Términos
Lógica Computacional 2023-I, Nota de clase 5

Favio Ezequiel Miranda Perea Araceli Liliana Reyes Cabello
Lourdes Del Carmen González Huesca Pilar Selene Linares Arévalo

26 de septiembre de 2022
Facultad de Ciencias UNAM

1. Introducción

Hasta ahora hemos estudiado la lógica proposicional donde los átomos son fórmulas sin estructura ni contenido alguno. Sin embargo en la realidad, estos átomos corresponden a información precisa y particular acerca de algún dominio de interés específico. Por ejemplo la fórmula $p \to q$ al observarla mediante un microscopio lógico, llamemoslo logiscopio, podría verse como:

$$\texttt{even}(n) = true \to n = 2 \cdot k$$

Es decir, al ver bajo el logiscopio notamos que p tiene en realidad la forma $\texttt{even}(n) = true$ y a su vez q es $n = 2 \cdot k$

Más aún, si utilizamos un objetivo de mayor potencia en nuestro logiscopio podremos observar una estructura más fina de la fórmula, a saber:

$$\forall n \ (\texttt{even}(n) = true \to \exists k \ (n = 2 \cdot k) \ )$$

Observemos que aquí estamos hablando de una clase particular de individuos a saber números (sospechamos que naturales o quizás enteros) y sus propiedades, en este caso la de ser par, implementada mediante una función (un programa funcional) `even` y la de ser un múltiplo de 2, implementada mediante la operación producto y una igualdad. Esta distinción entre individuos y sus propiedades es característica de la llamada lógica de primer orden, donde para hallar el significado de las fórmulas, que sigue siendo un valor booleano al final, requerimos ahora de un mundo particular llamado el universo de discurso, que en el caso del ejemplo anterior es el conjunto de números naturales o enteros pero que, dependiendo del caso particular, puede ser prácticamente cualquier cosa. De hecho en problemas de especificación en programación o desarrollo de software, este universo de discurso es muchas veces un fragmento del mundo real, pensemos por ejemplo en un software para control de vuelos, siendo en este caso el universo de discurso los aviones y las rutas de vuelo del mundo real. En lo que resta del curso vamos a estudiar distintos procedimientos para verificar argumentos con esta clase de fórmulas a la luz de los universos de discurso particulares. Pero antes tomemos un logiscopio aún más potente y apuntemos a lo que originalmente eran los átomos de nuestra fórmula. Lo que veremos es:

$$\texttt{even}(n) = true \text{ y } n = 2 \cdot k$$

Es decir, los átomos proposicionales son en realidad dos ecuaciones entre individuos, en este caso números y booleanos y nótese que para hallar el significado de la 'fórmula original primero debemos hallar el significado

de estas ecuaciones. De esto se encarga la lógica ecuacional que estudiaremos en poco tiempo. Por ejemplo, el tst de paridad se implementa mediante el siguiente programa funcional:

$$\begin{aligned} \texttt{even}\,0 &= \mathit{true} \\ \texttt{even}\,1 &= \mathit{false} \\ \texttt{even}\,(n+2) &= \texttt{even}\,n \end{aligned}$$

De manera que para calcular $\texttt{even}\,5$ podemos razonar ecuacionalmente como sigue:

$$\begin{aligned} \texttt{even}\,5 &= \texttt{even}\,3 \\ &= \texttt{even}\,1 \\ &= \texttt{false} \end{aligned}$$

La lógica que fundamenta y verifica la corrección del razonamiento anterior, el cual no es muy familiar desde la escuela secundaria o quizás primaria, es precisamente la lógica ecuacional.

Pero sigamos analizando nuestra fórmula ahora con el más potente objetivo del logiscopio, y lo que veremos ahora ya no serán las ecuaciones sino sus dos componentes, en este caso:

$$\texttt{even}(n) \quad \texttt{true} \quad n \quad 2 \cdot k$$

y aún más al acercar al logiscopio la primera y última expresión anterior veremos

$$n \quad 2 \quad k$$

Esta clase de bichos sintácticos nos son bien conocidos, son variables y constantes que representan a individuos particulares. A partir de estos podemos ir desenfocando las lentes del logiscopio hasta llegar nuevamente a la fórmula proposicional, pero por ahora sólo lo haremos para obtener los bichos sintácticos que representan individuos, que en nuestro ejemplo son:

- Las expresiones que nombran o denotan individuos de manera directa o simple, a saber variables y constantes:

$$n, \texttt{true}\,2,\ k, \ldots$$

- Las expresiones que nombran o denotan individuos de manera indirecta o sofisticada:

$$\texttt{even}(n),\ 2 \cdot k, \ldots$$

Para estudiar las expresiones que denotan individuos de manera general introducimos y estudiamos a continuación el concepto de término

1.1. Términos

Los términos son expresiones que denotan individuos de un dominio particular y podemos pensarlos como un lenguaje de programación genérico de propósito particular. Para poder definir instancias particulares de este lenguaje, por ejemplo para números, listas u otra estructura de datos pero también para cualquier otro propósito como puede ser un juego, un sistema de compras, nómina, etc..

