4
Linguagens que não são regulares

O objetivo deste capítulo é desenvolver um método que nos permita mostrar que uma dada linguagem não é regular. Nossa estratégia será a seguinte. Em primeiro lugar, mostraremos que toda linguagem regular satisfaz certa propriedade, conhecida como *propriedade do bombeamento*. Assim, para provar que uma dada linguagem não é regular basta constatar que não satisfaz esta propriedade. Isto é, provaremos que uma linguagem não é regular através de uma demonstração por contradição.

1. Propriedade do bombeamento

Começamos por introduzir a terminologia básica e obter uma primeira aproximação para a propriedade de bombeamento das linguagens regulares.

Considere o autômato M da figura abaixo.

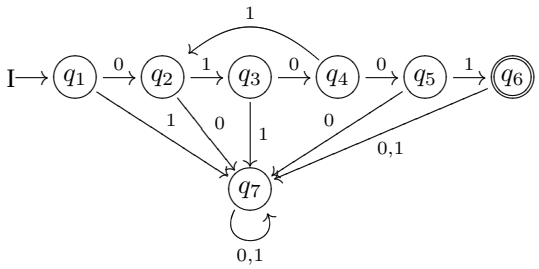

FIGURA 1

Entre as palavras aceitas por este autômato temos:

$$w = 01011001.$$

Se considerarmos o caminho indexado pela palavra $w = 01011001$, vemos que inclui um ciclo que começa em q_2 e acaba em q_4. Este ciclo corresponde à subpalavra $y = 101$ de 01011001. Mais precisamente, podemos decompor w na forma

$$w = \underbrace{0}_{x}\underbrace{101}_{y}\underbrace{1001}_{z} = xyz.$$

Observe que podemos percorrer o ciclo indexado por y várias vezes e ainda assim obter uma palavra que é aceita por M. Por exemplo, percorrendo o ciclo 3 vezes obtemos

$$xy^3z = \underbrace{0}_{x}\underbrace{101}_{y}\underbrace{101}_{y}\underbrace{101}_{y}\underbrace{1001}_{z},$$

que é aceita por M. De fato, podemos até remover a subpalavra y de w e ainda assim continuaremos com uma palavra aceita pelo autômato, neste caso $xz = 01001$. Resumindo, verificamos que a palavra $w = xyz$ é aceita por M e que admite uma subpalavra $y \neq \epsilon$ que pode ser removida ou repetida várias vezes sem que a palavra resultante fique fora de $L(M)$. Sempre que isto acontecer diremos que y é uma subpalavra de w que é *bombeável* em $L(M)$.

Naturalmente, o ponto chave é o fato da subpalavra y indexar um ciclo no grafo de M. Na verdade, esta não é a única subpalavra bombeável de w uma vez que podemos considerar o ciclo como começando em qualquer um de seus vértices. Assim, se tomarmos o início do ciclo como sendo q_4, concluímos que a subpalavra 110 também é bombeável; isto é,

$$010(110)^k01 \in L(M) \text{ para todo } \ k \geq 0.$$

É conveniente estabelecer a noção de bombeabilidade em um contexto mais geral que o das linguagens regulares. Seja L uma linguagem em um alfabeto Σ e seja $w \in L$. Dizemos que $y \in \Sigma^*$ é uma *subpalavra de w bombeável em L* se

(1) $y \neq \epsilon$;
(2) existem $x, z \in \Sigma^*$ tais que $w = xyz$;
(3) $xy^kz \in L$ para todo $k \geq 0$.

A única função da condição (1) é excluir o caso trivial $y = \epsilon$ da definição de palavra bombeável; já (2) significa que y é subpalavra de w. Quanto a (3), o ponto crucial a observar é que, para que seja y seja bombeável é preciso que seja possível omiti-la ou repeti-la no interior de w tantas vezes quanto desejarmos *sem que a palavra resultante deixe de pertencer a L*. Para entender melhor este ponto, considere o autômato M' da figura 2.

