6 Funciones recursivas generales I

José de Jesús Lavalle Martínez

Benemérita Universidad Autónoma de Puebla
Facultad de Ciencias de la Computación
Computabilidad CCOS 257

Contenido

Motivación

- En el capítulo anterior vimos una visión general de varias formalizaciones posibles del concepto de calculabilidad efectiva.

Motivación

- En este capítulo, nos centramos en una de ellas: la recursividad primitiva y la búsqueda, que nos dan la clase de funciones parciales recursivas generales.

Motivación

- En particular, desarrollamos herramientas para mostrar que ciertas funciones están en esta clase.

Motivación

- Estas herramientas se utilizarán en el Capítulo 3, donde estudiaremos la computabilidad mediante programas de máquinas de registros.

Funciones recursivas primitivas I

- Las funciones recursivas primitivas se han definido en el capítulo anterior como las funciones sobre $\mathbb{N}$ que se pueden construir a partir de funciones cero

$$f(x_1, \ldots, x_k) = 0.$$

Funciones recursivas primitivas I

- Las funciones recursivas primitivas se han definido en el capítulo anterior como las funciones sobre $\mathbb{N}$ que se pueden construir a partir de funciones cero

$$f(x_1, \ldots, x_k) = 0.$$

- la función sucesor

$$S(x) = x + 1.$$

Funciones recursivas primitivas I

- Las funciones recursivas primitivas se han definido en el capítulo anterior como las funciones sobre $\mathbb{N}$ que se pueden construir a partir de funciones cero

$$f(x_1, \ldots, x_k) = 0.$$

- la función sucesor

$$S(x) = x + 1.$$

- las funciones proyección

$$I_n^k(x_1, \ldots, x_k) = x_n,$$

Funciones recursivas primitivas I

- Las funciones recursivas primitivas se han definido en el capítulo anterior como las funciones sobre $\mathbb{N}$ que se pueden construir a partir de funciones cero

$$f(x_1, \ldots, x_k) = 0.$$

- la función sucesor

$$S(x) = x + 1.$$

- las funciones proyección

$$I_n^k(x_1, \ldots, x_k) = x_n,$$

- usando (cero o más veces) composición

$$h(\vec{x}) = f(g_1(\vec{x}), \ldots, g_n(\vec{x}))$$

Funciones recursivas primitivas I

- Las funciones recursivas primitivas se han definido en el capítulo anterior como las funciones sobre $\mathbb{N}$ que se pueden construir a partir de funciones cero

$$f(x_1, \ldots, x_k) = 0.$$

- la función sucesor

$$S(x) = x + 1.$$

- las funciones proyección

$$I_n^k(x_1, \ldots, x_k) = x_n,$$

- usando (cero o más veces) composición

$$h(\vec{x}) = f(g_1(\vec{x}), \ldots, g_n(\vec{x}))$$

- y recursión primitiva

$$\begin{aligned} h(\vec{x}, 0) &= f(\vec{x}) \\ h(\vec{x}, y+1) &= g(h(\vec{x}, y), \vec{x}, y) \end{aligned}$$

Funciones recursivas primitivas I

- Las funciones recursivas primitivas se han definido en el capítulo anterior como las funciones sobre $\mathbb{N}$ que se pueden construir a partir de funciones cero

$$f(x_1, \ldots, x_k) = 0.$$

- la función sucesor

$$S(x) = x + 1.$$

- las funciones proyección

$$I_n^k(x_1, \ldots, x_k) = x_n,$$

- usando (cero o más veces) composición

$$h(\vec{x}) = f(g_1(\vec{x}), \ldots, g_n(\vec{x}))$$

- y recursión primitiva

$$h(\vec{x}, 0) = f(\vec{x})$$
$$h(\vec{x}, y + 1) = g(h(\vec{x}, y), \vec{x}, y)$$

- donde $\vec{x}$ puede ser vacía

$$h(0) = m$$
$$h(y + 1) = g(h(y), y)$$

Ejemplo 1

- Supongamos que nos dan el número $m = 1$ y la función $g(w, y) = w \cdot (y + 1)$.

Ejemplo 1

- Supongamos que nos dan el número $m = 1$ y la función $g(w, y) = w \cdot (y + 1)$.
- Entonces la función h obtenida por recursión primitiva de g usando m es la función dada por el par de ecuaciones

$$
\begin{aligned}
h(0) &= m = 1 \\
h(y+1) &= g(h(y), y) = h(y) \cdot (y+1).
\end{aligned}
$$

Ejemplo 1

- Supongamos que nos dan el número $m = 1$ y la función $g(w, y) = w \cdot (y + 1)$.
- Entonces la función h obtenida por recursión primitiva de g usando m es la función dada por el par de ecuaciones

$$h(0) = m = 1$$
$$h(y + 1) = g(h(y), y) = h(y) \cdot (y + 1).$$

- Usando este par de ecuaciones, podemos proceder a calcular los valores de la función h:

$$\begin{aligned} h(0) &= m = 1 \\ h(1) &= g(h(0), 0) = g(1, 0) = 1 \\ h(2) &= g(h(1), 1) = g(1, 1) = 2 \\ h(3) &= g(h(2), 2) = g(2, 2) = 6 \\ h(4) &= g(h(3), 3) = g(6, 3) = 24 \end{aligned}$$

