Práctico 1
Inducción y recursión

Ejercicio 14

Bosquejo de solución

Nota: En los ejercicios que tratan sobre un alfabeto Σ y la clausura Σ^*, usaremos como válidas las siguientes propiedades y notaciones:

- Si tenemos dos tiras u y w de Σ^*, escribimos uw para representar la *concatenación* de ambas tiras. Esto es: la tira formada por todos los caracteres de u seguidos por todos los carateres de w[1].
- Si x es un símbolo del alfabeto Σ, usamos x indistintamente para representar ese símbolo como la tira de Σ^* formada por ese único símbolo.
- La tira vacía ε es el neutro de la concatenación de tiras. Esto es $w\varepsilon = \varepsilon w = w$ para toda tira $w \in \Sigma$.
- La concatenación es asociativa. Esto es: $(wu)v = w(uv) = wuv$.

a. Definición de `invertir`:

$$
\begin{aligned}
\texttt{invertir}(\varepsilon) &= \varepsilon \\
\texttt{invertir}(xw) &= \texttt{invertir}(w)x
\end{aligned}
$$

b. Definimos la propiedad $P(w) := (\bar{\forall} y \in \Sigma)\ \texttt{invertir}(wy) = y\ \texttt{invertir}(w)$.

Se demostrará usando PIP de Σ^* que $(\bar{\forall} w \in \Sigma^*)\ P(w)$

Paso Base: $P(\varepsilon)$: $(\bar{\forall} y \in \Sigma)\ \texttt{invertir}(\varepsilon y) = y\ \texttt{invertir}(\varepsilon)$

Demostración: Sea y un elemento cualquiera de Σ:

$\texttt{invertir}(\varepsilon y)$
$=$ (propiedad de Σ^*)
$\texttt{invertir}(y\varepsilon)$
$=$ (Definición de `invertir`)
$\texttt{invertir}(\varepsilon)y$
$=$ (Definición de `invertir`)
εy
$=$ (propiedad de Σ^*)
$y\varepsilon$
$=$ (definición de `invertir`)
$y\ \texttt{invertir}(\varepsilon)$
$\square$

Como la prueba se hizo para y arbitrario, vale para todo $y \in \Sigma$.

[1] Notar que la concatenación de tiras coincide con la función *append* definida en el Ejercicio 13.

Paso Inductivo: (H) $P(w)$: $(\bar{\forall} y \in \Sigma)$ $\texttt{invertir}(wy) = y\ \texttt{invertir}(w)$
(T) $P(xw)$: $(\bar{\forall} y \in \Sigma)$ $\texttt{invertir}(xwy) = y\ \texttt{invertir}(xw)$

Demostración: Consideramos un elemento cualquiera $y \in \Sigma$

$\texttt{invertir}(xwy)$
$=$ (def. de $\texttt{invertir}$)
$\texttt{invertir}(wy)\ x$
$=$ (hipótesis inductiva)
$y\ \texttt{invertir}(w)\ x$
$=$ (def. de $\texttt{invertir}$)
$y\ \texttt{invertir}(xw)$
$\square$

Como la prueba se hizo para y arbitrario, vale para todo $y \in \Sigma$.

c. Definición de tiras palíndromes:

I $\varepsilon \in A$.
II Si $x \in \Sigma$ entonces $x \in A$.
III Si $x \in \Sigma$ y $w \in A$ entonces $xwx \in A$

d. Comenzamos formulando el PIP para A

Si se cumple:

I $P(\varepsilon)$
II Si $x \in \Sigma$ entonces $P(x)$
III Si $x \in \Sigma$ y $P(w)$ entonces $P(xwx)$

entonces se cumple $(\bar{\forall} w \in A)\ P(w)$

Propiedad: Sea $P(w) := \texttt{invertir}(w) = w$

Paso Base (i): $P(\varepsilon)$: $\texttt{invertir}(\varepsilon) = \varepsilon$.

