5.4 Outras Definições das Máquinas de Turing

Podem-se imaginar numerosas variantes das Máquinas de Turing (MT). Definem-se aqui duas variações da definição de linguagens aceitas e decididas que não alteram a classe das linguagens aceitas e decidias pelas MT.

(1) Os estados de aceitação são substituídos por um único estado chamado *estado de parada* (*halt state)*; além disso, impõe-se que a função de transição seja definida em todas as situações. No *estado de parada* a MT aceita ou não a palavra de entrada em função de o início da fita conter ou não, respectivamente, "aceita" (1) ou "não aceita" (0).
   - ✓ A MT aceita a linguagem se, para toda palavra dela, sua execução atinge o estado de parada com o início da fita contendo *1*. Para as palavras que não pertencem à linguagem, a execução da MT pode ser infinita ou atingir o estado de parada com o início da fita contendo *0*.
   - ✓ A MT decide a linguagem se ela atinge sempre o estado de parada com o início da fita contendo *1* para as palavras da linguagem e *0* para as não pertencentes à linguagem.

(2) Dentre os estados da MT distinguem-se dois estados de parada q_Y e q_N e a função de transição é definida em todas as situações.
   - ✓ A MT aceita uma linguagem se, para todas suas palavras, a execução da MT atinge o estado de parada q_Y. Para as palavras não pertencentes à linguagem, a execução pode ser infinita ou atingir o estado de parada q_N.
      - ✓ A MT decide uma linguagem se ela atinge o estado de parada q_Y para todas suas palavras e o estado q_N para todas as palavras que não lhe pertencem.

5.5 Linguagens Recursivas e Recursivamente Enumeráveis

As linguagens respectivamente decididas e aceitas pelas MTs são chamadas *recursivas* e *recursivamente enumeráveis.*

Definições: Uma linguagem é *recursiva* se ela é decidida por uma MT. Uma linguagem é *recursivamente enumerável* se ela é aceita por uma MT.

5.6. Tese de Turing-Church

Enunciado: As linguagens reconhecidas por um procedimento efetivo são aquelas decididas por uma MT.

Essa tese (não é um teorema!) é chave nos estudos de calculabilidade, pois é a ferramenta complementar para formalizar a demonstração de que existem linguagens que não podem ser reconhecidas por um procedimento efetivo. Até o presente momento, um dos sentidos da bi-implicação da tese já está claro, visto que é evidente que uma linguagem decidida por uma MT o é também por um procedimento efetivo (pois é fácil simular uma MT à mão ou por meio de um computador – supondo-se que ele conte com uma memória que pode ser estendida indefinidamente). Contudo, é bem mais delicado provar o segundo sentido de tal vi-implicação, ou seja, que toda linguagem decidida por um procedimento efetivo também o é por uma MT. Para fazê-lo, o principal argumento invocado aqui é que toda modelização razoável da noção de procedimento efetivo coincide com a noção de linguagem decidida por uma MT.

5.6.1. Extensões das Máquinas de Turing

Examinam-se aqui algumas extensões da Máquina de Turing MT apresentada anteriormente e mostra-se que as linguagens aceitas ou decidias por essas extensões também o são por MT.

a) Máquina de Turing com fita infinita nos dois sentidos (MT-1), tal como mostrado na Figura 1:

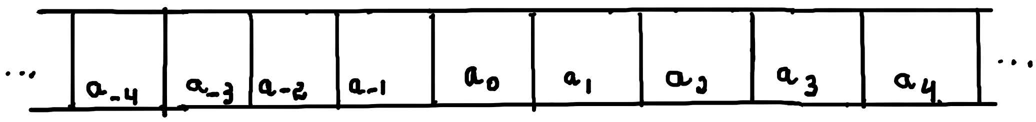


Figura 1: MT-1 com fita infinita à esquerda e à direita

Na Figura 1, os termos a_i são usados para referenciar posições genéricas da fita de MT-1, sendo que a_0 , particularmente, é reservada para representar a posição em que foi alocado o primeiro símbolo de uma palavra a ser processada por MT-1. Como em MT-1 uma palavra *w* a ser processada pode ser alocada em qualquer posição da fita, obviamente a_0 não se refere a uma posição fixa de MT-1. Assim sendo, considerando-se que o estado inicial de tal máquina é *s* e que a_0 referencia a posição da cabeça no momento em que começa o processamento de *w* (ou seja, apontando para o primeiro símbolo de *w),* na configuração inicial (s, α_1, w) de tal processamento, a palavra α_1 da fita alocada à esquerda da cabeça é irrelevante (lixo) ao processamento de *w* . Assim sendo, pode-se considerar que, na configuração inicial de MT-1, inexiste símbolo à esquerda da cabeça, ou seja, $\alpha_1 = \varepsilon$. Logo, tal configuração inicial de processamento pode ser expressa como (s, ε, w). Saliente-se que MT-1 difere da máquina MT finita à esquerda e infinita à direita EXCLUSIVAMENTE pelo fato de sempre poder executar transições à esquerda (ou seja, a restrição inerente às MTs de bloquear transições à esquerda a partir de qualquer configuração do tipo (q, ε, w) NÃO se aplica a MT-1).

A seguir, mostra-se que as linguagens aceitas ou decididas por MT-1 são exatamente as mesmas que são aceitas e decididas pela MT estudada anteriormente (fita finita à esquerda e infinita à direita). De fato, qualquer processamento efetuado pela fita infinita MT-1 pode ser efetuado por uma MT equivalente construída a partir de MT-1 da seguinte forma: efetua-se uma dobra em MT-1 de forma a produzir uma MT limitada à esquerda em que a_0 referencia a posição mais à esquerda de MT onde deve ser alocado o primeiro símbolo de uma palavra *w* a ser processada, conforme mostrado na Figura 2. Observa-se que cada posição de MT que armazenará um dos símbolos de *w* corresponde a uma casa composta por duas posições: uma superior e, outra, inferior. Assim sendo, caso se queira reproduzir em MT o processamento de uma palavra *w* processada por MT-1, seu primeiro símbolo deve ser alocado na casa mais à esquerda de MT e os demais nas casas subsequentes. Antecipe-se que o símbolo $ na parte inferior da MT será usado como marcador da casa mais à esquerda após o início do processamento de uma palavra qualquer.

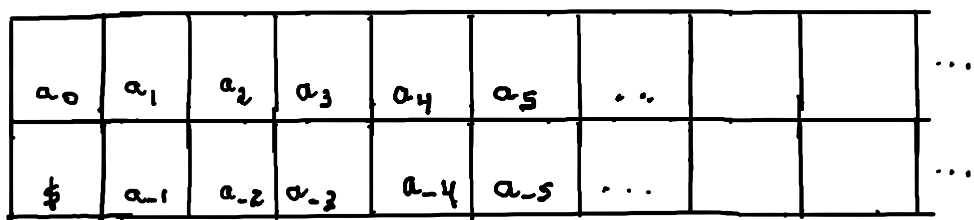


Figura 2: MT com fita finita à esquerda e infinita à direita

Dessa forma, cada casa da fita MT produzida contém não apenas um único símbolo, mas, sim, um par de símbolos (tais como os pares *(a_0, $)*, *($a_1$, a_{-1})*, ..., que ocupam cada casa da fita MT mostrada na Figura 2). Neste caso, para completar a demonstração, observa-se que um par de símbolos pode ser considerado como um símbolo único de um alfabeto maior. Além disso, é necessário adaptar a função de transição de MT de tal forma que ela simule, completamente, o comportamento de MT-1. Logo, supondo-se que:

MT-1 = (Q, Γ, Σ, δ, s , B, F), onde, $\Sigma \subseteq \Gamma$,

pode-se definir uma MT equivalente a MT-1 da seguinte maneira:

MT = (Q', Γ', Σ', δ', s' , B', F'), onde:

