TEORIA DAS FUNÇÕES RECURSIVAS

E

APLICAÇÕES À LÓGICA

IRALINO FIDÊNCIO CENTENARO

ORIENTADOR

PROF. DR. ANDRÉS RAGGIO

Dissertação apresentada ao Instituto de Matemática, Estatística e Ciência da Computação da Universidade Estadual de Campinas como requisito parcial para obtenção do título de Mestre em Matemática.

Este trabalho foi realizado com o auxílio financeiro da Ökumenisches Studienwerk e.V. - Bochum/Germany, da Financiadora de Estudos e Projetos(FINEP) e da Coordenação do Aperfeiçoamento de Pessoal de Nível Superior(CAPES).

maio de 1977

A meus pais Ulisses e Maria

AGRADECIMENTOS

Ao *Prof. Andrés Raggio* pela proposição do presente trabalho e pela sua segura e disponível orientação permitindo a elaboração do mesmo.

Aos meus pais, professores e colegas por seus estímulos e ensinamentos.

Aos colegas de *república* pelo apoio moral.

À Ökumenisches Studienwerk e.V., à FINEP, e à CAPES que, com seus apoios financeiros, permitiram a realização deste trabalho.

À FIDENE - IJUÍ/RS pelo incentivo ao meu aperfeiçoamento universitário.

Aos professores Wlademir Guimarães, Hannibal Barca de Lima e Castro e Ubiratan D'Ambrósio pela consideração e ajuda recebidas.

TEORIA DAS FUNÇÕES RECURSIVAS E APLICAÇÕES À LÓGICA

I N T R O D U Ç Ã O

A noção de sistema axiomático supõe as noções de propriedade efetiva e de regra efetiva. De fato, temos que saber efetivamente se uma sequência de símbolos é ou não um axioma ou se é ou não a aplicação correta de uma regra. Como a teoria das Funções Recursivas estuda essas noções de propriedade efetiva e regra efetiva não é surprendente que a aplicação desta teoria ao estudo dos sistemas axiomáticos possa permitir demonstrar resultados profundos. O que não se pôde prever foi aquilo que ocorreu na década de 30, quando a aplicação das Funções Recursivas aos sistemas axiomáticos produziu resultados espetaculares, como os obtidos por Gödel e Church. Hoje em dia, é possível dar a esses resultados uma forma muito elegante e muito breve. Neste trabalho, damos alguns exemplos desta possibilidade.

No Capítulo I, daremos uma caracterização formal de Função Recursiva, apresentando uma caracterização matemática da classe de objetos chamados Máquinas de Turing e mostrando a equivalência que existe entre Função Recursiva e função computável por uma Máquina de Turing.

No capítulo II, estudaremos propriedades recursivas de conjuntos de inteiros, dando ênfase ao fato de que um problema é solúvel se o correspondente conjunto de inteiros é recursivo. Como mediante uma gödelização podemos considerar o conjunto de fórmulas que se obtêm usando axiomas e regras como um conjunto de inteiros, apresentamos também, o resultado fundamental da aplicação da Teoria das Funções Recursivas aos sistemas axiomáticos. Mostraremos também que o conjunto

$$K = \{x \;/\; \phi_x(x) \text{ é convergente}\}$$

é recursivamente enumerável. Assim, no Capítulo III, daremos um sentido construtivo ao fracasso de $\overline{K}$ em ser recursivamente enumerável definindo então, a importante classe dos conjuntos criativos e produtivos.

Usando o resultado fundamental da aplicação da Teoria das Funções Recursivas aos sistemas axiomáticos, veremos no Capítulo IV, a aplicação de tal teoria num sistema lógico específico, a Aritmética Elementar, apresentando antes a utilidade da Teoria das Funções Recursivas na demonstração do Teorema da Incompletude de Gödel.

C A P Í T U L O I

FUNÇÕES RECURSIVAS E MÁQUINAS DE TURING

Neste capítulo daremos uma caracterização formal, matematicamente exata, de Função Recursiva. Para isto, utilizaremos a caracterização de Turing, apresentando uma caracterização matemática de uma classe de objetos chamados Máquinas de Turing, e mostrando a equivalência que existe entre Função Recursiva e função computável por uma Máquina de Turing. Assim tornaremos precisa a noção matemática intuitiva de função computável " por Algoritmo".

§ 1.1 NOÇÃO INFORMAL DE ALGORITMO

Como uma preliminar à caracterização formal, apresentamos certos aspectos das noções não formais de algoritmo e função computável por algoritmo, como elas ocorrem na matemática.

DEFINIÇÃO 1.1.1 Um *algoritmo* é um procedimento determinístico aplicado a qualquer classe de "inputs"(entradas) simbólicos e que eventualmente produzirá, para cada input, um "output" (saida) simbolico correspondente.

EXEMPLO 1 *Processo usual dado em cálculo elementar para diferenciar polinômios.*

Para nossos propósitos, consideraremos apenas algoritmos que produzem, como outputs, inteiros ou k-uplas de inteiros para um k fixado, em alguma notação padrão. Por exemplo, os números arábicos. Assim, para nós

DEFINIÇÃO 1.1.2 Um *algoritmo* é um procedimento para computar uma função com relação a alguma notação escolhida para inteiros.

EXEMPLO 2 *Funções que possuem algoritmos bem conhecidos:*

a) λx (x-ésimo número primo). O método do crivo de Eratosthenes é um algoritmo para esta função.

b) λxy (o maior divisor comum de x e y). O algoritmo Euclidiano.

c) λxy (x + y).

Descrição aproximada e intuitiva das características da noção não formal de algoritmo:

1.- Um algoritmo é dado como um conjunto finito de instruções.

2.- Existe um agente, usualmente humano, que computa, reagindo às instruções e executando as computações.

3.- Existem facilidades para executar, estocar e recuperar passos em uma computação.

4.- Seja P um conjunto de instruções como em 1.- e seja L um agente como em 2.-. Então L reage a P de tal modo que, para todo o input dado, a computação é executada discretamente, passo a passo, sem uso de métodos contínuos ou mecanismos análogos.

5.- L reage a P de tal modo que uma computação é levada adiante deterministicamente, sem recorrer a métodos ou mecanismos ao acaso. Exemplo: jogo de dados.

Essas características de 1.- a 5.-, embora afirmadas inexatamente, são inerentes à idéia de algoritmo. Podemos notar uma analogia com as máquinas de computar digitais: 1.- corresponde ao programa de um computador, 2.- a seus elementos lógicos, 3.- à sua memória estocada, 4.- à sua natureza digital e 5.- à sua natureza mecânica.

DEFINIÇÃO 1.1.3 Chamamos de *P-simbolismo* à especificação de expressões simbólicas que são aceitas como conjunto de instruções, como inputs e como outputs.

DEFINIÇÃO 1.1.4 Chamamos de *L-P especificações* à maneira como algumas instruções e o input determinam, num caminho uniforme, a subseqüente computação e como o output dessa

computação será identificado.

As características de 1.- a 5.- servem como um guia instrutivo para uma escolha útil do P-simbolismo e L-P especificações. Notemos que existiriam as seguintes características que são menos óbvias que 1.- a 5.- se existisse uma resposta para as seguintes perguntas.

6.- Existirá um limite fixo finito sobre o tamanho de inputs?

7.- Existirá um limite fixo finito sobre o conjunto de instruções?

8.- Existirá um limite fixo finito sobre o total de "memória" armazenada no espaço útil ?

O tamanho aqui é medido pelo número de símbolos elementares usados. Assumimos uma resposta negativa a 6.- e 7.-, pois uma teoria geral de algoritmos deverá referir-se a computações e em princípio são possíveis, deixando de lado considerações de ordem de limitação prática. Da mesma maneira responderemos negativamente a 8.-. Por outro lado, assumiremos uma resposta positiva para:

9.- Existirá, em algum sentido, um limite fixo finito sobre a capacidade ou habilidade do agente computante L ?

Para a resposta à próxima pergunta, apenas exigimos que a computação termine:

10.- Existirá, em algum caminho, um limite sobre a duração de uma computação ?

§ 1.2 FUNÇÕES PRIMITIVAS RECURSIVAS

Caracterizemos uma classe de funções tomando como elementos dessa classe todas as funções obtidas por definições recursivas.

DEFINIÇÃO 1.2.1 Uma *Definição Recursiva* para uma função é, numa idéia intuitiva, uma definição onde valores da função para argumentos dados são diretamente obtidos para valores da mesma função para o "mais simples" argumento ou para os valores das "mais simples funções".

A notação "mais simples" é tomada na caracterização escolhida como funções constantes, identidades, sucessor e outras .

Definições recursivas podem ser dadas para servir como algoritmos, sendo muito familiares em matemática.

EXEMPLO 3 *A função f definida por*

$$f(0) = 1$$

$$f(1) = 1$$

$$f(x+2) = f(x+1) + f(x)$$

dá a sequência de FIBONACCI: 1,1,2,3,5,8,13,.....

As funções primitivas Recursivas podem ser obtidas pela caracterização formal seguinte:

DEFINIÇÃO 1.2.2 *A classe de Funções Primitivas Recursivas* é a menor classe C isto é, intersecção de todas as classes C de funções tais que:

i) todas as funções constantes, $\lambda x_1 x_2 \ldots x_k (m)$, estão em C, $1 \leq k$, $0 \leq m$;

ii) a função sucessor, $\lambda x (x + 1)$, está em C;

iii) todas as funções identidade, $\lambda x_1 \ldots x_k (x_i)$, estão em C, $1 \leq i \leq k$;

iv) se f é uma função de k variáveis em C, e $g_1, g_2, \ldots, g_k$ são cada uma, funções de m variáveis em C, então a função $\lambda x_1 \ldots x_m (f(g_1(x_1,\ldots,x_m),\ldots,g_k(x_1,\ldots,x_m)))$ está em C, $1 \leq k,m$;

v) se h é uma função de k + 1 variáveis em C, e g é uma função de k - 1 variáveis satisfazendo

$$f(0,x_2,\ldots,x_k) = g(x_2,\ldots,x_k),$$

$$f(y + 1,x_2,\ldots,x_k) = h(y, f(y,x_2,\ldots,x_k),x_2,\ldots,x_k)$$

está em C, $1 \leq k$.

Observemos que em v), a "função de zero variáveis em C" é tomada para significar um inteiro fixado. Segue diretamente da definição

LEMA 1.2.1 f é primitiva recursiva $\Longleftrightarrow$ existe uma sequência de funções $f_1,\ldots,f_n$ tais que $f_n = f$ e para cada $j \leq n$, outra f_j está em C por i), ii), ou iii), ou f_j é diretamente obtida por alguma das f_i, $i \leq j$, por iv) ou v).

DEMONSTRAÇÃO Seja D a classe de todas as funções f para as quais existe uma tal sequência de funções $f_1,\ldots,f_n$.

D está evidentemente contida em toda a classe C que é fechada sobre i) a v), donde D é do mesmo modo fechada sobre i) a v). Portanto, D coincide com a intersecção de toda a tal classe C.

DEFINIÇÃO 1.2.3 Se uma sequência como no lema 1.2.1 é descrita para f, juntamente com uma especificação de como cada f_j é obtida para $j \leq n$, teremos uma *Derivação* para f como uma função Primitiva Recursiva.