Demostración: Se cumple por la primera regla de la definición de `invertir`.
$\square$

Paso Base (ii): **(H)** $x \in \Sigma$
(T) $P(x)$: $\texttt{invertir}(x) = x$

Demostración: Se prueba por la siguiente cadena de igualdades
$\texttt{invertir}(x)$
$=$ (propiedad de Σ^*)
$\texttt{invertir}(x\varepsilon)$
$=$ (definición de $\texttt{invertir}$)
$\texttt{invertir}(\varepsilon)\ x$
$=$ (def. de $\texttt{invertir}$)
εx
$=$ (propiedad de Σ^*)
x
$\square$

Paso Inductivo (iii): **(H)** $x \in \Sigma$

$P(w)$: $\texttt{invertir}(w) = w$

(T) $P(xwx)$: $\texttt{invertir}(xwx) = xwx$

Demostración: Se prueba por la siguiente cadena de igualdades:

$\texttt{invertir}(xwx)$

$=$ (def. $\texttt{invertir}$)

$\texttt{invertir}(wx)\ x$

$=$ (Lema parte (b))

$x\ \texttt{invertir}(w)\ x$

$=$ (hip. inductiva)

xwx

$\square$

Ejercicio 19 (Numerales binarios)

Bosquejo de solución

a. Veremos dos alternativas para definir la función `eval`.

La primera alternativa utiliza la función auxiliar $largo : \Sigma^* \to \mathbb{N}$, la cual devuelve el largo de una tira. Recordemos que el sistema de numeración binario es un sistema posicional, por lo que el valor de un símbolo depende tanto del símbolo como de su posición en la tira. En particular, en el sistema binario, la tira $a_n a_{n-1} ... a_1 a_0$ representa el decimal $a_n \times 2^n + a_{n-1} \times 2^{n-1} + ... + a_1 \times 2 + a_0$. A su vez, podemos obtener la posición de un símbolo en la tira si calculamos el largo restante de la tira; es decir, a_n está en la posición n ya que $largo(a_{n-1} ... a_1 a_0) = n$.

Teniendo en cuenta estas consideraciones, podemos definir `eval` siguiendo el ERP sobre B_D de la siguiente forma:

$$
\begin{aligned}
&\texttt{eval} : B_D \to \mathbb{N}\\
&\texttt{eval}(x) \quad = x\\
&\texttt{eval}(xw) = x \times 2^{largo(w)} + \texttt{eval}(w)
\end{aligned}
$$

La segunda alternativa para definir la función `eval` es utilizando recursión de cola. Para esto, definiremos una función auxiliar $\texttt{evalaux} : B_D \times \mathbb{N} \to \mathbb{N}$, la cual toma un parámetro adicional que funciona como acumulador. Este acumulador va llevando el resultado de computar el decimal representado por la tira hasta ese momento. A su vez, utilizaremos el conocimiento de la regla de Horner presente en la letra (Por esta razón, para entender esta parte puede ser útil primero comprender la solución de la parte b).

Definimos `evalaux` siguiendo el ERP sobre B_D de la siguiente forma:

$$
\begin{aligned}
&\texttt{evalaux} : B_D \times \mathbb{N} \to \mathbb{N}\\
&\texttt{evalaux}(x, n) \quad = n \times 2 + x\\
&\texttt{evalaux}(xw, n) = \texttt{evalaux}(w, n \times 2 + x)
\end{aligned}
$$

Finalmente, definimos `eval` siguiendo el ERP sobre B_D de la siguiente forma:

$$
\begin{aligned}
&\texttt{eval} : B_D \to \mathbb{N}\\
&\texttt{eval}(x) \quad = x\\
&\texttt{eval}(xw) = \texttt{evalaux}(xw, 0)
\end{aligned}
$$

b. Para definir $\texttt{eval} : B_I \to \mathbb{N}$, debemos considerar la regla de Horner presente en la letra. Observando la misma se puede ver claramente el paso inductivo:

$$\begin{aligned}
&(((1 \times 2 + 0) \times 2 + 1) \times 2 + 0) \times 2 + 0 \\
&= (((\texttt{eval}(1) \times 2 + 0) \times 2 + 1) \times 2 + 0) \times 2 + 0 \\
&= ((\texttt{eval}(10) \times 2 + 1) \times 2 + 0) \times 2 + 0 \\
&= (\texttt{eval}(101) \times 2 + 0) \times 2 + 0 \\
&= \texttt{eval}(1010) \times 2 + 0 \\
&= \texttt{eval}(10100)
\end{aligned}$$

