XVIII
EIGHTEENTH LECTURE

AUTOMATA & FORMAL LANGUAGES

RECAP Lecture $\overline{XVI}$

① $f : \mathbb{N}^k \dashrightarrow \mathbb{N}$ is COMPUTABLE if there is some register machine M s.t. $f = f_{M,k}$

② $X \subseteq \mathbb{N}^k$ is COMPUTABLE if the characteristic function χ_X is computable

Lecture $\overline{XVII}$

③ $X \subseteq \mathbb{N}^k$ is COMPUTABLY ENUMERABLE (C.E.) if the pseudocharacteristic (partial) function ψ_X is computable

RECAP ~~Examples~~

PROJECTIONS : $\pi_{k,i} : \vec{w} \longmapsto w_i$

CONSTANT FUNCTIONS : $c_{k,v} : \vec{w} \longmapsto v$

Shortlex ordering $<$ produces isomorphism

$$\# : (W, <) \longrightarrow (\mathbb{N}, <)$$

Arithmetical functions like $+, \cdot$ can be defined on W:

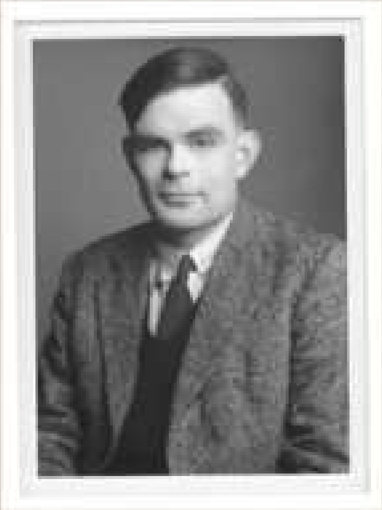
$w + v := u$ iff $\#u = \#w + \#v$.

SUCCESSOR FUNCTION : $s : w \longmapsto v$

where $\#v = \#w + 1$

is also computable.

Alonzo
CHURCH
1903-1995



Alan Turing

230 A. M. TURING [Nov. 12,

ON COMPUTABLE NUMBERS, WITH AN APPLICATION TO THE ENTSCHEIDUNGSPROBLEM

By A. M. TURING.

[Received 28 May, 1936.—Read 12 November, 1936.]

Church had solved the ENTSCHEIDUNGSPROBLEM using a "recursive function" approach before TURING's paper. The referees for TURING's paper for the LMS deemed TURING's approach sufficiently different from Church's to warrant publication.

§ 4.5 Church's recursive functions

Three operations

1. COMPOSITION
2. RECURSION
3. MINIMISATION

The following operations on partial functions were considered by Alonzo Church (1903–1995):
The functions

BASIC FUNCTIONS

$$\pi_{k,i} : \mathbb{W}^k \to \mathbb{W} : \vec{w} \to w_i \text{ (projection functions)}$$
$$c_{k,\varepsilon} : \mathbb{W}^k \to \mathbb{W} : \vec{w} \to \varepsilon \text{ (constant functions)}$$
$$s : \mathbb{W} \to \mathbb{W} : w \mapsto v \quad \text{(where } \#(v) = \#(w) + 1\text{; the successor function).}$$

are called *basic functions*. We have already proved that all basic functions are computable.

Suppose $f : \mathbb{W}^m \dashrightarrow \mathbb{W}$ and $g_1, ..., g_m : \mathbb{W}^k \dashrightarrow \mathbb{W}$ are partial functions, then the partial function h defined by

$$h(\vec{w}) := f(g_1(\vec{w}), ..., g_m(\vec{w}))$$

is called the *composition of f with $(g_1, ..., g_m)$*. The notational convention used for operations applies here as well: if any term on the right hand side is undefined, then so is the left hand side.

