O Teorema da Recursão

O problema da auto-referência

Este estudo começa tratando de um problema da computação que envolve a reprodução de máquinas. Inicialmente, assume-se que uma máquina não consegue produzir outra de si própria. O argumento por trás dessa suposição afirma que para que uma máquina A crie outra máquina B, é necessário que B seja menos complexa do que A, pois A precisará conter em si o projeto de si mesma, assim como o projeto de B.

Entretanto, o Teorema da Recursão prova o contrário. Para demonstrar isso, antes de enunciar o teorema precisaremos definir uma Máquina de Turing chamada, na literatura, de *AUTO*. A MT *AUTO* é uma máquina capaz de imprimir sua própria descrição, ignorando qualquer entrada.

Considere a função computável $q: \Sigma^* \rightarrow \Sigma^*$ e *w* uma cadeia qualquer. A descrição P_w de uma Máquina de Turing que imprime *w* é dada pelo resultado da função q aplicado em *w*, ou seja $q(w)$. Agora, dividiremos *AUTO* em duas partes, A e B, onde a primeira imprimirá uma descrição de B e a segunda imprimirá uma descrição de A.

Para A, usaremos a máquina P_B, que imprime na fita a descrição de B, [B]. Dessa forma, A = $q([B])$, e podemos usar isso para imprimir [A], já que temos [B] impresso na fita, após a execução de A. Então B computa $q([B])$ e obtém [A]. Uma descrição dessa MT é encontrada no livro *Introdução à Teoria da Computação*, Sipser. Segue a descrição:

A = P_B, e
B = "Sobre a entrada [M], onde [M] é uma porção de uma MT:

1. Compute $q([M])$.
2. Combine o resultado com [M] para montar uma MT completa.
3. Imprima a descrição dessa MT e pare."

Acima temos a definição de nossa máquina *AUTO*. Sipser nos dá uma breve descrição de como essa MT funciona:

1. Primeiro, A roda. Ela imprime [B] na fita.
2. B começa. Ela olha para a fita e encontra sua entrada, [B].
3. B calcula q([B]) = [A] e combina isso com com [B] na descrição de uma MT, [*AUTO*].
4. B imprime essa descrição e para.

Com isso, temos uma ferramenta poderosíssima para a área da computação: a **auto-referência**: uma máquina consegue produzir outra de si mesma.

O Teorema da Recursão

O teorema da recursão se aproveita dos resultados anteriores para mostrar que podemos facilmente construir uma MT que pode obter sua própria descrição e computar com ela. A formalização do teorema é a seguinte:

"**Teorema da recursão**: Seja T uma máquina de Turing que computa uma função t: $\Sigma^* \times \Sigma^* \rightarrow \Sigma^*$. Então existe uma máquina de turing R que computa uma função r: $\Sigma^* \rightarrow \Sigma^*$, onde para toda *w*:

$$r(w) = t([R], w)$$ "

O que o teorema diz é que podemos ter uma MT R tal que [R] = [ABT], onde A e B são as partes correspondentes em *AUTO* (sendo que, nesse caso, A = P_{BT}), e T contém a computação de R sobre uma cadeia *w*. O que a máquina R faz é: imprime na fita, após o conteúdo de *w*, a descrição de si mesma e, em seguida, utiliza essa descrição para computar *w*.

Uma consequência direta desse teorema é que podemos utilizar, na construção de uma MT generalizada M, a frase "**obtenha a própria descrição [M]**". Isso é útil não só para imprimir a própria descrição, mas também para utilizá-la para computar alguma cadeia ou, por exemplo, para contar o número de estados de uma MT.

Problema de aceitação de máquina de Turing

Usando do Teorema da Recursão, conseguimos provar, por contradição, que A_{mt} é indecidível.

"Assumimos que a máquina de Turing H decide A_{mt}, para os propósitos de se obter uma contradição. Construímos a seguinte máquina B.

B = Sobre a entrada w:

1. Obtenha, por meio do Teorema da Recursão, sua própria descrição $\langle B \rangle$.
2. Rode H sobre a entrada $\langle B,w \rangle$.
3. Faça o oposto do que H diz, ou seja, aceite se H rejeita e rejeite se H aceita."

