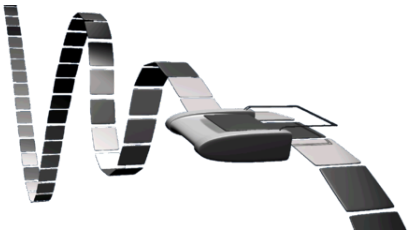
Máquina de Turing



Representação artística de uma máquina de Turing

A **máquina de Turing** é um dispositivo teórico conhecido como *máquina universal*, que foi concebido pelo matemático britânico Alan Turing (1912-1954), muitos anos antes de existirem os modernos computadores digitais (o artigo de referência foi publicado em 1936). Num sentido preciso, é um modelo abstrato de um computador, que se restringe apenas aos aspectos lógicos do seu funcionamento (memória, estados e transições) e não à sua implementação física. Numa máquina de Turing pode-se modelar qualquer computador digital.

Turing também se envolveu na construção de máquinas físicas para quebrar os códigos secretos das comunicações alemãs durante a Segunda Guerra Mundial, tendo utilizado alguns dos conceitos teóricos desenvolvidos para o seu modelo de *computador universal*.

1 Definição

1.1 Descrição informal

Uma máquina de Turing consiste em:

1. Uma *fita* que é dividida em células, uma adjacente à outra. Cada célula contém um símbolo de algum alfabeto finito. O alfabeto contém um símbolo especial *branco* (aqui escrito como ¬) e um ou mais símbolos adicionais. Assume-se que a fita é arbitrariamente extensível para a esquerda e para a direita, isto é, a máquina de Turing possui tanta fita quanto é necessário para a computação. Assume-se também que células que ainda não foram escritas estão preenchidas com o símbolo branco.
2. Um *cabeçote*, que pode ler e escrever símbolos na fita e mover-se para a esquerda e para a direita.
3. Um *registrador de estados*, que armazena o estado da máquina de Turing. O número de estados diferentes é sempre finito e há um estado especial denominado *estado inicial* com o qual o registrador de estado é inicializado.
4. Uma *tabela de ação* (ou *função de transição*) que diz à máquina que símbolo escrever, como mover o cabeçote ($\leftarrow$ para esquerda e $\rightarrow$ para direita) e qual será seu novo estado, dados o símbolo que ele acabou de ler na fita e o estado em que se encontra. Se não houver entrada alguma na tabela para a combinação atual de símbolo e estado então a máquina pára.

Note que cada parte da máquina é finita; é sua quantidade de fita potencialmente ilimitada que dá uma quantidade ilimitada de espaço de armazenamento.

1.2 Definição formal

1.2.1 Máquina de Turing com uma fita

Mais formalmente, uma máquina de Turing (com uma fita) é usualmente definida como uma Tupla $M = (Q, \Sigma, \Gamma, s, b, F, \delta)$, onde

- Q é um conjunto finito de estados
- Σ é um alfabeto finito de símbolos
- Γ é o alfabeto da fita (conjunto finito de símbolos)
- $s \in Q$ é o estado inicial
- $b \in \Gamma$ é o símbolo branco (o único símbolo que se permite ocorrer na fita infinitamente em qualquer passo durante a computação)
- $F \subseteq Q$ é o conjunto dos estados finais
- $\delta : Q \times \Gamma \Rightarrow Q \times \Gamma \times \{\leftarrow, \rightarrow\}$ é uma função parcial chamada função de transição, onde $\leftarrow$ é o movimento para a esquerda e $\rightarrow$ é o movimento para a direita.

Definições na literatura às vezes diferem um pouco, para tornar argumentos ou provas mais fáceis ou mais claras, mas isto é sempre feito de maneira que a máquina resultante tem o mesmo poder computacional. Por exemplo, mudar o conjunto $\{\leftarrow, \rightarrow\}$ para $\{\leftarrow, \rightarrow, P\}$, onde *P* permite ao cabeçote permanecer na mesma célula da fita em vez de mover-se para a esquerda ou direita, não aumenta o poder computacional da máquina (Fonte - Michael Sipser - Int Teoria da Computação).