Ejemplo 1

- Supongamos que nos dan el número $m = 1$ y la función $g(w, y) = w \cdot (y + 1)$.
- Entonces la función h obtenida por recursión primitiva de g usando m es la función dada por el par de ecuaciones

$$\begin{aligned} h(0) &= m = 1 \\ h(y+1) &= g(h(y), y) = h(y) \cdot (y+1). \end{aligned}$$

- Usando este par de ecuaciones, podemos proceder a calcular los valores de la función h:

$$\begin{aligned} h(0) &= m = 1 \\ h(1) &= g(h(0), 0) = g(1, 0) = 1 \\ h(2) &= g(h(1), 1) = g(1, 1) = 2 \\ h(3) &= g(h(2), 2) = g(2, 2) = 6 \\ h(4) &= g(h(3), 3) = g(6, 3) = 24 \end{aligned}$$

- Para calcular $h(4)$, primero necesitamos saber $h(3)$, y encontrar que necesitamos $h(2)$, y así sucesivamente.

Ejemplo 1

- Supongamos que nos dan el número $m = 1$ y la función $g(w, y) = w \cdot (y + 1)$.
- Entonces la función h obtenida por recursión primitiva de g usando m es la función dada por el par de ecuaciones

$$\begin{aligned} h(0) &= m = 1 \\ h(y+1) &= g(h(y), y) = h(y) \cdot (y+1). \end{aligned}$$

- Usando este par de ecuaciones, podemos proceder a calcular los valores de la función h:

$$\begin{aligned} h(0) &= m = 1 \\ h(1) &= g(h(0), 0) = g(1, 0) = 1 \\ h(2) &= g(h(1), 1) = g(1, 1) = 2 \\ h(3) &= g(h(2), 2) = g(2, 2) = 6 \\ h(4) &= g(h(3), 3) = g(6, 3) = 24 \end{aligned}$$

- Para calcular $h(4)$, primero necesitamos saber $h(3)$, y encontrar que necesitamos $h(2)$, y así sucesivamente.
- La función h en este ejemplo es, por supuesto, más conocida como función factorial, $h(x) = x!$.

Funciones recursivas primitivas II

- Debería quedar bastante claro que dado cualquier número m y cualquier función g de aridad dos, existe una función única h obtenida por recursión primitiva de g usando m.

Funciones recursivas primitivas II

- Debería quedar bastante claro que dado cualquier número m y cualquier función g de aridad dos, existe una función única h obtenida por recursión primitiva de g usando m.
- Es la función h la que calculamos como en el ejemplo anterior.

- Debería quedar bastante claro que dado cualquier número m y cualquier función g de aridad dos, existe una función única h obtenida por recursión primitiva de g usando m.
- Es la función h la que calculamos como en el ejemplo anterior.
- De manera similar, dada una función f de aridad k y una función g de aridad $(k+2)$, existe una función h única de aridad $(k+1)$ que se obtiene mediante recursión primitiva de f y g.

Funciones recursivas primitivas II

- Debería quedar bastante claro que dado cualquier número m y cualquier función g de aridad dos, existe una función única h obtenida por recursión primitiva de g usando m.
- Es la función h la que calculamos como en el ejemplo anterior.
- De manera similar, dada una función f de aridad k y una función g de aridad $(k+2)$, existe una función h única de aridad $(k+1)$ que se obtiene mediante recursión primitiva de f y g.
- Es decir, h es la función dada por el par de ecuaciones

$$\begin{aligned} h(\vec{x}, 0) &= f(\vec{x}) \\ h(\vec{x}, y+1) &= g(h(\vec{x}, y), \vec{x}, y). \end{aligned}$$

Funciones recursivas primitivas II

- Debería quedar bastante claro que dado cualquier número m y cualquier función g de aridad dos, existe una función única h obtenida por recursión primitiva de g usando m.
- Es la función h la que calculamos como en el ejemplo anterior.
- De manera similar, dada una función f de aridad k y una función g de aridad $(k+2)$, existe una función h única de aridad $(k+1)$ que se obtiene mediante recursión primitiva de f y g.
- Es decir, h es la función dada por el par de ecuaciones

$$h(\vec{x}, 0) = f(\vec{x})$$
$$h(\vec{x}, y+1) = g(h(\vec{x}, y), \vec{x}, y).$$

- Además, si f y g son funciones totales, entonces h también será total.

Ejemplo 2

- Considere la función de suma $h(x, y) = x + y$.

