Análise Léxica
Implementação do Autômato

Análise Léxica

Especificação dos *Tokens* através de Expressões Regulares (ERs)

Conversão das ERs para autômatos finitos com movimentos vazios (AFND-ε)

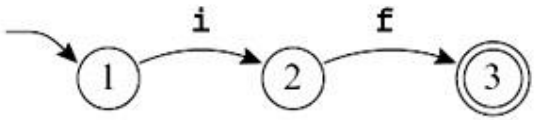
União dos vários AFND-ε em um único AFND-ε

Conversão do AFND-ε para um autômato finito determinístico (AFD)

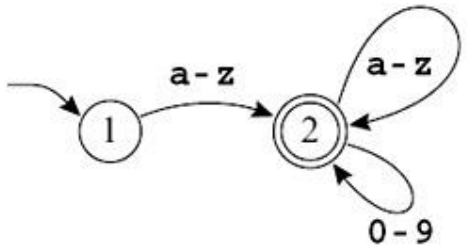
Minimização do AFD

Implementação do AFD em uma linguagem de programação

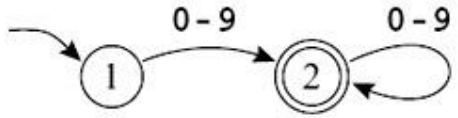
Autômatos Finitos: Um para cada *token*



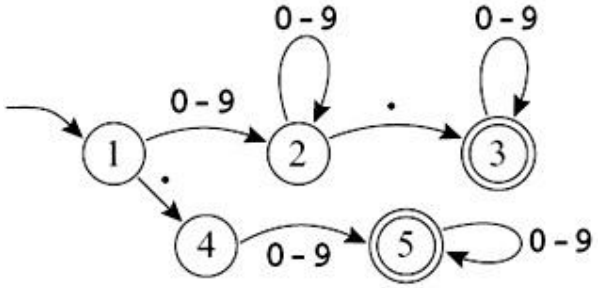
