Instituto Tecnológico de Buenos Aires

LÓGICA COMPUTACIONAL

Autor:

Tomás S. Marengo*

*tmarengo@itba.edu.ar - se aceptan correcciones y/o sugerencias.

1. Cardinalidad

DEF: A es coordinable con B (A$\sim$B) si $\exists$ f:A$\to$B biyectiva. A$\mathcal{R}$B si A$\sim$B $\Rightarrow \mathcal{R}$ es de equivalencia.

DEF: Sección inicial $I_n = \{1, 2, 3, ..., n\}, n \in \mathbb{N}$

OBS: $I_n \sim I_m \Leftrightarrow n = m$

DEF: A es **finito** si $A = \emptyset$ ó $\exists k \in \mathbb{N}_{\geq 1} / A \sim I_k$

DEF: A es **infinito** si no es finito.

Cardinales conocidos

$\#P(X) = 2^{\#X}$ $\quad \#\mathbb{R}^k = C$ $\quad \#[0,1] = C$ $\quad \#(\mathbb{Z} \times \mathbb{N}) = \aleph_0$ $\quad \#\mathbb{N}^k = \aleph_0$ $\quad \#\mathbb{Z} = \aleph_0$ $\quad \#\mathbb{Q}^k = \aleph_0$
$\#\emptyset = 0$ $\quad \#I_k = K$

1.1. Comparación de cardinales

A y B conjuntos $\#A = m$ y $\#B = k$

1. $m \leq k \Leftrightarrow \exists\ f : A \to B$ *inyectiva*
2. $m \geq k \Leftrightarrow \exists\ f : A \to B$ *sobreyectiva*
3. $m = k \Leftrightarrow \exists\ f : A \to B$ *biyectiva*
4. $m < k \Leftrightarrow m \leq k$ y $m \neq k$
5. $m > k \Leftrightarrow m \geq k$ y $m \neq k$

Teorema de Bernstein y propiedades

1. $m \leq k$ y $m \geq k \Rightarrow m = k$ " Si $\exists\ f$:$A \to B$ iny y $\exists\ f^*$:$A \to B$ *sobrey* $\Rightarrow \exists\ f'$:A$\to$ B *biy* "
2. $m \leq n \Leftrightarrow n \geq m$ " $\exists\ f : A \to B$ *iny* $\Leftrightarrow \exists\ f : A \to B$ *sobrey* "
3. $m \leq n$ y $n \leq m \Rightarrow n = m$ " $f : A \to B$ *iny* y $g : A \to B$ *sobrey* $\Rightarrow \exists\ h : A \to B$ *biy* "

Propiedades

1. Cualquier subconjunto de una sección inicial es finito.
2. $A \subseteq B$ y A es infinito $\Rightarrow B$ es infinito.
3. X infinito $\Rightarrow \exists\ f : \mathbb{N} \to X$ *iny* $(\aleph_0 = \#\mathbb{N} \leq \#X)$

1.2. Numerables

DEF: A es **numerable** si A es finito o $A \sim \mathbb{N}$

Propiedades

1. A es numerable y $A \neq \emptyset \Rightarrow \exists\ f : \mathbb{N} \to A$ *sobrey.*
2. $\exists f : \mathbb{N} \to A$ *sobrey* $\Rightarrow A$ es numerable.
3. $A \sim \mathbb{N}$ y $A \sim \mathbb{N} \Rightarrow A \times B \sim \mathbb{N}$
4. A y B numerables $\Rightarrow A \cup B$ es numerable.
5. $(A_n)_{n\in\mathbb{N}}$ una familia de conjuntos numerables $\Rightarrow \cup_{n\in\mathbb{N}} A_n$ es numerable.
6. A finito no vacío y $S = \cup_{m\in\mathbb{N}_{>0}} A^m \Rightarrow \#S = \aleph_0$

1.3. Infinito no numerable

Propiedad: $[0,1] = \{x \in \mathbb{R}/0 \leq x \leq 1\}$ es infinito no numerable.

Propiedades

1. X es infinito no numerable, A numerable $\Rightarrow X \cup A \sim X$
2. X es infinito no numerable, A numerable $\Rightarrow X - A \sim X$

1.4. Infinitos mayores y álgebra de cardinales

Teorema de Cantor: $\#X < \#P(X)$

Álgebra de cardinales

$a = \#X$, $b = \#Y$, $c = \#Z$

1. $a + b = \#(X \cup Y)\ X \cap Y = \emptyset$
2. $a \times b = \#(X \times Y)$
3. $b^a = \#\{f : X \to Y / \ f\ es\ función\}$
4. $(a^b)^c = a^{bc}$
5. $Si\ b \leq c \Rightarrow a^b \leq a^c, \quad b^a \leq c^a, \quad a \cdot b \leq a \cdot c$
6. $a^b \cdot a^c = a^{b+c}$

Equivalencias

1. $\aleph_0 = n + \aleph_0 = \aleph_0 + \aleph_0 = n \cdot \aleph_0 = \aleph_0 \cdot \aleph_0$
2. $C = 2^{\aleph_0} = C + \aleph_0 = C \cdot C = \aleph_0 \cdot C = C^{\aleph_0} = C^n$
3. $2^C = C^C$

2. Lógica Proposicional I

2.1. Lenguaje

DEF: A **alfabeto**, A un conjunto, $A \neq \emptyset$.

DEF: Una **expresión** es una sucesión finita de elementos de A o la cadena vacía λ.

DEF: $A^* = \cup_{n \in \mathbb{N}} A_n$.

DEF: Un lenguaje Σ sobre A es $\Sigma \subseteq A^*, \Sigma \neq \emptyset$.