Ejemplo 2

- Considere la función de suma $h(x, y) = x + y$.
- Para cualquier x fija, su valor en $y + 1$ (es decir, $x + y + 1$) se puede obtener a partir de su valor en y (es decir, $x + y$) con el simple paso de sumar uno:

$$x + 0 = x$$
$$x + (y + 1) = (x + y) + 1$$

Ejemplo 2

- Considere la función de suma $h(x, y) = x + y$.
- Para cualquier x fija, su valor en $y + 1$ (es decir, $x + y + 1$) se puede obtener a partir de su valor en y (es decir, $x + y$) con el simple paso de sumar uno:

$$x + 0 = x$$
$$x + (y + 1) = (x + y) + 1$$

- Este par de ecuaciones muestra que la suma se obtiene mediante recursividad primitiva a partir de las funciones $f(x) = x$ y $g(w, x, y) = w + 1$.

Ejemplo 2

- Considere la función de suma $h(x, y) = x + y$.
- Para cualquier x fija, su valor en $y + 1$ (es decir, $x + y + 1$) se puede obtener a partir de su valor en y (es decir, $x + y$) con el simple paso de sumar uno:

$$x + 0 = x$$
$$x + (y + 1) = (x + y) + 1$$

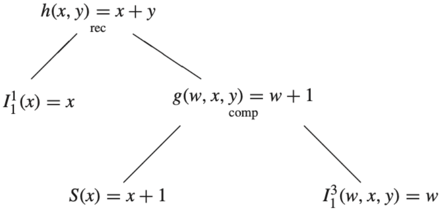
- Este par de ecuaciones muestra que la suma se obtiene mediante recursividad primitiva a partir de las funciones $f(x) = x$ y $g(w, x, y) = w + 1$.
- Estas funciones f y g son recursivas primitivas; f es la función de proyección I_1^1 y g se obtiene por composición de la función sucesor y I_1^3.

Ejemplo 2

- Considere la función de suma $h(x, y) = x + y$.
- Para cualquier x fija, su valor en $y + 1$ (es decir, $x + y + 1$) se puede obtener a partir de su valor en y (es decir, $x + y$) con el simple paso de sumar uno:

$$
\begin{aligned}
x + 0 &= x \\
x + (y + 1) &= (x + y) + 1
\end{aligned}
$$

- Este par de ecuaciones muestra que la suma se obtiene mediante recursividad primitiva a partir de las funciones $f(x) = x$ y $g(w, x, y) = w + 1$.
- Estas funciones f y g son recursivas primitivas; f es la función de proyección I_1^1 y g se obtiene por composición de la función sucesor y I_1^3.
- Al juntar estas observaciones, podemos formar un árbol que muestra cómo se construye la suma a partir de las funciones iniciales mediante composición y recursividad primitiva:



- De manera más general, para cualquier función recursiva primitiva h, podemos usar un árbol etiquetado ("árbol de construcción") para ilustrar exactamente cómo se construye h, como en el ejemplo de la suma.

- De manera más general, para cualquier función recursiva primitiva h, podemos usar un árbol etiquetado ("árbol de construcción") para ilustrar exactamente cómo se construye h, como en el ejemplo de la suma.
- En el vértice superior (raíz), colocamos h. En cada vértice minimal (una hoja), tenemos una función inicial: la función sucesora, una función cero o una función de proyección.

Funciones recursivas primitivas III

- De manera más general, para cualquier función recursiva primitiva h, podemos usar un árbol etiquetado ("árbol de construcción") para ilustrar exactamente cómo se construye h, como en el ejemplo de la suma.
- En el vértice superior (raíz), colocamos h. En cada vértice minimal (una hoja), tenemos una función inicial: la función sucesora, una función cero o una función de proyección.
- En cada uno de los demás vértices, mostramos una aplicación de composición o una aplicación de recursividad primitiva.

Funciones recursivas primitivas IV

- Una aplicación de composición.

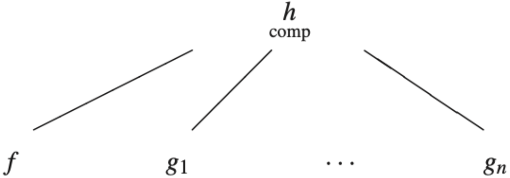
$$h(\vec{x}) = f(g_1(\vec{x}), \ldots, g_n(\vec{x}))$$

Funciones recursivas primitivas IV

- Una aplicación de composición.

$$h(\vec{x}) = f(g_1(\vec{x}), \ldots, g_n(\vec{x}))$$

- se puede ilustrar en un árbol mediante un vértice con $(n+1)$ ramificaciones:

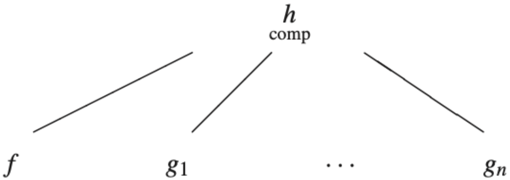


Funciones recursivas primitivas IV

- Una aplicación de composición.

$$h(\vec{x}) = f(g_1(\vec{x}), \ldots, g_n(\vec{x}))$$

- se puede ilustrar en un árbol mediante un vértice con $(n+1)$ ramificaciones:



- Aquí f debe ser una función de aridad n y $g_1, \ldots, g_n$ todas deben tener la misma aridad que h.

