Revisão P1
Linguagens Formais e Autômatos

Erickson G. Müller

1 Conteúdos

1. Conceitos e linguagens:
   - Símbolo
   - Alfabeto: conjunto de símbolos (terminais) utilizados para formar cadeias que pertencem a uma linguagem.
   - Cadeia
   - Sentença
   - Concatenação
   - Fechamento
   - Sufixo
   - Prefixo
   - Definição formal de gramática gerativa e derivação
   - Tipos de gramáticas
2. Conceitos e linguagens Símbolo, alfabeto, cadeia, sentença, concatenação, fechamento, sufixo, prefixo. Definição

2 Teoria de Linguagens

- Símbolo: qualidade e precedência
- Cadeia
- Tamanho de cadeia (caso cadeia vazia (ϵ))
- prefixo (próprio)
- sufixo (próprio)
- concatenação
- fechamento

1. V*
2. V+ = V* - EE

- linguagem finita
- linguagem infinita
- Reconhecedor: busca identificar se a sentença(ou cadeia, dependendo da apostila vs professor) pertence à linguagem ou não.
- sistema gerativo (gramática gerativa)

Computabilidade; Algoritmos e decibilidade; Soluções algoritmica decidíveis e indecidíveis. Processadores de linguagens (tradutores e interpretadores).

Para problemas computáveis podemos construir uma solução algoritmica que poderá ou não ter um resultado. Caso esse algoritmo não tenha um retorno, será considerado um algoritmo indecidível.

Esse conjunto de cadeias pode ser formado pela concatenação de símbolos que pertence ao alfabeto da linguagem.

Os processadores de linguagem verificam se a cadeia de símbolos **pertence** à linguagem. Os processadores de linguagens são soluções algoritmicas para **problemas decidíveis**. São divididos em tradutores e interpretadores.

Símbolos não têm significado intrínsico, a diferença entre eles é pela sua representação gráfica. O símbolo estará sob regras de substituição.

Os processadores de linguagens podem fazer transformações de linguagens de mesmo nível, podem passar de uma linguagem de código de uma máquina para outra máquina.

Tradutores recebem uma entrada chamada de fonte, fazem o processamento e resultam um objeto.

Nem todas as cadeias do fechamento do alfabeto são sentenças, todas as sentenças pertencem ao fechamento do alfabeto

Gramática: Conjunto de regras que descreve como as cadeias podem ser montadas e quando elas são válidas.

Sentença é toda cadeia de símbolos que pertence à linguagem.

Aplicando as regras da gramática, teremos sempre uma *cadeia ou sentença*

Análise Léxica
↓
Análise Sintática
↓
Análise Semântica
↓
Geração de Código Intermediário
↓
Otimização

Análise léxica faz o tratamento de erros, reconhecimento do tipo da variável através do identificador

Faz parte da saída da análise léxica uma estrutura chamada de fita que pertence o identificador dos tokens, essa fita é a entrada da análise sintática. O final dessa fita é um símbolo de final de sentença.

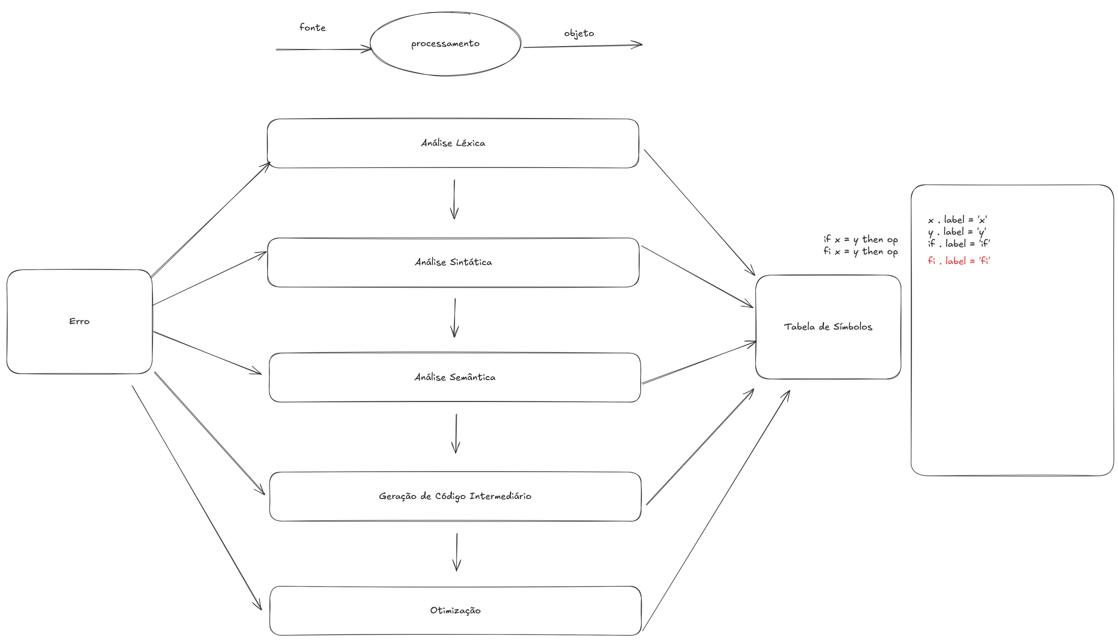
Análise sintática faz a análise da sentença usando o identificadores das cadeias que foram validadas na análise léxica. A sentença é avaliada através da concatenação de identificadores em uma determinada ordem.

