Informática
PUC-Rio

Linguagens Formais e Autômatos (LFA)

Aula de 18/11/2013

Linguagens Recursivamente Enumeráveis, Complexidade (Custo) de Tempo/Espaço, Transdutores para exibir complexidade de Tempo/Espaço

Linguagens Recursivamente Enumeráveis

Se **LINGUAGEM RECURSIVA** é aquela para a qual existe uma Máquina de Turing que para quando reconhece uma cadeia que a ela pertence **e também** para quando não reconhece uma outra cadeia que não pertence,

uma **LINGUAGEM RECURSIVAMENTE ENUMERÁVEL** é aquela para a qual existe uma Máquina de Turing que para quando reconhece uma cadeia que a ela pertence, mas que pode parar ou não ao processar uma cadeia que a ela não pertence.

Linguagens Recursivas
são também chamadas de
LINGUAGENS DECIDÍVEIS.

sivamente Enumeráveis

Se **LINGUAGEM RECURSIVA** é aquela para a qual existe uma Máquina de Turing que para quando reconhece uma cadeia que a ela pertence **e também** para quando não reconhece uma outra cadeia que não pertence,

uma **LINGUAGEM RECURSIVAMENTE ENUMERÁVEL** é aquela para a qual existe uma Máquina de Turing que para quando reconhece uma cadeia que a ela pertence, mas que pode parar ou não ao processar uma cadeia que a ela não pertence.

Linguagens Irrestritas ou de Tipo 0

São "reconhecidas" por **Máquinas de Turing de fita ilimitada**; para toda cadeia pertencente à linguagem, a MT para e aceita-a.

São "definidas" (especificadas) por Gramáticas Irrestritas; para toda cadeia pertencente à linguagem, há uma árvore de derivação completa cuja raíz é o símbolo inicial da gramática e cujas folhas (concatenadas) correspondem à cadeia em questão.

Linguagens recursivas e recursivamente enumeráveis são linguagens irrestritas ou de tipo 0.

Linguagens recursivamente enumeráveis são INDECIDÍVEIS

Tendo em vista que é uma característica das linguagens recursivamente enumeráveis que a MT que as reconhece **pode não parar de computar** ao receber uma cadeia que não pertença a ela, elas são – estritamente falando – **indecidíveis**.

Em outras palavras, as MTs que as reconhecem podem necessitar de **tempo infinito** para "decidir" que determinada cadeia não pertence a elas.

Complexidade (Custo) de Tempo/Espaço em MT's

Complexidade de TEMPO (CT):

CT é o **número máximo de transições processadas por uma computação de MT** quando iniciada por uma cadeia de comprimento **n**, independentemente de a cadeia ser aceita ou não.

CT é expresso como uma **função de n**. Como se trata do número MÁXIMO de transições, trata-se sempre do **pior caso**. Nem todas as cadeias de mesmo comprimento requerem o mesmo número de transições para serem decididas..

Complexidade de ESPAÇO (CE):

CE é o **número** que expressa a **quantidade MÁXIMA de células** de MT utilizadas na computação de cadeias de entrada de comprimento **n para ler/gravar INFORMAÇÃO DE PROCESSAMENTO** (i.e. não são contabilizadas as células da fita utilizadas apenas para a entrada), independentemente de a cadeia ser aceita ou não. Trata-se também do **pior caso** e CE é expresso como uma **função de n**.

Usando TRANSDUTORES para gerar representações de complexidade / custo

O que são TRANSDUTORES (finitos)?

- De forma geral, transdutores são Máquinas de Turing com "fitas (adicionais) de saída".

- Os exemplos a seguir foram produzidos com o JFLAP 7.0, versão de 2009. Nesta versão, MT's multifitas admitiam por *default* 3 movimentações: R (direita), L (esquerda), S (parada).

  No JFLAP da última versão as "preferências" das Máquinas de Turing têm de ser alteradas para "S" funcionar como esperado.