Proposición: A alfabeto, $E, F, G, H \in A^*, EF = GH, long(E) \geq long(G)$
$\Rightarrow \exists\, H' \in A^* / E = GH'$

2.2. Sintaxis

$A = VAR \cup \{(,)\} \cup \{\wedge, \vee, \rightarrow, \neg\}$

DEF: Fórmula

1. $VAR \subseteq F$
2. $\alpha \in F \Rightarrow \neg\alpha \in F$
3. $\alpha, \beta \in F \Rightarrow (\alpha \vee \beta), (\alpha \wedge \beta), (\alpha \rightarrow \beta) \in F$
4. $\alpha \in E^*$ es fórmula si se obtiene aplicando finitas veces 1) 2) y 3).

DEF: Cadena de Formación (c.f)
Una sucesión finita $X_1, X_2, ..., X_n$ de expresiones de A^* es una c.f si $X_i \in VAR$ o $\exists\, j < i / X_i = \neg X_j$ o $\exists\, k, j < i / X_i \in (X_k * X_j)$. Cada X_i se llama eslabón.

Teorema: $\alpha \in F \Leftrightarrow \exists\, X_1, ..., X_n = \alpha\ c.f$

DEF: Subcadena
Dada $X_1, ..., X_n$ c.f decimos que $X_{i1}, X_{i2}, ..., X_{ik}$ es subcadena si:

1. Si es c.f
2. $X_{ik} = X_n$
3. $1 \leq i_1 < i_2 < ... < i_k = n$

DEF: Una c.f es **minimal** si la única subcadena que tiene es ella misma.

DEF: $E \in A^*$ expresión.

- $c(E)$ = cantidad de conectivos que aparecen en E.
- $cb(E)$ = cantidad de conectivos binarios que aparecen en E.
- $peso(E) = p(E)$ = cantidad de '(' menos la cantidad de ')'.

Teorema

1. $p(\alpha) = 0$
2. Si $\bullet$ es un conectivo binario que aparece en $\alpha \Rightarrow$ la expresión de E a la izquierda de $\bullet$ en α verifica que tiene $p(E) > 0$.

DEF: Subfórmula

- $c(\alpha) = 0 \Rightarrow \alpha = P_j \in VAR \Rightarrow S(\alpha) = P_j$.
- $c(\alpha) > 0$
  1. $\alpha = \neg\beta \Rightarrow S(\alpha) = \{\alpha\} \cup S(\beta)$.
  2. $\alpha = (\beta_1 \bullet \beta_2) \Rightarrow S(\alpha) = \{\alpha\} \cup S(\beta_1) \cup S(\beta_2)$.

2.3. Semántica

DEF: Valuación

Una valuación es una función $v : FORM \to \{0, 1\}$ que verifica:

1. $v(\neg\alpha) = 1 - v(\alpha)$
2. $v(\alpha_1 \wedge \alpha_2) = min\{v(\alpha_1), v(\alpha_2)\}$
3. $v(\alpha_1 \vee \alpha_2) = max\{v(\alpha_1), v(\alpha_2)\}$
4. $v(\alpha_1 \to \alpha_2) = max\{1 - v(\alpha_1), v(\alpha_2)\}$

Teorema: Dada $f : VAR \to \{0, 1\}$, $\exists!$ val $v : FORM \to \{0, 1\}$ / v extiende a f ($v|_{VAR} = f$).
ó, se puede decir que: $v, w : FORM \to \{0, 1\}$ val / $v|_{VAR} = w|_{VAR} \Rightarrow v = w$

DEF: Clasificación semántica

1. α es tautología si $v(\alpha) = 1 \forall$ v val.
2. α es contradicción si $v(\alpha) = 0 \forall$ v val.
3. α es contigencia si $\exists$ v val / $v(\alpha) = 1$ y $\exists$ w val / $w(\alpha) = 0$

Teorema: $\alpha \in F$ / $var(\alpha) = \{p_j \in VAR$ / p_j aparece en $\alpha\}$
Si v, w son valuaciones tales que $v|_{VAR(\alpha)} = w|_{VAR(\alpha)} \Rightarrow v(\alpha) = w(\alpha)$

DEF: Equivalencia

Decimos que α es equivalente a β si $v(\alpha) = v(\beta)$ $\forall$ v *val*. $\alpha \equiv \beta$

DEF: Función booleana

Una función $f : \{0, 1\}^n \to \{0, 1\}$ se llama función booleana.

Teorema: *{f / f es función booleana}* $\overset{biy}{\to} FORM/\equiv$

3. Lógica Proposicional II

3.1. Satisfacibilidad y Consecuencias

DEF: Satisfacibilidad

1. $\Gamma \subseteq F$, decimos que Γ es satisfacible si $\exists$ v val / $v(\alpha) = 1\ \forall \alpha \in \Gamma$.
2. $\Gamma \subseteq F$, decimos que Γ es insatisfacible si $\nexists$ v val / $v(\Gamma) = 1$, es decir $\forall$ v val $\exists \alpha \in \Gamma$ / $v(\alpha) = 0$.

DEF: Consecuencia

1. Decimos que α es consecuencia de Γ si: $(v(\Gamma) = 1 \Rightarrow v(\alpha) = 1)\ \forall$ v val.
2. Decimos que $\alpha \notin C(\Gamma)$ si $\exists$ v val / $v(\Gamma) = 1$ y $v(\alpha) = 0$.

Teoremas y propiedades

1. $\alpha \in C(\Gamma) \Leftrightarrow \Gamma \cup \{\neg\alpha\}$ es insatisfacible.
2. $\Gamma = \{\gamma_1, ..., \gamma_n\}$: $\quad \alpha \in C(\Gamma) \Leftrightarrow ((\gamma_1 \wedge ... \wedge \gamma_n) \rightarrow \alpha)$ es tautología.
3. *Teorema de la deducción versión semántica:* $(\alpha \rightarrow \beta) \in C(\Gamma) \Leftrightarrow \beta \in C(\Gamma \cup \{\alpha\})$
4. $\Gamma \subset C(\Gamma)$
5. $C(C(\Gamma)) = C(\Gamma)$
6. $C(F) = F$
7. $\Gamma_1 \subset \Gamma_2 \Rightarrow C(\Gamma_1) \subset C(\Gamma_2)$
8. $C(\Gamma_1) \cup C(\Gamma_2) \subset C(\Gamma_1 \cup \Gamma_2)$
9. $C(\Gamma_1 \cap \Gamma_2) \subset C(\Gamma_1) \cap C(\Gamma_2)$
10. Si $C(\Gamma_1) = \Gamma_2$ y $C(\Gamma_2) = \Gamma_1 \Rightarrow \Gamma_1 = \Gamma_2$
11. Si Γ es satisfacible $\Rightarrow C(\Gamma)$ es satisfacible.