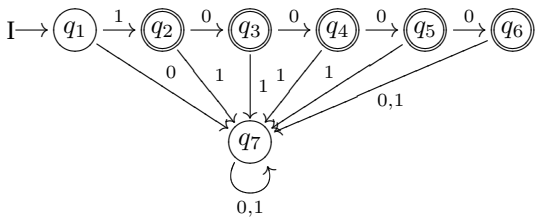

FIGURA 2

É claro que 0 é uma subpalavra de $10 \in L(M')$. Além disso, podemos repetir a subpalavra 0 várias vezes e ainda assim obter uma palavra em $L(M')$; de fato, 1, 10, 10^2, 10^3 e 10^4 pertencem a $L(M')$. Apesar disto, 0 não é uma subpalavra de 10 bombeável em $L(M')$, porque se $k \geq 5$ então $10^k \notin L(M')$.

Permanecendo com o autômato M' da figura 2, vemos que

$$L(M') = \{1, 10, 10^2, 10^3, 10^4\}.$$

Em particular, não há nenhuma palavra de $L(M')$ que admita uma subpalavra bombeável. Isto não é surpreendente. De fato, se uma linguagem admite uma palavra que tem uma subpalavra bombeável, então é claro que a linguagem é infinita. Há dois pontos importantes nesta discussão que você não deve esquecer:

- nenhuma palavra de uma linguagem finita L admite subpalavra bombeável em L;

- para que uma subpalavra seja bombeável é preciso que possa ser repetida *qualquer número de vezes* sem que a palavra resultante saia de L.

2. Lema do bombeamento

Diante do que acabamos de ver, uma pergunta se impõe de maneira natural:

> dada uma linguagem L, como achar uma palavra de L que admita uma subpalavra bombeável?

Em primeiro lugar, esta pergunta só faz sentido se L for infinita. Além disso, vamos nos limitar, de agora em diante, às linguagens regulares. Sendo assim, vamos supor que $L \subset \Sigma^*$ é uma linguagem infinita que é aceita por um autômato finito determinístico M no alfabeto Σ.

Se conhecemos M o problema é fácil de resolver: basta achar um ciclo em M. Mas suponha que, apesar de conhecer L, sabemos de M apenas que tem n estados. Será que esta informação é suficiente para achar uma palavra de L que tenha uma subpalavra bombeável em L? A resposta é sim, e mais uma vez trata-se apenas de achar um ciclo em M. Só que, como não conhecemos M, não temos uma maneira de identificar qual é este ciclo. Mesmo assim somos capazes de saber que um tal ciclo *tem que existir*. Fazemos isto recorrendo a um princípio que aprendemos a respeitar ainda criança, quando brincamos de dança das cadeiras.

PRINCÍPIO DA CASA DO POMBO. *Se, em um pombal, há mais pombos que casas, então dois pombos vão ter que ocupar a mesmo casa.*

Nossa aplicação deste princípio depende de termos, de um lado uma linguagem **infinita**, de outro um autômato **finito** determinístico. De fato, como L é infinita, terá palavras de comprimento arbitrariamente grande. Em particular, podemos escolher uma palavra w cujo comprimento é muito maior que o número n de estados de M. Considere o caminho indexado por w no grafo de M. Como M tem n estados e w tem muito mais do que n símbolos, este caminho tem que passar duas vezes por um mesmo estado. Mas um caminho no grafo de M no qual há estados repetidos tem que conter um ciclo. Entretanto, já sabemos que um ciclo no caminho indexado por w nos permite determinar uma subpalavra bombeável de w. Com isto provamos a seguinte *propriedade do bombeamento* das linguagens regulares:

> Seja M um autômato finito determinístico. Se w é uma palavra de $L(M)$ de comprimento maior que o número de estados do autômato, então w admite uma subpalavra bombeável em $L(M)$.

O lema do bombeamento, que é o principal resultado deste capítulo, não passa de uma versão refinada da propriedade do bombeamento enunciada acima.

LEMA DO BOMBEAMENTO. *Seja M um autômato finito determinístico com n estados e seja L a linguagem aceita por M. Se w é uma palavra de L com comprimento maior ou igaul a n então existe uma decomposição de w na forma $w = xyz$, onde*

(1) $y \neq \epsilon$;
(2) $|xy| \leq n$;
(3) $xy^k z \in L$ *para todo* $k \geq 0$.

Antes de passar à demonstração, observe que (1) e (3) nos dizem apenas que y é subpalavra de w bombeável em L. A única novidade é a condição (2). Esta condição

técnica permite simplificar várias demonstrações de não regularidade, reduzindo o número de casos que precisam ser considerados.

DEMONSTRAÇÃO. A estratégia adotada no início da seção consistiu em considerar o caminho no grafo de M indexado por w. Como observamos no capítulo 2, isto é formalizado através da computação de M determinada por w.