Para poder definir una instancia particular de términos lo primero que necesitamos son los nombres (las palabras reservadas) para generarlos para esto necesitamos la siguiente

Definición 1 *Una signatura es un conjunto de símbolos* $\Sigma = \mathcal{C} \cup \mathcal{F}$ *donde*

- $\mathcal{C}$ *es un conjunto de símbolos de constante.*
- $\mathcal{F}$ *es un conjunto de símbolos de función*
- $\mathcal{C} \cap \mathcal{F} = \varnothing$ *es decir, los símbolos de constante y función son ajenos.*
- *Cada símbolo de función* $f \in \mathcal{F}$ *tiene asociado un número natural* n*, llamado su índice o aridad, el cual indica el número de argumentos que* f *requiere. La notación* $f^{(n)}$ *indica que el símbolo* f *tiene índice* n*.*

Los términos del lenguaje son aquellas expresiones que representarán objetos en la semántica y su definición es:

Definición 2 *Dada una signatura* $\Sigma = \mathcal{C} \cup \mathcal{F}$ *definimos los términos sobre* Σ *recursivamente como sigue:*

- *Los símbolos de constante* $c \in \mathcal{C}$ *son términos.*
- *Las variables* $x_1, \ldots, x_n, \ldots$ *son términos.*
- *Si* $f^{(n)} \in \mathcal{F}$ *es un símbolo de función y* $t_1, \ldots, t_n$ *son términos entonces* $f(t_1, \ldots, t_n)$ *es un término.*
- *Son todos.*

El conjunto de términos de una signatura dada se denota con TERM_Σ*, o simplemente* TERM *si la signatura es clara.*

En algunas ocasiones se encuentra una definición de términos que no incluye constantes. Esta omisión no existe en realidad puesto que las constantes se consideran casos particulares de símbolos de función de índice cero como hemos mencionado antes, es decir un símbolo de función que no recibe argumentos.

La definición anterior puede resumirse mediante una gramática para los términos, que usaremos rutinariamente:

$$t ::= x \mid c \mid f(t, \ldots, t)$$

1.2. Expresiones aritméticas binarias

Como un ejemplo particular de términos presentamos un lenguaje simple de expresiones aritméticas binarias, dadas por la siguiente gramática

$$B ::= N \mid B + B \mid B \times B$$

$$N ::= 0 \mid 1 \mid N0 \mid N1$$

Vamos a representar este lenguaje mediante dos instancias distintas del lenguaje de términos

Con signatura infinita. Tómese la signatura $\Sigma = \mathcal{C} \cup \mathcal{F}$ donde $\mathcal{C} = \{0,1\}^+$ y $\mathcal{F} = \{+^{(2)}, \times^{(2)}\}$. Nótese que el conjunto de símbolos de constante consta de todas las cadenas no vacías de ceros y unos, por lo que es infinito. Más aún, $\mathcal{C}$ corresponde a las cadenas generadas por el símbolo no terminal N en la gramática anterior[1]

Algunos ejemplos de términos con esta signatura son:

[1] En realidad el uso de la signatura infinita no corresponde directamente a la gramática pues para que así fuera necesitariamos una infinidad de reglas de producción, a saber

$$N ::= 0 \mid 1 \mid 00 \mid 01 \mid 10 \mid 11 \mid 000 \mid \ldots$$

Pero el uso de una infinidad de reglas de producción no está permitido en una gramática formal.

- Dado que $\mathcal{C}$ tiene a todas las cadenas de ceros y unos todo número binario b se representa mediante la constante b.
- La expresión aritmética binaria $10+(11\times 101)$ se representa mediante el término $+(10,\times(11,101))$. Nótese que en realidad no hay diferencia entre la expresión y el término, excepto por el uso de la notación infija.

Con signatura finita. Tómese la signatura $\Sigma = \mathcal{C} \cup \mathcal{F}$ donde $\mathcal{C} = \{0, 1\}$ y $\mathcal{F} = \{+^{(2)}, \times^{(2)}, snoc_0^{(1)}, snoc_1^{(1)}\}$. Algunos ejemplos de términos con esta signatura son:

- Las expresiones 0 y 1 se representan mediante las constantes 0 y 1
- Las expresiones numéricas con más de un bit se representan con las operaciones snoc. Por ejemplo 101 corresponde al término $snoc_1(snoc_0(1)$
- La expresión aritmética binaria $10 + (11 \times 101)$ se representa mediante el término

$$+(snoc_1(0), \times(snoc_1(1), snoc_1(snoc_0(1))))$$

Observe que la signatura finita corresponde de manera más fiel a la gramática de las expresiones aritméticas binarias pero por supuesto la infinita es más cómoda para usarse.

2. Inducción y Recursión para términos

Como es usual en toda estructura sintáctica definida recursivamente podemos podemos definir funciones sobre los términos guiándonos por su sintaxis mediante el siguiente principio:

Definición Recursiva para Términos Para definir una función $h : \mathsf{TERM}_\Sigma \to A$, basta definir h como sigue:

- Definir $h(x)$ para $x \in \mathsf{Var}$.
- Definir $h(c)$ para cada constante $c \in \mathcal{C}$.
- Suponiendo que $h(t_1), \ldots, h(t_n)$ están definidas, definir $h(f(t_1, \ldots, t_n))$ para cada símbolo de función $f \in \mathcal{F}$ de índice n.

Veamos un par de ejemplos:

Ejemplo 2.1 Definimos el conjunto de subtérminos de un término t recursivamente como sigue:

- $\mathsf{Subt}(x) = \{x\}$
- $\mathsf{Subt}(c) = \{c\}$
- $\mathsf{Subt}(f(t_1, \ldots, t_n)) = \{f(t_1, \ldots, t_n)\} \cup \mathsf{Subt}(t_1) \cup \ldots \cup \mathsf{Subt}(t_n)$

Ejemplo 2.2 Definimos recursivamente el número de símbolos de función en un término como sigue:

- $\mathsf{nf}(c) = 0$.
- $\mathsf{nf}(x) = 0$.

