Linguagens Formais e Autômatos

Expressões Regulares (ER)

Cristiano Lehrer, M.Sc.

Introdução

- Toda Linguagem Regular pode ser descrita por uma expressão simples, denominada Expressão Regular:
  - Trata-se de um formalismo denotacional, também considerado gerador, pois pode-se inferir como construir as palavras de uma linguagem;
  - É definida a partir de conjuntos (linguagens) básicos e operações de concatenação e união;
  - São adequadas para a comunicação homem x homem e, principalmente, para a comunicação homem x máquina.

Definição

- Uma Expressão Regular (ER) sobre um alfabeto Σ é indutivamente definida como segue:
  - $\emptyset$ é uma ER e denota a linguagem vazia;
  - ε é uma ER e denota a linguagem contendo exclusivamente a palavra vazia, ou seja, $\{\varepsilon\}$;
  - Qualquer símbolo x pertencente a Σ é uma ER e denota a linguagem contendo a palavra unitária x, ou seja, $\{x\}$;
  - Se r e s são ER e denotam as linguagens R e S, respectivamente, então:
    - (r+s) é ER e denota a linguagem $R \cup S$;
    - (rs) é ER e denota a linguagem $RS = \{uv \mid u \in R \text{ e } v \in S\}$;
    - (r*) é ER e denota a linguagem R^*.

Precedência

- A omissão de parênteses em uma ER é usual, respeitando as seguintes convenções:
  - A concatenação sucessiva tem precedência sobre a concatenação e a união:
    - $a^* \rightarrow aa \rightarrow a + a$
  - A concatenação tem precedência sobre a união:
    - $aa \rightarrow a + a$

Exemplos

Expressão Regular	Linguagem Representada
aa	somente a palavra aa
ba*	todas as palavras que iniciam por b, seguido por zero ou mais a
(a + b)*	todas as palavras sobre {a, b}
(a + b)*aa(a + b)*	todas as palavras contendo aa como subpalavra
a*ba*ba*	todas as palavras contendo exatamente dois b
(a + b)*(aa+ bb)	todas as palavras que terminam com aa ou bb
(a + ε)(b + ba)*	todas as palavras que não possuem dois a consecutivos

Construção

Para o caso de se trabalhar com prefixo/subpalavra/sufixo:

```
ER = (((prefixos)(alfabeto)(subpalavras)(alfabeto)(sufixos)) +
      ((sobreposições prefixos/subpalavras)(alfabeto)(sufixos)) +
      ((prefixos)(alfabeto)(sobreposições subpalavras/sufixos)) +
      (sobreposições prefixos/subpalavras/sufixos))
```

Exemplo

Desenvolva uma expressão regular sobre o alfabeto Σ = {1, 2, 3} que produza a linguagem L = {w | w possui *312* como prefixo, *211* como subpalavra e *121* como sufixo}.

```
ER = (((prefixos)(alfabeto)(subpalavras)(alfabeto)(sufixos)) +
      ((sobreposições prefixos/subpalavras)(alfabeto)(sufixos)) +
      ((prefixos)(alfabeto)(sobreposições subpalavras/sufixos)) +
      (sobreposições prefixos/subpalavras/sufixos))


ER = ((312 (1 + 2 + 3)* 211 (1 + 2 + 3)* 121) +
      (31211 (1 + 2 + 3)* 121) +
      (312 (1 + 2 + 3)* 21121) +
      (3121121))
```

Simplificação (1/2)

```
ER = ((312 (1 + 2 + 3)* 211 (1 + 2 + 3)* 121) +
      (31211 (1 + 2 + 3)* 121) +
      (312 (1 + 2 + 3)* 21121) +
      (3121121))
```

```
312 (1 + 2 + 3)* 211 (1 + 2 + 3)* 121
312               11 (1 + 2 + 3)* 121
312 (1 + 2 + 3)* 21               121
312               1               121
```

Simplificação (2/2)

312	(1 + 2 + 3)* 2	1	1 (1 + 2 + 3)*	121
312	ε	1	1 (1 + 2 + 3)*	121
312	(1 + 2 + 3)* 2	1	ε	121
312	ε	1	ε	121

ER = (312 (ε + ((1 + 2 + 3)* 2)) 1 (ε + (1 (1 + 2 + 3)*)) 121)

Exemplo (1/6)

Desenvolva uma expressão regular sobre o alfabeto Σ = {x, y, z} que produza a linguagem L = {w | w possui *xyz* ou *zyx* como prefixo, *zxy* ou *zyz* como subpalavra e *yxy* ou *yyz* como sufixo}.

```
ER = (((prefixos)(alfabeto)(subpalavras)(alfabeto)(sufixos)) +

      ((sobreposições prefixos/subpalavras)(alfabeto)(sufixos)) +

      ((prefixos)(alfabeto)(sobreposições subpalavras/sufixos)) +

      (sobreposições prefixos/subpalavras/sufixos))
```

