# Máquina Norma

Professor Dr. Ricardo Ferreira de Oliveira

Estudaremos neste texto uma introdução ao estudo de mais uma máquina universal, a máquina Norma. Esta é uma máquina sobremaneira simples com poder computacional equivalente ao da máquina de Turing (ALVES, 2007, p. 1). Iniciamos nosso estudo apresentando a máquina de maneira formal e em seguida apresentamos exemplos de programas elaborados para a máquina Norma em dois tipos de apresentação: de forma iterativa e de forma monolítica.

Desta forma, iniciaremos com algoritmos bem simples escritos para a máquina Norma partindo para algoritmos mais sofisticados. Estaremos disponibilizando um simulador para que, opcionalmente, o aluno possa elaborar programas e testá-los.

## A máquina Norma

A máquina Norma (**N**umber The**O**retic **R**egister **MA**chine) é uma máquina de registradores proposta por Richard Bird em 1976 (Bird, 1976, p. 50), batizada assim em homenagem à sua esposa, Norma.

A máquina é um formalismo de Máquina Universal e uma característica interessante da máquina é sua distinção entre programa e máquina (DIVÉRIO; MENEZES, 2011, p. 70).

A memória da máquina Norma consiste de um número infinito (mas contável) de registradores naturais (normalmente denotados por letras do alfabeto maiúsculas: A, B, C, X, Y, etc...). Três instruções podem agir sobre cada registrador:

* Adicionar o valor corrente de um (incrementar 1)
* Subtrair o valor corrente de um (decrementar 1)
* Testar se o valor corrente do registrador é zero.

Conforme Divério e Menezes (2011) a máquina é definida formalmente como uma 7-upla definida como:

Norma = (N∞, N, N, ent, sai, {adK, subK}, {zeroK})

Onde:

1. N∞ denota o conjunto de todas as uplas com infinitos componentes sobre o conjunto de números naturais. Cada elemento do conjunto de valores de memória N∞ denota uma configuração possível dos seus registradores (A, B, X, Y, etc...).
2. O primeiro N denota o registrador X com o valor de entrada.
3. O segundo N denota o registrador Y com o valor da saída.
4. A função ent (função de entrada) carrega no registrador X o valor de entrada inicializando os demais registradores com 0. É definida como:

ent: N → N∞.

1. A função sai (função de saída) retorna o conteúdo do registrador Y e é definida como: sai: N → N∞.
2. O sexto elemento da 7-upla é um conjunto de interpretação de operações que contém as ações de incrementar (adK) e decrementar (subK) cada um dos K registradores. Desta forma adA adiciona 1 ao registrador A, adB adiciona 1 ao registrador B e assim por diante. Igualmente subA subtrai 1 do conteúdo do registrador A, subB subtrai 1 do registrador B e assim por diante. A família de operações adK é definida como adK: N∞ → N∞ e a família de operações subK é definida como subK: N∞ → N∞. Simplificadamente podemos denotar estas operações como: K := K+1
3. O sétimo elemento da tupla é um conjunto de interpretações de testes que consiste de uma família de testes associados a cada registrador K. Para cada registrador K é possível testar para zero obtendo um retorno verdadeiro ou falso. A família de operações zeroK é definida da seguinte forma: zeroK: N∞ → {verdadeiro, falso}.

Um exemplo de configuração para uma upla é (1, 1, 0, 0, 1, ...) outro exemplo é (5, 0, 0, 0, ...) e assim por diante. Uma upla contendo a seguinte configuração (5,0,0,0,...) para os registradores (A,B,X,Y,...) após a operação adB mudaria para a configuração (5,1,0,0,...). Após a operação subA, mudaria para a configuração (4,1,0,0,...) e assim por diante.

## 

## Formas de representação

Podemos representar as operações de diversas formas dependendo se estamos representando um programa monolítico ou iterativo e dependendo também se estamos utilizando algum programa simulador. Ilustrarei para cada tipo de operação, todas as formas de modo que o estudante pode usar qualquer uma delas indiferentemente. A Tabela 1 ilustra as formas de representação das operações dependendo se usamos um programa monolítico ou iterativo. O simulador SimNorma utiliza a notação monolítica ao passo que o simulador SimNorma-tool utiliza ambas as notações.