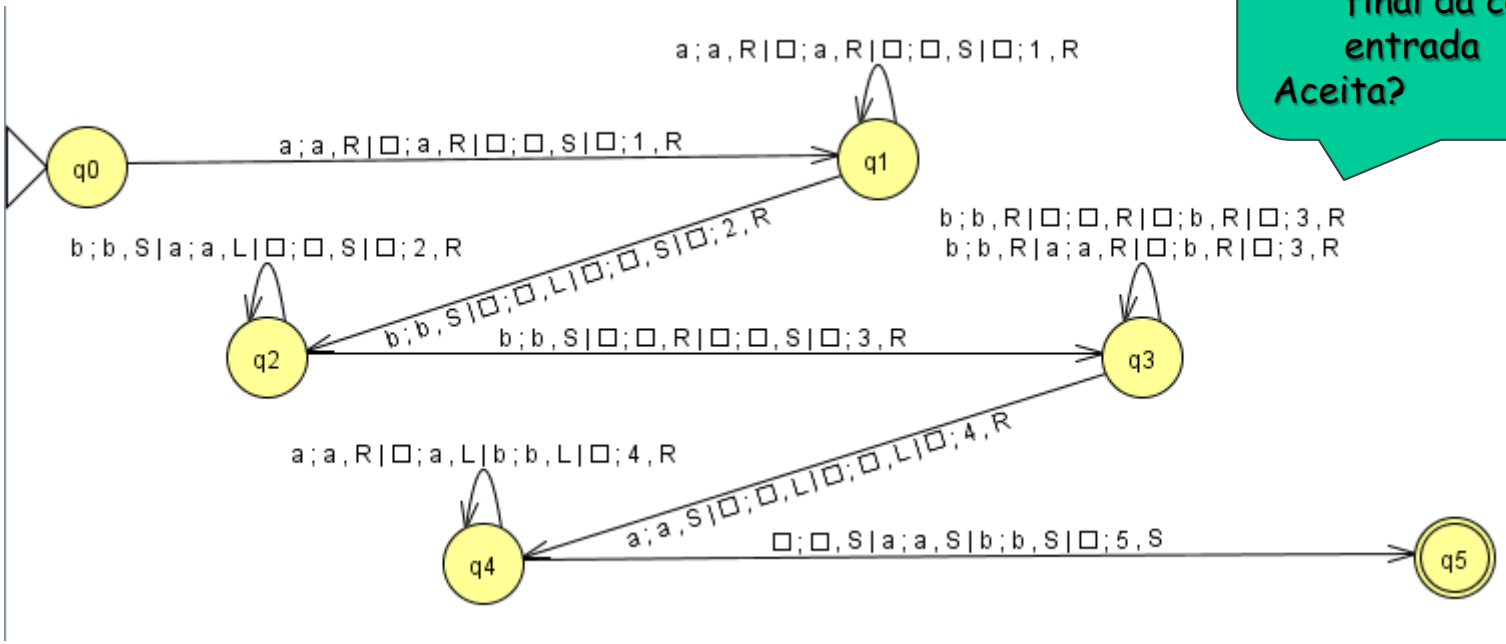
Complexidade de TEMPO

Seja a linguagem L definida sobre $\Sigma = \{a,b\}$; ela aceita cadeias $w = a^n b^m a^q$ para $n,m,q > 0$ e $m = n+q$.

Observe o comportamento de MT_{lenta} assim definida:

MT deve aceitar w se:
1) Para no estado final
2) O cursor está no final da cadeia de entrada

Aceita?

