Informática
PUC-Rio

Linguagens Formais e Autômatos (LFA)

Aula de 18/09/2013

Propriedades Formais de Linguagens Regulares – Lema do Bombeamento

Lema do Bombeamento

Informalmente:

cadeia = prefixo.<subcadeia bombaeada>.sufixo ; --cadeia $\in$ L

$cadeia_0$ = prefixo.sufixo; --$cadeia_0 \in$ L

$cadeia_2$ = prefixo.<subcadeia bombaeada>.<subcadeia bombaeada>.sufixo ; --$cadeia_2 \in$ L

...

$cadeia_n$ = prefixo.<subcadeia bombaeada>.<subcadeia bombaeada>...<subcadeia bombaeada>.sufixo ; --$cadeia_n \in$ L

Função do Lema do Bombeamento para LR's

As LR's exibem a propriedade de *bombeamento*. Tal propriedade se verifica em todas as cadeias pertencentes a uma linguagem regular e que tenham ao menos um *comprimento mínimo*. Trata-se de uma propriedade *necessária mas não suficiente* para determinar que a linguagem é regular. Ou seja: *se não for* exibida, *L não é* LR; mas *se for*, L *pode ser ou não* LR.

Lema do Bombeamento : Formalmente

Se L é uma linguagem regular, existe **<u>um inteiro p</u>** (tamanho mínimo para o bombeamento) tal que:

se $w \in L$ e $|w| \geq p$

então se segmentarmos w em $w = x.y.z$

para todo $i \geq 0$, -- i será o iterador

$|y| > 0$ e -- se $y=\varepsilon$ o princípio é inócuo

$|xy| \leq p$

$w' = x.y^i.z \in L$

De outra forma:

Qualquer cadeia S suficientemente longa de uma LR pode ser subdividida em 3 subcadeias: S0 (inicial), S1 (intermediária) e S2 (final).

Se susbstituirmos S1 por S1* (entre S0 e S2), as novas cadeias resultantes a cada iteração de S1 serão SEMPRE pertencentes a LR também.

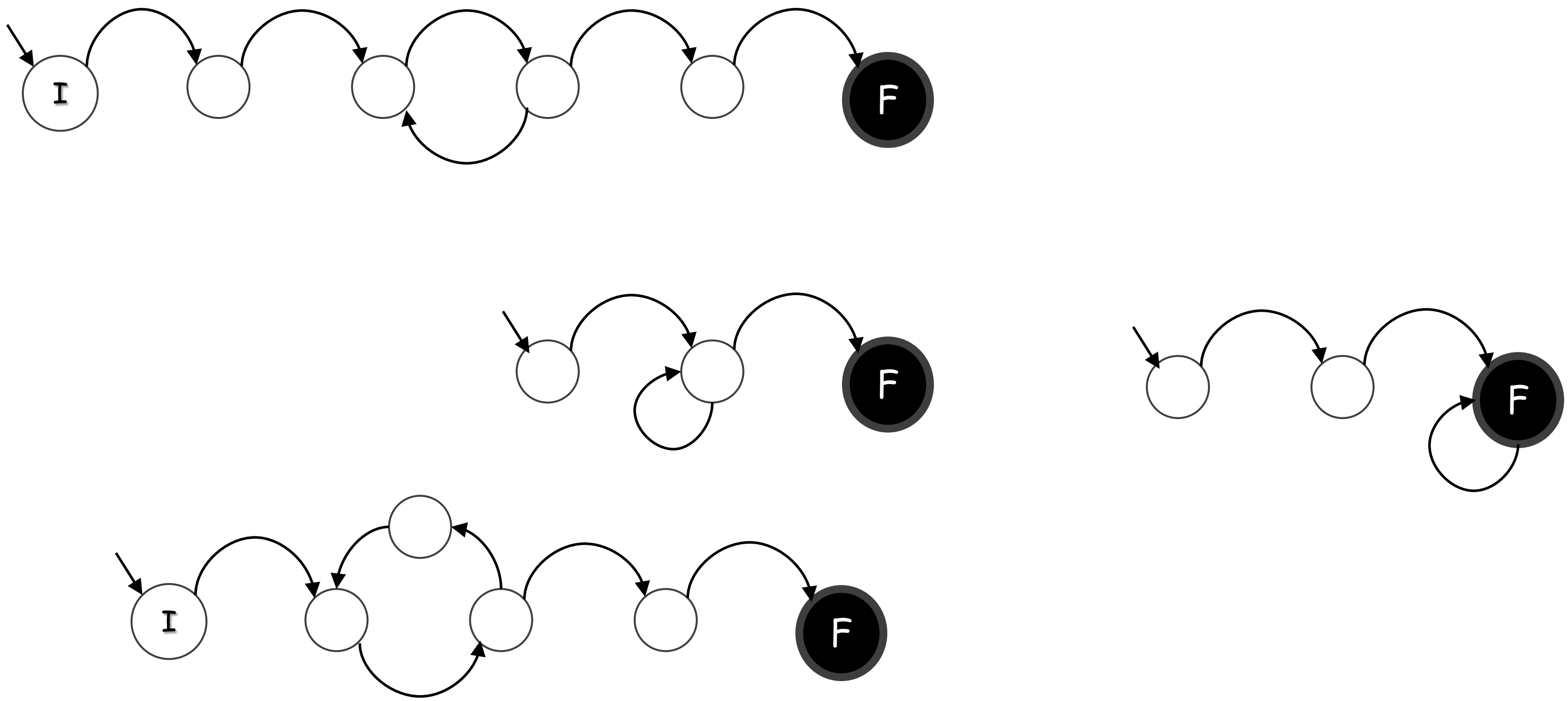
Por que S tem de ser “suficientemente longa”? Porque precisa poder ser decomposta em 3 subcadeias.