3.2. Independencia y base

DEF: Independencia

Γ es un conjunto de fórmulas independientes si $\forall \alpha \in \Gamma\ \alpha \notin C(\Gamma - \{\alpha\})$

DEF: Base

1. Γ es independiente.
2. Si Σ es un conjunto independiente de fórmulas tales que $\Gamma \subset \Sigma \Rightarrow \Gamma = \Sigma$

Propiedades

1. Si Γ es independiente y $\Sigma \subset \Gamma \Rightarrow \Sigma$ es independiente.
2. Si Γ contiene una tautología $\Rightarrow \Gamma$ es dependiente.
3. $C(\Gamma)$ es dependiente.

3.3. Compacidad f.s.

DEF: Finitamente satisfacible (f.s.) Γ es f.s. si todo subconjunto finito de Γ es satisfacible
LEMA: Sea Γ f.s. $p_i \in \text{VAR} \Rightarrow \Gamma \cup \{p_i\}$ es f.s. ó $\Gamma \cup \{\neg p_i\}$

Teorema de Compacidad: Γ es satisfacible $\Leftrightarrow \Gamma$ es f.s.

Proposición: Son equivalentes:

1. Teorema de compacidad
2. $\alpha \in C(\Gamma) \Rightarrow \exists\ \Gamma' \subseteq \Gamma$, Γ' finito / $\alpha \in C(\Gamma')$

3.4. Teoría axiomática

Proposición: Los axiomas son tautologías.
AX1: $(\alpha \to (\beta \to \alpha))$
AX2: $((\alpha \to (\beta \to \gamma)) \to ((\alpha \to \beta) \to (\alpha \to \gamma)))$
AX3: $((\neg\alpha \to \neg\beta) \to ((\neg\alpha \to \beta) \to \alpha))$

Regla Modus Ponens

1. $(\alpha \to \beta)$
2. α
3. β

DEF: Prueba
Una prueba para $\alpha \in F$ es una sucesión finita de fórmulas $\alpha_1, \alpha_2, ..., \alpha_k$ /

1. $\alpha_k = \alpha$
2. α_j es un axioma o se obtiene aplicando MP a α_i y α_t con $i, t < j$. Es decir $\alpha_t = (\alpha_i \to \alpha_j)$

DEF: Demostrable
α es demostrable si existe una prueba de α. En este caso α se llama teorema.

Teorema: α es demostrale $\to \alpha$ es tautología.

DEF: Deducibilidad

Decimos que α se deduce de Γ si existe una sucesión finita de fórmulas $\alpha_1, \alpha_2, ..., \alpha_n = \alpha$ (prueba a partir de Γ) que verifica que:

1. $\alpha_i \in \Gamma$ (α_i es un dato)
2. ó α_i es un axioma
3. ó α_i se obtiene aplicando MP a $\alpha_j = (\alpha_k \to \alpha_i)$ y α_k, $j, k < i$, $1 \leq i \leq n$

Teorema de la deducción versión axiomática: $\Gamma \vdash (\alpha \to \beta) \Leftrightarrow \Gamma \cup \{\alpha\} \vdash \beta$

Teorema de correctitud: $\Gamma \vdash \alpha \Rightarrow \alpha \in C(\Gamma)$

DEF: Consistente

Γ es consistente si $\nexists\ \varphi \in F\ /\ \Gamma \vdash \varphi$ y $\Gamma \vdash \neg\varphi$

Y Γ es inconsistente si $\exists\ \varphi \in F\ /\ \Gamma \vdash \varphi$ y $\Gamma \vdash \neg\varphi$

DEF: Maximal consistente

Γ es m.c si Γ es consistente y $\forall\ \varphi \in F$ tenemos que: $\varphi \in \Gamma$ ó $\exists\ \psi \in F\ /\ \Gamma \cup \{\varphi\} \vdash \psi$ y $\Gamma \cup \{\varphi\} \vdash \neg\psi$

Teoremas y propiedades

1. Γ satisfacible $\Rightarrow$ Γ consistente.
2. *Lema de Lindembaum:* Γ es consistente $\Rightarrow \exists\ \Gamma'$ m.c $/\ \Gamma \subseteq \Gamma'$
3. $\Gamma \cup \{\neg\varphi\}$ es inconsistente $\Leftrightarrow \Gamma \vdash \varphi$
4. $\Gamma \cup \{\varphi\}$ es inconsistente $\Leftrightarrow \Gamma \vdash \neg\varphi$
5. Γ m.c $\Rightarrow \varphi \in \Gamma'$ ó $\neg\varphi \in \Gamma'$ *(ó excluyente).*
6. Γ m.c $\Rightarrow (\Gamma \vdash \varphi \Leftrightarrow \varphi \in \Gamma)$

Teorema: Γ consistente $\Rightarrow$ Γ satisfacible.

Teorema de Completitud

$\alpha \in C(\Gamma) \Rightarrow \Gamma \vdash \alpha$

DEF: Un sistema axiomático S es consistente si $\nexists\ \varphi \in F\ /\ \vdash_s \varphi$ y $\vdash_s \neg\varphi$

4. Lógica de Primer Orden I

4.1. Lenguaje

DEF: Alfabeto

$A = VAR \cup CONECTIVOS \cup \{(,)\} \cup \mathcal{F} \cup \mathcal{C} \cup \mathcal{P}$

$VAR = \{X_0, X_1, X_2, ...\}$

$CONECTIVOS = \{\wedge, \vee, \rightarrow, \neg\} \cup \{\forall, \exists\}$

$\mathcal{F}$: símbolos de función. $\mathcal{F}$ podría ser $\emptyset$.