EXEMPLO 4 *Consideremos a função f dada pela derivação:*

	por	função de
$f_1 = \lambda\ x\ (x)$	iii)	(1 variável)
$f_2 = \lambda\ x(x+1)$	ii)	(1 variável)
$f_3 = \lambda\ x_1x_2x_3(x_2)$	iii)	(3 variáveis)
$f_4 = f_2f_3$	iv)	(3 variáveis)
f_5 satisfazendo		
$f_5(0,x_2) = f_1(x_2)$		
$f_5(y+1,x_2) = f_4(y,f_5(y,x_2),x_2)$	v)	(2 variáveis)
$f = f_6 = f_5(f_1,f_1)$	iii)	(1 variável)

Como é fácil verificar que f_6 é a função $\lambda\ x(\ 2x\ )$ e, acidentalmente, que f_5 é $\lambda xy\ (\ x + y\ )$, concluimos que a função $\lambda x\ (\ 2x\ )$ é primitiva Recursiva.

Uma derivação serve como um conjunto de instruções para computar efetivamente a função que a define.

EXEMPLO 5 *Computando* f(2) *no exemplo* 4, *a derivação nos leva à seguinte computação:*

$$
\begin{aligned}
f_2 &= f_6\ (2)\\
&= f_5(f_1(2), f_1(2))\\
&= f_5(2, f_1(2))\\
&= f_5(2,2)\\
&= f_4(1, f_5(1,2), 2)\\
&= f_4(1, f_4(0, f_5(0,2), 2), 2)\\
&= f_4(1, f_4(0, f_1(2), 2), 2)\\
&= f_4(1, f_4(0,2,2), 2)\\
&= f_4(1, f_2(f_3(0,2,2)), 2)\\
&= f_4(1, f_2(2), 2)\\
&= f_4(1,3,2)\\
&= f_2(f_3(1,3,2))\\
&= f_2(3)\\
&= 4
\end{aligned}
$$

Chegamos à computação unicamente pelo assíduo trabalho de ir à direita e à esquerda.

Notemos qué o agente computante é humano e não formalmente definido, porem a computação depende da Derivação e de um caminho simples e direto que nos leva a passos mecânicos, satisfazendo 1.- a 5.- de § 1.1.

Podemos escolher um P-simbolismo usual para expressar derivações e as L-P especificações são as regras simples de substituição para que a derivação e o input determinem a computação .

Virtualmente todas as funções algorítmicas da matemática ordinária são mostradas como primitivas recursivas, veja (9). Por outro lado, é pssível construir funções com algoritmos óbvios, que não são primitivas recursivas. Um tal algoritmo é a "Generalização da Exponencial de Ackermann", veja (11) p. 8-9.

§ 1.3 MÁQUINAS DE TURING

Partimos do fato de que, se existe um algoritmo para realizar uma tarefa, então, pelo menos em princípio, podemos construir uma máquina de computar para realizar essa tarefa. Tal máquina de computar é determinística no seguinte sentido: enquanto é uma operação, todo seu futuro é especificado completamente pelo seu estado para um momento . Assim, daremos uma caracterização matemática de uma classe de objetos que chamaremos *Máquinas de Turing* (M.Tu.). Elas serão definidas por analogia a computadores físicos de um certo tipo. Com isto chegaremos a uma caracterização matemática de uma classe de funções numéricas, as funções computáveis por essas M.Tu.. Estas funções serão chamadas "*Funções computáveis*" e identificaremos o conceito intuitivo de função efetivamente calculável com o novo conceito preciso de função computável.

Enquanto as máquinas de computar físicas são desvantajosas por terem somente uma região finita eficaz para estocar da-

dos de entrada, nós introduzimos uma máquina de computar que imprime símbolos num "tape" linear, assumindo como sendo infinito em ambas as direções, regular numa sequência infinita em duas mãos de quadrados de igual tamanho, que chamaremos de *células*. Veja fig. 1.

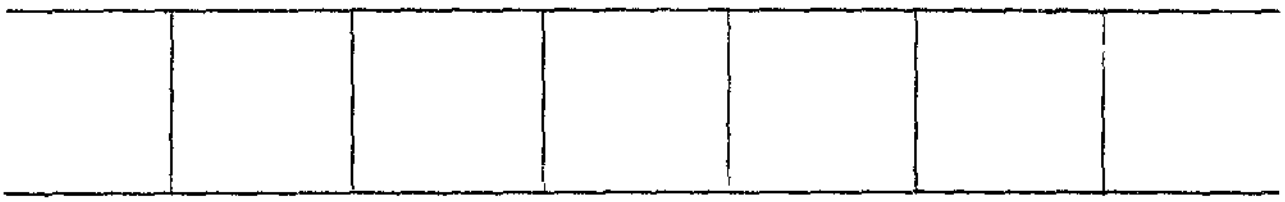


fig. 1

Assumiremos que a máquina é capaz de somente um número finito de estados internos distintos ou configurações e que sua próxima operação imediata, num dado momento, é determinada por sua configuração interna naquele momento, tomada em conjugação com a expressão finita da célula que então aparece no tape. A função da configuração interna é especificar o próximo ato da máquina, dado o reconhecimento do símbolo que aparece na célula explorada.

Os símbolos q_i, i=1,...,n denotarão configurações internas. Os símbolos S_i, i=0,1,...,n denotarão os símbolos que a máquina será capaz de imprimir e será chamado o seu alfabeto. Os símbolos R e L representarão um movimento de uma célula à direita e de uma célula à esquerda, respectivamente.

DEFINIÇÃO 1.3.1 Uma *Expressão* é uma sequência finita(possivelmente vazia) de símbolos escolhidos da lista:

$q_1, q_2, \ldots.; S_0, S_1, \ldots.; R, L$

DEFINIÇÃO 1.3.2 Uma *Quádrupla* é uma expressão tendo uma das seguintes formas:

1) $q_i S_j S_k q_\ell$
2) $q_i S_j R\, q_\ell$
3) $q_i S_j L\, q_\ell$

Uma quádrupla da forma 1), 2) ou 3) especifica o prómo ato da M.Tu. quando numa configuração interna q_i e explorando uma célula na qual aparece o símbolo S_j. Assim, a quádrupla 1) indica que o próximo ato é substituir S_j por S_k na célula explorada e entrar na configuração interna q_ℓ. As quádruplas do tipo 2) e 3) indicam movimento de uma célula para a direita e para a esquerda, respectivamente, entrando na configuração interna q_ℓ.

DEFINIÇÃO 1.3.3 Uma *Máquina de Turing* é um conjunto finito não vazio de quádruplas, não contendo duas quádruplas cujos dois primeiros símbolos sejam os mesmos.

Observemos que a última restrição da definição 1.3.3 garante que nenhuma M.Tu. será confrontada com duas diferentes instruções ao mesmo tempo. Esta propriedade é chamada *Restrição-Consistência*.

Os símbolos S_0 e S_1 terão um papel especial. S_0 será também denotado por B e S_1 por 1. B representa algo vazio. Assim, substituir um símbolo por B significa apagar esse símbolo.

Uma expressão simples e completa num presente estado da M. Tu. é:

1) A expressão no tape
2) A configuração interna
3) O quadrado explorado

DEFINIÇÃO 1.3.4 Uma *Descrição instantânea* é uma expressão que contém exatamente um q_i, não contendo nem R e nem L e é tal que q_i não é o símbolo mais à direita.

Se Z é uma M.Tu. e α é uma descrição instantânea, então dizemos que α é uma descrição instantânea de Z se os q_i que ocorrem em α são uma configuração interna de Z e se os S_i's que ocorrem em α são partes do alfabeto de Z.

DEFINIÇÃO 1.3.5 Uma expressão que consiste inteiramente de letras S_i é chamada uma *Expressão Tape*. No que segue P e Q designam expressões Tape.

Agora, visualizaremos cada descrição instantânea de uma M.Tu. como determinando, quando muito, uma descrição instantânea imediatamente subseqüente desta mesma M.Tu.. Desta maneira as descrições instantâneas de uma M.Tu. podem ser consideradas como um substituto abstrato para momentos sucessivos de tempo.

DEFINIÇÃO 1.3.6 Seja Z uma M.Tu. e seja α uma descrição instantânea de Z, onde q_i é a configuração interna que

ocorre em α, e onde S_j é o símbolo imediato à direita de q_i. Então chamamos q_i a *configuração interna de* Z em α e chamamos S_j o *símbolo explorado por* Z em α.

DEFINIÇÃO 1.3.7 A expressão tape obtida ao remover q_i de α é *chamada a expressão sobre a fita de* Z em α.

Partimos considerando nossa máquina como possuindo um tape infinito e no entanto a descrição instantânea α é sempre finita. Além disso, temos exigido que exista sempre um símbolo examinado individualmente em qualquer descrição instantânea. Esta aparente contradição é solucionada através do símbolo especial S_0 ou B. Consideraremos então uma máquina com um tape, que é sempre finito, mas que pode ser estendido. A máquina terá então a propriedade de que, ao chegar ao fim do tape, uma nova célula, na qual aparece um B, é unida sobre aquele fim do tape.

A seguinte definição dá uma maneira precisa pela qual uma descrição instantânea de uma M.Tu. é substituída por uma descrição instantânea sucessora.

DEFINIÇÃO 1.3.8 Seja Z uma M.Tu. e α, β descrições instantâneas. Então escrevemos $\alpha \longrightarrow \beta$ (Z) ou simplesmente $\alpha \longrightarrow \beta$, para significar que uma das seguintes alternativas acontece :

1) Existem expressões P e Q (possivelmente vazias)

tais que α é Pq_iS_jQ e β é $Pq_l S_kQ$, onde Z contém $q_iS_jS_kq_l$.

2) Existem expressões P e Q (possivelmente vazias) tais que α é $Pq_iS_jS_kQ$ e β é $PS_jq_l S_kQ$, onde Z contém $q_iS_jRq_l$.

3) Existe uma expressão P (possivelmente vazia) tal qte α é Pq_iS_j e β é $PS_jq_l S_0$, onde Z contém $q_iS_jrq_l$.

4) Existem expressões P e Q (possivelmente vazias) tais que α é $PS_kq_iS_jQ$ e β é $Pq_l S_kS_jQ$, onde Z contém $q_iS_jLq_l$.

5) Existe uma expressão Q (possivelmente vazia) tal que α é q_iS_jQ e β é $q_l S_0S_jQ$, onde Z contém $q_iS_jLq_l$.

Como consequência imediata da definição temos

TEOREMA 1.3.1 Se $\alpha \longrightarrow \beta(Z)$ e $\alpha \longrightarrow \gamma(Z)$, então $\beta = \gamma$.

TEOREMA 1.3.2 Se $\alpha \longrightarrow \beta(Z)$ e $Z \subseteq Z'$, então $\alpha \longrightarrow \beta(Z')$.

DEFINIÇÃO 1.3.9 Uma descrição instantânea α é chamada *Terminal* com respeito a Z se para nenhum β tivermos $\alpha \longrightarrow \beta(Z)$.

DEFINIÇÃO 1.3.10 Por uma *computação* de uma M.Tu. Z queremos dizer uma sequência finita $\alpha_1, \alpha_2, \ldots, \alpha_p$ de descrições instatâneas tais que $\alpha_i \longrightarrow \alpha_{i+1}(Z)$ para $1 \leq i \leq p$ e tal que α_p é terminal com respeito a Z.

Neste caso, escrevemos $\alpha_p = Res_Z(\alpha_1)$ e chamamos α_p o resultado de α_1 com respeito a Z.

