3 O PROCESSAMENTO DA ESPECIFICAÇÃO DA LINGUAGEM NATURAL

No capítulo 2, estudou-se como o formalismo adaptativo permite representar de forma parentetizada e etiquetada estruturas sintáticas justapostas, subordinadas ou coordenadas. Esta representação é equivalente a uma árvore. Tais estruturas assim especificadas são passíveis de serem pesquisadas para a análise sintática de uma determinada cadeia de entrada. O processamento da busca, de uma informação além de ser automatizada (de um escopo mais interno para o escopo mais externo), apresenta um desempenho no tempo proporcional ao número de escopos somado ao dobro do número de símbolos (elementos justapostos) que constituem a informação. Há que se recordar que as técnicas empregadas por (NETO, 1993) não dependem da natureza dos símbolos. Considere-se a figura 9.

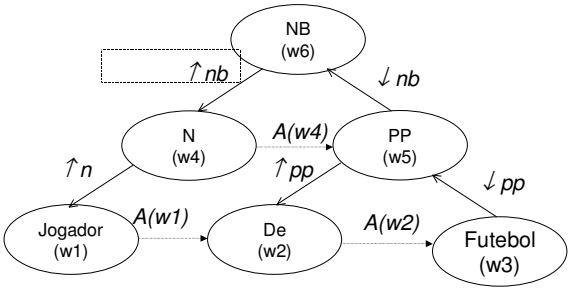


Fig. 9 – Estrutura aprendida no modo "treinamento" e utilização posterior

Nesta figura, apresenta-se esquematicamente (os detalhes foram apresentados no capítulo anterior), como uma estrutura pode ser "aprendida", a partir da extração dos símbolos nb, n e pp,(representada pelo símbolo "↑") e a sua utilização no modo uso, para a análise de

uma cadeia de entrada . As **inferências** são anotadas na cadeia de entrada e são representadas pela inserção dos mesmos símbolos. ,(representada pelo símbolo "↓")

Observe-se que na figura 9 a representação de um grupo preposicional PP está simplificada. Na verdade, o grupo preposicional é ele mesmo uma estrutura.

Os mecanismos aqui apresentados permitem a representação e tratamento de estruturas sintáticas associados aos seus atributos, segundo o seguinte formato::

identificador_regra (lista_opcional de parâmetros) =

{

lista de identificadores → lista de regras ;

}

Exemplo:

dp(&gen, &num) = { det &gen &num → { np(&gen, &num)}}

Trata-se da representação do grupo funcional determinante, constituído de um determinante, gênero e número e do grupo nominal

Observe-se a figura 10.

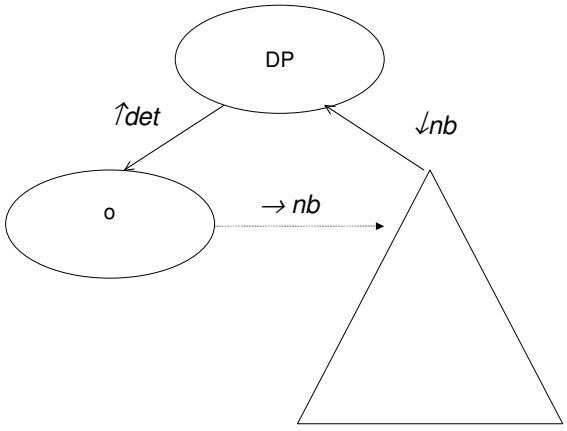


Figura 10 – Especificação do grupo determinante DP, fazendo uso do mecanismo de expansão

Obviamente a gramática da linguagem natural deve conter as diversas estruturas sintáticas, associadas aos seus atributos. Uma vez que cada uma das estruturas sintáticas sejam representadas, os mecanismos apresentados em (NETO, 1993), permitem que ***automaticamente*** a árvore seja expandida até às suas folhas, ou seja, que a estrutura seja internamente representada apenas por uma seqüência de símbolos (identificadores) em vez de estruturas, segundo as regras fornecidas pelo especialista.

Configura-se assim um modo de operação denominado em (NETO, 1993), como modo de expansão. A detecção da necessidade desse modo de operação é efetuado pela utilização do meta-símbolo "→".

Uma vez que internamete a representação da regra é constituída de apenas símbolos , a utilização dessas regras apresenta desempenho compatível com aqueles apresentados no capítulo 2.

Há ainda que se considerar a representação dos atributos. Considerados como parâmetros no modo de aprendizagem, no modo uso, a ocorrência dos argumentos segundo as regras apresentadas, configuram a concordância entre esses atributos.

Em outras palavras, o que se propõe nesta tese é que as regras de especificação de linguagem natural sejam interpretadas pelo transdutor adaptativo tal qual as macros em (NETO, 1993), empregando os mesmos mecanismos adaptativos. Justifica-se a adoção desta estratégia pelo desempenho de cada uma destas técnicas*: proporcional ao número de símbolos presentes na cadeia de entrada e ao número de símbolos presentes no alfabeto de entrada*, as quais incluem também o *tratamento dinâmico e incremental de não-determinismos e ambigüidades.*

. No caso de exánsão de macros em tratamento de linguagens de programação, as ações adaptativas associadas às declarações de macros encarregam-se de coletar o nome e os parâmetros formais da macro, efetuando alterações no autômato para que seja realizada uma

verificação de consistência entre os parâmetros e os argumentos das chamadas da macro. Além destas verificações, as ampliações do autômato promovem a associação entre os parâmetros formais e os argumentos, de forma que as ocorrências dos parâmetros formais sejam substituídos pelos correspondentes argumentos quando da expansão da macro.

Quando uma declaração de uma macro é identificada no código fonte, verifica-se sua sintaxe e o texto que lhe corresponde é simultaneamente memorizado na forma de um conjunto de transições de inserção do mesmo, associado ao identificador da macro. Em seguida, um novo símbolo na cadeia de entrada é inserido, indicando que a declaração da macro foi devidamente processada.

A seguir, relata-se o estudo detalhado do transdutor sintático adaptativo do exemplo "Expansão de Macros", de (NETO, 1993), aqui interpretado como uma máquina para o reconhecimento da sintaxe das regras de especificação de Linguagem Natural.

Inicialmente apresentam-se os mecanismos adaptativos para o modo de operação "treinamento" e em seguida, os mecanismos para o modo de operação "uso".

As funções adaptativas serão descritas através do formalismo gramatical. Também são apresentados os transdutores adaptativos em sua representação gráfica equivalente ilustrando a modificação do transdutor adaptativo quando as funções adaptativas são ativadas.

3.1 A Configuração Inicial do Transdutor Adaptativo e o Modo de Treinamento.

A figura 11. ilustra um autômato semelhante àquele apresentado em (NETO, 1993). Trata-se de um trecho do transdutor adaptativo que reconhece a estrutura sintática das regras a serem fornecidas pelo Lingüista. As transições aí indicadas consomem tokens, fornecidos pelo transdutor léxico.