$\mathcal{C}$: símbolos de constantes. $\mathcal{C}$ podría ser $\emptyset$.

$\mathcal{P}$: símbolos de predicados. $\mathcal{P}$ no puede ser $\emptyset$.

DEF: Término a partir de un alfabeto A

Es una expresión en A^* que se obtiene aplicando finitas veces las siguientes reglas:

1. Toda variable es un término.
2. Toda constante es un término.
3. Si $t_1, t_2, ..., t_k$ son términos y $f^k \in \mathcal{F} \Rightarrow f^k(t_1, t_2, ..., t_k)$ es un término.
4. Cualquier expresión de A^* que se obtiene aplicando finitas veces 1. 2. 3. es un término.

DEF: Fórmula a partir de un alfabeto A

1. $t_1, t_2, ..., t_k \in TERM$ y $p^k \in \mathcal{P} \Rightarrow P^k(t_1, t_2, ..., t_k)$ es una fórmula (Atómica).
2. $\alpha \in FORM \Rightarrow \neg\alpha \in FORM$
3. $\alpha, \beta \in FORM \Rightarrow (\alpha * \beta) \in FORM$
4. $\alpha \in FORM$, $x \in VAR \Rightarrow \forall x\ \alpha \in FORM$
5. $\alpha \in FORM$, $x \in VAR \Rightarrow \exists x\ \alpha \in FORM$
6. Cualquier expresión de A^* que se obtiene aplicando finitas veces 1. 2. 3. 4. 5. $\in FORM$

DEF: $\mathcal{L}$ de 1^{er} orden

Dado un alfabeto A, $\mathcal{L} \subseteq A^*$ es un lenguaje de 1^{er} orden si $\mathcal{L} = TERM \cup FORM$.

DEF: Un término se llama **cerrado** si no tiene variables

DEF: Variables libres y ligadas

1. Una aparición de una variable x en una fórmula está **ligada** si es alcanza por un cuantificador. En caso contrario decimos que la aparición es **libre**.
2. Una variable es **libre** en una fórmula si todas sus apariciones son libres.
3. Una variable está **ligada** en una fórmula si todas sus apariciones están ligadas.
4. Una fórmula se llamaba **enunciado** si todas sus variables están ligadas.

4.2. Semántica

DEF: Interpretación

Dado un alfabeto A y un lenguaje $\mathcal{L}$ de 1^{er} orden, una interpretación $\mathcal{I}$ de un lenguaje $\mathcal{L}$ consiste en:

1. $\mathcal{U} \neq \emptyset$. $\mathcal{U}$ universo.
2. $c \in \mathcal{C} \Rightarrow c$ se interpreta como $c_{\mathcal{I}} \in \mathcal{U}$
3. $f^k \in \mathcal{F} \Rightarrow f^k$ se interpreta como una función $f^k_{\mathcal{I}} : \mathcal{U}^k \to \mathcal{U}$
4. $p^k \in \mathcal{P} \Rightarrow p^k$ se interpreta como una relación k-aria en $\mathcal{U}$, es decir $p^k_{\mathcal{I}} \subseteq \mathcal{U}^k$

DEF: Valuación

Dado un lenguaje de 1^{er} orden y una interpretación $\mathcal{I}$ de $\mathcal{L}$.

Una función $v : VAR \to \mathcal{U}_{\mathcal{I}}$ se llama valuación.

DEF: Valuación extendida

Sea $\overline{v} : TERM \to \mathcal{U}_{\mathcal{I}}$ una extensión de v que verifica

1. $\overline{v}(x) = v(x)$
2. $\overline{v}(c) = c_{\mathcal{I}}$
3. $\overline{v}(f^k(t_1, ..., t_k)) = f^k_{\mathcal{I}}(\overline{v}(t_1), ..., \overline{v}(t_k))$

DEF: Valuación modificada

$$v_{x_i=a} : VAR \to \mathcal{U}_{\mathcal{I}} \quad / \quad v_{x_i=a} = \begin{cases} v(x) & \text{si} \quad x \neq x_i \\ a & \text{si} \quad x = x_i \end{cases}$$

Valor de verdad de las fórmulas

1. $\alpha = p^k(t_1, ..., t_k) \quad V_{\mathcal{I},v}(\alpha) = 1 \Leftrightarrow (\overline{v}(t_1), ..., \overline{v}(t_k)) \in p^k_{\mathcal{I}}$
2. $\alpha = \neg\beta \quad V_{\mathcal{I},v}(\alpha) = 1 - V_{\mathcal{I},v}(\beta)$
3. $\alpha = (\beta_1 \wedge \beta_2) \quad V_{\mathcal{I},v}(\alpha) = min\{V_{\mathcal{I},v}(\beta_1), V_{\mathcal{I},v}(\beta_2)\}$
4. $\alpha = (\beta_1 \vee \beta_2) \quad V_{\mathcal{I},v}(\alpha) = max\{V_{\mathcal{I},v}(\beta_1), V_{\mathcal{I},v}(\beta_2)\}$
5. $\alpha = (\beta_1 \to \beta_2) \quad V_{\mathcal{I},v}(\alpha) = max\{1 - V_{\mathcal{I},v}(\beta_1), V_{\mathcal{I},v}(\beta_2)\}$
6. $\alpha = \forall x\ \beta \quad V_{\mathcal{I},v}(\alpha) = 1 \Leftrightarrow V_{\mathcal{I},v_{x=a}}(\beta) = 1 \quad$ Para todo $a \in \mathcal{U}_{\mathcal{I}}$
7. $\alpha = \exists x\ \beta \quad V_{\mathcal{I},v}(\alpha) = 1 \Leftrightarrow V_{\mathcal{I},v_{x=a}}(\beta) = 1 \quad$ Para algún $a \in \mathcal{U}_{\mathcal{I}}$

