Matemática Discreta

Clase 8: Inducción estructural y recursión

Federico Olmedo y Alejandro Hevia

Departamento de Ciencias de la Computación
Universidad de Chile

Conjuntos inductivos

Los conjuntos finitos siempre se pueden definir *por extensión*, es decir enumerando todos sus elementos.

Conjuntos inductivos

Los conjuntos finitos siempre se pueden definir *por extensión*, es decir enumerando todos sus elementos. Un conjunto infinito, no.

Conjuntos inductivos

Los conjuntos finitos siempre se pueden definir *por extensión*, es decir enumerando todos sus elementos. Un conjunto infinito, no. Una manera muy común de definir conjuntos infinitos es inductivamente.

Conjuntos inductivos

Los conjuntos finitos siempre se pueden definir *por extensión*, es decir enumerando todos sus elementos. Un conjunto infinito, no. Una manera muy común de definir conjuntos infinitos es inductivamente.

Una definición inductiva de un conjunto se construye a partir de:

Conjuntos inductivos

Los conjuntos finitos siempre se pueden definir *por extensión*, es decir enumerando todos sus elementos. Un conjunto infinito, no. Una manera muy común de definir conjuntos infinitos es inductivamente.

Una definición inductiva de un conjunto se construye a partir de:

- Una o más reglas base, que determinan un subconjunto de elementos iniciales del conjunto en cuestión.

Conjuntos inductivos

Los conjuntos finitos siempre se pueden definir *por extensión*, es decir enumerando todos sus elementos. Un conjunto infinito, no. Una manera muy común de definir conjuntos infinitos es inductivamente.

Una definición inductiva de un conjunto se construye a partir de:

- Una o más reglas base, que determinan un subconjunto de elementos iniciales del conjunto en cuestión.
- Una o más reglas inductivas, que especifican cómo agregar nuevos elementos al conjunto a partir de elementos ya existentes.

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

- **Regla base:** $0 \in S$

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

- **Regla base:** $0 \in S$
- **Regla inductiva:** Si $n \in S$, entonces $n + 1 \in S$

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

- **Regla base:** $0 \in S$
- **Regla inductiva:** Si $n \in S$, entonces $n + 1 \in S$

Pregunta: ¿Qué conjunto representa S?

Los naturales como conjunto inductivo

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

- **Regla base:** $0 \in S$
- **Regla inductiva:** Si $n \in S$, entonces $n + 1 \in S$

Pregunta: ¿Qué conjunto representa S?

Los naturales como conjunto inductivo

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

- **Regla base:** $0 \in S$
- **Regla inductiva:** Si $n \in S$, entonces $n + 1 \in S$

Pregunta: ¿Qué conjunto representa S?

Pregunta: ¿Por qué es necesario el "sea S el menor conjunto que ..."?

Notación

Para describir las reglas de la definición inductiva se suele usar alternativamente una notación vertical, en forma de reglas de inferencia.

Notación

Para describir las reglas de la definición inductiva se suele usar alternativamente una notación vertical, en forma de reglas de inferencia.

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

$$\frac{}{0 \in S} \qquad\qquad \frac{n \in S}{n+1 \in S}$$

Notación

Para describir las reglas de la definición inductiva se suele usar alternativamente una notación vertical, en forma de reglas de inferencia.

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

$$\frac{}{0 \in S} \qquad\qquad \frac{n \in S}{n + 1 \in S}$$

- Lo que escribimos arriba de la línea horizontal son la(s) premisa(s) de la regla. Una regla puede no tener ninguna premisa, y en tal caso la línea horizontal se suele omitir.

Notación

Para describir las reglas de la definición inductiva se suele usar alternativamente una notación vertical, en forma de reglas de inferencia.

Ejemplo: Sea S el menor conjunto que satisface las siguientes reglas:

$$\frac{}{0 \in S} \qquad\qquad \frac{n \in S}{n+1 \in S}$$

- Lo que escribimos arriba de la línea horizontal son la(s) premisa(s) de la regla. Una regla puede no tener ninguna premisa, y en tal caso la línea horizontal se suele omitir.
- Lo que escribimos debajo de la línea horizontal es la conclusión de la regla

Ejercicios

Ejercicio: Explique qué conjuntos inductivos definen las siguientes reglas:

Ejercicios

Ejercicio: Explique qué conjuntos inductivos definen las siguientes reglas:

1. $$\frac{}{0 \in X} \qquad \frac{n \in X}{n + 2 \in X}$$

Ejercicios

Ejercicio: Explique qué conjuntos inductivos definen las siguientes reglas:

1. $$\frac{}{0 \in X} \qquad \frac{n \in X}{n+2 \in X}$$
2. $$\frac{}{1 \in Y} \qquad \frac{n \in Y}{3 \cdot n \in Y}$$

Ejercicios

Ejercicio: Explique qué conjuntos inductivos definen las siguientes reglas:

1. $$\frac{}{0 \in X} \qquad \frac{n \in X}{n + 2 \in X}$$
2. $$\frac{}{1 \in Y} \qquad \frac{n \in Y}{3 \cdot n \in Y}$$
3. $$\frac{}{(0, 1) \in Z} \qquad \frac{(m, n) \in Z}{(m + 1, 2 \cdot n) \in Z}$$

Principio de inducción estructural

La definición inductiva de un conjunto provee automáticamente un principio de inducción estructural, que permite probar propiedades sobre los elementos del conjunto.

Principio de inducción estructural

La definición inductiva de un conjunto provee automáticamente un principio de inducción estructural, que permite probar propiedades sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad\qquad \frac{n \in \mathbb{N}_0}{n+1 \in \mathbb{N}_0}$$

Principio de inducción estructural

La definición inductiva de un conjunto provee automáticamente un principio de inducción estructural, que permite probar propiedades sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad\qquad \frac{n \in \mathbb{N}_0}{n+1 \in \mathbb{N}_0}$$

El principio de inducción estructural sobre $\mathbb{N}_0$ dice que para probar que todo elementos de $\mathbb{N}_0$ satisface una propiedad P basta con

Principio de inducción estructural

La definición inductiva de un conjunto provee automáticamente un principio de inducción estructural, que permite probar propiedades sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad\qquad \frac{n \in \mathbb{N}_0}{n+1 \in \mathbb{N}_0}$$

El principio de inducción estructural sobre $\mathbb{N}_0$ dice que para probar que todo elementos de $\mathbb{N}_0$ satisface una propiedad P basta con

- **Caso base:** probar $P(0)$, y

Principio de inducción estructural

La definición inductiva de un conjunto provee automáticamente un principio de inducción estructural, que permite probar propiedades sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad\qquad \frac{n \in \mathbb{N}_0}{n+1 \in \mathbb{N}_0}$$

El principio de inducción estructural sobre $\mathbb{N}_0$ dice que para probar que todo elementos de $\mathbb{N}_0$ satisface una propiedad P basta con

- **Caso base:** probar $P(0)$, y
- **Caso inductivo:** asumiendo cierto $P(n)$, probar $P(n+1)$

Principio de inducción estructural

Ejemplo: Sea Z el menor conjunto que satisface las siguientes reglas:

$$\frac{}{(0,1) \in Z} \qquad\qquad \frac{(m,n) \in Z}{(m+1, 2 \cdot n) \in Z}$$

Principio de inducción estructural

Ejemplo: Sea Z el menor conjunto que satisface las siguientes reglas:

$$\frac{}{(0,1) \in Z} \qquad\qquad \frac{(m,n) \in Z}{(m+1, 2 \cdot n) \in Z}$$

El principio de inducción estructural sobre Z dice que para probar que todo elementos de Z satisface una propiedad P basta con

Principio de inducción estructural

Ejemplo: Sea Z el menor conjunto que satisface las siguientes reglas:

$$\frac{}{(0,1) \in Z} \qquad\qquad \frac{(m,n) \in Z}{(m+1, 2 \cdot n) \in Z}$$

El principio de inducción estructural sobre Z dice que para probar que todo elementos de Z satisface una propiedad P basta con

Principio de inducción estructural

Ejemplo: Sea Z el menor conjunto que satisface las siguientes reglas:

$$\frac{}{(0,1) \in Z} \qquad\qquad \frac{(m,n) \in Z}{(m+1, 2 \cdot n) \in Z}$$

El principio de inducción estructural sobre Z dice que para probar que todo elementos de Z satisface una propiedad P basta con

- **Caso base:** probar $P(0,1)$, y

Principio de inducción estructural

Ejemplo: Sea Z el menor conjunto que satisface las siguientes reglas:

$$\frac{}{(0,1) \in Z} \qquad\qquad \frac{(m,n) \in Z}{(m+1, 2 \cdot n) \in Z}$$

El principio de inducción estructural sobre Z dice que para probar que todo elementos de Z satisface una propiedad P basta con

- **Caso base:** probar $P(0,1)$, y
- **Caso inductivo:** asumiendo cierto $P(m,n)$, probar $P(m+1, 2 \cdot n)$

Esquema de recursion

La definición inductiva de un conjunto provee automáticamente un esquema de recursión, que permite definir funciones recursivas sobre los elementos del conjunto.