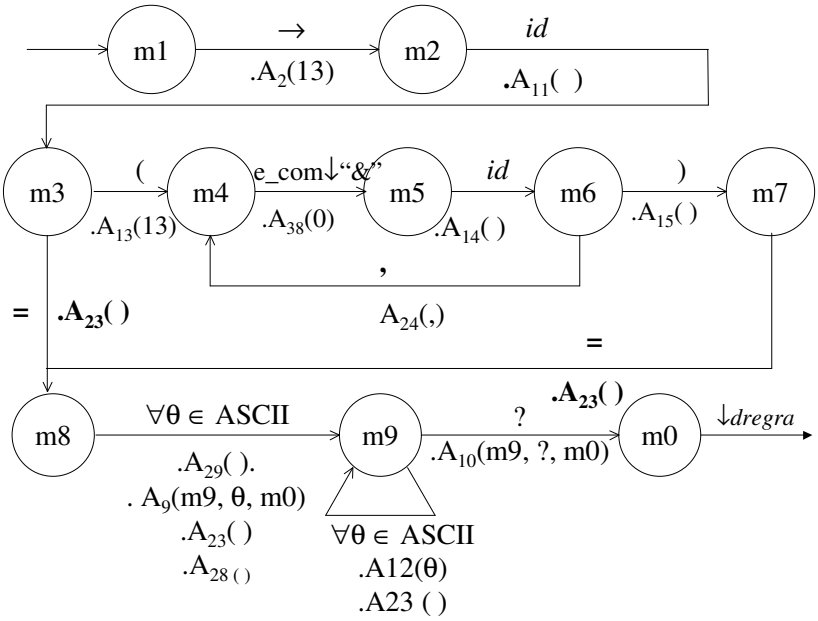


Figura 11 – Trecho de um Transdutor Sintático para aprendizagem das regras a serem fornecidas pelo Lingüista.

O transdutor léxico se encarrega de converter adequadamente os caracteres ASCII que constituem o texto original formado pelas regras fornecidas pelo Lingüista a serem memorizadas pelo sistema, e os substitui na cadeia de entrada pelos seus tokens.

Assim, por exemplo, seja um texto de entrada em que se especifica a inserção da regra:

dp(&gen, &num) = { det &gen &num → { np(gen, num) }}

o analisador léxico insere na cadeia de entrada a seqüência de tokens:

id (e_com id, e_com id) = { id e_com id e_com id → { id (e_com id, e_com id)}}

É necessário que o transdutor adaptativo responsável pela análise léxica, opere ora em modo isolado, consumindo apenas um símbolo como, &, , (e), ora opere em modo átomo consumindo uma seqüência de símbolos na extração dos identificadores de regras e dos parâmetros.

O transdutor sintático além de verificar a sintaxe das regras, configura o modo de operação do analisador léxico, mediante a execução de ações adaptativas presentes em suas

transições. A interação entre o transdutor sintático e o transdutor léxico é efetuada dinamicamente, empregando-se uma pilha explícita, que insere o estado onde o analisador sintático interrompeu sua operação e retornando ao mesmo, após a inserção do token na cadeia de entrada.

Outra tarefa realizada pelo transdutor adaptativo, através da ativação de funções nele presente no modo treinamento, consiste em promover a criação de máquinas iniciais reconhecedores dos argumentos, estabelecendo uma correspondência posicional entre os argumentos e os parâmetros, efetuando dessa forma a ampliação do transdutor adaptativo inicial.

No modo "treinamento", o corpo da regra é associado ao seu identificador e às máquinas reconhecedoras dos argumentos, na forma de produções de inserção, de forma que no modo "uso" uma vez que seu identificador seja referenciado, o texto que lhe corresponde possa ser inserido na cadeia de entrada.

Os identificadores das regras, bem como dos parâmetros, são extraídos pelo transdutor léxico conforme a estratégia para memorização de uma seqüência de símbolos, apresentada no capítulo 2 .

Assim para cada regra presente no modo treinamento é criado um transdutor conforme ilustra a figura 12. Cumpre observar que nesta figura são representados os blocos Cont_1, Cont_2 e Cont_3 que se destinam à contagem de regras em expansão, mecanismo que será detalhado posteriormente.

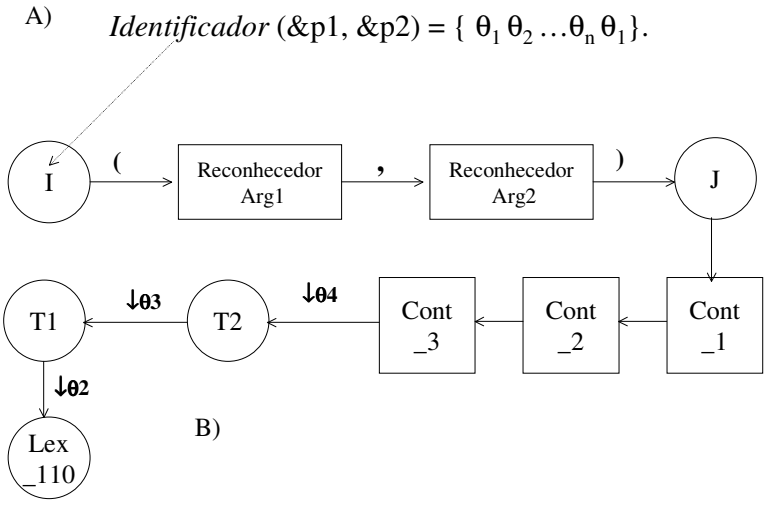
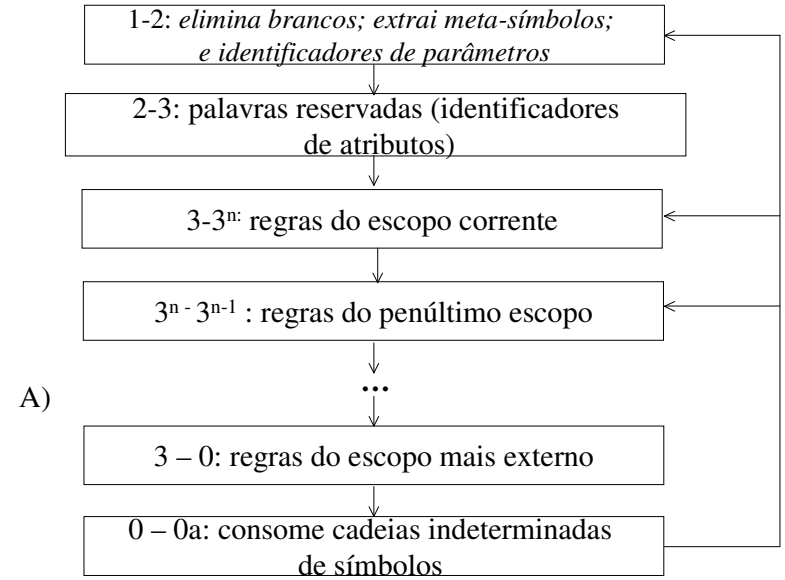


Fig.. 12- (A) Regra processada no modo treinamento
(B) Trecho de transdutor resultante da aprendizagem de uma regra

Na Fig. 13 a) transpõe-se o esquema macroscópico da estrutura do transdutor léxico apresentado em (NETO, 1993), em uma situação em que há o processamento de regras relativas a escopos subordinados presentes em uma construção frasal.