Seja Σ o alfabeto de M. Então podemos escrever $w = \sigma_1 \cdots \sigma_n$, onde $\sigma_1, \ldots, \sigma_n$ são elementos de Σ não necessariamente distintos. Seja q_1 o estado inicial de M. Temos, então, uma computação

$$(q_1, w) = (q_1, \sigma_1 \cdots \sigma_n) \vdash (q_2, \sigma_2 \cdots \sigma_n) \vdash \cdots \vdash (q_n, \sigma_n) \vdash (q_{n+1}, \epsilon).$$

Observe que também não estamos supondo que os estados $q_1, \ldots, q_{n+1}$ são todos distintos. De fato, dois destes estados têm que coincidir, porque M só tem n estados. Digamos que $q_i = q_j$, onde $1 \leq i < j \leq n+1$. Qualquer escolha de i e j que satisfaça as condições acima é suficiente para provar (1) e (3); mas não (2). Para garantir (2) precisamos escolher q_j como sendo o primeiro estado que coincide com algum estado anterior. Assumindo desde já que i e j são inteiros entre 1 e $n+1$, precisamos fazer a seguinte hipótese sobre j:

Hipótese: j é o menor inteiro para o qual existe $i < j$ tal que $q_i = q_j$.

Levando tudo isto em conta, podemos reescrever a computação na forma

$$(q_1, w) = (q_1, \sigma_1 \cdots \sigma_n) \vdash^* (q_i, \sigma_i \cdots \sigma_n) \vdash^* (q_j, \sigma_j \cdots \sigma_n) \vdash^* (q_n, \sigma_n) \vdash (q_{n+1}, \epsilon).$$

O ciclo que procuramos está identificado pelo trecho da computação que vai de q_i a $q_j = q_i$. Isto sugere que devemos tomar

$$x = \sigma_1 \cdots \sigma_{i-1}, \quad y = \sigma_i \cdots \sigma_{j-1} \quad \text{e} \quad z = \sigma_j \cdots \sigma_{n+1}.$$

Além disso, como $i < j$ temos que

$$y = \sigma_i \cdots \sigma_{j-1} \neq \epsilon,$$

de forma que a condição (1) é satisfeita. Usando esta notação, a computação fica

$$(q_1, w) = (q_1, xyz) \vdash^* (q_i, yz) \vdash^* (q_j, z) \vdash^* (q_{n+1}, \epsilon).$$

Note que, como $q_i = q_j$, a palavra y leva a computação do estado q_i ao estado q_i. Desta forma, repetindo ou omitindo y, podemos fazer este trecho repetir-se várias vezes no interior da computação sem alterar o estado em que computação termina, que continuará a ser q_{n+1}. Por exemplo, repetindo y uma vez temos a palavra xy^2z, que dá lugar à computação

$$(q_1, xy^2z) \vdash^* (q_i, y^2z) \vdash^* (q_j, yz) = (q_i, yz) \vdash^* (q_j, z) \vdash^* (q_{n+1}, \epsilon).$$

Como $xyz \in L(M)$ por hipótese, então q_{n+1} é um estado final de M. Portanto, $xy^2z \in L(M)$. De maneira semelhante $xy^kz \in L(M)$ para todo $k \geq 0$, o que prova (3).

Falta-nos apenas explicar porque (2) vale. Mas, $|xy| = j - 1$. Entretanto, q_j é o primeiro estado que coincide com algum estado anterior. Isto é, $q_1, \ldots, q_{j-1}$ são todos estados distintos. Como M tem n estados, isto significa que $j - 1 \leq n$. Portanto, $|xy| \leq n$, o que completa a demonstração.

Antes de passar às aplicações é preciso chamar a atenção para o fato de que a recíproca do lema do bombeamento é falsa. Isto é, o fato de uma linguagem L conter palavras que admitem subpalavras bombeáveis *não* garante que L seja regular. Portanto, não é possível

provar regularidade usando o lema do bombeamento. Voltaremos a discutir este ponto no exemplo 5.

3. Aplicações do lema do bombeamento

O maior obstáculo à aplicação do lema do bombeamento está na interpretação correta do seu enunciado. Seja M um autômato finito determinístico com n estados. Segundo o lema do bombeamento, dada **qualquer** palavra $w \in L(M)$ de comprimento maior que n **existe** uma subpalavra $y \neq \epsilon$ que é bombeável em $L(M)$. Note que o lema não diz que qualquer subpalavra de w é bombeável, mas apenas que **existe** uma subpalavra de w que é bombeável.

