LFA - PARTE 6

Os Limites da Computabilidade

O Que É Computável?
O Que É Possível De Ser Computado?

João Luís Garcia Rosa
LFA-FEC-PUC-Campinas 2002

1

Dois Conceitos Distintos

- Conceito de função teorético-numérica efetivamente calculável (intuitivo, filosófico, abaixo da cultura da matemática)
- Conceito de função Turing-computável (técnico)

2

Tese de Church–Turing

- Se *f* é efetivamente calculável, então *f* é Turing-computável. (não obviamente verdadeiro)
- Se *f* é Turing-computável, então *f* é efetivamente calculável. (obviamente verdadeiro)
- Uma alegação filosófica sobre a matemática

3

Suporte Empírico

- O modelo de Turing é o primeiro modelo de função calculável efetivamente
- A classe de funções Markov-computáveis é idêntica à classe de funções Turing-computáveis
- A classe de funções máquina de registradores-computáveis é idêntica à classe de funções Turing-computáveis

4

Argumento a partir da Convergência de Idéias Não Similares

- Se TCT é falsa, não se deve esperar uma das análises propostas da *função efetivamente calculável* para incluir alguma função *não* Turing-computável?
- O fato que nunca aconteceu sugere que não existe tal função, o que significa TCT verdadeira.

5

Tese de Church–Turing (cont.)

- **Contrapositivo:** Se *f* não é Turing-computável, então não é efetivamente calculável
- **Exemplo (Uma Máquina de Turing que Compila/Interpreta Programas em Linguagem C)**

6

Preparatório

- Assuma a enumeração de máquinas de Turing M_0, M_1, M_2, …
- Configuração de representação de valor (r-v)
- Terminologia: Se M_n pára na configuração r-v dado o próprio número de Gödel como entrada, então M_n *auto-pára*

7

Problema da Auto-Parada para Máquinas de Turing

- Primeiro problema insolúvel a ser considerado

8

Primeira Formulação

- Existe um algoritmo geral para determinar, para um $n \geq 0$ arbitrário, se M_n auto-pára?
- Se houver, então o Problema da Auto-Parada é solúvel

9

Preparação

- Função numérico-teorética parcial
- Função numérico-teorética não definida em nenhum lugar
- Toda máquina de Turing computa função unária
- A enumeração de máquinas de Turing induz à enumeração de funções Turing-computáveis unárias f_0, f_1, f_2, ...

10

Segunda Formulação

- Existe um algoritmo geral para determinar, para $n \geq 0$ arbitrário, se $f_n(n)$ é definida?
- Se existir, então o Problema da Auto-Parada é solúvel.

11

Preparatório

- $f^*(n) = 0$ se $f_n(n)$ é não definida
- $f^*(n) = f_n(n) + 1$ se $f_n(n)$ é definida
- f^* é bem definida
- Se o Problema da Auto-Parada é solúvel, então f^* é efetivamente computável

12

Terceira Formulação

- A função numérico-teorética unária f^* é efetivamente calculável?

13

Teorema

- Não existe máquina de Turing que compute f^*.
- Suponha algum M^* que compute f^*. Então há uma contradição.

14

Resumo

- O teorema diz que nenhuma máquina de Turing pode computar f^*
- Assumindo a Tese de Church–Turing, isto significa que f^* não é efetivamente calculável
- Mas f^* não efetivamente calculável significa que o Problema da Auto-Parada é insolúvel

15

O Problema da Parada (Completa) para Máquinas de Turing

- Existe um algoritmo geral para decidir, de $n \geq 0$ e $m \geq 0$ arbitrários, se M_n pára na configuração r-v quando começa percorrendo o símbolo mais a esquerda de $m + 1$ *1*s em uma fita não em branco? (primeira formulação)

16

Segunda Formulação

- Existe um algoritmo geral para determinar, para $n \geq 0$ e $m \geq 0$ arbitrários, se $f_n(m)$ é definida?

17

Preparatório

- $h^*(n, m) = 0$ se $f_n(m)$ é não definida
- $h^*(n, m) = f_n(m) + 1$ se $f_n(m)$ é definida
- h^* binária, total

18

Terceira Formulação

- *h** é efetivamente computável?

19

Teorema

- A função *h** não é Turing-computável.
- Suponha computada por *M**.
- Então *f** é Turing-computável, contradizendo o teorema
- Redução do Problema de Auto-Parada para o Problema de Parada Completa
- PAP para MT ≤ PPC para MT

20

Resumo

- O teorema diz que nenhuma máquina de Turing pode computar *h**
- Assumindo a Tese de Church–Turing, isto significa que *h** não é efetivamente calculável
- Mas *h** não efetivamente calculável significa que o Problema da Parada é insolúvel

21

Funções Características

- Dado o conjunto S de números naturais, a função $c_S : N \to N$
- $c_S(n) = 1$ se $n \in S$
- $c_S(n) = 0$ caso contrário (se $n \notin S$)
- *função característica do conjunto S*

22

Conjuntos Recursivos

- O conjunto S de números naturais é *recursivo* se a função característica $c_S(n)$ for Turing-computável
- $S = \{n|n$ é par$\}$ e $S^c = \{n|n$ é ímpar$\}$
- $\{n|n$ é primo$\}$
- Entendimento intuitivo (dado Church–Turing): conjuntos efetivamente decidíveis (sim/não)

23

Preliminar

- Viu-se que algumas máquinas de Turing auto-param enquanto outras não.
- K =def. $\{n|$máquina de Turing M_n auto-pára$\}$
- Fácil ver que, dado a insolubilidade do Problema da Auto-Parada, que K não é recursivo. Segue que K^c também não é recursivo.

24

Teorema de Rice

- Seja Γ qualquer conjunto não trivial de funções Turing-computáveis unárias (não vazio nem a classe de *todas* as funções Turing-computáveis unárias)
- Seja Ψ_Γ o conjunto de todos os números de gödel dos elementos de Γ (máquinas de Turing que computam)
- Então Ψ_Γ é um conjunto não recursivo de números naturais

25

Aplicações Práticas

- Seja Γ a classe de funções f definíveis em quase todo lugar (para todos os possivelmente finitamente muitos argumentos)
- A classe não trivial Ψ_Γ de números naturais
- Portanto não recursivo (por Rice)
- Portanto não efetivamente decidível (por Church–Turing)
- Então não perca tempo procurando por tal algoritmo

26

Aplicações Práticas (cont.)

- Programas C++ para computar alguma função f
- $\{n|M_n$ computa $f\}$ é classe não trivial Ψ_Γ de números naturais
- Portanto não recursivo (por Rice)
- Portanto não efetivamente decidível (por Church–Turing)
- Então não perca tempo procurando por tal programa

27