- Γ' = Γ X (Γ ∪ { $ }). Os símbolos do alfabeto Γ' da fita MT (ou seja, os símbolos que ocuparão cada uma das casas dessa fita) são pares ordenados cujos primeiros elementos são símbolos da fita MT-1 e cujos segundos elementos são elementos dessa mesma fita ou, então, *$*;
- Σ' = Σ X { B }. Os símbolos do alfabeto de entrada Σ' de MT, que, por definição, é um SUBCONJUNTO de seu alfabeto de fita Γ', correspondem aos pares ordenados específicos de Γ' cujos primeiros elementos são um símbolo de entrada de MT-1 e cujo segundo elementos é sempre o símbolo branco *B* (que pertence ao alfabeto Γ da fita MT-1). Assim sendo, cada símbolo de entrada de uma palavra a ser processada por MT corresponde a um desses pares do alfabeto de entrada Σ' que deverá ser alocado em uma dada casa da fita MT. Obviamente, no início desse processamento, MT deve estar no estado inicial e o primeiro símbolo *(x, B)* da palavra processada estará ocupando a primeira casa da fita MT;
- B' = (B, B). O símbolo branco de MT corresponde a um par formado pelo símbolo branco de MT-1;
- Q' = Q X {U, D} ∪ {s'}. Os estados de MT são os estados de MT-1 a qual se associam a informação *U (up)* ou *D (down)*, indicando se, em cada um desses estados, a cabeça de MT-1 está na parte direita ou esquerda de sua fita, respectivamente. Isso corresponde, em MT, a definir se deve-se considerar, respectivamente, a parte de cima ou de baixo do símbolo apontado por sua cabeça. O estado *s'* é um novo estado que servirá de estado inicial em MT. Convém relembrar, observando os exemplos da definição de MT estudados anteriormente, que cada estado NÃO É ASSOCIADO A UMA POSIÇÃO ESPECÍFICA DA CABEÇA, uma vez que, ao longo do processamento de uma palavra, a cabeça de uma MT pode apontar para uma mesma casa estando em estados diferentes, bem como pode apontar para casas diferentes estando em um mesmo estado;
- F' = F X {U, D}. Os estados de aceitação de MT correspondem aos estados de aceitação de MT-1 associados a *U* ou a *D*, dependendo de se localizarem à esquerda ou à direita da fita MT-1.

OBSERVAÇÃO: com base no disposto acima, conclui-se que:

- Uma palavra *w* processada a partir da configuração *(s, ε , w)* de MT-1, ao ser processada por MT, deverá ser alocada a partir da posição mais à esquerda da fita, de tal forma que a configuração inicial de tal processamento seja *(s', ε , w)* . Note que nessas configurações iniciais de ambas as máquinas, as casas que sucedem o símbolo de "branco" alocado à direita de *w* podem estar ocupadas pelo próprio símbolo "branco" ou podem estar vazias (desprovidas de símbolos), sendo seu conteúdo irrelevante ao processamento de *w.*
- A partir desses tópicos definidos acima, a título de exemplo, a configuração inicial do processamento da palavra *cdfh* pela máquina MT-1 apresentada na Figura 3 corresponde à configuração inicial de MT apresentada na Figura 4:

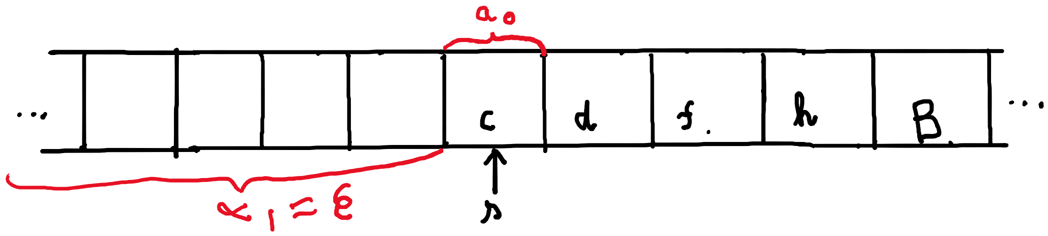


Figura 3: Configuração Inicial de MT-1 no processamento de *cdfh.*

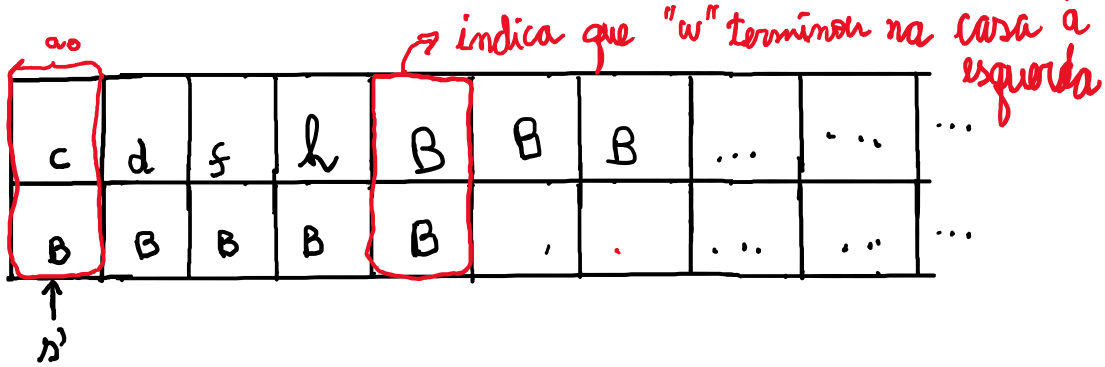


Figura 4: Configuração Inicial de MT no processamento de *cdfh*

Retomando e finalizando o processo de conversão de MT-1 em sua MT equivalente, falta definir agora a função de transição MT.

- Definindo a função de transição de MT correspondente àquela de MT-1 a partir das respectivas configurações iniciais:

    - ✓ Transições a partir do estado inicial s’, estando MT na posição mais à esquerda da fita e na configuração inicial:

Considerem-se as seguintes transições à direita e à esquerda de MT-1 a partir do estado inicial *s* ilustradas na Figura 5 (em vermelho e lilás, respectivamente):

δ(s,x) = (q, y, R)

δ(s,x) = (q, y, L)

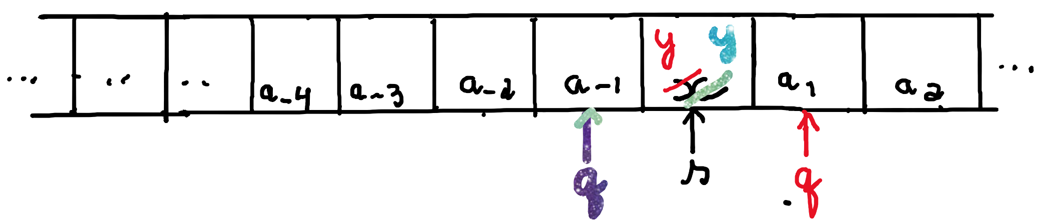


Figura 5: Transições MT-1 a Partir do Estado Inicial

Tais transições MT-1 equivalem, respectivamente, às seguintes transições MT a partir do seu estado inicial *s’*, AMBAS À DIREITA (pois MT é finita à esquerda), mostradas na Figura 6:

δ’(s’, (x,B)) = ((q, U), (y, $), R)

δ’(s’, (x,B)) = ((q, D), (y, $), R)

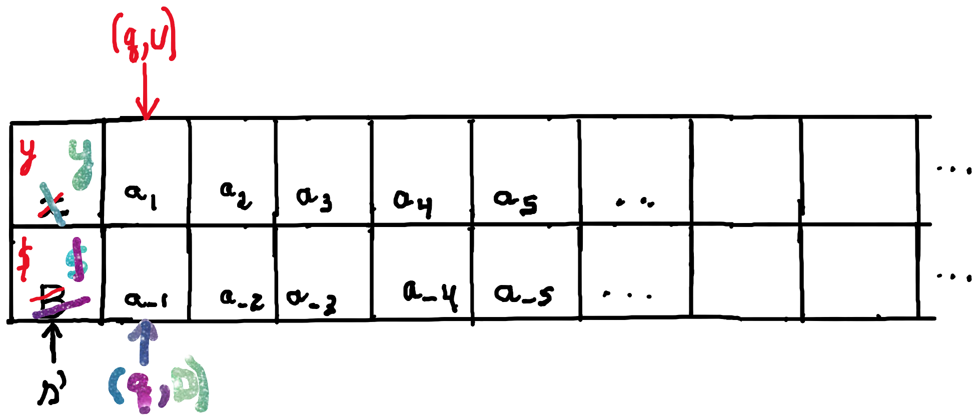


Figura 6: Transições à direita em MT, a partir da Configuração Inicial, que correspondem, respectivamente, às transições à esquerda e à direita de MT-1 a partir da configuração inicial

<u>OBS</u>: Note que, no processamento de uma palavra *w* por MT, $ é inserido na parte inferior mais à esquerda da fita após o processamento do primeiro símbolo de *w* (a partir do estado inicial), de forma a marcar a posição mais à esquerda da fita. A partir daí, *$* deverá ser mantido nesta mesma posição ao longo das demais transições.