Funciones recursivas primitivas V

- Una aplicación de recursividad primitiva para obtener una función h de aridad $(k+1)$

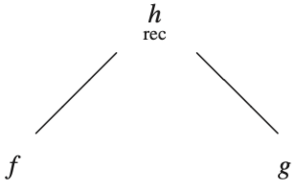
$$\begin{aligned} h(\vec{x},0) &= f(\vec{x}) \\ h(\vec{x},y+1) &= g(h(\vec{x},y),\vec{x},y) \end{aligned}$$

Funciones recursivas primitivas V

- Una aplicación de recursividad primitiva para obtener una función h de aridad $(k+1)$

$$h(\vec{x}, 0) = f(\vec{x})$$
$$h(\vec{x}, y+1) = g(h(\vec{x}, y), \vec{x}, y)$$

- Puede ser ilustrada por un vértice con ramificación binaria:

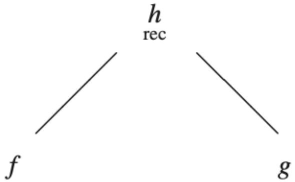


Funciones recursivas primitivas V

- Una aplicación de recursividad primitiva para obtener una función h de aridad $(k+1)$

$$h(\vec{x}, 0) = f(\vec{x})$$
$$h(\vec{x}, y+1) = g(h(\vec{x}, y), \vec{x}, y)$$

- Puede ser ilustrada por un vértice con ramificación binaria:



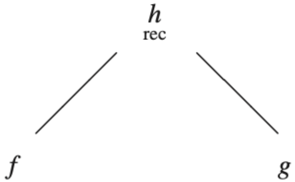
- Tenga en cuenta que si f es de aridad k entoces g debe tener aridad $(k+2)$ y h tendrá aridad $(k+1)$, como ya se había mencionado.

Funciones recursivas primitivas V

- Una aplicación de recursividad primitiva para obtener una función h de aridad $(k+1)$

$$h(\vec{x}, 0) = f(\vec{x})$$
$$h(\vec{x}, y+1) = g(h(\vec{x}, y), \vec{x}, y)$$

- Puede ser ilustrada por un vértice con ramificación binaria:



- Tenga en cuenta que si f es de aridad k entoces g debe tener aridad $(k+2)$ y h tendrá aridad $(k+1)$, como ya se había mencionado.
- Por ejemplo, si h es una función de aridad dos, entonces g debe ser una función de aridad tres y f debe ser una función de aridad uno.

Funciones recursivas primitivas VI

- El caso $k = 0$, donde una función h de aridad uno se obtiene mediante recursividad primitiva a partir de una función g de aridad dos usando el número m

$$h(m) = m$$
$$h(x+1) = g(h(x), x),$$

- El caso $k = 0$, donde una función h de aridad uno se obtiene mediante recursividad primitiva a partir de una función g de aridad dos usando el número m

$$
\begin{aligned}
h(m) &= m \\
h(x+1) &= g(h(x), x),
\end{aligned}
$$

- Puede ser ilustrado por un vértice con ramificación unaria.

- El caso $k = 0$, donde una función h de aridad uno se obtiene mediante recursividad primitiva a partir de una función g de aridad dos usando el número m

$$
\begin{aligned}
h(m) &= m \\
h(x+1) &= g(h(x), x),
\end{aligned}
$$

- Puede ser ilustrado por un vértice con ramificación unaria.

- En ambas formas de recursividad primitiva ($k > 0$ y $k = 0$), la característica clave es que el valor de la función en un número $t + 1$ se puede obtener de alguna manera a partir de su valor en t.

Funciones recursivas primitivas VI

- El caso $k = 0$, donde una función h de aridad uno se obtiene mediante recursividad primitiva a partir de una función g de aridad dos usando el número m

$$
\begin{aligned}
h(m) &= m \\
h(x+1) &= g(h(x), x),
\end{aligned}
$$

- Puede ser ilustrado por un vértice con ramificación unaria.

- En ambas formas de recursividad primitiva ($k > 0$ y $k = 0$), la característica clave es que el valor de la función en un número $t+1$ se puede obtener de alguna manera a partir de su valor en t.
- El papel de g es explicar cómo.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por “inducción estructural”.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.
- Para los dos pasos inductivos, observamos que la composición de funciones totales produce una función total, y la recursión primitiva aplicada a funciones totales produce una función total.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.
- Para los dos pasos inductivos, observamos que la composición de funciones totales produce una función total, y la recursión primitiva aplicada a funciones totales produce una función total.
- Entonces, para cualquier función recursiva primitiva, podemos avanzar hasta su árbol de construcción.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.
- Para los dos pasos inductivos, observamos que la composición de funciones totales produce una función total, y la recursión primitiva aplicada a funciones totales produce una función total.
- Entonces, para cualquier función recursiva primitiva, podemos avanzar hasta su árbol de construcción.
- En las hojas del árbol tenemos funciones totales.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.
- Para los dos pasos inductivos, observamos que la composición de funciones totales produce una función total, y la recursión primitiva aplicada a funciones totales produce una función total.
- Entonces, para cualquier función recursiva primitiva, podemos avanzar hasta su árbol de construcción.
- En las hojas del árbol tenemos funciones totales.
- Y cada vez que nos movemos a un vértice superior, todavía tenemos una función total.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.
- Para los dos pasos inductivos, observamos que la composición de funciones totales produce una función total, y la recursión primitiva aplicada a funciones totales produce una función total.
- Entonces, para cualquier función recursiva primitiva, podemos avanzar hasta su árbol de construcción.
- En las hojas del árbol tenemos funciones totales.
- Y cada vez que nos movemos a un vértice superior, todavía tenemos una función total.
- Finalmente, llegamos a la raíz superior y concluimos que la función que se está construyendo es total.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.
- Para los dos pasos inductivos, observamos que la composición de funciones totales produce una función total, y la recursión primitiva aplicada a funciones totales produce una función total.
- Entonces, para cualquier función recursiva primitiva, podemos avanzar hasta su árbol de construcción.
- En las hojas del árbol tenemos funciones totales.
- Y cada vez que nos movemos a un vértice superior, todavía tenemos una función total.
- Finalmente, llegamos a la raíz superior y concluimos que la función que se está construyendo es total.
- A continuación queremos crear un catálogo de funciones recursivas primitivas básicas.

