Proposicional Clássica

R_1

$$\frac{\alpha \wedge \beta}{\begin{array}{c}\alpha\\ \beta\end{array}}$$

R_2

$$\frac{\alpha \vee \beta}{\alpha \quad \beta}$$

R_3

$$\frac{\alpha \rightarrow \beta}{\neg\alpha \quad \beta}$$

R_4

$$\frac{\neg\neg\alpha}{\alpha}$$

R_5

$$\frac{\neg(\alpha \wedge \beta)}{\neg\alpha \quad \neg\beta}$$

R_6

$$\frac{\neg(\alpha \vee \beta)}{\begin{array}{c}\neg\alpha\\ \neg\beta\end{array}}$$

R_7

$$\frac{\neg(\alpha \rightarrow \beta)}{\begin{array}{c}\alpha\\ \neg\beta\end{array}}$$

Motivação

Se aplicarmos as regras a uma fórmula, vamos gerar uma árvore, onde cada ramo corresponde a uma ou mais valorações que satisfazem a fórmula, por isso é chamado Tableau Semântico.

Lembrando do nosso método semântico para verificar consequência lógica, i.e., dado um $BD = \{\phi_1, \cdots, \phi_n\}$ e uma pergunta φ, temos que $BD \models \varphi$ se e somente se $(\phi_1 \wedge \cdots \wedge \phi_n) \rightarrow \varphi$ é uma tautologia. Mas verificar se esta fórmula é uma tautologia é equivalente a verificar se sua negação é uma contradição. A intuição do método de Tableaux é aplicar as regras para mostrar que $\neg((\phi_1 \wedge \cdots \wedge \phi_n) \rightarrow \varphi)$ não possui nenhuma valoração que a faça verdadeira, i.e., ela é uma contradição. E portanto, $(\phi_1 \wedge \cdots \wedge \phi_n) \rightarrow \varphi$ é uma tautologia.

Definição: Um **ramo** θ de um tableaux τ é dito **fechado** se ele contiver α e $\neg\alpha$ para qualquer fórmula α.

Definição: Um **tableaux** τ é dito **fechado** se cada um dos seus ramos for fechado. E aberto caso contrário.

Método

1. O ramo inicial deve conter todas as fórmulas do BD seguidas da negação da pergunta;
2. aplique as regras as fórmulas no mesmo ramo no máximo uma vez;
3. se o tableaux fechar responda SIM;
4. se , em todos os ramos, todas as fórmulas já foram usadas uma vez e mesmo assim o tableaux não fechou responda NÃO.

Exemplo 1: $\{A \rightarrow B,\ B \rightarrow C\} \vdash A \rightarrow C$

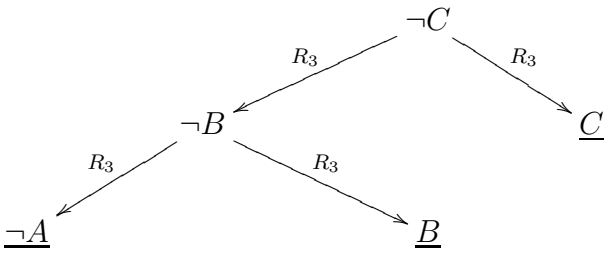
$BD \qquad A \to B$

$BD \qquad B \to C$

$Neg.\ perg. \qquad \neg(A \to C)$

$\downarrow R_7$

A



Exemplo 2: A $\vee B \vdash \neg(\neg A \wedge \neg B)$

$BD \qquad A \vee B$

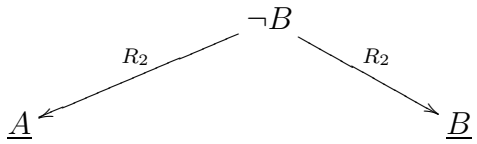
$Neg.\ perg. \qquad \neg\neg(\neg A \wedge \neg B)$

$\downarrow R_4$

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

$\downarrow R_1$

$\neg A$



Este tableaux é fechado, pois todas as valorações são contraditórias, logo $A \vee B$ e $\neg(\neg A \wedge \neg B)$ não é satisfatível.

Teorema (Completude): se BD $\models \alpha$ então existe tableaux fechado para BD, $\neg\alpha$.

Teorema (Correção): se existe um tableaux fechado para BD, $\neg\alpha$., então BD $\models \alpha$.

O método de Tableaux é refutacionalmente completo.

Exercícios:

1. $A \to B, \neg(A \vee B) \vdash \neg(C \to A)$
2. $(P \to Q), \neg(P \leftrightarrow Q) \vdash \neg P$
3. Guga é inteligente. Guga é determinado. Se Guga é determinado e atleta então ele não é um perdedor. Guga é atleta se ele é amante do tênis. Guga é amante do tênis se é inteligente. Guga não é um perdedor?
4. $(P \to (R \to Q)), (P \to R) \vdash (P \to Q)$
5. $\neg A \vee B, \neg(B \vee \neg C), C \to D \vdash \neg A \vee D$

2.4.3 Método de Resolução

Passo 1: Passar o BD para FNC e quebrar os comjunções em Cláusulas;

Passo 2: Negar a pergunta, passá-la para FNC e quebrar os comjunções em Cláusulas;

Passo 3: Juntas as cláusulas obtidas nos passos 1 e 2 e aplicar as regras até obter a cláusula vazia $\square$ (contradição).

Regras de Resolução

$$\frac{L_1, \cdots, L_i, \alpha, L_{i+1}, \cdots, L_n \quad M_1, \cdots M_j, \neg\alpha, M_{j+1}, \cdots, M_k}{L_1, \cdots L_n, M_1, \cdots, M_k}$$

$$\frac{L_1, \cdots, L_i, \alpha, L_{i+1}, \cdots, L_j, \alpha L_{j+1}, \cdots, L_n}{L_1, \cdots, L_i, \alpha, L_{i+1}, \cdots, L_j, L_{j+1}, \cdots, L_n}$$

Exemplo 1:

BD = $\{A \vee B,\ A \to C,\ B \to C\}$

Pergunta: C

BD $\vdash C$

Negando a pergunta e transformando a pergunta negada e o BD em cláusulas temos:

1. $A \vee B$ BD
2. $\neg A \vee C$ BD
3. $\neg B \vee C$ BD
4. $\neg C$ Negação da pergunta
5. $\neg B$ (3,4)
6. $\neg A$ (2,4)
7. A (5,1)
8. $\square$ (6,7)

Exemplo 2:

BD $= \{A \vee B,\ A \rightarrow B,\ B \rightarrow A\}$

Pergunta: $A \wedge B$

BD $\vdash A \wedge B$

Negando a pergunta e transformando a pergunta negada e o BD em cláusulas temos:

1. $A \vee B$ BD
2. $\neg A \vee B$ BD
3. $\neg B \vee A$ BD
4. $\neg A \vee \neg B$ Negação da pergunta
5. $\neg B \vee \neg B$ (3,4)
6. $\neg B$ (5)
7. A (6,1)
8. $\neg A$ (6,2)
9. $\square$ (7,8)

O método de Resolução é refutacionalmente correto e completo. Dado um banco de dados BD e uma pergunta ψ. Seja Ξ o conjunto de cláusulas resultante da

transformação do BD e de $\neg\psi$.

Correção: se $\Xi \vdash_{Res} \square$ então $BD \models \psi$

Completude: se $BD \models \psi$ então $\Xi \vdash_{Res} \square$

2.4.4 Provador de Dov Gabbay

Um provador automático de teoremas dirigido pelo objetivo. Isto é, parte da negação da pergunta.

- **Objetivo:** Dado um banco de dados e uma pergunta (objetivo) α, nós queremos responder SIM ou NãO para o caso da pergunta α seguir ou não do banco de dados.

- **2 Métodos até o momento:**
  - Tabela Verdade BD $\models \alpha$ (semanticamente);
  - Regras de Dedução Natural BD $\vdash \alpha$ (sintaticamente).

Tabela Verdade é mecânico mas **pouco eficiente**.

Dedução Natural requer **criatividade**.

- **Nosso objetivo:** Responder BD $\vdash \alpha$ automaticamente.

Provador Automático de Teoremas Dirijido pelo Objetivo (pergunta) :

- **Linguagem:**

Linguagem é a linguagem da lógica clássica proposicional com $\wedge$, $\rightarrow$ e o símbolo $\perp$ (absurdo). Nós escreveremos a $\neg\alpha \equiv \alpha \rightarrow \perp$.

Nós chamamos de cláusulas as formulas que podem ocorrer no banco de dados e objetivo as fórmulas que resultam da pergunta.

Definição: *Cláusula, Objetivo e Banco de Dados.*

(i) Qualquer átomo (símbolo proposicional) , incluindo $\perp$, é uma ***cláusula*** e também um ***objetivo.***

(ii) Se Θ é um objetivo e q um átomo, então $\Theta \rightarrow q$ é uma ***cláusula*** e também um ***objetivo.***

(iii) Se Θ_1 e Θ_2 são objetivos, então $\Theta_1 \wedge \Theta_2$ é também um ***objetivo.***

(vi) Um banco de dados BD é um conjunto de cláusulas.

(v) Nenhuma fórmula é uma cláusula a não ser as definidas em (i),(ii),(iii) e (vi).

Resumindo:

Cláusula	**Objetivo**
q ou $\perp$ (átomos)	q ou $\perp$ (átomos)
$\Theta \rightarrow q$	$\Theta \rightarrow q$
-	$\Theta_1 \wedge \Theta_2$

Transformação de fórmulas em Cláusulas e Objetivos:

1. $\neg\alpha \equiv (\alpha \rightarrow \perp)$
2. $\alpha \vee \beta \equiv (\alpha \rightarrow \perp) \rightarrow \beta$
3. $\alpha \rightarrow (\beta \rightarrow \sigma) \equiv (\alpha \wedge \beta) \rightarrow \sigma$
4. $\alpha \rightarrow (\beta \wedge \sigma) \equiv (\alpha \rightarrow \beta) \wedge (\alpha \rightarrow \sigma)$

5. Se na transformação do BD obtenho $\alpha_1 \wedge \alpha_2 \wedge ... \wedge \alpha_n$, nós colocamos $\alpha_1, \alpha_2, ..., \alpha_n$ no BD.

Exemplo:

$\neg A \vee B \rightarrow B \wedge C$

$(A \rightarrow \bot) \vee B \rightarrow B \wedge C$ (Regra 1)

$(A \rightarrow \bot) \rightarrow \bot) \rightarrow B \rightarrow B \wedge C$ (Regra 2)

$(((A \rightarrow \bot) \rightarrow \bot) \rightarrow B \rightarrow B) \wedge (((A \rightarrow \bot) \rightarrow \bot) \rightarrow B \rightarrow C)$ (Regra 4)

Toda fórmula da lógica proposicional é equivalente (pode ser traduzida) a uma conjunção de cláusulas.

Problema Original: $BD_0 \vdash \Theta_0$?

Problema Transformado: $BD \vdash \Theta$?

Regras de Computação:

1. Provar BD $\vdash \alpha \wedge \beta$, prove BD $\vdash \alpha$ e BD $\vdash \beta$.

2. Provar BD $\vdash \alpha \rightarrow \beta$, prove BD, $\alpha \vdash \beta$. Se α for uma conjunção $\alpha = \alpha_1 \wedge \alpha_2 \wedge ... \wedge \alpha_n$, adicione a BD α_1, α_2,..., α_n.

3. Provar BD $\vdash$ q, onde q é um átomo, temos 4 casos:

3.1. q $\in$ BD, responda SIM.

3.2.$\bot \in$ BD, responda SIM.

3.3.$\sigma \rightarrow$ q $\in$ BD, pergunte BD $\vdash \sigma$.

3.4.$\sigma \rightarrow \bot \in$ BD, pergunte BD $\vdash \sigma$.

4. Regra do Reinício (*restart*)

Nosso problema inicial era mostrar $BD \vdash \Theta$. No meio da computação, nós estamos perguntando $BD' \vdash \alpha$. Se não conseguimos prosseguir deste ponto, podemos perguntar qualquer pergunta já feita no **mesmo ramo**, por exemplo β, ao banco de dados atual, isto é, $BD' \vdash \beta$ e continuar a prova.