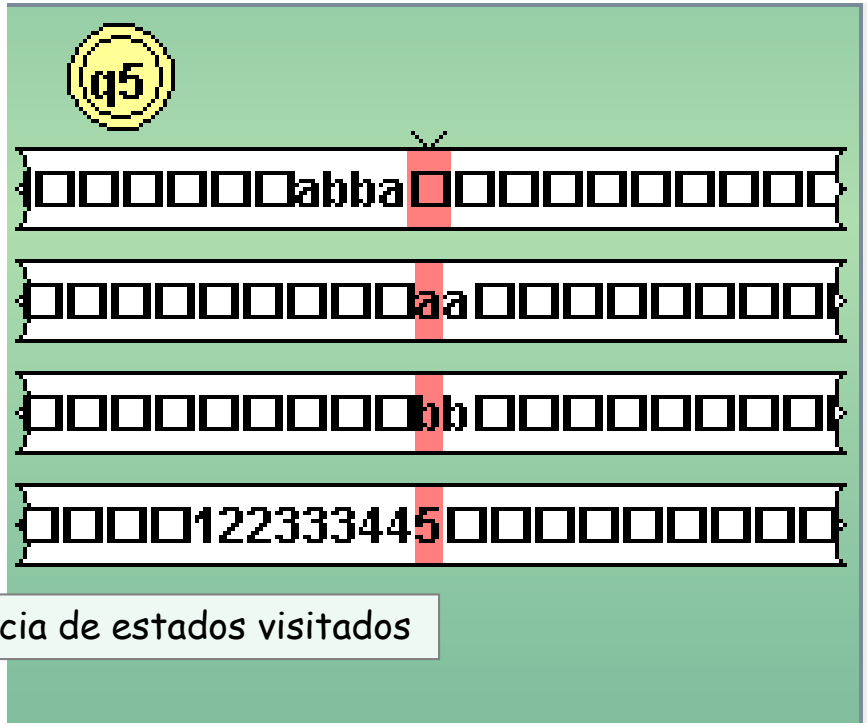


Rastros da Complexidade de Tempo de MT_{lenta}

Aceitação de abba

|abba| = 4 →

9 transições →



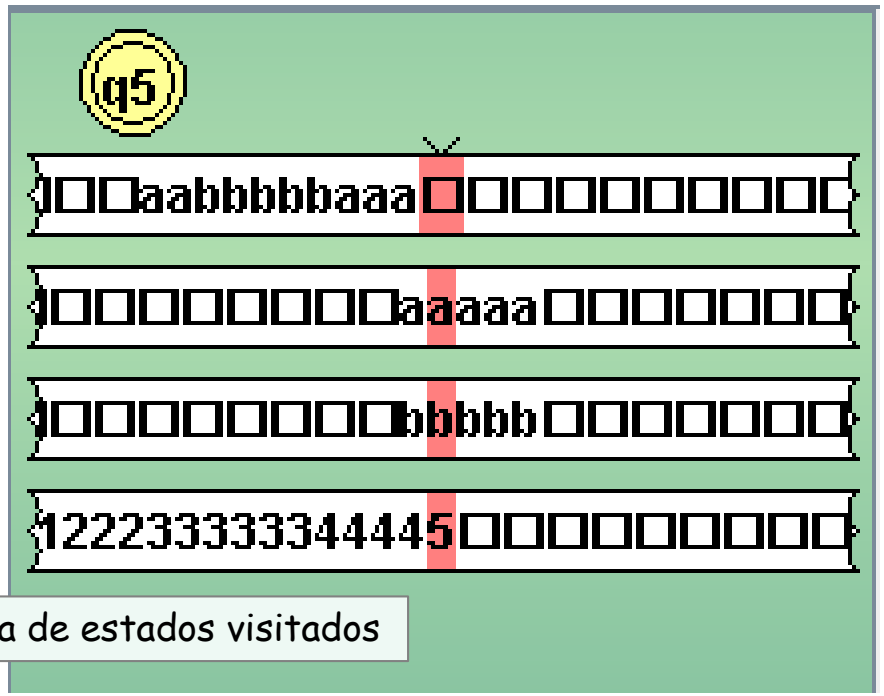
transdução com sequência de estados visitados