Funciones recursivas primitivas VII

- Toda función recursiva primitiva es total.
- Podemos ver esto por "inducción estructural".
- Para la base, todas las funciones iniciales (las funciones cero, la función sucesor y las funciones de proyección) son totales.
- Para los dos pasos inductivos, observamos que la composición de funciones totales produce una función total, y la recursión primitiva aplicada a funciones totales produce una función total.
- Entonces, para cualquier función recursiva primitiva, podemos avanzar hasta su árbol de construcción.
- En las hojas del árbol tenemos funciones totales.
- Y cada vez que nos movemos a un vértice superior, todavía tenemos una función total.
- Finalmente, llegamos a la raíz superior y concluimos que la función que se está construyendo es total.
- A continuación queremos crear un catálogo de funciones recursivas primitivas básicas.
- Estos elementos del catálogo se pueden utilizar luego como piezas listas para usar para la posterior creación de otras funciones recursivas primitivas.

Ejemplo 3

- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.

Ejemplo 3

- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.
- El símbolo "$\mapsto$" se lee "se mapea a".

Ejemplo 3

- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.
- El símbolo "$\mapsto$" se lee "se mapea a".
- El símbolo nos brinda una forma muy conveniente de nombrar funciones.

Ejemplo 3

- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.
- El símbolo "$\mapsto$" se lee "se mapea a".
- El símbolo nos brinda una forma muy conveniente de nombrar funciones.
- Por ejemplo, la función de elevar al cuadrado puede denominarse mediante la frase larga "la función que dado un número, lo eleva al cuadrado".

Ejemplo 3

- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.
- El símbolo "$\mapsto$" se lee "se mapea a".
- El símbolo nos brinda una forma muy conveniente de nombrar funciones.
- Por ejemplo, la función de elevar al cuadrado puede denominarse mediante la frase larga "la función que dado un número, lo eleva al cuadrado".
- Es matemáticamente conveniente utilizar una letra (como x o t).

Ejemplo 3

- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.
- El símbolo "$\mapsto$" se lee "se mapea a".
- El símbolo nos brinda una forma muy conveniente de nombrar funciones.
- Por ejemplo, la función de elevar al cuadrado puede denominarse mediante la frase larga "la función que dado un número, lo eleva al cuadrado".
- Es matemáticamente conveniente utilizar una letra (como x o t).
- Esto nos lleva a los nombres "la función cuyo valor en x es x^2 " o "la función cuyo valor en t es t^2".

Ejemplo 3

- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.
- El símbolo "$\mapsto$" se lee "se mapea a".
- El símbolo nos brinda una forma muy conveniente de nombrar funciones.
- Por ejemplo, la función de elevar al cuadrado puede denominarse mediante la frase larga "la función que dado un número, lo eleva al cuadrado".
- Es matemáticamente conveniente utilizar una letra (como x o t).
- Esto nos lleva a los nombres "la función cuyo valor en x es x^2 " o "la función cuyo valor en t es t^2".
- De manera más compacta, estos nombres se pueden escribir en símbolos como "$x \mapsto x^2$" o "$t \mapsto t^2$".

Ejemplo 3

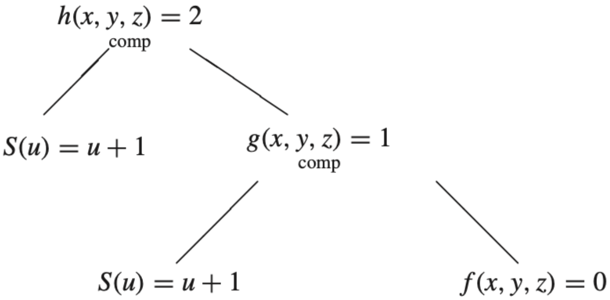
- Ya se ha demostrado que la suma $< x, y > \mapsto x + y$ es recursiva primitiva.
- El símbolo "$\mapsto$" se lee "se mapea a".
- El símbolo nos brinda una forma muy conveniente de nombrar funciones.
- Por ejemplo, la función de elevar al cuadrado puede denominarse mediante la frase larga "la función que dado un número, lo eleva al cuadrado".
- Es matemáticamente conveniente utilizar una letra (como x o t).
- Esto nos lleva a los nombres "la función cuyo valor en x es x^2 " o "la función cuyo valor en t es t^2".
- De manera más compacta, estos nombres se pueden escribir en símbolos como "$x \mapsto x^2$" o "$t \mapsto t^2$".
- La letra x o t es una variable ficticia; podemos usar cualquier letra aquí.