EXEMPLO 6 *Seja Z constituída pelas seguintes quádruplas:*

$q_1 S_0 R\ q_1$ $\qquad$ $q_1 S_3 S_5 q_2$ $\qquad$ $q_3 S_2 S_5 q_3$

$q_1 S_2 R\ q_1$ $\qquad$ $q_2 S_5 L\ q_3$ $\qquad$ $q_3 S_3 S_5 q_3$

$q_1 S_5 R\ q_1$ $\qquad$ $q_3 S_0 S_5 q_3$

Z tem a configuração interna q_1, q_2, q_3 e o alfabeto S_0, S_2, S_3, S_5.

O que segue são computações em Z, onde $\alpha_1 = S_2 q_1 S_0 S_5 S_3$

1) $S_2 q_1 S_0 S_5 S_3 \longrightarrow S_2 S_0 q_1 S_5 S_3$

$\longrightarrow S_2 S_0 S_5 q_1 S_3$

$\longrightarrow S_2 S_0 S_5 q_2 S_5$

$\longrightarrow S_2 S_0 q_3 S_5 S_5$.

Portanto, $Res_Z(S_2 q_1 S_0 S_5 S_3) = S_2 S_0 q_3 S_5 S_5$.

2) $q_1 S_3 \longrightarrow q_2 S_5$

$\longrightarrow q_3 S_0 S_5$

$\longrightarrow q_3 S_5 S_5$. Portanto $Res_Z(q_1 S_3) = q_3 S_5 S_5$.

Se $\alpha_1 = S_2 q_1 S_0 S_5 S_2$, então

$S_2 q_1 S_0 S_5 S_2 \longrightarrow S_2 S_0 q_1 S_5 S_2$

$\longrightarrow S_2 S_0 S_5 q_1 S_2$

$\longrightarrow S_2 S_0 S_5 S_2 q_1 S_0$

$\longrightarrow S_2 S_0 S_5 S_2 S_0 q_1 S_0$

$\longrightarrow \ldots\ldots\ldots$

Assim não se chega a uma descrição instantânea terminal.

Daí, $Res_Z(\alpha_1)$ é indefinido.

§ 1.4 FUNÇÕES COMPUTÁVEIS E FUNÇÕES PARCIALMENTE COMPUTÁVEIS

Na secção precedente vimos como as M. Tu. poderiam ser usadas para executar computações simbólicas. A fim de ter M.Tu.para realizar computações numéricas é necessário que introduzamos uma representação simbólica adequada para números. Para isto, escolhemos o símbolo S_1 (escrito 1) como básico. Se n é um inteiro positivo, escrevemos S_i^n para a expressão $S_iS_i\ldots S_i$, n-vezes que consiste de n ocorrências de S_i e tomamos S_i^0 para ser a expressão nula. Então podemos escrever:

DEFINIÇÃO 1.4.1 A cada número n associamos a expressão tape $\overline{n}$ onde $\overline{n} = 1^{n+1}$.

Assim $\overline{3}$ = 1111 e, em geral $\overline{n}$ = 111....1, n+1 vezes

DEFINIÇÃO 1.4.2 A cada k-upla $(n_1,n_2,\ldots,n_k)$ de inteiros associamos a expressão tape $\overline{(n_1,n_2,\ldots,n_k)}$ onde $\overline{(n_1,n_2,\ldots,n_k)} = \overline{n}_1B\overline{n}_2B\ldots.B\overline{n}_k$

Assim, $\overline{(2,3,0)}$ = 11B1111B1 . Esta notação é usada para input, dados de entrada. Para dados de saída, outputs, usamos:

DEFINIÇÃO 1.4.3 Seja M alguma expressão. Então <M> é o número de ocorrências de 1 em M.

Assim, $<11BS_4q_3>$ = 2; $< q_3q_2S_5>$ = 0. Note que $<m-1>$ = m e que $< PQ > =< P> + < Q >$.

DEFINIÇÃO 1.4.4 Seja Z uma M.Tu.. Então, para cada n, associamos a Z uma função n-ésima

$$\Psi_Z^{(n)}(x_1, x_2, \ldots, x_n)$$ como sgue:

Para cada n-upla $(m_1, m_2, \ldots, m_n)$ colocamos $\alpha_1 = q_1^{(m_1, m_2, \ldots, m_n)}$ e distinguimos dois casos

1) Existe uma computação de Z, $\alpha_1, \ldots, \alpha_p$. Neste caso colocamos

$$\Psi_Z^{(n)}(m_1, m_2, \ldots, m_n) = <\alpha_p> = <\mathrm{Res}_Z(\alpha_1)>$$

2) Não existe uma computação de Z, $\alpha_1, \ldots, \alpha_p$, isto é, $\mathrm{Res}_Z(\alpha_1)$ é indefinido. Neste caso deixamos

$$\Psi_Z^{(n)}(m_1, m_2, \ldots, m_n) \text{ indefinido.}$$

DEFINIÇÃO 1.4.5 Uma função n-ésima $f(x_1, \ldots, x_n)$ é *Parcialmente Computável* se existe uma M.Tu. Z tal que

$$f(x_1, \ldots, x_n) = \Psi_Z^{(n)}(x_1, \ldots, x_n).$$

Neste caso dizemos que Z computa f. Se, na adição, $f(x_1, \ldots, x_n)$ é uma função total, então ela é chamada *Computável*.

EXEMPLO 7 *As seguintes funções são computáveis e parcialmente computáveis*

a) A D I Ç Ã O

Seja $f(x, y) = x + y$. Construamos uma M.Tu.Zque computa $f(x,y)$, istoé, tal $\Psi_Z^{(2)}(x, y) = x + y$

Tomemos Z constituída das quádruplas

$q_1 1 B q_1$ $\quad$ $q_2 1 R q_2$ $\quad$ $q_3 1 B q_3$

$q_1 B R q_2$ $\quad$ $q_2 B R q_3$

Seja $\alpha_1 = q_1(\overline{m_1, m_2}) = q_1 \bar{m}_1 B \bar{m}_2$. Então

$= q_1 1 1^{m_1} B 1 1^{m_2}$

$\longrightarrow q_1 B 1^{m_1} B 1 1^{m_2}$

$\longrightarrow B q_2 1^{m_1} B 1 1^{m_2}$

$\longrightarrow \ldots\ldots\ldots\ldots\ldots$

$\longrightarrow B 1^{m_1} q_2 B 1 1^{m_2}$

$\longrightarrow B 1^{m_1} B q_3 1 1^{m_2}$

$\longrightarrow B 1^{m_1} B q_3 B 1^{m_2}$, que é terminal.

Assim, $\Psi_Z^{(2)}(m_1, m_2) = < Res_Z(\alpha_1) >$

$= < B 1^{m_1} B q_3 B 1^{m_2} >$

$= m_1 + m_2$.

b) FUNÇÃO IDENTIDADE

Seja $I(x) = x$. Mostremos que $I(x)$ é computável. Seja Z uma M.Tu. constituída da quádrupla $q_1 1 B q_1$.

Então $q_1 \bar{n} = q_1 1 1^n \longrightarrow q_1 B 1^n$, que é terminal. Portanto, $\Psi_Z^{(n)} = < q_1 B 1^n > = n$.

c) FUNÇÃO SUCESSOR

Seja $S(x) = x + 1$. Mostremos que $S(x)$ é computável. De fato, seja Z alguma M.Tu. a respeito da qual $q_1 \bar{m}$ é terminal para algum m. Assim, Z consiste da quádrupla $q_1 B B q_1$.

Então $\Psi_Z(m) = < q\ \overline{m}> = m + 1$.

d) SUBTRAÇÃO

Seja $f(x, y) = x - y$. Esta função está definida somente para $x \geq y$. Em Davis (3) p. 12-15 temos uma prova de que $f(x,y)$ é uma função parcialmente computável, isto é, está construída uma M.Tu. Z tal que $\Psi_Z^{(2)}(x, y) = x - y$.

A princípio, a classe de funções parciais algorítmicas dão para as M.Tu. poder que antes parecia limitado. Não obstante, a definição abastece-se com P-simbolismos e L-P especificações. Cada conjunto de quádruplas definindo uma M.Tu. pode ser visto como um conjunto de P instruções. Podemos tomar o agente computante L como humano. As L-Pespecificações são regras simples conforme uma sucessão de configurações máquina-tape que é determinada para o tape inicial e para P.

§ 1.5 RESULTADO BÁSICO

Uma função parcialmente computável pode ser pensada como uma função para a qual possuímos um algoritmo que nos capacita computar seu valor para elementos de seu domínio. Neste caso, quando a resposta é possível, o algoritmo a fornece. Se um elemento não pertence ao domínio da função, o algoritmo gastará um tempo infinito numa busca inútil por uma resposta.

Nosso propósito é identificar o conceito de função computável com o conceito intuitivo de função efetivamente calculável.

Historicamente este trabalho foi intensificado a partir do ano de 1936. A noção de Church de λ-definibilidade, veja(2); a noção de Herbrand-Gödel-Kleene de recursividade geral, veja (4); a noção de Turing de computabilidade, veja (12) e outras, que eram totalmente diferentes em sua formulação, foram provadas equivalentes, no sentido de que a classe das funções obtidas eram as mesmas em cada caso.

Portanto, as caracterizações de Church (2), Post (10), Kleene (5), Turing(12), Markow (7), embora variando amplamente na forma, cada uma, contudo, pode ser representada por uma certa escolha de P-simbolismos e L-P especificações.

Extensivo trabalho foi realizado para provar a equivalência dessas caracterizações. Resumiremos esse trabalho no seguinte *Resultado Básico*:

PARTE A "as propostas caracterizações de Turing, Kleene, Church, Post, Markow e certas outras, são equivalentes, isto é, as mesmas classes de funções parciais e de funções totais são obtidas em cada caso".

DEFINIÇÃO 1.5.1 As funções contidas na Parte A são chamadas *Funções Recursivas*.

As funções parciais dessa classe podem, naturalmente, ser

denominadas *funções Recursivas Parciais.*

PARTE B "Uma extensa variedade de funções particulares, cada uma parecendo ser algorítmica, foi estudada e cada uma foi demostrada ser uma função parcial recursiva".

Aqui uma variedade de princípios e técnicas úteis foram desenvolvidas para fazer essas demonstrações.

PARTE C "As provas para os resultados na Parte A tem estrutura comum. Em todo o exemplo, o fato de que uma classe de caracterizações formais de funções parciais está incluí em uma outra, demostrou-se existir um procedimento uniforme que completa e justifica o seguinte: dado algum conjunto de instruções P para a primeira caracterização , podemos achar um conjunto de instruções P para a segunda caracterização para a mesma função parcial".

Em Davis (3) encontramos um detalhado trabalho justificando nosso Resultado Básico. A definição de Turing foi especialmente útil como uma caracterização usual para que outras caracterizações possam ser reduzidas a esta.

§ 1.6 TESE DE CHURCH

A caracterização de Turing dá convenientes evidências

de que toda a função parcial na classe escolhida é computável por um procedimento que é, intuitivamente, "mecânico.

Com bases nessa evidência, muitos matemáticos têm aceito a exigência que a caracterização usual dá uma formalização satisfatória, ou "reconstrução racional" das noções informais. Esta exigência é muitas vezes referida como a *Tese de Church*.

Nosso trabalho baseia-se em condições matemáticas concretas a partir da classe de funções parciais formalmente caracterizadas em § 1.2. Essa caracterização é o sujeito matéria da teoria de funções Recursivas. Agora,para provas que se baseiam em métodos informais temos, em nosso favor, toda a evidência acumulada pela Tese de Church. Tais provas serão chamadas provas pela Tese de Church.

