

Recursive Function*

Xiaofeng Gao

Department of Computer Science and Engineering
Shanghai Jiao Tong University, P.R.China

CS363-Computability Theory

*Special thanks is given to Prof. Yuxi Fu for sharing his teaching materials.

Outline

- 1 Basic Functions
 - Three Basic Functions
- 2 Substitution
 - Definition
 - Variable Sequences
- 3 Recursion
 - Definition
 - Examples
 - Corollary
- 4 Minimalisation
 - Bounded Minimalisation
 - Unbounded Minimalisation
 - A Famous Example

The Basic Functions

Lemma. The following basic functions are computable.

- 1 The *zero function* 0 .
- 2 The *successor function* $x + 1$.
- 3 For each $n \geq 1$ and $1 \leq i \leq n$, the *projection function* U_i^n given by $U_i^n(x_1, \dots, x_n) = x_i$.

Proof

These functions correspond to the arithmetic instructions for URM.

- 1 0 : program $Z(1)$;
- 2 $x + 1$: program $S(1)$;
- 3 U_i^n : program $T(i, 1)$.

Substitution Theorem

Suppose that $f(y_1, \dots, y_k)$ and $g_1(\mathbf{x}), \dots, g_k(\mathbf{x})$ are computable functions, where $\mathbf{x} = x_1, \dots, x_n$. Then the function $h(\mathbf{x})$ given by

$$h(\mathbf{x}) \simeq f(g_1(\mathbf{x}), \dots, g_k(\mathbf{x}))$$

is a computable function.

Question: what is the domain of definition of $h(\mathbf{x})$?

Note: $h(x)$ is defined iff $g_1(\mathbf{x}), \dots, g_k(\mathbf{x})$ are all defined and $(g_1(\mathbf{x}), \dots, g_k(\mathbf{x})) \in \text{Dom}(f)$. Thus, if f and g_1, \dots, g_k are all total functions, then h is total.

Proof (Construction)

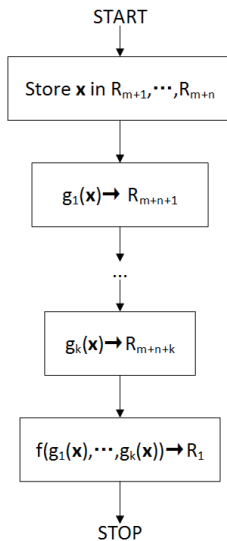
Let F, G_1, \dots, G_k be programs in standard form that compute f, g_1, \dots, g_k .

Let m be $\max\{n, k, \rho(F), \rho(G_1), \dots, \rho(G_k)\}$.

Registers:

$$[\dots]_1^m [\mathbf{x}]_{m+1}^{m+n} [g_1(\mathbf{x})]_{m+n+1}^{m+n+1} \dots [g_k(\mathbf{x})]_{m+n+k}^{m+n+k}$$

URM Program for Substitution



$$\begin{aligned}
 I_1 &: T(1, m+1) \\
 &\vdots \\
 I_n &: T(n, m+n) \\
 I_{n+1} &: G_1[m+1, \dots, m+n \rightarrow m+n+1] \\
 &\vdots \\
 I_{n+k} &: G_k[m+1, \dots, m+n \rightarrow m+n+k] \\
 I_{n+k+1} &: F[m+n+1 \dots, m+n+k \rightarrow 1]
 \end{aligned}$$

Computable Function with Variable Sequences

Theorem. Suppose that $f(y_1, \dots, y_k)$ is a computable function and that x_{i_1}, \dots, x_{i_k} is a sequence of k of the variables x_1, \dots, x_n (possibly with repetitions). Then the function h given by

$$h(x_1, \dots, x_n) \simeq f(x_{i_1}, \dots, x_{i_k})$$

is computable.

Proof. $h(\mathbf{x}) \simeq f(U_{i_1}^n(\mathbf{x}), \dots, U_{i_k}^n(\mathbf{x}))$.