A análise semântica verifica o tipo dos siímbolos e como estão sendo utilizados.

A geração de Código intermediário passa a sentença para código assembly.

Concatenação de símbolos de um alfabeto de uma linguagem, essa concatenação vai formar cadeias. Essas cadeias podem ser sentenças ou não. Caso a cadeia pertença à linguagem, é considerado uma sentença.

Fechamento do alfabeto de uma linguagem é a concatenação dos símbolos do alfabeto em qualquer ordem e em qualquer quantidade.



2.1 Definição Formal da Gramática

1. N: o conjunto de não terminais, ou variáveis.

2. T: é o conjunto de terminais, ou alfabeto. Sendo que N e T são conjuntos distintos.

3. P: é o conjunto distinto de produções, ou regras.
   Sendo que: $P \subset (N \cup T)^{+}.(N \cup T)^{*}$

4. S: é o símbolo não terminal que será o **símbolo inicial** de uma sentença

3 Definição formal de gramática gerativa e derivação

A única operação da gramática gerativa é a **substituição**.

4 Análise Sintática

Uma estrutura sintática de uma **gramática** é determinada dessa maneira:

$$< S > ::= < artigo >< substantivo >< verbo >$$

$< artigo > ::= a$
$< substantivo > ::= casa|coisa|lua$
$< verbo > ::= caiu|saiu$

Formalismo de Backus-Naur:

- $< substantivo > ::= casa$
- $< substantivo > ::= coisa$
- $< substantivo > ::= lua$
- $< verbo > ::= caiu$
- $< verbo > ::= saiu$
- $< artigo > ::= a$

Em seguida, temos a seguinte **produção**:
$< S > \rightarrow < artigo >< substantivo >< verbo >$
$< S > \rightarrow a < substantivo >< verbo >$
$< S > \rightarrow a - casa < verbo >$
$< S > \rightarrow a - casa - caiu$

Exercício:
Altere a gramática acima para que possa acrescentar as seguintes frases:

1. o cão saiu
2. o cofre fechou
3. o gato saiu
4. o gato caiu

$$< S > ::= < artigo >< substantivo >< verbo >$$

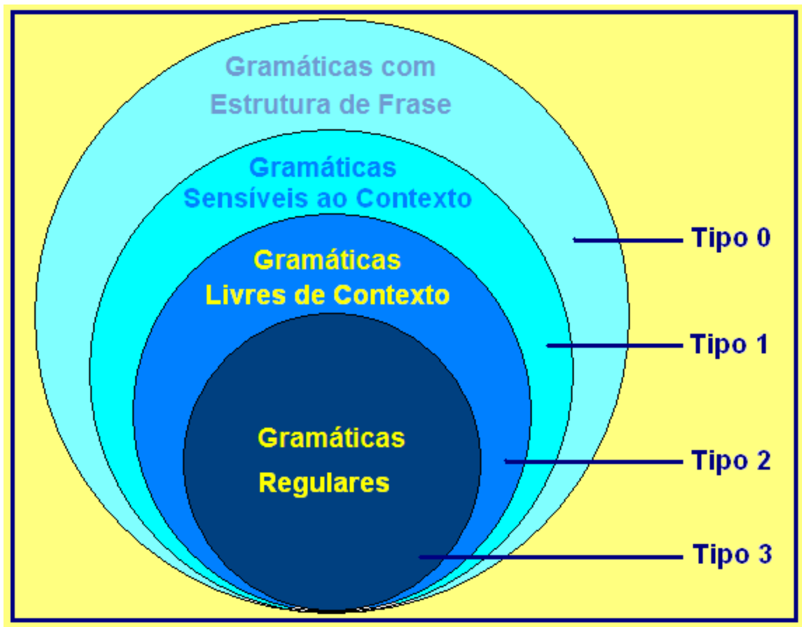
$< artigo > ::= a|o$
$< substantivo > ::= casa|coisa|lua|cao|cofre|gato$
$< verbo > ::= caiu|saiu|fechou$

