Capítulo 2: Inducción y recursión
Clase 3: Definiciones recursivas e Inducción estructural

Matemática Discreta - CC3101
Profesor: Pablo Barceló

Motivación

A veces es difícil definir un objeto explícitamente, pero no lo es tanto si lo definimos en términos de objetos de su mismo tipo. A este proceso lo llamamos recursión.

Podemos definir recursivamente secuencias, funciones, conjuntos, etc.

Ejemplo: La secuencia de las potencias de 2 está dada por $a_n = 2^n$. Definida recursivamente se ve como:

$$a_0 = 1 \qquad a_{n+1} = 2a_n$$

Es decir, damos el primer término y una regla de cómo construir un término de la serie desde los términos anteriores.

Definición recursiva de funciones

Ejemplo: La función $f(n) = n!$ puede ser definida recursivamente como:

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

Definición recursiva de funciones

Ejemplo: La función $f(n) = n!$ puede ser definida recursivamente como:

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

Ejercicio: Defina recursivamente la función $f(n) = \sum_{k=0}^{n} a_k$.

Números de Fibonacci

Quizás una de las más famosas definiciones recursivas es la siguiente:

Definición

Los números de Fibonacci $f_0, f_1, f_2, \ldots$ *se definen como* $f_0 = 0$, $f_1 = 1$, *y para todo* $n \geq 2$*:*

$$f_n = f_{n-1} + f_{n-2}$$

Ejercicio: Encuentre los primeros 6 números de Fibonacci.

Un ejercicio sobre Fibonacci

Las definiciones recursivas son terreno fértil para las demostraciones inductivas.

Un ejercicio sobre Fibonacci

Las definiciones recursivas son terreno fértil para las demostraciones inductivas.

Ejercicio: Demuestre que para todo $n \geq 3$, si $\alpha = (1 + \sqrt{5})/2$ entonces $f_n > \alpha^{n-2}$.

¿Cuál es el caso base en este caso?

Un ejercicio sobre Fibonacci

Caso base: Debemos demostrar el caso base tanto para 3 como para 4 (porque no podemos demostrar que $f_4 > \alpha^2$ desde la hipótesis inductiva).

Esto no es difícil puesto que:

- $f_3 = 2 > \alpha = (1 + \sqrt{5})/2$; y
- $f_4 = 3 > \alpha^2 = (3 + \sqrt{5})/2$.

Un ejercicio sobre Fibonacci

Caso inductivo: Considere entero $n+1$ tal que $n \geq 4$. Por hipótesis inductiva, $f_j > a^{j-2}$ para todo $j \in [3, n]$.

Pero, $f_{n+1} = f_n + f_{n-1}$. Luego, $f_{n+1} > \alpha^{n-2} + \alpha^{n-3} = \alpha^{n-3}(1+\alpha)$.

Por tanto, para demostrar que $f_{n+1} > \alpha^{n-1}$ basta demostrar que $(1+\alpha) \geq \alpha^2$.

Esto es fácil pues $(1+\alpha) = (3+\sqrt{5})/2 = \alpha^2$.

Otro ejercicio sobre Fibonacci

Ejercicio: Demuestre que para todo entero positivo n,

$$f_0 f_1 + f_1 f_2 + \cdots + f_{2n-1} f_{2n} = f_{2n}^2.$$

Conjuntos definidos recursivamente

Ya hemos visto cómo definir funciones recursivamente. Ahora definiremos conjuntos de esa forma.

La definición recursiva de conjuntos se basa en lo siguiente:

- Regla base: Un conjunto inicial de elementos que pertenecen al conjunto es especificado.
- Regla inductiva: Regla que define cómo agregar nuevos elementos al conjunto desde aquellos que ya están en el conjunto.
- Regla de exclusión: El conjunto no contiene nada más que aquello especificado por la regla base o que se obtiene recursivamente por aplicación de la regla recursiva.

(Comúnmente la regla de exclusión estará implícita).

Definición recursiva del conjunto de strings

Sea Σ un alfabeto. El conjunto Σ^* de los strings sobre alfabeto Σ se define recursivamente como:

Regla base: El string vacío ε pertenece a Σ^*.

Regla inductiva: Si el string w está en Σ^* y $x \in \Sigma$, entonces wx pertenece también a Σ^*.

Definiciones recursivas de operaciones sobre strings

Podemos además definir operaciones recursivamente sobre conjuntos definidos recursivamente.

Por ejemplo, la concatenación $w_1 \cdot w_2$ de strings puede ser definida recursivamente sobre Σ^* de la siguiente forma:

Regla base: Si $w \in \Sigma^*$ entonces $w \cdot \varepsilon = w$.

Regla inductiva: Si $w_1, w_2 \in \Sigma^*$ y $x \in \Sigma$, entonces $w_1 \cdot (w_2 x) = (w_1 \cdot w_2)x$.

El largo $\ell(w)$ de un string puede ser definido como:

Regla base: $\ell(\varepsilon) = 0$.

Regla inductiva: $\ell(wx) = \ell(w) + 1$.

Un ejercicio sobre strings

Para demostrar cosas sobre conjuntos definidos recursivamente podemos utilizar inducción estructural.

Ejercicio: Demuestre que para todo par de strings $w_1 w_2 \in \Sigma^*$, $\ell(w_1 w_2) = \ell(w_1) + \ell(w_2)$.

Sea $P(y)$ la proposición que expresa que $\ell(xy) = \ell(x) + \ell(y)$ para cualquier $x \in \Sigma^*$.

Demostraremos que $P(y)$ es cierto para todo $y \in \Sigma^*$.

Un ejercicio sobre strings

Basta demostrar dos cosas:

- $P(\varepsilon)$ es cierto: Esto es fácil pues

$$\ell(x\varepsilon) = \ell(x) = \ell(x) + 0 = \ell(x) + \ell(\varepsilon).$$

- Si $P(w)$ es cierto entonces $P(wa)$ es cierto, para $a \in \Sigma$:
  Considere el string w_1wa. Entonces $\ell(w_1wa) = \ell(w_1w) + 1$.
  Por hipótesis inductiva y definición de función ℓ,

$$\ell(w_1wa) = \ell(w_1w) + 1 = \ell(w_1) + \ell(w) + 1 = \ell(w_1) + \ell(wa).$$

Definición recursiva de árboles binarios

El conjunto de los árboles binarios completos puede ser definido recursivamente como sigue:

Regla base: Existe un árbol binario completo que consiste de un solo vértice r.

Regla inductiva: Si T_1 y T_2 son árboles binarios completos disjuntos, entonces existe un árbol binario completo $T_1 \cdot T_2$ que consiste de una raíz r junto con arcos que unen a r con la raíz del subárbol izquierdo T_1 y con la raíz del subárbol derecho T_2.

Definiciones recursivas sobre árboles binarios

Ejercicio: Defina recursivamente la función $n(T)$ que entrega el número de vértices de un árbol binario T.

Ejercicio: Defina la función $h(T)$ que entrega la distancia máxima entre la raíz de un árbol binario T y una de sus hojas.

Ejercicio: Demuestre que para todo árbol binario completo T, $n(T) \leq 2^{h(T)+1} - 1$.

Ejercicios finales

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

Ejercicio: De una definición recursiva del inverso de un string.

Ejercicio: Demuestre usando inducción estructural que $(w_1 w_2)^{-1} = w_1^{-1} w_2^{-1}$.

Un string w es un palíndrome si $w = w^{-1}$.

Ejercicio: Defina recursivamente la clase de los strings que son palíndromes.

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