Como descrito, temos duas máquinas de Turing H e B, onde H roda sobre a descrição de B com a mesma cadeia w e tem como resposta o oposto de B, chegando assim em uma contradição.

Máquinas de turing Mínimas

"Se M é uma máquina de Turing, então dizemos que o comprimento da sua descrição ⟨M⟩ de M é o número de símbolos na cadeia que descreve M. Dizemos que M é **mínima** se não existe máquina de Turing equivalente a M que tenha uma descrição mais curta. Seja

$$MIN_{mt} = \{ \langle M \rangle \mid M \text{ é uma MT mínima}\}. "$$

O que esta definição diz é que, dado uma MT_1, como existe uma MT que descreve MT_1, a descrição desta máquina deve ser do tamanho do número de símbolos da cadeia que a descreve. Dado o tamanho da descrição, se o seu tamanho é o mínimo possível então MT_1 é mínima. Esta Máquina de Turing será usada no próximo teorema para provar que uma máquina de Turing mínima não é Turing-reconhecível.

Máquina de turing mínima não é Turing-reconhecível.

Este teorema demonstra que uma máquina mínima de Turing não é Turing reconhecível. Em outras palavras , que não é uma recursivamente-enumerável.

Para provar este teorema, tomamos que uma MT A enumera MIN_{mt} e chegamos em uma contradição.

"Construímos a seguinte MT C.

C = Sobre a entrada de w:

1. Obtenha, através do teorema da recursão, sua própria descrição ⟨C⟩.
2. Rode o enumerador A até que uma máquina D apareça com uma descrição mais longa que aquela de C.
3. Simule D sobre a entrada w."

A partir do descrito, como o enumerador A contém uma descrição mais longa do que a descrição de C, feita no passo 1. Isso se deve ao fato de a MIN_{mt} ser infinita por definição. No passo dois, encontramos D que tem uma descrição mais longa da máquina C, sendo as duas equivalentes. Sendo assim D não pode ser mínima, mas D faz parte das MT de A, chegando assim em uma contradição.

Teorema de Ponto fixo

Para contextualizar, um ponto fixo representa um valor que não é alterado mesmo quando aplicada uma função. Este teorema é uma aplicação do Teorema da Recursão.

Para provar este teorema, selecionamos funções que são transformações computáveis de descrições de máquinas de Turing e para quaisquer transformações aplicadas, existe alguma máquina de Turing que seu comportamento não é alterado.

“Seja $t : \Sigma^* \rightarrow \Sigma^*$ uma função computável. Então existe uma máquina de Turing F para a qual $t(\langle F \rangle)$ descreve uma máquina de Turing equivalente a F.”

Se tivermos uma cadeia que não codifica legitimamente uma máquina de Turing, a máquina descrita sempre irá rejeitar.

No teorema , T é a transformação e F o ponto fixo do nosso problema.

“Seja F a seguinte máquina de Turing.

F = Sobre a entrada w:

1. Obtenha, como no teorema da recursão, sua propria descricao $\langle F \rangle$.
2. Compute $t(\langle F \rangle)$ para obter a descrição de MT G.
3. Simule G sobre w.”

Pelos simples fato de F simular G, temos uma correspondência entre $\langle F \rangle$ e $t(\langle F \rangle)$ onde ambos são iguais a G.

Aplicações

Uma aplicação que na prática é bem popular na computação são os vírus de computadores. A ideia principal é similar a ideia do vírus biológico. O vírus só se ativa quando é colocado de forma apropriada no computador hospedeiro, e uma vez ativado, o hospedeiro está “infectado”. Uma vez infectado, esse hospedeiro, como na biologia, pode transmitir esse vírus a outras máquinas, ou seja, o vírus é capaz de se auto-replicar. É justamente essa ideia de auto-replicação explicada acima no relatório que é a usada para que o vírus consiga se espalhar.

Além dessa aplicação prática de uso do teorema da recursão, ela veio como uma nova visão para a resolução de provas, em geral mais simples.

Referências

https://ocw.mit.edu/courses/electrical-engineering-and-computer-science/6-045j-automata-computability-and-complexity-spring-2011/lecture-notes/MIT6_045JS11_lec10.pdf

http://cs.umw.edu/~finlayson/class/fall14/cpsc326/notes/22-recursion-theorem.html

Livro : Introduction to the theory of computation - Michael Sipser.