Por exemplo, considere a linguagem L no alfabeto $\{0\}$ formada pelas palavras de comprimento par. É fácil construir um autômato finito com 2 estados que aceita L, portanto esta é uma linguagem regular e $n = 2$. Vamos escolher uma palavra de L de comprimento maior que 2; digamos, 0^6. Não é verdade que qualquer subpalavra de 0^6 é bombeável em L. Por exemplo, 0 é uma subpalavra de 0^6, já que temos uma decomposição $0^6 = 0^2 \cdot 0 \cdot 0^3$; mas bombeando 0 obtemos

$$0^2 \cdot 0^k \cdot 0^3 = 0^{5+k},$$

que não pertence a L se k por par. De fato, para que a subpalavra seja bombeável em L é preciso que tenha comprimento par. Assim, neste exemplo, poderíamos escolher as subpalavras 0^2, 0^4 ou 0^6 para bombear.

Tudo isto pode parecer óbvio. O problema é que um nível adicional de dificuldade surge nas aplicações, porque desejamos usar o lema para provar que uma linguagem **não** é regular. Imagine que temos uma linguagem L e que, por alguma razão, desconfiamos que L não é regular. Para provar que L *de fato* não é regular podemos proceder por contradição.

Suponha, então, por contradição, que L seja aceita por algum autômato finito determinístico com n estados. De acordo com o lema do bombeamento qualquer palavra $w \in L$ de comprimento maior que n terá que admitir uma subpalavra bombeável. Assim, para obter uma contradição, basta achar **uma** palavra em L (o que é uma boa notícia!) que não tenha **nenhuma** subpalavra bombeável (o que é uma má notícia!).

Um último comentário antes de passar aos exemplos. Neste esboço de demonstração por contradição supusemos que L é aceita por um autômato finito determinístico com n estados. Entretanto, ao fazer esta hipótese não podemos especificar um valor numérico para n. De fato, se escolhermos $n = 100$, tudo o que teremos provado é que a linguagem não pode ser aceita por um autômato com 100 estados. Mas nada impediria, em princípio, que fosse aceita por um autômato com 101 estados. Resta-nos aplicar estas considerações gerais em alguns exemplos concretos.

Exemplo 1. Considere a linguagem no alfabeto $\{0\}$ definida por

$$L_{primos} = \{0^p : p \text{ é um primo positivo}\}.$$

A primeira coisa a observar é que esta linguagem é infinita. Isto é uma conseqüência de teorema provado pelo matemático grego Euclides por volta de 300 a. C., segundo o qual existem infinitos números primos.

Em seguida devemos considerar se seria possível construir um autômato finito que aceitasse esta linguagem. Para isto, seria necessário que o autômato pudesse determinar se um dado número p é primo ou não. Em outras palavras, o autômato teria que se certificar que p não é divisível pelos inteiros positivos menores que p. Como a quantidade de inteiros menores que p aumenta com p, isto requer uma memória infinita; que é exatamente o que

um autômato finito não tem. Esta é uma boa indicação de que L_{primos} não é regular. Vamos comprovar nosso palpite usando o lema do bombeamento.

Suponha, então, por contradição, que L_{primos} é aceita por um autômato finito determinístico com n estados. Precisamos escolher uma palavra com comprimento maior que n em L_{primos}. Para fazer isto, basta escolher um primo $q > n$. A existência de um tal primo é conseqüência imediata do teorema de Euclides mencionado acima. Portanto, 0^q é uma palavra de L_{primos} de comprimento maior que n.

Nestas circunstâncias, o lema do bombeamento garante que existe uma decomposição $0^q = xyz$ de modo que $y \neq \epsilon$ é bombeável em L_{primos}. Como o que desejamos é contradizer esta afirmação, temos que mostrar que 0^q não admite nenhuma subpalavra bombeável. Neste exemplo é fácil executar esta estratégia neste grau de generalidade. De fato, uma subpalavra não vazia qualquer de 0^q tem que ser da forma 0^j para algum $0 < j \leq q$. Mas x e z também são subpalavras de 0^q; de modo que também são cadeias de zeros. Tomando, $x = 0^i$, teremos que $z = 0^{q-i-j}$.

