

(정의) 아래의 세 가지 initial function과 세 가지 함수의 결합으로만 이루어진 함수를 **primitive recursive function**이라고 정의한다.

단 k-tuple $\vec{n} = (n_1, \dots, n_k) \in N^k$ 이라하자.

1. Initial functions

(1) **zero**: $\forall k \geq 0: \zeta: N^k \rightarrow \{0\}$

$$\forall \vec{n} \in N^k: \zeta(\vec{n}) \stackrel{\text{def}}{=} 0.$$

(2) **successor**: $\sigma: N \rightarrow N$

$$\forall n \in N: \sigma(n) \stackrel{\text{def}}{=} n+1.$$

(3) **projection**: $\forall k \geq 1: 1 \leq i \leq k: \pi_i^k: N^k \rightarrow N,$

$$\forall \vec{n} \in N^k: \pi_i^k(\vec{n}) \stackrel{\text{def}}{=} n_i \quad \text{단 } \vec{n} = (n_1, \dots, n_i, \dots, n_k) \in N^k.$$

2. 함수의 결합

(1) **combination** of functions: $f: N^k \rightarrow N^n, g: N^k \rightarrow N^m$ 일 때,

$$f \times g: N^k \rightarrow N^{n+m}: \forall \vec{n} \in N^k: f \times g(\vec{n}) \stackrel{\text{def}}{=} (f(\vec{n}), g(\vec{n})).$$

(2) **composition** of functions: $f: N^k \rightarrow N^n, g: N^n \rightarrow N^m$ 일 때,

$$f \circ g: N^k \rightarrow N^m: \forall \vec{n} \in N^k: f \circ g(\vec{n}) \stackrel{\text{def}}{=} g(f(\vec{n})).$$

(3) **primitive recursion** of functions: $g: N^k \rightarrow N^m, h: N^{k+m+1} \rightarrow N^m$ 일 때,

$$f: N^{k+1} \rightarrow N^m: \forall \vec{n} \in N^k: \forall r \in N:$$

$$f(\vec{n}, \zeta(r)) \stackrel{\text{def}}{=}_{\mathbf{B}} g(\vec{n}),$$

$$f(\vec{n}, \sigma(r)) \stackrel{\text{def}}{=}_{\mathbf{R}} h(\vec{n}, r, f(\vec{n}, r)).$$

또는

$$f(\vec{n}, 0) \stackrel{\text{def}}{=}_{\mathbf{B}} g(\vec{n}),$$

$$f(\vec{n}, r+1) \stackrel{\text{def}}{=}_{\mathbf{R}} h(\vec{n}, r, f(\vec{n}, r)).$$

함수 zero, successor, projection, combination, composition, primitive recursion은 모두 TM **computable**이므로 primitive recursive function은 TM **computable**다.

임의의 k-tuple은 자연수와 1:1 대응하므로, $N^k \leftrightarrow^{1:1} N$, 위 정의의 k-tuple, m-tuple, n-tuple 또는 심지어 (k+1)-tuple 까지도 자연수로 바꾸어도 되고 역으로 자연수도 k-tuple로 변환하면 되므로, combination이 필요 없고, **zero**, **successor**, composition과 primitive recursion만 정의하면 된다. 심지어는 zero와 successor를 없애고 **constant**를 정의하여도 되나, zero와 successor는 자연수를 정리한 Peano의 Lemma의 기본이므로 남겨두고 정의하면 아래와 같다.

(정의) k-tuple과 자연수간의 변환 $N^k \leftrightarrow^{1:1} N$ 을 아래와 같이 정의한다.

$$(n_1, n_2, \dots, n_k) \leftrightarrow (n_1 \# n_2 \# \dots \# n_k).$$

임의의 함수 $f: N^k \rightarrow N^m$ 를 위의 변환을 이용하여 $f': N \rightarrow N$ 으로 바꿀 수 있다. 따라서 아

1) 고등학교에서는 $f(g(\vec{n}))$ 으로 정의하였음에 주의하라.

래의 정의와 같이 변환을 이용하여 자연수로 정의역을 표시할 수 있다. 하지만 함수에 정의역에 여러 개의 자연수를 가지는 것에 익숙하므로 그대로 정의역을 N^k 라고 보기도 한다.

(정의) 두 가지 initial function **zero**와 **successor**, **primitive recursion**으로만 이루어진 함수를 **primitive recursive function**이라고 정의한다.

1. **zero**: $\forall k \geq 0: \zeta: N \rightarrow N, \forall n \in N: \zeta(\vec{n}) \stackrel{\text{def}}{=} 0.$

2. **successor**: $\sigma: N \rightarrow N, \forall n \in N: \sigma(n) \stackrel{\text{def}}{=} n + 1.$

3. **primitive recursion** $g: N \rightarrow N, h: N \rightarrow N$ 일 때,

$$f: N \rightarrow N: \quad \forall n \in N: \forall r \in N: \\ f(n\#\zeta(r)) \stackrel{\text{def}}{=}_{\mathbf{B}} g(n), \\ f(n\#\sigma(r)) \stackrel{\text{def}}{=}_{\mathbf{R}} h(n\#r\#f(n\#r)).$$

또는

$$f(n\#0) \stackrel{\text{def}}{=}_{\mathbf{B}} g(n), \\ f(n\#(r+1)) \stackrel{\text{def}}{=}_{\mathbf{R}} h(n\#r\#f(n\#r)).$$

primitive recursive 함수의 예

1. **Constant**: $\forall a \in N: K_a: N \rightarrow N, \forall n \in N: K_a(n) \stackrel{\text{def}}{=} \zeta \circ \sigma^a(n) = a.$

Primitive recursive function $K_a(n)$ 나 $\zeta \circ \sigma^a(n)$ 대신에 상수 a 를 그대로 쓴다.

