Theories: understood to be deductively closed set of sentences

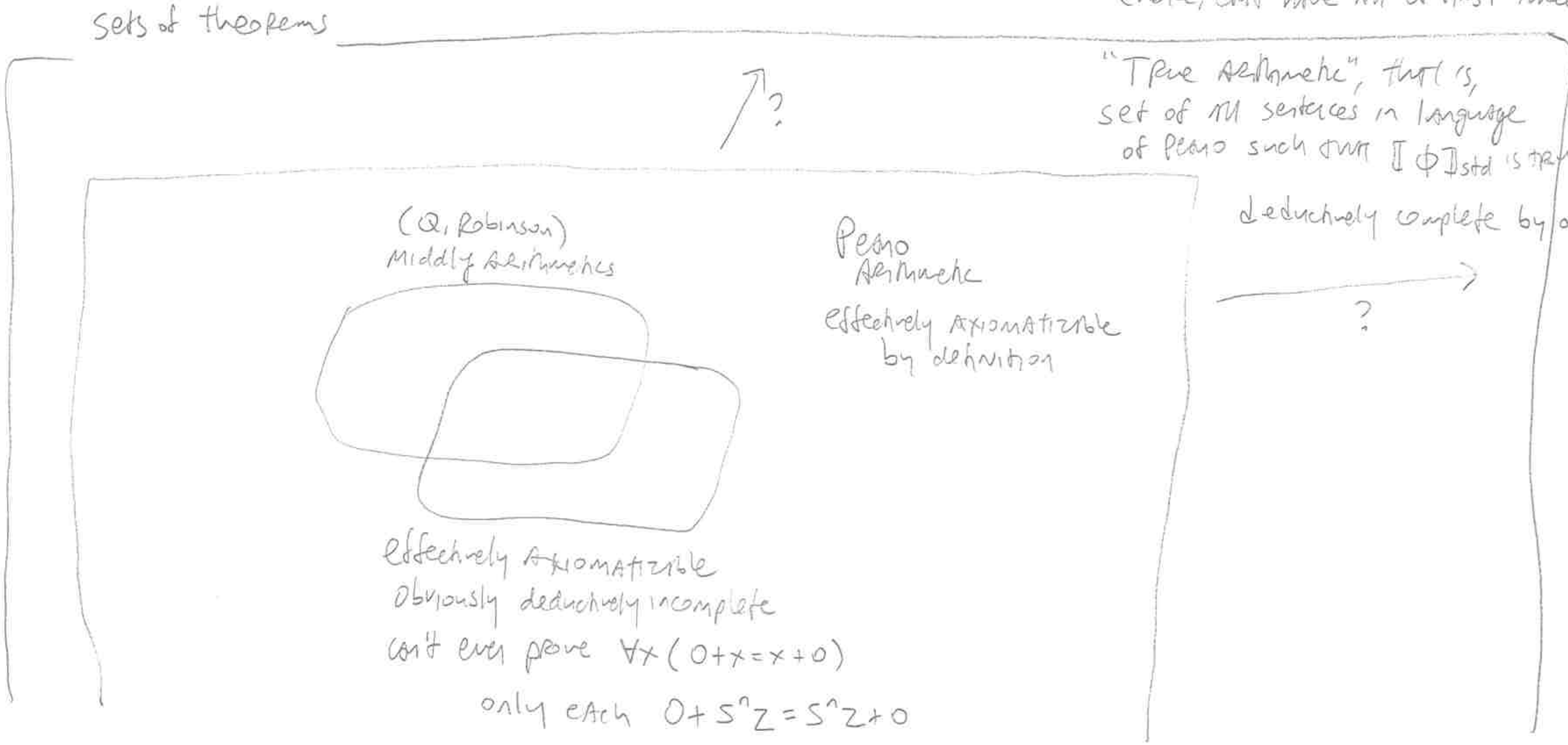
may or may not be
- Consistent
- deductively complete (Smith's "negation complete")
- effectively axiomatizable
- effectively decidable