Por lo tanto, podemos definir `eval` siguiendo el ERP sobre B_I de la siguiente forma:

$$\begin{aligned}
&\texttt{eval} : B_I \to \mathbb{N} \\
&\texttt{eval}(x) \quad = x \\
&\texttt{eval}(wx) = \texttt{eval}(w) \times 2 + x
\end{aligned}$$

c. Para definir correctamente la función `par`, se deben distinguir los casos en los que $x = 1$, ya que en estos la paridad cambia.

La paridad de la tira wx cambia en función de la paridad de la tira w. Es decir, si la tira w tiene cantidad de 1's par y $x = 1$, la tira wx tiene cantidad de 1's impar. Por otro lado, si la tira w tiene cantidad de 1's impar y $x = 1$, la tira wx tiene cantidad de 1's par. Finalmente, si $x = 0$, la paridad de la tira wx no cambia (si w tiene cantidad de 1's impar, wx tiene cantidad de 1's impar; si w tiene cantidad de 1's par, wx tiene cantidad de 1's par).

Teniendo en cuenta estas consideraciones, definimos `par` siguiendo el ERP sobre B_I de la siguiente forma:

$$\begin{aligned}
&\texttt{par} : B_I \to \{0, 1\} \\
&\texttt{par}(x) \quad = \text{si } x = 1 \text{ entonces } 0 \text{ sino } 1 \\
&\texttt{par}(wx) = \text{si } x = 1 \text{ entonces } 1 - \texttt{par}(w) \text{ sino } \texttt{par}(w)
\end{aligned}$$

Ejercicio 21 (Otros esquemas)

Bosquejo de solución

a. Ninguna de las tres funciones siguen el ERP de $\mathbb{N}$, por lo que para probar que son funciones hay que probar para cada una exhaustividad, no superposición y terminación

1 función g

1. **Exhaustividad**

   Para probarlo vamos a usar que se cumple la siguiente propiedad:

$$\mathbb{N} = \{0, 1\} \cup \{n + 2, n \in \mathbb{N}\}$$

   O sea todo natural es 0, 1 o tiene la forma de $n + 2$ siendo n un natural. Por lo tanto hay una ecuación que computa su correspondiente para cada natural.

2. No superposición
Ser cero, uno, o $n+2$ son condiciones mutuamente incompatibles. No existe $n \in \mathbb{N}$ tal que $n+2=0$ o $n+2=1$.

3. Terminación
Es claro que todas las llamadas recursivas se hacen en elementos menores. El conjunto $\mathbb{N}$ tiene un orden bien fundado y todas las llamadas que hay son en elementos menores, por lo que en algún momento la recursion va a terminar en alguno de los pasos base. Es decir $n+2>n+1>n$ por lo que las llamadas $g(n)$ y $g(n+1)$ son sobre elementos menores según la relación de orden $>$ usada en los naturales.

2 función h

1. Exhaustividad
Al igual que la parte anterior $\mathbb{N}$ es de la forma 0 o 1 o $n+2$. Y hay una ecuación que computa su correspondiente para cada natural.

2. No superposición
Ser cero, uno, o $n+2$ son condiciones mutuamente incompatibles. No existe $n \in \mathbb{N}$ tal que $n+2=0$ o $n+2=1$.

3. Terminación
Para probar que cumple esta condición, hay que probar que $n+2>(n+2)\ mod\ 2$. Como sabemos $n+2\ mod\ 2=0$ o $n+2\ mod\ 2=1$. También sabemos que para todo $n \in \mathbb{N}$ $n+2 \geq 2$ por lo que $n+2>1>0$ entonces $n+2>(n+2)\ mod\ 2$ para cualquier n. Por lo tanto la llamada $h((n+2)\ mod\ 2)$ es sobre un elemento menor según la relación de orden $>$ usada en los naturales.