- Definindo a função de transição de MT em que ela não está no início da fita:

  - ✓ Para o caso das transições à direita em MT-1 a partir de qualquer estado *q* em que a cabeça esteja à direita de a_0 (o que corresponde a um estado *(q, U)* em MT)- (Figura 7):

δ(q, x) = (r, y, R)

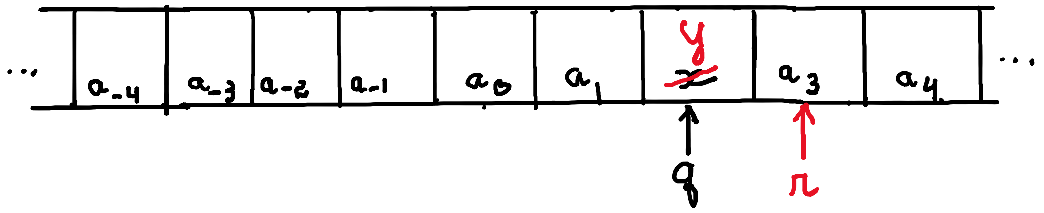


Figura 7: Transições à direita em MT-1, estando a máquina à direita de a_0

Tal transição MT-1 equivale à seguinte transição MT a partir do estado *(q, U)* mostrada na Figura 8:

δ'((q,U), (x, t≠\$))= ((r, U), (y, t), R).

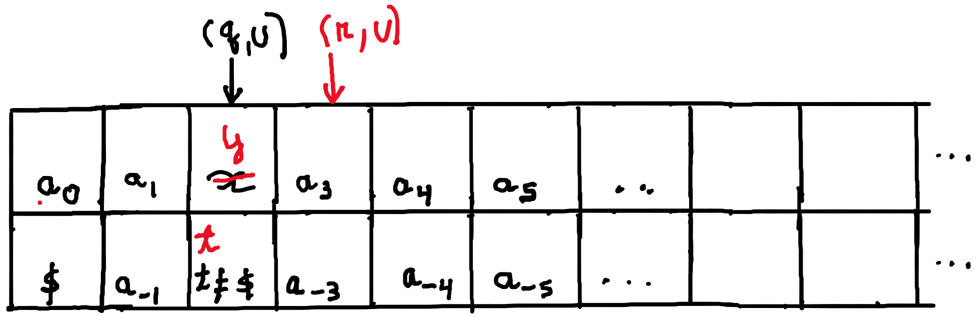


Figura 8: Transições à direita em MT, para transições à direita de MT-1 em que esta última está à direita de a_0

- ✓ Para o caso das transições à esquerda em MT-1 a partir de qualquer estado *q* em que a cabeça esteja à direita de a_0 (o que corresponde a um estado *(q, U)* em MT) - (Figura 9):

δ(q, x) = (r, y, L)

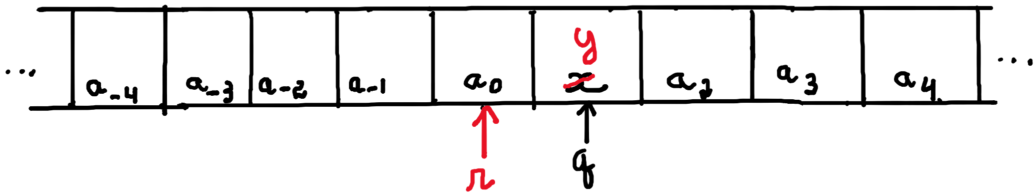


Figura 9: Transições à esquerda em MT-1, estando a máquina à direita de a_0

Tal transição MT-1 equivale à seguinte transição MT a partir do estado *(q, U)* mostrada na Figura 10:

δ'((q,U), (x, t≠$))= ((r, U), (y, t), L).

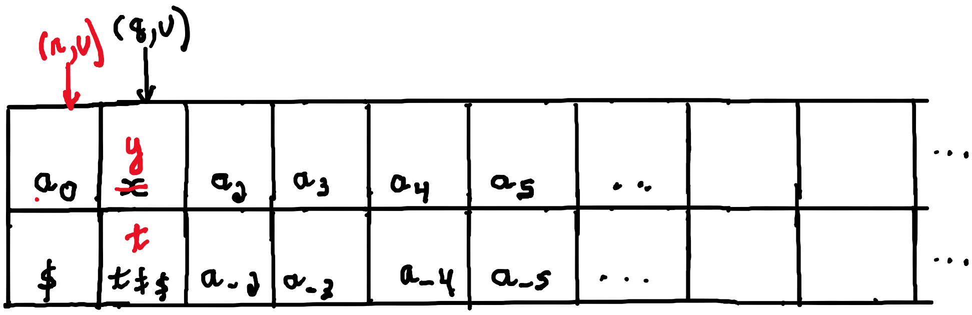


Figura 10: Transições à esquerda em MT , para transições à esquerda de MT-1, estando esta última à direita de a_0

- ✓ Para o caso de transições à esquerda em MT-1 a partir de qualquer estado *q* em que a cabeça esteja à esquerda de a_0 (o que corresponde a um estado *(q, D)* em MT) - Figura 11):

δ(q, t) = (r, y, L)

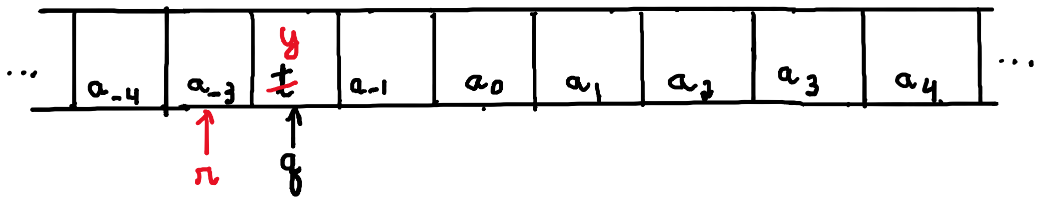


Figura 11: Transições à esquerda em MT-1 estando a máquina à esquerda de a_0

Tal transição MT-1 equivale à seguinte transição MT a partir do estado *(q, U)* mostrada na Figura 12:

δ'((q,D), (x, t≠$))= ((r, D), (x, y), R).

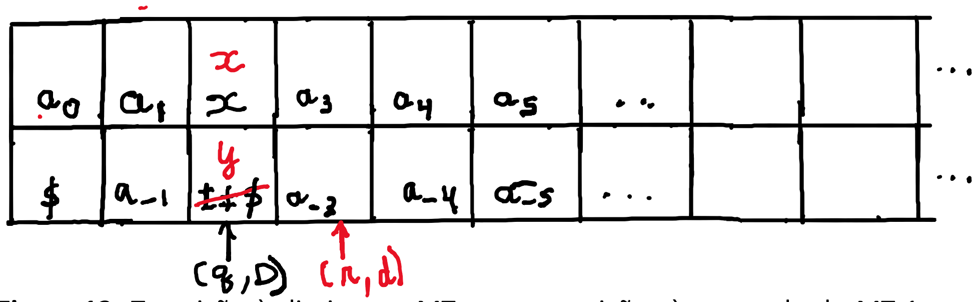


Figura 12: Transição à direita em MT, para transições à esquerda de MT-1 em que esta ultima está à esquerda de a_0

- ✓ Para o caso de transições à direita em MT-1 a partir de qualquer estado *q* em que a cabeça esteja à esquerda de a_0 (o que corresponde a um estado *(q, D)* em MT) - Figura 13):

δ(q, t) = (r, y, R)

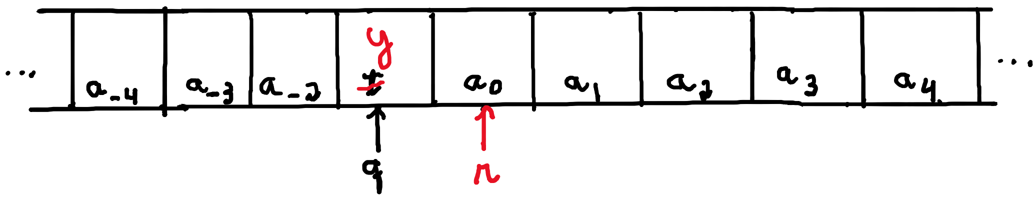


Figura 13: Transições à direita em MT-1, estando a máquina à esquerda de a_0

Tal transição MT-1 equivale à seguinte transição MT a partir do estado *(q, U)* mostrada na Figura 14:

δ'((q,D), (x, t≠$))= ((r, D), (x, y), L).