Homework 10
problems 108–109

(Smith's Thm 7 in Ch 5)

deductively complete & effectively axiomatizable $\longrightarrow$ effectively decidable $\not\longleftarrow$

Gödel's First Incompleteness Thm:
no strengthening of mildly arithmetics can have all four of those properties
(hence, can't have all of first three either)



Calculus/System/Algebra of Recursive Functions

Base functions

Picker/Identity/Projection Funcs

- $Id(x) = 1of1(x) = x$
- $1of2(x_1, x_2) = x_1 \quad 2of2(x_1, x_2) = x_2$
- $1of3(x_1, x_2, x_3) = x_1$ etc

Successor Func

- $S(x) = x+1$

zero Func(s)

- $\boxed{Z_0() \ ?}$
- $Z_1(x) = 0$
- $Z_2(x_1, x_2) = 0 = Z_1 \circ 1of2 = Z_1 \circ 2of2$
- $Z_3(x_1, x_2, x_3) = 0 = Z_1 \circ 1of3 = Z_1 \circ 2of3 = \ldots$

I'll write $x_1, \ldots x_n$ as $\bar{x}$
$x_0, \ldots x_{n-1}$
and $\#X$ will be N

I'll write $\#f$ for the number of args that f expects

Builder Functions

- Generalized Composition

where g expects $\#g$ args
and $f_1, \ldots f_{\#g}$ each expect $\#f$ args
then $g \circ (f_1, \ldots, f_{\#g}) \underbrace{(x_1, \ldots x_{\#f})}_{\text{call this } \bar{x}} = g(f_1(\bar{x}), \ldots f_{\#g}(\bar{x}))$

definable in terms of those primitives

$\boxed{\text{FirstZeroBefore}(f, lim)}$ / $\boxed{\text{LastZeroBefore}(f, lim)}$ $(\bar{x}) = \begin{cases} y \text{ when } y < lim \text{ and } f(\bar{x}, y) = 0 \text{ and } f(\bar{x}, y') > 0 \text{ for all } y' < y \\ \text{else } lim \text{ if there is no such } y < lim \end{cases}$

Primly Recursive funcs always total

- Prim Recursion

$\boxed{Rek(basef, buildf)}(\bar{x}, 0) = basef(\bar{x})$ ← where we might s/t want $Z_0()$

$(\bar{x}, Sk) = buildf(\bar{x}, prev, k)$
↑ where $prev = \boxed{Rek(basef, buildf)}(\bar{x}, k)$

(not-necely primly) Recursive funcs may be total or partial

- Minimization (unbounded search for result of 0)

$\boxed{\text{FirstZero}(f)}(\bar{x}) = \begin{cases} y \text{ when } f(\bar{x}, y) = 0 \text{ and } f(\bar{x}, y') \text{ is defined and } > 0 \text{ for all } y' < y \\ \text{Undefined if there is no such } y \end{cases}$

where f expects $\#x + 1$ args

Primitively Recursive total funcs on $\mathbb{N}$

total (not-necessarily-primitively) Recursive funcs on $\mathbb{N}$ whose canonical specifications are effly recognizable

All total (not-necessarily-primitively) Recursive funcs on $\mathbb{N}$

partial (not-necessarily-primitively) Recursive funcs on $\mathbb{N}$
All of their canonical specifications are effly recognizable

"PR" sometimes used to mean "primitively recursive"
sometimes to mean "partial (not-necely-prim) recursive"

Examples of Some Primitively Recursive Functions

SCRATCH WORK | **So officially**

$\text{plus}(x,y) =$ when y is 0, x
when y is Sk, $S\underbrace{(x+k)}_{\text{prev}}$

So $\text{basef}(x) = Id(x)$
So $\text{buildf}(x, \text{prev}, k) = S\,\text{prev} = (S \circ 2\text{of}3)(x, \text{prev}, k)$