Bombeando y, concluímos que

$$xy^kz = 0^i(0^j)^k0^{q-i-j} = 0^{i+jk+(q-i-j)} = 0^{q+(k-1)j}$$

deve pertencer a L_{primos} para todo $k \geq 0$. Mas isto só pode ocorrer se $q + (k-1)j$ for um número primo para todo $k \geq 0$. Entretanto, tomando $k = q + 1$, obtemos

$$q + (k-1)j = q + qj = q(1 + j)$$

que não pode ser primo porque tanto q quanto $j + 1$ são números maiores que 1. Temos assim uma contradição, o que confirma nossas supeitas de que L_{primos} não é regular.

Note que a condição (2) do lema do bombeamento não foi usada em nenhum lugar nesta demonstração. Como frisamos anteriormente, esta é uma condição técnica que serve para simplificar o tratamento de exemplos mais complicados, como veremos a seguir.

Exemplo 2. Nosso próximo exemplo é a linguagem

$$L = \{a^mb^m : m \geq 0\}$$

no alfabeto $\{a, b\}$. Também neste caso é fácil dar um argumento heurístico que nos leva a desconfiar que L não pode ser regular. Lembre-se que o autômato lê a entrada da esquerda para a direita. Assim, ele lerá toda a seqüência de as antes de chegar aos bs. Portanto, o autômato tem que lembrar quantos as viu para poder comparar com o número de bs. Mas a memória do autômato é finita, e não há restrições sobre a quantidade de as em uma palavra de L.

Para provar que L não é regular vamos recorrer ao lema do bombeamento. Suponha, por contradição, que L é aceita por um autômato finito determinístico com n estados. Em seguida temos que escolher uma palavra w de L com comprimento maior que n; digamos que $w = a^nb^n$. Como $|w| = 2n > n$, tem que existir uma decomposição

$$a^nb^n = xyz$$

de forma que as condições (1), (2) e (3) do lema do bombeamento sejam satisfeitas.

Mas que decomposições de a^nb^n satisfazem estas condições? Dessa vez começaremos analisando (2), segundo a qual $|xy| \leq n$. Isto é, xy é um prefixo de a^nb^n de comprimento menor ou igual a n. Como a^nb^n começa com n letras a, concluímos que a é o único símbolo que x e y podem conter. Portanto,

$$x = a^i \quad \text{e} \quad y = a^j.$$

Além disso, $j \neq 0$ pela condição (1). Já z reúne o que sobrou da palavra w, de modo que

$$z = a^{n-i-j}b^n.$$

Observe que não há razão pela qual xy tenha que ser igual a a^n, de modo que podem sobrar alguns as em z.

Resta-nos bombear y. Fazendo isto temos que

$$xy^kz = a^i \cdot (a^j)^k \cdot a^{n-i-j}b^n = a^{n+(k-1)j}b^n,$$

é um elemento de L para todo $k \geq 0$. Contudo, $a^{n+(k-1)j}b^n$ só pode pertencer a L se os expoentes de a e b coincidirem. Porém

$$n + (k-1)j = n \quad \text{para todo} \quad k \geq 0$$

implica que $j = 0$, contradizendo a condição (1) do lema do bombeamento.

Antes de passar ao próximo exemplo convém considerar a escolha que fizemos para a palavra de comprimento maior que n. Não parece haver nada de extraordinário nesta escolha, mas a verdade é que nem toda escolha de w seria satisfatória. Por exemplo, assumindo que $n \geq 2$, teríamos que $|a^{n-1}b^{n-1}| = 2n-2 \geq n$. Entretanto, esta não é uma boa escolha para w. A razão é que

$$a^{n-1}b^{n-1} = xyz \quad \text{e} \quad |xy| \leq n$$

não excluem a possibilidade de y conter um b. Isto nos obrigaria a considerar dois casos separadamente, a saber, $y = a^j$ e $y = a^jb$, o que complicaria um pouco a demonstração. Diante disto, podemos descrever o papel da condição (2) como sendo o de restringir os possíveis y. O problema é que isto não se dá automaticamente mas, como no exemplo acima, depende de uma escolha adequada para w.

Por sorte, na maioria dos casos, muitas escolhas para w são possíveis. Neste exemplo, bastaria tomar $w = a^rb^r$ com $r \geq n$. Entretanto, para algumas linguagens a escolha da palavra requer bastante cuidado, como mostra o próximo exemplo.