Ejemplo 4

- Cualquier función constante $\vec{x} \mapsto k$ se puede obtener aplicando composición k veces a la función sucesora y a la función cero $\vec{x} \mapsto 0$.

Ejemplo 4

- Cualquier función constante $\vec{x} \mapsto k$ se puede obtener aplicando composición k veces a la función sucesora y a la función cero $\vec{x} \mapsto 0$.
- Por ejemplo, la función de aridad tres que constantemente toma el valor 2 se puede construir mediante el siguiente árbol:



Ejemplo 5

- Para la multiplicación $< x, y > \mapsto x \times y$, primero observamos que

$$
\begin{aligned}
x \times 0 &= 0 \\
x \times (y+1) &= (x \times y) + x.
\end{aligned}
$$

Ejemplo 5

- Para la multiplicación $< x, y > \mapsto x \times y$, primero observamos que

$$
\begin{aligned}
x \times 0 &= 0 \\
x \times (y + 1) &= (x \times y) + x.
\end{aligned}
$$

- Esto muestra que la multiplicación se obtiene mediante recursión primitiva de las funciones $x \mapsto 0$ y $< w, x, y > \mapsto w + x$.

Ejemplo 5

- Para la multiplicación $< x, y > \mapsto x \times y$, primero observamos que

$$
\begin{aligned}
x \times 0 &= 0 \\
x \times (y+1) &= (x \times y) + x.
\end{aligned}
$$

- Esto muestra que la multiplicación se obtiene mediante recursión primitiva de las funciones $x \mapsto 0$ y $< w, x, y > \mapsto w + x$.
- Esta última función se obtiene mediante composición aplicada a funciones de suma y proyección.

Ejemplo 5

- Para la multiplicación $< x, y > \mapsto x \times y$, primero observamos que

$$x \times 0 = 0$$
$$x \times (y + 1) = (x \times y) + x.$$

- Esto muestra que la multiplicación se obtiene mediante recursión primitiva de las funciones $x \mapsto 0$ y $< w, x, y > \mapsto w + x$.
- Esta última función se obtiene mediante composición aplicada a funciones de suma y proyección.
- Ahora podemos concluir que cualquier función polinomial con coeficientes positivos es recursiva primitiva.

Ejemplo 5

- Para la multiplicación $< x, y >\mapsto x \times y$, primero observamos que

$$x \times 0 = 0$$
$$x \times (y + 1) = (x \times y) + x.$$

- Esto muestra que la multiplicación se obtiene mediante recursión primitiva de las funciones $x \mapsto 0$ y $< w, x, y >\mapsto w + x$.
- Esta última función se obtiene mediante composición aplicada a funciones de suma y proyección.
- Ahora podemos concluir que cualquier función polinomial con coeficientes positivos es recursiva primitiva.
- Por ejemplo, podemos ver que la función $p(x, y) = x^2y + 5xy + 3y^3$ es recursiva primitiva al aplicar repetidamente los Ejemplos 3, 4 y 5.

Ejemplo 6

- La exponenciación $< x, y > \mapsto x^y$ es similar:

Ejemplo 6

- La exponenciación $< x, y > \mapsto x^y$ es similar:
- $$x^0 = 1$$
$$x^{y+1} = x^y \times x.$$

Ejemplo 7

- La exponenciación $< x, y > \mapsto y^x$ se obtiene de la función anterior mediante composición con funciones de proyección.

Ejemplo 7

- La exponenciación $< x, y > \mapsto y^x$ se obtiene de la función anterior mediante composición con funciones de proyección.
- Las funciones de los Ejemplos 6 y 7 son funciones diferentes; asignan valores diferentes a $< 2, 3 >$.

Ejemplo 7

- La exponenciación $< x, y > \mapsto y^x$ se obtiene de la función anterior mediante composición con funciones de proyección.
- Las funciones de los Ejemplos 6 y 7 son funciones diferentes; asignan valores diferentes a $< 2, 3 >$.
- El hecho de que coincidan en $< 2, 4 >$ es un accidente.

Ejemplo 7

- La exponenciación $< x, y > \mapsto y^x$ se obtiene de la función anterior mediante composición con funciones de proyección.
- Las funciones de los Ejemplos 6 y 7 son funciones diferentes; asignan valores diferentes a $< 2, 3 >$.
- El hecho de que coincidan en $< 2, 4 >$ es un accidente.
- Deberíamos generalizar esta observación.