$\boxed{\text{m-ary}}$ $f : W^m \dashrightarrow W$

$\boxed{\text{m times k-ary}}$ $g_1, \dots, g_m : W^k \dashrightarrow W$

$\downarrow$

k-ary

$$h(\vec{w}) = f(g_1(\vec{w}), \dots, g_m(\vec{w})).$$

COMPOSITION

RECURSION

Suppose $f : \mathbb{W}^k \dashrightarrow \mathbb{W}$ and $g : \mathbb{W}^{k+2} \dashrightarrow \mathbb{W}$ are partial functions, then the function h defined by the recursion equations

k-ary

k+2-ary

$$\left.\begin{aligned} h(\vec{w}, \varepsilon) &= f(\vec{w}) \text{ and} \\ h(\vec{w}, s(v)) &= g(\vec{w}, v, h(\vec{w}, v)) \end{aligned}\right] \rightsquigarrow \text{k+1-ary}$$

is called the *recursion result of f and g*.

Suppose $f : \mathbb{W}^{k+1} \dashrightarrow \mathbb{W}$ is a partial function, then the partial function h defined by

MINIMI-SATION

k+1-ary

$$h(\vec{w}) := \begin{cases} v & \text{if for all } u \leq v \text{, we have that } f(u)\downarrow \text{ and} \\ & \quad v \text{ is } <\text{-minimal such that } f(\vec{w}, v) = \varepsilon \text{ or} \\ \uparrow & \text{if for all } v,\ f(\vec{w}, v) \neq \varepsilon \end{cases}$$

k-ary

is called the *minimisation result of f*.

We say that a class $\mathcal{C}$ of partial functions is closed under composition, recursion, or minimisation if, whenever $f, g, g_1, \ldots, g_m$ are in $\mathcal{C}$, then the composition of f with $(g_1, \ldots, g_m)$, the recursion result of f and g, or the minimisation result of f, respectively, are in $\mathcal{C}$.

$f : \mathbb{W}^k \dashrightarrow \mathbb{W}$

$g : \mathbb{W}^{k+2} \dashrightarrow \mathbb{W}$

$$\boxed{\begin{aligned} &h(\vec{w}, \varepsilon) = f(\vec{w}) \\ &h(\vec{w}, s(v)) = \\ &\quad g(\vec{w}, v, h(\vec{w}, v)) \end{aligned}}$$

<u>Remark</u> Since f, g partial, recursion can break down, leading to a partial function that is not defined from that point onwards.

Def. We say a class $\mathcal{C}$ of partial functions is closed under comp./rec./minimisation if whenever $f_1, \dots, f_k$ are in $\mathcal{C}$, then the results of applying comp./rec./minimisation are in $\mathcal{C}$.

Clearly $\mathcal{P}$, the class of all partial functions, is closed under all three operations.

Def. We call a partial function **RECURSIVE** if it is in the smallest class $\mathcal{C}$ closed under comp., rec., & minimisation containing all basic functions.

$\mathcal{C} = \bigcap \mathcal{C}'$

$\mathcal{C}'$ is closed under comp., rec., min.

This defines the smallest such class.

We call it **PRIMITIVE RECURSIVE** if it is in the smallest class $\mathcal{C}$ closed under comp. & rec. containing all basic functions.

HISTORICAL NOTE ON TERMINOLOGY.

Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme I[1]).

Von **Kurt Gödel** in Wien.

1.

Die Entwicklung der Mathematik in der Richtung zu größerer Exaktheit hat bekanntlich dazu geführt, daß weite Gebiete von ihr formalisiert wurden, in der Art, daß das Beweisen nach einigen wenigen mechanischen Regeln vollzogen werden kann. Die umfassendsten derzeit aufgestellten formalen Systeme sind das System der Principia Mathematica (PM)[2]) einerseits, das Zermelo-Fraenkelsche (von J. v. Neumann weiter ausgebildete) Axiomensystem der Mengenlehre[3]) andererseits. Diese beiden Systeme sind so weit, daß alle heute in der Mathematik angewendeten Beweismethoden in ihnen formalisiert, d. h. auf einige wenige Axiome und Schlußregeln zurückgeführt sind. Es liegt daher die Vermutung nahe, daß diese Axiome und Schlußregeln dazu ausreichen, alle mathematischen Fragen, die sich in den betreffenden Systemen überhaupt formal ausdrücken lassen, auch zu entscheiden. Im folgenden wird gezeigt, daß dies nicht der Fall ist, sondern daß es in den beiden angeführten Systemen sogar relativ einfache Probleme aus der Theorie der gewöhnlichen ganzen Zahlen gibt[4]), die sich aus den Axiomen nicht

Kurt GÖDEL
1906 - 1978

This is exactly what we just called closure under recursion.