Exemplo (2/6)

Desenvolva uma expressão regular sobre o alfabeto Σ = {x, y, z} que produza a linguagem L = {w | w possui *xyz* ou *zyx* como prefixo, *zxy* ou *zyz* como subpalavra e *yxy* ou *yyz* como sufixo}.

```
((prefixos)(alfabeto)(subpalavras)(alfabeto)(sufixos))

((xyz + zyx) (x + y + z)* (zxy + zyz) (x + y + z)* (yxy + yyz))
```

Exemplo (3/6)

Desenvolva uma expressão regular sobre o alfabeto Σ = {x, y, z} que produza a linguagem L = {w | w possui *xyz* ou *zyx* como prefixo, *zxy* ou *zyz* como subpalavra e *yxy* ou *yyz* como sufixo}.

`((sobreposições prefixos/subpalavras)(alfabeto)(sufixos))`

1. A sobreposição do prefixo *xyz* com a subpalavra *zxy* resulta na palavra *xyzxy*:

`(xyzxy (x + y + z)* (yxy + yyz))`

2. A sobreposição do prefixo *xyz* com a subpalavra *zyz* resulta na palavra *xyzyz*:

`(xyzyz (x + y + z)* (yxy + yyz))`

Exemplo (4/6)

Desenvolva uma expressão regular sobre o alfabeto Σ = {x, y, z} que produza a linguagem L = {w | w possui *xyz* ou *zyx* como prefixo, *zxy* ou *zyz* como subpalavra e *yxy* ou *yyz* como sufixo}.

`((prefixos)(alfabeto)(sobreposições subpalavras/sufixos))`

1. A sobreposição da subpalavra *zxy* com o sufixo *yxy* resulta na palavra *zxyxy*:

`((xyz + zyx) (x + y + z)* zxyxy)`

2. A sobreposição da subpalavra *zxy* com o sufixo *yyz* resulta na palavra *zxyyz*:

`((xyz + zyx) (x + y + z)* zxyyz)`

Exemplo (5/6)

Desenvolva uma expressão regular sobre o alfabeto Σ = {x, y, z} que produza a linguagem L = {w | w possui *xyz* ou *zyx* como prefixo, *zxy* ou *zyz* como subpalavra e *yxy* ou *yyz* como sufixo}.

`(sobreposições prefixos/subpalavras/sufixos)`

1. A sobreposição do prefixo/subpalavra *xyzxy* com a sobreposição da subpalavra/sufixo *zxyxy* resulta na palavra *xyzxyxy*:

`(xyzxyxy)`

2. A sobreposição do prefixo/subpalavra *xyzxy* com a sobreposição da subpalavra/sufixo *zxyyz* resulta na palavra *xyzxyyz*:

`(xyzxyyz)`

Exemplo (6/6)

Desenvolva uma expressão regular sobre o alfabeto Σ = {x, y, z} que produza a linguagem L = {w | w possui *xyz* ou *zyx* como prefixo, *zxy* ou *zyz* como subpalavra e *yxy* ou *yyz* como sufixo}.

```
ER = (((xyz + zyx) (x + y + z)* (zxy + zyz) (x + y + z)* (yxy + yyz)) +
      (xyzxy (x + y + z)* (yxy + yyz)) +
      (xyzyz (x + y + z)* (yxy + yyz)) +
      ((xyz + zyx) (x + y + z)* zxyxy) +
      ((xyz + zyx) (x + y + z)* zxyyz) +
      (xyzxyxy) +
      (xyzxyyz))
```

Simplificação

ER = (((xyz + zyx) (x + y + z)* (zxy + zyz) (x + y + z)* (yxy + yyz)) +

(xyzxy (x + y + z)* (yxy + yyz)) +

(xyzyz (x + y + z)* (yxy + yyz)) +

((xyzxy + xyzyz) (x + y + z)* (yxy + yyz)) +

((xyz + zyx) (x + y + z)* zxyxy) +

((xyz + zyx) (x + y + z)* zxyyz) +

((xyz + zyx) (x + y + z)* (zxyxy + zxyyz)) +

(xyzxyxy) +

(xyzxyyz))

(xyzxyxy + xyzxyyz))

Simplificação (2/2)

```
ER = (((xyz + zyx) (x + y + z)* (zxy + zyz) (x + y + z)* (yxy + yyz)) +
      ((xyzxy + xyzyz) (x + y + z)* (yxy + yyz)) +
      ((xyz + zyx) (x + y + z)* (zxyxy + zxyyz)) +
      (xyzxyxy + xyzxyyz))
```