1.2.2 Máquina de Turing com *k* fitas

Uma máquina de Turing com k fitas também pode ser descrita como uma 7-upla $M = (Q, \Sigma, \Gamma^1, \Gamma^2, ..., \Gamma^k, s, b, F, \delta)$, onde

- Q é um conjunto finito de estados
- Σ é um alfabeto finito de símbolos
- $\Gamma^i, i = 1, ..., k$ é o alfabeto da fita i (conjunto finito de símbolos)
- k é o número de fitas
- $s \in Q$ é o estado inicial
- $b \in \Gamma$ é o símbolo branco
- $F \subseteq Q$ é o conjunto dos estados finais
- $\delta : Q \times \Gamma \Rightarrow Q \times \Gamma \times \{\leftarrow, \rightarrow\}$ é uma função parcial chamada função de transição, onde $\leftarrow$ é o movimento para a esquerda e $\rightarrow$ é o movimento para a direita.

Note que uma máquina de turing com "k" fitas não é mais poderosa que uma máquina de Turing tradicional.

2 Detalhes adicionais requeridos para visualizar ou implementar máquinas de Turing

Nas palavras de Van Emde Boas (1990), p. 6: "O objeto do conjunto teórico [sua descrição formal sete-tupla similar ao acima] fornece apenas informação parcial sobre como a máquina agirá e com o que suas computações se parecerão."

Por exemplo,

- Serão necessárias muitas decisões sobre com o que os símbolos realmente se parecem, e uma forma de contraprova de símbolos de escrita e leitura indefinidamente.
- As operações shift left e shift right podem deslocar a cabeça da fita sobre a fita, mas quando realmente construir uma máquina de Turing, isso é mais prático para fazer a fita deslizar para frente e para trás sob a cabeça em vez disso.
- A fita pode ser finita, e automaticamente estendida com espaços em branco, conforme necessário (o qual é o mais próximo da definição matemática), mas isso é mais comum para pensar sobre isso como infinitamente alongado em ambos os lados e sendo preenchida com espaços em branco, exceto sobre o fragmento finito dado explicitamente onde a cabeça da fita está. (Isto não é, claro, implementável na prática.) A fita não pode ser fixada em comprimento, desde que não corresponderia à definição dada e limitaria seriamente o alcance das computações que a máquina pode realizar àquelas do autômato linearmente limitado.

2.1 Definições alternativas

Para tornar as provas e argumentos mais fáceis ou mais claras, encontramos na literatura definições levemente diferentes, mas isso sempre é feito de tal maneira que a máquina resultante tenha o mesmo poder computacional. Por exemplo, modificando o conjunto $\{L, R\}$ para $\{L, R, N\}$, onde *N* ("Nada" ou "Sem operação") permitiria a máquina ficar sobre a mesma célula da fita em vez de mover para a esquerda ou para direita, não aumenta o poder computacional das máquinas.

A convenção mais comum representa cada "instrução de Turing" na "tabela de Turing" por uma de nove 5-uplas, pela convenção de Turing/Davis (Turing (1936) em *Undecidable*, p. 126-127 e Davis (2000) p. 152):

(definição 1): **(q_i, S_j, S_k/E/N, L/R/N, q_m)**

> **(** estado atual **q_i ,** símbolo escaneado **S_j ,** símbolo de impressão **S_k**/apagar **E**/nada **N ,** move_um_quadrado_da_fita esquerda **L**/ direita **R**/ nada **N ,** novo estado **q_m)**

Outros autores (Minsky (1967) p. 119, Hopcroft e Ullman (1979) p. 158, Stone (1972) p. 9) adotam uma convenção diferente, com um novo estado **q_m** listado imediatamente depois do símbolo escaneado S_j:

(definição 2): **(q_i, S_j, q_m, S_k/E/N, L/R/N)**

> **(** estado atual **q_i ,** símbolo escaneado **S_j ,** novo estado **q_m ,** símbolo de impressão **S_k**/apagar **E**/nada **N ,** move_um_quadrado_da_fita esquerda **L**/direita **R**/nada **N)**

Para o restante desse artigo a "definição 1" (a convenção de Turing/Davis) será utilizada.

