UNIVERSIDADE DE SÃO PAULO
INSTITUTO DE CIÊNCIAS MATEMÁTICAS E DE COMPUTAÇÃO
http://www.icmc.usp.br
Departamento de Ciências de Computação

SCC-205 - Capítulo 3
Linguagens Sensíveis ao Contexto e Autômatos Limitados Linearmente

João Luís Garcia Rosa[1]

[1]Departamento de Ciências de Computação
Instituto de Ciências Matemáticas e de Computação
Universidade de São Paulo - São Carlos
http://www.icmc.usp.br/~joaoluis

2009

Sumário

Sumário

Definição

- Como uma generalização das regras livres de contexto, introduz-se agora regras sensíveis ao contexto que também especificam a substituição de um símbolo não terminal, mas requer um contexto para aplicação.
- Portanto a regra $bC \rightarrow bc$ é *sensível ao contexto* porque ela diz que um não terminal C pode ser substituído pelo símbolo terminal c apenas no contexto de um b precedente.
- **Definição**: Uma gramática G é **sensível ao contexto** se cada produção ou é da forma
  1. $yAz \rightarrow ywz$, para $A \in V$, $y, z \in (V \cup \Sigma)^*$, $w \in (V \cup \Sigma)^+$; ou
  2. $S \rightarrow \lambda$, dado que S não aparece no lado direito de nenhuma produção.
- Uma linguagem é sensível ao contexto se pode ser gerada por uma gramática sensível ao contexto.

Exemplo de GSC

- **Fato**: Seja G uma gramática tal que toda produção de G é da forma $v \rightarrow w$, com $|v| \leq |w|$, exceto que pode existir uma única produção-λ $S \rightarrow \lambda$ se S não aparece do lado direito de nenhuma produção. Então, existe uma gramática sensível ao contexto G' que é equivalente a G: $L(G) = L(G')$ [1].
- **Exemplo** de Gramática Sensível ao Contexto:
  - $G = (\Sigma, V, S, P)$, $\Sigma = \{a, b\}$, $V = \{S, B, C\}$ e $P = \{$
    1. $S \rightarrow aSBC$,
    2. $S \rightarrow aBC$,
    3. $CB \rightarrow BC$,
    4. $aB \rightarrow ab$,
    5. $bB \rightarrow bb$,
    6. $bC \rightarrow bc$,
    7. $cC \rightarrow cc\}$

Exemplo de GSC

- A linguagem $L(G)$ contém a palavra $a^n b^n c^n$ para cada $n \geq 1$, pois pode-se usar a produção (1) $n-1$ vezes para chegar a $S \Rightarrow^* a^{n-1}S(BC)^{n-1}$. Depois usa-se a (2) para $S \Rightarrow^* a^n(BC)^n$. A produção (3) permite arranjar os Bs e Cs tal que todo B preceda todos os Cs. Por exemplo, se $n = 3$:
$$aaaBCBCBC \Rightarrow aaaBBCCBC \Rightarrow aaaBBCBCC \Rightarrow aaaBBBCCC$$
- Portanto, $S \Rightarrow^* a^n B^n C^n$. Depois usa-se (4) uma vez para chegar a $S \Rightarrow^* a^n b B^{n-1} C^n$. Aí usa-se a produção (5) $n-1$ vezes: $S \Rightarrow^* a^n b^n C^n$. Finalmente, usa-se a produção (6) uma vez e a produção (7) $n-1$ vezes para chegar a $S \Rightarrow^* a^n b^n c^n$. É necessário mostrar também que as cadeias $a^n b^n c^n$, $n \geq 1$, são as únicas cadeias terminais em $L(G)$.