3. función j

1. Exhaustividad
Sabemos que $\mathbb{N}$ es de la forma 0, $2n+2$ o $2n+1$. Los naturales de la forma $2n+2$ son los pares excepto el cero, los naturales de la forma $2n+1$ son los impares. $\mathbb{N}=\{0\} \cup Pares \cup Impares$. Y hay una ecuación que computa su correspondiente para cada tipo de natural.

2. No superposición
No hay superposición entre 0, $2n+2$ o $2n+1$ ya que 0 no es de la forma $2n+2$ ni $2n+1$, y como se sabe $Pares \cap Impares = \emptyset$.

3. Terminación
Hay que ver que para todo $n \in \mathbb{N}$ $2n+2>n$ y $2n+1>n$. Restando n de ambos lados de la inecuación, se ve que $n+2>0$ y $n+1>0$ que se cumple para todo $n \in \mathbb{N}$. Por lo tanto la llamada $j(n)$ es sobre un elemento menor según la relación de orden $>$ usada en los naturales.

b. A continuación se probara que $g=f$ y que $h=f$. Para probar que $f=g$ hay que probar que para todo $n \in \mathbb{N} f(n)=g(n)$.

1. Probaremos que $f=g$. Para probar esto usaremos la propiedad de que $\mathbb{N}=Pares \cup Impares$. Y probaremos con dos inducciones sobre los naturales.

Probaremos por inducción que $(\bar{\forall} n \in \mathbb{N}) g(2n+1)=f(2n+1)=0$: Haremos una demostración usando el PIP en los naturales. La propiedad es la siguiente:

$$\mathbb{P}(n) := g(2n+1)=0 \tag{1}$$

Demostración:

- **Paso base:** $g(1) = 0$
  Demostración: Trivialmente por definición de g
- **Paso Inductivo:**
  (H) $g(2n+1) = 0$
  (T) $g(2(n+1)+1) = 0$
  Demostración:

$$
\begin{aligned}
& g(2(n+1)+1) \\
& = \text{(aritmética)} \\
& g((2n+1)+2) \\
& = \text{(definición de } g\text{)} \\
& g(2n+1)) + min\{g(2n+1), g((2n+1)+1)\} \\
& = (\textbf{Hipótesis Inductiva}) \\
& 0 + min\{0, g(2n+2)\} \\
& = (g(n) \text{ es mayor o igual a cero, se demuestra al final del ejercicio}) \\
& 0
\end{aligned}
$$

Queda probado por PIP que se cumple $(\bar{\forall} n \in \mathbb{N}) g(2n+1) = f(2n+1) = 0$

Ahora probaremos que $(\bar{\forall} n \in \mathbb{N}) g(2n) = f(2n) = 1$ usando el PIP en los naturales. La propiedad es la siguiente:

$$\mathbb{P}(n) := g(2n) = 1 \tag{2}$$

Demostración:

- **Paso base:** $g(0) = 1$
  Demostración: Trivialmente por definición de g
- **Paso Inductivo:**
  (H) $g(2n) = 1$
  (T) $g(2(n+1)) = 1$
  Demostración:

$$
\begin{aligned}
& g(2(n+1)) \\
& = \text{(aritmética)} \\
& g(2n+2) \\
& = \text{(definición de } g\text{)} \\
& g(2n) + min\{g(2n), g(2n+1)\} \\
& = (\textbf{Hipótesis Inductiva}) \\
& 1 + min\{1, g(2n+1)\} \\
& = (g(2n+1) = 0 \text{ demostrado anteriormente}) \\
& 1 + min\{1, 0\} \\
& = \text{(aritmética)} \\
& 1
\end{aligned}
$$

Queda probado por PIP que se cumple $(\bar{\forall} n \in \mathbb{N}) g(2n) = f(2n) = 1$
Como para todo n en los naturales es par o impar. Se cumple que $(\bar{\forall} n \in \mathbb{N}) f(n) = g(n)$.

2. Probaremos que $f = h$. Para probar esto usaremos la propiedad de que $\mathbb{N} = Pares \cup Impares$. Y probaremos con dos inducciones sobre los naturales.