Na tabela seguinte, o modelo original de Turing permitiu apenas as primeiras três linhas, que ele chamou N1, N2, N3 (cf Turing em *Undecidable*, p. 126). Ele permitiu a rasura do "quadrado escaneado" nomeando o 0th símbolo S_0 = "apagar" ou "branco", etc. Entretanto, ele não permitiu a não impressão, então toda linha de instrução inclui "símbolo de impressão S_k" ou "apagar" (cf nota de rodapé 12 em Post (1947), *Undecidable* p. 300). As abreviações são de Turing(*Undecidable* p. 119). Posteriormente ao paper original de Turing em 1936–1937, os

modelos de máquina têm permitido todos os nove tipos possíveis de 5-tuplas:

Qualquer tabela de Turing (lista de instruções) pode ser construída a partir das nove 5-tuplas acima. Por razões técnicas, as três não-impressão ou instruções "N" (4, 5, 6) pode geralmente ser dispensadas. Para exemplos, veja exemplos de máquina de Turing.

Menos frequentemente, o uso de 4-tuplas são encontrados: estes representam uma outra atomização das instruções de Turing (cf Post (1947), Boolos & Jeffrey (1974, 1999), Davis-Sigal-Weyuker (1994)); também veja mais em Máquina de Post–Turing.

2.2 O "estado"

A palavra "estado" utilizada em contexto de máquinas de Turing pode ser uma fonte de confusão, como isso pode significar duas coisas. A maioria dos comentaristas depois de Turing utilizaram "estado" para denotar o nome/designador da instrução atual para ser realizada — i.e. , os conteúdos do registro de estado. Mas Turing (1936) fez uma distinção forte entre um registro do que ele chamou de m-configuração da máquina, (seu estado interno) e o "estado de progresso" da máquina (ou pessoa) através da computação - o estado atual do sistema total. O que Turing chamou "a fórmula do estado" inclui ambos a instrução atual e *todos* os símbolos sobre a fita:

Anteriormente em seu *paper*, Turing estendeu isto ainda mais: ele dá um exemplo onde ele posicionou um símbolo da configuração atual m —o rótulo da instrução— abaixo do quadrado escaneado, juntamente com todos os símbolos sobre a fita (*Undecidable*, p. 121); Isto ele chama "a *configuração completa*" (*Undecidable*, p. 118). Para imprimir a "configuração completa" sobre uma linha, ele posiciona o estado-rótulo/m-configuração para a *esquerda* do símbolo escaneado.

Uma variante disto é vista em Kleene (1952), onde Kleene mostra como escrever o número de Gödel de uma "situação" da máquina: ele posiciona o símbolo da "m-configuração" q_4 sobre o quadrado escaneado em grosseiramente o centro dos 6 quadrados não-brancos sobre a fita (veja a imagem da fita de Turing nesse artigo) e coloca isso para a *direita* do quadrado escaneado. Mas Kleene refere-se ao próprio "q_4" como "o estado da máquina" (Kleene, p. 374-375). Hopcroft e Ullman chamam este composto a "descrição instantânea" e segue a convensão de Turing de colocar o "estado atual" (rótulo-instrução, m-configuração) para a *esquerda* do símbolo escaneado (p. 149).

Exemplo: estado total do problema do castor ocupado de 3 estados e 2 símbolos após 3 "movimentos" (pegando de exemplo "run" na figura abaixo):

1**A**1

Isso significa: após três movimentos, a fita tem ... 000110000 ... sobre esta, a cabeça está escaneando o 1 mais a direita, e o estado é **A**. Brancos (nesse caso, representado por "0"s) pode ser parte do estado total como mostrado aqui: **B**01; a fita possui um único 1 sobre esta, mas a cabeça está escaneando o 0 ("branco") para sua esquerda e o estado é **B**.

"Estado" no contexto de máquinas de Turing devem ser esclarecidas quanto ao que está sendo descrito: (*i*) a instrução atual, ou (*ii*) a lista de símbolos sobre a fita juntamente com a instrução atual, ou (*iii*) a lista de símbolos sobre a fita juntamente com a instrução atual posicionada pela esquerda do símbolo escaneado ou pela direita do símbolo escaneado.

O biógrafo de Turing, Andrew Hodges (1983: 107), tem notado e discutido esta confusão.