Sumário

Lema da Cadeia Vazia

- Note que nem toda gramática livre de contexto é sensível ao contexto, porque as produções livres de contexto podem ser da forma $A \to \lambda$. Entretanto, toda linguagem livre de contexto é uma linguagem sensível ao contexto.
- **Lema**: (**O Lema da Cadeia Vazia**). Seja G uma gramática livre de contexto que envolve regras da forma $A \to \lambda$ para qualquer $A \in V$. Então existe uma gramática livre de contexto G' tal que
  1. $L(G) = L(G')$;
  2. Se $\lambda \notin L(G)$ então não existem produções da forma $A \to \lambda$ em G'; mas
  3. Se $\lambda \in L(G)$, então existe uma única produção-λ em G', $S' \to \lambda$, onde S' é o símbolo inicial de G' e S' não aparece no lado direito de nenhuma produção em G'.

Sumário

Lema da Cadeia Vazia: Prova

- **PROVA**: Suponha que $\lambda \notin L(G)$. Para $C \in V$ considere todas as produções $C \to w$, onde w não é vazio. Agora suponha que w contém variáveis $A_1, ..., A_k, B_1, ..., B_j$ onde A_i, $1 \leq i < k$, são aquelas variáveis em w para as quais $A_i \Rightarrow^* \lambda$. Adicione a G a produção $C \to w'$, onde w' é obtida de w apagando zero ou mais ocorrências de uma variável dos A_i's. Então, por exemplo,

$$C \to aA_1BA_2$$

leva a quatro produções

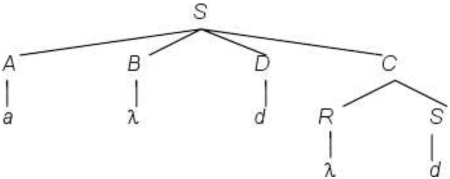
$$C \to aA_1BA_2 \mid aBA_2 \mid aA_1B \mid aB$$

Quando este processo estiver completo, remova todas as produções-λ, incluindo qualquer produção-λ nova que tenha sido incluída no processo. A gramática resultante é G'. Então $L(G) = L(G')$ como se segue.

Lema da Cadeia Vazia: Prova

- Se uma derivação de alguma cadeia $w \in L(G)$ envolveu produções-λ, então a árvore de derivação para w terá nós rotulados com λ, como no seguinte exemplo.

$S \rightarrow ABDC$
$C \rightarrow RS$
$B \rightarrow \lambda$
$A \rightarrow a$
$D \rightarrow d$
$R \rightarrow \lambda$
$S \rightarrow d$



- Para obter uma derivação G' para w a partir de uma derivação G para w, comece no topo da árvore G e apague qualquer sub-árvore da raiz da árvore que deriva λ. O topo da árvore deve agora ilustrar uma produção a partir de G'.

Lema da Cadeia Vazia: Prova

- Então considere os sucessores imediatos do nó raiz que não derivam λ. Aplique o mesmo princípio a cada um deles: se alguma das suas sub-árvores deriva λ, apague a sub-árvore. Continue desta forma para baixo na árvore. A árvore final representa uma derivação G'. Além disto, esta árvore deriva w, porque uma subcadeia de w é removida pelo processo de apagamento se e somente se ela for λ. Por outro lado, se $w \in L(G')$ e a derivação de w envolve uma regra recém adicionada, então simule a derivação de G' usando a regra G original apropriada e regras-λ. Isto é, faça o processo descrito acima na ordem reversa, novamente de cima para baixo.

Lema da Cadeia Vazia

- No caso onde $\lambda \in L(G)$, primeiro adicione um novo símbolo inicial a G', por exemplo S', e adicione duas novas produções, $S' \rightarrow S$ e $S' \rightarrow \lambda$. Depois repita o processo descrito previamente em todas as produções exceto as produções S'. Então $L(G) = L(G')$.

- **Teorema**: Uma linguagem L é *sensível ao contexto* se e somente se existe alguma gramática G tal que $L = L(G)$ onde toda produção de G da forma $u \rightarrow v$ tem a propriedade de que $0 < |u| \leq |v|$ com uma exceção: se $\lambda \in L(G)$, então a regra $S \rightarrow \lambda$ está também presente e neste caso S não pode aparecer no lado direito de nenhuma produção.