5. Checagem de LOOP: Nunca pergunte $BD \vdash \alpha$ pela 2ª vez para ao mesmo BD e mesmo α.

Exemplos:

(1) $(A \rightarrow B, B \rightarrow C), A \vdash C$ $(A \rightarrow B), (B \rightarrow C), A \vdash B$ Regra (3.3) $(A \rightarrow B), (B \rightarrow C), A \vdash A$ Regra (3.3) SIM ! Regra (3.1)

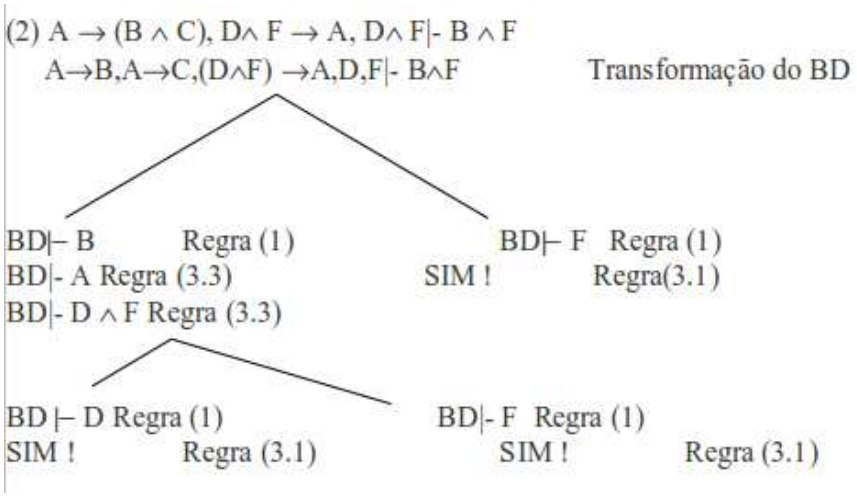


Figura 2.1:

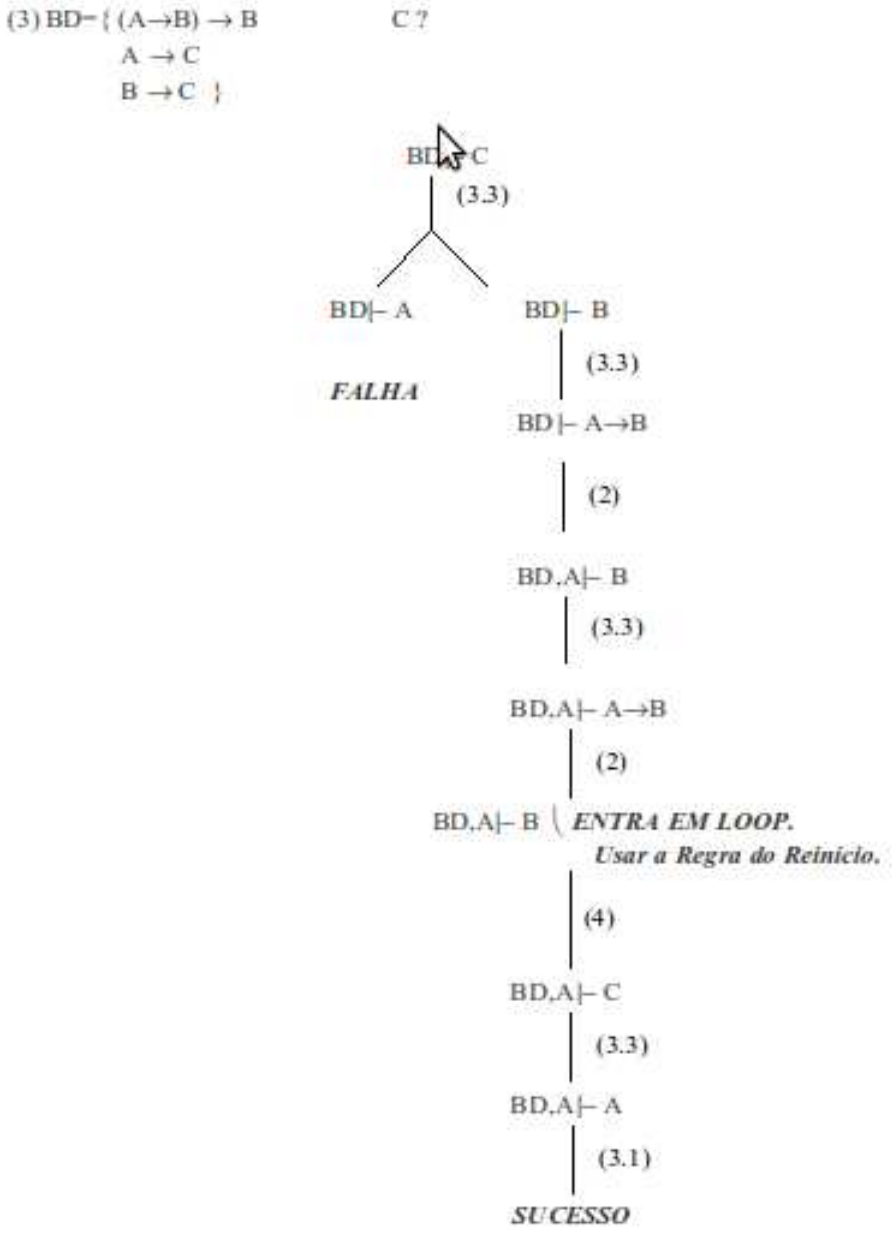

(3) BD⊢ { (A→B) → B
A → C
B → C }
C ?
BD⊢ C
(3.3)
BD⊢ A
FALHA
BD⊢ B
(3.3)
BD⊢ A→B
(2)
BD,A⊢ B
(3.3)
BD,A⊢ A→B
(2)
BD,A⊢ B
ENTRA EM LOOP.
Usar a Regra do Reinicio.
(4)
BD,A⊢ C
(3.3)
BD,A⊢ A
(3.1)
SUCESSO

Figura 2.2:

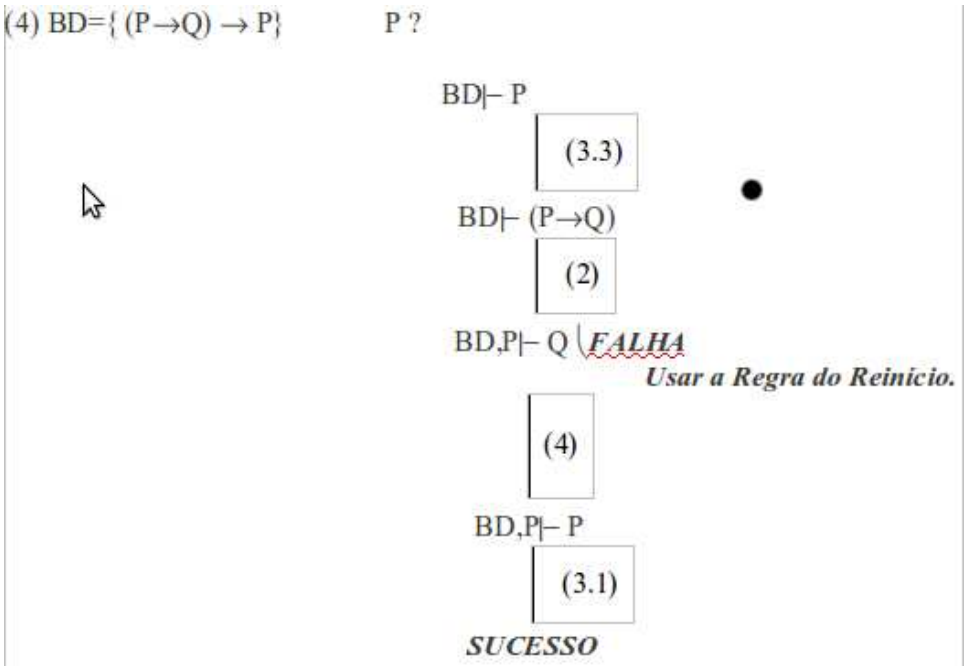


Figura 2.3:

Execícios:

(1) $(A \to B) \to B \vdash (B \to A) \to A$

(2) $(A \to B) \to B \vdash A \vee B$

(3) $(\neg A \to A) \vdash A$

(4) $(A \to B) \vdash (C \vee A) \to (C \vee B)$

(5) $A \vdash A \vee B$

(6) $A \to B, \neg B \vdash \neg A$

(7) $\vdash A \vee (A \to B)$

(8) $(A \vee B) \vee C \vdash A \vee (B \vee C)$

2.4.5 Sistema Axiomático

- Outro sistema dedutivo.

- Mais antigo e mais utilizado para fins teóricos.
- Vários axiomas e uma única regra de inferência.

Sejam α, β e γ fórmulas quaisquer da linguagem proposicional.

Axiomas Lógicos:

Implicação:

(1) $\alpha \to (\beta \to \alpha)$

(2)$(\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to \alpha \to \gamma))$

Conjunção:

(3) $(\alpha \wedge \beta) \to \alpha$

(4) $(\alpha \wedge \beta) \to \beta$

(5) $\alpha \to (\beta \to (\alpha \wedge \beta))$

Disjunção:

(6) $\alpha \to \alpha \vee \beta$

(7) $\beta \to \alpha \vee \beta$

(8) $((\alpha \to \gamma) \wedge \beta \to \gamma)) \to ((\alpha \vee \beta) \to \gamma)$

Negação:

(9) $\alpha \to \neg\neg\alpha$

(10) $\neg\neg\alpha \to \alpha$

(11) $(\alpha \to \beta) \to ((\alpha \to \neg\beta) \to \neg\alpha)$

Regra de Inferência:

Modus Pones (M.P.)

$$\frac{\alpha \quad \alpha \to \beta}{\beta}$$

- Nosso cálculo dedutivo possui um conjunto infinito de axiomas lógicos. Para cada fórmula α, β e γ, nós temos axiomas diferentes.
- (1),...,(11) são chamadas de axiomas esquema.
- A única regra é a Modus Pones (M.P.).

Definição:

Uma fórmula α é dita um ***teorema*** de um conjunto de fórmulas $\Gamma(\Gamma \vdash \alpha)$ se e somente se existe uma seqüência de fórmulas $\alpha_1, ..., \alpha_n$ tal que $\alpha_n = \alpha$ e cada α_i é:

(i) uma instância de um axioma esquema;

(ii) ou for obtida por M.P. aplicada a α_l e α_k e $l, k < i$.

(iii) ou um membro de Γ.

A seqüência de fórmulas $\alpha_1, ..., \alpha_n$ é chamada de uma prova de α a partir de Γ.

Exemplos:

(1) $\Gamma = \{A \wedge B, A \to C\} \vdash C \vee D$?

1. $A \wedge B \to A$ axioma 3
2. $A \wedge B$ Γ
3. A M.P.(1,2)
4. $A \to C$ Γ
5. C M.P.(3,4)
6. $C \to (C \vee D)$ axioma 6

7. $C \vee D$ M.P.(5,6)

$(2) \vdash A \to A$

1. $A \to ((A \to A) \to A)$ axioma 1
2. $(A \to ((A \to A) \to A)) \to ((A \to (A \to A)) \to (A \to A))$ axioma 2
3. $(A \to (A \to A)) \to (A \to A)$ M.P.(1,2)
4. $A \to (A \to A)$ axioma 1
5. $A \to A$ M.P.(4,3)

Exercícios:

Provar usando o Método Axiomático:

1) $A \to B, B \to C \vdash A \to C$

2) $(A \vee B) \to C \vdash A \to (B \vee C)$

3) $A \to (B \vee C) \vdash (A \to B) \vee (A \to C)$

Observação: É importante notar (e possível provar) que os todos métodos dedutivos estudados para a lógica clássica proposicional são equivalentes, ou seja, uma fórmula que pode ser provada utilizando um deles, sempre poderá ser provada utilizando qualquer dos outros. Isso é importante, na medida em que nos permite provar uma determinada propriedade dos sistemas dedutivos em geral, provando-a apenas para o método axiomático, que embora difícil de ser usado na prática para provar um teorema, é bastante simples no que diz respeito à sua construção, o que facilita a demonstração de propriedades teóricas, como a completude e a corretude, que veremos na próxima seção.