Esquema de recursion

La definición inductiva de un conjunto provee automáticamente un esquema de recursión, que permite definir funciones recursivas sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad \frac{n \in \mathbb{N}_0}{n + 1 \in \mathbb{N}_0}$$

Esquema de recursion

La definición inductiva de un conjunto provee automáticamente un esquema de recursión, que permite definir funciones recursivas sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad\qquad \frac{n \in \mathbb{N}_0}{n+1 \in \mathbb{N}_0}$$

El esquema de recursión sobre $\mathbb{N}_0$ dice que para definir una función f sobre el conjunto $\mathbb{N}_0$ debemos

Esquema de recursion

La definición inductiva de un conjunto provee automáticamente un esquema de recursión, que permite definir funciones recursivas sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad\qquad \frac{n \in \mathbb{N}_0}{n+1 \in \mathbb{N}_0}$$

El esquema de recursión sobre $\mathbb{N}_0$ dice que para definir una función f sobre el conjunto $\mathbb{N}_0$ debemos

- **Caso base:** definir $f(0)$, y

Esquema de recursion

La definición inductiva de un conjunto provee automáticamente un esquema de recursión, que permite definir funciones recursivas sobre los elementos del conjunto.

Ejemplo: Considere la definición inductiva del conjunto de los naturales dada por las siguientes reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad \frac{n \in \mathbb{N}_0}{n+1 \in \mathbb{N}_0}$$

El esquema de recursión sobre $\mathbb{N}_0$ dice que para definir una función f sobre el conjunto $\mathbb{N}_0$ debemos

- **Caso base:** definir $f(0)$, y
- **Caso inductivo:** asumiendo que conocemos el valor de $f(n)$, definir $f(n+1)$

Ejemplos de funciones recursivas

Ejemplo: Usando el esquema de recursión anterior, podemos definir la función factorial sobre $\mathbb{N}_0$ de la siguiente manera:

$$f(0) \;=\; 1 \qquad\qquad f(n+1) \;=\; (n+1)\cdot f(n)$$

Ejemplos de funciones recursivas

Ejemplo: Usando el esquema de recursión anterior, podemos definir la función factorial sobre $\mathbb{N}_0$ de la siguiente manera:

$$f(0) \;=\; 1 \qquad\qquad f(n+1) \;=\; (n+1)\cdot f(n)$$

Ejemplo: Usando el esquema de recursión anterior, podemos definir la suma S_n de 0 a n de la siguiente manera:

$$S_0 \;=\; 0 \qquad\qquad S_{n+1} \;=\; (n+1) + S_n$$

Ejemplos de funciones recursivas

Ejemplo: Usando el esquema de recursión anterior, podemos definir la función factorial sobre $\mathbb{N}_0$ de la siguiente manera:

$$f(0) = 1 \qquad\qquad f(n+1) = (n+1) \cdot f(n)$$

Ejemplo: Usando el esquema de recursión anterior, podemos definir la suma S_n de 0 a n de la siguiente manera:

$$S_0 = 0 \qquad\qquad S_{n+1} = (n+1) + S_n$$

Observación: las cláusulas recursivas de los ejemplos anteriores (y en general) se suelen escribir—equivalentemente—de la siguiente manera:

$$\begin{aligned} f(n) &= n \cdot f(n-1) \quad \forall n \geq 1 \\ S_n &= n + S_{n-1} \quad \forall n \geq 1 \end{aligned}$$

Ejemplos de funciones recursivas

Ejemplo: La llamada sucesión de Fibonacci puede definirse recursivamente de la siguiente manera:

$$F_0 = 0 \qquad F_1 = 1 \qquad F_{n+2} = F_{n+1} + F_n$$

Ejemplos de funciones recursivas

Ejemplo: La llamada sucesión de Fibonacci puede definirse recursivamente de la siguiente manera:

$$F_0 \;=\; 0 \qquad\qquad F_1 \;=\; 1 \qquad\qquad F_{n+2} \;=\; F_{n+1} + F_n$$

Observación: Esta definición no se adhiere al esquema de recursión anterior, sino al generado por las reglas:

$$\frac{}{0 \in \mathbb{N}_0} \qquad\qquad \frac{}{1 \in \mathbb{N}_0} \qquad\qquad \frac{n \in \mathbb{N}_0 \qquad n+1 \in \mathbb{N}_0}{n+2 \in \mathbb{N}_0}$$

que proveen una definición alternativa del conjunto de los naturales.