Sumário

Máquina de Turing

- Vai-se relacionar agora a teoria das linguagens formais com a teoria abstrata da computação.
- Como se pode caracterizar o conjunto de funções computadas por programas de computador?
- Esta questão - quais funções são realizáveis por algoritmos e quais não são? - tem suas raízes mais diretas no trabalho de Alan Turing nos anos 1930 [6].
- Usando, o que agora se chama de modelo de máquina de Turing, Turing mostrou que certos problemas naturais em computação não podem ser computados por nenhum algoritmo, real ou imaginado.

Máquina de Turing

- Realmente, Turing mostrou apenas que estes problemas não são calculáveis especificamente por máquinas de Turing; mais tarde as investigações de outros pesquisadores levaram a crença geral que a computabilidade de Turing é sinônimo da computabilidade em qualquer outro sistema de especificação de algoritmo suficientemente poderoso.
- Introduz-se as máquinas de Turing apresentando os principais resultados de Turing, a universalidade das máquinas de Turing e a insolubilidade do problema da parada (*halting*).
- Na sua forma mais geral, o último resultado diz que nenhum algoritmo pode corretamente decidir se um programa de computador arbitrário pára em uma entrada arbitrária.

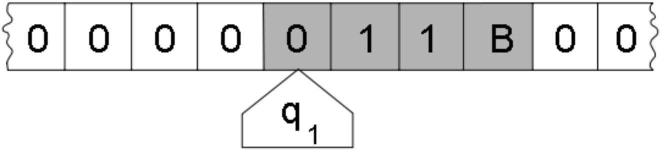
Máquina de Turing

- Vai-se mostrar como codificar este resultado na teoria da linguagem.
- Usando esta codificação prova-se vários resultados, incluindo a asserção de que nenhum algoritmo pode corretamente decidir se uma gramática livre de contexto arbitrária é ambígua.

Máquina de Turing: Definição

- Uma máquina de Turing T pode ser descrita (Figura 3) como um controle de estados finitos equipado com um dispositivo de armazenamento externo na forma de uma fita finita que pode ser estendida indefinidamente em ambas as direções.

Figure: Uma máquina de Turing: A fita pode ser estendida indefinidamente pela adição de B's (brancos) em qualquer lado [7].



Máquina de Turing: Definição

- Como mostrado na Figura 3, a fita é dividida em quadrados.
- Cada quadrado da fita pode estar em branco (B) ou pode carregar qualquer símbolo de um alfabeto de fita finito Σ especificado.
- Por conveniência, um símbolo especial (aqui, a letra B) não em Σ é reservado para denotar um quadrado em branco na fita.
- O controle de estados finitos é acoplado à fita através de uma cabeça de leitura/escrita.
- A qualquer instante dado, a cabeça percorre um quadrado da fita e o controle de estados finitos está em um estado.

Máquina de Turing: Definição

- Dependendo deste estado e do símbolo no quadrado percorrido pela cabeça da fita, a máquina irá, em um passo, fazer o seguinte:
  1. Entrar em um novo estado do controle de estados finitos;
  2. Sobrescrever um símbolo no quadrado percorrido (é possível sobrescrever o mesmo símbolo e portanto deixar o quadrado da fita sem mudar, ou sobrescrever um B e "apagar" o símbolo da fita);
  3. Deslocar a cabeça para esquerda ou para a direita um quadrado.
- Como o alfabeto da fita é finito e o controle de estados finitos tem finitamente muitos estados, a operação de uma máquina de Turing pode ser completamente descrita por um conjunto finito de quíntuplas da forma *(estado antigo, símbolo percorrido, estado novo, símbolo escrito, direção de movimento)*.

Máquina de Turing: Exemplo

- **Exemplo** : A seguinte máquina de Turing aceita o conjunto $\{a^{2n} | n \geq 0\}$ de cadeias de comprimento par de a's: quando começa no estado "par" no a mais a esquerda, com a configuração de fita inicial $Baaa...aaaB$ ela irá terminar no estado "aceitação" apenas no caso do número de a's ser par. Aqui está a máquina:

$$\begin{gathered}(\text{par } a \text{ ímpar } B\ R)\\(\text{ímpar } a \text{ par } B\ R)\\(\text{par } B \text{ aceitação } B\ R)\end{gathered}$$

- Quando ela alcança primeiro um B ela muda para o estado "aceitação" se ela viu um número par (incluindo 0) de a's mas de outra forma ela irá parar porque não há quíntupla para guiar seu próximo movimento.

