(정의) 함수 $f: A \to B$가 total이 아닐 때, 즉 $\exists a \in A$, $f(a)$가 정의되지 않을 때, 이 함수를 partial이라고 부르고 $f: A \rightharpoonup B$로 쓴다.

(정의) Minimization(μ)

$g: N^{k+1} \rightharpoonup N$일 때, $f: N^k \rightharpoonup N$를 아래와 같이 정의한다.

$$\forall \vec{x} \in N^k:\ f(\vec{x}) = \mu y\,[g(\vec{x}, y) = 0]$$
$$= Min(\mu)\ y \in N\ [g(\vec{x}, y) = 0\ \wedge\ \exists_{z<y}\ g(\vec{x}, z)]$$

함수 $g(\vec{x}, y)$를 y에 관하여 최소화(minimize)한 함수 $f(\vec{x})$는 $g(\vec{x}, y)$를 0으로 하는 가장 작은 y값 을 가지는데, 이 때 y미만에서는 함수 $g(\vec{x}, y)$가 정의되어 있어야 한다.

Primitive recursion에 minimization을 추가하면 μ-recursive function이라고 한다.

(정의) Partial function g가 **computable**이면 minimization을 가한 함수 f도 **computable**이다.

(증명) **for** $\forall \vec{x} \in N^k$ **do**

```
y := 0;
while (¬ Exit loop) do
    if g(x, y) = 0 then f(x) := y; Exit loop fi
    if g(x, y) = undefined then f(x) := undefined; Exit loop fi
    y := σ(y)
    od
od
```

Turing과 Church의 주장(Turing-Church's Thesis)

(사실) **μ-recursive (partial) function**은 TM **computable**이다.

(정리) **TM**은 **μ-recursive (partial) function**으로 **표현 가능**하다.

(증명) TM $M = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$일 때, TM M이 하는 일을 **μ-recursive partial function** $f: N \rightharpoonup N$로 아래와 같이 정의한다.

TM M의 tape 값을 base가 $|\Gamma|$인 자연수로 표시[1]할 수 있고, 상태와 transition도 마찬가지이다.

TM의 현재 tape head의 왼쪽 tape 스트링이 $\alpha \in \Gamma^*$이고 현재 상태가 $p \in Q$, tape head의 오른쪽 tape 스트링이 $\beta \in \Gamma^*$일 때, TM의 현 상황(configuration)을 (α, p, β)로 표시할 수 있다. 이를 tape 전체의 내용 $w \in N$, 상태 번호 $p \in N$, head 위치 $n \in N$를 이용하여, TM의 현 상황을 (w, p, n)으로 표현 할 수 있다.

1) Tape의 내용을 왼쪽에서부터 읽지 아니하고 오른쪽부터 거꾸로 읽는 것이 좋다.

현 상황 (w, p, n)에서 다음 상황 (w', p', n')을 찾아내는 함수 *step*: $N^3 \rightarrow N^3$을 primitive recursion으로 쉽게 정의할 수 있다[2],

TM M의 현 상황 (w, p, n)과 TM M이 몇 번 수행되었는가를 포함하여 **primitive recursive** 함수 *run*: $N^4 \rightarrow N^3$을 아래와 같이 정의한다.

$$run(w, p, n, 0) = (w, p, n)$$
$$run(w, p, n, t+1) = step(run(w, p, n, t))$$

최종 상태를 하나로 줄여 그 상태 번호를 0이라 하고, 시작 상태 번호가 1이고, 입력 tape 내용이 $w \in N$이고, 시작 할 때 head 위치가 1이라면, 아래와 같이 **μ-recursive partial function**으로 최소 *stoptime*: $N \rightharpoonup N$을 정의할 수 있다.

$$stoptime(w) = \mu t \ [\pi_2^3(run(w, 1, 1, t)) = 0]$$

마지막으로 TM M이 하는 일과 같은 partial 함수 $f : N \rightharpoonup N$을 아래와 같이 정의한다.

$$f(w) = \pi_1^3(run(w, 1, 1, stoptime(w)))$$

우리는 위 두 개의 증명으로 **TM**이 **computable**이라는 **Turing**의 주장과 **μ-recursive partial function**이 **computable**이라는 **Church**의 주장이 **같다**는 사실을 알았다. 이를 **computable**에 관한 Turing과 Church의 **주장**(Turing-Church's Thesis)이라고 부른다, 그러나 두 개가 같다는 증명이 TM이나 μ-recursive function이 **computable**이라는 증명이 될 수는 없다. 즉 TM 이나 μ-recursive function 이상의 계산(computable)이 없다는 증명은 되지 않는다. 그래서 우리는 Theorem이라는 표현 보다는 **Thesis**(**주장**)이라는 표현을 쓴다.

Turing machine이나 μ-recursive function이외에도 λ-calculus(Church 1934-41, Kleene 1935, Rosser 1935), combinatory logic(Schönfinkel 1924, Curry 1929), Post correspondence system(Post 1936), type 0 grammar(Chomsky 1959), while program(Meyer, Richie 1967)등이 모두 **같은 부류**임이 차례로 증명되어 Turing과 Church에 **주장**에 힘을 실어주고 있다.

마지막으로 TM이 **항상 끝나는** 경우를 type 1, recursive라고 부르는데 이것 함수 값이 **항상 정의된** μ-recursive **total** function이라고 보고, **끝나지 않는** 경우도 포함하면 type 0, recursively enumerable라 부르고 함수 값이 **정의되지 않을 수**도 있는 μ-recursive **partial** function이라고 본다.

2) 구체적인 내용은 9장 보조 TP 1 참조.