Form New Functions

- **Rearrangement:** $h_1(x_1, x_2) \simeq f(x_2, x_1)$;
- **Identification:** $h_2(x) \simeq f(x, x)$;
- **Adding Dummy Variables:** $h_3(x_1, x_2, x_3) \simeq f(x_2, x_3)$.

Recursion Equations

Suppose that $f(\mathbf{x})$ and $g(\mathbf{x}, y, z)$ are functions. The function obtained from $f(\mathbf{x})$ and $g(\mathbf{x}, y, z)$ by recursion is defined as follows:

$$\begin{cases} h(\mathbf{x}, 0) \simeq f(\mathbf{x}), \\ h(\mathbf{x}, y + 1) \simeq g(\mathbf{x}, y, h(\mathbf{x}, y)). \end{cases}$$

An Example

The function $f(\mathbf{x}_1, \mathbf{x}_2, \mathbf{x}_3) = \mathbf{x}_1 + \mathbf{x}_2 + \mathbf{x}_3$ is computable.

Proof. Since $x + y$ is computable, by substituting $x_1 + x_2$ for x , and x_3 for y in $x + y$ we can claim that f is computable.

Note: When the functions g_1, \dots, g_k substituted into f , it is not necessarily involving all of the variables x_1, \dots, x_n to guarantee the computability of the new function.

Domain of h

h may not be total unless both f and g are total.

The domain of h satisfies:

$$\begin{aligned} (\mathbf{x}, 0) \in \text{Dom}(h) & \quad \text{iff} \quad \mathbf{x} \in \text{Dom}(f); \\ (\mathbf{x}, y + 1) \in \text{Dom}(h) & \quad \text{iff} \quad (\mathbf{x}, y) \in \text{Dom}(h) \\ & \quad \text{and } (\mathbf{x}, y, h(\mathbf{x}, y)) \in \text{Dom}(g). \end{aligned}$$

Uniqueness

Theorem. Let $\mathbf{x} = \{x_1, \dots, x_n\}$, and suppose that $f(\mathbf{x})$ and $g(\mathbf{x}, y, z)$ are functions; then there is a unique function $h(\mathbf{x}, y)$ satisfying the recursion equations

$$\begin{cases} h(\mathbf{x}, 0) \simeq f(\mathbf{x}), \\ h(\mathbf{x}, y + 1) \simeq g(\mathbf{x}, y, h(\mathbf{x}, y)). \end{cases}$$

Note: When $n = 0$ (\mathbf{x} do not appear), the recursion equations take the form

$$\begin{cases} h(0) = a, \\ h(y + 1) \simeq g(y, h(y)). \end{cases}$$

Proof

Registers:

$$[\dots]_1^m [\mathbf{x}]_{m+1}^{m+n} [y]_{m+n+1}^{m+n+1} [k]_{m+n+2}^{m+n+2} [h(\mathbf{x}, k)]_{m+n+3}^{m+n+3}.$$

Program:

$$T(1, m + 1)$$

\vdots

$$T(n + 1, m + n + 1)$$

$$F[1, 2, \dots, n \rightarrow m + n + 3]$$

$$I_q : J(n + m + 2, n + m + 1, p)$$

$$G[m + 1, \dots, m + n, m + n + 2, m + n + 3 \rightarrow m + n + 3]$$

$$S(n + m + 2)$$

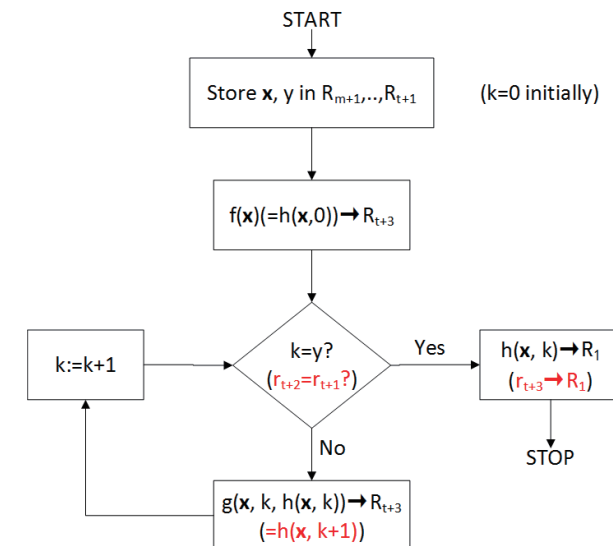
$$J(1, 1, q)$$

$$I_p : T(n + m + 3, 1)$$

Computability Theorem

Theorem. $h(\mathbf{x}, y)$ is computable if $f(\mathbf{x})$ and $g(\mathbf{x}, y, z)$ are computable.

Flow Diagram



Addition

Let $\text{add}: \mathbb{N}^2 \rightarrow \mathbb{N}$, $\text{add}(x, y) := x + y$.

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

Therefore,

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

where

$$\begin{aligned}f: \mathbb{N} &\rightarrow \mathbb{N}, & f(x) &:= x, \\ g: \mathbb{N}^3 &\rightarrow \mathbb{N}, & g(x, y, z) &:= z + 1.\end{aligned}$$

Multiplication

Let $\text{mult}: \mathbb{N}^2 \rightarrow \mathbb{N}$, $\text{mult}(x, y) := x \cdot y$.

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

Therefore,

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

where

$$\begin{aligned}f: \mathbb{N} &\rightarrow \mathbb{N}, & f(x) &:= 0, \\ g: \mathbb{N}^3 &\rightarrow \mathbb{N}, & g(x, y, z) &:= z + x.\end{aligned}$$

Power Function

Let $\text{power}: \mathbb{N}^2 \rightarrow \mathbb{N}$, $\text{power}(x, y) := x^y$

$$\begin{aligned}\text{power}(x, 0) &= x^0 \simeq 1 \\ \text{power}(x, y + 1) &= x^{(y+1)} \simeq x^y \cdot x\end{aligned}$$

Therefore,

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

where

$$\begin{aligned}f: \mathbb{N} &\rightarrow \mathbb{N}, & f(x) &:= 1, \\ g: \mathbb{N}^3 &\rightarrow \mathbb{N}, & g(x, y, z) &:= z \cdot x.\end{aligned}$$

Predecessor

Let $\text{pred}: \mathbb{N} \rightarrow \mathbb{N}$, $\text{pred}(x) := x - 1 = \begin{cases} x - 1 & \text{if } x > 0, \\ 0 & \text{otherwise.} \end{cases}$

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

Therefore,

$$\begin{aligned}\text{pred}(0) &= f(x) = 0 \\ \text{pred}(x + 1) &= g(x, \text{pred}(x))\end{aligned}$$

where

$$\begin{aligned}f: \mathbb{N} &\rightarrow \mathbb{N}, & f(x) &:= 0, \\ g: \mathbb{N}^2 &\rightarrow \mathbb{N}, & g(x, y) &:= x.\end{aligned}$$

Conditional Subtraction

Let **sub**: $\mathbb{N}^2 \rightarrow \mathbb{N}$, $sub(x, y) := x \dot{-} y \stackrel{\text{def}}{=} \begin{cases} x - y, & \text{if } x \geq y, \\ 0, & \text{otherwise.} \end{cases}$

$$\begin{aligned} sub(x, 0) &= x \dot{-} 0 \simeq x \\ sub(x, y + 1) &= x \dot{-} (y + 1) \simeq (x \dot{-} y) \dot{-} 1. \end{aligned}$$

Therefore,

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

where

$$\begin{aligned} f: \mathbb{N} &\rightarrow \mathbb{N}, & f(x) &:= x, \\ g: \mathbb{N}^2 &\rightarrow \mathbb{N}, & g(x, y, z) &:= z \dot{-} 1. \end{aligned}$$