Ejemplo 7

- La exponenciación $< x, y > \mapsto y^x$ se obtiene de la función anterior mediante composición con funciones de proyección.
- Las funciones de los Ejemplos 6 y 7 son funciones diferentes; asignan valores diferentes a $< 2, 3 >$.
- El hecho de que coincidan en $< 2, 4 >$ es un accidente.
- Deberíamos generalizar esta observación.
- Por ejemplo, si f es recursiva primitiva y g está definido por la ecuación

$$g(x, y, z) = f(y, 3, x, x)$$

entonces g también es recursiva primitiva, y se obtiene por composición a partir de f y funciones de proyección y constantes.

Ejemplo 7

- La exponenciación $< x, y > \mapsto y^x$ se obtiene de la función anterior mediante composición con funciones de proyección.
- Las funciones de los Ejemplos 6 y 7 son funciones diferentes; asignan valores diferentes a $< 2, 3 >$.
- El hecho de que coincidan en $< 2, 4 >$ es un accidente.
- Deberíamos generalizar esta observación.
- Por ejemplo, si f es recursiva primitiva y g está definido por la ecuación

$$g(x, y, z) = f(y, 3, x, x)$$

  entonces g también es recursiva primitiva, y se obtiene por composición a partir de f y funciones de proyección y constantes.
- Diremos en esta situación que g se obtiene de f mediante *transformación explícita*.

Ejemplo 7

- La exponenciación $< x, y > \mapsto y^x$ se obtiene de la función anterior mediante composición con funciones de proyección.
- Las funciones de los Ejemplos 6 y 7 son funciones diferentes; asignan valores diferentes a $< 2, 3 >$.
- El hecho de que coincidan en $< 2, 4 >$ es un accidente.
- Deberíamos generalizar esta observación.
- Por ejemplo, si f es recursiva primitiva y g está definido por la ecuación

  $$g(x, y, z) = f(y, 3, x, x)$$

  entonces g también es recursiva primitiva, y se obtiene por composición a partir de f y funciones de proyección y constantes.
- Diremos en esta situación que g se obtiene de f mediante *transformación explícita*.
- La transformación explícita permite cambiar variables, repetir variables, omitir variables y sustituir constantes.

Ejemplo 8

- La función factorial $x!$ satisface el par de ecuaciones recursivas

$$
\begin{aligned}
0! &= 1 \\
(x+1)! &= x! \times (x+1).
\end{aligned}
$$

Ejemplo 8

- La función factorial $x!$ satisface el par de ecuaciones recursivas

$$\begin{aligned} 0! &= 1 \\ (x+1)! &= x! \times (x+1). \end{aligned}$$

- De este par de ecuaciones se deduce que la función factorial se obtiene mediante recursividad primitiva (usando el Ejemplo 3) de la función $g(w,x) = w \cdot (x+1)$.

Ejemplo 9

- La función predecesor $pred(x) = x - 1$ (excepto que $pred(0) = 0$) se obtiene mediante recursión primitiva de I_2^2

Ejemplo 9

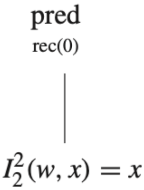
- La función predecesor $pred(x) = x - 1$ (excepto que $pred(0) = 0$) se obtiene mediante recursión primitiva de I_2^2
- $$\begin{aligned} pred(0) &= 0 \\ pred(x+1) &= x. \end{aligned}$$

Ejemplo 9

- La función predecesor $pred(x) = x - 1$ (excepto que $pred(0) = 0$) se obtiene mediante recursión primitiva de I_2^2
-

$$pred(0) = 0$$
$$pred(x+1) = x.$$

- Este par de ecuaciones conduce al árbol:



Ejemplo 10

- Defina la función de resta propia $x \dot{-} y$ mediante la ecuación $x \dot{-} y = \text{máx}(x - y, 0)$.

Ejemplo 10

- Defina la función de resta propia $x \dot{-} y$ mediante la ecuación $x \dot{-} y = \max(x - y, 0)$.
- Esta función es recursiva primitiva:

$$x \dot{-} 0 = x$$

$$x \dot{-} (y + 1) = pred(x \dot{-} y)$$

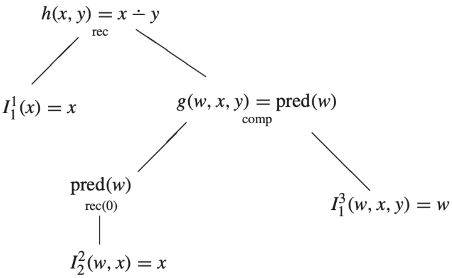
Ejemplo 10

- Defina la función de resta propia $x \dot{-} y$ mediante la ecuación $x \dot{-} y = \text{máx}(x - y, 0)$.
- Esta función es recursiva primitiva:

$$x \dot{-} 0 = x$$

$$x \dot{-} (y + 1) = pred(x \dot{-} y)$$

- Este par de ecuaciones de recursividad produce el siguiente árbol de construcción:

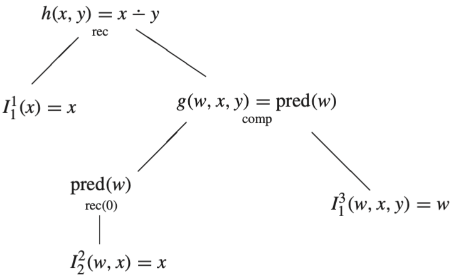


Ejemplo 10

- Defina la función de resta propia $x \dot{-} y$ mediante la ecuación $x \dot{-} y = \max(x - y, 0)$.
- Esta función es recursiva primitiva:

$$x \dot{-} 0 = x$$
$$x \dot{-} (y+1) = pred(x \dot{-} y)$$

- Este par de ecuaciones de recursividad produce el siguiente árbol de construcción:



- Por cierto, el símbolo $\dot{-}$ a veces se lee como "monus".