Máquina de Turing: Exemplo

- Por outro lado, a máquina com quíntuplas:

$$(\text{par } a \text{ ímpar } B\ R)$$
$$(\text{ímpar } a \text{ par } B\ R)$$
$$(\text{par } B \text{ aceitação } B\ R)$$
$$(\text{ímpar } B \text{ ímpar } B\ R)$$

- tem a propriedade de que em cadeias de comprimento par de *a*'s ela pára no estado de aceitação, mas em cadeias de comprimento ímpar ela "roda" para sempre.
- Isto ilustra como as máquinas de Turing são realmente diferentes dos Autômatos Finitos (AFDs).
- Em particular, as computações da máquina de Turing podem fornecer resultados indefinidos - computações que nunca terminam.
- Tal comportamento não é possível com AEFs.

Máquina de Turing

- Assim como com os AEFs, pode-se dar uma representação equivalente das quíntuplas no Exemplo 8 usando uma tabela de estado, como se segue:

estado antigo	símbolo percorrido	
	a	B
par	ímpar B R	aceitação B R
ímpar	par B R	

- A fim de tornar o comportamento da máquina determinístico, necessita-se que quando duas quíntuplas têm a mesma combinação (estado antigo, símbolo percorrido), então as duas quíntuplas são idênticas.
- Veja agora a definição formal de uma máquina de Turing e a linguagem que ela aceita.

Máquina de Turing: Definição

- **Definição**: Uma máquina de Turing M é uma quíntupla $M = (Q, \Sigma, q_0, q_a, \delta)$, onde
  1. Q é um conjunto finito de estados;
  2. Σ é um alfabeto finito da fita. Escreve-se $\Sigma' = \Sigma \cup \{B\}$ para Σ aumentado por um símbolo de fita distinto B, o símbolo branco;
  3. q_0 é o estado inicial distinto;
  4. q_a é o estado de aceitação distinto;
  5. δ é a função (parcial) de transição de estado,
     $$\delta : Q \times \Sigma' \rightarrow Q \times \Sigma' \times \{L, R, S\}$$
     sujeita a condição de que $\delta(q_a, x)$ não é definida para nenhum x em Σ'. Pode-se representar δ por uma lista de quíntuplas, com $(q\ x\ q'\ x'\ D)$ na lista apenas no caso em que $\delta(q, x) = (q', x', D)$.

Máquina de Turing: Descrição Instantânea

- Pode-se descrever a configuração de uma máquina de Turing M especificando:
  1. a cadeia w_1 impressa na fita a esquerda da cabeça de leitura/escrita;
  2. o estado corrente q; e
  3. a cadeia w_2 na fita, começando na cabeça de leitura/escrita.
- Pode-se resumir isto na cadeia

$$w_1 q w_2$$

  que é chamada de **descrição instantânea** (DI) de M.
- Note que w_1 e w_2 não são únicos - a cadeia $w_1 w_2$ deve conter todos os quadrados não brancos da fita de M, mas ela pode ser estendida adicionando brancos em qualquer lado: $1q_0 1B11$ e $BB1q_0 1B11BBB$ são duas formas de escrever a mesma DI.

Máquina de Turing: Descrição Instantânea

- As máquinas de Turing têm fitas, cabeças de leitura/escrita, etc., e não podem obviamente ser descritas usando a maquinaria da teoria da linguagem.
- As descrições instantâneas, por outro lado, são realmente cadeias de símbolos e como tais podem ser manipuladas por uma gramática formal.
- Pode-se ligar a teoria da linguagem e a noção de computabilidade da máquina de Turing, o que será feito mais tarde nesta parte, mostrando como uma gramática reescrevendo uma descrição instantânea pode confiavelmente seguir a trilha das transições de uma computação da máquina de Turing.