1. Decimos que α es **satisfacible** si $\exists\ \mathcal{I}$ y v / $V_{\mathcal{I},v}(\alpha) = 1 \quad$ Notación: $\mathcal{I} \models \alpha[v]$
2. Decimos que α es **verdadera** o **válida** en una interpretación $\mathcal{I}$ si $V_{\mathcal{I},v}(\alpha) = 1\ \forall v$
   Notación: $\mathcal{I} \models \alpha$
3. Decimos que α es **universalmente válida** si $V_{\mathcal{I},v}(\alpha) = 1\ \forall\mathcal{I}\ \forall v \quad$ Notación: $\models \alpha$

5. Lógica de Primer Orden II

5.1. Expresables

DEF: Conjunto expresable

A es expresable si $\exists\ \alpha \in FORM$ con una única variable libre y todas las demás variables ligadas tal que:

$$V_{\mathcal{I},v_{x=a}}(\alpha(x)) = 1 \Leftrightarrow x \in A \quad \text{ó dicho de otra forma:} \begin{cases} Si \quad x \in A \Rightarrow V_{\mathcal{I},v_{x=a}}(\alpha(x)) = 1 \\ Si \quad x \notin A \Rightarrow V_{\mathcal{I},v_{x=a}}(\alpha(x)) = 0 \end{cases}$$

DEF: Elemento distinguible

Decimos que a es distinguible si $\{a\}$ es expresable.

5.2. Isomorfismo

DEF: Isomorfismo

Una función $F : \mathcal{U}_{\mathcal{I}_1} \to \mathcal{U}_{\mathcal{I}_2}$ se llama isomorfismo si:

1. F es biyectiva.
2. $c \in \mathcal{C}. \quad c_{\mathcal{I}_1} \in \mathcal{U}_{\mathcal{I}_1}$ y $c_{\mathcal{I}_2} \in \mathcal{U}_{\mathcal{I}_2} \Rightarrow f(c_{\mathcal{I}_1}) = c_{\mathcal{I}_2}$.
3. $f^k \in \mathcal{F} \quad F(f^k_{\mathcal{I}_1}(u_1, ..., u_k)) = f^k_{\mathcal{I}_2}(F(u_1), ..., F(u_k))$
4. $p^k \in \mathcal{P} \quad (u_1, ..., u_k) \in p^k_{\mathcal{I}_1} \Leftrightarrow (F(u_1), ..., F(u_k)) \in p^k_{\mathcal{I}_2}$

Isomorfismo: Lema, Teorema y Corolarios

Sea $\mathcal{L}$ de 1° orden. $\mathcal{I}_1$ y $\mathcal{I}_2$ interpretaciones de $\mathcal{L}$. Sea $h : \mathcal{U}_{\mathcal{I}_1} \to \mathcal{U}_{\mathcal{I}_2}$ iso. Sea v una val en $\mathcal{I}_1$.

Lema: $\overline{h \circ v} = h \circ \overline{v}$

Teorema: $\mathcal{I}_1 \models \alpha[v] \Leftrightarrow \mathcal{I}_2 \models \alpha[h \circ v]$

Corolario 1: Si $\mathcal{I}_1 \models \alpha \Leftrightarrow \mathcal{I}_2 \models \alpha \quad \to$ Sirve para probar que $\mathcal{I}_1 \not\approx \mathcal{I}_2$.

Corolario 2: Si $a \in \mathcal{U}$ distinguible $\Rightarrow h(a) = a \quad \to$ Sirve para probar no distinguibles.

Corolario 3: A expresable en $\mathcal{I} \Rightarrow h(A) \subseteq A \quad \to$ Sirve para probar no expresables.

Corolario 4: Si $a \in \mathcal{U}_{\mathcal{I}_1}$ es distinguible en $\mathcal{I}_1 \Rightarrow h(a) \in \mathcal{U}_{\mathcal{I}_2}$ es distinguible en $\mathcal{I}_2$

$\to$ Sirve para armar un iso o para probar que $\mathcal{I}_1 \not\approx \mathcal{I}_2$.

6. Lenguaje S

Tres tipos de variables:

1. **De entrada:** $X_1, X_2, ...$
2. **De salida:** Y *inicializada en cero.*
3. **Auxiliares:** $Z_1, Z_2, ...$ *inicializadas en cero.*

Etiquetas:

$A_1, B_1, C_1, D_1, E_1, A_2, ...$

Tres tipos de instrucciones:

1. **Tipo I:** $V \leftarrow V + 1$
2. **Tipo II:** $V \leftarrow V - 1$
3. **Tipo III:** $IF\ V \neq 0\ GOTO\ L$

DEF: Programa
Es una lista finita de instrucciones $I_1, I_2, ..., I_n$ que se escribe una debajo de la otra

DEF: Macro
Es una pseudoinstrucción que representa un segmento de programa. Cada vez que en un programa aparece una macro hay que reemplazarla por el segmento de programa que representa, dicho segmento de programa se denomina expansión de la macro.

Macros conocidas

1. Salto incondicional: $GOTO\ L$
2. Asignación de cero: $V \leftarrow 0$
3. Asignación de variables: $V \leftarrow X_1$
4. Salto con condicion cero: $IF\ V = 0\ GOTO\ L$

DEF: Estado
Un estado de un programa P es una lista finita de igualdades de la forma $V = m$. Donde V es una variable y $m \in \mathbb{N}$. Hay una única igualdad para cada variable que aparece en P. Deben aparecer todas las variables que aparecen en P

DEF: Snapshot

Supongamos que un programa P tiene longitud n, es decir, tiene n instrucciones $I_1, I_2, ..., I_n$. Para un estado σ de P y un $i \in \{1, 2, ..., n, n+1\}$ tenemos que el par (i, σ) es un snapshot de P la cual se llama terminal si $i = n + 1$.

i dice a qué instrucción apunta antes de ser ejecutada. En σ están los valores de las variables antes de ejecutar la instrucción i.