Wir schalten nun eine Zwischenbetrachtung ein, die mit dem formalen System P vorderhand nichts zu tun hat, und geben zunächst folgende Definition: Eine zahlentheoretische Funktion[25]) $\varphi(x_1, x_2 \ldots x_n)$ heißt **rekursiv definiert** aus den zahlentheoretischen Funktionen $\psi(x_1, x_2 \ldots x_{n-1})$ und $\mu(x_1, x_2 \ldots x_{n+1})$, wenn für alle $x_2 \ldots x_n$, k[26]) folgendes gilt:

$$\boxed{\begin{aligned} &\varphi(0, x_2 \ldots x_n) = \psi(x_2 \ldots x_n) \\ &\varphi(k+1, x_2 \ldots x_n) = \mu(k, \varphi(k, x_2 \ldots x_n), x_2 \ldots x_n). \end{aligned}} \qquad (2)$$

Eine zahlentheoretische Funktion φ heißt **rekursiv**, wenn es eine endliche Reihe von zahlentheor. Funktionen $\varphi_1, \varphi_2 \ldots \varphi_n$ gibt, welche mit φ endet und die Eigenschaft hat, daß jede Funktion φ_k der Reihe entweder aus zwei der vorhergehenden rekursiv definiert ist oder

[25]) D. h. ihr Definitionsbereich ist die Klasse der nicht negativen ganzen Zahlen (bzw. der n-tupel von solchen) und ihre Werte sind nicht negative ganze Zahlen.

[26]) Kleine lateinische Buchstaben (ev. mit Indizes) sind im folgenden immer Variable für nicht negative ganze Zahlen (falls nicht ausdrücklich das Gegenteil bemerkt ist).

Gödel called what we nowadays call "primitive recursive" recursive.

Terminology changed with Church.

Example

(1) $\pi_{1,0} : \mathbb{W}^1 \longrightarrow \mathbb{W}$ primitive recursive
$= id$

(2) $\pi_{3,2} : \mathbb{W}^3 \longrightarrow \mathbb{W}$ primitive recursive
$(w,v,u) \longmapsto u$

(3) $s : \mathbb{W} \longrightarrow \mathbb{W}$ primitive recursive
$w \longmapsto s(w)$

(4) $s \circ \pi_{3,2} : (w,v,u) \longmapsto s(\pi_{3,2}(w,v,u))$
$= s(u)$.

primitive rec., as comp. of (2) & (3)

(5)
$$h(w, \varepsilon) = \pi_{1,0}(w) = w.$$
$$h(w, s(v)) = s \circ \pi_{3,2}(w, v, h(w,v))$$
$$= s(h(w,v))$$

Thus h is primitive recursive.

$$h(x,0) := x$$
$$h(x, m+1) := h(x,m)+1$$

$$\boxed{x+m := h(x,m)}$$

This is the famous Grassmann recursion equation for addition.

$h : \mathbb{W}^2 \longrightarrow \mathbb{W}$ is the function corresponding to addition (relative to shortlex) on $\mathbb{W}$:

$$h(w,v) = u \iff \#u = \#w + \#v.$$

The other Grassmann recursion equations:

$$h(x,0) := 0 \qquad\qquad h(x,0) := 1$$

$$h(x,m+1) := h(x,m) + x \qquad h(x,m+1) := h(x,m) \cdot x$$

MULTIPLICATION EXPONENTIATION

This allows to do precisely the same as with addition & get functions h_m, h_e s.t.

$$h_m(w,v) = u \iff \#u = \#w \cdot \#v$$

$$h_e(w,v) = u \iff \#u = \#w^{\#v}.$$

These are primitive recursive.