§ 1.7 NÚMEROS DE GÖDEL E TEOREMA s-m-n

Mostremos agora como a teoria das Ms.Tu. e das funções computáveis podem ser desenvolvidas em termos de funções primitivas recursivas e por meio de Ms.Tu.

Usaremos números naturais como um código ou cifra para expressar a teoria das Ms. Tu.. Os símbolos básicos usados em nossa descrição das Ms.Tu. são R, L

$S_0, S_1, S_2, \ldots\ldots$

$q_1, q_2, q_3, \ldots\ldots$

A cada um desses símbolos associamos um número impar ≥ 3, como

segue:

$$\begin{array}{cccccccccccccccccc} 3 & , & 5 & , & 7 & , & 9 & , & 11 & , & 13 & , & 15 & , & 17 & , & 19 & , & 21 & , \ldots\ldots\ldots \\ \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & & \uparrow & \\ R & , & L & , & S_0 & , & q_1 & , & S_1 & , & q_2 & , & S_2 & , & q_3 & , & S_3 & , & q_4 & , \ldots\ldots\ldots \end{array}$$

Assim, para cada i, S_i é associado com 4i + 7, e q_i é associado com 4i + 5. Portanto, para alguma expressão M existe agora associada uma sequência finita de inteiros impares $a_1, a_2, \ldots, a_n$. Por exemplo, para a quádrupla q_1 1 R q_2 está associada a sequência 9, 11, 3, 13. Para a descrição instantânea $q_1$1 1 1 1 está associada a sequência 9, 11, 11, 11, 11.

Vejamos como associar um número com uma tal sequência e portanto com cada expressão.

DEFINIÇÃO 1.7.1 Seja M uma expressão consistindo dos símbolos $\gamma_1, \gamma_2, \ldots, \gamma_n$. Seja $a_1, a_2, \ldots, a_n$ os correspondentes inteiros associados com esses símbolos. Então o *Número de Gödel* de M é o inteiro

$$x = \prod_{k=1}^{n} Px(k)^{a_k} .$$

Escrevemos gn(M) = x. Se M é a expressão vazia, escrevemos gn(M) = 1. Assim, $gn(q_1\ 1\ R\ q_2) = 2^9.3^{11}.5^3.7^{13}$.

Observemos que Px(k) é o k-ésimo primo na ordem de magnitude.

DEFINIÇÃO 1.7.2 Seja $M_1, \ldots, M_n$ uma sequência finita de expressões. Então, o número de Gödel desta sequên -

cia de expressões é o número $\prod_{k=1}^{n} Px(k)^{gn(M_k)}$

Assim, o número de Gödel da sequência $q_1\ 1\ B\ q_1$, $q_1 B\ R\ q_2$ é $2^{2^9 3^{11} 5^7 7^9} \cdot 3^{2^9 3^7 5^3 7^{13}}$, um número um tanto grande.

DEFINIÇÃO 1.7.3 Seja Z uma M.Tu..Sejam $M_1,\ldots,M_n$ alguns arranjos de quádruplas de Z. Então o número de Gödel da sequência $M_1, \ldots, M_n$ é chamado um número de Gödel da M.Tu. Z.

Temos adotado a caracterização de Turing como básica . Notamos em § 1.2 que um conjunto de instruções é um conjunto de quádruplas que definem uma M.Tu.. Vimos acima, como é possível listar todos os conjuntos de instruções. Este procedimento pode ser visto como um procedimento que associa a cada inteiro x o conjunto de instruções dentro do(x+1)-ésimo lugar na lista de todos os conjuntos de instruções.Assumimos agora que tenhamos selecionado um tal procedimento listado. Assim,

DEFINIÇÃO 1.7.4 P_x é o conjunto de instruções associadas com o inteiro x na lista fixa de todos os conjuntos de instruções. x é chamado o *índice ou número de Gödel de* P_x.

$\phi_x^{(k)}$ é a função parcial de k variáveis determinada por P_x. x é também chamado um índice ou número de Gödel para $\phi_x^{(k)}$.

Claramente o procedimento listado nos dá:

a) um algoritmo para atingir algum x para o correspondente P_x, e

b) um algoritmo para atingir algum conjunto constituído de quádru

plas P ao correspondente inteiro x tal que P é P_x.

Dos fatos já implícitos no Resultado Básico destaque - mos o teorema seguinte:

TEOREMA s-m-n Para todo $m,n \geq 1$, existe uma função recursiva s_n^m de m+1 variáveis tal que para todo $x,y_1,\ldots, y_n$

$$\lambda z_1 \ldots z_n (\phi_x^{(m+n)}(y_1,\ldots,y_n,z_1,\ldots,z_n)) = \phi_{s_n^m}^{(n)}(x,y_1,\ldots,y_n).$$

DEMONSTRAÇÃO Tomemos o caso m=n=1. A demonstração é análoga para os outros casos. Consideremos a família de todas as funções parciais de uma variável que são expressas como

$\lambda z(\phi_x^{(2)}(y,z))$ para variáveis x e y. Usando a caracterização usual para funções de duas variáveis, podemos ver isto como uma nova caracterização formal para uma classe de fun - ções parciais recursivas de uma variável.

Pela Parte C do Resultado Básico, existe um procedimen to efetivo para retornar de um conjunto de instruções para um con junto anterior. Portanto, pela tese de Church, pode haver uma função recursiva f de duas variáveis tal que

$\lambda z\ (\phi_x^{(2)}(y,\ z)\ ) = \phi f(x,y)$. Esta f é nossa desejada s_1^1. c.q.d.

Em Davis (3) e Kleene (5) encontramos uma prova formal de que as funções s_n^m são primitivas recursivas.

****

C A P Í T U L O II

CONJUNTOS RECURSIVAMENTE ENUMERÁVEIS - SISTEMAS AXIOMÁTICOS

O conceito de recursividade tem virtudes amplas e claras e defeitos tais como a insolubilidade. Observemos que por recursividade estamos entendendo a classe de funções parciais "efetivas", "computáveis", "efetivamente computáveis", "mecanicamente computáveis" ou "algorítmicas". De acordo com nossa teoria, os problemas solúveis e insolúveis são representados, por meio de códigos, como um conjunto de inteiros. O problema será solúvel se o correspondente conjunto de inteiros é recursivo.

Neste capítulo estudaremos propriedades recursivas de conjuntos de inteiros, em particular aquelas propriedades que tem a ver com a solubilidade e insolubilidade. A mais básica é a propriedade de possuir uma função característica recursiva. Um problema será insolúvel se um certo conjunto ou relação falha na obtenção de uma função característica recursiva. Daremos também o Teorema Fundamental dos sistemas axiomáticos em nossa teoria

desenvolvida.

§ 2.1 CONJUNTOS RECURSIVAMENTE ENUMERÁVEIS

DEFINIÇÃO 2.1.1 Um conjunto é *Recursivo* se ele possui uma função característica recursiva.

Isto quer dizer, A é recursivo se e somente se existe uma função recursiva f tal que $\forall x$, $x \in A \Longrightarrow f(x) = 1$ e $x \in \bar{A} \Longrightarrow f(x) = 0$.

Intuitivamente, A é recursivo se existe um procedimento efetivo para decidir, dado algum x, se ou não $x \in A$.

EXEMPLO 8 *Os seguintes conjuntos são todos recursivos:*

a) O conjunto $\{0,2,4,\ldots\}$ *de inteiros pares;*

b) N *e* $\emptyset$ *;*

c) Os conjuntos finitos;

d) Os conjuntos com complemento finito.

Os conjuntos c) e d) são recursivos desde que uma lista explícita do conjunto finito considerado possa ser usada para dar instruções para a função característica.

Notemos, pela aplicação trivial da Tese de Church, que A recursivo $\Longrightarrow \bar{A}$ recursivo.

DEFINIÇÃO 2.1.2 A é *Recursivamente Enumerável* se $A = \emptyset$ ou exis

te uma função f recursiva total tal que A = imagem de f.

Intuitivamente, isto quer dizer que existe um procedimento efetivo para listar os membros do conjunto.

Os teoremas seguintes nos mostram uma conexão entre recursividade e enumerabilidade recursiva.

TEOREMA 2.1.1 A recursivo ⟹ A recursivamente enumerável.

DEMONSTRAÇÃO Caso (i) A é ∅. Então A é recursivamente enumerável por definição.

Caso (ii) A é finito e $\neq \emptyset$

Seja $A = \{ n_0, n_1, \ldots, n_k \}$. Definimos f por

$$f(x) = \begin{cases} n_x, & \text{para } x \leq k \\ n_k, & \text{para } k < x \end{cases}$$

Caso (iii) A é infinito

Seja g a função característica. Definimos f por

$$f(0) = \mu y(g(y) \doteq 1)$$

$$f(x + 1) = \mu y(g(y) = 1 \text{ e } f(x) < y)$$

Nos casos(ii) e (iii), f é uma função recursiva pela Tese de Church, e A = imagem de f. c.q.d.

TEOREMA 2.1.2 A é recursivo ⟺ A e $\bar{A}$ são ambos recursivamente enumeráveis.

DEMONSTRAÇÃO (⟹) Desde que A é recursivo ⟹ $\bar{A}$ é recursivo,

isto é imediato pelo teorema 2.1.1.

(<==) Se ambos A e $\overline{A}$ são $\emptyset$, A é imediatamente recursivo. Se nem A e nem $\overline{A}$ são $\emptyset$, então A = imagem de f e $\overline{A}$ = imagem de g para certas funções recursivas f e g, que dão um procedimento efetivo para testar elementos em A. Isto é, para testar um dado x, examinemos, em turno, f(0), g(0), f(1), g(1), ... Se aparece x como um valor de f, então $x \in A$. Se x aparece como um valor de g, então $x \in \overline{A}$. Também, x pode aparecer como um valor de f ou de g, desde que $A \cup \overline{A} = N$. Este procedimento pode ser mais intuitivamente descrito como segue: geramos listas para A e $\overline{A}$ simultaneamente. Ao mesmo tempo levamos a cabo um ziguezague para perseguir x sob as duas listas. x pode eventualmente aparecer. A lista em que x aparece determina se x está em A ou $\overline{A}$. c.q.d.

Veremos em § 2.2 que a recíproca do teorema 2.1.1 não é verdadeira e portanto, pelo teorema 2.1.2, que existem conjuntos recursivamente enumeráveis cujos complementos não são recursivamente enumeráveis.

§ 2.2 TEOREMA BÁSICO

O seguinte teorema dá uma caracterização alternativa

de enumerabilidade recursiva. É um resultado fundamental.

TEOREMA BÁSICO A é recursivamente enumerável <==> A é o domínio de uma função recursiva (isto é, $(\exists x)$ (A = domínio de ϕ_x)).

DEMONSTRAÇÃO (==>) *Caso (i)* A = Ø.

Seja Ψ a função parcial recursiva totalmente divergente. Então A = domínio de Ψ.

Caso (ii) A ≠ Ø. Então A = imagem de f.

Definimos Ψ pelas instruções: computar $\Psi(x)$, gerar a imagem de f ; se, e quando x aparece na imagem de f, dar o output x. Ψ é claramente recursiva parcial, e A = domínio de Ψ .