- $\mathsf{nf}(f(t_1, \ldots, t_n)) = 1 + \mathsf{nf}(t_1) + \ldots + \mathsf{nf}(t_n)$.

Para demostrar propiedades de las definiciones recursivas dirigidas por la sintaxis requerimos de un principio de inducción estructural.

Definición 3 (Principio de Inducción Estructural para TERM_Σ) *Sea $\mathcal{P}$ una propiedad acerca de términos. Para demostrar que $\mathcal{P}$ es válida para todos los términos, basta seguir los siguientes pasos:*

- *Base de la inducción: mostrar que*
  1. *Si $x \in \mathsf{Var}$ entonces $\mathcal{P}$ es válida para x.*
  2. *Si $c \in \mathcal{C}$ entonces $\mathcal{P}$ es válida para c.*
- *Hipótesis de inducción: suponer $\mathcal{P}$ para cualesquiera $t_1, \ldots, t_n \in \mathsf{TERM}_\Sigma$.*
- *Paso inductivo: usando la hipótesis de inducción mostrar que*

  Si $f \in \mathcal{F}$ es un símbolo de función de índice n entonces $f(t_1, \ldots, t_n)$ cumple $\mathcal{P}$.

Un ejemplo importante de definición recursiva es el de la aplicación de una sustitución a un término, que vemos a continuación.

3. Sustitución

En el caso de la lógica de proposiciones, la operación de sustitución $A[p := B]$ es una operación ternaria donde A es la fórmula y $[p := B]$ es la sustitución a ser aplicada. Sin embargo la sustitución no existe por si sola. En el caso de los términos y más adelante para las fórmulas, necesitamos definir y estudiar a las sustituciones por separado, lo cual hacemos a continuación.

Definición 4 *Una sustitución en un lenguaje de términos TERM_Σ es una función $\rho : \mathsf{Var} \to \mathsf{TERM}_\Sigma$ tal que $\rho(x) = x$ casi para todas las variables, es decir $\rho(x) \neq x$ únicamente para un número finito de variables digamos $x_1, \ldots, x_k$. En tal caso si $\rho(x_i) = s_i$ la sustitución puede denotarse mediante una colección finita de pares de la forma $\rho = [x_1 := s_1, x_2 := s_2, \ldots, x_k := s_k]$, donde*

- *$x_1, \ldots, x_k \in \mathsf{Var}$ son variables distintas.*
- *$s_1, \ldots, s_k \in \mathsf{TERM}_\Sigma$.*
- *$x_i \neq s_i$ para cada $1 \leq i \leq n$*

Alternativamente se puede usar la notación $\rho = [x_1, x_2, \ldots, x_k := s_1, s_2, \ldots, s_k]$ o también $\rho = [\vec{x} := \vec{s}]$ para abreviar la notación.

3.1. Aplicación de una sustitución a un término

La aplicación de una sustitución $\rho = [\vec{x} := \vec{s}]$ a un término t, denotada $t\rho$ o $t[\vec{x} := \vec{s}]$, se define como el término obtenido al reemplazar **simultáneamente** todas las presencias de x_i en t por s_i. Este proceso de define

recursivamente como sigue:

$$\begin{aligned} x[\vec{x} := \vec{s}] &= s_i \qquad \text{si } x \text{ es una de las } x_i \;\; 1 \leq i \leq k \\ z[\vec{x} := \vec{s}] &= z \qquad \text{si } z \neq x_i, \;\; 1 \leq i \leq k \\ c[\vec{x} := \vec{s}] &= c \qquad \text{si } c \in \mathcal{C} \\ f(t_1, \ldots, t_n)[\vec{x} := \vec{s}] &= f(t_1[\vec{x} := \vec{s}], \ldots, t_n[\vec{x} := \vec{s}]) \qquad \text{si } f^{(n)} \in \mathcal{F}. \end{aligned}$$

Obsérvese entonces que la aplicación de una sustitución a un término es simplemente una sustitución textual tal y como sucede en lógica de proposiciones.

3.2. Composición de sustituciones

Una operación de gran importancia es la composición de sustituciones que no es más que la composición de funciones la cual puede caracterizarse de una manera más simple para ser implementada de acuerdo a la siguiente

Proposición 1 *Sean* $\rho = [x_1 := t_1, \ \ldots, \ x_n := t_n]$ *y* $\tau = [y_1 := s_1, \ \ldots, \ y_m := s_m]$ *dos sustituciones. La composición, denotada* $\rho \circ \tau$ *o simplemente* $\rho\tau$*, es una sustitución definida por:*

$$\rho\tau = [z_1 := t_1\tau, \ \ldots, \ z_k := t_k\tau, \, w_1 := r_1, \ \ldots, w_l := r_l]$$

donde $k \leq n, l \leq m$*, las variables* z_i *son exactamente aquellas* x_p *tales que* $x_p \neq t_p\tau$*, y los pares* $w_j := r_j$ *son aquellos pares* $y_q := s_q$ *tales que* $y_q \notin \{x_1, \ldots, x_n\}$ *(es decir, son los pares cuya primera componente no es ninguna de* $x_1, \ldots, x_n$*).*