Rastros da Complexidade de Tempo de MT_{lenta}

Aceitação de aabbbbbaaa

|aabbbbbaaa| = 10

16 transições

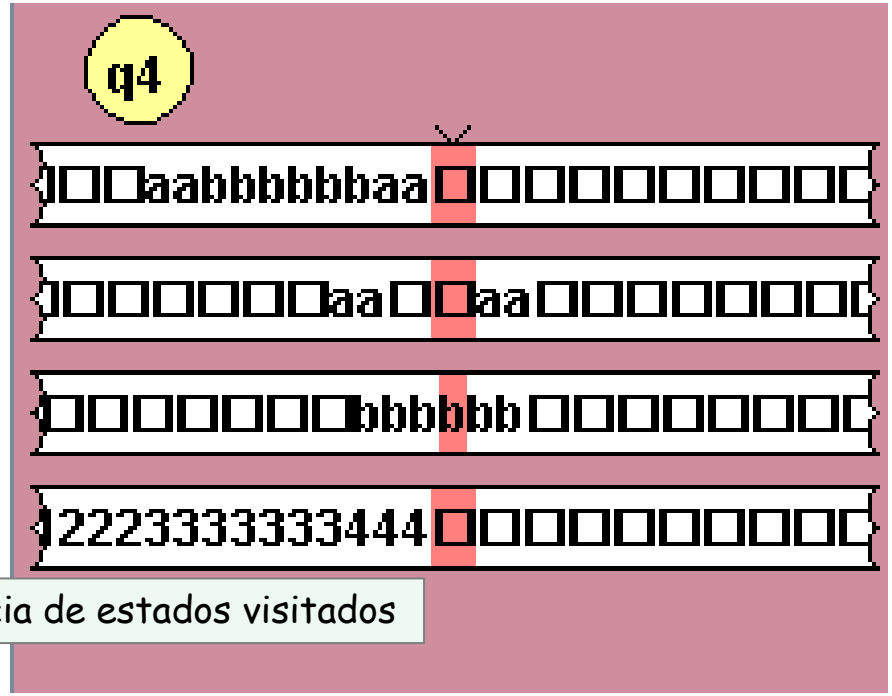


transdução com sequência de estados visitados

Rastros da Complexidade de Tempo de MT_{lenta}

Rejeição de aabbbbbbaa

|aabbbbbbaa| = 10



17 transições

transdução com sequência de estados visitados

Exercício 1:

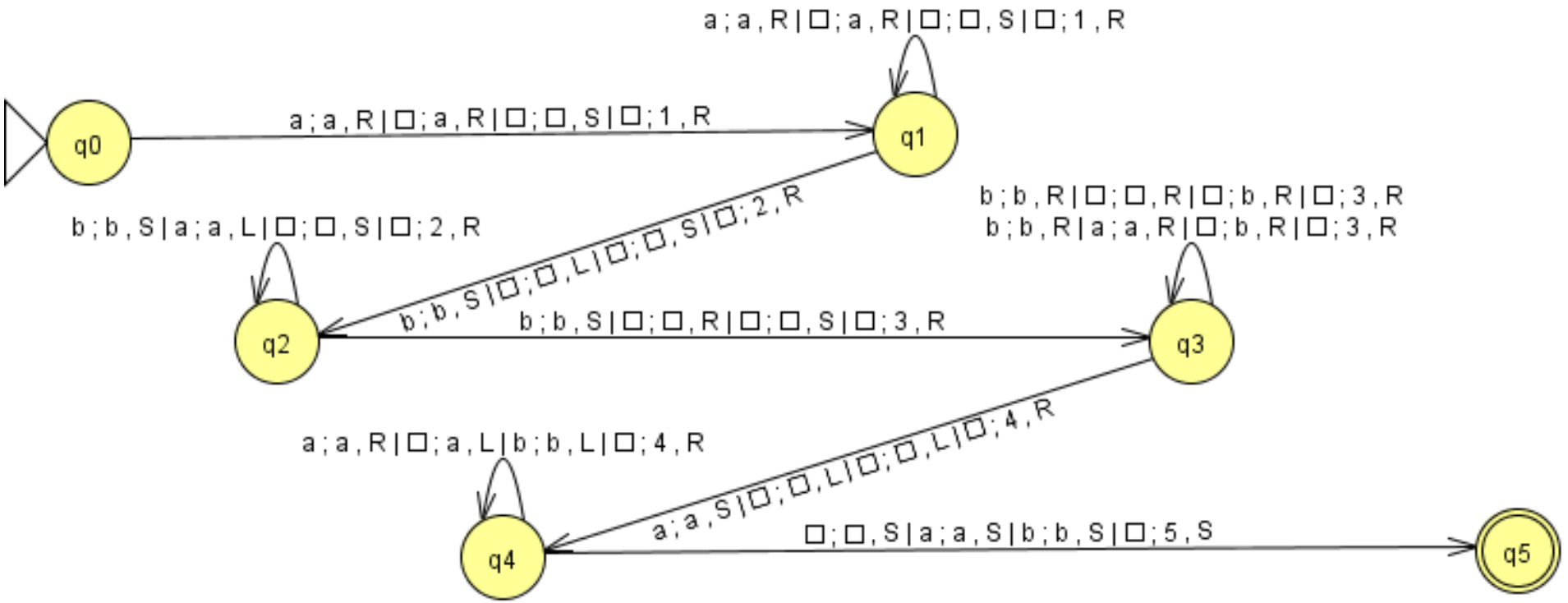
Como tornar MT_{lenta} menos lenta (e correta, se preciso)?