Ejemplo 11

- Supongamos que f es recursiva primitiva y defina las funciones s y p mediante las ecuaciones

$$s(\vec{x}, y) = \sum_{t<y} f(\vec{x}, t) \quad \text{y} \quad p(\vec{x}, y) = \prod_{t<y} f(\vec{x}, t)$$

(sujeto a las convenciones estándar para la suma vacía $\sum_{t<0} f(\vec{x}, t) = 0$ y el producto vacío $\prod_{t<0} f(\vec{x}, t) = 1$).

Ejemplo 11

- Supongamos que f es recursiva primitiva y defina las funciones s y p mediante las ecuaciones

$$s(\vec{x}, y) = \sum_{t<y} f(\vec{x}, t) \quad \text{y} \quad p(\vec{x}, y) = \prod_{t<y} f(\vec{x}, t)$$

(sujeto a las convenciones estándar para la suma vacía $\sum_{t<0} f(\vec{x}, t) = 0$ y el producto vacío $\prod_{t<0} f(\vec{x}, t) = 1$).
- Entonces tanto s como p son recursivas primitivas.

Ejemplo 11

- Supongamos que f es recursiva primitiva y defina las funciones s y p mediante las ecuaciones

$$s(\vec{x}, y) = \sum_{t<y} f(\vec{x}, t) \quad \text{y} \quad p(\vec{x}, y) = \prod_{t<y} f(\vec{x}, t)$$

(sujeto a las convenciones estándar para la suma vacía $\sum_{t<0} f(\vec{x}, t) = 0$ y el producto vacío $\prod_{t<0} f(\vec{x}, t) = 1$).

- Entonces tanto s como p son recursivas primitivas.
- Para p, tenemos el par de ecuaciones:

$$\begin{aligned} p(\vec{x}, 0) &= 1 \\ p(\vec{x}, y+1) &= p(\vec{x}, y) \cdot f(\vec{x}, y) \end{aligned}$$

Ejemplo 12

- Defina la función z mediante la ecuación.

$$z(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{si } x > 0. \end{cases}$$

Ejemplo 12

- Defina la función z mediante la ecuación.

$$z(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{si } x > 0. \end{cases}$$

- Es decir, la función z busca si su entrada es cero y devuelve Sí (es decir, 1) si es cero; de lo contrario, devuelve No (es decir, 0).

Ejemplo 12

- Defina la función z mediante la ecuación.

$$z(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{si } x > 0. \end{cases}$$

- Es decir, la función z busca si su entrada es cero y devuelve Sí (es decir, 1) si es cero; de lo contrario, devuelve No (es decir, 0).
- La función z es recursiva primitiva.

Ejemplo 12

- Defina la función z mediante la ecuación.

$$z(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{si } x > 0. \end{cases}$$

- Es decir, la función z busca si su entrada es cero y devuelve Sí (es decir, 1) si es cero; de lo contrario, devuelve No (es decir, 0).
- La función z es recursiva primitiva.
- Podemos ver esto en la ecuación $z(x) = 0^x$.

Ejemplo 12

- Defina la función z mediante la ecuación.

$$z(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{si } x > 0. \end{cases}$$

- Es decir, la función z busca si su entrada es cero y devuelve Sí (es decir, 1) si es cero; de lo contrario, devuelve No (es decir, 0).
- La función z es recursiva primitiva.
- Podemos ver esto en la ecuación $z(x) = 0^x$.
- Más directamente, podemos verlo en la ecuación $z(x) = 1 \dot{-} x$.

Ejemplo 12

- Defina la función z mediante la ecuación.

$$z(x) = \begin{cases} 1 & \text{si } x = 0 \\ 0 & \text{si } x > 0. \end{cases}$$

- Es decir, la función z busca si su entrada es cero y devuelve Sí (es decir, 1) si es cero; de lo contrario, devuelve No (es decir, 0).
- La función z es recursiva primitiva.
- Podemos ver esto en la ecuación $z(x) = 0^x$.
- Más directamente, podemos verlo en la ecuación $z(x) = 1 \dot{-} x$.
- Y aún más directamente, podemos verlo en las ecuaciones de recursividad.

$$\begin{aligned} z(0) &= 1 \\ z(x+1) &= 0 \end{aligned}$$

mostrando que z se obtiene mediante recursividad primitiva (usando el Ejemplo 3) de la función $g(w, x) = 0$.