Probaremos por inducción que $(\bar{\forall} n \in \mathbb{N}) h(2n+1) = f(2n+1) = 0$: Haremos una demostración usando el PIP en los naturales. La propiedad es la siguiente:

$$\mathbb{P}(n) := h(2n+1) = 0 \tag{3}$$

Demostración:

- **Paso base:** $h(1) = 0$
  Demostración: Trivialmente por definición de h
- **Paso Inductivo:**
  (H) $h(2n+1) = 0$
  (T) $h(2(n+1)+1) = 0$
  Demostración:

$$\begin{aligned}
&h(2(n+1)+1)\\
&= \text{(aritmética)}\\
&h((2n+1)+2)\\
&= \text{(definición de } h\text{)}\\
&h((2n+1)+2 \ mod \ 2)\\
&= \textbf{(aritmética)}\\
&h((2(n+1)+1) \ mod \ 2)\\
&= \text{(definición de } mod\text{)}\\
&h(1)\\
&= \text{(definición de } h\text{)}\\
&0
\end{aligned}$$

Queda probado por PIP que se cumple $((\bar{\forall} n) \in \mathbb{N}) h(2n+1) = f(2n+1) = 0$

Ahora probaremos que $(\bar{\forall} n \in \mathbb{N}) h(2n) = f(2n) = 1$ usando el PIP en los naturales. La propiedad es la siguiente:

$$\mathbb{P}(n) := h(2n) = 1 \tag{4}$$

Demostración:

- **Paso base:** $h(0) = 1$
  Demostración: Trivialmente por definición de h
- **Paso Inductivo:**
  (H) $h(2n) = 1$
  (T) $h(2(n+1)) = 1$
  Demostración:

$$\begin{aligned}
&h(2(n+1))\\
&= \text{(aritmética)}\\
&h(2n+2)\\
&= \text{(definición de } h\text{)}\\
&h((2n+2 \ mod \ 2)\\
&= \text{(aritmética)}\\
&h(2(n+1) \ mod \ 2)\\
&= \textbf{(definición de } mod\textbf{)}\\
&h(0)\\
&= \text{(definición de } h\text{)}\\
&1
\end{aligned}$$

Queda probado por PIP que se cumple $(\bar{\forall} n \in \mathbb{N}) h(2n) = f(2n) = 1$

Como para todo n en los naturales es par o impar. Se cumple que $(\bar{\forall} n \in \mathbb{N}) f(n) = h(n)$.

En las dos anteriores demostraciones por inducción, no se usaron hipótesis inductivas. Esto parece indicar que no es necesario utilizar el PIP para demostrar las propiedades. Sabiendo que para todo $n \in \mathbb{N}$, existe $m \in \mathbb{N}$ tal que $n = 2m$ o $n = 2m+1$. Podemos dividir en casos y probar que $(\bar{\forall} n \in \mathbb{N}) f(n) = h(n)$ sin necesidad de usar inducción.

$(\bar{\forall} n \in \mathbb{N})$ si n es **par** entonces existe $m \in \mathbb{N}$ tal que $n = 2m$

(H) n es de la forma $n = 2m$

(T) $h(2m) = 1$

Demostración:

caso: $m = 0$ entonces $h(0) = 1$

caso: $m > 0$

$h(2m)$

$=$ (aritmética)

$h((2m-2)+2)$

$=$ (definición de h)

$h(2m \ mod \ 2)$

$=$ (definición de mod)

$h(0)$

$=$ (definición de h)

1

$(\bar{\forall} n \in \mathbb{N})$ si n es **impar** entonces existe $m \in \mathbb{N}$ tal que $n = 2m+1$

(H) n es de la forma $n = 2m+1$

(T) $h(2m+1) = 1$

Demostración:

caso: $m = 1$ entonces $h(1) = 0$

caso: $m > 1$

$h(2m+1)$

$=$ (aritmética)

$h((2m-1)+2)$

$=$ (definición de h)

$h((2m+1) \ mod \ 2)$

$=$ (definición de mod)

$h(1)$

$=$ (definición de h)

0

Por lo que las pruebas fueron con n genérico dividiendo en par o impar sabiendo que la unión de estos conjuntos es el conjunto de los naturales. Por lo que fue demostrado sin necesidad de inducción.