Dada una foto (i, σ) no terminal se define la foto sucesora (j, τ) de la siguiente manera:

1. Si la i-ésima instrucción es $V \leftarrow V + 1 \Rightarrow j = i + 1$, $\tau = \sigma$
   salvo que si $V = m \in \sigma \Rightarrow V = m + 1 \in \tau$
2. Si la i-ésima instrucción es $V \leftarrow V - 1 \Rightarrow j = i + 1$, $\tau = \sigma$
   salvo que si $V = m > 0 \in \sigma \Rightarrow V = m - 1 \in \tau$ y si $V = 0 \in \sigma \Rightarrow V = 0 \in \tau$
3. Si la i-ésima instrucción es $IF\ V \neq 0\ GOTO\ L \Rightarrow \tau = \sigma$, $j = \begin{cases} i+1 & si \quad V = 0 \in \sigma \\ k & si \quad V \neq 0 \in \sigma \end{cases}$

   Siendo $k = \begin{cases} min\{t / I_t \text{ está etiquetado con L}\} & \text{Si alguna instrucción está etiquetada con L} \\ n+1 & \text{Si ninguna instrucción está etiquetada con L} \end{cases}$

DEF: Cómputo

Un cómputo de un programa P a partir de una snapshot $d_1 = (i, \sigma)$ es una lista finita $d_1, d_2, ..., d_k$ de fotos siendo d_{j+1} la foto sucesora de d_j, y d_k la foto terminal.

DEF: Estado Inicial

Sea P un programa y $u_1, u_2, ..., u_m \in \mathbb{N}$. El estado inicial de P para dichos valores es:
$\sigma = (X_1 = u_1, X_2 = u_2, ..., X_m = u_m, X_j = 0\ (j > m), Z_j = 0, Y = 0)$ si Z_j aparece en P.
$d_1 = (1, \sigma)$ la foto inicial.

DEF: Cómputo a partir de un estado inicial

Sea P un programa, $u_1, u_2, ..., u_m \in \mathbb{N}$ y σ_1 el estado inicial para ellos. Existen dos posibilidades:

1. Que el programa termine ante dichas entradas $\Rightarrow$ Existe un cómputo de P a partir de $d_1 = (1, \sigma)$, es decir existe $d_1, ..., d_k$ / d_k terminal. $\quad \Psi_p^m(u_1, ..., u_m) = Y$ en d_k
2. Que el programa no termine ante dichas entradas $\Rightarrow$ No existe un cómputo de P a partir de $d_1 = (1, \sigma)$. $\quad \Psi_p^m(u_1, ..., u_m) = \uparrow$

DEF: Función computable

1. Una función $f : \mathbb{N}^k \to \mathbb{N}$ es **parcialmente computable** si existe un programa P / $f = \Psi_p^k$
   Si $\vec{x} \in Domf \Rightarrow f(\vec{x}) = \Psi_p^k(\vec{x})$ $\quad$ y $\quad$ Si $\vec{x} \notin Domf \Rightarrow \Psi_p^k(\vec{x}) = \uparrow$
2. Una función $f : \mathbb{N}^k \to \mathbb{N}$ es **computable** si es parcialmente computable y **total** $(Domf = \mathbb{N}^k)$

7. Funciones RP

7.1. Primeras definiciones y teoremas

Teorema: La composición de funciones computables es computable.

DEF: Esquema Recursivo Tipo I (ERI)

Sea $h : \mathbb{N} \to \mathbb{N}$ y $g : \mathbb{N}^2 \to \mathbb{N}$. Decimos que h se obtiene a partir de g por un ERI si se puede escribir de la siguiente forma: $\begin{cases} h(0) = k \in \mathbb{N} \\ h(n+1) = g(n, h(n)) \end{cases}$

Teorema: Si h se obtiene a partir de g por un ERI y g es computable $\Rightarrow$ h es computable.

DEF: Esquema Recursivo Tipo II (ERII)

Sea $h : \mathbb{N}^{n+1} \to \mathbb{N}$, $g : \mathbb{N}^{n+2} \to \mathbb{N}$ y $q : \mathbb{N} \to \mathbb{N}$. Decimos que h se obtiene a partir de g y q por un ERII si se puede escribir de la siguiente forma: $\begin{cases} h(x_1, ..., x_n, 0) = q(x_1, ..., x_n) \\ h(x_1, ..., x_n, y+1) = g(x_1, ..., x_n, y, h(x_1, ..., x_n, y)) \end{cases}$

Teorema: Si h se obtiene a partir de g y q por un ERII y g y q son computables $\Rightarrow$ h es computable.

DEF: Funciones iniciales

1. $cero : \mathbb{N} \to \mathbb{N}$ / $cero(x) = 0$
2. $suc : \mathbb{N} \to \mathbb{N}$ / $suc(x) = x + 1$
3. $\pi_j^n : \mathbb{N}^n \to \mathbb{N}$ / $\pi_j^n(x_1, ..., x_n) = x_j$

DEF: Recursiva Primitiva (RP)

Una función es RP si es inicial o se obtiene aplicando finitas operaciones válidas a las funciones iniciales. Siendo operaciones válidas: composición, ERI y ERII.

Teorema: Si $f : \mathbb{N}^k \to \mathbb{N}$ es RP $\Rightarrow$ f es computable.

Observaciones

1. Si f no es total $\Rightarrow$ f no es RP.
2. Existen funciones computables que no son RP, como la función de Ackermann (no vale la vuelta).

Teorema: Si f es composición de funciones RP, o se obtiene a partir de un ERI o un ERII a partir de funciones RP $\Rightarrow$ f es RP.