Aplicación: Listas

El conjunto $list(A)$ de listas sobre A puede definirse inductivamente de la siguiente manera:

Aplicación: Listas

El conjunto *list*(A) de listas sobre A puede definirse inductivamente de la siguiente manera:

- **Regla base:** la lista vacía nil pertenece a *list*(A), y

Aplicación: Listas

El conjunto $list(A)$ de listas sobre A puede definirse inductivamente de la siguiente manera:

- **Regla base:** la lista vacía nil pertenece a $list(A)$, y
- **Regla inductiva:** si $a \in A$ y $l \in list(A)$ entonces $a : l \in list(A)$, donde $a : l$ denota la lista que se obtiene agregando a a la cabeza de l.

Aplicación: Listas

El conjunto $list(A)$ de listas sobre A puede definirse inductivamente de la siguiente manera:

- **Regla base:** la lista vacía nil pertenece a $list(A)$, y
- **Regla inductiva:** si $a \in A$ y $l \in list(A)$ entonces $a : l \in list(A)$, donde $a : l$ denota la lista que se obtiene agregando a a la cabeza de l.

Equivalentemente, en formato de reglas de inferencia:

$$\frac{}{\text{nil} \in list(A)} \qquad\qquad \frac{a \in A \qquad l \in list(A)}{a : l \in list(A)}$$

Aplicación: Listas

$$\frac{}{\text{nil} \in \mathit{list}(A)} \qquad \frac{a \in A \qquad l \in \mathit{list}(A)}{a : l \in \mathit{list}(A)}$$

Aplicación: Listas

$$\frac{}{\text{nil} \in \mathit{list}(A)} \qquad \frac{a \in A \qquad l \in \mathit{list}(A)}{a : l \in \mathit{list}(A)}$$

Pregunta: ¿Qué dice el esquema de recursión sobre $\mathit{list}(A)$?

Aplicación: Listas

$$\frac{}{\text{nil} \in \mathit{list}(A)} \qquad \frac{a \in A \qquad l \in \mathit{list}(A)}{a : l \in \mathit{list}(A)}$$

Pregunta: ¿Qué dice el esquema de recursión sobre *list*(A)?

Sea $f : A \to B$. La función

$$\mathit{map}_f \;:\; \mathit{list}(A) \to \mathit{list}(B)$$

que transforma una listra sobre A en una lista sobre B aplicando f elemento a elemento puede definirse recursivamente de la siguiente manera:

Aplicación: Listas

$$\frac{}{\text{nil} \in \mathit{list}(A)} \qquad\qquad \frac{a \in A \qquad l \in \mathit{list}(A)}{a : l \in \mathit{list}(A)}$$

Pregunta: ¿Qué dice el esquema de recursión sobre $\mathit{list}(A)$?

Sea $f : A \to B$. La función

$$\mathit{map}_f \ : \ \mathit{list}(A) \to \mathit{list}(B)$$

que transforma una listra sobre A en una lista sobre B aplicando f elemento a elemento puede definirse recursivamente de la siguiente manera:

$$\mathit{map}_f(\text{nil}) \ = \ \text{nil} \qquad\qquad \mathit{map}_f(a : l) \ = \ f(a) \ : \ \mathit{map}_f(l)$$

Aplicación: Listas

$$\frac{}{\text{nil} \in \mathit{list}(A)} \qquad \frac{a \in A \qquad l \in \mathit{list}(A)}{a : l \in \mathit{list}(A)}$$

Aplicación: Listas

$$\frac{}{\text{nil} \in \mathit{list}(A)} \qquad \frac{a \in A \qquad l \in \mathit{list}(A)}{a : l \in \mathit{list}(A)}$$

Pregunta: ¿Qué dice el principio de inducción estructural sobre $\mathit{list}(A)$?

Aplicación: Listas

$$\frac{}{\text{nil} \in \mathit{list}(A)} \qquad\qquad \frac{a \in A \qquad l \in \mathit{list}(A)}{a : l \in \mathit{list}(A)}$$

Pregunta: ¿Qué dice el principio de inducción estructural sobre $\mathit{list}(A)$?