Demostración. Ejercicio ⊣

Proposición 2 *La composición de sustituciones es asociativa, es decir, si* σ, ρ *y* τ *son tres sustituciones entonces* $(\sigma\rho)\tau = \sigma(\rho\tau)$*.*

Demostración. Ejercicio ⊣

Ejemplo 3.1 Sean

$$\sigma = [x := f(y), \ y := w] \qquad \rho = [x := g(w), \ z := b] \qquad \tau = [y := b, \ w := f(c), \ v := w]$$

Entonces

$$\begin{aligned} \sigma\rho &= [x := f(y), \ y := w, \ z := b] \\ \rho\tau &= [x := g(f(c)), \ z := b, \ y := b, \ w := f(c), \ v := w] \end{aligned}$$

$$\begin{aligned} (\sigma\rho)\tau &= [x := f(b), \ y := f(c), \ z := b, \ w := f(c), \ v := w] \\ \sigma(\rho\tau) &= [x := f(b), \ y := f(c), \ z := b, \ w := f(c), \ v := w] \end{aligned}$$

4. Semántica de términos

Veamos ahora la semántica para los términos, ya hemos dicho que los términos son nombres para un mundo particular fijado de acuerdo a las necesidades particulares de cada problema. A continuación formalizamos esta idea.

Definición 5 *Un dominio o universo (de discurso) es un conjunto no vacío* $\mathcal{U}$

Definición 6 *Una interpretación o modelo para una signatura dada* Σ *es un par ordenado* $\mathcal{M} = \langle \mathcal{U}, \mathcal{I} \rangle$ *donde*

- $\mathcal{U}$ *es un universo de discurso.*
- $\mathcal{I}$ *es una función de interpretación cuyo dominio es* Σ *y cumple lo siguiente:*
  - *Si* $c \in \mathcal{C}$ *entonces* $\mathcal{I}(c) \in \mathcal{U}$.
  - *Si* $f^{(n)} \in \mathcal{F}$ *entonces* $\mathcal{I}(f)$ *es una función de* $\mathcal{U}^n$ *en* $\mathcal{U}$.

Es común usar la notación $c^{\mathcal{I}}$ y $f^{\mathcal{I}}$ en vez de $\mathcal{I}(c)$ e $\mathcal{I}(f)$ respectivamente. Con esta notación se tiene que, de acuerdo a la definición anterior, $c^{\mathcal{I}} \in \mathcal{U}$ y $f^{\mathcal{I}} : \mathcal{U}^n \to \mathcal{U}$.

Las siguientes definiciones dependen todas de una Σ-interpretación arbitraria pero fija $\mathcal{M} = \langle \mathcal{U}, \mathcal{I} \rangle$.

Definición 7 (Estado o Asignación) *Un estado, asignación o valuación de las variables es una función* $\sigma : \mathsf{Var} \to \mathcal{U}$.

Al igual que en la lógica proposicional los estados son necesarios para poder darle significado a las variables.

Definición 8 (Interpretación de Términos) *Sea* σ *un estado de las variables. Definimos la función de interpretación o significado de los términos con respecto a* σ, $\mathcal{I}_\sigma : \mathsf{TERM} \to \mathcal{U}$ *como sigue:*

$$\begin{aligned} \mathcal{I}_\sigma(x) &= \sigma(x) \\ \mathcal{I}_\sigma(c) &= c^{\mathcal{I}} \\ \mathcal{I}_\sigma\left(f(t_1, \ldots, t_n)\right) &= f^{\mathcal{I}}(\mathcal{I}_\sigma(t_1), \ldots, \mathcal{I}_\sigma(t_n)) \end{aligned}$$

Esta definición captura de manera general los ejemplos de evaluadores que vimos en clase. Por ejemplo, en el caso de un término funcional puede pensarse que la interpretación $f^{\mathcal{I}}$ es una implementación del procedimiento denotado por el símbolo de función f.

Veamos algunos ejemplos de interpretación de términos:

Ejemplo 4.1 *Sea* $\Sigma = \{0, s^{(1)}, +^{(2)}, \times^{(2)}\}$. $\mathcal{M} = \langle \mathbb{N}, \mathcal{I} \rangle$ *donde* $\mathcal{I}(0) = 0$, $\mathcal{I}(s) = s^{\mathcal{I}}$, $+^{\mathcal{I}}$ *es la suma de naturales,* $\times^{\mathcal{I}}$ *es el producto de naturales. y* $s^{\mathcal{I}}(n) = n +^{\mathcal{I}} 1$.

- *Si* $\sigma(x) = 4$, $\sigma(z) = 7$, $\sigma(y) = 11$ *entonces*
  - $\mathcal{I}_\sigma(s(s(0)) \times (x + s(y))) = 2 \times^{\mathcal{I}} (4 +^{\mathcal{I}} 12) = 32$
  - $\mathcal{I}_\sigma(z \times s(s(s(x))) + s(y)) = 7 \times^{\mathcal{I}} 7 +^{\mathcal{I}} (11 + 1) = 61$
- *Si* $\sigma(x) = 3$, $\sigma(z) = 80$, $\sigma(y) = 15$ *entonces*