Exemplo 3. Um argumento heurístico semelhante ao usado para a linguagem anterior sugere que

$$L = \{a^mb^r : m \geq r\}$$

não deve ser regular. Vamos provar isto usando o lema do bombeamento.

Suponhamos, por contradição, que L seja aceita por um autômato finito determinístico com n estados. Neste exemplo, como no anterior, uma escolha possível para uma palavra de comprimento maior que n em L é a^nb^n. Da condição (2) do lema do bombeamento concluímos que, se $a^nb^n = xyz$, então

$$x = a^i \quad \text{e} \quad y = a^j.$$

Já condição (1) nos garante que $j \neq 0$. Como $z = a^{n-i-j}b^n$, obteremos, ao bombear y, que

$$xy^kz = a^i \cdot (a^j)^k \cdot a^{n-i-j}b^n = a^{n+(k-1)j}b^n.$$

Mas, para que esta palavra esteja em L é preciso que

$$n + (k-1)j \geq n,$$

donde segue que $(k-1)j \geq 0$. Por sua vez, $j \neq 0$ força que $k-1 \geq 0$, ou seja, que $k \geq 1$. Mas, para que y seja bombeável é preciso que $xy^kz \in L$ para todo $k \geq 0$, e não apenas $k \geq 1$. Portanto, temos uma contradição com o lema do bombeamento, o que prova que L não é regular.

Desta vez estivemos perto de não chegar a lugar nenhum! De fato, uma contradição só é obtida porque tomando $k = 0$,

$$a^{n+(k-1)j}b^n = a^{n-j}b^n$$

não pertence a L. Entretanto, neste exemplo, muitas escolhas aparentemente adequadas de w não levariam a nenhuma contradição. Por exemplo, é fácil se deixar sugestionar pelo sinal $\geq$ e escolher $w = a^{n+1}b^n$. Esta palavra tem comprimento maior que n e qualquer decomposição da forma $a^{n+1}b^n = xyz$ requer que x e y só tenham as. Entretanto, tomando

$$x = a^i, \quad y = a^j \quad \text{e} \quad z = a^{n+1-i-j}b^n,$$

e bombeando y, obtemos

$$xy^kz = a^{n+1+(k-1)j}b^n$$

que pertence a L desde que $1 \geq (1-k)j$. Infelizmente, neste caso isto não leva a contradição nenhuma, a não ser que $j > 1$, e não temos como descartar a possibilidade de j ser exatamente 1.

A próxima linguagem requer uma escolha ainda mais sutil da palavra w.

Exemplo 4. Considere agora a linguagem

$$L_{uu} = \{uu : u \in \{0,1\}^*\}.$$

Como nos exemplos anteriores, é fácil descrever um argumento heurístico para justificar porque seria de esperar que L_{uu} não fosse regular, e deixaremos isto como exercício. Para provar a não regularidade de L_{uu} pelo lema do bombeamento, suporemos que esta linguagem é aceita por um autômato finito determinístico com n estados.

O principal problema neste caso é escolher uma palavra de comprimento maior que n que nos permita chegar facilmente a uma contradição. A escolha mais óbvia é $u = 0^n$, que, infelizmente, não leva a nenhuma contradição, como mostra o exercício 5. Felizmente uma variação simples desta palavra se mostra adequada, a saber $u = 0^n1$. Neste caso, $w = 0^n10^n1$ tem comprimento $2n + 2 > n$, e qualquer decomposição

$$0^n10^n1 = xyz$$

satisfaz

$$x = 0^i,\ y = 0^j \ \text{ e } \ z = 0^{n-i-j}10^n1$$

para algum $i \geq 0$ e $j \geq 1$. Bombeando y obtemos

$$xy^kz = 0^{n+(k-1)j}10^n1$$

Para saber se esta palavra pertence ou não a L_{uu} precisamos descobrir se pode ser escrita na forma vv para algum $v \in \{0,1\}^*$. Igualando

$$0^{n+(k-1)j}10^n1 = vv$$

concluímos que v tem que terminar em 1. Como só há um outro 1 na palavra,

$$v = 0^{n+(k-1)j}1 = 0^n1.$$

Isto é, $n + (k-1)j = n$. Como $j \neq 0$, esta igualdade só é verdadeira se $k = 1$. Mas isto contradiz o lema do bombeamento, segundo o qual xy^kz deveria pertencer a L_{uu} para todo $k \geq 0$.