Em 13 B) apresenta-se o trecho do transdutor léxico, onde figuram transições adaptativas que consomem símbolos isolados tais como •, ⊥, ■ , &, ou ainda, transições que consomem um único símbolo ASCII com inserção do respectivo token. Também é representado o trecho com origem no estado 70, a partir do qual se dá a extração e memorização de identificadores de parâmetros. Entre os estados 2 e 3, devem figurar os



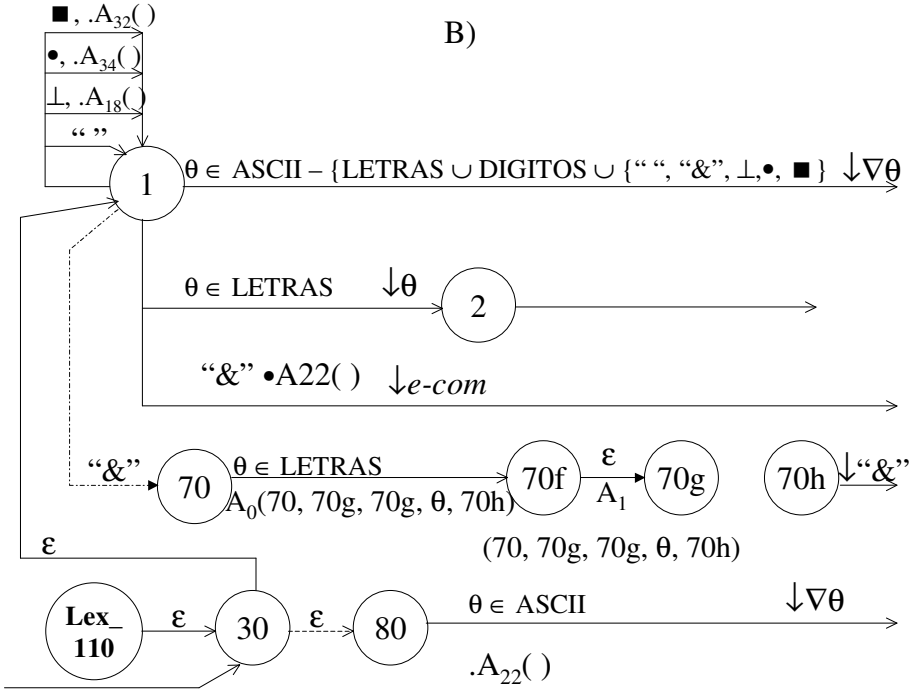


Figura 13 – Transdutor Léxico (Extraído de (NETO, 1993))
A) Esquema macroscópico representando o processamento de regras relativas a n escopos subordinados presentes em uma construção frasal.
B) Módulo 1 –2

transdutores responsáveis pelo reconhecimento de palavras reservadas. Nesta tese se propõe que tais palavras reservadas sejam os atributos gramaticais tais como gênero (masculino, feminino, neutro), pessoa (1, 2, 3), papel temático, etc.

Note-se que ao estado Lex_110 devem convergir todos os transdutores criados no modo "treinamento".

Uma vez que se descreveu a configuração inicial do transdutor léxico-sintático, prossegue-se neste item com a análise dos mecanismos adaptativos propostos por (NETO, 1993), responsáveis pela auto-expansão do transdutor ao longo da operação em modo "treinamento".

Configuração do modo de operação do Analisador Léxico: as funções adaptativas A_{22} e A_{23}

A função adaptativa A_{23} configura o próximo estado do analisador léxico para o estado 80, a partir do qual o analisador léxico consome um símbolo isolado ASCII da cadeia de entrada e o substitui pelo seu respectivo token. A inserção do token na cadeia de entrada é sucedida pela ativação da função adaptativa A_{22}, que configura o analisador léxico de forma que a extração dos símbolos se faça a partir do estado 1, origem de transições responsáveis pela extração de uma seqüência de símbolos, bem como de alguns símbolos isolados isolados.

A_{23} ( ) = {

-[N30 $\rightarrow$ N1]

+[N30 $\rightarrow$ N80] }

A_{22} ( ) = {

-[N30 $\rightarrow$ N80]

+[N30 $\rightarrow$ N1] }

Avaliação de complexidade:

A execução das ações adaptativas A_{23} e A_{22} não altera o número de regras presentes na gramática. O tempo consumido para efetuar cada uma delas é o de duas transições elementares.

Considere-se novamente o autômato representado na figura 11. À medida que se sucede a verificação da sintaxe do cabeçalho da regra, um outro trecho de transdutor adaptativo é criado. Tal transdutor é constituído de:

a) um conjunto de transdutores iniciais capazes de reconhecerem os argumentos associados aos parâmetros das regras;
b) um conjunto de transições responsáveis pela substituição do identificador de uma regra pelo texto que lhe corresponde.

O novo trecho do transdutor adaptativo é criado e incorporado ao analisador léxico de forma que esteja disponível no modo "uso" da regra.

Atribuição de Tipo: a função adaptativa $A_2(t)$

$A_2(t)$ atualiza o ponteiro N50_TipoAtual, o qual indica o tipo "default" da informação em reconhecimento. No início do processo de aceitação de uma regra, lhe é atribuído o tipo void _13

$$A_2(t) = \{ j:$$
$$-[N50\_TipoAtual \rightarrow j]$$
$$+ [N50\_TipoAtual \rightarrow t] \}$$

Avaliação de complexidade:

Esta função é ativada uma vez para cada nova regra a ser inserida no sistema. Por outro lado, não altera o número de regras da gramática adaptativa. Segue-se o autômato correspondente.

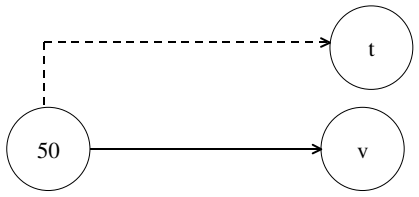


Figura 14 - Atualização do Ponteiro para o tipo de cadeia em extração

A criação de uma gramática auxiliar : função adaptativa A_{11}

Tão logo se memorize o identificador da regra, uma sub-gramática auxiliar é inicializada, pela ativação da função adaptativa A_{11}. Esta sub-gramática, que apresenta como raiz o não-terminal N15 é incrementalmente ampliada. Entre os não-terminais N18 e N15 serão criadas produções associadas à contagem dos parâmetros de uma regra.

Segue-se a descrição da função adaptativa A_{11}, bem como o mapeamento da sub-gramática inicializada, para o autômato correspondente.

A_{11} (t) = { w, x, y, z:
- [N15 $\rightarrow$ x]
- [N18 $\rightarrow$ z]
+ [N15 $\rightarrow$ N14]
+ [N18 $\rightarrow$ N15] }

Avaliação de complexidade:

Esta ação é executada uma vez para cada regra a ser inserida no modo "treinamento".

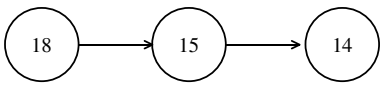


Figura 15 - Inicialização de uma Lista Auxiliar

A Função Adaptativa A_{13}

A função adaptativa A_{13} é ativada pelo consumo do token (. Tem por função disponibilizar uma regra que consome o mesmo símbolo no modo uso, porém associado às funções adaptativas A_{23} e A_SbAt.

A_{13} também atualiza o ponteiro N50_TipoAtual para N16 (corresponde ao tipo parâmetro), visto que ao consumo do símbolo (, segue-se a extração dos parâmetros da regra.