O que é o “bombeamento”? Se pegarmos a cadeia intermediária (S1) e a concatenarmos consigo mesma i vezes, para i ≥ 0, toda concatenação S0, S1*, S2 resulta em uma sentença da mesma LR

Bombeamento e Reconhecimento por AFD

- Se uma linguagem L qualquer é REGULAR, então ela é aceita por um Autômato Finito AF predefinido contendo "n" estados.
- Se AF é capaz de reconhecer uma cadeia w $\in$ L cujo comprimento é **maior** do que "n" (ou seja, |w| > n), só pode ser porque AF passa por algum dos n estados de L mais de uma vez (ou seja: há um **ciclo** no processamento de w).
- Logo, a cadeia w pode ser dividida em 3 subcadeias: a que VEM ANTES do ciclo (S0), a ONDE ESTÁ o ciclo (S1), e a que VEM DEPOIS do ciclo (S2). Ou seja: S = S0,S1,S2.
  - O comprimento da cadeia até o ponto do ciclo tem de ser menor ou igual ao número de estados do AF que reconhece L; e
  - O comprimento de S1 (parte reconhecida pelo ciclo) tem de ser ao menos "1".
- Como há um ciclo, então qualquer iteração de S1 (ou seja, S1*) é aceitável, significando que qualquer cadeia S0, S1*, S2 pertence também a L.

Padrões para o "Bombeamento" (exemplos)