Sign

Let **sg**: $\mathbb{N} \rightarrow \mathbb{N}$,

$$sg(x) \stackrel{\text{def}}{=} \begin{cases} 0, & \text{if } x = 0, \\ 1, & \text{if } x \neq 0. \end{cases}$$

$$\begin{aligned} sg(0) &\simeq 0, \\ sg(x + 1) &\simeq 1. \end{aligned}$$

$$\overline{sg}(x) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } x = 0, \\ 0, & \text{if } x \neq 0. \end{cases}$$

$$\overline{sg}(x) \simeq 1 \dot{-} sg(x).$$

Other Examples

Absolute Function (ABS): $|x - y| \simeq (x \dot{-} y) + (y \dot{-} x).$

Factorial: $x!$

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

Minimum: $\min(x, y) \simeq x \dot{-} (x \dot{-} y).$

Maximum: $\max(x, y) \simeq x + (y \dot{-} x).$

Remainder

rm(x, y) $\stackrel{\text{def}}{=}$ the remainder when y is divided by x:

$$rm(x, y + 1) \stackrel{\text{def}}{=} \begin{cases} rm(x, y) + 1, & \text{if } rm(x, y) + 1 \neq x, \\ 0, & \text{if } rm(x, y) + 1 = x. \end{cases}$$

The recursive definition is given by

$$\begin{aligned} rm(x, 0) &= 0, \\ rm(x, y + 1) &= (rm(x, y) + 1)sg(|x - (rm(x, y) + 1)|). \end{aligned}$$

Quotient

$qt(x, y) \stackrel{\text{def}}{=} \text{the quotient when } y \text{ is divided by } x:$

$$qt(x, y + 1) \stackrel{\text{def}}{=} \begin{cases} qt(x, y) + 1, & \text{if } rm(x, y) + 1 = x, \\ qt(x, y), & \text{if } rm(x, y) + 1 \neq x. \end{cases}$$

The recursive definition is given by

$$\begin{aligned} qt(x, 0) &= 0, \\ qt(x, y + 1) &= qt(x, y) + \overline{sg}(|x - (rm(x, y) + 1)|). \end{aligned}$$

Definition by Cases

Suppose that $f_1(\mathbf{x}), \dots, f_k(\mathbf{x})$ are computable functions, and $M_1(\mathbf{x}), \dots, M_k(\mathbf{x})$ are decidable predicates, such that for every \mathbf{x} exactly one of $M_1(\mathbf{x}), \dots, M_k(\mathbf{x})$ holds. Then the function $g(\mathbf{x})$ given by

$$g(\mathbf{x}) \simeq \begin{cases} f_1(\mathbf{x}), & \text{if } M_1(\mathbf{x}) \text{ holds,} \\ f_2(\mathbf{x}), & \text{if } M_2(\mathbf{x}) \text{ holds,} \\ \vdots \\ f_k(\mathbf{x}), & \text{if } M_k(\mathbf{x}) \text{ holds.} \end{cases}$$

is computable.

Proof. $g(\mathbf{x}) \simeq c_{M_1}(\mathbf{x})f_1(\mathbf{x}) + \dots + c_{M_k}(\mathbf{x})f_k(\mathbf{x})$.

Conditional Division

$$\text{div}(x, y) \stackrel{\text{def}}{=} \begin{cases} 1, & \text{if } x|y, \\ 0, & \text{if } x \nmid y. \end{cases} : \quad \text{div}(x, y) = \overline{sg}(rm(x, y)).$$

Algebra of decidability

Suppose that $M(\mathbf{x})$ and $Q(\mathbf{x})$ are decidable predicates; then the following are also decidable.