- $\mathcal{I}_\sigma(s\Big(s(z) \times (x + s(y))\Big)) = \Big(81 \times^{\mathcal{I}} (3 + 16)\Big) +^{\mathcal{I}} 1 = 1540$
- $\mathcal{I}_\sigma(s(x + s(s(0) + x)) + s(z \times s(y))) = s(3 +^{\mathcal{I}} s(1 +^{\mathcal{I}} 3)) +^{\mathcal{I}} s(80 \times^{\mathcal{I}} 16) = 4 +^{\mathcal{I}} 1281 = 1285$

Ejemplo 4.2 *Sea* $\Sigma = \{\bar{0}, \bar{1}, \bar{2}, \ldots, s^{(1)}, +^{(2)}\}$. $\mathcal{M} = \langle List(\mathbb{N}), \mathcal{I}\rangle$ *donde* $List(\mathbb{N})$ *es el conjunto de listas finitas de naturales,* $\mathcal{I}(\bar{n}) = \bar{n}^{\mathcal{I}}$ *es la función que agrega* n *al final de una lista;* $+^{\mathcal{I}}$ *es la concatenación de dos listas,* $s^{\mathcal{I}}$ *es la reversa de una lista.*

- *Si* $\sigma(x) = [3, 5],\ \sigma(z) = [8, 8],\ \sigma(y) = [1, 2, 3]$ *entonces*
  - $\mathcal{I}_\sigma(\bar{4}\bar{0}(\bar{1}\bar{3}(z + s(x)))) = [8, 8, 5, 3, 13, 40]$
  - $\mathcal{I}_\sigma(s(s(x)) + (z + s(\bar{5}(y)))) = [3, 5] +^{\mathcal{I}} ([8, 8] +^{\mathcal{I}} [5, 3, 2, 1]) = [3, 5, 8, 8, 5, 3, 2, 1]$
- *Si* $\sigma(x) = [5, 6, 7],\ \sigma(z) = [2],\ \sigma(y) = [9, 11]$ *entonces*
  - $\mathcal{I}_\sigma(\Big(\overline{666}(x + s(y))\Big)) = 6\bar{6}\bar{6}^{\mathcal{I}}([5, 6, 7] +^{\mathcal{I}} [11, 9]) = [5, 6, 7, 11, 9, 666]$
  - $\mathcal{I}_\sigma(\overline{32}(x+\bar{0}(\overline{33}(z)+y))+\bar{1}(z+s(x))) = [5, 6, 7, 2, 33, 9, 11, 0, 32]+^{\mathcal{I}}[2, 7, 6, 51] = [5, 6, 7, 2, 33, 9, 11, 0, 32, 2, 7, 6, 51]$

Veamos ahora algunas propiedades imporantes de la interpretación de términos.

Definición 9 (Estado modificado o actualizado) *Sea* $\sigma : \mathsf{Var} \to \mathcal{U}$ *un estado de las variables. Dadas las variables* $x_1, \ldots, x_k$ *y los elementos del universo* $m_1, \ldots, m_k \in \mathcal{U}$*, definimos el estado que actualiza a* σ *en* $x_1, \ldots, x_k$ *con* $m_1, \ldots, m_k$*, denotado* $\sigma[x_1, \ldots, x_k/m_1, \ldots, m_k]$ *o bien* $\sigma[\vec{x}/\vec{m}\,]$*, como sigue:*

$$\sigma[\vec{x}/\vec{m}\,](y) = \begin{cases} \sigma(y) & si\ y \notin \{x_1, \ldots, x_k\} \\ \\ m_i & si\ y = x_i \quad 1 \le i \le k \end{cases}$$

Obsérvese que $\sigma[\vec{x}/\vec{m}\,]$ es un estado que difiere de σ únicamente en los valores de $\vec{x}$ y que la expresión $\sigma[\vec{x}/\vec{m}\,]$ es sólo una notación para dicho estado, la aplicación a una variable y es entonces $\sigma[\vec{x}/\vec{m}\ ](y)$. Como puede observarse esta notación recuerda a una sustitución, esto no es coincidencia, sino que los dos conceptos están fuertemente relacionados como veremos adelante.

Lema 1 (Lema de coincidencia para términos) *Sean* $t \in \mathsf{TERM}$ *y* σ_1, σ_2 *dos estados de las variables tales que* $\sigma_1(x) = \sigma_2(x)$ *para toda variable* x *que figura en* t*. Entonces* $\mathcal{I}_{\sigma_1}(t) = \mathcal{I}_{\sigma_2}(t)$*.*

Demostración. Inducción sobre t. Ejercicio ⊣

Veamos ahora la relación entre los estados modificados y las sustituciones. Supóngase que tenemos un término t donde figura la variable x y que queremos evaluar $t[x := r]$ en algún estado de cierta interpretación, para esto aplicamos la definición recursiva y cada vez que encontramos una presencia del subtérmino r en $t[x := r]$ debemos su valor; esta situación es poco práctica, pensemos por ejemplo en que r es un término costoso de evaluar y que hay muchas presencias de x en t y por lo tantomuchas presencias de r en $t[x := r]$. Una mejor estrategia para evaluar $t[x := r]$ sería evaluar r en el estado dado σ, lo cual nos da un cierto valor m y después evaluar el término t en el estado modificado $\sigma[x/m]$. El resultado debe ser el mismo y ahora hemos evitado la evaluación repetida de r. La corrección de este procedimiento la asegura el siguiente

Lema 2 (Lema de sustitución para términos) *Sean $r \in \mathsf{TERM}$, σ un estado de las variables, $\rho = [\vec{x} := \vec{s}\,]$ una sustitución y $m_1, \ldots, m_k \in \mathcal{U}$ tales que $\mathcal{I}_\sigma(s_i) = m_i \;\; 1 \leq i \leq k$. Entonces*

$$\mathcal{I}_\sigma\left(r[\vec{x} := \vec{s}\,]\right) = \mathcal{I}_{\sigma[\vec{x}/\vec{m}\,]}(r)$$

Demostración. Inducción sobre t. Ejercicio ⊣

Nótese que el lema de sustitución relaciona fuertemente la noción sintáctica de sustitución con la noción semántica de actualización de un estado de modo que es posible transferir un razonamiento sintáctico en una cuestión semántica y viceversa.