A_{13} =

{ I, X, Y, Z. J*:

/* remoção de produções para a ativação de tipo parâmetro*/

- [N50_TipoAtual $\rightarrow$ X]

- [Y $\rightarrow$ {A_SbAt(I)} N13_AUX]

- [N13_AUX $\rightarrow$ VOID_13]

/*remoção de regras que permitiriam a extração e devolução de símbolos na cadeia de entrada No transdutor em construção só se permitirá a ocorrência do símbolo)*/

- [θ Z $\leftarrow$ θ Y] $\forall$ $\theta \in$ ASCII – (LETRAS $\cup$ DIGITOS) /* 66 regras */

/*inserção de regra que consome o símbolo (, associado a ações adaptativas para o modo uso */

+ [I $\rightarrow$ {A_{23}, A_SbAt(I)} (J]

/*transdutor em construção ainda é do tipo void_13 */

+ [J $\rightarrow$ VOID_13]

/*próxima cadeia a ser extraída deverá ser do tipo parâmetro (N16) */

+ [N50_TipoAtual $\rightarrow$ N16]

/*configura léxico para operar em modo isolado para extração de "&" */

A_{23} }

Avaliação de Complexidade:

A execução da ação adaptativa é efetuada uma vez para cada regra no modo treinamento. Sua execução diminui em 66, o número de regras presentes até o instante do reconhecimento do símbolo (. Consome-se um tempo equivalente ao de 74 regras elementares. (2 regras elementares referem-se à execução de A_{23}).

A Função adaptativa A_{38}

As duas seguintes regras correspondem à transição que consome o símbolo *e_com*, presente no autômato da figura 11

$$M4 \rightarrow e\_com\ M5\_Aux$$
$$M5\_Aux \leftarrow \{A_{38}(0)\}\ \&\ M5$$

Observe-se que o símbolo *e*_com, que precede o identificador de um parâmetro, é substituído na cadeia de entrada pelo símbolo "&". Trata-se de uma estratégia para manter disponível na cadeia de entrada o símbolo "&", de forma em seqüência se extraia o parâmetro. Isto se faz necessário porque o trecho do transdutor que reconhece identificadores de parâmetros tem origem no estado 70, enquanto que o transdutor que reconhece identificadores de regras tem origem no estado 3. (Confira-se na Fig. 13 b))

Uma vez consumido o identificador de parâmetro, o token *id* é inserido na cadeia de entrada. Segue-se a descrição da função adaptativa A_{38}.

A_{38} = {

/ *"& é substituído por *e_com* */

-[N1 → {A_{22}} "&" N17]

/ *"&" é sucedido pelo reconhecimento de identificadores de parâmetros*/

+ [N1 → "&" N70] }

Avaliação de complexidade:

O número de regras não é alterado pela execução da ação adaptativa A_{38}. Sua execução é o de duas ações elementares. É efetuada para cada parâmetro de cada regra no modo Treinamento.

As Funções adaptativas A_{24} e A_{15}

Estas funções adaptativas são ativadas pelo consumo dos tokens associados aos delimitadores da lista de parâmetros "," e ")", respectivamente e promovem a criação de regras que consomem os mesmos símbolos, na lista de argumentos. Segue-se a descrição das mesmas.

$A_{15}(x) = \{I , J^*:$
$-[I \rightarrow VOID\_13]$
$+[I \rightarrow x\ J]$
$+[\ I \rightarrow VOID\_13] \mid$

Avaliação de Complexidade:

Esta função adaptativa é executada na ocorrência do símbolo) na cadeia de entrada no modo Treinamento. Acrescenta em 1 o número de regras na gramática existente. O custo no tempo é aquele de execução de 3 regras elementares.

$A_{24}(x) = \{I , J^*:$
$-[I \rightarrow VOID\_13]$
$+[I \rightarrow \{A_{23}(\ )\}\ x\ J]$
$+[\ I \rightarrow VOID\_13]\ \ \}$

Avaliação de Complexidade

Esta função adaptativa é executada sempre que o símbolo **,** é encontrado na cadeia de entrada no modo Treinamento. Acrescenta em 1, o número de regras na gramática existente. O custo no tempo de sua execução coincide com o de 3 regras elementares.

As Funções adaptativas A_{14} e A_{36}

O identificador do parâmetro, que se segue ao símbolo "&", é memorizado em um autômato e é substituído na cadeia de entrada pelo token *id.*

Uma vez que o símbolo *id* seja consumido pelo trecho sintático do transdutor adaptativo, a função adaptativa A14 é ativada e realiza as seguintes tarefas:

- Inicializa uma sub-gramática destinada à extração argumento que lhe corresponderá no tempo de uso da gramática.
- Cria e efetua a manutenção de uma lista dos não-terminais associados aos identificadores aos identificadores dos parâmetros.
- Promove a execução da ação adaptativa A_{36} que acrescenta à sub-gramática recém-inicializada, *mecanismos de tratamento de não-determinismos e ambigüidades,* no que diz respeito à extração de identificadores de argumentos.

A figura 16 ilustra o mapeamento da sub-gramática inicial referente aos argumentos, para autômato adaptativo. Observe-se o argumento w da ação adaptativa A_{27}. Este argumento refere-se ao não-terminal associado ao identificador do parâmetro recém-reconhecido, apontado portanto pelo não-terminal N90.

Segue-se o conjunto de ações que constituem a ação adaptativa A_{14}.

A14 ( ) = { W, X, N, S. M*, J*, K*, R*, T*:

/* ao consumo do próximo símbolo "&" deverá ser inserido o token e_com */

-[N1 → "&" N70]

+[N1 → {A_{22}( )} N17]

/* verifica qual é o último identificador extraído */

?[N90 → W]

/* insere identificador recém-criado na lista de parâmetros */

-[N → {A_{19}(S)} N60]

+[M → {A_{19}(W)} N60]

+[N → {A_{19}(S)} M]

/* cria autômato para reconhecimento de argumento K, associado ao parâmetro W: associação memorizada através dos argumentos da ação adaptativa A_{27} */

+[I → {A_{27} (J , θ, K, W), A_{23}( ), A_{25}( )} θ J} ∀ θ ∈ ASC II] /* 128 transições */

+[J → {A_{12} (θ), A_{23}( )} θ J] ∀ θ ∈ ASC II – {"&"} /* 127 transições */

+[J → { A_{12} ("&"), A_{23}( )}]

+[T → θ J] ∀ θ ∈ (LETRAS ∪ DIGITOS) } /* ASC II: 128 transições*/

+ [θ T ← θ J] ∀ θ ∈ ASC II - (LETRAS ∪ DIGITOS) }

/* complementa ação adaptativa A_{14} */

A_{36} (W, T, J, K, R) }

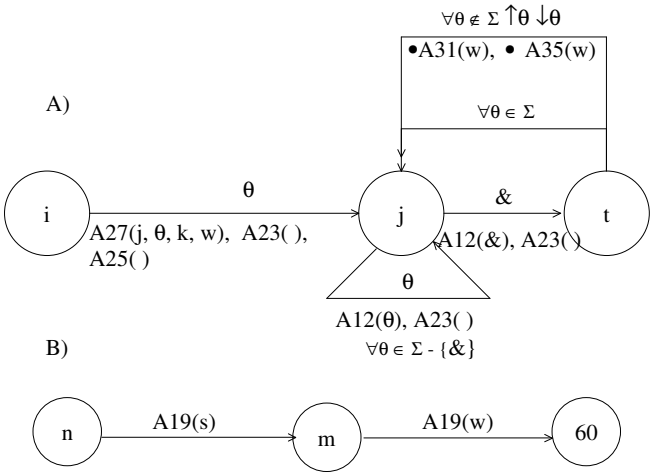


Fig. 16 - A) Configuração da Máquina M_i, destinada ao reconhecimento de um argumento A_i, criada pela execução da ação adaptativa A_{14}, obtida no modo Treinamento. B) Lista de Não - Terminais associados aos Parâmetros.