- ① $\text{not } M(\mathbf{x})$
- ② $M(\mathbf{x}) \text{ and } Q(\mathbf{x})$
- ③ $M(\mathbf{x}) \text{ or } Q(\mathbf{x})$

Proof:

- ① $1 - c_M(\mathbf{x})$
- ② $c_M(\mathbf{x}) \cdot c_Q(\mathbf{x})$
- ③ $\max(c_M(\mathbf{x}), c_Q(\mathbf{x}))$

Bounded Sum and Bounded Product

Bounded sum:

$$\sum_{z < 0} f(\mathbf{x}, z) \simeq 0,$$

$$\sum_{z < y+1} f(\mathbf{x}, z) \simeq \sum_{z < y} f(\mathbf{x}, z) + f(\mathbf{x}, y)$$

Bounded product:

$$\prod_{z < 0} f(\mathbf{x}, z) \simeq 1,$$

$$\prod_{z < y+1} f(\mathbf{x}, z) \simeq \left(\prod_{z < y} f(\mathbf{x}, z) \right) \cdot f(\mathbf{x}, y)$$

They are computable if $f(\mathbf{x}, z)$ is total and computable.

Bounded Sum and Bounded Product

By substitution the following functions are also computable

$$\sum_{z < k(\mathbf{x}, \mathbf{w})} f(\mathbf{x}, z)$$

and

$$\prod_{z < k(\mathbf{x}, \mathbf{w})} f(\mathbf{x}, z)$$

if $k(\mathbf{x}, \mathbf{w})$ is total and computable.Bounded Minimization Operator, or μ -Operator $\mu z < y(\dots)$: the least z less than y such that \dots

$$\mu z < y (f(\mathbf{x}, z) = 0) \stackrel{\text{def}}{=} \begin{cases} \text{the least } z < y, & \text{such that } f(\mathbf{x}, z) = 0; \\ y & \text{if there is no such } z. \end{cases}$$

 μ -Operator**Theorem.**If $f(\mathbf{x}, z)$ is total and computable, then so is $\mu z < y (f(\mathbf{x}, z) = 0)$.

Proof

Consider $h(\mathbf{x}, v) = \prod_{u \leq v} \text{sg}(f(\mathbf{x}, u))$ (Computable).

Given \mathbf{x}, y , suppose $z_0 = \mu z < y (f(\mathbf{x}, z) = 0)$. Easy to see,

if $v < z_0$, then $h(\mathbf{x}, v) = 1$;

if $z_0 \leq v < y$, then $h(\mathbf{x}, v) = 0$;

Thus $z_0 = \sum_{v < y} h(\mathbf{x}, v)$.

So $\mu z < y (f(\mathbf{x}, z) = 0) \simeq \sum_{v < y} (\prod_{u \leq v} \text{sg}(f(\mathbf{x}, u)))$ is computable.

Suppose that $R(\mathbf{x}, y)$ is a decidable predicates. Then the following statements are valid:

- ① the function $f(\mathbf{x}, y) \simeq \mu z < y R(\mathbf{x}, z)$ is computable;
- ② the following predicates are decidable:
 - a) $M_1(\mathbf{x}, y) \equiv \forall z < y R(\mathbf{x}, z)$;
 - b) $M_2(\mathbf{x}, y) \equiv \exists z < y R(\mathbf{x}, z)$.

Proof.

- ① $f(\mathbf{x}, y) = \mu z < y (\overline{\text{sg}}(C_R(\mathbf{x}, z)) = 0)$.
- ② a) $c_{M_1}(\mathbf{x}, y) = \prod_{z < y} c_R(\mathbf{x}, z)$.
- b) $M_2(\mathbf{x}, y) \equiv \text{not } (\forall z < y (\text{not } R(\mathbf{x}, z)))$

Bounded Minimization Operator, or μ -Operator

Corollary: If $f(\mathbf{x}, z)$ and $k(\mathbf{x}, \mathbf{w})$ are total and computable functions, then so is the function

$$\mu z < k(\mathbf{x}, \mathbf{w}) (f(\mathbf{x}, z) = 0).$$

Proof. By substitution of $k(\mathbf{x}, \mathbf{w})$ for y in the computable function $\mu z < y (f(\mathbf{x}, z) = 0)$.