IF



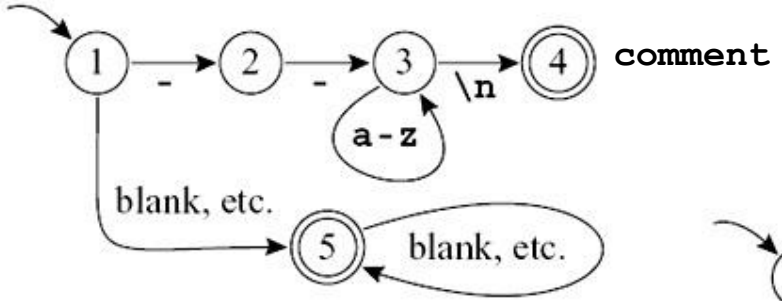
ID



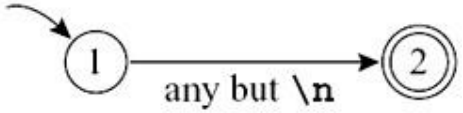
NUM



REAL

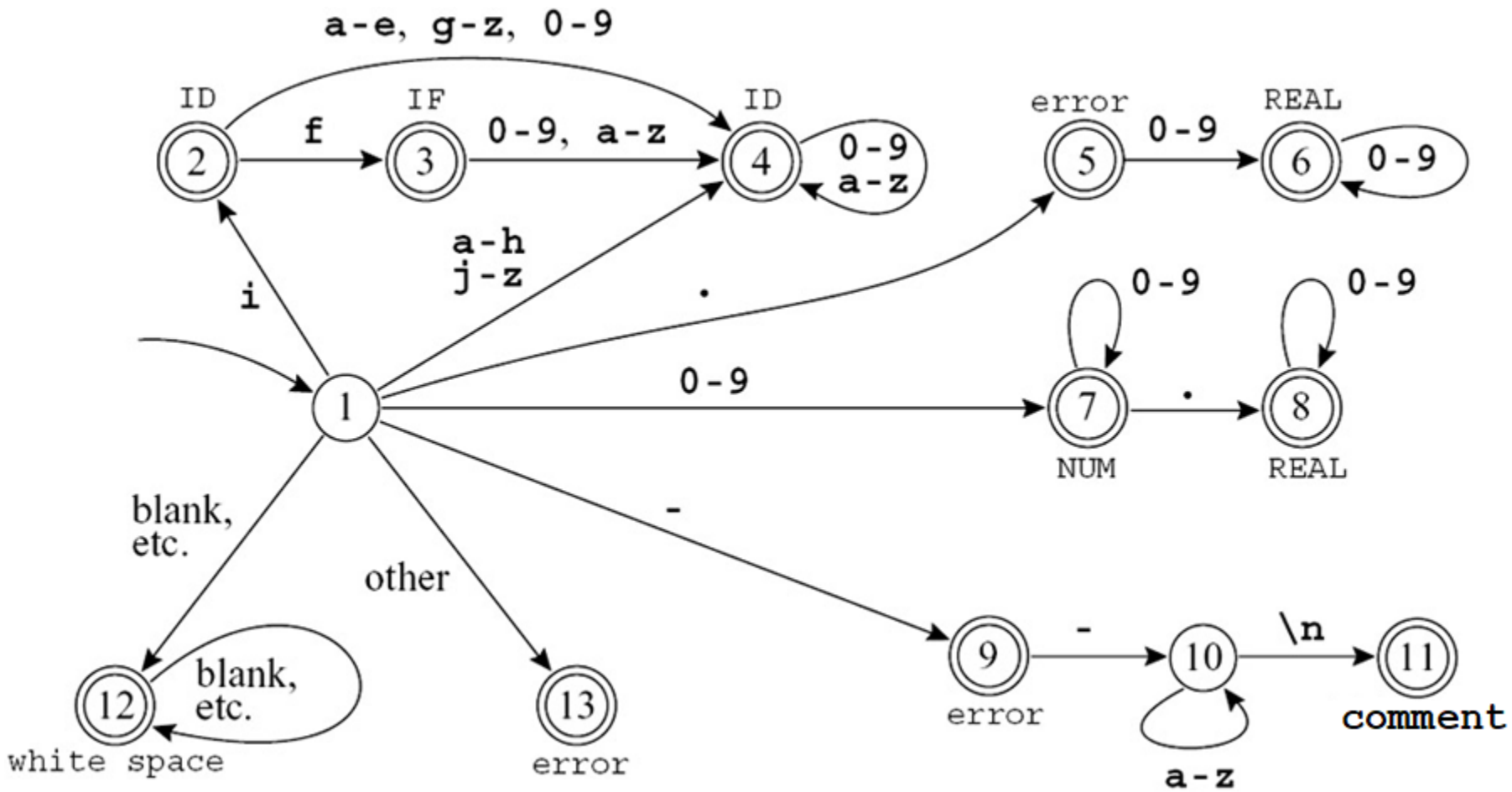


white space



error

Autômato Combinado

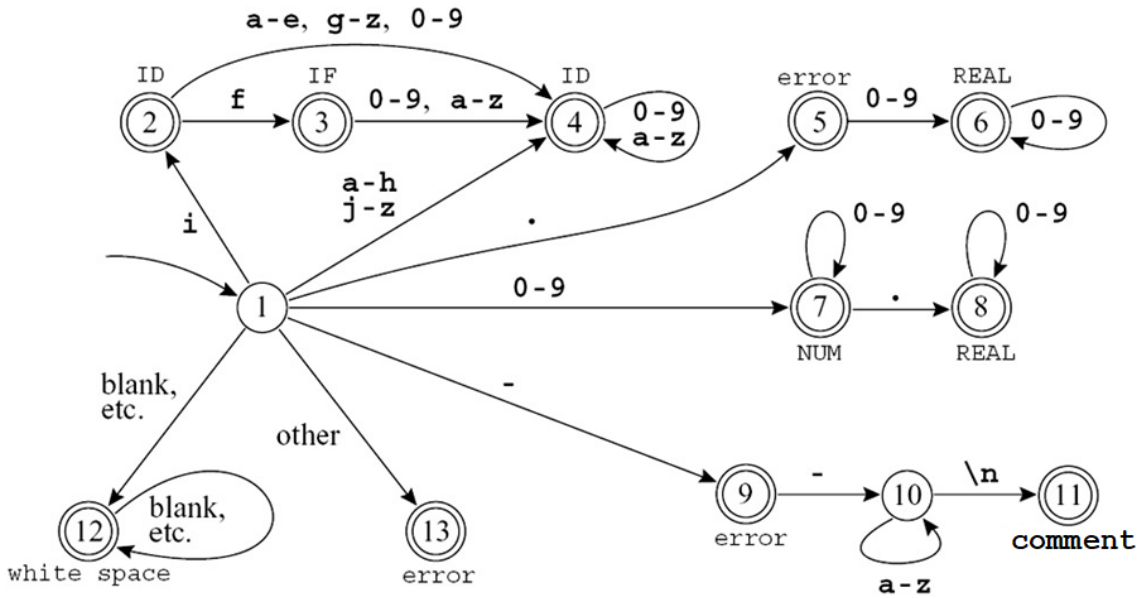


Autômato Combinado

```
int edges[][]={/*0 1 2...-...e f g h i j...m n...o...t...  */
  /* state 0 */ {0,0,0...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 1 */ {7,7,7...9...4,4,4,4,2,4...4,4...4...4...},
  /* state 2 */ {4,4,4...0...4,3,4,4,4,4...4,4...4...4...},
  /* state 3 */ {4,4,4...0...4,4,4,4,4,4...4,4...4...4...},