(<==) Seja A = domínio de Ψ, quando Ψ é parcial recursiva. Definimos um procedimento efetivo que listará A se A é não vazio. O procedimento consiste em realizar os seguintes estágios:

Estágio 1 Computar um passo na computação de $\Psi(0)$. Se $\Psi(0)$ converge em um passo, coloque 0 na lista para A.

.....

Estágio n+1 Computar n+1 passos em cada uma das n+1 computações para $\Psi(0)$, $\Psi(1)$,...., $\Psi(n)$. Para cada um desses passos que convergem sobre ou antes de (n+1)-ésimo passos, adicionamos o input à lista para A.

.....

Agora definimos η como segue:

$\eta(0)$ = primeiro membro da lista;

$$\eta(x+1) = \begin{cases} \mu y \text{ (y foi adicionado à lista no estágio x+1 e} \\ \quad y \notin \{ \eta(0), \eta(1), \ldots, \eta(x) \}), \text{ se um tal y existe;} \\ \eta(0) \ , \text{ em outros casos.} \end{cases}$$

Pela tese de Church, η é uma função parcial recursiva.

Se $A = \emptyset$, A é recursivamente enumerável por definição.

Se $A \neq \emptyset$ então, pela construção, η é uma função total e imagem de η = domínio de Ψ = A. Assim, A é recursivamente enumerável. c.q.d.

A demonstração acima procede por"ajustar completamente" as computações para $\Psi(0)$, $\Psi(1)$, ...

Introduzimos agora, como "nomes" para conjuntos recursivamente enumeráveis, os índices das correspondentes funções parciais recursivas.

DEFINIÇÃO 2.2.1 W_x = *domínio de* ϕ_x.x *é chamado um número de Gödel ou índice recursivamente enumerável* para o conjunto recursivamente enumerável.

Os métodos do teorema Básico produzem os seguintes corolários

COROLÁRIO 2.2.1 A é recursivamente enumerável<==> A é a imagem de uma função parcial recursiva. Isto é,
$(\exists x)$ (A = imagem de ϕ_x))

DEMONSTRAÇÃO (==>) *Caso (i)*.Como no teorema.

Caso (ii) A Ψ construída na prova do teorema servirá aqui, para A = domínio de Ψ = imagem de Ψ.

($\Longleftarrow$) Como no teorema, exceto que, nos estágios sucessivos, os outputs são adicionados antes que os inputs na lista que seria feita. c.q.d

Deste corolário segue que A é o domínio de uma função parcial recursiva $\Longleftrightarrow$ A é a imagem de uma função parcial recursiva. O seguinte corolário expressa este fato numa forma melhor.

COROLÁRIO 2.2.2 Existem funções recursivas f e g tais que $\forall x$

imagem de $\phi f(x)$ = domínio de ϕ_x,

e

domínio de $\phi g(x)$ = imagem de ϕ_x .

DEMONSTRAÇÃO *Para* f. Dado x, definimos

$$\Psi(y) = \begin{cases} y , & \text{se } \phi_x(y) \text{ converge;} \\ \text{divergente} , & \text{em outros casos.} \end{cases}$$

Claramente, imagem de ϕ = domínio de ϕ_x. Desde que as instruções para Ψ dependam de um caminho uniforme efetivo em x $\Psi(x) = \phi f(x)$ para alguma f recursiva. (Implicitamente usamos o teorema s-m-n).

Para g. Dado x, definimos um procedimento listado semelhante ao da demonstração do teorema; exceto que os outputs são listados antes que os inputs, ϕ_x é usada no lugar de Ψ.

Definimos

$$\theta(y) = \begin{cases} y, & \text{se } y \text{ aparece na lista;} \\ \text{divergente}, & \text{em outros casos.} \end{cases}$$

Claramente, domínio de θ = imagem de ϕ_x. Desde que as instruções para θ dependam de um caminho uniforme efetivo em x, $\theta = \phi g(x)$ para alguma g recursiva. c.q.d.

Um terceiro corolário afirma que para conjuntos não vazios, alguém pode ir, por um caminho uniforme efetivo, do domínio ou imagem, listando instruções a instruções de uma função total.

COROLÁRIO 2.2.3 Existem funções recursivas f' e g' tais que

(i) imagem de $\phi f'(x)$ = domínio de ϕ_x e
(domínio de $\phi_x \neq \emptyset \Longrightarrow \phi f'(x)$ é total)

(ii) imagem de $\phi g'(x)$ = imagem de ϕ_x e
(imagem de $\phi_x \neq \emptyset \Longrightarrow \phi g'(x)$ é total)

DEMONSTRAÇÃO *Para (i)* tomemos a η recursiva parcial construída no teorema básico, com ϕ_x no lugar de Ψ. Então, um índice para η depende efetivamente em x, isto é,

$\eta = \phi f'(x)$ para alguma f' recursiva.

Para (ii) coloque g' = f'g, onde g é como no corolário 2.2.2 c.q.d:

O teorema básico é uma poderosa ferramenta para mostrar a enumerabilidade recursiva. Por exemplo, mostremos que o conjunto

$\{ x \ / \ \phi_x(x)$ é convergente $\}$ é recursivamente enumerável.

Demos o nome K a tal conjunto, pois ele terá um papel importante em nosso trabalho.

DEFINIÇÃO 2.2.3 $K = \{ x \ / \ \phi_x(x)$ é convergente$\} = \{ x \ / \ x \in W_x \}$

TEOREMA 2.2.2 Existe um conjunto recursivamente enumerável mas não recursivo, e K é um tal conjunto.

DEMONSTRAÇÃO Definamos

$$\Psi(x) = \begin{cases} 1, & \text{se } \phi_x(x) \text{ é convergente} \\ \text{divergente}, & \text{se } \phi_x(x) \text{ é divergente} \end{cases}$$

Ψ é evidentemente recursiva parcial, e K = domínio de Ψ. Portanto, pelo teorema básico, K é recursivamente enumerável.

Para mostrar que K não é recursivo, assumamos que K é recursivo. Pelo teorema 2.1.2, $\overline{K} = W_m$ para algum m. Então

mas
$$m \in K \iff m \in W_m, \text{ pela definição de } K;$$
$$m \in \overline{K} \iff m \in W_m, \text{ pela escolha de } m.$$

Isto é uma contradição, e portanto K não pode ser recursivo. c.q.d.

Assim, K passa a ser recursivamente enumerável e $\overline{K}$ não recursivamente enumerável. O conjunto K encorpora e abrevia o problema "da parada"

Concluímos esta secção com uma aplicação ilustrativa adicional do teorema básico.

TEOREMA 2.2.3 Se Ψ é uma função recursiva parcial e A é recursivamente enumerável então, $\Psi^{-1}(A)$ é recursivamente enumerável.

DEMONSTRAÇÃO Para verificar resultados desta espécie, na prática, é melhor fiar-se na intuição e descrever procedimentos em termos desiguais. Por exemplo, neste caso: Liste os membros de A. Ao mesmo tempo liste, "ajustando completamente" computações para Ψ para todos os possíveis inputs . Encontre aqueles inputs que produzem outputs em A.

Para uma prova mais formal, contudo, o teorema básico pode ser útil. Neste caso, usemos o teorema básico para o conjunto A = domínio de ϕ_n para algum n. Então, $\Psi^{-1}(A)$ = domínio de $\phi_n\Psi$ e, outra vez pelo teorema básico, $\Psi^{-1}(A)$ é recursivamente enumerável.

c.q.d.

§ 2.3 SISTEMAS AXIOMÁTICOS

Um sistema axiomático consiste de:

a) Uma linguagem

b) Um conjunto decidível de axiomas

c) Um conjunto finito de regras elementares do tipo $\frac{A_1, A_2, \ldots, A_n}{B}$

EXEMPLO 9 *Um sitema axiomático da lógica de enunciados. A linguagem contém variáveis de enunciados e os conetivos* $\neg$ *(negação), e* $\Longrightarrow$ *(implicação).*

Os axiomas são:

1) $A \Longrightarrow (B \Longrightarrow A)$

2) $(A \Longrightarrow (B \Longrightarrow C)) \Longrightarrow ((A \Longrightarrow B) \Longrightarrow (A \Longrightarrow C))$

3) $(\neg A \Longrightarrow \neg B) \Longrightarrow (B \Longrightarrow A)$

A regra é:

$$\frac{A, \quad A \Longrightarrow B}{B}$$

Mediante uma gödelização podemos considerar o conjunto de teoremas de um sistema axiomático (isto é, o conjunto de fórmulas que se obtém usando os axiomas e as regras) como um conjunto de números naturais.

Assim sendo, o resultado fundamental da aplicação da teoria das funções recursivas aos sistemas axiomáticos é o seguinte:

TEOREMA FUNDAMENTAL Um conjunto A de fórmulas é axiomatizável se e somente se o conjunto dos números de Gödel das fórmulas em A é recursivamente enumerável.

DEMONSTRAÇÃO (<===) Se o conjunto de números de Gödel de fórmulas em A é recursivamente enumerável então, existe uma função recursiva f(x) cuja imagem é o dito conjunto . Seja a função recursiva f(x) definida por um sistema de equações.

Tomemos tais equações como axiomas e formulemos uma regra de substituição de iguais por iguais, de variáveis por numerais, e uma regra que permite passar de uma expressão f(x) = m à fórmula cujo número de Gödel é m. Ou seja

$$\frac{f(n) = m}{B}$$

(onde m = número de Gödel de B)

(===>) Um conjunto de fórmulas A é axiomatizável se existe um sistema de axiomas que permite deduzir esse conjunto A e somente A.

Assim sendo, podemos por em uma lista infinita as demonstrações de tal sistema axiomático. Bastará, para isto, escrever em ordem de complexidade crescente tais demonstrações, medindo sua complexidade por seu número de Gödel. Pela tese de Church sabemos que se esta listagem é efetiva, então existe uma função recursiva cuja imagem é o conjunto de números de Gödel da sequência demonstrável em uma listagem.

c.q.d.

****

C A P Í T U L O III

CONJUNTOS CRIATIVOS E PRODUTIVOS

No capítulo II, provamos que $\overline{K}$ *não é recursivamente enumerável. Neste capítulo damos um sentido construtivo ao fracasso do conjunto* $\overline{K}$ *em ser recursivamente enumerável. Com isto, definiremos a importante classe dos conjuntos criativos e produtivos.*

§ 3.1 CONJUNTOS CRIATIVOS

Para o índice de algum subconjunto recursivamente enumerável de $\overline{K}$ podemos encontrar um inteiro que está em $\overline{K}$, mas não no subconjunto.

Pondo isto mais formalmente temos, pela definição de $\overline{K}$, que

$$W_x \subseteq \overline{K} \Longrightarrow \quad x \in \overline{K} - W_x$$

De fato, suponhamos que $x \in W_x$ então, por definição de K, $x \in K$ mas, isto contradiz a hipótese de que $W_x \subseteq \overline{K}$. Portanto, $x \in \overline{W}_x$. Suponhamos que $x \in K$ então, por definição $x \in W_x$ mas então, $x \in \overline{K}$, contradição. Portanto, $x \in \overline{K}$.

Logo,

$$W_x \subseteq \overline{K} \Longrightarrow x \in \overline{K} - W_x.$$

Esta propriedade de $\overline{K}$ é dada em uma formulação recursiva invariante na definição seguinte.

DEFINIÇÃO 3.1.1 A é *Produtivo* se existe uma função Ψ recursiva parcial tal que

$$(\forall x)(W_x \subseteq A \Longrightarrow (\Psi(x) \text{ é convergente}, \Psi(x) \in A - W_x)).$$

Ψ é chamada uma *função produtiva parcial* para A.