Exercícios : Diga se são regulares as L abaixo

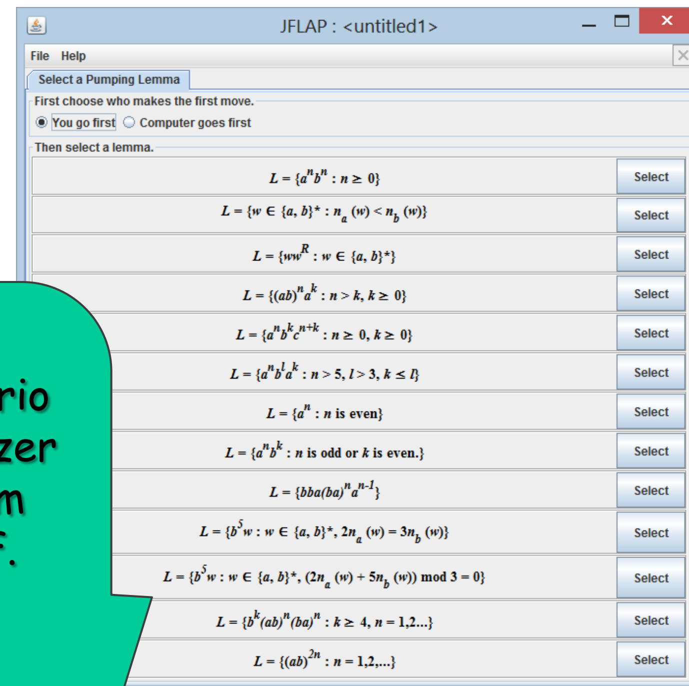
1) $L = (ab)^{2n}$ para $n > 0$

2) $L = a^n.b^k$ para n ímpar OU k par

3) $L = a^n$ para $n>0$ e n par

4) $L = w.w^{rev}$

5) $L = a^n.b^n$



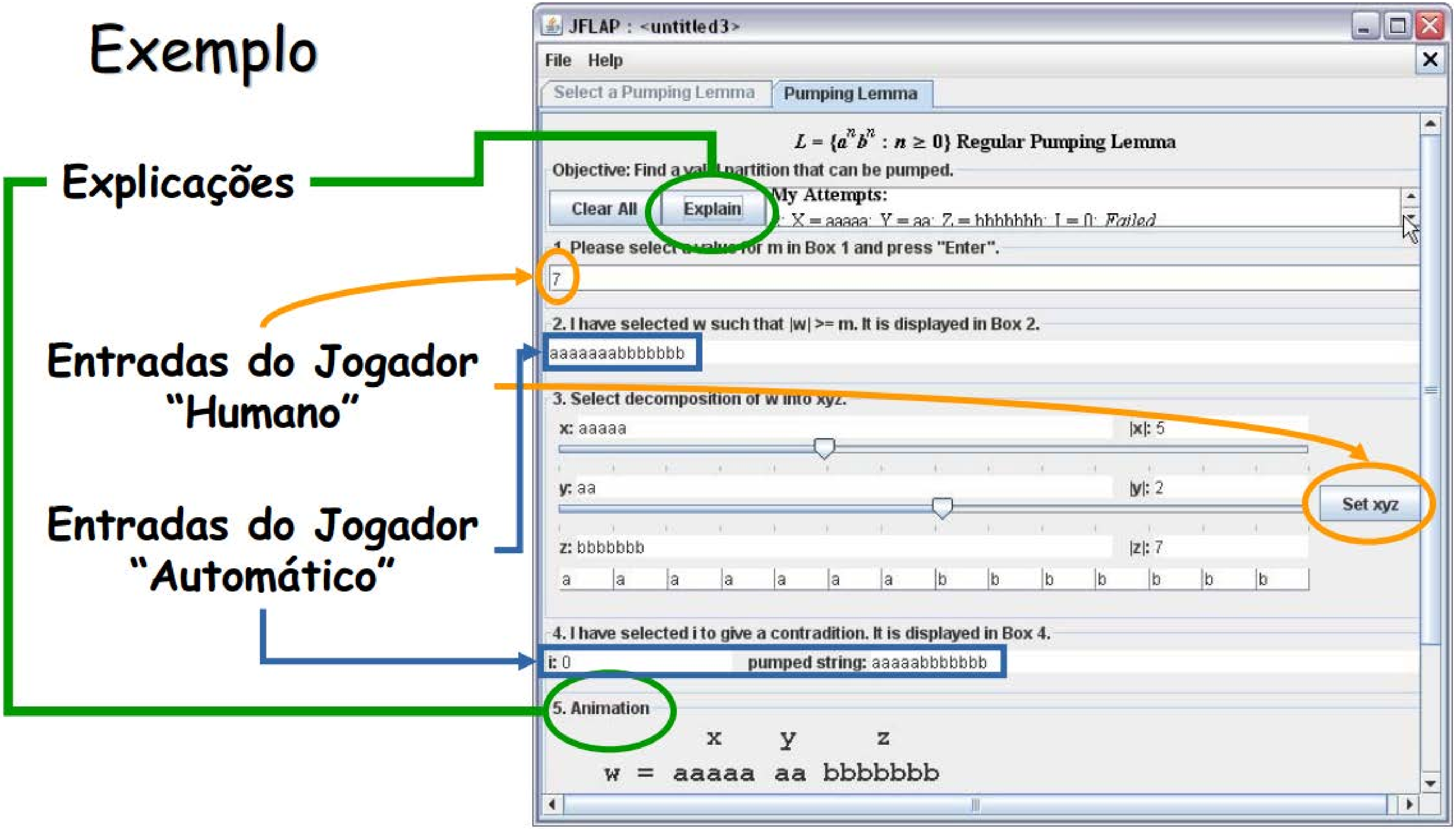
Lembre:
O bombeamento é necessário mas não suficiente para dizer se L é LR. L precisa também ser reconhecida por um AF. Algumas linguagens não LR passarão pelo teste do bombeamento. Mas, se não passarem não são LR.