3 Exemplo

A máquina de Turing a seguir tem um alfabeto {¬, 1}, onde ¬ representa o símbolo branco. Ela espera uma série de 1's na fita, com o cabeçote inicialmente no 1 mais à esquerda, e duplica os 1's com um ¬ no meio. Por exemplo, "111" torna-se "111¬111". O conjunto dos estados é {s1, s2, s3, s4, s5} e o estado inicial é s1. A tabela de ação é dada a seguir.

Est. Símb. Símb. Est. Act. Lido Escr. Mv. Novo ---- ----- ----- --- ---- s1 1 ¬ → s2 s2 1 1 → s2 s2 ¬ ¬ → s3 s3 ¬ 1 ← s4 s3 1 1 → s3 s4 1 1 ← s4 s4 ¬ ¬ ← s5 s5 1 1 ← s5 s5 ¬ 1 → s1

A primeira linha desta tabela pode ser lida como: "Se a máquina estiver no estado s1 e o símbolo lido pelo cabeçote for 1, então escreva o símbolo ¬, mova uma posição para a direita e mude o estado para s2".

Uma computação nesta máquina de Turing pode ser, por exemplo: (a posição do cabeçote é indicada mostrando-se a célula em negrito)

Passo Estado Fita ----- ------ ---------- 1 s1 **1**1 2 s2 ¬**1** 3 s2 ¬1¬ 4 s3 ¬1¬¬ 5 s4 ¬1¬1 6 s5 ¬**1**¬1 7 s5 ¬1¬1 8 s1 1**1**¬1 9 s2 1¬¬1 10 s3 1¬¬**1** 11 s3 1¬¬1¬ 12 s4 1¬¬**1**1 13 s4 1¬¬11 14 s5 1¬¬11 15 s1 11¬11

O comportamento desta máquina pode ser descrito como um laço (loop): Ele inicia em s1, substitui o primeiro 1 com um ¬, então usa o s2 para mover para a direita, passando pelos 1's e pelo primeiro ¬ encontrado. S3 então passa pela próxima sequência de 1's (inicialmente há nenhuma) e substitui o primeiro ¬ que encontra por um 1. S4 move de volta para a esquerda, passando pelos 1's até encontrar um ¬ e vai para o estado s5. S5 então move para a esquerda, passando pelos 1's até achar o ¬ que foi originalmente escrito por s1. Ele substitui o ¬ por 1, move uma posição para a direita e entra no estado s1 novamente

para outra execução do laço. Isso continua até s1 achar um ¬ (este é o ¬ que fica entre as duas cadeias de 1's), situação na qual a máquina pára.

4 Máquinas de Turing determinísticas e não-determinísticas

Se a tabela de ação tem no máximo uma entrada para cada combinação de símbolo e estado então a máquina é uma **máquina de Turing determinística** (MTD). Se a tabela de ação contém múltiplas entradas para uma combinação de símbolo e estado então a máquina é uma **máquina de Turing não-determinística** (MTND ou MTN).

5 Máquinas de Turing universais

- Artigo principal Máquina de Turing universal

Toda máquina de Turing computa uma certa função computável parcial a partir da cadeia dada formada pelos símbolos do alfabeto. Neste sentido ela comporta-se como um computador com um programa fixo. No entanto, como Alan Turing descreveu, podemos codificar a tabela de ação de qualquer máquina de Turing em uma cadeia de símbolos. Portanto podemos tentar construir uma máquina de Turing que espera em sua fita uma cadeia descrevendo a tabela de ação seguida por uma cadeia descrevendo a fita de entrada, e então computa a fita que a máquina de Turing codificada teria computado.

6 Comparação com máquinas reais

Frequentemente diz-se que as máquinas de Turing, ao contrário de autômatos mais simples, são tão poderosas quanto máquinas reais, e são capazes de executar qualquer operação que um programa real executa. O que está faltando neste enunciado é que praticamente qualquer programa particular executando em uma *máquina particular* e dada uma entrada finita é, na verdade, nada além de um autômato finito determinístico, já que a máquina em que executa pode estar apenas em uma quantidade finita de *configurações*. Máquinas de Turing poderiam de fato ser equivalentes a uma máquina que tenha uma quantidade ilimitada de espaço de armazenamento. Podemos questionar então por que as máquinas de Turing são modelos úteis de computadores reais. Há várias maneiras de responder a isto:

1. A diferença está apenas na habilidade de uma máquina de Turing de manipular uma quantidade ilimitada de dados. No entanto, dada uma quantidade finita de tempo, uma máquina de Turing (como uma máquina real) pode apenas manipular uma quantidade finita de dados.
2. Como uma máquina de Turing, uma máquina real pode ter seu espaço de armazenamento aumentado conforme a necessidade, através da aquisição de mais discos ou outro meio de armazenamento. Se o suprimento destes for curto, a máquina de Turing pode se tornar menos útil como modelo. Mas o fato é que nem as máquinas de Turing nem as máquinas reais precisam de quantidades astronômicas de espaço de armazenamento para fazer a maior parte das computações que as pessoas normalmente querem que sejam feitas. Frequentemente o tempo de processamento requerido é o maior problema.
3. Máquinas reais são muito mais complexas que uma máquina de Turing. Por exemplo, uma máquina de Turing descrevendo um algoritmo pode ter algumas centenas de estados, enquanto o autômato finito determinístico equivalente em uma dada máquina real tem quadrilhões.
4. Máquinas de Turing descrevem algoritmos independentemente de quanta memória eles utilizam. Há um limite máximo na quantidade de memória que qualquer máquina que conhecemos tem, mas este limite pode crescer arbitrariamente no tempo. As máquinas de Turing nos permitem fazer enunciados sobre algoritmos que (teoricamente) valerão eternamente, independentemente dos avanços na arquitetura de computadores *convencionais*.
5. Máquinas de Turing simplificam o enunciado de algoritmos. Algoritmos executando em máquinas abstratas equivalentes a Turing são normalmente mais gerais que suas contrapartes executando em máquinas reais, porque elas têm tipos com precisão arbitrária disponíveis e nunca precisam tratar condições inesperadas (incluindo, mas não somente, acabar a memória).

Uma maneira em que máquinas de Turing são pobres modelos para programas é que muitos programas reais, tais como sistemas operacionais e processadores de texto, são escritos para receber entradas irrestritas através da execução, e portanto não param. Máquinas de Turing não modelam tal "computação contínua" bem (mas ainda podem modelar porções dela, tais como procedimentos individuais).

Outra limitação de máquinas de Turing é que elas não modelam a organização estrita de um problema específico. Por exemplo, computadores modernos são na verdade instâncias de uma forma mais específica de máquina de computação, conhecido como máquina de acesso aleatório. A principal diferença entre esta máquina e a máquina de Turing é que esta utiliza uma fita infinita, enquanto a máquina de acesso aleatório utiliza uma sequência indexada numericamente (tipicamente um campo inteiro). O resultado desta distinção é que há otimizações computacionais que podem ser executadas baseadas nos índices em memória, o que não é possível numa máquina

de Turing geral. Assim, quando máquinas de Turing são utilizadas como base para tempo de execuções restritos, um "falso limite inferior" pode ser provado em determinados tempos de execução de algoritmos (graças à premissa falsa de simplificação da máquina de Turing). Um exemplo disto é uma "ordenação por contagem", o que aparentemente viola o limite inferior $theta(n \log n)$ em algoritmos de ordenação.

7 Máquinas de Turing físicas

Não é difícil simular uma máquina de Turing num computador moderno (exceptuando pela quantidade de memória limitada existente nos computadores actuais). O *site* de busca Google, em comemoração aos 100 anos de Alan Turing, publicou um *doodle* dia 23 de junho de 2012, em forma de uma máquina de Turing.[1]

É possível construir uma máquina de Turing com base puramente mecânica. O matemático Karl Scherer construiu essa máquina em 1986, usando conjuntos de construção de metal, plástico e alguma madeira. A máquina, com 1,5 m de altura, usa puxões de fios para ler, movimentar e escrever informação, a qual é, por sua vez, representada por rolamentos. A máquina encontra-se atualmente em exibição na entrada do Departamento de Ciência de Computadores da Universidade de Heidelberg, na Alemanha.