Para nossos propósitos, estamos especialmente interessados em conjuntos recursivamente enumeráveis cujos complementos são produtivos. Tais conjuntos são chamados, por Post, criativos.

DEFINIÇÃO 3.1.2 A é *Criativo* se

a) A é recursivamente enumerável, e

b) $\overline{A}$ é produtivo.

EXEMPLO 10 *K é um conjunto criativo, desde que K seja recursivamente enumerável e $\overline{K}$ seja produtivo com a função identidade $\lambda x(x)$ sendo a função produtiva parcial.*

EXEMPLO 11 *O conjunto* $T = \{ x / \phi_x$ é total$\}$ *é produtivo.*

De fato, seja W_x dado. Tomemos $\phi_{f'(x)}$ como no corolário 2.2.3. Isto é, a imagem de $\phi_{f'(x)} = W_x$ e $\phi_{f'(x)}$ é total se W_x é não vazio.

Definimos D por

$$\phi_{g(x)}(z) = \begin{cases} \text{a) } 0, & \text{se } \phi_{f'(x)} \text{ com o input 0 não converge em z passos ou menos.} \\ \text{b) } \phi_{\phi_{f'(x)}(z-z_0)}(z) + 1, & \text{em outros casos, onde } z_o \text{ é o número exato de passos requeridos para } \phi_{f'(x)} \text{ com o input 0 para convergir.} \end{cases}$$

Demonstremos que se $W_x \subseteq T \Rightarrow$ existe g tal que $g(x) \in T - W_x$. Distinguimos dois casos:

i) $W_x = \emptyset$. Então, f'(x) é o índice de uma função recursiva que nunca converge. Portanto, aplicamos a) da definição D e por conseguinte, g(x) é o índice de uma função constante igual a 0. Isto é, $g(x) \in T$.

ii) $W_x \neq \emptyset$. Então, $\phi_{f'(x)}$ é total e $\phi_{f'(x)}(z-z_0) \in W_x \subseteq T$, por definição de f'(x) e pela hipótese de que $W_x \subseteq T$.

Pois bem, se $z \leq z_o$ então, $\phi_{g(x)}(z) = o$. Se $z > z_o$ então, $\phi_{f'(x)}(z-z_0)$ está definida e pelo resultado anterior é o índice

de uma função total, de tal modo que $\phi_{\phi_{f'(x)}(z-z_0)}(z)$ está definida e a fortiori $\phi_{\phi_{f'(x)}(z-z_0)}(z) + 1 = \phi_{g(x)}$ está definida.

Por conseguinte, g(x) é o índice de uma função total. Ou seja, $g(x) \in T$.

Mostremos agora que $g(x) \notin W_x$. Suponhamos que $g(x) \in W_x$. Como f'(x) é o índice de uma função que enumera W_x então, existe i tal que $g(x) = \phi_{f'(x)}(i)$. Tomemos $z = z_o + i$. Então,

$\phi_{g(x)}(z) = \phi_{\phi_{f'(x)}(i)}(z)$ e também por b) da definição D

$\phi_{g(x)}(z) = \phi_{\phi_{f'(x)}(z_0+i-z_0)}(z) + 1 = \phi_{\phi_{f'(x)}(i)}(z) + 1,$

contradição.

Portanto, $g(x) \notin W_x$ e g é a desejada função parcial produtiva.

c.q.d.

O seguinte teorema e seu respectivo Lema é uma consequência imediata da definição.

TEOREMA 3.1.1 A produtivo ⟹ A não recursivamente enumerável.

COROLÁRIO 3.1.1 A criativo ⟹ A não recursivo.

§ 3.2 PRODUTIVIDADE

No capítulo IV, trataremos das aplicações da teoria das funções recursivas à lógica. Para isto, o conceito de produtividade desempenha um papel fundamental.

Segue da definição de produtividade que se um conjunto A é produtivo então, existe um procedimento efetivo pelo qual, dado algum subconjunto recursivamente enumerável de A, podemos obter um maior subconjunto recursivamente enumerável de A.

No que segue, procuraremos caminhos de intersecção destes procedimentos.

TEOREMA 3.2.1 A produtivo ⟹ A tem um subconjunto infinito recursivamente enumerável.

DEMONSTRAÇÃO (Informal) . Seja Ψ uma função produtiva parcial para A. Obtemos uma g recursiva cuja imagem é um subconjunto infinito de A. As instruções para g são indutivas. Seja z_0 um índice recursivamente enumerável para o conjunto não vazio. Compute g(0). Coloque $g(0) = \Psi(z_0)$. Desde que $W_{z_0} \subseteq A$, $\Psi(z_0)$ está definida e está em A. Compute g(n+1). Seja z_{n+1} um índice recursivamente enumerável para o conjunto finito

$\{ g(0), \ldots\ldots, g(n) \}$. Coloque $g(n+1) = \Psi(z_{n+1})$. Desde que $W_{z_n} \subseteq A$, $\Psi(z_{n+1})$ está definida e está em A. Evidentemente g é uma função um-um.

c.q.d.

O seguinte teorema responde à pergunta: Dado um conjunto produtivo, pode ser encontrada uma função produtiva total para ele ?

TEOREMA 3.2.2 A é produtivo <=> existe uma função recursiva (total) f tal que A é produtivo, com f sendo uma função produtiva parcial.

Para uma demonstração, veja Rogers (11) p.92.

DEFINIÇÃO 3.2.1 Quando uma função produtiva é total, ela é chamada uma *função produtiva*.

Originariamente, Post definiu um conjunto criativo para ser um conjunto recursivamente enumerável cujo complemento é produtivo com uma função (total) produtiva.

A asserção de que A não é recursivamente enumerável pode ser expressa por

$$(\forall x)(\exists y)\ (y \in W_x \rightarrow A \vee y \in A - W_x)$$

Diz-se que esta asserção vale "construtivamente ou "informalmente" se existe uma função recursiva f tal que

$$(\forall x)\ (f(x) \in W_x - A \vee f(x) \in A - W_x)$$

Assim sendo,

DEFINIÇÃO 3.2.2 A é *completamente produtivo* se existe uma função recursiva f tal que para toto x, ou

$$f(x) \in W_x - A \text{ ou } f(x) \in A - W_x.$$

f é chamada uma *função completa* para A.

EXEMPLO 12 $\overline{K}$ *é completamente produtivo com* $\lambda x(x)$ *como função produtiva completa. Para* $x \in W_x \Longrightarrow x \notin \overline{K}$ *e* $x \notin W_x \Longrightarrow x \in \overline{K}$, *pela definição de* $\overline{K}$.

A definição seguinte afirma que existe um procedimento produtivo para ir de subconjuntos recursivamente enumeráveis para maiores subconjuntos recursivamente enumeráveis.

DEFINIÇÃO 3.2.3 A é *semiprodutivo* se existe uma função parcial recursiva tal que

$$(\forall x)(W_x \subset A \Longrightarrow (\Psi(x) \text{ é convergente} \wedge W_x \subset W_{\Psi(x)} \wedge W_x \neq W_{\Psi(x)} \wedge W_{\Psi(x)} \subset A))$$

****

C A P Í T U L O IV

APLICAÇÕES DA TEORIA DAS FUNÇÕES RECURSIVAS À LÓGICA

Pela técnica de Gödel de designar números a símbolos e a séries finitas de símbolos, como vimos em § 1.7, todos os problemas que dizem respeito a sistemas lógicos matemáticos tornam-se problemas que dizem respeito a conjuntos de inteiros. Aqui está uma das grandes utilidades da teoria que desenvolvemos. Em particular, um conjunto de fórmulas constitue um sistema axiomático se o correspondente conjunto de números de Gödel é recursivamente enumerável. O sistema axiomático é decidível se esse conjunto é recursivo, indecidível se ele não é recursivo. A existência de sistemas axiomáticos indecidíveis é assim equivalente à existência de conjuntos recursivamente enumeráveis que não são recursivos.

Neste capítulo, após a introdução de alguma terminologia, aplicaremos a teoria das funções recursivas a um sistema lógi-

gico matemático específico, a Aritmética Elementar, apresentando antes a utilidade de tal teoria na demonstração do teorema da Imcompletude de Gödel.

§ 4.1 TERMINOLOGIA

Nesta secção introduzimos alguma terminologia para a aplicação da teoria das funções recursivas à lógica.

DEFINIÇÃO 4.1.1 Chamamos de *Fórmulas Bem Formadas* de um sistema lógico formalizado, que denotamos por fbfs, a uma certa classe de séries finitas de símbolos obtidos de um certo alfabeto básico.

As fbfs são especificadas neste sentido quando existe um procedimento efetivo para decidir que séries finitas são fbfs e que séries finitas não o são. Na investigação de um sistema lógico, as fbfs constitui a classe de objetos básicos de estudo.

EXEMPLO 13 *Definição de fbfs na lógica de enunciados.*

a) O alfabeto α constitui-se de $\rightarrow$, $\Longrightarrow$, (,); $p_0, p_1, p_2, \ldots\ldots$

b) Uma série finita de símbolos do alfabeto α é, por exemplo:

$$\rightarrow p_0)(\rightarrow$$

Definição de fbf:

1) $p_0, p_1, p_2, \ldots$ são fbfs,

2) Se A é fbf então, $(\rightarrow A)$ é fbf,

3) Se A e B são fbfs então, $(A \Longrightarrow B)$ é também fbf,

4) Cláusula extremal. Só são fbfs aquelas séries finitas de símbolos do alfabeto α cuja condição de ser fbf se pode demonstrar usando 1), 2), e 3).

Assim,

i) $(p_2 \Longrightarrow p_3)$ é fbf, pois p_2 e p_3 são fbfs pela cláusula 1) e $(p_2 \Longrightarrow p_3)$ é fbf por 3).

ii) $(\Longrightarrow p_2, p_3)$ não é fbf pois, para isto teríamos que aplicar 2) ou 3). Mas, a cláusula 2) não se aplica pois, o símbolo $\rightarrow$ não aparece e 3) também não pois, isto exigiria que o símbolo $\Longrightarrow$ fosse fbf porém, não é o caso.

Se uma fbf não tem variáveis livres, dizemos que é uma *sentença*. Na interpretação intuitiva, as sentenças correspondem às proposições verdadeiras ou falsas.

Para aplicar os conceitos da teoria das Funções Recursivas pode-se usar um código para as fbfs. Limitaremos nossa discusão a códigos que são aplicações sobre N. Assim,

DEFINIÇÃO 4.1.2 O inteiro que está associado com uma fbf sob um código é chamado o *número de Gödel* dessa fbf .

Assumimos como usual que as operações de codificar e decodificar são efetivas.

Seja um sistema lógico dado. Então, um código associa com cada conjunto de fbfs um conjunto de inteiros. Por outro lado, seja um conjunto de fbfs dado. Então, alguma propriedade recursiva invariante que vale para o conjunto de números de Gödel obtidos sob um código valerá também para o conjunto de números de Gödel obtidos sob algum outro código. Portanto, propriedades recursivas invariantes podem ser diretamente associadas com conjuntos de fbfs. Podemos falar, por exemplo, de um conjunto recursivo de fbfs, ou de um conjunto produtivo de fbfs.

Em um sistema lógico particular, por exemplo, um conjunto de fbfs pode ser distinguido como "demonstrável" sob certas regras específicas de prova, ou outro conjunto de fbfs pode ser distinguido como "verdadeiro" sob alguma definição de verdade, usulmente não construtiva.