**Tabela 1 – Formas de representação de operações**

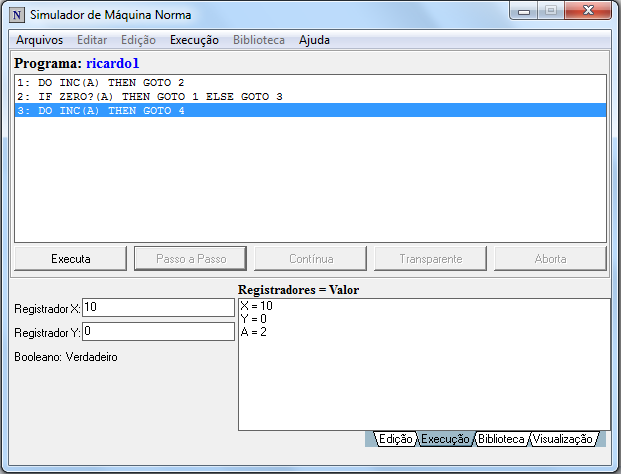
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Operação | Monolítico | Iterativo | SimNorma | SimNorma-tool |
| adA – Incrementa o registrador A | faça A:=A+1 vá para 90 | A := A+1 | DO INC(A) THEN GOTO 90 | faca ad-A va\_para 90  ou faça(ad-A) |
|  |  |  |  |  |
| subA–decrementa o registrador A | faça A:=A-1 vá para 90 | A := A-1 | DO DEC(A) THEN GOTO 90 | faca sub-A va\_para 90  ou faça(sub-A) |
|  |  |  |  |  |
| zeroA – Compara A com zero e toma uma ação para verdadeiro e outra para falso. | se A=0 então vá\_para 50 senão vá\_para 30 | Se A=0 então () senão () | IF ZERO?(A) THEN GOTO 50 ELSE GOTO 30 | se zero-A entao va\_para 50 senao va\_para 30  ou se zero-A entao () senão () |
|  |  |  |  |  |

**Fonte: próprio autor**

## Simuladores

Existem dois simuladores disponibilizados na internet para o estudante de teoria da computação. O uso dos simuladores não é obrigatório nesta matéria, mas pode ser útil. Os links estão disponibilizados como material complementar. A figura 1 ilustra o simulador SimNorma descrito em Divério et al. (2002). Observe o programa digitado. No simulador SimNorma, a sintaxe está em inglês. As instruções correspondem a um programa monolítico, logo no fim de cada instrução há um desvio para a próxima instrução a ser executada. As ações de incremento usam a sintaxe: “DO INC(K) THEN GOTO Nº”. As ações de decremento usam a sintaxe: “DO DEC(K) THEN GOTO Nº”. O teste de zero utiliza a sintaxe “zero?(K)”. O programa digitado é meramente ilustrativo e termina com o valor 2 no registrador A.

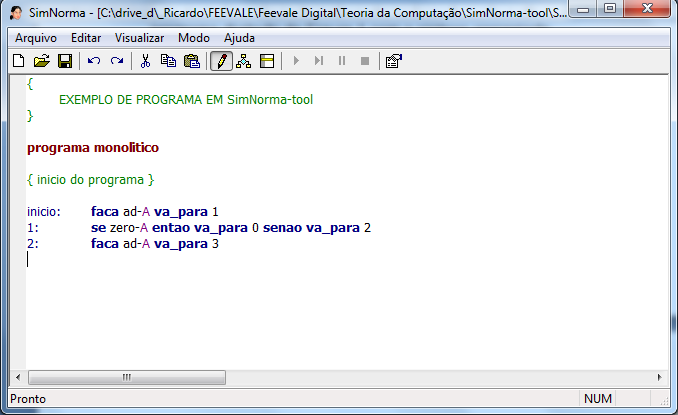
**Figura 1 – O simulador SimNorma**



**Fonte: próprio autor**

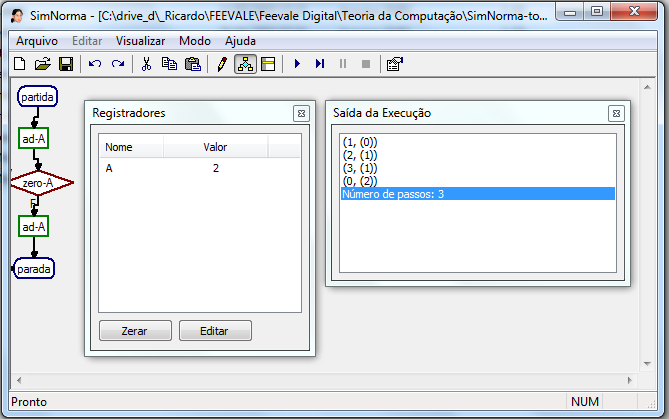
O simulador SimNorma-tool está ilustrado na Figura 2 com o mesmo programa digitado da Figura 1 mas com a sintaxe que lhe é peculiar. Observe que neste simulador a sintaxe está em português. Observe também que neste simulador é preciso determinar se estamos criando um programa em um sistema monolítico ou em um sistema iterativo. A Figura 3 ilustra a saída deste programa. Observe que no final após três passos, o algoritmo termina na instrução 0 com o valor 2 no registrador A. Isto está representado da seguinte maneira: (0, (2)).