2. Arithmetic function

$plus: N \times N \rightarrow N$ $plus(n, 0) \stackrel{\text{def}}{=}_{\mathbf{B}} n,$ $plus(n, \sigma(m)) \stackrel{\text{def}}{=}_{\mathbf{R}} \sigma(+ (n, m)).$	$+: N \times N \rightarrow N$ $n + 0 \stackrel{\text{def}}{=}_{\mathbf{B}} n,$ $n + \sigma(m) \stackrel{\text{def}}{=}_{\mathbf{R}} \sigma(n + m)$ $= (n + m) + 1.$
---	--

오른 쪽 정의와 같이 연산자 $+$ 가 infix operator임을 그대로 이용해도 된다.

$mult: N \times N \rightarrow N$ $mult(n, \zeta(m)) \stackrel{\text{def}}{=}_{\mathbf{B}} \zeta(n),$ $mult(n, \sigma(m)) \stackrel{\text{def}}{=}_{\mathbf{R}} plus(n, mult(n, m)).$	$\cdot: N \times N \rightarrow N$ $n \cdot 0 \stackrel{\text{def}}{=}_{\mathbf{B}} 0,$ $n \cdot (\sigma(m)) \stackrel{\text{def}}{=}_{\mathbf{R}} n \cdot m + n.$
--	---

$expo: N \times N \rightarrow N$ $expo(n, 0) \stackrel{\text{def}}{=}_{\mathbf{B}} 1,$ $expo(n, \sigma(m)) \stackrel{\text{def}}{=}_{\mathbf{R}} mult(n, expo(n, m)).$	$\cdot^{\cdot}: N \times N \rightarrow N$ $n^0 \stackrel{\text{def}}{=}_{\mathbf{B}} 1,$ $n^{m+1} \stackrel{\text{def}}{=}_{\mathbf{R}} n^m \cdot n.$
--	---

Successor 함수(σ)의 역을 생각해 보자.

$$\sigma^{-1}: N \rightarrow N \\ \sigma^{-1}(0) \stackrel{\text{def}}{=}_{\mathbf{B}} 0, \\ \sigma^{-1}(n+1) \stackrel{\text{def}}{=}_{\mathbf{R}} n.$$

$$\begin{aligned} \dot{\div} : N \times N &\rightarrow N & n \dot{\div} m &\stackrel{\text{def}}{=} \text{if } n \geq m \rightarrow n - m \mid n \leq m \rightarrow 0 \text{ fi} \\ n \dot{\div} 0 &\stackrel{\text{def}}{=}_{\mathbf{B}} n, \\ n \dot{\div} (m + 1) &\stackrel{\text{def}}{=}_{\mathbf{R}} \sigma^{-1}(n \dot{\div} m). \end{aligned}$$

3. Boolean functions

$$\begin{aligned} = : N \times N &\rightarrow \{0, 1\} & n = m &\stackrel{\text{def}}{=} \text{if } n = m \rightarrow 1 \mid n \neq m \rightarrow 0 \text{ fi} \\ n = m &\stackrel{\text{def}}{=} 1 \dot{\div} ((n \dot{\div} m) + (m \dot{\div} n)) \end{aligned}$$

$\neq, >, \geq, <, \leq : N \times N \rightarrow \{0, 1\}$ 도 정의하여 보아라.

4. Boolean operators

$$\vee : \{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\}$$

$$\wedge : \{0, 1\} \times \{0, 1\} \rightarrow \{0, 1\}$$

$$\neg : \{0, 1\} \rightarrow \{0, 1\}$$

도 정의하여 보아라.

Ackerman Function(1928)

$$\begin{aligned} A : N^2 &\rightarrow N \\ A(0, y) &=_0 y + 1 \\ A(\sigma(x), 0) &=_{\mathbf{1}} A(x, 1) \\ A(\sigma(x), \sigma(y)) &= A(x, A(\sigma(x), y)) \\ A(2, 1) &= A(1, A(2, 0)) \\ &= A(1, A(1, 1)) \\ &= A(1, A(0, A(0, 1))) \\ &= A(1, A(0, 2)) \\ &= A(1, 3) \\ &= A(0, A(1, 2)) \\ &= A(0, A(0, A(1, 1))) \\ &= A(0, A(0, A(0, A(1, 0)))) \\ &= A(0, A(0, A(0, A(0, 1)))) \\ &= A(0, A(0, A(0, 2))) \\ &= A(0, A(0, 3)) \\ &= A(0, 4) \\ &= 5 \end{aligned}$$

$A(3, 2)$ 를 한 번 해 보시오!

Ackerman function은 **computable**이지만, **primitive recursion**은 아니다²⁾.

2) 증명은 recursion의 큰 수(big number)로 하는데 생략한다.