DEFINIÇÃO 4.1.3 Um conjunto de fbfs assim distinguido é muitas vezes chamado uma *Teoria*.

Como demonstramos em § 2.3, as fbfs de uma teoria podem ser, muitas vezes, listadas. Este é o caso quando a teoria consiste das fbfs "demonstráveis" sob certas regras formais de prova. Assim,

DEFINIÇÃO 4.1.4 Uma teoria, isto é, um conjunto de fbfs, é *Axiomatizável* se ela pode ser efetivamente listada, ou melhor dito, se ela é recursivamente enumerável.

Similarmente,

DEFINIÇÃO 4.1.5 Uma teoria é dita ser *decidível* se ela é recursiva.

A existência de conjuntos recursivamente enumeráveis, mas não recursivos, semelhantes a K, sugerem a possibilidade que certas teorias axiomatizáveis bem conhecidas não podem ser decidíveis. O teorema de Church, veja (1) a), mostra que as fbfs demonstráveis da lógica quantificacional formam um conjunto de fbfs indecidível.

Veremos em § 4.3 que as fbfs demonstráveis da Aritmética Elementar, sob algumas das várias axiomatizações usuais também formam um conjunto de fbfs indecidível.

§ 4.2 O TEOREMA DA INCOMPLETUDE DE GÖDEL

Vários são os caminhos para uma formalização do "fenômeno da incompletude de Gödel, veja (4) a). Talvez o mais simples é dizer:

> *Todo o sistema formal que tem uma certa complexidade mínimal e para o qual a noção de fbfs verdadeiras pode ser definido em um certo caminho natural, o conjunto de fbfs verdadeiras é produtivo e portanto não recursivamente enumerável.*

Assim, pela definição 4.1.4 as fbfs verdadeiras não formam uma teoria axiomatizável.

Pela teoria desenvolvida neste trabalho podemos ver, informalmente, que isto deve ser assim. Consideremos algum sistema matemático formal flexível e bastante inclusivo para criar asserções a respeito das Máquinas de Turing e de seus índices.

Afirmações tais como "21 é o índice para uma função recursiva total" serão então expressíveis dentro da teoria.

Consideremos agora afirmações da forma "n é o índice para uma função recursiva total", onde n é o numeral para um inteiro. Neste caso, não podemos esperar por um procedimento que listará todas as afirmações verdadeiras e nenhuma das afirmações falsas desta forma pois, $\{ x / \phi_x$ é total $\}$ é, como conhecemos pelo exemplo 11, um conjunto produtivo.

De fato, das instruções para algum procedimento onde listamos somente afirmações verdadeiras deta forma, podemos efetivamente obter, pela produtividade de $\{ x / \phi_x$ é total $\}$, uma nova afirmação verdadeira desta forma que não está listada.

Similarmente, consideremos afirmações da forma "P_n com o input n diverge", isto é, da forma "$n \notin K$". Aqui também não podemos esperar por um procedimento onde possamos listar todas as afirmações verdadeiras e nenhuma falsa desta última forma, desde que $\overline{K}$ é produtivo.

§ 4.3 ARITMÉTICA ELEMENTAR

Consideraremos agora um sistema lógico específico, a Aritmética Elementar.

As sentenças da Aritmética Elementar são as fbfs que podem ser construídas por:

a) símbolos variáveis para inteiros não negativos, +, x, =, 0,1,...
b) quantificadores sobre inteiros não negativos, $\forall$, $\exists$;
c) conetivos sentenciais $\neg$, $\wedge$, $\vee$, $\Longrightarrow$, $\Longleftrightarrow$;
d) toda a variável em uma sentença deve ser ligada por algum quantificador.

Por exemplo, $(\forall a)(\neg a = 2 \Longrightarrow (\exists b)\ a = b \times b)$, é uma sentença que cria a asserção falsa de que todo inteiro diferente de 2 é um quadrado.

Por um caminho direto e inteiramente de acordo com nossa intuição, é possível definir o conjunto de sentenças verdadeiras da Aritmética Elementar. Essa teoria, o conjunto de sentenças verdadeiras, é chamada *teoria elementar dos números*. Assim, podemos falar de sentenças verdadeiras e sentenças falsas. Assumimos que essa definição de verdade tenha sido dada. Inesperadamente , uma vasta variedade de afirmações combinatórias podem ser expressas dentro da Aritmética Elementar.

Em particular, podemos encontrar uma expressão F com uma só variável não quantificada tal que, quando substituimos o nu

meral x_0 em lugar da variável não quantificada, a sentença resultante, denotada por F_{x_0}, afirma intuitivamente, que $x_0 \in K$. Mais precisamente, estabelecemos

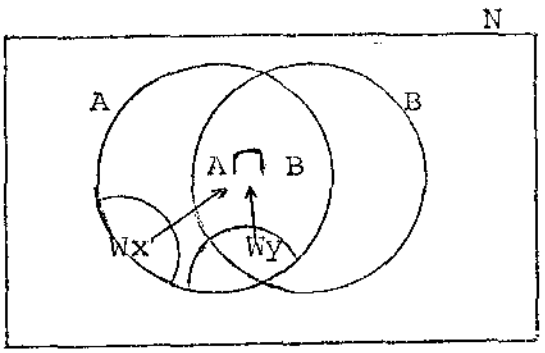
PROPRIEDADE 1 $\quad x \in K \iff (F_x$ é verdadeira), e

$x \notin K \iff (F_x$ é falsa) $\iff ((\neg F_x)$ é verdadeira)

O seguinte lema é básico em nossa discussão.

LEMA BÁSICO $\quad (\ B$ recursivo $\wedge\ A \cap B$ produtivo $) \Longrightarrow A$ produtivo

DEMONSTRAÇÃO INFORMAL



Seja W_x igual ao domínio de uma função parcial recursiva Ψ. Suponhamos $W_x \subset A$. Distinguimos dois casos:

i) $W_x \cap B = \emptyset \Longrightarrow W_x \subset A - B$. Pela produtividade de $A \cap B \Longrightarrow$ existe um objeto x tal que $\Psi(x)$ é convergente e $\Psi(x) \in A \cap B - W_x$

ii) $W_x \cap B \neq \emptyset$. Dado W_x existe y tal que $W_y = W_x \cap B$. Pela produtividade de $A \cap B$ existe uma funçao Ψ_y tal que $\Psi(y)$ é convergente e $\Psi(y) \in A - W_x$. Veja esquema acima.

DEMONSTRAÇÃO (formal) Seja B recursivo e $A \cap B$ produtivo com respeito a uma função parcial recursiva Ψ. Provemos que existe uma função parcial recursiva ϕ tal que se $W_x \subset A \Longrightarrow \phi(x)$ é convergente e $\phi(x) \in A - W_x$. Aqui W_x é igual ao domínio de uma função parcial recursiva cujo índice é x.

Como B é recursivo, $W_x \cap B$ é recursivamente enumerável e existe uma função recursiva g tal que $W_x \cap B = W_{g(x)}$, pela tese de Church. Além disso, $W_{g(x)} \subset A \cap B$. Por conseguinte, $\Psi g(x)$ é convergente e $\Psi g(x) \in A \cap B - W_{g(x)}$. Logo, $\Psi g(x) \in A$.

Suponhamos que $\Psi g(x) \in W_x$, como além disso $\Psi g(x) \in B$, então $\Psi g(x) \in W_x \cap B = W_{g(x)}$. Mas isto, contradiz o fato de que $\Psi g(x) \notin W_{g(x)}$. Por conseguinte, $\Psi g(x) \notin W_x$.

Logo definimos $\phi(x) = \Psi g(x)$; ϕ é uma função produtiva para A e portanto, A é produtivo.

c.q.d.

LEMA 4.3.2 Os conjuntos de sentenças:

a) $\{ F_x / x \in N \}$ é recursivo.

b) $\{ F_x / x \notin K \} = \{ F_x / F_x \text{ é falso} \} = \{ (\rightarrow F_x) / F_x \text{ é verdadeira} \}$ é produtivo.

DEMONSTRAÇÃO *Para a)*. De fato, dado um inteiro qualquer podemos decidir se é ou não o número de Gödel da sentença F_y para algum y inteiro. Pela Tese de Church o conjunto $\{ gn(F_x) / x \in N \}$ é recursivo.

Para b) De fato, como para cada $x \in N$ podemos calcular o número de Gödel de F_x, obtemos o conjunto

$$A = \{ gn(F_x) \ / \ F_x \text{ é falso } \} = \{ gn((\neg F_x)) \ / \ F_x \text{ é verdadeira } \}.$$

Seja $W_x \sqsubset A$, onde W_x é como na definição 2.2.1. Mostremos que existe uma função recursiva parcial $\Psi(x)$ tal que $\Psi(x) \in A - W_x$.

Pela propriedade 1, existe uma função recursiva parcial ϕ biunívoca, mas não sobre, assim definida $\phi(x) = gn(F_x)$, tal que $\phi(\overline{K}) = A$. Pela tese de Church existe uma função recursiva g tal que $\phi^{-1}(W_x) = W_{g(x)} \sqsubset \overline{K}$. Por hipótese, existe a função identidade I, que é produtiva para $\overline{K}$, tal que $Ig(x) \in \overline{K} - W_{g(x)}$. Logo, definimos $\Psi(x) = \phi Ig(x)$, que é a função produtiva para A. Portanto, A é produtivo.

c.q.d.

Assumindo o lema 4.3.2 podemos aplicar o lema Básico para obter o seguinte teorema.

TEOREMA 4.3.1 (a) As sentenças verdadeiras da Aritmética Elementar formam um conjunto produtivo e portanto, não recursivo.

(b) As sentenças falsas da Aritmética Elementar formam um conjunto produtivo.

DEMONSTRAÇÃO *Para a)* Sejam

A = conjunto de sentenças verdadeiras da Aritmética Elementar.

B = conjunto de sentenças da Aritmética Elementar do tipo $(\neg F_x)$, $x \in N$.

$A \cap B$ = conjunto de sentenças da Aritmética Elementar do tipo $(\neg F_x)$ que são verdadeiras.

Pelo lema 4.3.2, B é recursivo e $A \cap B$ é produtivo, então pelo Lema Básico segue que A é produtivo.

Para b) Mesmo raciocínio da demonstração para a).
c.q.d.

Assim, segue que o conjunto de sentenças verdadeiras e o conjunto de sentenças falsas da Aritmética Elementar constituem uma teoria indecidível e não axiomática. Isto é, o conjunto de sentenças verdadeiras e o conjunto de sentenças falsas da Aritmética Elementar não são nem axiomatizáveis, e nem decidíveis.

Existem vários caminhos usuais para especificar regras de prova na Aritmética Elementar. Um dos quais é baseado nos *Axiomas de Peano*, incluíndo as instâncias do axioma de indução que podem ser expressos na Aritmética Elementar. A teoria resultante, isto é, o conjunto de sentenças demonstráveis, é chamada *Aritmética de Peano*.