**Figura 2 – O simulador SimNorma-tool**



**Fonte: próprio autor**

**Figura 3 – Saída de execução no simulador SimNorma-tool**



**Fonte: próprio autor**

## 

## Exemplos de programas

Estaremos agora ilustrando alguns programas em máquina Norma. Para cada exemplo ilustramos na forma monolítica e na forma iterativa.

**Exemplo 1**. Neste exemplo não sabemos o conteúdo do registrador “A” e desejamos zerar o registrador. Tudo o que temos que fazer é ir decrementando o registrador e testando até que este fique com 0. Quando terminamos, vamos para 0 de modo a parar o programa.

Forma monolítica:

|  |
| --- |
| 1: se A=0 então vá\_para 0 senão vá\_para 2  2: faça A:=A-1 vá\_para 1 |

Forma iterativa:

|  |
| --- |
| até A=0  faça (A:=A-1); |

**Exemplo 2**. Neste segundo exemplo desejamos atribuir o valor 3 para o registrador “A”. Neste caso, precisamos primeiro zerar o conteúdo de “A” e depois incrementar 3 vezes o registrador “A”.

Forma monolítica:

|  |
| --- |
| 1: se A=0 então vá\_para 3 senão vá\_para 2  2: faça A:=A-1 vá\_para 1  3: faça A:=A+1 vá\_para 4  4: faça A:=A+1 vá\_para 5  5: faça A:=A+1 vá\_para 0 |

Forma iterativa:

|  |
| --- |
| até A=0  faça (A:=A-1);  A:=A+1;  A:=A+1;  A:=A+1; |

**Exemplo 3**. Nosso próximo exemplo ilustra a operação de somar o conteúdo do registrador “B” no registrador “A”. Observe que, desta vez, não precisamos zerar o conteúdo do registrador “A”. Contudo, necessitamos zerar o conteúdo de um registrador auxiliar “C”. Este registrador é necessário para repor o valor do registrador “B” após somar este conteúdo ao registrador “A”.

Forma monolítica:

|  |
| --- |
| 1: se C=0 então vá\_para 3 senão vá\_para 2  2: faça C:=C-1 vá\_para 1  3: se B=0 então vá\_para 7 senão vá\_para 4  4: faça A:=A+1 vá\_para 5  5: faça B:=B-1 vá\_para 3  6: faça C:=C+1 vá\_para 3  7: se C=0 então vá\_para 0 senão vá\_para 8  8: faça B:=B+1 vá\_para 9  9: faça C:=C-1 vá\_para 7 |

Forma iterativa:

|  |
| --- |
| até C=0  faça (C:=C-1);  até B=0  faça (A:=A+1; B:=B-1; C:=C+1);  até C=0  faça (B:=B+1; C:=C-1); |

No próximo material “Mais exemplos em máquina Norma.docx” estaremos desenvolvendo exemplos mais complexos de programação em linguagem Norma. Até lá!

Referências

ALVES, Débora Pandolfi. **Equivalência de Máquinas Universais: Demonstração, Análise e Simulação**. 2007. Trabalho de Conclusão de Curso (Monografia) – Curso de Ciência da Computação, Unisinos, São Leopoldo, RS. Disponível em:<http://debora.wait4.org/tc.pdf>

BIRD, Richard Simpson. **Programs and Machines**: an introduction to the theory of computation. London: John Wiley, 1976.

DIVERIO, Tiarajú; MENEZES, Paulo Blauth. **Teoria da computação**: máquinas universais e computabilidade. Porto Alegre: Bookman, 2011. (Livros didáticos informática UFRGS 5) [Recurso Eletrônico/Biblioteca Virtual Universitária].

DIVERIO, Tiarajú et al. Simulators: Tools for Teaching Theory of Computation. In: WATSON, Deryn; ANDERSEN, Jane (eds). **Networking the learner**: computers in education. Copenhagen: IFIP/Springer, 2002. p. 483-493. Disponível em: <https://link.springer.com/content/pdf/10.1007%2F978-0-387-35596-2\_48.pdf>