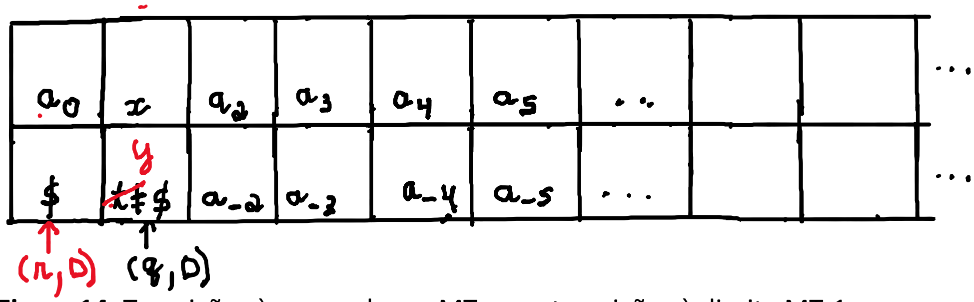


Figura 14: Transições à esquerda em MT, para transições à direita MT-1 em que esta ultima está à esquerda de a_0

- Definindo a função de transição de MT em que ela está no início da fita, sem estar na configuração inicial:

  - ✓ Para o caso das transições à direita em MT-1 a partir de qualquer estado *q* em que a cabeça esteja em a_0 e MT-1 não esteja na configuração inicial (o que corresponde ao estado *(q, U)* mais à esquerda de MT)- (Figura 15):

$\delta(q, x) = (r, y, R)$

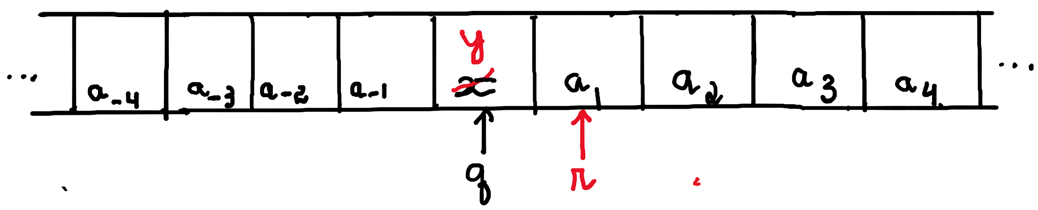


Figura 15: Transição à direita em MT-1, estando a máquina em a_0 (sem estar na configuração inicial).

Tal transição MT-1 equivale à seguinte transição à direita em MT a partir do estado *(q, U)* mostrada na Figura 16:

$\delta'((q,U), (x, \$))= ((r, U), (y, \$), R).$

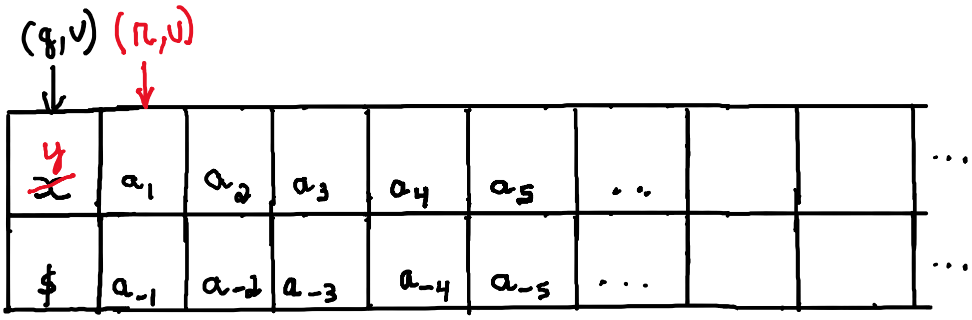


Figura 16: Transições à direita em MT, para transição à direita de MT-1, estando esta última em a_0 (sem estar na configuração inicial).

- ✓ Para o caso das transições à esquerda em MT-1 a partir de qualquer estado q em que a cabeça esteja em a_0 , MT-1 não esteja na configuração inicial e o símbolo apontado de MT é o da parte superior (o que corresponde ao estado *(q, U)* mais à esquerda de MT)- (Figura 17):

δ(q, x) = (r, y, L)

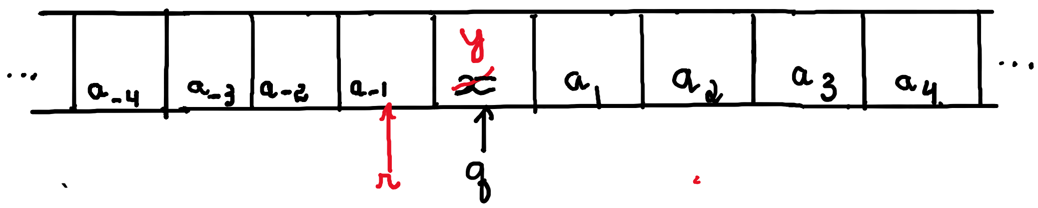


Figura 17: Transição à esquerda em MT-1, estando a máquina na casa da dobra (a_0)

Tal transição MT-1 equivale à seguinte transição à direita de MT a partir do estado *(q, U)* mostrada na Figura 18:

δ'((q,U), (x, \$))= ((r, D), (y, \$), R).

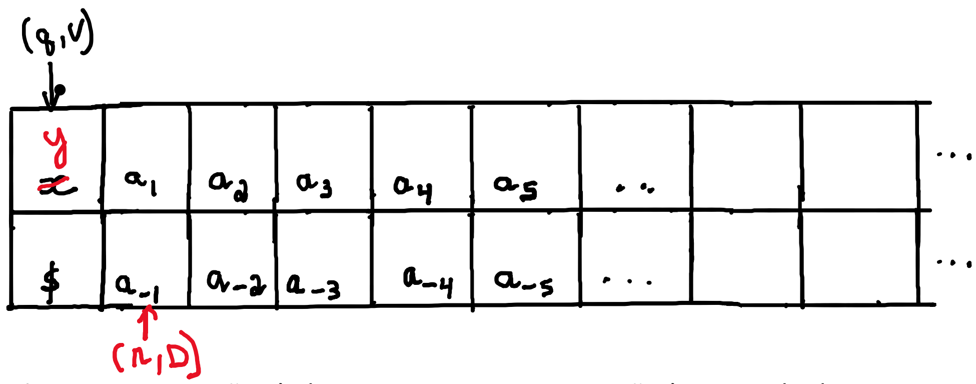


Figura 18: Transições à direita em MT, para transição à esquerda de MT-1, estando esta última em a_0 (sem estar na configuração inicial).

✓ Para o caso das transições à esquerda em MT-1 a partir de qualquer estado *q* em que a cabeça esteja em a_0 , MT-1 não esteja na configuração inicial e o símbolo apontado de MT é o da parte inferior (o que corresponde ao estado *(q, D)* mais à esquerda de MT)- (Figura 19):

$\delta(q, t) = (r, y, L)$

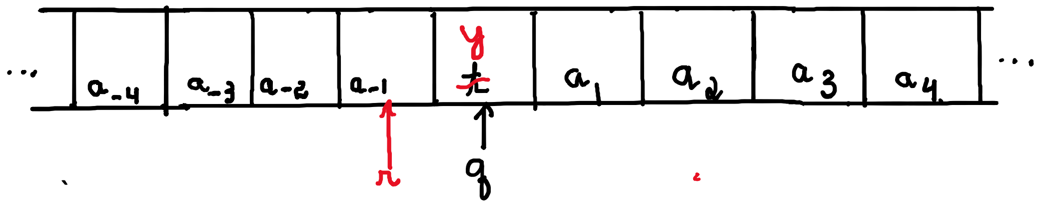


Figura 19: Transição à esquerda em MT-1, estando a máquina em a_0 (sem estar na configuração inicial).