DEF: Predicado RP

Un predicado $P^k \subseteq \mathbb{N}^k$ de k variables es una relación k-aria de números naturales. Se asocia a P^k una función $C_{P^k} : \mathbb{N}^k \to \{0,1\}$ / $C_{P^k}(\vec{x}) = \begin{cases} 1 & \vec{x} \in P^k \\ 0 & \vec{x} \notin P^k \end{cases}$ (Función característica).

Decimos que "P^k es RP (computable)" si C_{P^k} es RP (computable).

Teorema: P^k y Q^k RP (computables) $\Rightarrow (P \wedge Q)$, $(P \vee Q)$, $(P \to Q)$ y $\neg P$ son RP (computables).

Teorema: Sean $g_1, ..., g_m, h : \mathbb{N}^n \to \mathbb{N}$ RP (computables), $P_1^n, ..., P_m^n$ RP (computables),

$P_i \cap P_j = \emptyset$ si $i \neq j$ y $f : \mathbb{N}^n \to \mathbb{N}$ / $f(\vec{x}) = \begin{cases} g_1(\vec{x}) & si \quad \vec{x} \in P_1^n \\ g_2(\vec{x}) & si \quad \vec{x} \in P_2^n \\ ... & si \quad ... \\ g_m(\vec{x}) & si \quad \vec{x} \in P_m^n \\ h(\vec{x}) & si \quad no \end{cases}$ $\Rightarrow f$ es RP (computable).

7.2. Funciones Acotadas

Teorema: Sea $f : \mathbb{N}^{n+1} \to \mathbb{N}$, RP (computable). Sean SA_f, $PA_f : \mathbb{N}^{n+1} \to \mathbb{N}$.

1. SUMA ACOT: $SA_f(\vec{x}, y) = \sum_{k=0}^{y} f(\vec{x}, k)$ es RP (computable).
2. PRODUCTORIA ACOT: $PA_f(\vec{x}, y) = \prod_{k=0}^{y} f(\vec{x}, k)$ es RP (computable).

Teorema: Sea P^{n+1} un predicado RP (computable). Sean EA_P, $UA_P : \mathbb{N}^{n+1} \to \{0,1\}$.

1. CUANT. EXISTENCIAL ACOT: $EA_P(\vec{x}, y) = \exists\, t \leq y\ C_P(\vec{x}, t)$ es RP (computable).
2. CUANT. UNIVERSAL ACOT: $UA_P(\vec{x}, y) = \forall\, t \leq y\ C_P(\vec{x}, t)$ es RP (computable).

Teorema: Sea P^{n+1} un predicado RP (computable). Sean EAE_P, $UAE_P : \mathbb{N}^{n+1} \to \{0,1\}$.

1. CUANT. EXISTENCIAL ACOT. EST: $EAE_P(\vec{x}, y) = \exists\, t < y\ C_P(\vec{x}, t)$ es RP (computable).
2. CUANT. UNIVERSAL ACOT. EST : $UAE_P(\vec{x}, y) = \forall\, t < y\ C_P(\vec{x}, t)$ es RP (computable).

Teorema: Sea P^{n+1} un predicado RP (computable). Sean ma_P, $MA_P : \mathbb{N}^{n+1} \to \mathbb{N}$.

1. MÍNIMO ACOT: $ma_P(\vec{x}, y) = \begin{cases} min\ t \leq y\ C_P(\vec{x}, t) & si\ \exists\, t \leq y\ /\ C_P(\vec{x}, t) = 1 \\ 0 & si\ no \end{cases}$
2. MÁXIMO ACOT: $MA_P(\vec{x}, y) = \begin{cases} max\ t \leq y\ C_P(\vec{x}, t) & si\ \exists\, t \leq y\ /\ C_P(\vec{x}, t) = 1 \\ 0 & si\ no \end{cases}$

7.3. Funciones RP populares

1. *Funciones Iniciales: cero, suc y* π_j^n
2. $f : \mathbb{N} \to \mathbb{N} \ / \ f(x) = x!$
3. $suma : \mathbb{N}^2 \to \mathbb{N} \ / \ suma(x, y) = x + y$
4. $prod : \mathbb{N}^2 \to \mathbb{N} \ / \ prod(x, y) = xy$
5. $pot : \mathbb{N}^2 \to \mathbb{N} \ / \ pot(x, y) = (x + 1)^y$
6. $pred : \mathbb{N} \to \mathbb{N} \ / \ pred(x) = \begin{cases} x - 1 & x \neq 0 \\ 0 & x = 0 \end{cases}$
7. $rt : \mathbb{N}^2 \to \mathbb{N} \ / \ rt(x, y) = \begin{cases} x - y & x \geq y \\ 0 & x < y \end{cases}$
8. $dist : \mathbb{N}^2 \to \mathbb{N} \ / \ dist(x, y) = |x - y|$
9. $\alpha : \mathbb{N} \to \mathbb{N} \ / \ \alpha(x) = \begin{cases} 1 & x = 0 \\ 0 & x \neq 0 \end{cases}$
10. $h_k : \mathbb{N} \to \mathbb{N} \ / \ h_k(x) = k$
11. $impar : \mathbb{N} \to \mathbb{N} \ / \ impar(x) = \begin{cases} 0 & x \ par \\ 1 & x \ impar \end{cases}$
12. $eq : \mathbb{N}^2 \to \mathbb{N} \ / \ eq(x, y) = \begin{cases} 0 & x \neq y \\ 1 & x = y \end{cases}$
13. $max : \mathbb{N}^2 \to \mathbb{N} \ / \ max(x, y) = \begin{cases} x & x \geq y \\ y & x < y \end{cases}$
14. $I_f : \mathbb{N}^2 \to \mathbb{N} \ / \ I_f(n, x) = f^n(x)$
15. $menor : \mathbb{N}^2 \to \mathbb{N} \ / \ menor(x, y) = \begin{cases} 0 & x \geq y \\ 1 & x < y \end{cases}$
16. $coc : \mathbb{N}^2 \to \mathbb{N} \ / \ coc(x, y) = \begin{cases} \lfloor \frac{x}{y} \rfloor & y \neq 0 \\ 0 & y = 0 \end{cases}$
17. $resto : \mathbb{N}^2 \to \mathbb{N} \ / \ resto(x, y) = \begin{cases} r_y(x) & y \neq 0 \\ 0 & y = 0 \end{cases}$
18. $div : \mathbb{N}^2 \to \mathbb{N} \ / \ div(x, y) = \begin{cases} 1 & y \mid x \\ 0 & y \nmid x \end{cases}$
19. $primo : \mathbb{N} \to \mathbb{N} \ / \ primo(x) = \begin{cases} 1 & x \ primo \\ 0 & si \ no \end{cases}$
20. $p : \mathbb{N} \to \mathbb{N} \ / \ p(x) = \begin{cases} x\text{-}\acute{e}simo \ primo & x \neq 0 \\ 1 & x = 0 \end{cases}$
21. $dig_x : \mathbb{N} \to \mathbb{N} \ / \ dig_x(n) = n\text{-}\acute{e}simo \ d\acute{i}gito \ despu\acute{e}s \ de \ la \ coma, \ con \ dig_x(0) = 1$