$\text{plus} = \text{Rek}(Id,\ S \circ 2\text{of}3)$

$\text{mul}(x,y) =$ when y is 0, 0
when y is Sk, $\text{plus}(\underbrace{x \cdot k}_{\text{prev}}, x)$

so $\text{basef}(x) = \text{zero}_1(x)$
so $\text{buildf}(x, \text{prev}, k) = \text{plus}(\text{prev}, x) = (\text{plus} \circ (2\text{of}3, 1\text{of}3))(x, \text{prev}, k)$

$\text{mul} = \text{Rek}(\text{zero},\ \text{plus} \circ (2\text{of}3, 1\text{of}3))$

$\text{fact}(y) =$ when y is 0, 1
when y is Sk, $\text{mul}(\underbrace{k!}_{\text{prev}}, Sk)$

so $\text{basef}() = S(\text{zero}_0()) = (S \circ \text{zero}_0)()$
so $\text{buildf}(x, \text{prev}, k) = \text{mul}(\text{prev}, Sk) = (\text{mul} \circ (1\text{of}2,\ S \circ 2\text{of}2))(\text{prev}, Sk)$

$\text{fact} = \text{Rek}(S \circ \text{zero}_0,\ \text{mul} \circ (1\text{of}2,\ S \circ 2\text{of}2))$

OR

$\text{fact}(x,y) =$ when y is 0, 1
when y is Sk, $\text{mul}(\underbrace{k!}_{\text{prev}}, Sk)$
(any x always ignored)

so $\text{basef}(x) = S(\text{zero}_1(x)) = (S \circ \text{zero}_1)(x)$
so $\text{buildf}(x, \text{prev}, k) = \text{mul}(\text{prev}, Sk) = (\text{mul} \circ (2\text{of}3,\ S \circ 3\text{of}3))(x, \text{prev}, Sk)$

$\text{fact} = \text{Rek}(S \circ \text{zero}_1,\ \text{mul} \circ (2\text{of}3,\ S \circ 3\text{of}3))$

"Ackermann function"

let $H_1(b,n) = b+n$

"hyper-operations"

$H_2(b,n) = b \cdot n = \underbrace{b+(b+(b+\ldots))}_{n \text{ times}}$

$H_3(b,n) = b \uparrow n = \underbrace{b \cdot (b \cdot (b \cdot \ldots))}_{n \text{ times}}$

$H_4(b,n) = b \uparrow\uparrow n = \underbrace{b\uparrow(b\uparrow(b\uparrow\ldots))}_{n \text{ times}} = \left. b^{b^{\cdot^{\cdot^{b}}}} \right\}$ tower of n

$H_5(b,n) = b \uparrow\uparrow\uparrow n = b \Uparrow (b \Uparrow (b \Uparrow \ldots))$

$\boxed{\text{Then } A((x,y)) = H_y(2,x)}$ = Péter/Robinson's $\text{Ack}(y, x-3)+3$

$A(x,1) = 2+x$
$A(x,2) = 2x$
$A(x,3) = 2\uparrow x$
$A(x,4) = 2 \Uparrow x$
...

	$x=3$	$x=4$	$x=5$	$x=6$
$y=1$	5	6	7	8
$y=2$	6	8	10	12
$y=3$	8	16	32	64
$y=4$	16	65536	2^{65536}	$2^{2^{65536}}$
...				

Can some formal **language** (the language of some theory T) "**express**" (or arithmetically defines) a property or relation R / a function f } these symbols identify/describe the relation/func in our metalanguage

formula ϕ expresses R =def

p.16 Smith p.23 when $R(\bar{x})$ then $[\![\phi(S^{x_1}z, \ldots S^{x_{\#x}}z)]\!]_{std}$ is true, where std is the intended/standard model for T

when $\neg R(\bar{x})$ then $[\![\phi(S^{x_1}z \ldots S^{x_{\#x}}z)]\!]_{std}$ is false, that is