O conceito de máquina de Turing foi usado como ferramenta educativa na obra de ficção científica *The Diamond Age* (1995), escrita por Neal Stephenson. A personagem principal, Nell, possui um livro interactivo que a ensina a pensar criativa e logicamente apresentando-lhe puzzles numa história, os quais, sendo máquinas de Turing, se tornam cada vez mais complexos à medida que a narrativa se desenvolve. Estes puzzles começam por ser simples aparelhos mecânicos e evoluem para processos econômicos abstractos, atingindo um ponto em que se assiste à interação entre completos reinos ficcionais.

8 História

8.1 Contexto histórico: máquinas computacionais

Robin Gandy (1919-1995), um estudante de Alan Turing (1912-1954) e seu amigo ao longo da vida, traça a linhagem da noção de "máquina de calcular" até a Babbage (em cerca de 1834) e, na verdade, propõe a "Tese de Babbage":

A análise de Gandy da Máquina Analítica de Babbage descreve as seguintes cinco operações (cf. p. 52-53):

1. As funções aritméticas +, -, ×, onde - indica subtração "apropriada" x - y = 0 se y ≥ x
2. Qualquer sequência de operações é uma operação
3. Iteração de uma operação (repetir n vezes uma operação P)
4. Iteração condicional (repetir n vezes uma operação P condicional sobre o "sucesso" do teste T)
5. Transferência condicional (i.e. condicional "goto").

Gandy afirma que "as funções que podem ser calculadas por (1), (2), e (4) são precisamente as que são Turing computáveis . " (p. 53). Ele cita outras propostas de "máquinas de calcular universais" incluídas as de Percy Ludgate (1909), Leonardo Torres y Quevedo (1914), Maurice d'Ocagne (1922), Louis Couffignal (1933), Vannevar Bush (1936), Howard Aiken (1937). No entanto:

8.2 O Entscheidungsproblem (o "problema de decisão"): O décimo problema de Hilbert de 1900

Com relação aos problemas de Hilbert propostos pelo famoso matemático David Hilbert em 1900, um aspecto do problema #10 tinha ficado flutuando por quase 30 anos antes de ser enquadrado com precisão. A expressão original de Hilbert para o 10º problema é a seguinte:

Em 1922, essa noção de " Entscheidungsproblem "desenvolveu-se um pouco, e H. Behmann afirmou que

Até o Congresso Internacional de Matemáticos em 1928, Hilbert "fez suas perguntas bastante precisas. Primeiro, a matemática é completa ... Segundo, a matemática é consistente ... E terceiro, a matemática édecidível ? " (Hodges p. 91, Hawking p. 1121). As duas primeiras questões foram respondidas em 1930 por Kurt Gödel na mesma reunião onde Hilbert fez seu discurso de aposentadoria (para desgosto de Hilbert); o terceiro- o Entscheidungsproblem teve que esperar até meados de 1930.

O problema era que uma resposta primeiro exigia uma definição precisa de "prescrição definitiva geral aplicável", que o professor de Princeton Alonzo Church viria a

chamar de " método efetivo ", e em 1928 não existia tal definição. Mas ao longo dos próximos 6-7 anos Emil Post desenvolveu sua definição de um trabalhador passando de um cômodo para um outro cômodo escrevendo e apagando marcas em uma lista de instruções (Post 1936), assim como Church e os seus dois alunos Stephen Kleene e J. B. Rosser pelo uso do lambda-cálculo de Church e a teoria da recursão de Gödel (1934). O artigo de Church (publicado 15 de abril de 1936) mostrou que o Entscheidungsproblem era de fato “indecidível” e superou Turing por quase um ano (o artigo de Turing foi submetido em 28 de maio de 1936 e publicado em janeiro de 1937). Nesse meio tempo, Emil Post apresentou um breve artigo no outono de 1936, Turing, pelo menos, tinha prioridade sobre Post. Enquanto Church arbitrou o artigo de Turing, Turing teve tempo para estudar o artigo de Church e adicionar um Apêndice onde ele provava que o lambda-cálculo de Church e suas máquinas calculariam as mesmas funções.

E Post só havia proposto uma definição de calculabilidade e criticou a “definição” de Church, mas não tinha provado nada.