Ahora probaremos que $(\bar{\forall} n \in \mathbb{N}) g(n) \geq 0$ usando el PIP en los naturales. La propiedad es la siguiente:

$$\mathbb{P}(n) := (\bar{\forall} k \leq n) g(k) \geq 0 \tag{5}$$

Demostración:

- **Paso base:** $\mathbb{P}(0) := (\bar{\forall} k \leq 0) g(k) \geq 0$
  Demostración: El único caso a considerar es $g(0)$. Y se cumple trivialmente por definición de g
- **Paso Inductivo:**
  (H) $(\bar{\forall} k \leq n) g(k) \geq 0$
  (T) $(\bar{\forall} k \leq n+1) g(k) \geq 0$
  Demostración: $(\bar{\forall} k \leq n+1) g(k) \geq 0$.
  Por hipotesis inductiva $(\bar{\forall} k \leq n) g(k) \geq 0$ por lo que resta probar que $g(n+1) \geq 0$

  caso: $n = 0$ entonces $g(1) = 1$ y $g(1) = 0$. Probado $(\bar{\forall} k \leq 1) g(k) \geq 0$
  caso: $n > 0$
  $g(n+1)$
  $=$ (aritmética)
  $g((n-1)+2)$
  $=$ (definición de g)
  $g(n-1) + min\{g(n-1), g(n)\}$
  $\geq$ (Hipótesis Inductiva $g(n-1)$, $g(n)$ son mayores o iguales que 0)
  0

Queda probado por PIP que se cumple $(\bar{\forall} n \in \mathbb{N}) g(n) \geq 0$

Ejercicio 22 (Árboles binarios)

Bosquejo de solución

a. Indique si las siguientes afirmaciones son verdaderas o falsas:

I. El lenguaje N_1 está incluído en el lenguaje N_3. **VERDADERA**

Intuitivamente, N_3 está definido por todas las cláusulas, mientras que N_1 solo por las dos primeras, entonces para todo elemento de N_1 se cumple que pertenece también a N_3 porque puede ser definido con la misma secuencia de reglas. Para probar que esta afirmación es verdadera, es decir, $(\bar{\forall} t \in N_1) t \in N_3$, utilizaremos el PIP sobre N_1.

Identificación de la propiedad: $P(t) := t \in N_3$.

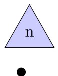
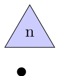
Paso Base

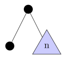
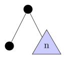
T) $P(\bullet) : \bullet \in N_3$

Demo)

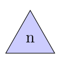
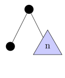
Por la cláusula 1 de la definición de N_3, tenemos que $\bullet \in N_3$. Por lo tanto, se cumple la tesis.

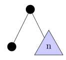
Paso Inductivo

HI) $P($  $)$:  $\in N_3$

TI) $P($  $)$:  $\in N_3$

Demo)

Por Hipótesis Inductiva sabemos que  $\in N_3$. Tenemos que probar que  $\in N_3$.

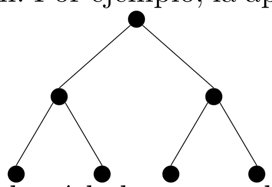
Notar que la hipótesis es la misma que la precondición de la regla 2 para el conjunto N_3, entonces aplicando esta cláusula, podemos afirmar que  $\in N_3$.

Como nos encontramos bajo las hipotesis del PIP para N_1, podemos afirmar que $(\bar{\forall} t \in N_1) t \in N_3$. Es decir, probamos que $N_1 \subseteq N_3$.

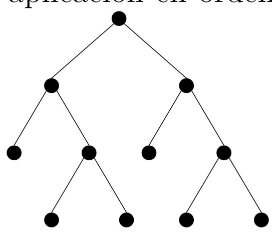
II. El lenguaje N_2 está incluído en el lenguaje N_4. **FALSA**

N_4, definido con las cláusulas 2 y 3, ambas inductivas, determinan el conjunto vacío. Para probar que N_2 no está incluído en N_4, basta con dar un testigo, en este caso encontrar un árbol t que pertenezca a N_2 pero no a N_4. Por ejemplo, $\bullet \in N_2$, por la cláusula 1, pero $\bullet \notin N_4$.