Tal transição à esquerda de MT-1 equivale à seguinte transição à direita de MT a partir do estado *(q, D)* mostrada na Figura 20:

$\delta'((q,D), (x, \$))= ((r, D), (y, \$), R)$.

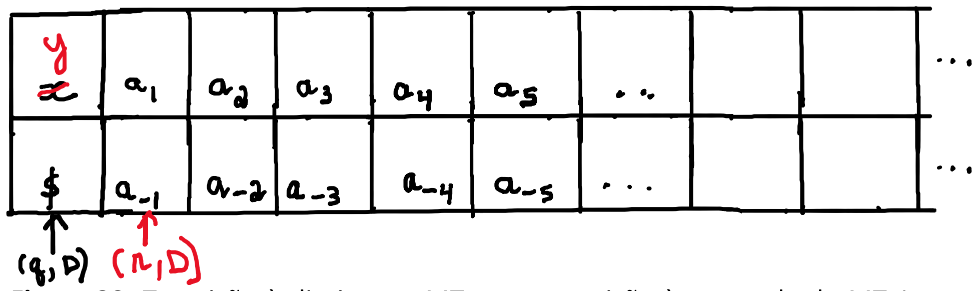


Figura 20: Transição à direita em MT, para transição à esquerda de MT-1, estando esta última em a_0 (sem estar na configuração inicial).

<u>OBS</u>: note que nas transições de MT a partir do estado *(q, D)* em que a cabeça está apontando para a parte inferior da primeira casa da fita e MT não está na configuração inicial, apesar de a cabeça estar apontando para o $, o símbolo substituído é *x* (que ocupa a posição superior da casa). Isso ocorre porque, na verdade, a cabeça de MT-1 está apontando para o símbolo que ocupa a posição a_0 , que corresponde ao símbolo *x* em MT ($ é um símbolo especifico de MT, usado apenas como um marcador de sua fita, que nunca pode ser substituído depois que é alocado na parte inferior da primeira casa, na primeira transição efetuada a partir da configuração inicial de MT).

✓ Para o caso das transições à direita em MT-1 a partir de qualquer estado *q* em que a cabeça esteja em a_0 e MT-1 não esteja na configuração inicial (o que corresponde ao estado *(q, D)* mais à esquerda de MT)- (Figura 21):

δ(q, x) = (r, y, R)

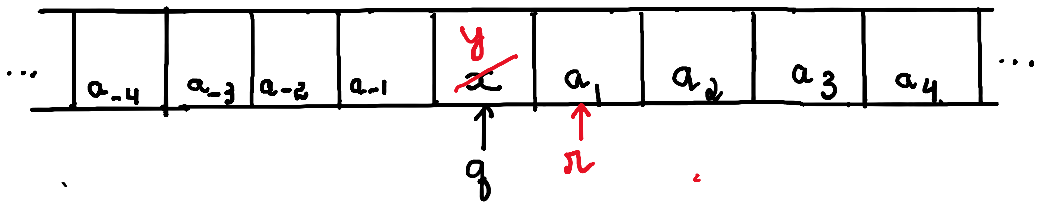


Figura 21: Transição à direita em MT-1, estando a máquina em a_0 sem estar na configuração inicial).

Tal transição à direita de MT-1 equivale à seguinte transição à direita de MT a partir do estado *(q, D)* mostrada na Figura 22:

δ'((q,D), (x, $))= ((r, U), (y, $), R).

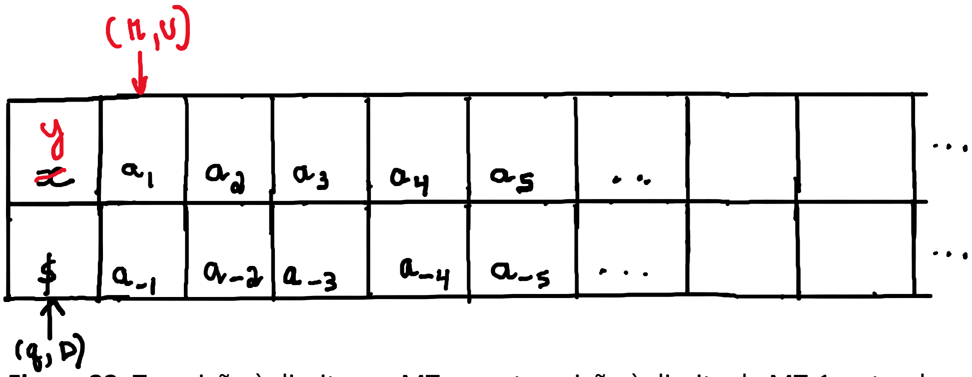


Figura 22: Transição à direita em MT, para transição à direita de MT-1, estando esta última em a_0 (sem estar na configuração inicial).

Finalmente, observa-se que o comportamento de MT-1 a partir de uma configuração inicial *(s , ε , w)* é o mesmo que o de MT a partir da configuração inicial *(s', ε , w'),* onde *w'* é a própria palavra *w* em que cada caracter *a* ∈ Σ é substituído por *(a, b)* ∈ Σ' . A partir dessa correspondência na primeira transição a partir da configuração inicial, por meio das transições estudadas acima, haverá correspondência entre todas as execuções de ambas as máquinas.

b) Múltiplas Fitas

Consideram-se aqui arquiteturas de Máquinas de Turing dispondo de várias fitas finitas à esquerda e infinita à direita, cada uma com sua respectiva cabeça de leitura, tal como ilustra a Figura 23 para uma MT-2 composta por duas fitas:

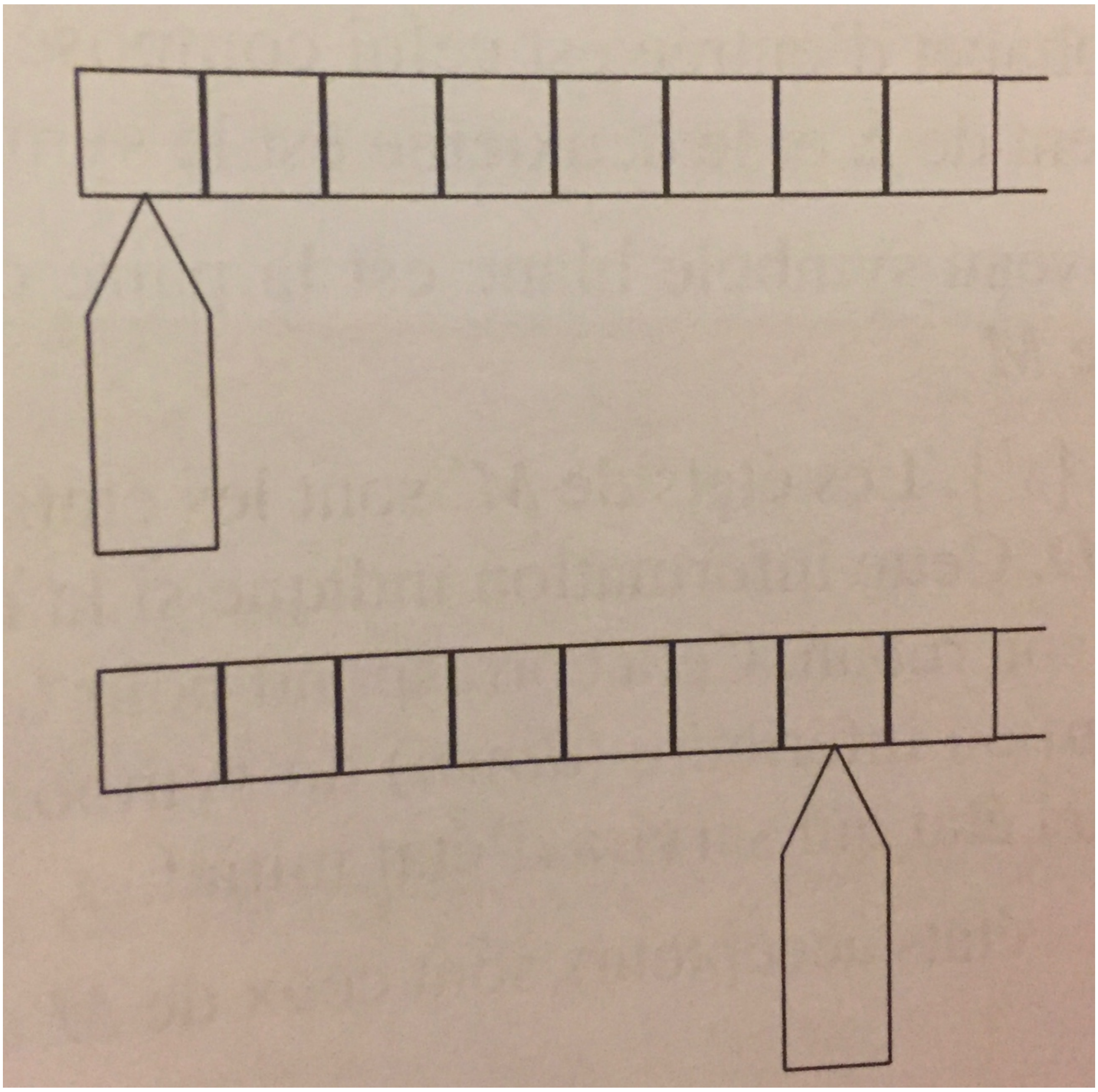


Figura 23- MT-2 : duas fitas finitas à esquerda e duas cabeças de leitura

É importante se salientar que uma transição em MT-2 incide sobre suas duas fitas, simultaneamente, ou seja, um único passo de execução de MT-2 desloca ambas as cabeças de leitura independentemente (mesmo em sentidos opostos) e pode alterar o conteúdo de ambas as posições apontadas por essas cabeças (com quaisquer dois valores do alfabeto da fita) na configuração a partir da qual é efetuado o disparo da transição. O mapeamento que representa as transições de MT-2 é:

δ: $Q\mathrm{x}\Gamma\mathrm{x}\Gamma \rightarrow Q\mathrm{x}\Gamma\mathrm{x}\Gamma\mathrm{x}\{L,R\}\mathrm{x}\{L,R\}$.

Analogamente ao discutido no item anterior, MT-2 pode ser convertida em uma MT cujo alfabeto de fita é um conjunto de quádruplas. Dois dos elementos de cada quádrupla são utilizados para representar o conteúdo de cada fita, ao passo que os dois remanescentes são usados para representar a posição de cada cabeça de leitura (por exemplo, tais elementos podem valer *um* nas casas em que se encontram as cabeças e *zero* nas demais). Assim sendo, cada casa da MT equivalente conterá 4 posições. Por exemplo, a primeira casa mais à esquerda de MT deverá conter, a partir de sua posição mais alta (superior), os seguintes elementos, respectivamente: conteúdo da casa mais à esquerda da primeira fita de MT-2; conteúdo da casa mais à esquerda da segunda fita de MT-2; *X* (valendo *0* ou *1*, respectivamente, em função de a cabeça da primeira fita de MT-2 *não estar* ou *estar* na sua casa mais à esquerda); e *Y* (valendo *0* ou *1*, respectivamente, em função de a cabeça da segunda fita de MT-2 *não estar* ou *estar* na sua casa mais à esquerda).

A fim de simular uma etapa de execução (uma transição) de MT-2 em MT deve-se:

- Encontrar a posição das cabeças de leitura e determinar os símbolos lidos;
- Modificar esses símbolos, deslocar as cabeças de leitura e mudar de estado.

Apesar de a construção exata de tal MT ser significativamente complexa, a ideia aqui é mostrar sua factibilidade.

Observação sobre a complexidade da TM: Demonstra-se que para simular n transições de uma MT de várias fitas em uma MT de uma única fita, precisaríamos, no máximo, de n^2 transições, o que corresponde a uma complexidade polinomial e não exponencial (exponencial seria, por exemplo, 2^n). Obviamente, é enorme a discrepância entre ambos os tipos de complexidade (de fato, por exemplo, se n=10, então n^2=100, enquanto que 2^10= ?). Veremos na parte em que será estudada a complexidade dos problemas que, apesar de o processamento paralelo prover ganhos significativos de tempo no caso de problemas polinomiais, ele é inócuo no caso dos problemas exponenciais, uma vez que ele não quebra a complexidade exponencial ! Na prática isso significa que um problema exponencial com elevado valor de *n* não será tratável nem mesmo com o uso de vários processadores em paralelo !

5.6.2. Máquinas a Memória de Acesso Direto

Mostra-se aqui que um computador real pode ser simulado por uma Máquina de Turing. Suponha-se que a máquina a ser simulada comporte uma memória de acesso direto (Random Access Memory- RAM) e um certo número de registradores, incluindo um contador de programa (*Programa Counter* – PC). Tal máquina pode ser simulada por uma MT de múltiplas fitas, em que uma delas é utilizada para a memória e, as demais, para os registradores, conforme Figura 24:

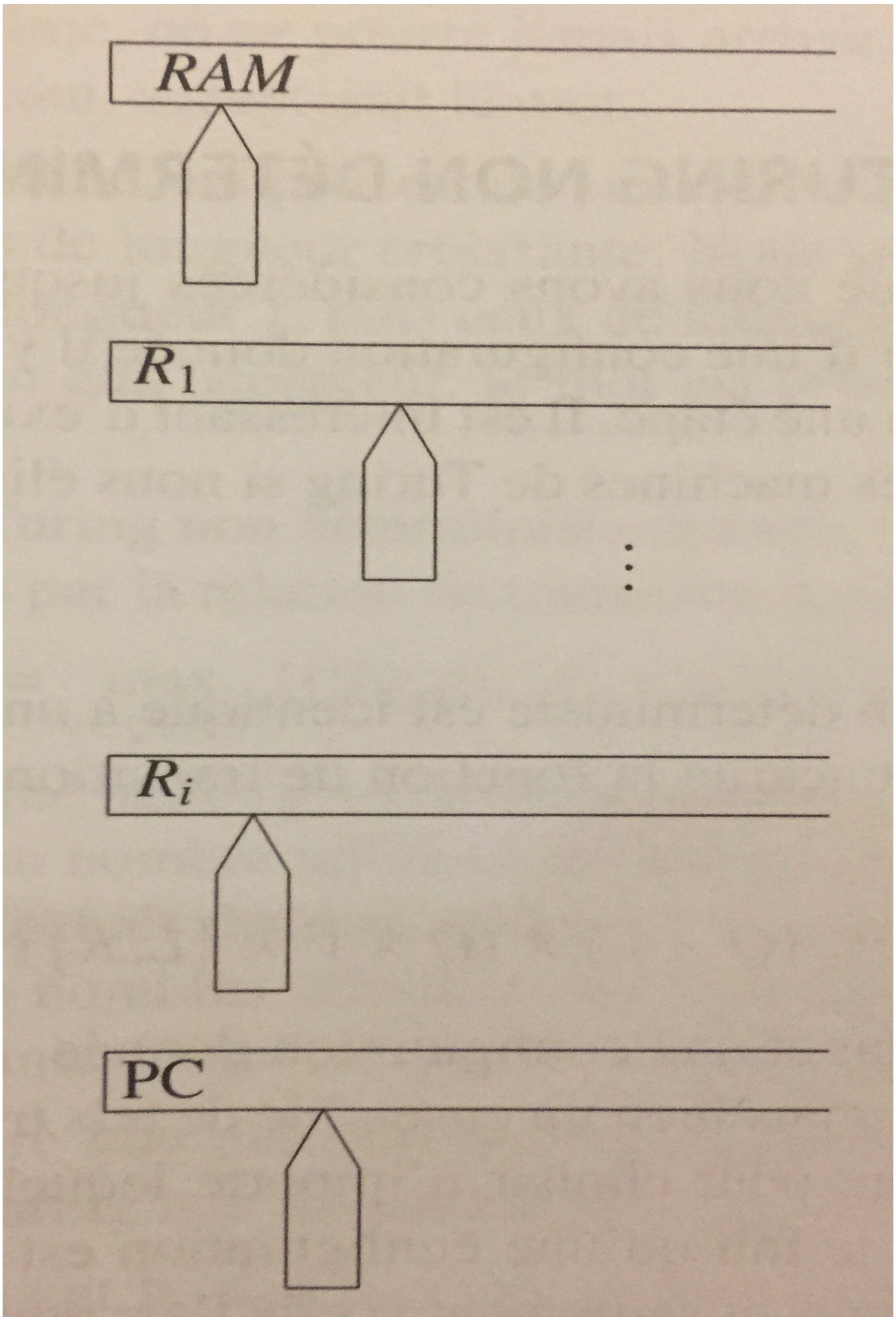


Figura 24- MT a múltiplas fitas simulando um computador real

O conteúdo da memória é representado por pares de palavras do tipo *(endereço, conteúdo)*, onde os elementos de cada par são separados pelo símbolo * e os pares sucessivos são separados pelo símbolo #, conforme Figura 25:

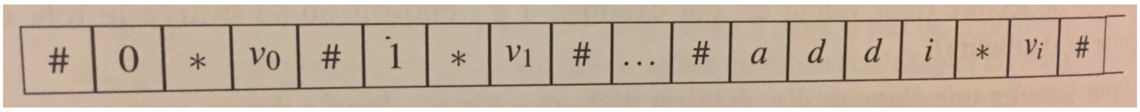


Figura 25- Fita representando o conteúdo da fita RAM da MT simuladora

A fim de simular a máquina à memória RAM, a MT simuladora deve repetir o seguinte ciclo:

(1) Percorrer a fita RAM até encontrar o endereço correspondente ao conteúdo do Contador de Programa (PC);
(2) Ler e decodificar a instrução presente em tal endereço;
(3) Caso pertinente, encontrar os operandos de tal instrução;
(4) Executar a instrução, o que eventualmente implica na modificação da memória e/ou dos registradores;
(5) Incrementar o PC (exceto se for uma instrução de apagamento) e avançar para o próximo ciclo.

OBSERVAÇÕES SOBRE A COMPLEXIDADE DAS MTs:

- É provado que para simular *n* movimentos de uma MT M_1 de várias fitas em uma MT *M* de *uma* única fita, necessitar-se-ia, no máximo, de n^2 movimentos. Logo, um processamento executado em complexidade polinomial em M_1 também poderá ser executado em tempo polinomial em *M* (complexidade exponencial seria, por exemplo, 2^n);
- Observe que é enorme a diferença entre complexidade polinomial e exponencial: por exemplo, se $n = 10$, $n^2 = 10$, enquanto que $2^n =$ **? ?**
- No capítulo em que se estudará a complexidade de problemas, será visto que o processamento paralelo, apesar de conseguir reduzir significativamente o tempo de execução de problemas com complexidade polinomial, não consegue converter a complexidade de problemas exponenciais em complexidade polinomial. Na prática, isso significa que um problema exponencial com elevado valor de *n* não é tratável nem mesmo com o uso de diversos processadores.

5.7. Máquinas de Turing Não Determinísticas

As MTs estudadas até agora são determinísticas no sentido de jamais permitir mais de uma transição a partir de uma mesma configuração. O objetivo desta seção é mostrar que MT não determinísticas têm o mesmo potencial de processamento das MT determinísticas estudadas.

5.7.1. Definição

Uma MT não determinística difere da MT determinística estudada unicamente pelo seguinte fato: na não determinística, as transições são definidas por meio de *relações de transição* e, não, por *funções de transição*, como ocorre nas determinísticas. Uma relação de transição é definida da seguinte forma:

$\Delta \subseteq (Q \times \Gamma) \times (Q \times \Gamma \times \{L, R\},$

Que permite definir, a partir de uma dada configuração, não apenas uma, mas um conjunto de triplas $(Q \times \Gamma \times \{L, R\}$. Logo, uma configuração é derivável em um único passo a partir de uma outra em que o estado é q e o símbolo lido é a se, e somente se, existe uma transição $((q, a), (q', x, X)) \in \Delta$ que leva até ela.

5.7.2. Eliminação do Não-determinismo

Teorema 5.1. *Toda linguagem aceita por uma MT não determinística o é também por uma MT determinística.*

O teorema se refere apenas a linguagens aceitas porque o conceito de linguagem decidida não faz sentido nas máquinas não determinísticas (uma vez que elas são concebidas para reconhecer as instâncias positivas de uma linguagem).

O Teorema 5.1 representa uma comprovação a mais da tese de Church-Turing, mostrando que o não determinismo não aumenta o poder computacional das TM **!**

Poder-se-ia pensar em demonstrar tal teorema de uma das seguintes maneiras:

Simulando na MT determinística todas as transições da MT não determinística. O problema, nesta alternativa é que essas transições não podem ser simuladas em paralelo, uma vez que essas simulações requerem o uso da fita, não podendo ser executadas simultaneamente. A título de esclarecimento, um exemplo de simulações paralelas ocorre na demonstração de que um autômato finito não determinístico (AFND) pode ser convertido em um determinístico (AFD) equivalente, tal como ilustra a Figura 26 mostrando a conversão de algumas transições entre tais autômatos (note que nem todas as transições do AFND estão representadas no AFD – de fato, distintamente do AFD, o AFND mostrado não aceita as palavras *$a^n b$):*

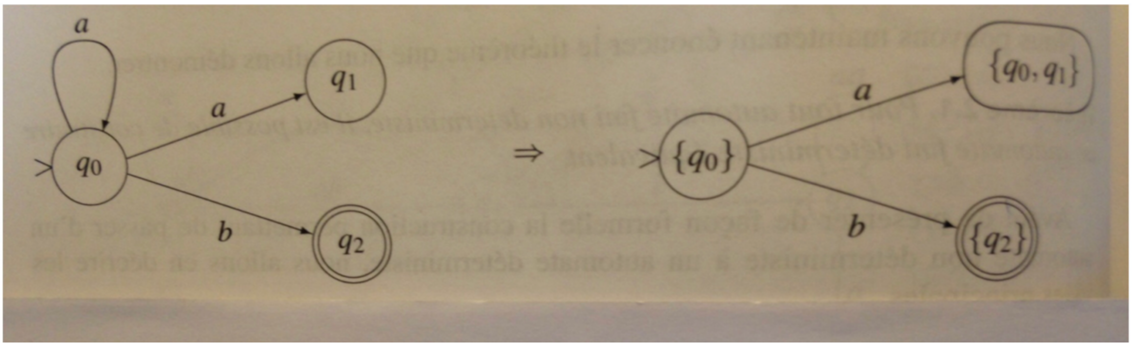


Figura 26- Simulando paralelamente transições de um AFND em um AFD

Por outro lado, poder-se-ia pensar em efetuar tais simulações serialmente. O problema, neste caso, é que, se por falta de sorte, se recair em uma simulação que entre em loop, nenhuma outra será simulada, incluindo aquela que, eventualmente, levaria à aceitação de uma palavra.

Uma outra maneira de se tentar demonstrar o teorema 5.1 (que é a adotada aqui) é a seguinte: simular todas as execuções, mas considerando seus prefixos de comprimento crescente (simular inicialmente todas as execuções de prefixo de tamanho 1, depois os de tamanho 2, na sequência, os de tamanho 3 etc). Tão logo as execuções cheguem a um estado de aceitação, a palavra é aceita, conforme formalizado a seguir:

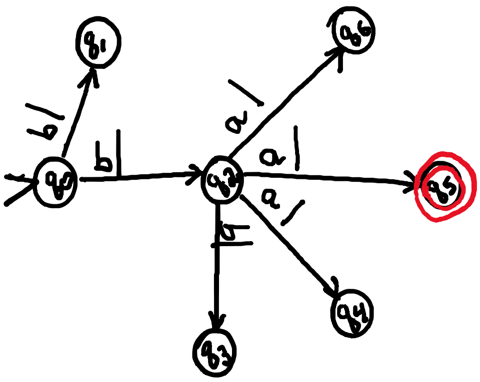
Para uma dada MT não determinística há um número máximo de escolhas possíveis disponibilizadas pela relação de transição (esse número máximo se refere a transições efetuadas a partir de um determinado par *(q, a)*). Tal número é mostrado na Figura 27:

$$r = \max_{q \in Q, a \in \Gamma} |\{((q,a),(q',x,X)) \in \Delta\}|.$$

Figura 27: número máximo de escolhas possíveis em uma MT não determinística

Para cada par *(q, a)* , numeremos as escolhas permitidas pela relação de transição de *1* ao número máximo de escolhas possíveis (sendo que tal número máximo varia entre *1* e *r*). A partir de então, para descrever as escolhas efetuadas em um prefixo de execução de comprimento *m*, basta dar uma sequência de *m* números no máximo iguais a *r* (se não há uma transição correspondente à escolha, a execução se interrompe sem aceitar o prefixo).