2.4.6 Relações entre Sintaxe e Semântica

Uma das aspectos mais importantes da lógica proposicional é a maneira como a sintaxe se relaciona com a semântica.

SEMÂNTICA	**SINTAXE**
Valor verdade	prova/dedução
valida	teorema
implica logicamente $\Gamma \models \alpha$	$\Gamma \vdash \alpha$
tabela verdade	cálculo dedutivo

Nós queremos relacionar o fato de uma fórmula α ser um teorema de um conjunto de fórmulas Γ ($\Gamma \vdash \alpha$) com a propriedade de Γ implicar logicamente em α ($\Gamma \models \alpha$).

Teorema da Corretude

"Tudo que o cálculo dedutivo prova é semanticamente válido."

Se $\Gamma \vdash \alpha$ **então** $\Gamma \models \alpha$

Se uma fórmula é provada a partir de um conjunto de fórmulas então ela é logicamente implicada por este conjunto de fórmulas.

Este teorema nos assegura que tudo que provamos no sistema dedutivo é ***correto*** em relação à semântica. Isto é, nosso sistema dedutivo só prova teoremas que semanticamente estão ***corretos***.

Como se prova:

1) Prova-se que os axiomas do cálculo dedutivo são semânticamente válidos, isto é, são tautologias;

2) Prova-se que as regras de inferência sempre derivam conclusões verdadeiras a partir de premissas verdadeiras.

Teorema da Completude

"Tudo que é semânticamente válido é provado pelo cálculo dedutivo."

Se $\Gamma \models \alpha$ **então** $\Gamma \vdash \alpha$

Se Γ implica logicamente em α então existe uma prova de α a partir de Γ no sistema dedutivo.

O sistema dedutivo é ***completo*** em relação à semântica pois para toda fórmula α que é logicamente implicada por Γ existe uma prova α a partir de Γ no sistema dedutivo.

Tudo que é semanticamente obtido pode ser também obtido no sistema dedutivo.

$\Gamma \models \alpha \therefore \alpha$ é conseqüência lógica de Γ

$\Rightarrow$ Toda atribuição de valores verdade que satisfaz Γ também satisfaz α.

Sendo $\Gamma = \{ \alpha_1, ..., \alpha_n \}$

Quero provar que se $\models (\alpha_1 \wedge ... \wedge \alpha_n) \rightarrow \alpha$ então $\vdash (\alpha_1 \wedge ... \wedge \alpha_n) \rightarrow \alpha$

A maneira mais usual de se provar Completude é usando-se a técnica de modelo canônico:

Modelo Canônico

A técnica do modelo canônico se baseia numa propriedade da Lógica Proposicional que provar Completude é equivalente a provar que qualquer conjunto consistente é satisfatível. Enunciaremos e provaremos este fato a seguir:

Se $\Gamma \models \alpha$ **então** $\Gamma \vdash \alpha$

$\Updownarrow$

Se $\Gamma \cup \{\alpha\}$ é consistente **então** $\Gamma \cup \{\alpha\}$ é satisfatível

Seja $\Gamma' = \Gamma \cup \alpha$.

1. Suponha que se $\Gamma \models \alpha$ então $\Gamma \vdash \alpha$.

2. Suponha que Γ' é consistente.

3. Suponha, por contradição, que Γ' é insatisfatível.

4. Se Γ' é insatisfatível, então, por definição, não existe nenhuma atribuição de valores verdade que satisfaça todos os membros de Γ'. Sendo assim, podemos dizer que $\Gamma' \models \beta$ e $\Gamma' \models \neg\beta$, para uma fórmula qualquer β.

5. Pela suposição em (1), temos que $\Gamma' \vdash \beta$ e $\Gamma' \vdash \neg\beta$;

6. A partir de (5) podemos concluir que $\Gamma' \vdash \beta \wedge \neg\beta$

7. Ora, (6) contradiz o fato que Γ' é consistente.

Assim, por contradição, podemos afirmar que:

Se $\Gamma \models \alpha$ **então** $\Gamma \vdash \alpha$

$$\Downarrow$$

Se $\Gamma \cup \{\alpha\}$ é consistente **então** $\Gamma \cup \{\alpha\}$ é satisfatível

Vamos agora provar a implicação contrária:

1. Suponha que se Γ é consistente então Γ é satisfatível.

2. Suponha $\Gamma \models \alpha$

3. Suponha, por contradição, que $\Gamma \nvdash \alpha$

4. Então $\Gamma \cup \{ \neg\alpha \}$ é consistente, já que $\Gamma \nvdash \alpha$ e portanto não poderá ocorrer que $\Gamma \vdash \alpha \wedge \neg\alpha$

5. Então, pela suposição (1), $\Gamma \cup \{ \neg\alpha\}$ é satisfatível.

6. Logo, existe uma valoração v que satisfaz $\Gamma \cup \{\neg\alpha\}$.

7. Mas isto é uma contradição, pois por (2) v satisfaz α também.

Assim, estabelecemos a volta:

$$\textbf{Se } \Gamma \models \alpha \textbf{ então } \Gamma \vdash \alpha$$

$$\Uparrow$$

$$\textbf{Se } \Gamma \cup \{\alpha\} \text{ é consistente } \textbf{então } \Gamma \cup \{\alpha\} \text{ é satisfatível}$$

Observações:

- Um conjunto de fórmulas Γ é consistente se e somente se $\neg(\Gamma \vdash \alpha \wedge \neg\alpha)$
- Uma valoração s é um modelo para $\Gamma = \{\alpha_1, ..., \alpha_n\}$ se e somente se $s(\alpha_i) = V$ para todo $\alpha_i \in \Gamma$.
- Pelo modelo canônico, para provar a completude basta mostrar que todo conjunto consistente de fórmulas é satisfatível.

Prova do Teorema da Completude:

Dado um conjunto de fórmulas consistente Γ, nós precisamos construir uma valoração s' e mostrar que s' satisfaz Γ.

1o passo: Estender o conjunto consistente Γ para um conjunto Δ satisfazendo as seguintes condições:

a) $\Gamma \subseteq \Delta$

b) Δ é maximal e consistente, isto é, para toda fórmula α da linguagem, ou $\alpha \in \Delta$ ou $\neg\alpha \in \Delta$.

2o passo: Seja L a linguagem proposicional e Φ o conjunto dos símbolos proposicionais.

Vamos construir uma valoração que satisfaz Γ a partir de Δ.

s : $\Phi \rightarrow$ {V,F} para todo símbolo proposicional A $\in \Phi$.

s(A) = V se A $\in \Delta$

s(A) = F se A $\notin \Delta$

Nós podemos estender s para um s' que valorize fórmulas,

s': L $\rightarrow$ {V,F}.(Como definido em aulas anteriores)

Lema da Verdade

Seja Γ um conjunto de fórmulas e α uma fórmula. s'(α) = V $\Leftrightarrow \alpha \in \Delta$

Prova do Lema da Verdade:

Por indução, sobre o comprimento da fórmula, isto é, no número de símbolos lógicos nela contidos ($\neg, \vee$, etc).

a) Caso base:

$|\alpha|$=0 (Fórmula Atômica)

$\alpha \in \Phi, \alpha = A$

s'(A) = s(A) = V $\Leftrightarrow$ A $\in \Delta$ (pela definição de s)

b) Hipótese de Indução: o lema vale para fórmula de tamanho $|\alpha| \leq$ n.

c) Queremos mostrar que vale para $|\alpha|$ = n+1

Considere $|\alpha|$ = n+1

Temos então vários casos:

i) $\alpha = \neg\beta$

s'($\neg\beta$) = V sse s'(β) = F (definição de s')

sse $\beta \notin \Delta$ (pela hipótese de indução)

Logo, $\neg\beta \in \Delta$ (pois Δ é maximal)

Logo α é V sse $\alpha \in \Delta$.

ii) $\alpha = \beta \rightarrow \gamma$

s'$(\beta \rightarrow \gamma) = V$ sse s'(β)= F $\vee$ s'(γ)= V

sse $\beta \notin \Delta \vee \gamma \in \Delta$

sse $\neg\beta \in \Delta \vee \gamma \in \Delta$

sse $\Delta \vdash \neg\beta \vee \gamma$

sse $\Delta \vdash \beta \rightarrow \gamma$, pois $\Delta \vdash (\neg\beta \vee \gamma) \leftrightarrow (\beta \rightarrow \gamma)$

sse $(\beta \rightarrow \gamma) \in \Delta$, pois Δ é consistente.

Observação: os casos iii e iv são similares e ficam como exercício.

iii) $\alpha = \beta \wedge \gamma$

iv) $\alpha = \beta \vee \gamma$

Vamos agora, a partir do lema demonstrado, provar o Teorema da Completude:

1. Suponha Γ é consistente.

2. Estenda Γ para Δ maximal e consistente.

3. Construir s e s'

4. Seja $\Gamma = \{\alpha_1, ..., \alpha_n\}$. Como $\Gamma \subseteq \Delta, \alpha_i \in \Delta \Leftrightarrow$ s'(α_i) = V, para todo i (pelo "lema da verdade").

5. Logo, s' satisfaz Γ e portanto Γ é satisfatível.

2.5 Aplicação

Suponha que um banco deseja fazer um programa escrito na linguagem da LCP para decidir o perfil de um dado cliente e decidr qual a plicação que é mais apropirida para ele. O banco classifica o perfil do cliente como: conservador, moderado e arrojado. As aplicações são: poupança, ações e mista (metada na poupança e metade em ações). Um cliente conservador deve aplicar em poupança. Um cliente moderado deve dividir a aplicação entre poupança e ações. E um cliente arrojado deve aplicar tudo em ações. Porém, dependendo da renda mensal ele pode ser aconselhado a aplicar fora do seu perfil. O perfil do cliente é identificado fazendo com que ele responda ao seguinte formulário:

1. Sua idade é maior que 50 anos? ( ) Sim ( ) Não
2. Sua idade é menor que 30 anos? ( ) Sim ( ) Não
3. Casado? ( ) Sim ( ) Não
4. Filhos? ( ) Sim ( ) Não
5. Renda mensal maior que R$ 10.000,00?

Representação:

$I_{>50}$ - idade é maior que 50 anos

$I_{<30}$ - idade é menor que 30 anos

C - casado

F - filhos

$R_{>10K}$ - Renda mensal maior que R$ 10.000,00

Co - perfil conservador

Mo - perfil moderado

Ar - perfil arrojado

$Poup$ - aplicar em poupança

Ac - aplicar em ações

Mx - aplicar em poupança e ações

Regras:

1. $(I_{>50} \vee \neg(I_{<30}) \wedge C \wedge F \rightarrow Co$
2. $(I_{<30} \vee \neg I_{>50}) \wedge \neg C \wedge \neg F \rightarrow Ar$
3. $\neg(I_{<30} \wedge I_{>50}) \rightarrow Mo$
4. $I_{<30} \wedge C \wedge F \rightarrow Mo$
5. $I_{>50} \wedge \neg C \wedge \neg F \rightarrow Mo$
6. $Co \rightarrow Poup$
7. $Co \wedge R_{>10K} \rightarrow Mx$
8. $Ar \rightarrow Ac$
9. $Ar \wedge \neg R_{>10K} \rightarrow Mx$
10. $Mo \rightarrow Mx$
11. $Mo \wedge (R_{>10K} lor \neg F) \rightarrow Ac$
12. $Mo \wedge (\neg R_{>10K} landF) \rightarrow Poup$

Consultas: Dado que um cliente respondeu o questionário da seguinte forma:

- $I_{>50}$
- $\neg I_{<30}$
- C

- F
- $\neg R_{>10K}$ - Renda mensal maior que R$ 10.000,0

Consulta:

1. $\vdash Poup$, i.e., ele deve aplicar em poupança

1. Prove as consultas em Ded. Nat.
2. Prove as consultas em Tableaux
3. Prove as consultas em Resolução

Observações:

1. Será que falta alguma regra? Tem algum caso que não está sendo tratado? Qual?
2. Suponha que você deseja guardar todas as respostas de todos os clientes e gostaria de expressar a seguinte regra: para todo cliente X, X não pode ser conservador e moderado ao mesmo tempo. Como você expressaria esta fraze na linguagem da LCP?