5. Ecuaciones

Una vez que contamos con la interpretación de términos como individuos de un mundo el siguiente paso es preguntarnos acerca de relaciones entre estos individuos y cómo representarlas. Por el momento sólo nos ocuparemos de la relación fundamental de igualdad expresada en la sintaxis mediante ecuaciones.

Definición 10 *Una ecuación o igualdad es una expresión de la forma $t = s$ donde $t, s \in \mathsf{TERM}$.*

El símbolo $=$ es universalmente conocido y corresponde a la noción de igualdad. Nótese que este símbolo no pertenece nunca a una signatura. Las ecuaciones forman parte de las fórmulas más simples, las atómicas, en la lógica de predicados que estudiaremos más adelante.

Discutamos ahora el significado de las ecuaciones el cual es obvio intutivamente, la ecuación $s = t$ será verdadera syss s y t representan al mismo individuo.

5.1. Igualdad semántica

Definición 11 *Dados un modelo $\mathcal{M} = \langle \mathcal{U}, \mathcal{I} \rangle$, un estado σ y una ecuación $E =_{def} t = s$, decimos que E es satisfacible en $\mathcal{M}$ y σ, denotado $\mathcal{M}, \sigma \models E$ si y sólo si $\mathcal{I}_\sigma(s) = \mathcal{I}_\sigma(t)$*

De acuerdo a esta definición, una ecuación $t = s$ es satisfacible en un estado dado σ syss en dicho estado el significado de los dos términos t y s es exactamente el mismo. Nótese que el símbolo de igualdad en $\mathcal{I}_\sigma(s) = \mathcal{I}_\sigma(t)$ denota a la igualdad en el universo. Podríamos distinguir el símbolo del lenguaje del símbolo del universo por ejemplo usando $\approx$ para la igualdad del lenguaje, sin embargo creemos que esto no es necesario.

Veamos algunos ejemplos:

Ejemplo 5.1 *En relación al modelo $\mathcal{M} = \langle \mathbb{N}, \mathcal{I} \rangle$ del ejemplo 4.1 se tiene que*

Si $\sigma(x) = 5, \sigma(z) = 3, \sigma(y) = 2$ entonces

- $\mathcal{M}, \sigma \models x = z + y$
- $\mathcal{M}, \sigma \models s(s(0) \times x = x + x$
- $\mathcal{M}, \sigma \models s(s(x)) \times y = z \times z + x$
- $\mathcal{M}, \sigma \not\models s(x) = s(z) \times s(0)$
- $\mathcal{M}, \sigma \not\models s(x + y) = z \times x$

Ejemplo 5.2 *En relación al modelo $\mathcal{M} = \langle List(\mathbb{N}), \mathcal{I} \rangle$ del ejemplo 4.2 se tiene que*

Si $\sigma(x) = [2], \sigma(z) = [3, 3], \sigma(y) = [2, 4, 6], \sigma(w) = [4]$ entonces

- $\mathcal{M}, \sigma \models z = s(z)$
- $\mathcal{M}, \sigma \models s(x + z) = \bar{2}(z)$
- $\mathcal{M}, \sigma \models x + \bar{6}(w) = y$
- $\mathcal{M}, \sigma \models \bar{6}(s(\bar{2}(w))) = \bar{6}(x + w)$
- $\mathcal{M}, \sigma \not\models s(x) = s(z)$
- $\mathcal{M}, \sigma \not\models s(x + y) = s(x) + s(y)$

5.2. Unificación e igualdad sintáctica

Una noción de igualdad de gran importancia es aquella que no requiere de una interpretación semántica sino de una sustitución sintáctica. En sentido estricto una igualdad $t = s$ sólo tendría solución sintáctica cuando t y s son exactamente el mismo término. Sin embargo, existe una noción más general que más adelante justificaremos semánticamente y que nos permite igualar términos bajo sustitución. Por ejemplo la ecuación $f(g(x), z) = f(w, h(a, y))$ puede resolverse sintácticamente si aplicamos la sustitución $\rho = [w := g(x), z := h(a, y)]$ a ambos términos, obteniendo la ecuación $f(g(x), h(a, y)) = f(g(x), h(a, y))$ donde ambos lados de la igualdad son idénticos.

El proceso de unificación consiste en encontrar, dado un conjunto de términos W, una sustitución ρ de tal forma que el conjunto resultante $W\sigma$ conste de un solo elemento.
En adición a sus aplicaciones en programación lógica, la unificación también es importante para los sistemas de reescritura de términos, el razonamiento automátizado y los sistemas de tipos. A continuación estudiamos el caso general así como un algoritmo de unificación.