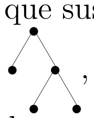
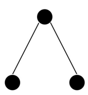
III. El lenguaje N_3 está incluído en el lenguaje N_1. **FALSA**

Como en el caso anterior, basta con dar un testigo, un árbol que perteneza a N_3 y no a N_1, para probar la falsedad de la afirmación. Por ejemplo, la aplicación sucesiva de las reglas: 1, 3, 3 generan el siguiente árbol:  Notar como este no puede ser un elemento de N_1 debido a que los árboles generados con las reglas 1 y 2 solo "crecen" hacia la derecha, en particular, si no son de la forma $\bullet$, siempre tienen un solo nodo a la izquierda por la regla 2, propiedad que el testigo no cumple.

IV. El lenguaje N_3 está incluído en el lenguaje $N_1 \cup N_2 \cup N_4$. **FALSA**

Testigo: el árbol formado por la aplicación en orden de las reglas 1, 2, 2 y 3, cuyo resultado es el siguiente: $T_{1223} =$ 

Para probar que $T_{1223} \notin N_1 \cup N_2 \cup N_4$, probaremos que no pertenece a ninguno de los 3 conjuntos de la unión. La prueba la haremos observando ciertas propiedades que se pueden inferir a partir de las reglas de construcción de los conjuntos.

- $T_{1223} \notin N_1$, por la propiedad que observamos anteriormente, los árboles de N_1 si no son $\bullet$, por la cláusula 2 siempre tienen un solo nodo a la izquierda. Claramente T_{1223} no cumple la propiedad.
- De forma similar ocurre con N_2, podemos observar la siguiente propiedad de los árboles en N_2. Si son distintos de $\bullet$, por la cláusula 3 se cumple que los subárboles del árbol formado son iguales, propiedad que también se debe cumplir para los mismos, si fueron creados por regla 3. Verifiquemos entonces que el testigo, $T_{1223} \notin N_2$, es decir, no cumple la propiedad. Claramente, T_{1223} es distinto de $\bullet$, entonces, si pertenece a N_2 tiene que haber sido creado por la cláusula 3. En primera instancia, verifica la parte de la propiedad, que sus subárboles derecho e izquierdo son iguales. Veamos ahora si el subárbol , que debe pertenecer a N_2, cumple también la propiedad. Como el subárbol no es $\bullet$, y debe pertenecer a N_2, tiene que haber sido creado con la cláusula 3. Entonces, por la propiedad antes mencionada, sus subárboles tienen que ser iguales, lo cual no cumple, porque $\bullet$ es diferente a . Concluyendo, efectivamente que $T_{1223} \notin N_2$.

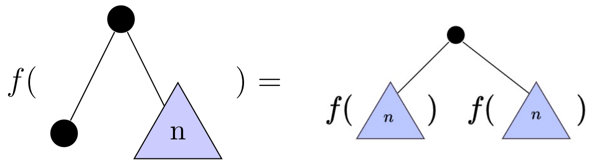
- Como N_4 es el conjunto vacío, es claro que $T_{1223} \notin N_4$.

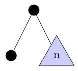
b. Cada árbol de los lenguajes anteriores codifica (o representa) al natural correspondiente a la altura del mismo; por ejemplo, la cláusula base de N_1 codifica el cero.

Teniendo en cuenta la observación que no todos los lenguajes definidos son libres, intentaremos seguir el ERP con cuidado.

I.

$$f : N_1 \to N_2$$
$$f(\bullet) = \bullet$$

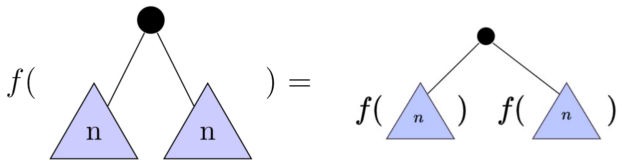


Notar la aplicación de f para (), su codificación en los naturales corresponde a 1 + la altura del sub-árbol (), entonces, para que su contraparte pertenezca a N_2 (regla 3) es necesario, para ambas sub-ramas del árbol resultado, aplicar f.