Ejercicio: Dadas $f : A \to B$ y $g : B \to C$ probar que para toda $l \in \mathit{list}(A)$,

$$\big(map_g \circ map_f\big)(l) \;=\; map_{g \circ f}(l)$$

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Caso base:

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Caso base: Tengo que probar que la propiedad vale para nil.

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Caso base: Tengo que probar que la propiedad vale para nil.

$$\left(map_g \circ map_f\right)(\text{nil})$$

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Caso base: Tengo que probar que la propiedad vale para nil.

$$\big(map_g \circ map_f\big)(\text{nil}) \;=\; map_g\big(map_f(\text{nil})\big) \qquad\qquad \text{(Def. } \circ\text{)}$$

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Caso base: Tengo que probar que la propiedad vale para nil.

$$
\begin{aligned}
\big(map_g \circ map_f\big)(\text{nil}) &= map_g\big(map_f(\text{nil})\big) && (\text{Def. } \circ) \\
&= map_g(\text{nil}) && (\text{Def. } map_f)
\end{aligned}
$$

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Caso base: Tengo que probar que la propiedad vale para nil.

$$\begin{aligned}
\left(map_g \circ map_f\right)(\text{nil}) &= map_g\left(map_f(\text{nil})\right) && (\text{Def. } \circ) \\
&= map_g(\text{nil}) && (\text{Def. } map_f) \\
&= \text{nil} && (\text{Def. } map_g)
\end{aligned}$$

Aplicación: Listas

La prueba procede por inducción estructural sobre l.

Caso base: Tengo que probar que la propiedad vale para nil.

$$
\begin{aligned}
\big(map_g \circ map_f\big)(\text{nil}) &= map_g\big(map_f(\text{nil})\big) && (\text{Def. } \circ) \\
&= map_g(\text{nil}) && (\text{Def. } map_f) \\
&= \text{nil} && (\text{Def. } map_g) \\
&= map_{g\circ f}(\text{nil}) && (\text{Def. } map_{g\circ f})
\end{aligned}
$$

Aplicación: Listas

Caso inductivo:

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

$$\left(map_g \circ map_f\right)(a : l)$$

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

$$\big(map_g \circ map_f\big)(a : l) \;=\; map_g\big(map_f(a : l)\big) \qquad \text{(Def. } \circ\text{)}$$

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

$$\begin{aligned} (map_g \circ map_f)(a : l) &= map_g\big(map_f(a : l)\big) && \text{(Def. } \circ\text{)} \\ &= map_g\big(f(a) \;:\; map_f(l)\big) && \text{(Def. } map_f\text{)} \end{aligned}$$

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

$$\begin{aligned}
\big(map_g \circ map_f\big)(a : l) &= map_g\big(map_f(a : l)\big) && \text{(Def. } \circ\text{)} \\
&= map_g\big(f(a)\ :\ map_f(l)\big) && \text{(Def. } map_f\text{)} \\
&= g\big(f(a)\big)\ :\ map_g\big(map_f(l)\big) && \text{(Def. } map_g\text{)}
\end{aligned}$$

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

$$\begin{aligned}
\big(map_g \circ map_f\big)(a : l) &= map_g\big(map_f(a : l)\big) && \text{(Def. } \circ\text{)} \\
&= map_g\big(f(a) \ : \ map_f(l)\big) && \text{(Def. } map_f\text{)} \\
&= g\big(f(a)\big) \ : \ map_g\big(map_f(l)\big) && \text{(Def. } map_g\text{)} \\
&= (g \circ f)(a) \ : \ \big(map_g \circ map_f\big)(l) && \text{(Def. } \circ\text{)}
\end{aligned}$$

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

$$
\begin{aligned}
\big(map_g \circ map_f\big)(a : l) &= map_g\big(map_f(a : l)\big) && \text{(Def. } \circ\text{)} \\
&= map_g\big(f(a) \,:\, map_f(l)\big) && \text{(Def. } map_f\text{)} \\
&= g\big(f(a)\big) \,:\, map_g\big(map_f(l)\big) && \text{(Def. } map_g\text{)} \\
&= (g \circ f)(a) \,:\, \big(map_g \circ map_f\big)(l) && \text{(Def. } \circ\text{)} \\
&= (g \circ f)(a) \,:\, map_{g \circ f}(l) && \text{(H.I)}
\end{aligned}
$$

Aplicación: Listas

Caso inductivo: Tengo que probar que la propiedad vale para $a : l$ asumiendo que vale para l.