Definición 12 *Sea W un conjunto no vacío de términos. Un unificador de W es una sustitución ρ tal que $|W\rho| = 1$, es decir tal que el conjunto $W\rho$ resultante de aplicar ρ a todos los elementos de W consta de un mismo elemento. Si W tiene un unificador decimos que W es unificable.*

Ejemplo 5.3 Sea $W = \{g(x, f(y)), g(x, f(x)), g(u, v)\}$ un conjunto de términos, entonces la sustitución $\rho = [x := a,\ y := a,\ u := a,\ v := f(a)\}$ es un unificador de W ya que $W\rho = \{g(a, f(a))\}$

Un conjunto de términos puede tener una infinidad de unificadores o ninguno, dado un conjunto finito de fórmulas W, podemos decidir mediante un algoritmo si W es unificable o no; si W es unificable el algoritmo proporciona un unificador llamado **unificador más general**.

Definición 13 *Un unificador ρ de un conjunto de términos W, se llama unificador más general (**umg**) si para cada unificador τ de W, existe una sustitución ϑ, tal que $\rho\vartheta = \tau$.*

La importancia de los unificadores más generales es que nos permiten representar de manera finita un número infinito de sustituciones.

Ejemplo 5.4 Sean $W = \{fgaxgyb, fzguv\}$ con $f^{(2)},\ g^{(2)}$ y

$$\tau = [x := a, z := gaa, y := u, v := b] \qquad \rho = [z := gax, y := u, v := b].$$

Entonces τ y ρ son unificadores de W y es fácil ver que ρ resulta ser el **umg** y en particular si $\vartheta = [x := a]$ entonces $\rho\vartheta = \tau$.

Antes de discutir un algoritmo de unificación es conveniente hacer un análisis intuitivo del problema. Basta analizar un conjunto de dos términos digamos $W = \{t_1, t_2\}$, queremos ver si el conjunto W es unificable. Hay que analizar varios casos:

1. Los términos t_1 y t_2 son constantes.
   En este caso $t_i\sigma = t_i$ para cualquier sustitución σ, de manera que W será unificable si y sólo si t_1 y t_2 son la misma constante.

2. Alguno de los términos t_1, t_2 es una variable.
   Supongamos que $t_1 = x$, si la variable x figura en t_2 entonces W no es unificable (¿por qué?), en caso contrario $\rho = [x := t_2]$ unifica a W.

3. t_1 y t_2 son términos funcionales.
   En este caso W es unificable si y sólo si se cumplen las siguientes dos condiciones:

   - Los símbolos principales (es decir los primeros de izquierda a derecha), de t_1 y t_2 son el mismo.
   - Cada par correspondiente de subexpresiones de t_1 y t_2 deben ser unificables.

Veamos unos ejemplos de las ideas anteriores

Ejemplo 5.5 Considerando los mismos símbolos de función que el ejemplo anterior, tenemos los siguientes ejemplos:

- El conjunto $\{c, d\}$ no es unificable pues consta de dos constantes diferentes.
- El conjunto $\{x, fy\}$ es unificable mediante $\sigma = [x := fy]$.
- El conjunto $\{gxw, hya\}$ no es unificable pues g y h son símbolos distintos.
- El conjunto $\{fxgyw, fagbhw\}$ no es unificable pues los subtérminos w y hw no son unificables.

Implementamos a continuación estas ideas mediante un eficiente algoritmo de unificación.

5.3. El algoritmo de unificación de Martelli-Montanari

Describimos en esta sección all algoritmo de unificación de Martelli-Montanari [2]:

Entrada: un conjunto de ecuaciones $\{s_1 = r_1, \ldots, s_k = r_k\}$ tales que se desea unificar simultáneamente s_i con r_i para $1 \leq i \leq k$.

Salida: un unificador más general μ tal que $s_i\mu = r_i\mu$ para toda $1 \leq i \leq n$.

Para unificar un conjunto de términos:

- se colocan como ecuaciones las expresiones a unificar
- escoger una de ellas de manera *no determinística* que tenga la forma de alguna de las siguientes opciones y realizar la acción correspondiente:

[2] "An efficient unification algorithm", ACM Trans. Prog. Lang. and Systems vol. 4, 1982, p.p. 258-282

	Nombre de la regla	$t_1 = t_2$	Acción
[Desc]	Descomposición	$fs_1 \dots s_n = ft_1 \dots t_n$	sustituir por $\{s_i = t_i\}$
[DFalla]	Desc. fallida	$fs_1 \dots s_n = gt_1 \dots t_n$ donde $g \neq f$	falla
[Elim]	Eliminación	$x = x$	eliminar
[Swap]	Intercambio	$t = x$ donde t no es una variable	sustituir por $x = t$
[Sust]	Sustitución	$x = t$ donde x no figura en t	eliminar $x = t$ y aplicar la sustitución $[x := t]$ a las ecuaciones restantes
[SFalla]	Sust. fallida	$x = t$ donde x figura en t y $x \neq t$	falla

- el algoritmo termina cuando no se puede llevar a cabo alguna acción o cuando falla.
  En caso de éxito se obtiene el conjunto vacío de ecuaciones y el unificador más general se obtiene al componer todas las sustituciones usadas por la regla de sustitución en el orden en que se generaron.

Observaciones: El caso en que se deba unificar a dos constantes iguales a, se genera la ecuación $a = a$ que por la regla de descomposición se sustituye por el conjunto vacío de ecuaciones dado que constantes se consideran funciones sin argumentos, así cualquier ecuación de la forma $a = a$ se elimina. En el caso de una ecuación $a = b$ (con dos constantes distintas) falla por la regla de descomposición fallida.