Avaliação de Complexidade:

A ação adaptativa A_{14} aumenta em 385, o número de regras para cada identificador de parâmetro presente em cada regra a ser incluída (aprendida) no sistema . O seu desempenho

coincide com o de 390 ações elementares. Esta avaliação não levou em conta a execução da ação adaptativa A_{36}, comentada no entanto, a seguir.

A função adaptativa A_{36}

Esta função adaptativa, quando ativada, realiza as seguintes tarefas:

- Cria produções que permitam que em tempo de uso, o transdutor correspondente em construção possa contabilizar o número de parâmetros presentes na lista de parâmetros;
- Acrescenta à sub-gramática recém-criada por A_{14} regras concernentes à extração de argumentos, a saber:
  - regras que finalizam a extração de argumentos;
  - a detecção de um argumento homônimo ao parâmetro formal. Esta tarefa, em (NETO, 1993) é realizada pelas ações adaptativas A_3 e A_4 , comentadas a seguir.

Detecção de um argmento homônimo ao parâmetro formal: as funções adaptativas A3 e A4.

Estas funções adaptativas são empregadas na criação de aceitador de uma cadeia de símbolos quaisquer, com tratamento de cadeias anteriormente reconhecidas e passíveis de apresentarem prefixo comum, ou até mesmo serem iguais às cadeias a serem aceitas.

Considere-se uma situação em que se deseja reconhecer uma seqüência de símbolos quaisquer entre dois estados j e t. Inicialmente a transição entre os dois estados j e t, se apresenta configurada apenas para o consumo do meta-símbolo *e-com*, que marca o início de um identificador de parâmetro ou argumento. (Observe-se que no caso geral, o símbolo inicialmente presente é irrrelevante). Considere-se no entanto que já foi detectada em algum

instante do reconhecimento uma seqüência de símbolos, armazenada em uma lista, cujo estado final de aceitação é w.

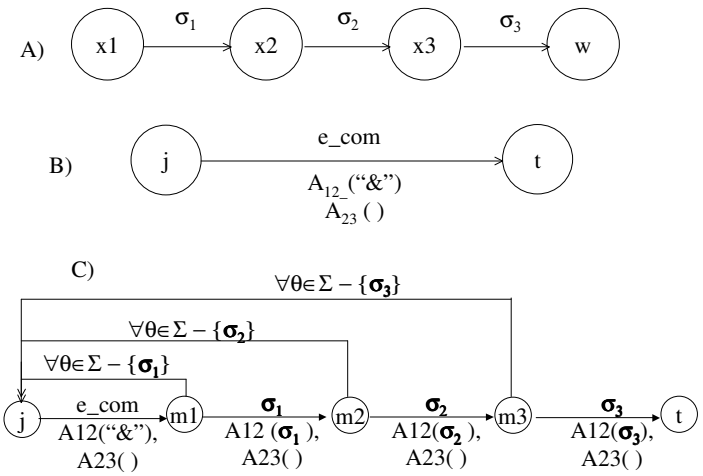


Fig. 17: Resultado da execução das ações adaptativas A3 e A4: A) W é o estado associado a um identificador de parâmetro. Antecedem-lhe três transições que consomem a cadeia de símbolos $\sigma_1\sigma_2\sigma_3$
B) Entre j e t, existe inicialmente apenas uma transição que consome o símbolo isolado e_com. C) Autômato resultante da aplicação das ações adaptativas A_3 e A_4

A técnica empregada nas funções adaptativas A_3 e A_4 permite que o trecho de autômato correspondente ao identificador de parâmetros seja copiado para o autômato para extração de argumentos. Dessa forma, ainda que o argumento seja homônimo ao parâmetro formal, em tempo de *uso*, estará disponível uma cópia deste identificador no escopo demarcado para o tratamento de argumentos.

Graças ao mecanismo de cópia, que associa cadeias de símbolos iguais de tipos distintos (no caso tipo parâmetro e tipo argumento), o autômato é construído de forma que os identificadores dos argumentos sejam extraídos ***deterministicamente*** ainda que apresentem uma sub-cadeia coincidente com a do parâmetro formal.

Tem-se que:

$A_3(W, T, J) = \{\sigma, X:$

/*verifica, incrementalmente os símbolos que constituem o identificador do parâmetro */

?[X → σ W]

/*executa ação adaptativa A_4 */

A_4 (σ, X, W, T, J) }

A_4 (σ, X, W, T, J) = { M*:

-[J → {A_{12}("&"), A_{23}( )} e_com T]

+[M → {A_{12}(σ), A_{23}( )} σ T]

+[M → θ J] ∀θ ∈ Σ - { σ }

+[J → {A_{12}("&"), A_{23}( )} e_com M]

A_3 (X, M, J) }

Avaliação de complexidade:

Se houver n transições com n símbolos, tem-se que o autômato se expande em 128*n transições elementares . e n estados.

O consumo no tempo é de (131+n) unidades de execução de transições elementares. Acrescente-se o custo no tempo de n chamadas da ação adaptativa A_4 e n chmadas da ação A_3

Além da execução das ações adaptativas A_3 e A_4, para criação de autômato que detecte argumento homônimo ao parâmetro, a ação adaptativa A_{36} acrescenta transições ao autômato para finalização da extração do argumento, bem como para contabilizar o número de parâmetros. Para a contagem de parâmetros, emprega-se o símbolo especial "⊥".

Segue-se a descrição da ação adaptativa A_{36}.

A_{36} (W, T, J, K, X, R) = {
A_3 (W, T, J)
+ [J → { A_{26}(J,•, K, W) } K]
+[K → Void_13]
-[X → N15]

+[X ← ⊥ R]

+[R → N15] }

Avaliação de Complexidade:

A execução da ação adaptativa A_{36} apresenta desempenho dependente da execução das ações adaptativas A_3 e A_4, e portanto dependente do número de símbolos presentes que constituem o identificador do parâmetro formal. Acrescido ao desempenho destas, A_{36} cria para cada parâmetro mais 3 transições: duas destinadas à finalização de extração de argumento e uma destinada à contagem de parâmetro. O desempenho no tempo é dependente da execução das ações adaptativas A_3 e A_4, acrescido do custo de execução de 5 ações elementares para cada parâmetro presente na lista de parâmetros de uma regra.

A inicialização da máquina M_i, para reconhecimento de argumento associado ao parâmetro P_i é apresentada na figura 18.