Capítulo 3

Logica Clássica de Primeira Ordem

$\forall x, \text{EmpUFRJ}(x) \rightarrow \text{FuncPub}(x)$

$\forall x \exists y, \text{Suc}(y, x)$

Lógica Clássica Proposicional

+

Variáveis

+

Cosntantes

+

Funções

+

Tabelas (Predicados)

- Formaliza o raciocínio dedutivo
- Lógica Proposicional é um modelo muito restrito:

  Não podemos descrever propriedades sobre os elementos do universo de discurso: todos os elementos do domínio têm propriedade P; existem elementos do domínio que têm a propriedade P; não podemos representar relações e funções. Ex: Teoria dos Números < 0, suc, $<$, $+$, . , -,... $>$

3.1 Linguagem da Lógica Clássica de Primeira Ordem

Linguagem: alfabeto + regras gramaticais

Definição 1 *Um alfabeto de 1a ordem consiste dos seguintes conjuntos de símbolos:*

Símbolos Lógicos:

1. ***Conectivos lógicos:*** $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\neg$, $\forall$, $\exists$.
2. ***Símbolos auxiliares:*** *(e).*
3. ***Conjunto enumerável de variáveis:*** $V = \{v1, v2, ...\}$

   Símbolos não Lógicos:
4. ***Conjunto enumerável de constantes:*** $C = \{c1, c2...\}$
5. ***Conjunto enumerável de símbolos de função:*** $F = \{f1, f2, ...\}$ *A cada símbolo funcional está associado um número inteiro* $n > 0$*, chamado de aridade.*
6. ***Conjunto enumerável de símbolos predicativos (Predicados):*** $P = \{P1, P2, ...\}$

A cada símbolo predicativo está associado um número inteiro $n > 0$*, chamado aridade.*

- ***Variáveis:*** *representam elementos quaisquer do domínio.*
- ***Constantes:*** *dão nome a elementos particulares do domínio.*
- ***Funções:*** *representam operações sobre elementos do domínio.*
- ***Predicados:*** *representam propriedades ou relações entre elementos do domínio.*

$\forall$ ***Quantificador universal:*** *"para todo elemento do domínio".*

$\exists$ ***Quantificador existencial:*** *"existe ao menos um indivíduo".*

Exemplo: $\forall x(\exists y \text{ ANCESTRAL}(y, x) \wedge \text{ANCESTRAL(João,José)})$

Definição 1 *Os* ***termos*** *da linguagem de 1a ordem são definidos recursivamente como:*

(i) *toda variável e constante é um termo;*

(ii) *se* $t1, t2, ..., tn$ *são termos e* f *um símbolo funcional de aridade* n, $f(t1, t2, ..., tn)$ *é um termo;*

(iii) *nada mais é termo.*

Definição 1 *As* ***fórmulas*** *da lógica de 1a ordem são definidas recursivamente como:*

(i) *Se* P *é um predicado de aridade* n *e* $t1, t2, ..., tn$ *são termos, então* $P(t1, t2, ..., tn)$ *é uma fórmula chamada* ***fórmula atômica****;*

(ii) *Se* α *e* β *são fórmulas, então* $(\neg\alpha), (\alpha \wedge \beta), (\alpha \vee \beta), (\alpha \rightarrow \beta)$ *também são fórmulas;*

(iii) *Se alpha é uma fórmula e* x *uma variável, então* $\forall x\ \alpha$ *e* $\exists x\ \alpha$ *também são fórmulas;*

(iv) *Nada mais é fórmula*

De uma forma alternativa podemos definir a linguagem de primeira ordem por meio de uma notação BNF.

Termos:

$t ::= x \mid c \mid f(t_1, ..., t_n)$

Fórmulas:

$\alpha ::= P(t_1, ..., t_n) \mid (\alpha_1 \wedge \alpha_2) \mid (\alpha_1 \vee \alpha_2) \mid (\alpha_1 \rightarrow \alpha_2) \mid \neg\alpha \mid \forall x\alpha(x) \mid \exists x\alpha(x)$
onde P é um símbolo predicativo n-ário e $t_1, ..., t_n$ são termos.

Observações:

1. $\alpha \leftrightarrow \beta \equiv (\alpha \rightarrow \beta) \wedge (\beta \rightarrow \alpha)$
2. Convenções:
   - **(i)** x, y, z, ... Variáveis;
   - **(ii)** a, b, c, ... Constantes;
   - **(iii)** f, g, h, ... Funções;
   - **(iv)** A, B, C, P, U, ... Predicados;
3. $\neg\exists x$ GOSTA(x, collor) $\equiv \forall x\neg$GOSTA(x,collor)

Exercícios: Nos exercícios seguintes, represente cada proposição na linguagem da lógica de primeira ordem, especificando em cada caso o significado dos símbolos não lógicos utilizados.

1. Todo cachorro é um animal. Todo animal morre. Rex é um cachorro.

2. Qualquer pessoa passando nos exames de história e ganhando na loteria é feliz. Porém qualquer pessoa que estuda ou tem sorte pode passar em todos os exames. João não estudou, mas ele tem sorte. Qualquer pessoa que tem sorte ganha na loteria. João é feliz?

3. Toda pessoa que não é pobre e é esperta é feliz. Pessoas que lêem não são burras. João sabe ler e é rico. Pessoas felizes têm vidas excitantes. Existe alguém que tem vida excitante?

4. Ninguém conquistou o mundo. Portanto, todo mundo é livre.

5. Todo elétron tem carga negativa. Nenhum pósitron tem carga negativa. Portanto, nenhum pósitron é um elétron.

6. Se Jane está doente, ela não virá trabalhar. Se ela não vier trabalhar, nenhum de nós terá nada para fazer. Assim, se Jane está doente, nenhum de nós terá nada para fazer.

Exemplo: Conselheiro Financeiro:

1. Função: ajuda a decidir se devemos investir em investimentos tipo poupança ou no mercado de ações.

2. O investimento recomendado depende do ganho mensal da pessoa e quanto ela tem em poupança.

3. Pessoas com valor inadequado de poupança devem sempre aumentar o valor da poupança como 1a prioridade, independente do seu ganho.

4. Pessoas com valor adequado de poupança e um ganho adequado devem considerar um investimento mais arriscado, porém mais lucrativo no mercado de ações.

5. Pessoas com ganho inadequado e com poupança adequada podem considerar em dividir o valor a ser investido entre poupança e mercado de ações. Isto aumenta a poupança e tenta aumentar o ganho.

6. Poupança adequada se valor poupado maior do que 5.000 x no de dependentes.

7. Ganho adequado se valor ganho maior do que 15.000 + (4.000 x no de dependentes) e é estável. Um ganho instável, mesmo que grande, é sempre inadequado.

- TOTALPOUP(x): x é total na poupança.
- DEPEND(y): número de dependente
- minpoup(y): função que retorna o valor da poupança mínima (5.000) vezes o número de dependentes
- TOTALGANHO(x, estável): ganho total é x e este é estável.
- minganho(y): função que retorna o valor do ganho mínimo mais o acréscimo por dependente.
- MAIOR(x,y): x > y
- POUP(status): status é uma variável que indica a situação (adequada/inadequada) da poupança.
- GANHO(status): aqui, status indica a situação (adequada/inadequada) do ganho.
- INVEST(tipo): tipo é uma variável que guarda o tipo de investimento recomendado (poupança/combinado/ações)

Especificação:

1. POUP(inadequado) *to* INVEST(poupança)
2. POUP(adequado) $\wedge$ GANHO(inadequado) $\rightarrow$ INVEST(combinado)
3. POUP(adequado) $\wedge$ GANHO(adequado) $\rightarrow$ INVEST(ações)
4. $\forall x$ TOTALPOUP(x) $\wedge \exists y$ (DEPEND(y) $\wedge$ MAIOR(x, minpoup(y))) $\rightarrow$ POUP(adequado) onde minpoup(x) = 5.000x

5. $\forall x$ TOTALPOUP(x) $\wedge$ $\exists y$ (DEPEND(y) $\wedge$ $\neg$ MAIOR(x, minpoup(y)) $\rightarrow$ POUP(inadequado)

6. $\forall x$ TOTALGANHO(x, estável) $\wedge$ $\exists y$ (DEPEND(y) $\wedge$ MAIOR(x, minganho(y))) *to* GANHO(adequado)

7. $\forall x$ TOTALGANHO(x, estável) $\wedge$ $\exists y$ (DEPEND(y) $\wedge$ $\neg$MAIOR(x, minganho(y))) $\rightarrow$ GANHO(inadequado)

8. $\forall x$ TOTALGANHO(x, instável) $\rightarrow$ GANHO(inadequado)

   Entrada:

9. TOTALPOUP(22.000)

10. TOTALGANHO(25.000, estável)

11. DEPEND(3)

Exercício: Qual a saída produzida pela especificação acima para a entrada dada?

3.2 Sistemas Dedutivos

Sistemas dedutivos para lógica de primeira ordem.

Definição 1 *Dizemos que uma variável x ocorre livre em uma fórmula α se somente se:*

(i) *α é uma fórmula atômica e x ocorre em α;*

(ii) *α é uma fórmula da forma $\beta \wedge \gamma$, $\beta \vee \gamma$, $\beta \rightarrow \gamma$ e x ocorre livre em β ou γ;*

(iii) *α é uma fórmula da forma $\neg\beta$ e x ocorre livre em β;*

(iv) *α é uma fórmula da forma $\forall y\beta$ ou $\exists y\beta$ e x ocorre livre em β e $x \neq y$.*

Exemplos: x ocorre livre?

1. P(x,y) SIM
2. $\forall$y(P(x,y) $\wedge$ Q(y,x) $\rightarrow$ R(y)) SIM
3. $\forall$y($\forall$x(P(x) $\rightarrow$ Q(y)) $\rightarrow$ R(x)) SIM
4. $\forall$y $\forall$z (($\forall$xP(x,y) $\rightarrow$ Q(z)) $\wedge$ (Q(x) $\rightarrow$ R(x,y))) SIM
5. P(z,y) NÃO
6. $\forall$y $\exists$x (P(x,y) $\rightarrow$ Q(y)) NÃO

Definição 1 *Uma fórmula α é uma sentença (ou uma fórmula fechada) se somente se α não tem nenhuma variável ocorrendo livre.*

Definição 1 *Seja α uma fórmula, x uma variável e t um termo. Pela substituição de x por t em $\alpha(\alpha(x/t))$ entendemos a expressão resultante da troca de todas as ocorrências livres de x por t.*

Exemplos:

1. $\forall$ y(P(x, y,f(x, y))) $\rightarrow$ Q(g(x), h(g(x)) x/h(a)

   $\forall$ y(P(h(a), y, f(h(a), y)) $\rightarrow$ Q(g(h(a), h(g(h(a))))

2. $\forall$ y($\forall$ x(Q(x, y, g(z)) $\rightarrow$ P(f(x), y))) $\rightarrow$ R(y(g(x))) x/f(z)

   $\forall$ y($\forall$ x(Q(x, y, g(z)) $\rightarrow$ P(f(x), y))) $\rightarrow$ R(y(g(f(z))))

3. [$\forall$y(P(x, y, f(x,y))) $\rightarrow$ Q(y,z) x/g(z)] z/a

   $\forall$y (P(g(z), y, f(g(z), y)) $\rightarrow$ Q(y, z) z/a

   $\forall$y (P(g(a), y, f(g(a), y)) $\rightarrow$ Q(y, a)

Definição:

Denotaremos por:

α (x_1 , x_2 , ..., x_n) (x_1 /t_1 , x_2 /t_2 , ..., x_n /t_n)

a substituição simultânea (paralelo) de todas as ocorrências livres de x_1 , ..., x_n por t_1 , ..., t_n respectivamente.