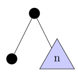
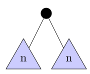
Como la definición de N_1 es libre[2] y las ecuaciones de f cumplen las hipótesis del ERP asociado a la definición dada, entonces es posible asegurar que f es una función.

II.

$$f : N_2 \to N_3$$
$$f(\bullet) = \bullet$$



Dado que la definición de N_2 es libre[3] y las ecuaciones de f cumplen las hipótesis del ERP asociado a la definición dada, entonces es posible asegurar que f es una función.

Notar que también es válida una definición que, en lugar de basarse en la regla 3 para el resultado en la segunda ecuación (donde se aplica la recursión), utlilce la regla 2, ya que  y  tienen la misma altura y ambos pertenecen a N_3.

III. Para este caso, definir $f : N_3 \to N_4$ es imposible, debido a que N_4, el conjunto de llegada, es el conjunto vacío. Entonces la correspondencia de un elemento de N_3 a un elemento de N_4 con igual altura no es posible. Por ejemplo, para $\bullet$, se puede afirmar lo siguiente: No existe un árbol t en N_4 tal que, la altura de t sea igual a la altura de $\bullet$, que es 0. Por lo que cualquier función que se quiera definir sobre estos dos conjuntos, en este orden, no va a cumplir la condición de Exhaustividad.

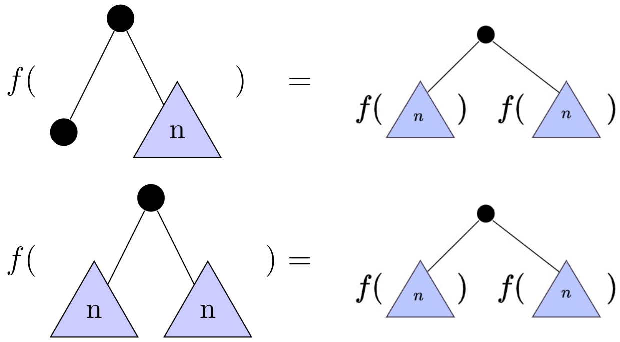
[2]Los árboles creados por las relgas 1 y 2 son de alturas diferentes, en particular en la regla 2 se aumenta siempre la altura en 1, garantizando que no se pueden construir dos árboles con la misma altura, en particular dos árboles iguales.

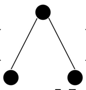
[3]Análogo a N_1.

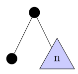
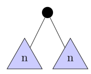
IV.

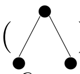
$f : N_3 \to N_2$

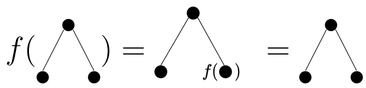
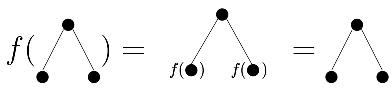
$f(\bullet) = \bullet$

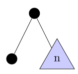
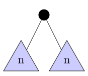


Notar que, con la definición de N_3, existen dos caminos para construir el elemento (), ya sea con la aplicación sucesiva de las reglas 1 y 2 o de la 1 y 3. Indicando que N_3 no es libre. Verifiquemos entonces que f está bien definida y si cumple o no con las condiciones de suficiencia.

Exhaustividad La cumple, todo árbol en N_3 es de la forma: $\bullet$,  o  .

No superposición No Superposición, es decir, que ningún elemento del dominio, a través de la función, va a parar a elementos distintos del codominio. En esta condición debemos tener especial cuidado cuando nuestro lenguaje no es libre. En el caso de N_3, $f($  $)$ puede ser evaluado en dos de las reglas de la definición de f. Basta con verificar que en ambos casos son el mismo para la definición de f propuesta:

- 
- 

Terminación Se da la misma noción de orden que en los casos anteriores. Sobrecargando el operador de desigualdad podemos decir que  $<$  y  $<$ , para la segunda y tercer ecución, respectivamente.