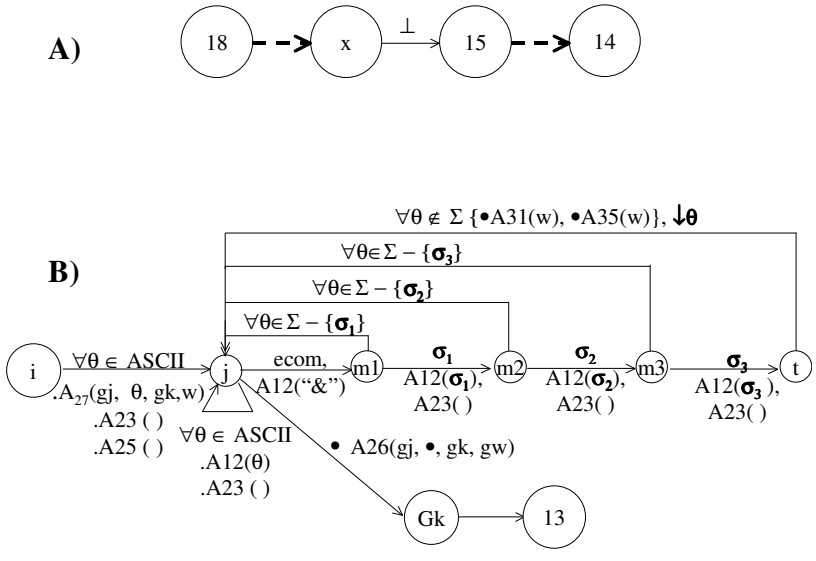


Fig. 18- A) Contagem de Parâmetros
B)Autômato Incial para Reconhecimento de um Argumento, obtido pela execução das ações adaptativas A_{14}

Como exemplo, a figura 19 ilustra a configuração do trecho do autômato criado, após a execução das ações adaptativas, A_{11}, A_{13}, A_{38}, A_{14}, A_{36}, A_{15} e A_{24}, na aprendizagem do cabeçalho de uma regra com três parâmetros.

As máquinas M_1, M_2, M_3 na figura 19 representam autômatos cuja configuração é a mesma que aquela ilustrada em 4.8 b)

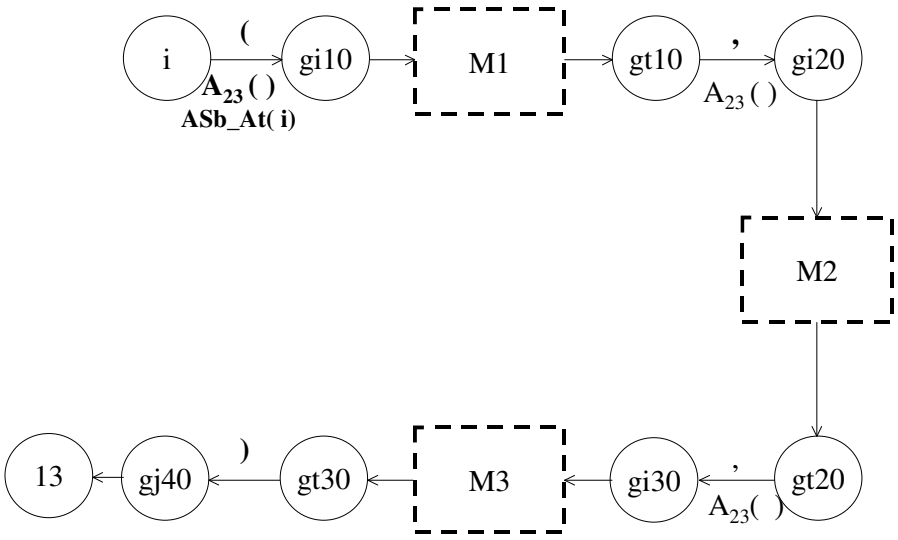


Figura 19 – Autômato para Reconhecimento dos Argumentos Associados aos Parâmetros

Os procedimentos até agora efetuados permitiram inicializar o trecho do transdutor responsável pelo reconhecimento dos argumentosss, bem como estabelecer a correspondência entre o identificador dos parâmetros recém-extraídos ao argumento a ser encontrado na cadeia de entrada no tempo de uso.

O processo de aprendizagem da regra continua, a fim de memorizar o corpo da mesma. Para tanto, o analisador léxico é preparado para extrair símbolos isolados através da execução da ação adaptativa A_{23}. Isto é necessário dado que o primeiro símbolo disponível na cadeia de entrada será interpretado como o delimitador do corpo da regra. Alterado o modo de

operação para o modo símbolo isolado, é executada função adaptativa A_{29}, que insere antes do não-terminal void_13, um mecanismo de contagem de regras a ser executado no modo uso, bem como um marcador de fim de regra.

Contagem de regras em processamento: a função adaptativa A_{29}

No corpo de uma regra, apresentam-se geralmente outras regras, as quais podem ser executadas seqüencialmente ou ainda é possível que uma regra referencie outras.

Para que, no modo "uso", haja o controle de quantas regras estão sendo aplicadas dentro do escopo de uma outra regra, a ação adaptativa A_{29} insere um mecanismo de contagem, o qual por sua vez é efetuado pelas ações adaptativas A_{16} e A_{33}. No modo uso, a informação de que o processamento de uma regra foi finalizado é obtida quando se encontrar na cadeia de entrada o meta-símbolo "•"; sendo assim a ação adaptativa A_{29} também acrescenta ao autômato em construção uma transição de inserção deste meta-símbolo.

Segue-se a descrição algébrica da função adaptativa A_{29}:

A_{29} ( ) =
{ I, K, J*, M*, N*:
-[N50_TipoAtual → K]
-[I → Void_13]
/* A_{16} e A_{33}efetuam mecanismo de contagem */
+ [I → {A_{16} (J, M) } J]
+ [J → {A_{33} (J, M)} M]
/* transição de inserção de símbolo de fim de regra */
+[M ← • N]
+[N → Void_13]
+[N50_TipoAtual → K] }

Avaliação de Complexidade:

Para cada regra são acrescentadas 3 produções e o tempo de execução é o de 7 ações elementares.

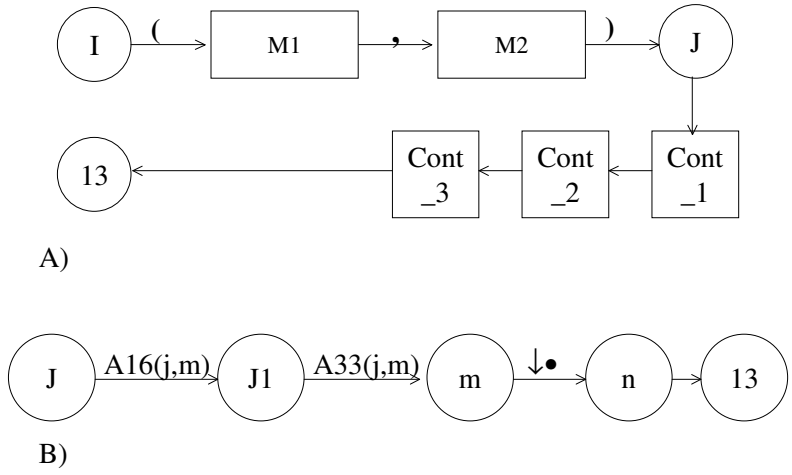


Figura 20 – Mecanismo de contagem de número de regras a serem memorizadas pelo sistema.
A) mecanismos de contagem representado em blocos;
B) Trecho de Autômato Adaptativo para contagem de regras.