8.3 Alan Turing uma-máquina(-automática)

Na primavera de 1935, Turing como aluno de mestrado em Kings College Cambridge , Reino Unido , aceitou o desafio; ele tinha sido estimulado pelas palestras do lógico M. H. A. Newman “e aprendeu com elas do trabalho de Gödel e o Entscheidungsproblem ... Newman usou a palavra “mecânico” ... Em seu obituário de Turing em 1955 Newman escreveu:

Gandy afirma que:

Enquanto Gandy acredita que a declaração de Newman acima era “enganosa”, esta opinião não era compartilhada por todos. Turing tinha um interesse ao longo da vida por máquinas: “Alan tinha sonhado de inventar máquinas de escrever quando menino; [sua mãe] Sra. Turing teve uma máquina de escrever, e ele poderia muito bem ter começado perguntando a si mesmo o que significava chamar uma máquina de escrever 'mecânica'" (Hodges, p. 96). Enquanto em Princeton em busca de seu PhD, Turing construiu um multiplicador com lógica booleana (veja abaixo). Sua tese de doutorado, intitulada “Sistemas de Lógica Baseado em ordinais”, contém a seguinte definição de “uma função computável":

Quando Turing retornou ao Reino Unido, ele acabou se tornando juntamente responsável por quebrar os códigos secretos alemães criados por máquinas de criptografia chamadas de “O Enigma”, ele também se envolveu no projeto da ACE (Automatic Computing Engine), “a proposta ACE [de Turing] foi efetivamente independente, e as suas raízes não estava no EDVAC [iniciativa dos EUA], mas em sua própria máquina universal "(Hodges p. 318). Argumentos continuam sobre a origem e a natureza do que tem sido chamado por Kleene (1952) Tese de Turing . Mas o que Turing provou com seu modelo de máquina-computacional aparece em seu artigo On Computable Numbers, With an Application to the Entscheidungsproblem (1937):

De Turing exemplo (a sua segunda prova): Se alguém perguntar por um procedimento geral para nos dizer: “Será que esta máquina imprime 0”, a pergunta é “indecidível”.

8.4 1937-1970: O “computador digital”, o nascimento da “ciência da computação”

Em 1937, enquanto em Princeton trabalhando em sua tese de doutorado, Turing construiu um multiplicador (lógico-booleano) digital do nada, fazendo seus próprios relés eletromecânicos (Hodges p. 138). “A tarefa de Alan foi para encarnar o projeto lógico de uma máquina de Turing em uma rede de relé-operacinados interruptores ...” (Hodges p. 138). Embora Turing pudesse estar inicialmente curioso e experimentando, trabalhos muito sérios, no mesmo sentido, estavam sendo desenvolvidos na Alemanha (Konrad Zuse (1938)), e nos Estados Unidos (Howard Aiken) e George Stibitz (1937); os frutos do seus trabalhos foram usados pelos militares do Eixo e dos Aliados na Segunda Guerra Mundial (cf Hodges p. 298-299).No início e meados da década de 1950, Hao Wang e Marvin Minsky reduziram a máquina de Turing a uma forma mais simples (um precursor da máquina de Post-Turing de Martin Davis); simultaneamente pesquisadores europeus estavam reduzindo o novíssimo computador eletrônico a um computador como objeto teórico equivalente ao que estava sendo chamado de “máquina de Turing”. No final dos anos 1950 e início dos anos 1960, os desenvolvimentos, coincidentemente paralelamente, Melzak e Lambek (1961), Minsky (1961), e Shepherdson e Sturgis (1961) continuaram o trabalho europeu e reduziram a máquina de Turing para uma máquina mais amigável, como um computador de modelo abstrato chamado de counter machine ; Elgot e Robinson (1964), Hartmanis (1971), Cook e Reckhow (1973) levaram este trabalho ainda mais com a máquina de registo e de acesso

aleatório máquina modelos, mas basicamente todos são apenas máquinas multi-fita de Turing com um conjunto de instruções aritméticas.