$$
\begin{aligned}
\big(map_g \circ map_f\big)(a : l) &= map_g\big(map_f(a : l)\big) && \text{(Def. } \circ\text{)} \\
&= map_g\big(f(a) \;:\; map_f(l)\big) && \text{(Def. } map_f\text{)} \\
&= g\big(f(a)\big) \;:\; map_g\big(map_f(l)\big) && \text{(Def. } map_g\text{)} \\
&= (g \circ f)(a) \;:\; \big(map_g \circ map_f\big)(l) && \text{(Def. } \circ\text{)} \\
&= (g \circ f)(a) \;:\; map_{g \circ f}(l) && \text{(H.I)} \\
&= map_{g \circ f}(a : l) && \text{(Def. } map_{g\circ f}\text{)}
\end{aligned}
$$

Ejercicios

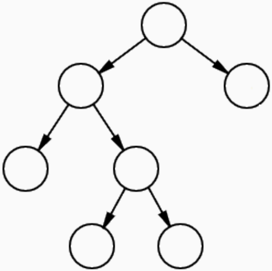
Ejercicio*: Definir recursivamente la función *length* que devuelve el largo de una lista.

Ejercicio*: Probar que para toda $f : A \to B$ y $l \in list(A)$,

$$length\big(map_f(l)\big) \ = \ length(l)$$

Aplicación: Árboles binarios

Ejemplo:



Aplicación: Árboles binarios

El conjunto $\mathcal{AB}$ de árboles binarios puede ser definido inductivamente como sigue:

Aplicación: Árboles binarios

El conjunto $\mathcal{AB}$ de árboles binarios puede ser definido inductivamente como sigue:

- **Regla base:** El árbol binario  (consistente en un solo nodo—una hoja) está en $\mathcal{AB}$.

Aplicación: Árboles binarios

El conjunto $\mathcal{AB}$ de árboles binarios puede ser definido inductivamente como sigue:

- **Regla base:** El árbol binario (consistente en un solo nodo—una hoja) está en $\mathcal{AB}$.
- **Regla inductiva:** Si T_1 y T_2 son árboles binarios en $\mathcal{AB}$, entonces el árbol binario $T_1 \blacktriangle T_2$ (que consiste en un nodo raíz del que cuelgan T_1 y T_2) también está en $\mathcal{AB}$.

Aplicación: Árboles binarios

El conjunto $\mathcal{AB}$ de árboles binarios puede ser definido inductivamente como sigue:

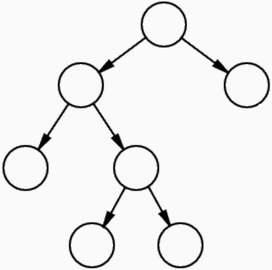
- **Regla base:** El árbol binario (consistente en un solo nodo—una hoja) está en $\mathcal{AB}$.
- **Regla inductiva:** Si T_1 y T_2 son árboles binarios en $\mathcal{AB}$, entonces el árbol binario $T_1 \blacktriangle T_2$ (que consiste en un nodo raíz del que cuelgan T_1 y T_2) también está en $\mathcal{AB}$.

En forma de reglas de inferencia

$$\frac{}{\;\in \mathcal{AB}} \qquad\qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Aplicación: Árboles binarios

Por ejemplo, el árbol binario anterior



se representa en $\mathcal{AB}$ como

$$((\faLeaf \blacktriangle (\faLeaf \blacktriangle \faLeaf)) \blacktriangle \faLeaf)$$

Aplicación: Árboles binarios

$$\frac{}{\text{🍃} \in \mathcal{AB}} \qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Aplicación: Árboles binarios

$$\frac{}{\text{🍃} \in \mathcal{AB}} \qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Pregunta: ¿Qué dice el esquema de recursión sobre $\mathcal{AB}$?

Aplicación: Árboles binarios

$$\frac{}{\text{🍃} \in \mathcal{AB}} \qquad\qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Pregunta: ¿Qué dice el esquema de recursión sobre $\mathcal{AB}$?

La función $n\colon \mathcal{AB} \to \mathbb{N}$ que devuelve el número de nodos de un árbol binario puede definirse recursivamente como sigue:

Aplicación: Árboles binarios

$$\dfrac{}{\text{🍃} \in \mathcal{AB}} \qquad\qquad \dfrac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Pregunta: ¿Qué dice el esquema de recursión sobre $\mathcal{AB}$?