A Aritmética de Peano, consiste das sentenças da A - ritmética Elementar que são dedutíveis, a partir da lógica dos quantificadores e dos axiomas seguintes:

i) $(\forall a)(\forall b)\ (a + 1 = b + 1 \Longrightarrow a = b)$;

ii) $(\forall a)\ (\neg 0 = a + 1)$;

iii) $(\forall a)\ (a + 0 = a)$;

iv) $(\forall a)(\forall b)\ (a + (b + 1) = (a + b) + 1)$;

v) $(\forall a)\ (a \times 0 = 0)$;

vi) $(\forall a)(\forall b)\ (a \times (b + 1) = (a \times b) + 1)$;

vii) $0 + 1 = 1$;

viii) Todas as sentenças da forma $(.((1 + 1) + 1)....+ 1) = \chi$, onde χ é o numeral do inteiro x, e o x é o número de 1's que aparecem na esquerda;

xix) Todas as sentenças da forma

$$(\ ...0... \wedge (\forall a)\ (...a... \Longrightarrow\ ...(a + 1)...)) \Longrightarrow (\forall a)...a...$$

Este último axioma, chama-se axioma da indução completa.

A Aritmética de Peano possui a

PROPRIEDADE 2 $\quad x \in K \Longrightarrow (F_x$ é demonstrável)

Fazendo a *Suposição Especial* de que as sentenças da Aritmética Elementar falssas não são demonstráveis, temos

LEMA 4.3.3 $x \in K \iff (F_x$ é demonstrável), e portanto

$x \notin K \iff (F_x$ é não demonstrável)

DEMONSTRAÇÃO Temos a propriedade 2 e a suposição especial de que todas as sentenças demonstráveis são verdadeiras. Logo, se F_x é demonstrável e $x \notin K$ então, F_x é falsa pela propriedade 1. Mas então, F_x não é demonstrável pela suposição especial. Portanto, se F_x é demonstrável então, $x \in K$.

c.q.d.

Aplicando o lema básico temos o seguinte:

TEOREMA 4.3.1 (c) As sentenças não demonstráveis formam um conjunto produtivo.

(b) As sentenças demonstráveis formam um conjunto criativo e portanto não recursivo.

DEMONSTRAÇÃO *Para* c) . Sejam

A = conjunto das sentenças não demonstráveis

B = conjunto das sentenças do tipo F_x, $x \in N$.

$A \cap B$ = conjunto das sentenças do tipo F_x, $x \in N$ que não são demonstráveis.

Usando o lema 4.3.2, pois pelas propriedades 1 e 2 ($(\neg F_x)$ é verdadeira) $\iff$ (F_x é falso) $\iff x \notin K \iff$ (F_x não demons.), temos que B é recursivo e $A \cap B$ é produtivo e portanto, pelo

Lema Básico, segue que A é produtivo.

Para d)

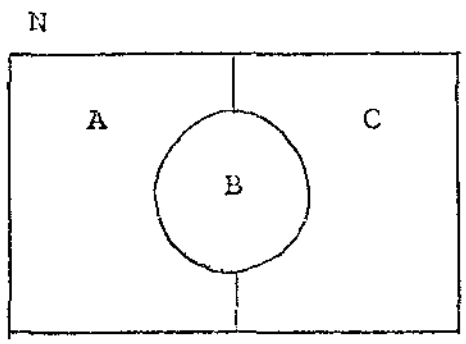
Seja A = conjunto das sentenças demonstráveis. A é recursivamente enumerável, pois existe um procedimento efetivo para listar todos os elementos de A.

Por outro lado $\overline{A} = B \sqcup C$, onde

B = Conjunto de números de Gödel das sentenças não demonstráveis e

C = conjunto de números que não são números de Gödel de sentenças.

Aplicamos o Lema Básico da seguinte maneira: $A \sqcup B$ é recursivo, pois é o conjunto de números de Gödel de sentenças da Aritmética de Peano. $(A \sqcup B) \cap \overline{A} = B$, que é produtivo pela parte c). Portanto, pelo lema básico $\overline{A}$ é produtivo. Por conseguinte A é criativo. c.q.d. Veja esquema abaixo.



A = conjunto de números de Gödel de sentenças demonstráveis.

B = comjunto de números de Gödel de sentenças não demonstráveis.

C = conjunto de números que não são números de Gödel de sentenças.

N = conjunto de números Naturais.

Consideremos o conjunto $\{ F_x / (\neg F_x)$ é demonstrável $\}$. Este conjunto é recursivamente enumerável pois, existe um procedimento efetivo para listar todos os membros desse conjunto, e além disso, pela suposição especial é um subconjunto das sentenças não demonstráveis do tipo F_x, $x \in N$.

Usando a produtividade de $\overline{K}$, temos

TEOREMA 4.3.1 (e) Existe um x_0 tal que nem F_{x_0}, nem $(\neg F_{x_0})$ é demonstrável.

DEMONSTRAÇÃO Dado $x \in N$, podemos clacular o número de Gödel da fórmula F_x e então,

$G = \{ gn(F_x) / (\neg F_x)$ é demonstrável $\}$ é recursivamente enumerável.

Pela suposição especial $G \sqsubset D$ = conjunto de sentenças não demonstráveis do tipo F_x, onde D é um conjunto produtivo pela parte c). Logo D possui uma função produtiva Ψ.

Suponhamos que $G = W_m$, e que $\Psi(m) \in D - G$. Mas então, $\Psi(m) \in D \Longrightarrow F_{\Psi(m)}$ não é demonstrável. Como $\Psi(m) \notin G \Rightarrow \neg F_{(m)}$ não é demonstrável. Logo, nem $F_{\Psi(m)}$, nem $\neg F_{\Psi(m)}$ são demonstráveis . Tomamdo $x_0 = \Psi(m)$ segue o resultado. Além disso,

$(\neg F_{x_0}$ é verdadeira) $\Longleftrightarrow$ (F_{x_0} é falsa) $\Longleftrightarrow$ $x_0 \notin K \Longleftrightarrow$

$\Longleftrightarrow$ (F_{x_0} não é demonstrável), donde segue imediatamente

COROLÁRIO 4.3.1 $\neg F_{x_0}$ é verdadeira e F_{x_0} é falsa

A *Suposição Especial* de que as sentenças falsas da Aritmética Elementar são demonstráveis é essencial. Para o caso da Aritmética de Peano, esta suposição pode, ela própria, ser demonstrada dentro da mais geral teoria Aritmética Conjuntista .

Resumindo informalmente, uma axiomatização da matemática não pode captar exatamente todas as afirmações verdadeiras da Aritmética Elementar, parte a) do teorema.

Para alguma axiomatização que produz somente sentenças verdadeiras da Aritmética Elementar, uma nova sentença verdadeira não demonstrável pode ser encontrada nessa axiomatização.

Esses fatos dão um significado especial ao estudo de produtividade. Post supôs que tais fatos manifestaram uma qualidade criativa essencial da matemática. Daí o nome conjuntos criativos.

****

N O T A Ç Õ E S

Usamos as seguintes notações convencionais para:

1) *Teoria dos conjuntos*

Se A é um conjunto, $\overline{A}$ é seu complemento.

A = B significa, que A e B são idênticos como conjuntos.

$x \in A$ e $x \notin A$ significa respectivamente, que x é ou não elemento de A.

{ / } é a notação para indicar formação de conjunto.

$A \cup B$ é a união dos conjuntos A e B.

$A \cap B$ é a intersecção dos conjuntos A e B.

A - B é a diferença " " A e B.

$A \subset B$ significa, que A é um subconjunto de B.

$N = \{0,1,2,....\}$. Neste trabalho a palavra número, inteiro são sinônimos de número natural.

2) *Funções*

ϕ, Ψ,.... denotam funções parciais. Usamos, muitas vezes, a notação funcional. Assim, $\phi(x,y)=z$ significa, que a 3-upla ordenada $\langle x,y,z\rangle \in \phi$.

Se ϕ é uma função parcial, dizemos que ϕ é *definida* ou *convergente* em x se, x é o domínio de ϕ, do contrário será *não definida* ou *divergente*.

f, g, h,.... denotam funções recursivas totais. f(x) = y signi-

fica que $\langle x,y\rangle \in f$. Uma função n-ésima cujo domínio é o conjunto de todas as n-uplas é chamada *Função total*.

3) *Notação lambda de Church para definição de funções parciais*

Seja $(-x-)$ uma expressão tal que dado qualquer inteiro no lugar de x, a expressão define quando muito um valor correspondente. Então, $\lambda x(-x-)$ denota a função parcial $x \longrightarrow (-x-)$. Por exemplo, $\lambda x(\phi_1(x) + \phi_2(x))$ é a função parcial Ψ tal que o domínio de Ψ = domínio de $\phi_1 \cap$ domínio de ϕ_2 e $\Psi(x) = \phi_1(x) + \phi_2(x)$, para todo o x no domínio de Ψ. Usamos também a notação lambda para funções de k variáveis escrevendo, $\lambda x_1 x_2 \ldots x_k$ em vez de λ_k.

$\Psi\phi$ indica a composição de Ψ e ϕ.

Ψ^{-1} indica a função inversa.

4) *Ordem parcial*

$<$ é a ordem parcial estrita.

$\leq$ é a ordem parcial não estrita.

5) *Convenções da lógica elementar*

$\wedge$ = e

$\vee$ = ou

$\Longrightarrow$ = se,...,então

$\Longleftrightarrow$ = se e somente se

$\neg$ = negação

$\forall, \exists$ = quantificadores universal e existencial, respectivamente.

6) $\mu x(\ldots x \ldots)$ é o menor inteiro x tal que ...x... é verdadeira.

****

B I B L I O G R A F I A

(1) CHURCH, A., *An Unsolvable Problem of Elementary Number Theory*, Americam Journal of Mathematics, vol.58 pp. 345 - 363, 1936.

a) *A note on the Entscheidungsproblem*, The journal of symbolic logic, vol. 1, pp.40-41,101-102.

(2) CHURCH, A., *The Calculi of lambda-conversion*, Annals of Mathematics Studies, nº 6, Princeton University Press, Princeton,N.J. 1941; reimprimido,1951.

(3) DAVIS, M., *Computability & Unsolvability*, MacGraw-Hill Book Company, New York, 1958.

(4) GÖDEL,K., *Über die Länge von Beweisen*, Ergebnisse eines mathematisches Kolloquium, Helf 4, pp.34-38,1936

a) *Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme,I*,Monatshefte für Mathematik und Physik,vol.38,pp.173-198, 1931.

(5) KLEENE,S.C., *General Recursive functions of natural numbers*, Mathematische Annlen, vol.112,pp.727-742,1936 .

(6) KLEENE,S.C., *Introduction to metamathematics*, D.Van Nostrand Company, Inc.,Princeton N.J.,1952.

(7) MARKOV,A.A., *The theory of algorithms* (Russia), Trudy Mathematicheskogo Instituta imeni V.A. Steklova, vol.42, 1951. (tradução inglesa, 1961, National Science Foundation, Washington, D.C.).

(8) MYHILL,J. *Creative Sets*, Zeitschrift für Mathematische Logik und Grundlagen der Mathematik, vol.1, pp. 97-108., 1955.

(9) PÉTER,R., *Rekursive funktionem*, Akadémiai Kiadó, Budapest , 1951.

(10) POST,E.L., *Finite combinatory processes-formulation,1*, The Journal of symbolic Logic, vol. 1, pp. 103-105 , 1936.

(11) ROGERS,H.Jr., *Theory of Recursive Functions and Effective Computability*, MacGraw-Hill Book Company, New York, 1967.

(12) TURING,A.M., *Computability and λ-definability*, The journal of Symbolic Logic, vol.2, pp.153-163, 1937.

(13) TURING,A.M., *On Computable Numbers with an Application to the Entscheidungsproblem*, Proceedings of the London Mathematical Society, ser.2, vol.42 pp.230-265 , 1936 - 1937.

****