Após a execução de A_{29}, o primeiro símbolo da cadeia de entrada é extraído e é memorizado como argumento de A_9.

Detecção do símbolo de delimitação de uma regra: a função adaptativa A9

A extração do primeiro símbolo θ_1 do corpo da regra ativa esta função adaptativa. Este símbolo é memorizado em seu argumento e a transição adaptativa entre M_9 e M_0, bem como o laço em M9, são alterados de forma que na próxima ocorrência do símbolo θ_1, o mesmo seja consumido nas transições entre M_9 e M_0. Implementa-se assim, a interpretação do primeiro símbolo do corpo da regra como seu delimitador.

A função adaptativa A_9 é descrita como:

$A_9(I, J, \sigma) =$

{ X:

$-[I \rightarrow \{A_{10} (I, X, J)\}\ X\ J]$

$+[I \rightarrow \{A_{10} (I, \sigma, J)\}\ \sigma\ J]$

-[I → {A_{12} (σ), A_{23}( )}σ J] }

Avaliação de Complexidade:

A ação adaptativa é efetuada apenas uma vez para cada nova regra presente no modo treinamento. Decrementa de 1 o número regras presentes na gramática. O seu desempenho no tempo é de 3 regras elementares.

A figura 21 ilustra a alteração do transdutor pela execução da da ação adaptativa A_9.

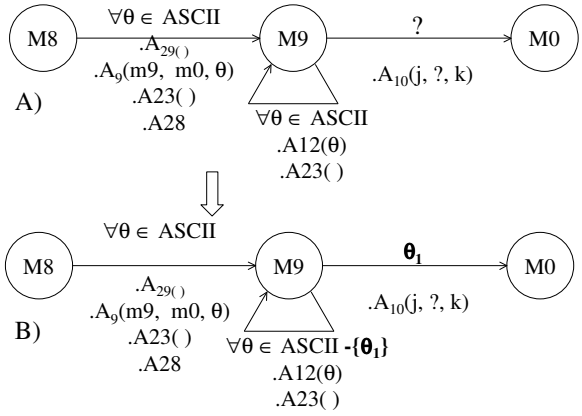


Figura 21 – O primeiro símbolo θ_1 na cadeia de entrada é interpretado como delimitador do corpo da regra

O analisador léxico é programado novamente para operar no modo símbolo isolado, através da execução da ação adaptativa A_{23} . Em seguida, a ação A_{28} é executada.

Função Adaptativa A_{28}

A função adaptativa A_{28}, simplesmente desconecta a seqüência de produções entre os não-terminais N18 e N15, indicadoras do número de parâmetros (observe-se a figura 18 -A)) e conecta tais produções ao final das produções concernentes ao reconhecimento dos identificadores dos argumentos da regra. A ação adaptativa A_{28} invoca a ação adaptativa A_{25}.

Ação Adaptativa A_{25}

Esta ação adaptativa inicializa novamente a lista auxiliar, que agora se destinará a armazenar temporariamente o texto referente à regra. A ação adaptativa A_{25} também configura o analisador léxico de forma que uma vez que seja encontrado o símbolo que inicializa um identificador de parâmetro, este seja marcado como *e_com.*

Todos os demais símbolos são lidos e armazenados entre os não-terminais 15 e 14, através da execução da ação adaptativa A_{12}, até que seja encontrado o símbolo θ_1.

Seguem-se as descrições algébricas das ações adaptativas A_{28}, A_{25} e A_{12}.

A_{28} =

{ X, σ, J:

-[I → VOID_13]

/* contagem de parâmetros é desconectado da área de rascunho e inserido no transdutor em construção para contagem de argumentos */

-[N18 ← σ X]

+[I ← σ X]

-[J → N15]

+[J → VOID_13]

/* reinicia lista auxiliar para armazenar temporariamente corpo da regra */

A_{25} }

A_{25}( ) = { X, Y:

/* reinicia lista auxiliar para armazenar temporariamente corpo da regra */

A_{11}

/* "&" é substituído por e_com */

-[N1 → "&" X]

+[X → "&" N1]

-[N1 → {A_{22}( )}"&" Y]

+[Y → {A_{22}( )} "&" N1]

+ ["&" N80 →{A_{22}( )} e_com Z]

+[Z → ε] }

Avaliação de Complexidade:

As funções adaptativas A_{28}e A_{25} são ativadas na ocorrência de um símbolo delimitador do início do corpo de uma regra na cadeia de entrada. A execução de ambas normalmente *não* acrescenta regras à gramática e apresenta o desempenho no tempo equivalente ao de 14 ações elementares, sendo 4 ações elementares associadas à execução da ação adaptativa A_{11}.

Função adaptativa A_{12}

Esta função adaptativa é ativada para cada símbolo ASCII presente no corpo da regra. e cria para cada símbolo uma produção de inserção do mesmo, na sub-gramática auxiliar delimitada pelos não-terminais N14 e N15. Segue-se sua descrição algébrica:

$A_{12}(\sigma) =$

{ X, GJ*:
-[N15 $\rightarrow$ X]
+[N15 $\rightarrow$ GJ]
+[GJ $\leftarrow$ σ X]
}

Avaliação de Complexidade:

Para cada símbolo, presente na regra, exceto os símbolos qque a delimitam a gramática é acrescentada de uma produção. O tempo de execução é constante e igual ao de 3 produções elementares

Função adaptativa A_{10}

Esta ação adaptativa é executada quando for novamente encontrada a segunda ocorrência do símbolo θ_1, que é dessa forma interpretado como delimitador do corpo da regra.

Através da execução de A_{10}, a seqüência de produções entre os não-terminais N15 e N14, é desconectada e concectada ao Analisador Léxico, resultando em um autômato semelhante ao da figura 22.

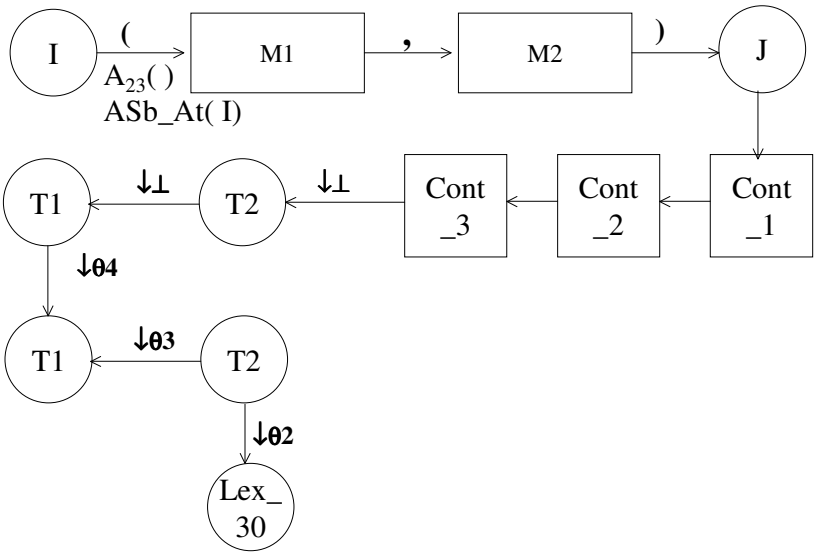


Figura 22 – Autômato conectado ao Léxico após a execução da ação adaptativa A_{10}.