OBS: α (x_1 /t_1) ... (x_n /t_n) é em série da esquerda para direita.

α (x_1 /t_1 , x_2 /t_2 , ..., x_n /t_n) é em paralelo.

Exemplos:

1. P(x, f(y), g(x,y)) (x/a, y/b)

   P(a, f(b), g(a,b))

2. $(\forall y \forall x P(x, y)) \rightarrow Q(g(x)) \wedge R(h(y))$ $(x/f(a), y/z)$

   $(\forall y \forall x P(x, y)) \rightarrow Q(g(f(a))) \wedge R(h(z))$

3. P(x,y,f(x,z)) (x/g(z), z/y, y/a)

   P(g(z), a, f(g(z),y)

4. P(x,y,f(x,z)) (x/g(z)) (z/y) (y/a)

   P(g(z),y,f(g(z),z)) (z/y) (y/a)

   P(g(y),y,f(g(y),y)) (y/a)

   P(g(a),a,f(g(a),a))

Definição:

Uma variável x é substituível em uma fórmula α por um termo t se, para cada variável y ocorrendo em t, não existe nenhuma subfórmula de α da forma $\forall$ yβ ou $\exists$ yβ onde x ocorre livre em β.

O que queremos evitar com esta condição é que o quantificador $\forall$ y ou $\exists$ y capture alguma variável de t.

Exemplo:

($\forall$ y CHEFE(x,y) $\rightarrow$ GERENTE(x)) x/y

($\forall$ y CHEFE(y,y) $\rightarrow$ GERENTE(y))

3.3 Dedução Natural

- Regras para $\wedge$, $\vee$, $\rightarrow$, $\neg$, ABS e RA são as mesmas do caso proposicional.
- ***Regras do Quantificador Universal*** **:**

$\forall$-I

$$\frac{\alpha(a)}{\forall x \alpha(a/x)}$$

Condição: Condição:a não ocorre em nenhuma fórmula do BD e nem em nenhuma suposição em aberto.

$\forall$-E

$$\frac{\forall x \alpha}{\alpha(x/t)}$$

Condição: x é substituível em α por **t**

Regras para o Quantificador Existencial:

$\exists$-I

$$\frac{\alpha(a)}{\exists x \alpha(a/x)}$$

$\exists$-E

Condição: a não ocorrem em θ, nem no BD e nem em qualquer suposição na qual θ depende, a não ser $\alpha(x/a)$.

$$\dfrac{\exists x\alpha(x) \qquad \begin{array}{c}[\alpha(a)]^i \\ \vdots \\ \theta\end{array}}{\theta^i}$$

Exemplo 1: $\vdash \forall x\forall y P(x,y) \rightarrow \forall y\forall x P(x,y)$

1.	$[\forall x\forall y P(x,y)]^1$	Suposição
	1.1 $\forall y P(a,y)$	$\forall$-E (1)
	1.2 $P(a,b)$	$\forall$-E(1.1)
	1.3 $\forall x P(x,b)$	$\forall$-I (1.2)
	1.4 $\forall y\forall x P(x,y)$	$\forall$-I (1.3)
2.	$\forall x\forall y P(x,y) \rightarrow \forall y\forall x P(x,y)^1$	$\rightarrow$ E(1,1.4)

Exemplo 2: $\forall x(Q(y) \rightarrow P(x)) \rightarrow (Q(y) \rightarrow \forall x P(x))$

1.	$[\forall x(Q(y) \rightarrow P(x))]^1$		Suposição
	1.1 $Q(y) \rightarrow P(a)$		$\forall$-E (1)
	1.2 $[Q(y)]^2$		Suposição
		1.2.1 $P(a)$	$\rightarrow$-E (1.1,1.2)
		1.2.2 $\forall x P(x)$	$\forall$-I (1.2.1)
	1.3 $(Q(y) \rightarrow \forall x P(x)^2$		$\rightarrow$-I (1.2,1.2.2)
2.	$\forall x(Q(y) \rightarrow P(x)) \rightarrow (Q(y) \rightarrow \forall x P(x))$		$\rightarrow$-I (1,1.3)

Exemplo 3:

1.	$\forall x \forall y P(x,y)$	
2.	$\forall x \forall y (P(x,y) \rightarrow Q(x) \wedge R(y))$	Pergunta: $\exists$ xT(x)?
3.	$\exists x R(x) \wedge \forall x Q(x) \rightarrow \exists x (S(x) \wedge T(x))$	
4.	$\forall y P(a,y)$	$\forall$-E
5,	$P(a,b)$	$\forall$-E(4)
6.	$\forall y (P(a,y) \rightarrow Q(a) \wedge R(y))$	$\forall$-E(2)
7.	$P(a,b) \rightarrow Q(a) \wedge R(b)$	$\forall$-E(6)
8.	$Q(a) \wedge R(b)$	$\rightarrow$-E(5,7)
9.	$Q(a)$	$\wedge$-E(8)
10.	$\forall x Q(x)$	$\forall$-I(9)
11.	$R(b)$	$\wedge$-E(8)
12.	$\exists x R(x)$	$\exists$-I(11)
13.	$\exists x R(x) \wedge \forall x Q(x)$	$\wedge$-I(10,12)
14.	$\exists x (S(x) \wedge T(x))$	$\rightarrow$-E(13,3)
15.	$[S(a) \wedge T(a)]^1$	Suposição
	15.1 $T(a)$	$\wedge$-E(15)
	15.2 $\exists x T(x)$	$\exists$-I(15.1)
16.	$\exists x T(x)^1$	$\exists$-E(14,15,15.2)

3.4 Método de Tableaux

O método de Tableuax apresentado nesta seção foi proposto por Smullian [3] e pode ser encontrado em [4]. Este se caracteriza por não usar unificação. Existem outros Tableaux que utilizam unificação mas não serão estudados neste curso.

O método é o mesmo da Lógica Clássica Proposicional acrescentando-se quatro novas regras para tratar dos quantificadores. Começamos com o ramo incial contendo o BD e a negação da pergunta e aplicamos as regras até obter um Tableaux fechado ou esgotar todas as possibilidades. A seguir apresentamos as novas regras

para os quantificadores $\forall$ e $\exists$.

- Regras para $\wedge$, $\vee$, $\rightarrow$, $\neg$, são as mesmas do caso proposicional, i.e., R_1, R_2, R_3, R_4, R_5, R_6 e R_7.

- ***Regras do Quantificador Universal*** **:**

R_8

$$\frac{\forall x\alpha}{\alpha(x/t)}$$ **Condição:** t é um termo qualquer

R_9

$$\frac{\neg\forall x\alpha}{\neg\alpha(x/t)}$$ **Condição:** t é um termo novo no Tableaux

- ***Regras para o Quantificador Existencial:***

R_{10}

$$\frac{\exists x\alpha}{\alpha(x/t)}$$ **Condição:** t é um termo novo no Tableaux

R_{11}

$$\frac{\neg\exists x\alpha}{\neg\alpha(x/t)}$$ **Condição:** t é um termo qualquer

- **Condição**: nas regras R_8 e R_{11} as fórmulas $\forall x\alpha$ and $\neg\exists x\alpha$ podem ser usadas mais de uma vez no mesmo ramos.

Exemplo 1: $\vdash \forall x P(x) \rightarrow \exists x P(x)$

1.	$\neg(\forall x P(x) \rightarrow \exists x P(x))$	Negação Perg.
2.	$\forall x P(x)$	R_7
3.	$\neg\exists x P(x))$	R_7
4.	P(a)	R_{11}
5.	$\neg P(a)$	R_8

Exemplo 2: $\{\exists x \exists y P(x,y),\ \forall x \forall y (P(x,y) \rightarrow Q(x) \wedge R(y))\} \vdash \exists z R(z)$

1.	$\exists x \exists y P(x,y)$	BD
2.	$\forall x \forall y (P(x,y) \rightarrow Q(x) \wedge R(y))$	BD
3.	$\neg\exists z R(z)$	Negação Perg.
4.	$\exists y P(a,y)$	$R_{10}(1)(x/a)$
5.	$P(a,b)$	$R_{10}(4)(y/b)$
6.	$\forall y (P(a,y) \rightarrow Q(a) \wedge R(y))$	$R_8(2)(x/a)$
7.	$(P(a,b) \rightarrow Q(a) \wedge R(b))$	$R_8(6)(x/a)$
8.	$\neg P(a,b) \mid (Q(a) \wedge R(b))$	$R_7(7)$
9.	$\mid Q(a)$	$R_1(8)$
10.	$\mid R(b)$	$R_1(8)$
11.	$\mid \neg R(b)$	$R_{11}(3)$

3.5 Método Axiomático

- Os conectivos $\wedge$, $\vee$, $\rightarrow$, $\neg$ são definidos pelos mesmos axiomas esquema da Lógica Proposicional.

Axiomas Lógicos:

•***Implicação:***

(1) $\alpha \rightarrow (\beta \rightarrow \alpha\ )$

(2) $\alpha \rightarrow (\beta \rightarrow \gamma)) \rightarrow ((\alpha \rightarrow \beta) \rightarrow \alpha \rightarrow \gamma))$

•***Conjunção:***

(3) $(\alpha \wedge \beta) \rightarrow \alpha$

(4) $(\alpha \wedge \beta) \rightarrow \beta$

(5) $\alpha \rightarrow (\beta \rightarrow (\alpha \wedge \beta))$

•***Disjunção:***

(6) $\alpha \rightarrow \alpha \vee \beta$

(7) $\beta \rightarrow \alpha \vee \beta$

(8) $((\alpha \rightarrow \gamma) \wedge (\beta \rightarrow \gamma)) \rightarrow ((\alpha \vee \beta) \rightarrow \gamma)$

•***Negação:***

(9) $\alpha \rightarrow \neg\neg\alpha$

(10) $\neg\neg\alpha \rightarrow \alpha$

(11) $(\alpha \rightarrow \beta) \rightarrow ((\alpha \rightarrow \neg\beta) \rightarrow \neg\alpha)$

• **Quantificador Universal:**

(12) $\forall x\alpha \rightarrow \alpha$ (x/t),onde x é substituível por t em α; ($\forall$ -elim)

(13)$(\alpha \rightarrow \beta) \rightarrow (\alpha \rightarrow \forall x\beta$), onde x não ocorre livre em α; ($\forall$ -introd)

(14) $\exists x\alpha(x) \leftrightarrow \neg\forall x\neg\alpha(x)$ ***por definição***

Exemplo 1:

1. $\forall x P(x)$
2. $\forall x((P(x) \wedge Q(x)) \rightarrow R(x)) \vdash \forall y R(y)$?
3. $\forall x Q(x)$

4. $\forall x P(x) \to P(y)$ axioma 12 x/y

5. P(y) MP(1,4)

6. $\forall x Q(x) \to Q(y)$ axioma 12 x/y

7. Q(y) MP(3,6)

8. $P(y) \to (Q(y) \to (P(y) \wedge Q(y)))$ axioma 5

9. $Q(y) \to (P(y) \wedge Q(y))$ MP(5,8)

10. P(y) $\wedge$ Q(y) MP(7,9)

11. $\forall x((P(x) \wedge Q(x)) \to R(x)) \to ((P(y) \wedge$ Q(y)$) \to R(y))$ axioma12 x/y

12. $(P(y) \wedge Q(y)) \to R(y)$ MP(2,11)

13. R(y) MP(10,12)

14. $R(y) \to (\forall x P(x) \to R(y))$ axioma1

15. $\forall x P(x) \to R(y)$ MP(13,14)

16. $(\forall x P(x) \to R(y)) \to (\forall x P(x) \to \forall y R(y))$ axioma13

17. $\forall x P(x) \to \forall y R(y)$ MP(15,16)

18. $\forall y R(y)$ MP(1,17)

OBS: As noções de prova, teorema e a relação de derivabilidade $\Gamma \vdash \alpha$ são análogas às da Lógica Proposicional.