La función $n \colon \mathcal{AB} \to \mathbb{N}$ que devuelve el número de nodos de un árbol binario puede definirse recursivamente como sigue:

$$n(\text{🍃}) \;=\; 1 \qquad\qquad n(T_1 \blacktriangle T_2) \;=\; 1 + n(T_1) + n(T_2)$$

Aplicación: Árboles binarios

$$\frac{}{\faLeaf \in \mathcal{AB}} \qquad\qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Pregunta: ¿Qué dice el esquema de recursión sobre $\mathcal{AB}$?

La función $n \colon \mathcal{AB} \to \mathbb{N}$ que devuelve el número de nodos de un árbol binario puede definirse recursivamente como sigue:

$$n(\faLeaf) \;=\; 1 \qquad\qquad n(T_1 \blacktriangle T_2) \;=\; 1 + n(T_1) + n(T_2)$$

La función $h \colon \mathcal{AB} \to \mathbb{N}$ que devuelve la altura de un árbol binario puede definirse recursivamente como sigue:

Aplicación: Árboles binarios

$$\frac{}{\faLeaf \in \mathcal{AB}} \qquad\qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Pregunta: ¿Qué dice el esquema de recursión sobre $\mathcal{AB}$?

La función $n\colon \mathcal{AB} \to \mathbb{N}$ que devuelve el número de nodos de un árbol binario puede definirse recursivamente como sigue:

$$n(\faLeaf) \;=\; 1 \qquad\qquad n(T_1 \blacktriangle T_2) \;=\; 1 + n(T_1) + n(T_2)$$

La función $h\colon \mathcal{AB} \to \mathbb{N}$ que devuelve la altura de un árbol binario puede definirse recursivamente como sigue:

$$h(\faLeaf) \;=\; 0 \qquad\qquad h(T_1 \blacktriangle T_2) \;=\; 1 + \text{máx}\{h(T_1), h(T_2)\}$$

Aplicación: Árboles binarios

$$\frac{}{\text{🍃} \in \mathcal{AB}} \qquad\qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Aplicación: Árboles binarios

$$\frac{}{\text{🍃} \in \mathcal{AB}} \qquad\qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Pregunta: ¿Qué dice el principio de inducción estructural sobre $\mathcal{AB}$?

Aplicación: Árboles binarios

$$\frac{}{\text{🍃} \in \mathcal{AB}} \qquad\qquad \frac{T_1 \in \mathcal{AB} \quad T_2 \in \mathcal{AB}}{T_1 \blacktriangle T_2 \in \mathcal{AB}}$$

Pregunta: ¿Qué dice el principio de inducción estructural sobre $\mathcal{AB}$?

Ejercicio*: Demuestre que para todo $T \in \mathcal{AB}$,

$$n(T) \leq 2^{h(T)+1} - 1$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base:

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para .

$$n(\;)$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para .

$n($$) = 1$ (Def n)

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para 🍃.

$$\begin{aligned} n(\text{🍃}) &= 1 && \text{(Def } n\text{)} \\ &= 2^{0+1} - 1 \end{aligned}$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para $\faLeaf$.

$$
\begin{aligned}
n(\faLeaf) &= 1 && \text{(Def } n\text{)} \\
&= 2^{0+1} - 1 \\
&= 2^{h(\faLeaf)+1} - 1 && \text{(Def } h\text{)}
\end{aligned}
$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para 🍃.

$$
\begin{aligned}
n(\text{🍃}) &= 1 && \text{(Def } n\text{)} \\
&= 2^{0+1} - 1 \\
&= 2^{h(\text{🍃})+1} - 1 && \text{(Def } h\text{)}
\end{aligned}
$$

Caso inductivo: Tengo que probar que la propiedad vale para $T_1 \blacktriangle T_2$ asumiendo que vale para T_1 y para T_2.

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para 🍃.

$$
\begin{aligned}
n(\text{🍃}) &= 1 && \text{(Def } n\text{)} \\
&= 2^{0+1} - 1 \\
&= 2^{h(\text{🍃})+1} - 1 && \text{(Def } h\text{)}
\end{aligned}
$$

Caso inductivo: Tengo que probar que la propiedad vale para $T_1 \blacktriangle T_2$ asumiendo que vale para T_1 y para T_2.

$$n(T_1 \blacktriangle T_2)$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para 🍃.

$$
\begin{aligned}
n(\text{🍃}) &= 1 && \text{(Def } n\text{)} \\
&= 2^{0+1} - 1 \\
&= 2^{h(\text{🍃})+1} - 1 && \text{(Def } h\text{)}
\end{aligned}
$$

Caso inductivo: Tengo que probar que la propiedad vale para $T_1 \blacktriangle T_2$ asumiendo que vale para T_1 y para T_2.