O "jogo do bombeamento" no JFLAP

Joga-se assim:

1. O jogador *A* escolhe um inteiro m (corresponde a 'p').
2. O jogador B escolhe a cadeia w tal que w pertence a L e $|w| \geq m$.
3. O jogador *A* segmenta w em xyz de tal modo que $|xy| \leq m$ e $|y| \geq 1$.
4. O jogador B propõe um número de iterações i tentando mostrar que xy^iz não pertence a L. Se conseguir, B ganha. Se não conseguir, *A* ganha.

Exemplo


Explicações
Entradas do Jogador "Humano"
Entradas do Jogador "Automático"
JFLAP : <untitled3>
File Help
Select a Pumping Lemma
Pumping Lemma
L = {a^n b^n : n ≥ 0} Regular Pumping Lemma
Objective: Find a valid partition that can be pumped.
Clear All
Explain
My Attempts:
1. Please select a value for m in Box 1 and press "Enter".
7
2. I have selected w such that |w| >= m. It is displayed in Box 2.
aaaaaaabbbbbbb
3. Select decomposition of w into xyz.
x: aaaaa
|x|: 5
y: aa
|y|: 2
Set xyz
z: bbbbbbb
|z|: 7
4. I have selected i to give a contradition. It is displayed in Box 4.
i: 0
pumped string: aaaaabbbbbbb
5. Animation
x y z
w = aaaaa aa bbbbbbb

Para testar o Lema do Bombeamento

- Tome-se a linguagem L
- Tome-se um inteiro **p** correspondente ao suposto número estados de um AF que reconheceria L (se ela fosse regular).
- Tome-se uma cadeia qualquer $w \in L$ sendo $|w| \geq p$

Se encontrarmos UM CASO (uma cadeia) em que o bombeamento funciona, a linguagem passou no teste?

Não: tem de funcionar para QUALQUER caso (qualquer w) "de tamanho suficiente".
Portanto, a demonstração do Lema do Bombeamento tem de se pautar pelas propriedades formais de L e suas manifestações dentro de cadeias quaisquer delimitadas pelo tamanho estipulado.

Encontre-se uma partição de w = x.y.z tal que **<u>se verifiquem as seguintes condições:</u>**

- $|xy| \leq p$
- $|y| > 0$
- $xy^*z \in L$

Note:

- As tentativas de segmentação xy (prefixo+cadeia_a_bombear) podem acontecer em qualquer ponto "até" que se atinja o limite "p". Portanto, se uma segmentação não der certo, devem-se tentar outras.

Exemplo de Aplicação

$L = bba(ba)^n a^{n-1}$

Seja p = 9

w= bbababab a a

x = bbababa

y = baa

1 ocorrência de ba +
1 ocorrência de a

z = a

$|xy| \leq 9$? Não!

Adiantaria "diminuir" $|xy|$?

Espaço restante:

bba baba baaa:

- 'bba' não pode ser bombeado, pois é "fixo"
- no intervalo disponível, as únicas alternativas seriam: bombear a* ou b* - o que criaria cadeias obviamente rejeitáveis - ou então (ba)* - que não daria conta da dependência de a^{n-1} ao final.

Será que o problema seria a escolha de "p"?

Continuação do Exemplo L = $bba(ba)^n a^{n-1}$

Adianta variar "p"? Afinal, o lema fala em cadeia "suficientemente longa", mas não define "p".