Correção:
$< S > ::= o < A >$
$< A > ::= gato < B > |cao < C > |cofre < D >$
$< B > ::= < C > |caiu$
$< C > ::= saiu$
$< D > ::= fechou$

4.1 Derivação

Se $a \rightarrow b$, então a **produz** b
se $a \rightarrow b \rightarrow c \rightarrow ... \rightarrow z$ então $a \rightarrow^{*} z$ e a produz z pelo **fecho derivativo**
Então $a \rightarrow^{0} a$
$a \rightarrow^{1} b$ se $a \rightarrow b$

5 Hierarquia de Chomsky



Linguagens de tipo 3 (regulares) são resolvidas por gramáticas regulares

O formalismo que resolve problemas de linguagens regulares é o **autômato finito**

Linguagens do tipo 2 são resolvidas por gramáticas livre de contexto

O formalismo que resolve é o **autômato de pilha**

Gramáticas livre de contexto resolvem problemas do tipo 2 e do tipo 3.

Linguages do tipo 1 são linguagens sensíveis ao contexto e são resolvitos por gramáticas irrestritas.

Máquinas irrestritas resolvem problemas de linguagens do tipo 0 e do tipo 1.

5.1 Gramáticas Regulares (tipo 3)

Tem 3 formatos de produção possíveis, que é um não terminal que define o nome da regra e as produções podem ser nos formatos:
$A ::= a|aA|\epsilon$

5.2 Gramáticas Livres de Contexto

$A ::= (N \cup T)^*$

5.3 Gramáticas Sensíveis ao Contexto

Não inclui o Epsilon, portanto o menor tamanho das produções é 1 $(N \cup T)^{+} ::= (N \cup T)^{*}$, sendo que $|(N \cup T)^{+}| \leq |(N \cup T)^{*}|$.

A exceção à comparação dos tamanhos é quando. $< S >::= \epsilon$

$< S >$ não pode aparecer no lado direito de P.

5.4 Gramáticas Irrestritas

Não há restrição nenhuma, logo pode ser formado por qualquer concatenação: $(N \cup T)^{+} ::= (N \cup T)^{*}$

6 Exemplos

$L(g) = \{x | x \in (0,1)^{*}$, onde o número de ocorrências de 1 é ímpar$\}$

- Considerando que $A ::= a|aA|\epsilon$
- $< S >::= 0 < S > |1 < A >$ nessa regra o número de ocorrências de 1 é par., portanto estamos na iminência de substituir o S por uma de suas produções.
- $A ::= 0 < A > |1 < S > |\epsilon$ criamos essa regra para poder encerrar a frase. O S é chamado para impedir que o número de ocorrências do símbolo 1 seja par.

Agora,
$L(g) = \{x | x \in (0,1)^{+}$, onde o número de ocorrências de 1 é par$\}$

- Considerando que $A ::= a|aA|\epsilon$
- $< S >::= 0 < B > |1 < A >$ Tamanho da cadeia maior que 0
- $A ::= 0 < A > |1 < B >$
- $B ::= 0 < B > |1 < A > |\epsilon$

Outro,
$L(g) = \{x | x \in (0,1)^{+}$, onde o número de ocorrências de 0 é ímpar e 1 é par$\}$

- Considerando que $A ::= a|aA|\epsilon$
- $< S >::= 0 < C > |1 < B >$ Se 0 for par e 1 for par.
- $< B >::= 0 < D > |1 < S >$ Se 0 for par e 1 for ímpar.
- $< C >::= 0 < S > |1 < D > |\epsilon$ Se 0 for ímpar e 1 for par.
- $< D >::= 0 < B > |1 < C >$ Se 0 for ímpar e 1 for ímpar.