  /* state 4 */ {4,4,4...0...4,4,4,4,4,4...4,4...4...4...},
  /* state 5 */ {6,6,6...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 6 */ {6,6,6...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 7 */ {7,7,7...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 8 */ {8,8,8...0...0,0,0,0,0,0...0,0...0...0...},
    ...
};
```

entrada

Ausência de aresta



Reconhecimento da Maior SubString

A tabela anterior é usada para aceitar ou recusar uma *string*

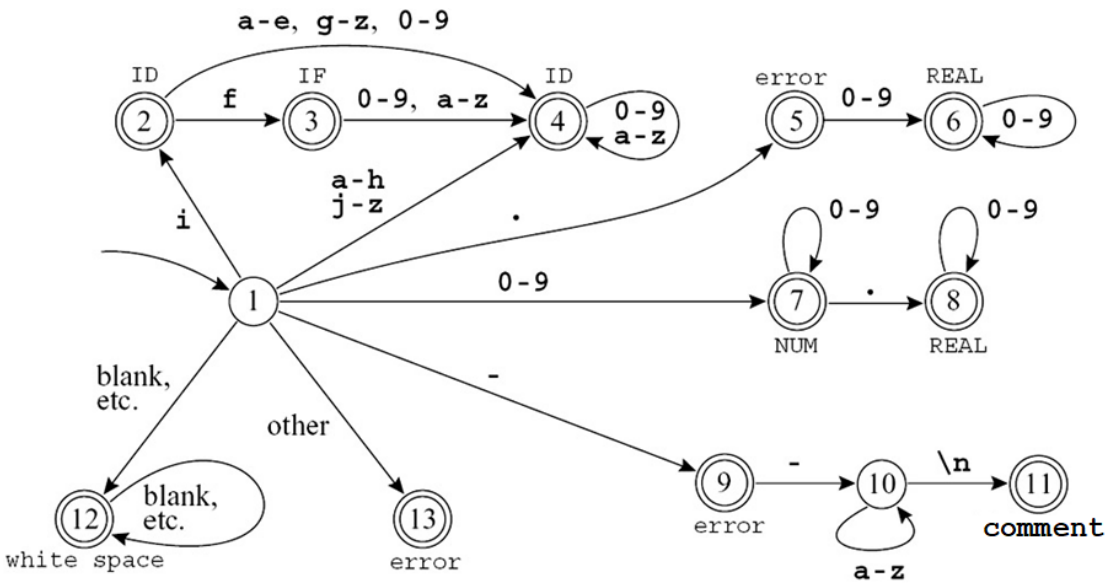
Porém, precisa-se garantir que a maior *string* seja reconhecida

Duas informações são necessárias:

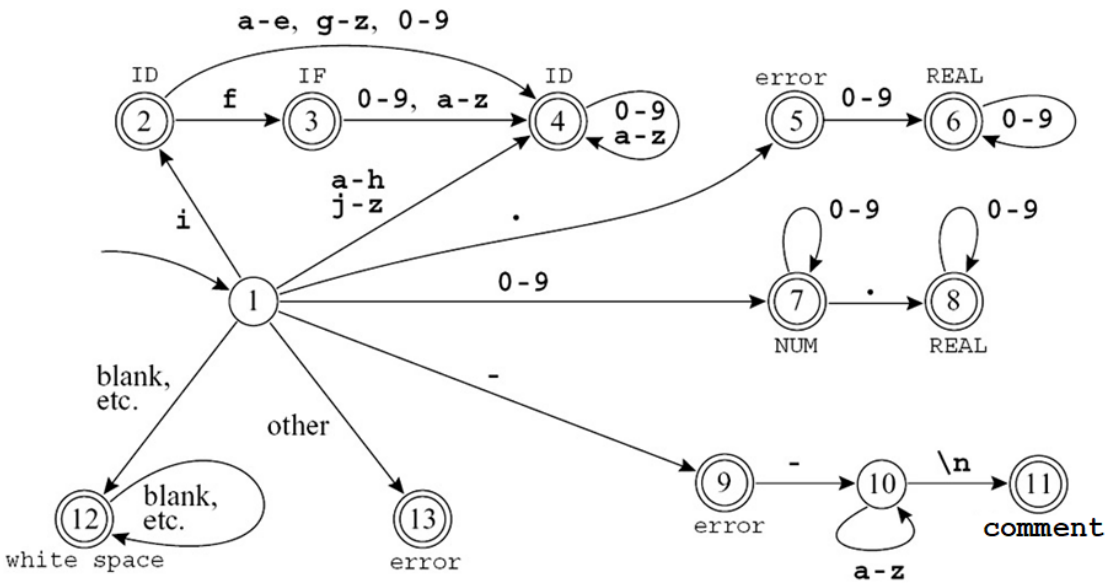
- Último estado final
- Posição da entrada no último estado final

```
int edges[][]={/*0 1 2...-...e f g h i j...m n...o...t... */
  /* state 0 */ {0,0,0...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 1 */ {7,7,7...9...4,4,4,4,2,4...4,4...4...4...},
  /* state 2 */ {4,4,4...0...4,3,4,4,4,4...4,4...4...4...},
  /* state 3 */ {4,4,4...0...4,4,4,4,4,4...4,4...4...4...},
  /* state 4 */ {4,4,4...0...4,4,4,4,4,4...4,4...4...4...},
  /* state 5 */ {6,6,6...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 6 */ {6,6,6...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 7 */ {7,7,7...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 8 */ {8,8,8...0...0,0,0,0,0,0...0,0...0...0...},
    ...
};
```