Se **aumentarmos** "p", estaremos TAMBÉM aumentando a cadeia w (pois |w|≥ p). Portanto, isto não é promissor.

Se **diminuirmos** "p", o tamanho mínimo *suficientemente longo* de '(ba)' e 'a' teria de preservar ao menos |a|=1 (ou não vemos nada do que se passa com os 'a' finais). Chegaríamos então a: w= bbababaa. 2xba e 1xa
Resolveria? Não!

O Lema do Bombeamento para L acima descrita **não funciona** e portanto L **não é regular**.

O problema é que o bombeamento de qualquer y* para |y|> 0 faz:

- Uma "retração" de w (caso y^0)
- Infinitas "expansões" de w (caso y+) onde, para funcionar:
  - Ou bem y* encapsula dentro de si todas as dependências entre as subcadeias de w
  - Ou então não há dependências entre subcadeias de w.

Exploração das condições "limite"

O exemplo de **L = bba(b)^{n}a^m para n>m** é interessante.

Mostra que podemos cair no erro de **casos especiais**, que o jogo no JFLAP justamente explora. (Se o jogador humano não for 'esperto' para detectar os casos especiais, pode não conseguir 'derrubar' o adversário no jogo).

O 'esperto' é desenvolvermos a habilidade de enxergar rapidamente a diferença entre o caso especial e o caso geral das cadeias $w \in L$. (E é este o objetivo do jogo).

Uma armadilha com L = $bba(b)^n a^m$ para n>m

Sejam:

p = 7

w = bbabbbbaa |w| = 9

x = bba |x| = 3

y = b |y| = 1

z = bbbaa |z| = 5

Reparem que com este sufixo z pouco importa quantos b bombeamos com y*. A cadeia w resultante sempre pertencerá a L.

$\lvert xy \rvert \leq 7$	OK
$\lvert y \rvert > 0$	OK
$xy^*z \in L$	OK

Deu certo? Demonstrei o que queria? ☺

Não, vejam só.

A cadeia w = bbabbbbaa é peculiar; tem 4 b e 2 a. Ela nos permite a segmentação xyz proposta, na qual o movimento de retração e as sucessivas expansões durante o bombeamento sempre caem 'dentro' de L.

Pensando <u>melhor</u> sobre L = $bba(b)^n a^m$ para n>m

Comecemos da mesma forma por p = 7.

Procuremos uma cadeia w' de tamanho mínimo 7, cujas restrições sejam **as mesmas de L**, mas **expressas de outra forma**:
$bba(b)^n a^{n-i}$ para $1 \leq i \leq n$.

Esta formulação mostra mais claramente o **<u>espaço de trabalho</u>** para examinarmos o bombeamento.

w' = bba??????

w' = bba?????? |w| ≥ 7

O espaço de trabalho para o bombeamento vai:

De m = 0 (limite superior de i=n)

w' = bbabbbbbb....

até m = n-1 (limite inferior de i=1)

Então, seja w' = bbabbbaa |w| = 8

O bombeamento de b* não vai funcionar aqui. Na retração da cadeia com b^0, a condição $b^n a^m$ para n>m será violada.

Ou seja, nossa cadeia anterior **<u>era</u>** mesmo um caso especial.

Para lembrar ao fazer exercícios com o LB

- Trabalhar com as condições formais do Lema do Bombeamento (LB).
- Trabalhar com as propriedades formais da linguagem L dentro de uma cadeia de tamanho limitado mas *suficientemente longo*.
- Tamanho *suficientemente longo* é aquele em que as condições relevantes de teste:
  - se manifestam na cadeia w e
  - resistem ao passo de retração em y*.
- Identificar claramente o 'espaço de trabalho' para o bombeamento dentro da cadeia w.
  - Observar limites inferiores e superiores das condições de iteração já expressas na definição da linguagem.
  - Observar subcadeias persistentes, fora das condições de iteração (em prefixos, sufixos e até infixos).
- Demonstrar formalmente a (não) preservação das condições formais relevantes no espaço de trabalho em função do bombeamento.

Explorem o jogo no JFLAP com isto em mente.