Theorem. The following functions are computable.

(a) $D(x)$ = the number of divisors of x ;

(b) $Pr(x) = \begin{cases} 1, & \text{if } x \text{ is prime,} \\ 0, & \text{if } x \text{ is not prime.} \end{cases}$;

(c) p_x = the x -th prime number;

(d) $(x)_y = \begin{cases} k, & k \text{ is the exponent of } p_y \text{ in the prime} \\ & \text{factorisation of } x, \text{ for } x, y > 0, \\ 0, & \text{if } x = 0 \text{ or } y = 0. \end{cases}$.

Proof.

$$(a) D(x) \simeq \sum_{y \leq x} \text{div}(y, x).$$

$$(b) Pr(x) \simeq \overline{\text{sg}}(|D(x) - 2|).$$

(c) p_x can be recursively defined as follows:

$$\begin{aligned} p_0 &\simeq 0, \\ p_{x+1} &\simeq \mu z \leq (p_x! + 1)(z > p_x \text{ and } z \text{ is prime}). \end{aligned}$$

$$(d) (x)_y \simeq \mu z < x(p_y^{z+1} \nmid x).$$

Unbounded Minimization

μ -function:

$$\mu y(f(\mathbf{x}, y) = 0) \simeq \begin{cases} \text{the least } y \text{ such that} \\ (i) \ f(\mathbf{x}, y) \text{ is defined for all } z \leq y, \text{ and} \\ (ii) \ f(\mathbf{x}, y) = 0, \\ \text{undefined if otherwise.} \end{cases}$$

Prime Coding

Suppose $s = (a_1, a_2, \dots, a_n)$ is a finite sequence of numbers. It can be coded by the number

$$b = p_1^{a_1+1} p_2^{a_2+1} \dots p_n^{a_n+1}.$$

Then the length of s can be recovered from

$$\mu z < b((b)_{z+1} = 0),$$

and the i -th component can be recovered from

$$(b)_i \dot{-} 1.$$

Theorem

If $f(\mathbf{x}, y)$ is computable, so is $\mu y(f(\mathbf{x}, y) = 0)$.

Proof

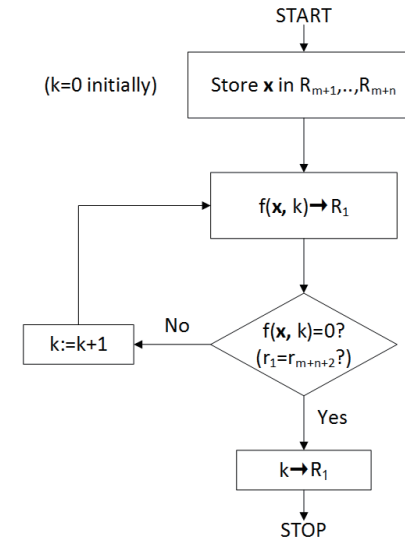
Let F be a program in standard form that computes $f(\mathbf{x}, y)$. Let m be $\max\{n + 1, \rho(F)\}$.

Registers: $[\dots]_1^m [\mathbf{x}]_{m+1}^{m+n} [k]_{m+n+1}^{m+n+1} [0]_{m+n+2}^{m+n+2}$.

Program:

$$\begin{array}{l} T(1, m + 1) \\ \vdots \\ T(n, m + n) \\ I_p : F[m + 1, m + 2, \dots, m + n + 1 \rightarrow 1] \\ J(1, m + n + 2, q) \\ S(m + n + 1) \\ J(1, 1, p) \\ I_q : T(m + n + 1, 1) \end{array}$$

Flow Diagram



Corollary

Suppose that $R(x, y)$ is a decidable predicate; then the function

$$\begin{aligned} g(x) &= \mu y R(\mathbf{x}, y) \\ &= \begin{cases} \text{the least } y \text{ such that } R(\mathbf{x}, y) \text{ holds,} & \text{if there is such a } y, \\ \text{undefined,} & \text{otherwise.} \end{cases} \end{aligned}$$

is computable.