3.6 Semântica

Nesta seção apresentaremos a semântica da Lógica Clássica de Primeira Ordem somente para sentenças, isto é, fórmulas sem ocorrência de varáveis livres.

- Revisão da Semântica da Lógica Proposicional:

Maria foi ao cinema. Se ela foi ao cinema então ela comprou pipoca e assistiu ao filme. Se ela comprou pipoca então ela tem dinheiro ou ela pegou emprestado com João. Se ela pegou emprestado com João então João tem dinheiro.

C: Maria foi ao cinema.

P: Maria comprou pipoca.

F: Maria assistiu ao filme.

D: Maria tem dinheiro.

E: Maria pegou dinheiro emprestado com João.

J: João tem dinheiro.

C

$C \rightarrow P \wedge F$

$P \rightarrow D \vee E$

$E \rightarrow J$

$v(C) = F$

$v(P) = V$

$v(D) = v(E) = F$

$v(J) = V$

Não é um modelo para o conjunto de fórmulas, pois não satisfaz todas as fórmulas.

$v(C) = V$

$v(P) = V$

$v(F) = V$

$v(D) = F$

$v(E) = V$

$v(J) = V$

é um modelo.

$v(C) = V$

$v(P) = V$

$v(F) = V$

$v(D) = V$

$v(E) = F$

$v(J) = F$

é um modelo.

A semântica da lógica de primeira ordem tem como objetivo atribuir significados às fórmulas da linguagem.

- Uma fórmula só tem significado quando uma interpretação é dada a seus símbolos não lógicos.
- $\forall x(Q(x) \rightarrow P(x))$ é verdadeira ou falsa?

Nós só podemos dizer se esta fórmula é V ou F se interpretarmos seus símbolos não-lógicos.

Primeiro, precisamos saber qual o universo em que as variáveis estão quantificando. Por exemplo: números inteiros, números reais, pessoas...

Depois, precisamos interpretar os predicados, funções e constantes.

Exemplo: $\forall x(Q(x) \rightarrow P(x))$

Interpretação:

•**universo:**pessoas

•**predicados:** Q: é funcionário da UFRJ. P: é funcionário público.

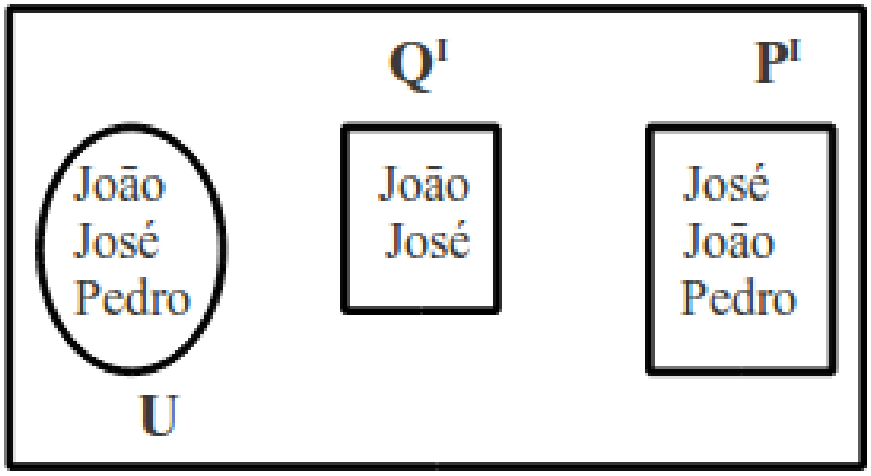


Figura 3.1:

$\forall x(Q(x) \rightarrow P(x))$ é verdadeira na interpretação acima.

Exemplo 2:

U={João, José, Pedro}

$Q^I = \{< Joao >, < Jose >\}$

$P^I = \{< Jose >, < Pedro >\}$

$\forall x(Q(x) \rightarrow P(x))$ é falsa nesta interpretação.

Exemplo 3: $\forall x(Q(x) \rightarrow P(x))$

$U = Z$ (inteiros)

$Q^I = \{< 0 >, < 1 >, ...\}$(naturais)

$P^I = \{... < -2 >, < -1 >, < 0 >, < 1 >, < 2 >, ...\}$ (inteiros)

$\forall x(Q(x) \rightarrow P(x))$ é verdadeira nesta interpretação.

Exemplo 4: $\exists x(P(x) \wedge Q(x, c))$

$U = R$ (reais)

$Q^I = x > c$

$P^I = x$ é racional

$c^I = 0$

"Existe algum nÚmero real que também é racional e maior do que zero."

Exemplo 5: $\forall x(P(x) \wedge Q(x) \rightarrow R(x, f(c)))$

$U = Z$ (inteiros)

$c^I = 0$

$f^I = x + 1$

$Q^I = \{< 2 >, < 4 >, < 6 >, ...\}$

$P^I = \{< 1 >, < 2 >, < 3 >, ...\}$

$R^I = x > y$

"Todo número inteiro positivo e par é maior do que 1." (verdadeiro)

Exemplo 6: $\forall x(P(x) \wedge Q(x) \rightarrow R(x, f(c)))$

$U = Z$ (inteiros)

$c^I = 4$

$f^I = x + 1$

$Q^I = \{< 2 >, < 4 >, < 6 >, ...\}$

$P^I = \{< 1 >, < 2 >, < 3 >, ...\}$

$R^I = x > y$

"Todo número inteiro positivo e par é maior do que 4." (falso)

Exemplo 7: $(\forall y C(x, y)) \rightarrow G(x)$

U = {José,João,Pedro,Paulo}

C : x é chefe de y

G : x é gerente

C^I	
João	José
João	Paulo
João	Pedro
João	João
Paulo	João
Paulo	Paulo
Paulo	Pedro
Paulo	José
Paulo	José

G^I
João
Pedro

x = João $\Longrightarrow$ V

x = José $\Longrightarrow$ V

x = Pedro $\Longrightarrow$ V

x = Paulo $\Longrightarrow$ F

Porém, pela nossa definição da linguagem de LPO, podemos ter variáveis livres ocorrendo nas fórmulas, por exemplo

$$\forall x C(x, y)$$

A variável y ocorre livre nesta fórmula.

Em geral, para sabermos se uma fórmula é verdadeira ou falsa, nós precisamos saber o universo e interpretar cada símbolo não-lógico neste universo e dar valor as variáveis livres.

(1) Interpretar variáveis livres e constantes em elementos do domínio.

(2) Interpretar predicados em relações entre elementos do domínio.

(3) Interpretar funções em funções sobre o domínio.

Definição: Definimos uma **interpretação** como sendo um par ordenado $\mathcal{I} =< D, I >$ onde D é um conjunto não-vazio de indivíduos chamado **domínio**. E I é uma função chamada de **função de interpretação**, definida como:

1. I associa a cada variável livre x um elemento do domínio $d^I \in D$.

   $I(x) = d^I$

2. I associa a cada constante c, um elemento do domínio $c^I \in D$.

   $I(c) = c^I$

3. I associa a cada símbolo funcional n-ário f uma função n-ária $f^I : D^n \to D$ tal que $I(f(t_1,...,t_n)) = f^I(I(t_1),...,I(t_n))$, onde $t_1,...,t_n$ são termos.

4. I associa a cada símbolo predicativo n-ário P uma relação n-ária sobre D. $I(P) = P^I$, $P^I \subseteq D^n$, ie, $P^I \subseteq D \times D\times,,,\times D$, n vezes.

Definição:

Seja L uma linguagem de primeira ordem e α e β, fórmulas de L, $t_1, ..., t^n$ termos, P um símbolo predicativo n-ário e $< D, I >$ uma interpretação. Definimos a função de avaliação de fórmulas de L como:

$V_{\mathcal{I}} : W \to \{V, F\}$, onde W é o conjunto de fórmulas, tal que:

(1) $V_{\mathcal{I}}(P(t_1, ..., t_n)) = V$ se somente se $< I(t_1), ..., I(t_n) > \in P^I$. F caso contrário.

(2) $V_{\mathcal{I}}(\neg\alpha) = V$ se $V_I(\alpha) = F$. F caso contrário.

(3) $V_{\mathcal{I}}(\alpha \wedge \beta) = V$ se $V_{\mathcal{I}}(\alpha) = V$ e $V_{\mathcal{I}}(\beta) = V$. F caso contrário.

(4) $V_{\mathcal{I}}(\alpha \vee \beta) = F$ se $V_{\mathcal{I}}(\alpha) = F$ e $V_{\mathcal{I}}(\beta) = F$. V caso contrário.

(5) $V_{\mathcal{I}}(\alpha \to \beta) = F$ se $V_{\mathcal{I}}(\alpha) = V$ e $V_{\mathcal{I}}(\beta) = F$. V caso contrário.

(6) $V_{\mathcal{I}}(\forall x\alpha) = V$ se somente se para todo $d \in D$, se $I(x) = d$ então $V_{\mathcal{I}}(\alpha) = V$. F caso contrário.

(7) $V_{\mathcal{I}}(\exists x\alpha) = V$ se para algum $d \in D$, $I(x) = d$ e $V_{\mathcal{I}}(\alpha) = V$. F caso contrário.

Definição:

Seja L uma linguagem de 1a ordem. I uma interpretação para L, Γ um conjunto de fórmulas de L e α uma fórmula.

1. I **satisfaz** $\alpha(\models_I \alpha)$ se somente se $V_I(\alpha) = V$;
2. I **satisfaz** Γ se somente se satisfaz cada membro de Γ ;
3. Γ é **satisfatível** se somente se existe uma interpretação I que satisfaça Γ;
4. α é **válida** $(\models \alpha)$ se somente se para toda interpretação I, $\models_I \alpha$, i.e., $V_I(\alpha) = V$ para todo I; (*válida é equivalente a tautologia*)
5. Γ **implica logicamente** em $\alpha(\Gamma \models \alpha$) se somente se para toda interpretação I, se I satisfaz Γ, então I satisfaz α;
6. Γ é **insatisfatível** se somente se Γ não é satisfatível, i.e., não existe uma interpretação I que satisfaz Γ;
7. Uma interpretação I que satisfaz Γ é dita **modelo** para Γ.

Exemplo1: $\forall x(P(x) \rightarrow$ E(x,s(x))

Esta fórmula é satisfatível?

Interpretação: D = { João, José, Pedro, 0,100, 200}

P : pessoas

E : empregado

s : salário

Pessoas
João
José

Empregado	
João	100
José	200
Pedro	0

Salário	
José	100
João	200
...	...

OBS: As funções têm que ser totais,i.e., devem retornar algum valor pertencente ao domínio a cada elemento do domínio.

$V_I(\forall x(P(x) \rightarrow E(x, s(x))) = ?$

•Para todo $d \in D$:

d = João

V_I(P(João))= V V_I (E(João,200)) = F $\Rightarrow V_I(\forall x(P(x) \rightarrow$ E(x,s(x))) = F

Trocando os valores na tabela de salário:

Salário	
José	200
João	100
...	...