Ejemplo 5.1 Sean $f^{(2)}, g^{(1)}, h^{(2)}$ y $W = \{fgxhxu,\ fzhfyyz\}$. Mostramos el proceso de ejecución del algoritmo de manera similar al razonamiento de verificación de correctud de argumentos por interpretaciones.

1.	$\{fgxhxu = fzhfyyz\}$	Entrada
2.	$\{gx = z,\ hxu = hfyyz\}$	Desc,1
3.	$\{z = gx,\ hxu = hfyyz\}$	Swap,2
4.	$\{hxu = hfyygx\}$	Sust,3, $[z := gx]$
5.	$\{x = fyy,\ u = gx\}$	Desc,4
6.	$\{u = gfyy\}$	Desc,5, $[x := fyy]$
7.	$\varnothing$	Desc,6, $[u := gfyy]$

El unificador se obtiene al componer las sustituciones utilizadas desde el inicio:

$$\begin{aligned} \mu &= [z := gx][x := fyy][u := gfyy] \\ &= [z := gfyy, x := fyy][u := gfyy] \\ &= [z := gfyy, x := fyy, u := gfyy] \end{aligned}$$

Veamos ahora un ejemplo de un conjunto no unificable:

Ejemplo 5.2 Sean $f^{(3)}, g^{(1)}$ y $W = \{fxyx,\ fygxx\}$.

1.	$\{fxyx = fygxx\}$	Entrada
2.	$\{x = y,\ y = gx,\ x = x\}$	Desc,1
3.	$\{x = y,\ y = gx\}$	Elim,2
4.	$\{y = gy\}$	Sust,3, $[x := y]$
5.	X	SFalla,4

Para terminar enunciamos el teorema de correción total del algoritmo.

Teorema 1 (Corrección total del algoritmo de Martelli-Montanari) *Dado un conjunto finito de expresiones W, el algoritmo MM termina, dando como resultado el mensaje "W no es unificable", en el caso en que W no sea unificable, y un unificador más general μ de W en el caso en que W sea unificable.*

6. Ejercicios

Verificar si los siguientes conjuntos son unificables utilizando el algoritmo de Martelli-Montanari, puedes obviar algunos pasos pero debes mostrar los que generan sustituciones.

a) $W = \{h(f(a), g(x)),\ h(z, z)\}$

b) $W = \{f(w, f(x, h(z))),\ f(g(x), f(x, y)),\ f(g(x), f(a, b))\}$

c) $W = \{fxgfayz, fbgfagxcfyx\}$ con $f^{(2)}, g^{(2)}$

d) $W = \{f\Big(x,\ g(f(a, y), z\Big), f\Big(b,\ g(f(a, g(x, c)), f(y, x))\Big)\}$

e) $W = \{Q(x, f(x, y)),\ Q(y, f(y, a)),\ Q(b, f(b, a))\}$

f) $W = \{Pxfy,\ Pgyafb,\ Pgbzw\}$ con $P^{(2)}, f^{(1)}, g^{(2)}$

g) $W = \{Qazgabc,\ Qafxgaby,\ Qaffwgaxgcba\}$ con $Q^{(3)}, f^{(1)}, g^{(3)}$

h) $W = \{Pxfxgy,\ Pafgaga,\ Pyfyga\}$ con $P^{(3)}, f^{(1)}, g^{(1)}$

i) $W = \{Rfayz, Rxyfz, Ryfab\}$ con $R^{(3)}, f^{(1)}$.

j) $W = \{P(x, f(x), c),\ P(u, b, z)\}$

k) $W = \{Q(y, z),\ Q(x, f(a)),\ Q(f(z), z)\}$.

l) $W = \{R(w, f(b), f(g(y))), R(a, x, f(g(y))),\ R(z, f(z), f(u))\}$

m) $W = \{T(u, v, w, z),\ T(f(z), x, g(h(a, b)), g(c)),\ T(f(g(y)), z, w, g(y))\}$

A. Sobre la sintaxis de los términos

Con el objetivo de eliminar lo más posible el uso de paréntesis anidados, los términos, y más adelante los predicados, se escribiran sin usar paréntesis ni comas que separen sus argumentos. Esto no causa ambigüedad alguna dado que en la signatura cada símbolo tiene definido un índice y número de argumentos. Por ejemplo, si sabemos que $f^{(2)}, g^{(3)}$ es decir, el símbolo de función f espera dos argumentos y g espera tres, entonces el término atómica $g(f(x, a), b, z)$ puede escribirse como $gfxabz$. Los paréntesis pueden restaurarse sin ambigüedad pues sabemos el índice de f es 2, de donde el único término que empieza con f es fxa siendo la variable x y la constante a sus dos argumentos. De esta forma fxa es el primero de los tres argumentos de g. En la cadena restante bz tienen que aparecer dos términos y la única manera para que esto suceda es con la constante b y la variable z.

Veamos otros ejemplos, dando primero los índices de los símbolos; la expresión sín paréntesis y la correspondiente expresión con paréntesis.

- $h^{(2)}, f^{2}, g^{(1)} \quad hgfxbfgyz \quad h\Big(g(f(x, b)), f(g(y), z)\Big)$
- $f^{(1)}, g^{(2)}, h^{(3)} \quad gfwhxgzaffb \quad g\Big(f(w), h(x, g(z, a), f(f(b)))\Big)$