-- O exercício é para ser feito idealmente em duplas, com o(a) colega a seu lado.

-- A especificação de MT_{lenta} no JFLAP está no slide seguinte. Considera-se como condição de **aceitação** da cadeia a **parada** de MT_{lenta} no **estado final** e o **cursor posicionado ao final da cadeia entrada**.

-- Estas condições, porém, podem ser diferentes (por exemplo, cursor em outra posição, parada em estado não final, etc.). O que não pode mudar é que cadeias não pertencentes a L sejam rejeitadas e cadeias pertencentes sejam aceitas (L é decidível).

Deve reconhecer $a^n b^m a^q$ para n,m,q > 0 e m=n+q



Exercício 2:

Como $\varepsilon \notin L$, podemos dizer que L é uma linguagem de tipo 1?

-- Há uma MT de fita **limitada** que reconhece L?

-- Se sua resposta for positiva, compare a **complexidade de tempo** da **MT de fita limitada** que reconhece L com a da MT_{lenta} .
O que se observa e conclui?

Diminuir a complexidade de uma MT é torná-la **mais eficiente**. Por vezes, eficiência de tempo **pode vir em detrimento de** eficiência de espaço, ou vice-versa.

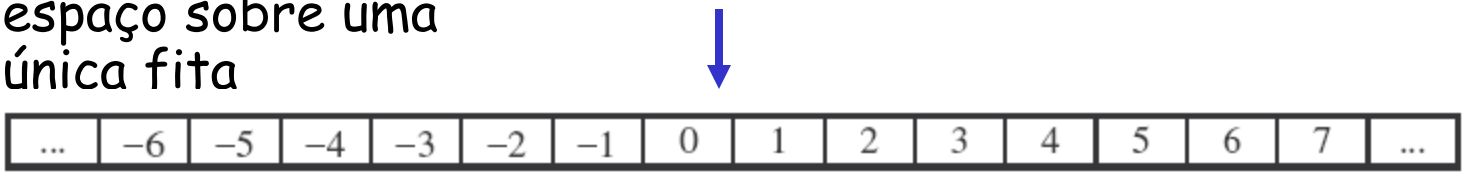
Fatos interessantes

Na apostila online de Chris Cooper (http://www.ics.mq.edu.au/~chris/langmach/chap09.pdf) mostra-se como uma MT de fita dupla pode ser convertida em MT equivalente de fita única.

O processo começa por uma representação unidimensional do espaço bidimensional:

1. Rotular as fitas superior e inferior com inteiros positivos ímpares (acima) e inteiros positivos pares (abaixo)
2. Em seguida projetar este espaço sobre uma única fita

	...	−5	−3	−1	1	3	5	7	9	...	
	...	−6	−4	−2	0	2	4	6	8	...	



Continuação:

3. Em seguida estabelecem-se as seguintes correspondências:

   - mov UP na MT(bidim) → mov LEFT na MT(unidim)
   - mov DOWN na MT(bidim) → mov RIGHT na MT(unidim)
   - mov LEFT na MT(bidim) → mov LEFT LEFT na MT(unidim)
   - mov RIGHT na MT(bidim) → mov RIGHT RIGHT na MT(unidim)

Com estas correspondências, a MT da aula passada que computava a função swap(n,m) para n representado em uma fita e m na outra pode ser computada por MT' de uma única fita.

Exercício 3: para casa com 2 MT's para swap(n,m)

Tabela de Transição Bidim

	0	1	
0	0U1	1U2	read bottom track
1	0D3	0D5	0 read on bottom
2	1D5	1D6	1 read on bottom
3	0L3	1L4	halt if both tracks read 0
4	0R7	1L4	
5	1R0	0R0	swap 1 with 0
6		1R0	"swap" 1 with 1

1. Estude os passos de conversão MT(bidim) → MT(unidim)
2. Compare a complexidade de tempo das duas MT's

Tabela de Transição Unidim

	0	1
0	0L1	1L2
1	0R3	0R5
2	1R5	1R6
3	0L3	1L4
3′	0L3	1L3
3″	0L4	1L4
4	0R7	1L4
4′	0L7	1L7
4″	0L4	1L4
5	1R0	0R0
5′	0R0	1R0
5″	0R0	1R0
6		1R0
6′	0R0	1R0