Máquina de Turing: Descrição Instantânea

- **Definição**: Seja M uma máquina de Turing com conjunto de estados Q e alfabeto de fita aumentado Σ'. Então uma **descrição instantânea** (DI) de M é uma cadeia w_1qw_2 do conjunto $(\Sigma')^* \cdot Q \cdot (\Sigma')^*$. Cada DI w_1qw_2 é equivalente às DIs Bw_1qw_2 e w_1qw_2B.
- Diz-se que M pára em w_1qw_2 se nem w_1qw_2 nem w_1qw_2B for da forma w_1qxw para um par (q, x) para o qual $\delta(q, x)$ é definido.
- Se M não pára para w_1qw_2, define-se como as DIs são transformadas pela ação de M.
- Primeiro, escreve-se w_1qw_2 como $w_{1A}yqxw_{2A}$ onde $w_{1A}y = w_1$ se $w_1 \neq \lambda$, e $w_{1A} = \lambda$, $y = B$ caso contrário; e $xw_{2A} = w_2$ se $w_2 \neq \lambda$, e $x = B$, $w_{2A} = \lambda$ caso contrário.

Máquina de Turing: Descrição Instantânea

- Define-se então a relação próxima DI
$$w_1 q w_2 \Rightarrow w_1' q' w_2'$$
como se segue:
  1. se $\delta(q, x) = (q', x', L)$ então
  $$w_1' q' w_2' = w_{1A} q' y x' w_{2A}$$
  2. se $\delta(q, x) = (q', x', S)$ então
  $$w_1' q' w_2' = w_{1A} y q' x' w_{2A}$$
  3. se $\delta(q, x) = (q', x', R)$ então
  $$w_1' q' w_2' = w_{1A} y x' q' w_{2A}.$$
- Como de costume, estende-se $\Rightarrow$ para formar seu fechamento reflexivo e transitivo $\Rightarrow^*$ tal que
$$\alpha \Rightarrow^* \alpha'$$
significa que M pode transformar a DI α na DI α' através de 0 ou mais transições de DI sucessoras como descrito em (1)-(2) anteriormente.

Sumário

Máquina de Turing

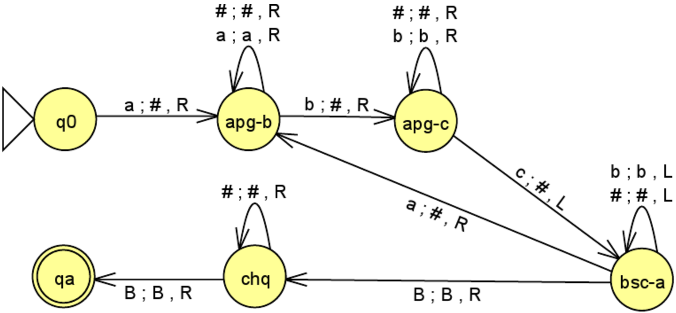
- Para uma máquina de Turing M sobre o alfabeto Σ, define-se $T(M)$, o conjunto aceitação de M, como o conjunto das cadeias w sobre Σ tal que se w é escrita em uma fita em branco e M começa processando no estado q_0 no símbolo mais a esquerda de w, então M eventualmente pára no estado de aceitação. Formalmente:
- **Definição**: O conjunto de aceitação $T(M)$ de uma máquina de Turing M sobre o alfabeto Σ é o conjunto

$$T(M) = \{w \in \Sigma^* \mid \exists\, w_1, w_2 \in (\Sigma')^* \mid q_0 w \Rightarrow^* w_1 q_a w_2\}.$$

Máquina de Turing: Exemplo

- **Exemplo**: A seguinte máquina de Turing aceita a linguagem $\{a^n b^n c^n \mid n > 0\}$:

Figure: Uma máquina de Turing para processar a linguagem $\{a^n b^n c^n \mid n > 0\}$.