Proof. $g(\mathbf{x}) = \mu y (\overline{\text{sg}}(c_R(\mathbf{x}, y))) = 0$.

Discussion

The μ -operator allows one to define partial functions.

E.g., given $f(x, y) = |x - y^2|$, $g(x) \simeq \mu y (f(x, y) = 0)$,

we have g is the non-total function

$$g(x) = \begin{cases} \sqrt{x}, & \text{if } x \text{ is a perfect square} \\ \text{undefined,} & \text{otherwise.} \end{cases}$$

Remark

Using the μ -operator, one may define total functions that are not primitive recursive.

Remark: The set of primitive recursive functions are those definable from the basic functions using substitution and recursion.

Ackermann Function

Fact. The Ackermann function is computable.

Definition. A finite set \mathbf{S} of triples is said to be **suitable** if the followings hold:

- (i) if $(0, y, z) \in S$ then $z = y + 1$;
 - (ii) if $(x + 1, 0, z) \in S$ then $(x, 1, z) \in S$;
 - (iii) if $(x + 1, y + 1, z) \in S$ then $\exists u. ((x + 1, y, u) \in S) \wedge ((x, u, z) \in S)$.
- Three conditions correspond to the three clauses in the definition of ψ .

The definition of a suitable set \mathbf{S} ensures the following property:

If $(x, y, z) \in \mathbf{S}$, then

- (i) $z = \psi(x, y)$;
- (ii) \mathbf{S} contains all the earlier triple $(x_1, y_1, \psi(x_1, y_1))$ that are needed to calculate $\psi(x, y)$.

Ackermann Function

The **Ackermann function** is defined as follows:

$$\begin{aligned}\psi(0, y) &\simeq y + 1, \\ \psi(x + 1, 0) &\simeq \psi(x, 1), \\ \psi(x + 1, y + 1) &\simeq \psi(x, \psi(x + 1, y)).\end{aligned}$$

Computability Proof

Moreover, for any particular pair of numbers (m, n) there is a suitable set \mathbf{S} such that $(m, n, \psi(m, n)) \in \mathbf{S}$. For instance, let \mathbf{S} be the set of triples $(x, y, \psi(x, y))$ that are used in the calculations of $\psi(m, n)$.

Note a triple (x, y, z) can be coded up by single positive number $2^x 3^y 5^z$. A finite set $\{u_1, \dots, u_k\}$ can be coded up by $p_{u_1} \cdots p_{u_k}$.

Hence a finite set of triples can be coded by a single number v . Let \mathbf{S}_v denote the set of triples coded by the number v . then

$$(x, y, z) \in \mathbf{S}_v \Leftrightarrow p_{2^x 3^y 5^z} \text{ divides } v.$$

So ' $(x, y, z) \in \mathbf{S}_v$ ' is a decidable predicate of x, y, z , and v ; and if it holds, then $x, y, z < v$.

Computability Proof (Cont.)

Let $R(x, y, v)$ be “ v is a legal code and $\exists z < v((x, y, z) \in \mathbf{S}_v)$ ”.

$R(x, y, v)$ is decidable using the techniques and functions of earlier sections.

Thus the function $f(x, y) = \mu v R(x, y, v)$ is a computable function that searches for the code of a suitable set containing (x, y, z) for some z .

As a result, the Ackermann function $\psi(x, y) = \mu z((x, y, z) \in \mathbf{S}_{f(x, y)})$ is computable.

Ackermann Function

The Ackermann function is not primitive recursive.
It grows faster than all the primitive recursive functions.