8.5 1970-presente: a máquina de Turing como um modelo de computação

Hoje, o contador, registrador e máquinas de acesso aleatório e seu pai a máquina de Turing continuam a ser os modelos de escolha para os teóricos que investigam questões da teoria da computação . Em particular, a teoria da complexidade computacional faz uso da máquina de Turing;

Dependendo dos objetos um gosta de manipular os cálculos (números como inteiros não negativos ou cordas alfanuméricos), dois modelos têm obtido uma posição dominante na teoria da complexidade baseada em máquina:

9 Ver também

- Formiga de Langton, uma representação simples da máquina de Turing em duas dimensões
- Tese de Church-Turing, que diz que máquinas de Turing podem executar qualquer computação que pode ser executada
- brainfuck, é uma linguagem Turing completa, desenhada para desafiar e confundir os programadores

10 Referências

[1] Google. "Doodles". www.google.com. Consultado em 23 de junho de 2012.

11 Bibliografia

- Rolf Herken: *The Universal Turing Machine - A Half-Century Survey*, Springer Verlag, ISBN 3-211-82637-8
- Paul Strathern: *Turing and the Computer - The big idea*, Anchor Books/Doubleday, ISBN 0-385-49243-X
- Turing, A., *On Computable Numbers, With an Application to the Entscheidungsproblem*, Proceedings of the London Mathematical Society, Series 2, Volume 42, 1936; reimpresso em M. David (ed.), *The Undecidable*, Hewlett, NY: Raven Press, 1965;
- Boolos, G. and Jeffrey, R., *Computability and Logic*, 2ª ed., Cambridge: Cambridge University Press, 1980.
- Rogozhin, Yurii, "A Universal Turing Machine with 22 States and 2 Symbols", *Romanian Journal Of Information Science and Technology*, 1(3), 259-265, 1998. (surveys known results about small universal Turing machines)

12 Ligações externas

- Turing Machines (em inglês) Enciclopédia de Filosofia de Stanford
- The Church-Turing Thesis (em inglês) Enciclopédia de Filosofia de Stanford
- Turing Machine Simulator (em inglês) simulador de máquina de Turing

13 Fontes, contribuidores e licenças de texto e imagem

13.1 Texto

- **Máquina de Turing** *Fonte:* https://pt.wikipedia.org/wiki/M%C3%A1quina_de_Turing?oldid=44042465 *Contribuidores:* Patrick-br, LeonardoG, Muriel Gottrop, Angeloleithold, E2mb0t, Heitor, LeonardoRob0t, Alexg, Rafael.afonso, Campani, Nuno Tavares, NTBot, Rodrigo Rocha, RobotQuistnix, Leslie, Clara C., OS2Warp, Adailton, Zwobot, Nmsalgueiro, Fasouzafreitas, YurikBot, Cícero, Luís Felipe Braga, Villarinho, Leonardo.stabile, Njr, Marcelo Victor, Thijs!bot, Escarbot, JAnDbot, Kid A~ptwiki, Ricvelozo, TXiKiBoT, VolkovBot, BOTijo, Kaktus Kid, Lopima, LeoBot, Marcelopge, Vitor Mazuco, ThrasherÜbermensch, Luckas-bot, Eamaral, Salebot, Xqbot, RibotBOT, Ricardo Ferreira de Oliveira, Michelmenega, TobeBot, Rjbot, Arthyole, Ci.cp, EmausBot, JackieBot, ZéroBot, Sygmn, Nelson Teixeira, Stuckkey, Antero de Quintal, Legobot, Luiz Vasconcelos Filho, Zicane, Mariama Serafim, Vítor, Astrogildaadroalda e Anónimo: 51

13.2 Imagens

- **Ficheiro:Commons-logo.svg** *Fonte:* https://upload.wikimedia.org/wikipedia/commons/4/4a/Commons-logo.svg *Licença:* Public domain *Contribuidores:* This version created by Pumbaa, using a proper partial circle and SVG geometry features. (Former versions used to be slightly warped.) *Artista original:* SVG version was created by User:Grunt and cleaned up by 3247, based on the earlier PNG version, created by Reidab.
- **Ficheiro:Turing_Machine.png** *Fonte:* https://upload.wikimedia.org/wikipedia/commons/b/b7/Turing_Machine.png *Licença:* Public domain *Contribuidores:* No machine-readable source provided. Own work assumed (based on copyright claims). *Artista original:* No machine-readable author provided. Porao assumed (based on copyright claims).

13.3 Licença

- Creative Commons Attribution-Share Alike 3.0