Sumário

ALL: Definição

- Vai-se considerar agora os **autômatos limitados linearmente**.
- Estes dispositivos são máquinas de Turing não determinísticas com a restrição de que a cabeça de leitura/escrita não tem a permissão de deixar os quadrados da fita que possuem a entrada original.
- Mostrar-se-á que estas máquinas são precisamente os aceitadores para linguagens sensíveis ao contexto.
- **Definição**: Um autômato limitado linearmente (ALL) é uma máquina de Turing não determinística que deve operar dentro do espaço definido pela colocação original da entrada na fita.

Sumário

O Lema do Alfabeto

- **Lema do Alfabeto**: Seja L uma linguagem tal que para qualquer $w \in \Sigma^*$, a pertinência de w em L pode ser decidida por uma máquina de Turing M usando $k \cdot |w|$ quadrados da fita, onde k é um inteiro positivo independente de w. Então L pode ser reconhecida por alguma outra máquina de Turing M' usando apenas $|w|$ quadrados da fita.
    - Então o lema do alfabeto estabelece que se alguma máquina de Turing pode reconhecer uma linguagem L no espaço que é uma função linear do comprimento de uma cadeia de entrada, então alguma outra máquina de Turing pode de fato reconhecer L dentro do espaço exatamente igual do comprimento da entrada.
    - O relacionamento linear identificado no lema justifica o nome autômatos limitados linearmente.

Limites à Esquerda e à Direita

- Assume-se, quando conveniente, que a entrada da máquina de Turing é limitada a esquerda por um símbolo \$ e a direita por um ¢. Assume-se que estes símbolos são não apagáveis e que a cabeça não pode mover sobre eles.
- **Teorema**: Se $L = T(M)$ para algum ALL M, então L é gerável por alguma gramática sensível ao contexto.
- PROVA: Vai-se assumir que $\lambda \notin L$. Para $\lambda \in L$ usa-se o teorema para $L - \{\lambda\}$ e com cuidado adiciona-se uma produção $S \rightarrow \lambda$ depois do fato. Para estabelecer o resultado assumindo $\lambda \notin L$, primeiro escreve-se uma GSC para a linguagem

$$\{w\$q_0w \mid w \in \Sigma^*\}$$

Limites à Esquerda e à Direita

- A cadeia a direita do símbolo \$ é de comprimento $|w| + 2$. Já que a máquina neste caso é um ALL, as DIs nunca serão maiores que isto. Entretanto, no término da simulação de gramática de M, não se pode apagar todos os símbolos a direita de \$, porque as regras de apagamento não são sensíveis ao contexto. Ao invés disto, se q_a aparece, faz-se o seguinte:
  1. Primeiro, converte-se todos os símbolos a direita de \$ para um novo símbolo, "/", tal que fica-se com a seguinte cadeia: $w\$/^n q_a$
  2. Se $w = x_1 x_2 ... x_n$, usa-se regras sensíveis ao contexto para construir a seguinte cadeia: $r/x_1/x_2/.../x_n\#q$ (onde r e q são novos símbolos terminais);
  3. Apaga-se todas as ocorrências de r, $/$, $\#$ e q.

Bibliografia I

[1] Chomsky, N.
On Certain Formal Properties of Grammars.
Information and Control, 2, 1959, pp. 137–167.

[2] Hopcroft, J. E., Ullman, J. D.
Formal Languages and Their Relation to Automata.
Addison-Wesley Publishing Company, 1969.

[3] Hopcroft, J. E., Ullman, J. D. e Motwani, R.
Introdução à Teoria de Autômatos, Linguagens e Computação.
Tradução da segunda edição americana. Editora Campus, 2003.

Bibliografia II

[4] Moll, R. N., Arbib, M. A., and Kfoury, A. J.
An Introduction to Formal Language Theory.
Springer-Verlag, 1988.

[5] Rosa, J. L. G.
Linguagens Formais e Autômatos.
Notas de Aula. Engenharia de Computação. Pontifícia Universidade Católica de Campinas, 2007.

[6] Turing, A.M.
On Computable Numbers, with an Application to the Entscheidungsproblem.
Proceedings of the London Mathematical Society, 2 42: 230-65, 1937.

Bibliografia III

[7] Wikipedia - The Free Encyclopedia.
http://en.wikipedia.org/wiki/Turing_machine