Os exemplos anteriores mostram que a demonstração pelo lema do bombeamento de que uma certa linguagem L não é regular obedece a um padrão, que esboçamos abaixo:

Suponhamos, por contradição, que L seja aceita por um autômato finito determinístico com n estados.

- Escolha uma palavra $w \in L$, *de comprimento maior que* n, de modo que as possibilidades para uma decomposição da forma $w = xyz$ sejam bastante limitadas.
- Bombeie y e mostre que se $xy^kz \in L$ então uma contradição é obtida.

As principais dificuldades em fazer funcionar esta estratégia são as seguintes:

- a escolha de uma palavra w adequada;
- a identificação correta da condição que a pertinência $xy^kz \in L$ impõe sobre os dados do problema.

No exercício 7 temos um exemplo de demonstração em que vários erros foram cometidos na aplicação desta estratégia. Resolver este exercício pode ajudá-lo a evitar os erros mais comuns que surgem na aplicação do lema do bombeamento.

Infelizmente o lema do bombeamento está longe de ser uma panacéia infalível. Para ilustrar isto, vamos considerar mais um exemplo.

Exemplo 5. Seja L a linguagem no alfabeto $\{a, b, c\}$ formada pelas palavras da forma $a^ib^jc^r$ para as quais $i, j, r \geq 0$ e, se $i = 1$ então $j = r$. Mostraremos que a única palavra de L que não admite uma subpalavra bombeável é ϵ.

Há dois casos a considerar. No primeiro, $i \neq 1$ e j e r não são ambos nulos. Neste caso a subpalavra bombeável é $y = b$ se $j \neq 0$ ou $y = c$ se $r \neq 0$. O segundo caso consiste em supor que $i = 1$, ou que $i \neq 1$ mas $j = r = 0$. Desta vez podemos tomar $y = a$ como sendo a palavra bombeável.

Como cada palavra de L se encaixa em um destes dois casos, provamos que toda palavra de L admite uma subpalavra bombeável. Entretanto, esta linguagem não é regular. Assim, constatamos neste exemplo que:

- a recíproca do lema do bombeamento é falsa; isto é, não basta que o resultado do lema do bombeamento seja verdadeiro para que a linguagem seja regular;
- nem sempre o lema do bombeamento basta para mostrar que uma linguagem não é regular.

Provaremos que a linguagem acima de fato não é regular no capítulo ???. Para isto, além do lema do bombeamento, vamos precisar usar um resultado sobre a estabilidade das linguagens regulares por intersecção.

4. Exercícios

1. Considere o autômato finito determinístico $\mathcal{M}$ no alfabeto $\{0, 1\}$, com estado inicial q_1, conjunto de estados finais $\{q_5, q_6, q_8\}$ e função de transição δ dada pela seguinte tabela:

δ	0	1
q_1	q_2	q_6
q_2	q_6	q_3
q_3	q_4	q_2
q_4	q_2	q_5
q_5	q_5	q_5
q_6	q_7	q_4
q_7	q_8	q_4
q_8	q_4	q_5

(a) Esboce o diagrama de estados do autômato M.
(b) Ache uma subpalavra de 010011101000 que possa ser bombeada na linguagem $L(M)$.
(c) Seja $w = 00$. Verifique que $w, w^2 \in L(M)$. w é bombeável em $L(M)$?

2. Considere a linguagem L no alfabeto $\{0, 1\}$ dada por

$$L = \{10, 101^20, 101^201^30, 101^201^301^40, \dots\}.$$

(a) Mostre que L é infinita mas não admite nenhuma palavra que tenha uma subpalavra bombeável.
(b) Mostre que L não é regular.

3. Seja M um autômato finito determinístico e L a linguagem aceita por M. Vimos que para encontrar uma palavra de L que contém uma subpalavra bombeável basta encontrar um caminho no grafo de M que contenha um ciclo. Suponha agora que não conhecemos M, mas que conhecemos uma expressão regular r que denota L. De que forma podemos usar r para achar uma palavra de L que tem uma subpalavra bombeável?