Exemplo: simulemos as transições em uma MT determinística das seguintes transições de uma MT não determinística:



- Numerando as escolhas permitidas para cada par genérico *(q,a)* :

$$(q_0, b) \begin{cases} q_0 \rightarrow q_1 & : \; 1 \\ q_0 \rightarrow q_2 & : \; 2 \end{cases}$$

$$(q_2, b) \begin{cases} q_2 \rightarrow q_3 & : \; 1 \end{cases}$$

$$(q_2, a) \begin{cases} q_2 \rightarrow q_4 & : \; 1 \\ q_2 \rightarrow q_5 & : \; 2 \\ q_2 \rightarrow q_6 & : \; 3 \end{cases}$$

- Calculando *r* : conforme Figura 27, $r = 3$ (relativo ao número de escolhas associadas ao par (q_2, a));

- Descrevendo as escolhas para o prefixo *b* (sequências de tamanho $m = 1$): sequências de *1* número no máximo igual a *3*:

  1 (correspondente à escolha $q_0 \rightarrow q_1$);
  2 (correspondente à escolha $q_0 \rightarrow q_2$);

- Descrevendo as escolhas para o prefixo *ba* (sequências de tamanho $m = 2$): sequências de *2* números no máximo igual a *3*:

  1,1 (sequência correspondente às escolhas $q_0 \rightarrow q_1$ e $q_1 \rightarrow ?$, respectivamente): EXECUÇÃO PARA SEM ACEITAR O PREFIXO;

  1,2 (sequência correspondente às escolhas $q_0 \rightarrow q_1$ e $q_1 \rightarrow ?$, respectivamente): EXECUÇÃO PARA SEM ACEITAR O PREFIXO;

  1,3 (sequência correspondente às escolhas $q_0 \rightarrow q_1$ e $q_1 \rightarrow ?$, respectivamente): EXECUÇÃO PARA SEM ACEITAR O PREFIXO;

  2,1 (sequência correspondente às escolhas $q_0 \rightarrow q_2$ e $q_2 \rightarrow q_4$, respectivamente): EXECUÇÃO TERMINA SEM ACEITAR O PREFIXO;

  2,2 (sequência correspondente às escolhas $q_0 \rightarrow q_2$ e $q_2 \rightarrow q_5$, respectivamente): EXECUÇÃO TERMINA ACEITANDO O PREFIXO;

  2,3 (sequência correspondente às escolhas $q_0 \rightarrow q_2$ e $q_2 \rightarrow q_6$, respectivamente): EXECUÇÃO TERMINA SEM ACEITAR O PREFIXO;

Assim sendo, pode-se construir uma MT determinística de 3 fitas que simula uma MT não determinística da seguinte forma:

(1) A primeira fita contém a palavra de entrada e não é modificada (permite recuperar a palavra de entrada a cada vez que recomeçamos uma execução);
(2) A segunda fita servirá para conter as sequências de números inferiores a *r;*
(3) A terceira fita da MT determinística será usada para simular o comportamento da MT não determinística;

Uma vez construída, a MT determinística procede como descrito abaixo:

(1) Ela gera na segunda fita, em ordem crescente de comprimento, todas as sequências finitas de números inferiores a *r*;
(2) Para cada uma dessas sequências, ela simula a MT não determinística (na terceira fita) utilizando as escolhas representadas pela sequência;

(3) Ela para e aceita assim que a simulação de uma execução da MT não determinística atinge um estado de aceitação.

Assim sendo, considerando o exemplo da MT não determinística apresentado há pouco, abaixo são apresentados alguns exemplos de como a MT determinística poderia simulá-la:

- No processamento da palavra *b*: a palavra *b* é alocada na primeira fita. Como a palavra tem comprimento *1,* são alocadas na segunda fita as duas sequências de tamanho *1* mostradas acima. Cada uma dessas duas sequências de tamanho *1* (correspondentes ao prefixo *b*) é processada em série na fita 3 (na ordem em que aparecem em tal fita). No início de cada um desses processamentos, o prefixo *b* é resgatado na fita 1 e colocado na fita 3. Obviamente, ambos os processamentos falham e a palavra *b* é rejeitada;
- No processamento da palavra *ba*: a palavra *ba* é alocada na primeira fita. Como a palavra tem tamanho *2*, as duas sequências de tamanho *1*, seguidas das seis sequências de tamanho *2* são alocadas na segunda fita. Inicialmente é efetuado o processamento das duas sequências de tamanho *1* (correspondentes ao prefixo *b*), em série, na fita 3. No início de cada um desses processamentos, o prefixo *b* é resgatado na fita 1 e colocado na fita 3. Como o prefixo *b* não foi aceito nesses processamentos, começa o processamento das sequências de comprimento *2* (relativas ao prefixo *ba*). No início de cada um deles, o prefixo *ba* é resgatado na fita 1 e colocado na fita 3. Ao final do processamento da quinta sequência de tamanho *2* (sequência *2,2* , correspondente às escolhas $q_0 \rightarrow q_2$ e $q_2 \rightarrow q_5$, respectivamente), o prefixo é aceito e a execução termina aceitando a palavra *ab* .

Conforme será visto posteriormente, apesar de as MT não determinísticas não serem interessantes como procedimentos efetivos (uma vez que podem apresentar execuções que rejeitam palavras pertencentes à linguagem por ela aceita), elas são necessárias para definir o conceito de problemas NP, NP-completo e NP-Hard❗

5.8. Máquinas de Turing Universais

Saliente-se que pode-se conceber uma MT universal M capaz de simular qualquer outra MT M’, ou seja, M recebe a descrição de M’ e uma palavra *w* e consegue simular a execução de *w* em M’. Em termos computacionais, M é um interpretador de MTs denominado MT Universal.

Na prática, podemos encontrar na internet numerosos simuladores de TM. Tais simuladores executam qualquer MT e são executados em um PC que também pode ser representado por uma máquina de 1 fita ❗ É claro que, na prática, tal máquina é complexa demais para ser modelada.

5.9. Funções Calculáveis por Uma Máquina de Turing

Além de aceitadores de uma linguagem, as MTs também podem ser usadas como calculadoras de funções. Para tanto, o argumento da função deve ser posto como palavra de entrada na fita e o valor calculado para a função deve estar escrito na fita após o término da execução.

Definição: Uma MT calcula uma função $f: \Sigma^* \rightarrow \Sigma^*$ se, para toda palavra de entrada *w*, ela para sempre em uma configuração onde $f(w)$ se encontra na fita.

Assim sendo, uma função é calculável por uma MT se existe uma MT que a calcule. Observe-se então que "decidir" uma linguagem consiste em calcular uma função cujo contradomínio é *{0, 1}*, o que permite re-enunciar a tese de Turing-Church como: *As funções calculáveis por um procedimento efetivo são as funções calculáveis por uma MT*.

Na prática, essa conceituação de funções calculáveis por meio de um procedimento efetivo lastreada nas funções calculáveis por uma MT não constitui a forma mais adequada de formalização de tal conceito. Por exemplo, outras conceituações de funções calculáveis por um procedimento efetivo lastreadas no cálculo de funções recursivas (como as funções μ-recursivas ou o λ-calculo) são mais adaptáveis aos modelos computacionais existentes do que essa lastreada nas funções calculáveis por uma MT. Contudo, é demonstrado que a tese de Church-Turing se mantém válida em todas essas formulações, ou seja, todas as funções calculáveis por meio de funções recursivas, o são também por uma MT.