8. Computabilidad - Halt

Teorema: Sea $<,>: \mathbb{N}^2 \to \mathbb{N} \ / \ <x,y>= 2^x(2y+1)-1$. Entonces $<,>$ es biyectiva.

Teorema: Sea $l: \mathbb{N} \to \mathbb{N} \ / \ l(z)=x \quad$ y $\quad r: \mathbb{N} \to \mathbb{N} \ / \ r(z)=y \quad / \quad z=<x,y>$.
Entonces l, r y $<,>$ son RP.

DEF: Numeración de Gödel
Sea $k \in \mathbb{N}$. Para cada k se define una función: $[\ ,...,\ ]: \mathbb{N}^{k+1} \to \mathbb{N} \ / \ [(a_0,...,a_k)] = p_0^{a_0}...p_k^{a_k}$
donde p_i representa el $(i+1)$-primo.

Teorema: Las funciones que calculan los números de Gödel son inyectivas, RP y no sobreyectivas.

DEF: Indicadores de Números de Gödel

1. $\cdot[\cdot]: \mathbb{N}^2 \to \mathbb{N} \ / \ x[i] = V_{p_i}(x)$
2. $|\cdot|: \mathbb{N} \to \mathbb{N} \ / \ |n| = \begin{cases} long(n) & x \neq 0 \\ 0 & x=0 \end{cases}$

Teorema: Las funciones antes definidas son RP.

8.1. Codificación de Programas

Tenemos como objetivo asignarle un $n \in \mathbb{N}$ a cada programa en lenguaje S, se define $\#P \in \mathbb{N} \ /$

1. Si dos programas P y P' son distintos $\Rightarrow \#P \neq \#P'$.
2. Dado $n \in \mathbb{N} \Rightarrow \exists$ un programa en S $/ \ \#P = n$.

Obs: La última instrucción no puede ser $Y \leftarrow Y$ excepto que sea la única.

Listas

1. **Variables:** $Y, X_1, Z_1, X_2, Z_2, ...$
2. **Etiquetas:** $A_1, B_1, C_1, D_1, E_1, A_2, B_2, ...$
3. **Instrucciones:** $\begin{cases} V \leftarrow V+1 \\ V \leftarrow V-1 \\ IF\ V \neq 0\ GOTO\ L \\ V \leftarrow V \end{cases}$

Codificar una Instrucción: Sea I una instrucción.
Definimos $\#: \{Instrucciones\} \to \mathbb{N} \ / \ \#I =< a,<b,c\check{}>$ biyectiva.

$\mathbf{a} = \begin{cases} 0 & \text{Si I no tiene etiqueta.} \\ \#L & \text{Si tiene etiqueta, siendo \#L la posición de la etiqueta en la lista 2.} \end{cases}$

$$\mathbf{b} = \begin{cases} 0 & I = V \to V \\ 1 & I = V \to V + 1 \\ 2 & I = V \to V - 1 \\ \#L + 2 & I = IF\ V \neq 0\ GOTO\ L \end{cases}$$

$\mathbf{c} = \#V - 1$, siendo $\#V$ la posición de la variable en la lista 1.

Codificar un Programa: Sea P un programa con $I_1, I_2, ..., I_k$ instrucciones.
Definimos $\# : \{Programas\} \to \mathbb{N}\ /\ \#P_{I_1,...,I_k} = [(\#I_1, ..., \#I_k)]_k - 1$ biyectiva.

8.2. Funciones no computables

Teorema: Existen funciones no computables.

DEF: Problema de la parada (Halting)

$$Halt : \mathbb{N}^2 \to \mathbb{N}\ /\ Halt(x, y) = \begin{cases} 1 & \text{Si el programa de código } y \text{ termina ante la entrada } x \\ 0 & \text{Si no (es decir } \Psi_P(x) = \uparrow\ /\ \#P = y) \end{cases}$$

Teorema: Halt es no computable.

Tesis de Church-Turing

En teoría de la computabilidad, la tesis de Church-Turing formula hipotéticamente la equivalencia entre los conceptos de función computable y máquina de Turing, que expresado en lenguaje corriente vendría a ser "todo algoritmo es equivalente a una máquina de Turing". No es un teorema matemático, es una afirmación formalmente indemostrable que, no obstante, tiene una aceptación prácticamente universal.

DEF: Programas Universales
Para cada $n > 0$ se define $\Phi^n : \mathbb{N}^{n+1} \to \mathbb{N}\ /\ \Phi^n(x_1, ..., x_n, e) = \Psi_P^n(x_1, ..., x_n)\ /\ \#P = e$

Teorema: Φ^n es parcialmente computable $\forall\ n > 0$.