Outro,
$L(g) = \{x|x \in (0,1)^+$, onde $|x|$ é par se x inicia por 0, senão $|x|$ é ímpar$\}$

- $< S >::= 0 < A > |1 < C >$ Inicial
- $< A >::= 0 < B > |1 < B >$ Se inicia por 0 e $|x|$ é ímpar
- $< B >::= 0 < A > |1 < A > |\epsilon$ Se inicia por 0 e $|x|$ é par
- $< C >::= 0 < D > |1 < D >$ se inicia por 1 e $|x|$ é par
- $< D >::= 0 < C > |1 < C > |\epsilon$ se inicia por 1 e $|x|$ é ímpar

Correção do professor

- $< S >::= 0 < A > |1 < B >$ Inicial
- $< A >::= 0 < B > |1 < B >$ Se inicia por 0 e $|x|$ é ímpar ou se inicia por 1 e $|x|$ é par.
- $< B >::= 0 < A > |1 < A > |\epsilon$ Se inicia por 0 e $|x|$ é par ou se inicia por 1 e $|x|$ é ímpar.

Outro,
$L(g) = \{x|x \in (a,b,c)^+$, onde $|a| + |c|$ é par se x inicia por b, senão $|b|$ é ímpar. x não possui c concatenados. $\}$

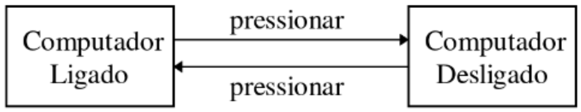
- $< S >::= a < D > |b < A > |c < E >$ Inicial
- $< A >::= a < B > |b < A > |c < C > |\epsilon$ se x inicia por **b** e $c + a$ é **par** e a anterior não é **c**.
- $< B >::= a < A > |b < B > |c < C >$ se x inicia por **b** e $c + a$ é **ímpar** e a anterior não é **c**.
- $< C >::= a < B > |b < C > |\epsilon$ se x inicia por **b** e $c + a$ é **par** e a anterior é **c**.
- $<>::=$ se x não inicia por **b** e $|b|$ é **ímpar**
- $< D >::= a < D > |b <> |c <>$ se x não inicia por **b** e $|b|$ é **par** e a anterior não é c
- $< D >::= a < D > |b <>$ se x não inicia por **b** e $|b|$ é **par** e a anterior é c

Correção do professor:

- $S ::= a < E > |b < A > |c < F >$
- $A ::= a < B > |b < A > |c < c > |\epsilon$ a+c é par e !c
- $B ::= a < A > |b < B > |c < D >$
-

7 Autômato Finito

Reconhecimento de padrões em processamento de texto (cadeias).



Podem ser determinísticos ou não determinísticos.
Se $w \in L$

- AFD
- AFND

Definição formal

- $K \rightarrow$ conjunto de estados.
- $\Sigma \rightarrow$ alfabeto da linguagem.
- $\delta \rightarrow \kappa_x \Sigma \rightarrow \kappa$.
- $q_o \rightarrow \in \kappa$ estado inicial.
- $F \rightarrow$ estado final.

Determinismo: $\delta(q, a) \rightarrow p$
Indeterminismo: $\delta(q, a) \rightarrow p|r|s$

$$\kappa_x \Sigma \rightarrow \kappa$$

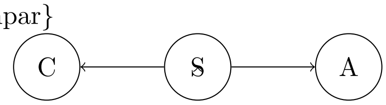
Estado —$^{\Sigma}$ Transição *a transição é rotulada por um símbolo do alfabeto da linguagem*
$\alpha \in L$ se $\delta(q_o, \alpha) \rightarrow^* p \wedge p \in F$ Alcançamos um estado p e esse estado é final.
$L(AF) = \{\alpha | \delta(q, \alpha) \rightarrow p \wedge p \in F\}$