We encode recursion functions in trees:

T finitely branching tree

ℓ labelling on T with the following labels:

		ARITY	BRANCHING NUMBERS
PROJECTION (BASIC)	$B^{\pi}_{k,i}$	k	0
CONSTANT (BASIC)	$B^{c}_{k,i}$	k	0
SUCCESSOR (BASIC)	B^{S}	1	0
COMPOSITION	$C_{n,k}$	k	$n+1$
RECURSION	R_k	$k+1$	2
MINIMISATION	M_k	k	1

Corrected after the lecture!

A labelled tree (T,ℓ) is called a recursion tree if the branching of the tree corresponds exactly to branching numbers of its labels and

① if $\ell(s) = C_{n,k}$, then the first successor of s has a label with arity n and others labels with arity k.

② if $\ell(s) = R_k$, then the first successor has a label with arity k and second arity $k+2$

③ if $\ell(s) = M_k$ then the unique succ. has arity $k+1$.

Such a tree is called primitive if it has no M-labels.

Our example

$$\begin{tikzcd}[column sep=small]
 & R_1 \arrow[dl, no head] \arrow[dr, no head] & & \\
B^{\pi}_{1,0} & & C_{1,3} \arrow[dl, no head] \arrow[dr, no head] & \\
 & B^{s} & & B^{\pi}_{3,2}
\end{tikzcd}$$

Corrected after the lecture!

is the recursion tree that corresponds to our example.

(It is primitive.)

We can assign functions to computation trees in the obvious way:

$$f_{T,e} : \mathbb{N}^k \dashrightarrow \mathbb{N}$$

Definition of $f_{T,\ell}$ by recursion on the height of tree.

By recursion on the height of the tree, we can assign functions $f_{T,\ell}$ to each non-empty labelled tree (T, ℓ) such that the arity of the function assigned to the root is equal to the arity of the label $\ell(\varepsilon)$:

(a) By construction, a labelled tree has height 1 if and only if $\ell(\varepsilon)$ is a basic label, i.e., $\mathsf{B}^{\pi}_{k,i}$, B^{c}_{k}, or B^{s}. In this case, let $f_{T,\ell} = \pi_{k,i}$, $f_{T,\ell} = c_{k,\varepsilon}$, or $f_{T,\ell} = s$, respectively.

(b) Suppose the height of the tree is $k+1 > 1$ and $\ell(\varepsilon) = \mathsf{C}_{n,k}$. Recursively, we assume that the construction is already done for all trees of height $\leq k$. Note that all of the subtrees starting with the immediate successors of ε are labelled trees with height $\leq k$, so we have already assigned functions of the right arity to them. By construction, ε has $n+1$ successors: the first one is assigned a function f of arity n, and all others are assigned functions g_i of arity k. We let $f_{T,\ell}$ be the composition of f with $(g_1, \ldots, g_n)$.

(c) Suppose the height of the tree is $k+1 > 1$ and $\ell(\varepsilon) = \mathsf{R}_k$. By construction, ε has two successors: the first one is assigned a function f of arity k and the second one is assigned a function g with arity $k+2$. We let $f_{T,\ell}$ be the recursion result of f and g.

(d) Suppose the height of the tree is $k+1 > 1$ and $\ell(\varepsilon) = \mathsf{M}_k$. By construction, ε has a unique successor that is assigned a function f of arity $k+1$. We let $f_{T,\ell}$ be the minimisation result of f.

Theorem 4.22 A function f is recursive iff there is a recursion tree (T,ℓ) s.t. $f = f_{T,\ell}$.

Also: a function f is primitive recursive iff there is a primitive recursion tree (T,ℓ) s.t. $f = f_{T,\ell}$.

Proof. "$\Longleftarrow$" is just induction on the height of the recursion tree.

"$\Longrightarrow$" we need to show that the class of all functions of type

$$f_{T,\ell}$$

for some recursion tree (T,ℓ) contains the basic fns & is closed under comp., rec., minimisation.

That's true by construction.

q.e.d.