•Para todo $d \in D$:

d = João

V_I (P(João))= V V_I (E(João,100))= V $\Rightarrow V_I(\forall x(P(x) \rightarrow$ E(x,s(x))) = V

d = José

V_I (P(José))= V V_I (E(José,200))= V $\Rightarrow V_I(\forall x(P(x) \rightarrow$ E(x,s(x))) = V

d = Pedro

V_I (P(Pedro))= F V_I (E(Pedro,...))= ? $\Rightarrow V_I(\forall x(P(x) \rightarrow$ E(x,s(x))) = V

d = 0

V_I (P(0))= F V_I (E(0,...))= ? $\Rightarrow V_I(\forall x(P(x) \rightarrow$ E(x,s(x))) = V

d = 100

V_I (P(100))= F V_I (E(100,...))= F $\Rightarrow V_I(\forall x(P(x) \rightarrow$ E(x,s(x))) = V

d = 200

V_I (P(200))= F V_I (E(200,...))= F $\Rightarrow V_I(\forall x(P(x) \rightarrow$ E(x,s(x))) = V

Exemplo 2: $\forall x(P(x) \wedge \exists y Q(x,y))$

Não Satisfaz	Satisfaz
$D = \{0,1\}$	$D = \{0,1\}$
$P^I = \{< 0 >\}$	$P^I = \{< 0 >, < 1 >\}$
$Q^I = \{< 0,1 >\}$	$Q^I = \{< 0,1 >, < 1,0 >\}$

Exemplo 3: $\forall x(P(x,y) \rightarrow Q(x) \vee R(c))$

Não Satisfaz	Satisfaz
$D = \{0,1\}$	$D = \{0,1\}$
$y^I = 0$	$y^I = 0$
$c^I = 0$	$c^I = 0$
$P^I = \{< 1,1 >\}$	$P^I = \{< 0,0 >, < 1,0 >\}$
$Q^I = \{< 1 >\}$	$Q^I = \vdash$
$R^I = \{< 0 >\}$	$R^I = \{< 1 >\}$

Exercício:

Dada a seguinte estrutura:

$D = \{joao, jose, ana, maria\}$

Filhiacao	
jose	joao
maria	jose
joao	ana

Homem
jose
jose

Mulher
ana
maria

Pai	
joao	jose
jose	maria

Interprete a fórmula $\forall x \forall y(F(y,x) \wedge H(x) \rightarrow P(x,y))$ e verifique formalmente se ela é verdadeira ou falsa.

3.7 Relação entre Sintaxe e Semântica

TEOREMA DA CORRETUDE:

Se $\Gamma \vdash \alpha$ então $\Gamma \models \alpha$..

TEOREMA DA COMPLETUDE:

Se $\Gamma \models \alpha$ então $\Gamma \vdash \alpha$.

3.8 Tabelas - SQL × Lógica

Nesta seção ilustramos o poder de expressão da LCPO para representar consultas em SQL.

Dada as seguintes tabelas e as consultas em SQL represente as consultas em LCPO.

CLIENTE		
Cliente	**Rua**	**Cidade**
João	Monte	Rio
José	Sol	Maceió
Pedro	Flores	Curitiba

AGÊNCIA		
Agência	**Fundos**	**Cidade**
345	100.000,00	Rio
243	340.000,00	Salvador
610	520.000,00	Porto Alegre

EMPRÉSTIMO			
Agência	**Empréstimo**	**Cliente**	**Valor**
345	107	João	2.000,00
243	340	José	7.000,00
619	573	Maria	10.000,00

DEPÓSITO			
Agência	**Num. C.C,**	**Cliente**	**Saldo**
345	10050	João	500,00
243	10129	Ana	750,00
619	23040	Maria	200,00

Representar as seguintes consultas em lógica. Supondo que as relações $=, \neq, <, >, \leq, \geq$ são pré-definidas.

1. Todos os clientes tendo conta na agência 345.

   select Cliente
   from DEPÓSITO
   where Agência = 345

2. Todos os clientes tendo um empréstimo na agência 345.

   select Cliente
   from EMPRÉSTIMO
   where Agência = 345

3. Todos os clientes tendo um empréstimo, uma conta ou ambos na agência 345.

   select Cliente
   from DEPÓSITO
   where Agência = 345
   Union
   select Cliente
   from EMPRÉSTIMO
   where Agência = 345

4. Todos os clientes tendo um empréstimo e uma conta na agência 345.

   select Cliente
   from DEPÓSITO
   where Agência = 345
   Intersect
   select Cliente
   from EMPRÉSTIMO
   where Agência = 345

5. Todos os clientes tendo conta na agência 345 mas não tendo um empréstimo lá.

select Cliente
from DEPÓSITO
where Agência = 345
Minus
select Cliente
from EMPRÉSTIMO
where Agência = 345

6. Todos os clientes e suas cidades que tem empréstimo em alguma agência.

select CLIENTE.Cliente, CLIENTE.Cidade
from EMPRÉSTIMO, CLIENTE
where EMPRESTIMO.Cliente=CLIENTE.Cliente

7. Todos os clientes e suas cidades que tem empréstimo na agência 345.

select CLIENTE.Cliente, CLIENTE.Cidade
from EMPRÉSTIMO, CLIENTE
where EMPRESTIMO.Cliente=CLIENTE.Cliente **and** Agência=345

8. Todos os clientes que tem conta em alguma agência que João tem conta.

select Cliente
from DEPÓSITO
where Agência **In**
select Agência
from DEPÓSITO
where Cliente=João

9. Ache todas as agências tem um fundo maior que alguma agência em Salvador.

select Agência
from AGÊNCIA
where Fundos > **Any**
select Fundos
from AGÊNCIA
where Agência = Salvador

10. Ache todas as agências tem um fundo maior que todas as agências em Salvador.

select Agência
from AGÊNCIA
where Fundos > **All**

select Fundos
from AGÊNCIA
where Agência = Salvador

11. Ache todos os clientes tem que conta em agências em Salvador.

select Cliente
from DEPÓSITO S
where

select Agência
from DEPÓSITO T
where S.Cliente = T.Cliente
Contains
select Agência
from AGÊNCIA
where AGÊNCIA.Cidade = Salvador

3.9 Estruturas e Teorias

Nesta seção gostariamos de apresentar alguns exemplos de estruras relacionais conhecidas e como certas formulas podem ser interpretadas nestas. Achar um conjunto de fórmulas que são verdadeiras exatamente em uma certa clásse de estruturas. Estudaremos os números naturais, grafos, ordens e árvores.

Quando juntamos um conjunto de fórmulas não lógicas a axiomatização da Lógica de Primeira ordem obtemos uma **Teoria**. A partir da teoria podemos deduzir propriedades (teoremas) sobre a estrutura sendo representada pela teoria.

Grafos, Ordens e Árvores

Garfos

Um grafo $G = (V, A)$ é uma par onde V é um conjunto não vazio de vértices e A é uma relação binária sobre V, $A \subseteq V \times V$.

Um linguagem, básica, de primeira ordem para representar grafos deverá ter um símbolo predicativo 2-ário para ser interpretado como A. E o domínio da interpretação deve ser o conjunto de vétices V.

Linguagem: predicado 2-ário R.

Interpretação:

- $D = V$
- $I(R) = A$

Podemos escrever fórmulas que impõem condições sobre o tipo de grafo. Por exemplo, a fómula

$$\forall x R(x, x)$$

é verdadeira, na interpretação a cima se e somente se a relação A for **refleiva**.

Outros exemplos de condições são:

Condição		Fórmula
Rx.	Reflexividade	$\forall x R(x,x)$
IRx.	Ireflexividade	$\forall x \neg R(x,x)$
Sm.	Simétria	$\forall x \forall y R(x,y) \rightarrow R(y,x)$
Tr.	Transitividade	$\forall x \forall y (\exists z (R(x,z) \wedge R(z,y))) \rightarrow R(x,y)$
Sl.	Serial (Total)	$\forall x \exists y R(x,y)$
Eu.	Euclidiana	$\forall x \forall y \forall z (R(x,z) \wedge R(x,y)) \rightarrow R(z,y)$
ASm.	Anti-Simétrica	$\forall x \forall y R(x,y) \wedge R(y,x) \rightarrow x = y$
Tc.	Tricotomia	$\forall x \forall y (R(x,y) \vee x = y \vee R(y,x)$

Outra clásse de grafos muito usada em computação é clásse dos grafos k-coloríveis. Estes são os grafos que podem ser coloríveis com k cores respeitando as seguintes condições:

1. todo vértice é atribuida uma única cor;
2. vértices vizinhos tem cores distintas.

Estes grafos formam uma estrutura com mais k relações unárias para representar as cores, $G = (V, A, Cor_1, \cdots, Cor_k)$. Para expressar estes grafos precisamos estender nossa linguagem comok símbolos de predicados $C_1, \cdots C_k$ e interpretá-los como

- $I(C_i) = Cor_i$, para todo $1 \leq i \leq k$

exercício: Escreva as fórmulas para expressar as condições 1 e 2 para um grafo ser 3-colorível.

Se juntar algumas destas fórmulas aos axiomas da Lógica de Primeira Ordem obteremos uma teoria dos grafos, por exemplo podemos ter a teoria dos grafos reflexivos e simétricos e etc.

Ordens

Um relação de ordem pode ser vista como um grafo onde o conjunto de aresta A é a prórpria relação de ordem $\leq$ ou $<$ dependendo se a ordem é estrita ou não. Para ter uma ordem algumas condições devem ser impostas:

Ordem		Fórmulas
Pré	Pré-Ordem	Rx + Tr
Par.	Ordem Parcial	Rx + Tr + ASm
Tot	Odem Total(linear)	Rx + Tr + ASm + Tc
Est.	Estrita	Subst. Rx por IRx em Pré, Par, Tot

Se juntar algumas destas fórmulas aos axiomas da Lógica de Primeira Ordem obteremos uma teoria das ordens, por exemplo podemos ter a teoria dos grafos parciais e etc.

Árvores

Uma árvore é um grafo conexo com um vértice especial chamado raiz tal que deste vértice só existe um único caminho para qualquer outro vértice. Uma árvore pode ser vista como um grafo $G = (V, A, raiz)$. Nós vamos estender a linguagem dos grafos com uma constante r para denotar a $raiz$,

- $I(r) = raiz$

exercício: Escreva as fórmulas para expressar que um grafo é uma árvore. Dica: defina um novo símbolo de predicado, na linguagem, para expressar caminho entre dois vértices, $C(x, y)$ se exsite um caminho de x para y e/ou use a relação de $=$.

Se juntar estas fórmulas aos axiomas da Lógica de Primeira Ordem obteremos uma teoria das árvores.

Teoria dos Números

Outro exemplo de estrutura são os números Naturais e as operações básicas de aritimética. Dada a seguinte estrutura $A_E = \langle \mathbb{N}, 0, \mathrm{S}, <, +, ., \mathrm{E} \rangle$ sobre os Naturais nós podemos escrever as seguintes fórmulas (axiomas) e interpretá-los nesta estrutura.

Axiomas de A_E

S1. $\forall x S(x) \neq 0$

S2. $\forall x \forall y (S(x) = S(y) \rightarrow x = y)$

L1. $\forall x \forall y (x < S(y) \leftrightarrow x \leq y)$

L2. $\forall x \not< 0$

L3. $\forall x \forall y (x < y \vee x = y \vee y < x)$

A1. $\forall x (x + 0) = x$

A2. $\forall x \forall y (x + S(y)) = S(x + y)$

M1. $\forall x (x.0) = 0$

M2. $\forall x \forall y (x.S(y)) = (x.y) + x$

E1. $\forall x (xE0) = S(0)$

E2. $\forall x \forall y (xES(y)) = (xEy).x$

Um leitor mais familiarizado notará que os seguintes axiomas foram retirados de A_E:

S3. $\forall y(y \neq 0 \rightarrow \exists x\ y = S(x))$

Indução. $(\varphi(0) \wedge \forall x(\varphi(x) \rightarrow \varphi(S(x)))) \rightarrow \forall x\varphi(x)$

Se juntarmos estes axiomas com a nossa axiomatixação da Lógica de Primeira Ordem teremos uma axiomatização para a aritimética dos números naturais, i.e, uma teoria dos números Naturais.

De fato, mesmo sem estes axiomas, nós podemos provar um teorema muito iteressante:

Teorema 1 *Uma relação R é recursiva sse R é representável em $Cn(A_E)$.*

Capítulo 4

Lógicas Modais

Este material está sendo construido durante o curso. Faltam várias figuras, provas, exemplos e explicações.

4.1 Linguagem

4.1.1 Alfabeto modal sobre Φ

Dado um conjunto Φ de símbolos proposicionais, $\Phi = \{p, q, ...\}$, o *alfabeto modal* sobre Φ é constituído por: cada um dos elementos de Φ; o símbolo $\perp$ (absurdo); os conectivos lógicos $\neg$ (negação), $\rightarrow$ (implicação), $\wedge$ (conjunção) e $\vee$ (disjunção); os operadores modais $\Box$ (necessidade) e $\Diamond$ (possibilidade); e os parênteses, como símbolos auxiliares.