7.1 exemplo

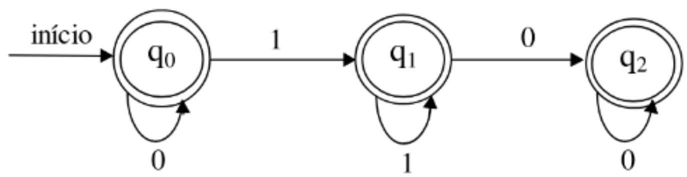
$L(G) = \{\alpha | \alpha \in (a, b)^+$ onde números de a's é ímpar$\}$
$S ::= a < A > |b < S > |a$
$A ::= a < S > |b < A > |\varepsilon$

7.2 Diagrama de transição e Tabela de transição

Considerando a gramática $L(G) = \{\alpha | \alpha \in (a, b)^+$ onde números de a's é ímpar$\}$



Forma Gráfica:



Forma Tabular:

	δ	0	1
$\rightarrow *$	q_0	q_0	q_1
$*$	q_1	q_2	q_1
$*$	q_2	q_2	-

no estado q^2, quando for 1 o tracinho significa que é um ε

7.3 Expressões regulares

$(0,1)^*$
Cadeias que terminam por 01

$$(0+1)^*.0.1$$

Cadeias com exatamente 2 0's

$$1^*01^*01^*$$

Cadeias com tamanho par

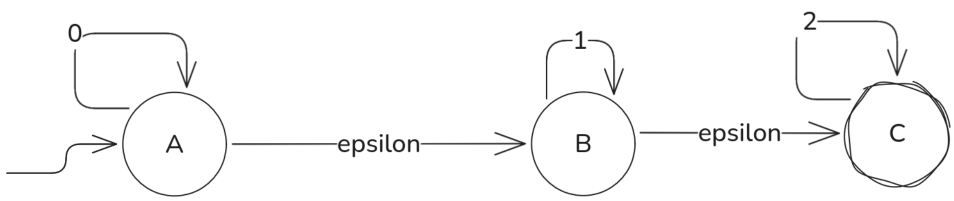
$$((0+1)(0+1))^*$$

Cadeias com tamanho ímpar

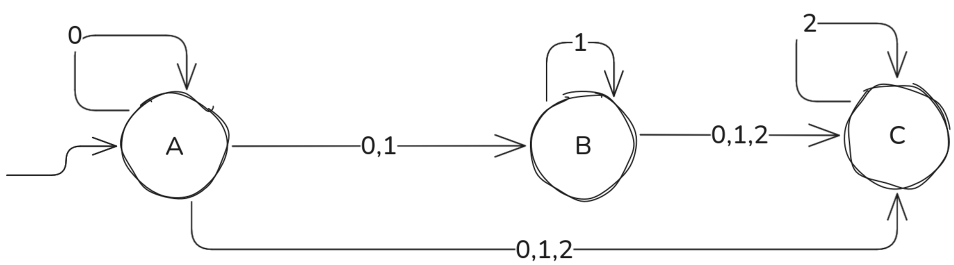
$$(0+1)((0+1)(0+1))^*$$

7.4 Eliminação de epsilon Transições

$A ::= 0 < A > | \varepsilon < B >$
$B ::= 1 < B > | \varepsilon < C >$
$C ::= 2 < C > | \varepsilon$



Conjuntos das produções
$A = \{A, B, C\}$
$B = \{B, C\}$
$C = \{C\}$
$A ::= 0 < A > |1 < B > |2 < C > |\varepsilon$
$B ::= 1 < B > |2 < C > |\varepsilon$
$C ::= 2 < C > |\varepsilon$



7.5 Transformação de AFND para AFD

$S ::= 0 < S > |1 < A > |A$
$A ::= 0 < C > |B$
$B ::= 1 < B > |0 < A > |0$
$C ::= 1 < A > |B$

Identificar os ε

$S ::= 0 < S > |1 < A > |\varepsilon < A >$
$A ::= 0 < C > |\varepsilon < B >$
$B ::= 1 < B > |0 < A > |0$
$C ::= 1 < A > |\varepsilon < B >$

Eliminar

$S ::= 0 < S > |1 < A > |0 < C > |1 < B > |0 < A > |\varepsilon$
$A ::= 0 < C > |1 < B > |0 < A > |\varepsilon$
$B ::= 1 < B > |0 < A > |\varepsilon$
$C ::= 1 < A > |1 < B > |0 < A > |\varepsilon$