$[\![\neg\phi(S^{x_1}z, \ldots S^{x_{\#x}}z)]\!]_{std}$ is true

formula ψ expresses f =def

when $f(\bar{x}) = y$ then $[\![\psi(S^{x_1}z, \ldots S^{x_{\#x}}z, S^{y}z)]\!]_{std}$ is true

when $f(\bar{x}) \neq y$ then $[\![\psi(S^{x_1}z \ldots S^{x_{\#x}}z, S^{y}z)]\!]_{std}$ is false, that is

$[\![\neg\psi(S^{x_1}z, \ldots S^{x_{\#x}}z, S^{y}z)]\!]_{std}$ is true

won't necessarily be any term η such that $\psi(\ldots, u)$ is $\eta(\ldots) = u$

Can the **theory** "**capture**" (case-by-case prove) (or represents) the property or relation R / the function f

Smith p.23 when $R(\bar{x})$ then $T \vdash \phi(S^{x_1}z \ldots S^{x_{\#x}}z)$

when $\neg R(\bar{x})$ then $T \vdash \neg\phi(S^{x_1}z \ldots S^{x_{\#x}}z)$ ← note stronger than just $T \nvdash \phi(S^{x_1}z, \ldots S^{x_{\#x}}z)$

p.69 Smith p.75 when $f(\bar{x}) = y$ then $T \vdash \psi(S^{x_1}z \ldots S^{x_{\#x}}z, S^{y}z)$

when $f(\bar{x}) \neq y$ then $\boxed{T \vdash \neg\psi(S^{x_1}z \ldots S^{x_{\#x}}z, S^{y}z)}$

or when $f(\bar{x}) = y$ then
$T \vdash \forall w(\psi(S^{x_1}z \ldots S^{x_{\#x}}z, w) \supset w = S^{y}z)$
(replaces both clauses)

instead stronger claim that there is just one u such that $\psi(\bar{x}, u)$
$T \vdash \exists u \forall w(\psi(S^{x_1}z \ldots S^{x_{\#x}}z, w) \supset w = u)$

Facts

The "middling arithmetics" (and stronger first-order theories, like Peano arithmetic)
can "capture" all primitively recursive functions

(Smith Ch17) in fact all (total or partial) (not-necessarily primitive) recursive functions
and by formulas that have "quantifier complexity Σ_1"
(aka $\exists$-Rudimentary) (Smith Ch 8)

e.g. $zero_1(x)$ expressed/captured by $Z = \square$

$succ(x)$ by $S(\underline{S^x Z}) = \square$

$pred(x)$
where $pred(0)=0$ by $(\underline{S^x Z} = Z \wedge \square = Z) \vee (\underline{S^x Z} = S\square)$

$div(n,d)$
where $div(n,0)=0$ by $(\underline{S^d Z} = Z \wedge \square = Z) \vee \exists u\,(u < \underline{S^d Z} \wedge \underline{S^n Z} = \square \cdot \underline{S^d Z} + u)$

$Rem(n,d)$
where $Rem(n,0)=n$ by $(\underline{S^d Z} = Z \wedge \square = \underline{S^n Z}) \vee$
$(\square < \underline{S^d Z} \wedge \exists u\,(u \leq \underline{S^n Z} \wedge \underline{S^n Z} = u \cdot \underline{S^d Z} + \square))$

$Rec(basef, buildf)(\bar{x}, y)$ by $\exists q\,($ decode$(q,0) =$ basef$(\underline{S^{x_1}Z} \ldots \underline{S^{x_{\#x}}Z})$
$\wedge\ \forall u\,(u < \underline{S^y Z} \supset$ decode$(q, Su) =$ buildf$(\underline{S^{x_1}Z} .. \underline{S^{x_{\#x}}Z},$ decode$(q,u), u))$
$\wedge$ decode$(q, \underline{S^y Z}) = \square\,)$

encodes a sequence

Gödel's β-function coding of sequences

In Smith's Ch10
my decode(q,i) =
$\beta(c,d,i) = Rem(c, d(i+1)+1)$