4.1.2 Linguagem modal induzida pelo alfabeto modal sobre Φ

A *linguagem modal induzida pelo alfabeto modal sobre* Φ é definida indutivamente da seguinte forma:

$$\varphi ::= p \mid \bot \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \vee \varphi_2 \mid \varphi_1 \rightarrow \varphi_2 \mid \neg\varphi \mid \Box\varphi \mid \Diamond\varphi$$

4.2 Semântica

4.2.1 *Frames*

Um *frame* é um par $F = (W, R)$ onde W é um conjunto não-vazio de *estados* e R é uma relação binária em W dita *relação de acessibilidade*. Diz-se que $s_2 \in W$ é *acessível* a partir de $s_1 \in W$ se, e somente se, $(s_1, s_2) \in R$.

No exemplo da figura 4.1 o conjundo de estados é $W = \{s_1, s_2, s_3, s_4, s_5\}$ e a relação de acessibilidade é $R = \{(s_1, s_2), (s_1, s_3), (s_3, s_3), (s_3, s_4), (s_2, s_4), (s_2, s_5), (s_4, s_1), (s_4, s_5), (s_5, s_5)\}$. O frame é $F = (W, R)$.

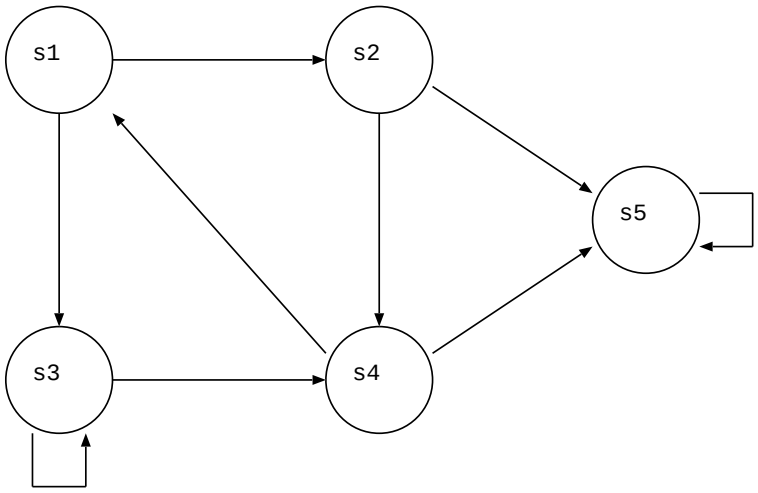


Figura 4.1: Exemplo de um Frame.

4.2.2 Modelos

Um *modelo* sobre o conjunto Φ é um par $M = (F, V)$ onde $F = (W, R)$ é um *frame* e V é uma função de Φ no conjunto das partes de W, que faz corresponder a todo símbolo proposicional $p \in \Phi$ o conjunto de estados nos quais p é satisfeito, i.e., $V : \Phi \longmapsto Pow(W)$.

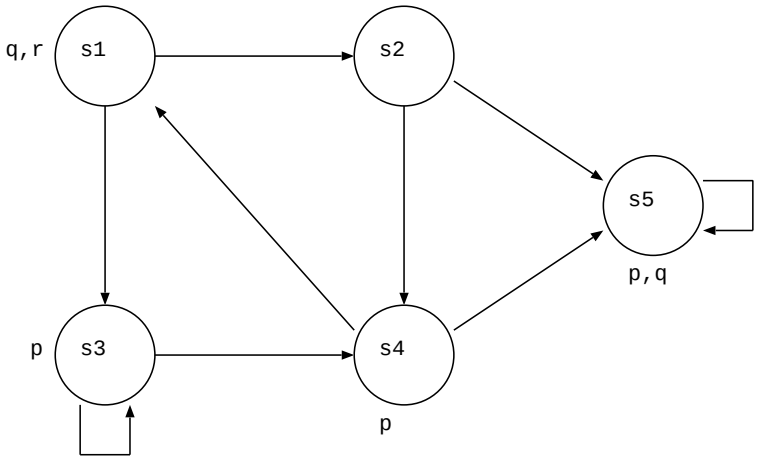


Figura 4.2: Exemplo de um Modelo.

No exemplo da figura 4.2 o frame é o mesmo da figura 4.1 e a função V é:

- $V(p) = \{s_3, s_4, s_5\}$
- $V(q) = \{s_1, s_5\}$
- $V(r) = \{s_1\}$

4.2.3 Satisfação

Seja $M = (F, V)$ um modelo e $w \in W$ um estado. A notação $M, w \Vdash \varphi$ indica que a fórmula φ é *satisfeita* pelo modelo M no estado w, o que é definido indutivamente como:

- $M, w \Vdash p$ sse $w \in V(p)(\forall p \in \Phi)$
- $M, w \nVdash \bot$
- $M, w \Vdash \neg\varphi$ iff $M, w \nVdash \varphi$,
- $M, w \Vdash \varphi \to \varphi'$ sse $M, w \nvDash \varphi$ **ou** $M, w \Vdash \varphi'$
- $M, w \Vdash \varphi \wedge \varphi'$ sse $M, w \Vdash \varphi$ **e** $M, w \Vdash \varphi'$
- $M, w \Vdash \varphi \vee \varphi'$ sse $M, w \Vdash \varphi$ **ou** $M, w \Vdash \varphi'$
- $M, w \Vdash \Box\varphi$ sse para todo $w' \in W$ **se** wRw' **implica** $M, w' \Vdash \varphi$
- $M, w \Vdash \Diamond\varphi$ sse existe $w' \in W$, wRw' **e** $M, w' \Vdash \varphi$

Nós podemos generalizar a noção de satisfação para conjuntos de fórmulas. Se $\Gamma = \{\phi_1, \cdots, \phi_n\}$ então $M, w \Vdash \Gamma$ sse $M, w \Vdash \phi_i$, para todo $1 \leq i \leq n$.

Exemplo: Seja M o modelo da figura 4.2. Queremos verificar se $M, s_2 \Vdash \Box p$.

$M, s_2 \Vdash \Box p$ sse para todo $w' \in W$ **se** $s_2 R w'$ **implica** $M, w' \Vdash p$, nós precisamos verificar para $w' \in \{s_1, s_2, s_3, s_4, s_5\}$. Como temos uma implicação, para os não vizinhos de s_2 a implicação é vacoamente verdadeira. Então precisamos verificar somente para

- $w' = s_4$, $M, s_4 \Vdash p$ sse $s_4 \in V(p)$ o que é verdade;
- $w' = s_5$, $M, s_5 \Vdash p$ sse $s_5 \in V(p)$ o que é verdade.

A seguir apresentamos um algoritmo para verificar se uma fórmula modal φ é satisfeita num modelo $M = (W, R, V)$[1] num estado w.

[1]Usaremos no texto $M = (W, R, V)$ quando na verdade deveriamos usar $M = (F, V)$ e $F = (W, R)$.

função Satisfaz(φ,M, w): **booleano**

caso φ:

p:	**se** $w \in V(p)$ **então retorna verdadeiro**
	senão retorna falso
$\bot$:	**retorna falso**
$\neg\varphi_1$:	**retorna not** Satisfaz(φ_1,M,w)
$\varphi_1 \wedge \varphi_2$:	**retorna** Satisfaz(φ_1,M,w) **and** Satisfaz(φ_2,M,w)
$\varphi_1 \vee \varphi_2$:	**retorna** Satisfaz(φ_1,M,w) **or** Satisfaz(φ_2,M,w)
$\varphi_1 \rightarrow \varphi_2$:	**retorna not** Satisfaz(φ_1,M,w) **or** Satisfaz(φ_2,M,w)
$\Diamond\varphi_1$:	**para todo** w' t. q. wRw' **faça**
	se Satisfaz(φ_1,M,w')**então retorna verdadeiro**
	retorna falso
$\Box\varphi_1$:	**para todo** w' t. q. wRw' **faça**
	se not Satisfaz(φ_1,M,w')**então retorna falso**
	retorna verdadeiro

Complexidade: para cada conectivo booleano são feitas, no pior caso, duas chamadas e para cada ocorrência de símbolo proposicional temos uma chamada. Para os conectivos modais temos que percorrer a lista de adjacências, no pior caso, para todos os estados de W. Logo a complexidade é $O(|\varphi| \times (|W| + |R|))$, isto é, linear no tamanho da fórmula e no tamanho do modelo.

Exercício 1 *No modelo da figura 4.2 verifique:*

1. $M, w \Vdash \Box(p \rightarrow \Diamond p)$
2. $M, w \Vdash \Box(\Diamond(p \wedge \Diamond(r \wedge s)))$

3. $M, w \Vdash \Diamond(p \wedge \Diamond(p \wedge \Diamond r))$

4. $Satisfaz(\Box\Box\neg r, M, s_1)$

5. $Satisfaz(\Box(\neg r \rightarrow \Diamond p), M, s_1)$

6. $Satisfaz(\Box(\neg r \wedge \Diamond\Diamond(p \wedge q)), M, s_1)$

7. $Satisfaz(\Box\Diamond q, M, s_1)$

4.2.4 Clásses de Frames

Nesta seção apresentamos algumas clásses de frames que são mais usuais.

Seja um frame $F = (W, R)$ e $\mathcal{F}$ a clásse de todos os frames.

Clásse dos Frames Reflexivos $\mathcal{F}_r$

Composta pelos frames cuja a relação de acessibilidade seja reflexiva.

$$\forall x \in W \ (xRx)$$

Figura 4.3: Exemplo de Frame Reflexivo

Clásse dos Frames Simétricos $\mathcal{F}_s$

Composta pelos frames cuja a relação de acessibilidade seja simétrica.

$$\forall x, y \in W \ (xRy \rightarrow yRx)$$

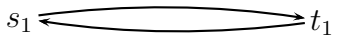


Figura 4.4: Exemplo de Frame Simétrico

Clásse dos Frames Transistivos $\mathcal{F}_t$

Composta pelos frames cuja a relação de acessibilidade seja transitiva.

$$\forall x, y, z \in W \ (xRy \wedge yRz \rightarrow xRz)$$

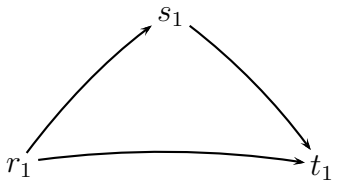


Figura 4.5: Exemplo de Frame Transitivo

Clásse dos Frames Seriais $\mathcal{F}_{serial}$

Composta pelos frames cuja a relação de acessibilidade seja serial.

$$\forall x \exists y \ (xRy)$$

Figura 4.6: Exemplo de Frame Serial

Referências Bibliográficas

[1] F. S. C. Silva, M. Finger e A. C. V. Melo. *Lógica para a Computação.* Thomson Learning, 2006.

[2] H. B. Enderton. *A Mathematical Indroduction to Logic.* Academic Press, 1972.

[3] R. M. Smullyan. *First Order Logic.* Springer-Verlag, 1968.

[4] M. M. C. Costa. *Introdução à Lógica Modal Aplicada à Computação.* VIII Escola de Computação. Gramado, 1992.

[5] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic.* Theoretical Tracts in Computer Science. Cambridge University Press, 2001.

[6] R. Goldblatt. Logics of Time and Computation. *CSLI* Lecture Notes 7,1992.

[7] D. Harel, D. Kozen D. and Tiuryn. Dynamic Logics. MIT Press, 2000.

[8] Chellas, B. (1980). *Modal Logic, An Introduction.* Cambridge UP, Cambridge, U.K.

[9] Halpern, J. Y., R. Fagin, Y. Moses and M. Y. Vardi (1995). *Reasoning about knowledge.* MIT Press, Massachusets, U.S.A.

[10] Hughes, G. E, Cresswell, M.J. (1996). *A New Introduction to Modal Logic.* Routledge, London and New York.