Last Final	Current State	Current Input	Accept Action
0	1	`if --not-a-com`	



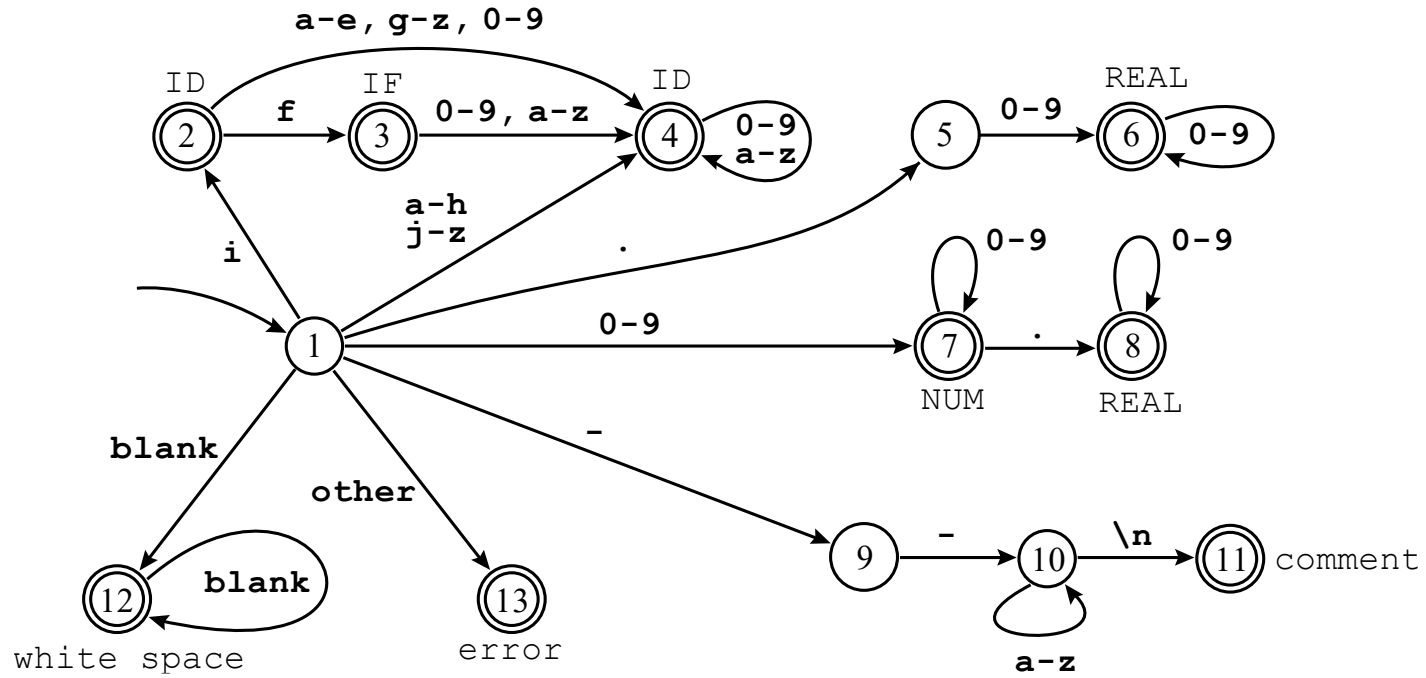
```
int edges[][]={/*0 1 2...-...e f g h i j...m n...o...t... */
  /* state 0 */ {0,0,0...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 1 */ {7,7,7...9...4,4,4,4,2,4...4,4...4...4...},
  /* state 2 */ {4,4,4...0...4,3,4,4,4,4...4,4...4...4...},
  /* state 3 */ {4,4,4...0...4,4,4,4,4,4...4,4...4...4...},
  /* state 4 */ {4,4,4...0...4,4,4,4,4,4...4,4...4...4...},
  /* state 5 */ {6,6,6...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 6 */ {6,6,6...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 7 */ {7,7,7...0...0,0,0,0,0,0...0,0...0...0...},
  /* state 8 */ {8,8,8...0...0,0,0,0,0,0...0,0...0...0...},
    ...
};
```



Last Final	Current State	Current Input	Accept Action
0	1	⌶if --not-a-com	
2	2	|i⌶f --not-a-com	
3	3	|if⌶ --not-a-com	
3	0	|if⊤⊥--not-a-com	*return* IF
0	1	if⌶ --not-a-com	
12	12	if| ⌶--not-a-com	
12	0	if| ⊤-⊥-not-a-com	*found white space; resume*
0	1	if ⌶--not-a-com	
9	9	if |-⌶-not-a-com	
9	10	if |-⊤-⌶not-a-com	
9	10	if |-⊤-n⊥ot-a-com	
9	10	if |-⊤-no⊥t-a-com	
9	10	if |-⊤-not⊥-a-com	
9	0	if |-⊤-not-⌶a-com	*error, illegal token '-'; resume*
0	1	if -⌶-not-a-com	
9	9	if -|-⌶not-a-com	
9	0	if -|-⊤⊥ot-a-com	*error, illegal token '-'; resume*

```
int edges[][]={/*0 1 2...9 - a...e f g h i j...m n...r...t...z */
  /* state 0 */ {0,0,0...0,0,0...0,0,0,0,0,0...0,0...0...0...0},
  /* state 1 */ {7,7,7...7,9,4...4,4,4,4,2,4...4,4...4...4...4},
  /* state 2 */ {4,4,4...4,0,4...4,3,4,4,4,4...4,4...4...4...4},
  /* state 3 */ {4,4,4...4,0,4...4,4,4,4,4,4...4,4...4...4...4},
  /* state 4 */ {4,4,4...4,0,4...4,4,4,4,4,4...4,4...4...4...4},
  /* state 5 */ {6,6,6...6,0,0...0,0,0,0,0,0...0,0...0...0...0},
  /* state 6 */ {6,6,6...6,0,0...0,0,0,0,0,0...0,0...0...0...0},
  /* state 7 */ {7,7,7...7,0,0...0,0,0,0,0,0...0,0...0...0...0},
  /* state 8 */ {8,8,8...8,0,0...0,0,0,0,0,0...0,0...0...0...0},
    ...
};
```

Last Final	Current State	Current Input	Accept Action
0	1	`if if8 @ 666`	



Lista de Exercícios

Lista 5

- Exercícios Teóricos e Práticos