$$
n(T_1 \blacktriangle T_2) = 1 + n(T_1) + n(T_2) \qquad \text{(Def } n\text{)}
$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para $\faLeaf$.

$$\begin{aligned} n(\faLeaf) &= 1 && \text{(Def } n\text{)} \\ &= 2^{0+1} - 1 && \\ &= 2^{h(\faLeaf)+1} - 1 && \text{(Def } h\text{)} \end{aligned}$$

Caso inductivo: Tengo que probar que la propiedad vale para $T_1 \blacktriangle T_2$ asumiendo que vale para T_1 y para T_2.

$$\begin{aligned} n(T_1 \blacktriangle T_2) &= 1 + n(T_1) + n(T_2) && \text{(Def } n\text{)} \\ &\leq 1 + 2^{h(T_1)+1} - 1 + 2^{h(T_2)+1} - 1 && \text{(H.I)} \end{aligned}$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para $\faLeaf$.

$$\begin{aligned} n(\faLeaf) &= 1 && \text{(Def } n\text{)} \\ &= 2^{0+1} - 1 \\ &= 2^{h(\faLeaf)+1} - 1 && \text{(Def } h\text{)} \end{aligned}$$

Caso inductivo: Tengo que probar que la propiedad vale para $T_1 \blacktriangle T_2$ asumiendo que vale para T_1 y para T_2.

$$\begin{aligned} n(T_1 \blacktriangle T_2) &= 1 + n(T_1) + n(T_2) && \text{(Def } n\text{)} \\ &\leq 1 + 2^{h(T_1)+1} - 1 + 2^{h(T_2)+1} - 1 && \text{(H.I)} \\ &= 2^{h(T_1)+1} + 2^{h(T_2)+1} - 1 \end{aligned}$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para 🍃.

$$
\begin{aligned}
n(🍃) &= 1 && \text{(Def } n\text{)}\\
&= 2^{0+1} - 1 \\
&= 2^{h(🍃)+1} - 1 && \text{(Def } h\text{)}
\end{aligned}
$$

Caso inductivo: Tengo que probar que la propiedad vale para $T_1 \blacktriangle T_2$ asumiendo que vale para T_1 y para T_2.

$$
\begin{aligned}
n(T_1 \blacktriangle T_2) &= 1 + n(T_1) + n(T_2) && \text{(Def } n\text{)}\\
&\leq 1 + 2^{h(T_1)+1} - 1 + 2^{h(T_2)+1} - 1 && \text{(H.I)}\\
&= 2^{h(T_1)+1} + 2^{h(T_2)+1} - 1 \\
&\leq 2 \cdot 2^{\text{máx}\,\{h(T_1),h(T_2)\}+1} - 1 && (a, b \leq \text{máx}\,\{a, b\})
\end{aligned}
$$

Aplicación: Árboles binarios

La prueba procede por inducción estructural sobre T.

Caso base: Tengo que probar que la propiedad vale para 🍃.

$$
\begin{aligned}
n(\text{🍃}) &= 1 && \text{(Def } n\text{)} \\
&= 2^{0+1} - 1 \\
&= 2^{h(\text{🍃})+1} - 1 && \text{(Def } h\text{)}
\end{aligned}
$$

Caso inductivo: Tengo que probar que la propiedad vale para $T_1 \blacktriangle T_2$ asumiendo que vale para T_1 y para T_2.

$$
\begin{aligned}
n(T_1 \blacktriangle T_2) &= 1 + n(T_1) + n(T_2) && \text{(Def } n\text{)} \\
&\leq 1 + 2^{h(T_1)+1} - 1 + 2^{h(T_2)+1} - 1 && \text{(H.I)} \\
&= 2^{h(T_1)+1} + 2^{h(T_2)+1} - 1 \\
&\leq 2 \cdot 2^{\text{máx}\,\{h(T_1), h(T_2)\}+1} - 1 && (a, b \leq \text{máx}\,\{a, b\}) \\
&= 2^{h(T_1 \blacktriangle T_2)+1} - 1 && \text{(Def } h\text{)}
\end{aligned}
$$

Ejercicios finales

- El inverso w^{-1} de un string w es el string que consiste de los símbolos de w en orden inverso.
- Un string w es un palíndrome si $w = w^{-1}$.

Ejercicio: Defina inductivamente la clase de los strings palíndromes sobre un alfabeto Σ.

Ejercicio: Defina recursivamente la función $P(n)$ que define el número de palíndromes de largo n sobre un alfabeto Σ.