4. Ache uma palavra que contenha uma subpalavra bombeável na linguagem denotada pela expressão regular

$$(1 \cdot 1 \cdot 0) \cdot (((1 \cdot 0)^* \cdot 0) \cup 0).$$

5. Considere a linguagem

$$L_{uu} = \{uu : u \in \{0, 1\}^*\}.$$

Mostre que, tomando $u = 0^n$, a palavra uu admite uma subpalavra bombeável em L_{uu}.
SUGESTÃO: Tome uma subpalavra de comprimento par.

6. Mostre que se L é uma linguagem regular infinita, então L admite pelos menos uma palavra que tem uma subpalavra bombeável.

7. Considere a linguagem

$$L = \{0^{2^n} : n \geq 0\}.$$

Determine os erros cometidos na demonstração abaixo de que L não é regular. Corrija estes erros e dê uma demonstração correta da não regularidade de L.

Suponha que L é aceita por um autômato finito determinístico. Seja $w = 0^{2^n}$. Pelo lema do bombeamento podemos decompor w na forma $w = xyz$, onde

$$x = 0^r, y = 0^s \quad \text{e} \quad z = 0^{2^n - r - s}.$$

Bombeando y obtemos

$$xy^k z = 0^{2^n + (k-1)s}.$$

Mas para que esta palavra pertença a L é preciso que $2^n + (k-1)s = 2^n$, o que só é possível se $(k-1)s = 0$. Como $s \neq 0$, concluímos que k só pode ser igual a 1, o que contradiz o lema do bombeamento.

8. Verifique quais das linguagens dadas abaixo são regulares e quais não são. Em cada caso justifique cuidadosamente sua resposta.
   (a) $\{0^i 1^{2i} : i \geq 1\}$;
   (b) $\{(01)^i : i \geq 1\}$;
   (c) $\{1^{2n} : n \geq 1\}$;
   (d) $\{0^n 1^m 0^{n+m} : n, m \geq 1\}$;
   (e) $\{1^{2^n} : n \geq 0\}$;
   (f) $\{w : w = w^r \text{ onde } w \in \{0,1\}\}$;
   (g) $\{wxw^r : w, x \in \{0,1\}^* \setminus \{\epsilon\}\}$.

   Se w é uma palavra em um alfabeto Σ então w^r é a palavra obtida invertendo-se a ordem das letras em w. Portanto se uma palavra satisfaz $w = w^r$ então é um palíndromo.

9. Uma palavra w no alfabeto $\{(,)\}$ é *balanceada* se:
   (a) em cada prefixo de w o número de (s não é menor que o número de)s e
   (b) o número de (s em w é igual ao número de)s.

   Isto é, w é balanceada se pode ser obtida a partir de uma expressão aritmética corretamente escrita pela omissão das variáveis, números e símbolos das operações. Mostre que a linguagem L que consiste nas palavras balanceadas no alfabeto $\{(,)\}$ não é regular.

10. Use o lema do bombeamento para mostrar que, se uma linguagem L contém uma palavra de comprimento maior ou igual a n e é aceita por um autômato finito determinístico com n estados, então L é infinita. Use isto para descrever um algoritmo que permite decidir se a linguagem aceita por um autômato finito determinístico dado é ou não infinita.

11. Seja M um autômato finito determinístico com n estados e seja L a linguagem aceita por M.
    (a) Use o lema do bombeamento para mostrar que se L contém uma palavra de comprimento maior ou igual que $2n$, então ela contém uma palavra de comprimento menor que $2n$.
    (b) Mostre que L é infinita se e somente se admite uma palavra de comprimento maior ou igual a n e menor que $2n$.
    (c) Descreva um algoritmo baseado em (3) que, tendo como entrada um autômato finito determinístico M, determina se $L(M)$ é finita ou infinita.

    SUGESTÃO: Para provar (a) use o lema do bombeamento com $k = 0$.

12. Seja $\mathcal{M}$ um autômato finito determinístico com n estados e um alfabeto de m símbolos.
    (a) Use (a) para mostrar que $L(\mathcal{M})$ é não vazia se e somente se contém uma palavra de comprimento menor ou igual a n.

(b) Explique como isto pode ser usado para criar um algoritmo que verifica se a linguagem de um autômato finito determinístico é ou não vazia.
(c) Suponha que a linguagem aceita por $\mathcal{M}$ é vazia. Quantas são as palavras que terão que ser testadas antes que o algoritmo de (b) possa chegar a esta conclusão? O que isto nos diz sobre a eficiência deste algoritmo?