Observe-se na figura que, no modo uso, na ocorrência do identificador de uma regra na cadeia de entrada resultará na substituição do texto que lhe corresponde e subseqüente ativação do analisador léxico.

Segue-se a descrição algébrica da ação adaptativa A_{10}.

$A_{10}(I, \sigma, J) =$

{ K, X, Y, Z, R, S, T, U, v, GM*:

-[K → VOID_13]

-[X → “&” N1]

+[N1 → “&” X]

-[R →{A_{22}( ) } “&” N1]

+[N1 → {A_{22}( ) }“&” R]

-[“&” N80 → {A_{22}( )} e_com Z]

+[I → {A_{12} (σ), A_{23}( )} σ J]

/* texto do corpo da regra é associado ao fim ao identificador da regra e desconectado da lista auxiliar para armazenamento temporário (entre os não terminais N15 e N14) */

-[N15 → T]

+[K → T]

-[U ← v N14]

/*final do texto realimenta léxico */

+[U ← v N30]

/*restauração da última produção referente à sintaxe de uma regra */

-[I → {A_{10} (I, s, J)} s J]

+[I → {A_{10} (I, ?, J)} ? J]

}

Avaliação de Complexidade:

Esta função é ativada apenas uma vez a cada duas ocorrências do símbolo delimitador de uma regra, visto que após sua execução, restaura o autômato de forma que a seguinte ocorrência do mesmo símbolo possa ser interpretado como se fosse a primeira. Uma vez que, apresenta 7 regras elementares de exclusão e 6 regras elementares de inserção, ela não altera o número total de regras da gramática e seu desempenho no tempo corresponde ao de 13 ações elementares.

Ao final da execução da ação adaptativa A_{10} é inserido um símbolo na cadeia de entrada, denotando que foi finalizada a aprendizagem da regra.(dregra)

A seguir, são apresentados os mecanismos adaptativos empregados no modo Uso

3.2 Execução das Ações Adaptativas do Transdutor no Modo Uso.

No item anterior foi possível constatar como no modo treinamento, um transdutor adaptativo para o processamento de uma regra é criado e incorporado ao analisador léxico para seu uso posterior.

No modo uso, os transdutores criados anteriormente, são empregados de forma que as ações adaptativas neles presentes efetuem o reconhecimento identificadores das regras e de seus argumentos, realizem a associação entre os parâmetros formais e os argumentos das regras e promovam a substituição das regras pelo texto que lhe correspondem.

A seguir, estas ações adaptativas são apresentadas.

O reconhecimento de um argumento de uma regra: as funções adaptativas A_{27}, A_{25} e A_{12}

Considere-se a figura 23. Aí apresenta-se um trecho do autômato adaptativo,

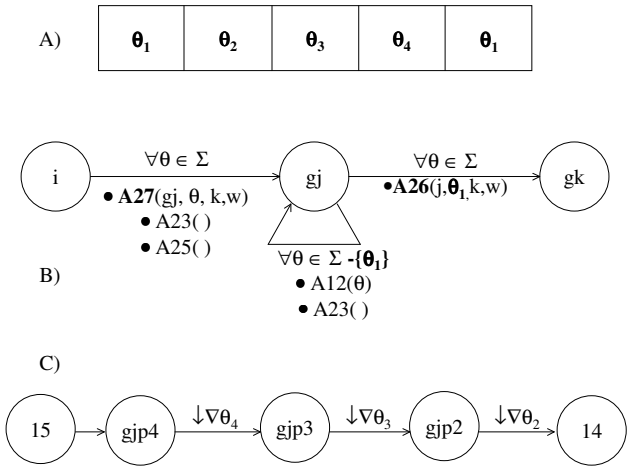


Figura 23 – Processamento de um Argumento de uma Regra
A) Cadeia de entrada original – cadeia de entrada constituída de símbolos associados ao identificador.
B) Resultado da Execução da ação adpatativa A_{27}, após o consumo do primeiro símbolo θ1, presente na cadeia de símbolos que constituem o argumento.
C) Resultado da Execução da ação adaptativa A_{12}: memorização da cadeia de símbolos que constituem o argumento

construído no modo de aprendizagem, e se destina ao reconhecimento de um argumento. Como se pode observar, faz uso das açãoes adaptativas A_{27}, A_{25}, A_{12}, além de A_{23}.

Os identificadores dos argumentos das regras são constituídos, tais como os respectivos parâmetros, de uma cadeia de caracteres ASCII..

A ativação da função adaptativa A_{27}, promove o armazenamento do primeiro símbolo θ_1 constituinte do identificador em seu argumento, elimina a transição com o símbolo especial "• ", substituindo-o pelo símbolo θ_1, constituindo-se em um mecanismo muito análogo àquele efetuado pelas ações adaptativas A_9 e A_{10}.

A ação adaptativa A_{25} inicializa a lista $18 \rightarrow 15 \rightarrow 14$ e configura o analisador léxico de forma que se o símbolo "&" for extraído da cadeia de entrada ele deverá ser entendido como um outro símbolo da cadeia de entrada qualquer. Entre os estados 15 e 14 serão criadas transições de inserção dos símbolos presentes na cadeia de entrada, até que haja uma nova ocorrência do símbolo θ_1. Isto será possível graças à execução da ação adaptativa A_{12}, responsável por este empilhamento. Há que se notar que os símbolos das transições de inserção estão presentes na forma de tokens, visto que serão empregadas na parte sintática do transdutor. Este efeito é obtido, graças às execuções da ação adaptativa A_{23} para cada símbolo do argumento. Em particular, a presença de um símbolo "&" será mapeado para o seu token e_com. Na nova ocorrência do símbolo θ_1, será executada a ação adaptativa A_{26} .

Na figura 23 C), observa-se que o estado 15 aponta para o último símbolo da cadeia de entrada referente ao argumento, enquanto que o estado 14 será apontado pelo primeiro símbolo da cadeia de entrada.

A seguir é apresentada a descrição algébrica da ação adaptativa A_{27}

$$A_{27}(J, \sigma, K, W) = \{ X:$$

$$-[J \rightarrow \{A_{26}(J, X, K, W)\}\ X\ K]$$

$$+[J \rightarrow \{A_{26}(J, \sigma, K, W)\}\ \sigma\ K]$$

$$-[J \rightarrow \{A_{12}(\sigma), A_{23}( )\}\ \sigma\ J]\ \}$$

Avaliação de Complexidade:

A função adaptativa A_{27} é ativada para cada argumento presente na referência a uma determinada regra. O número de produções da gramática é decrementado de 1. O seu desempenho no tempo é aquele de 3 produções elementares.

Uma vez executada a ação adaptativa A_{27}, para a detecção do símbolo delimitador do identificador de um argumento, a extração dos demais símbolos é efetuada. No modo treinamento, o reconhecedor foi construído de tal forma, que ao longo da extração é verificada a ocorrência de um argumento homônimo ao parâmetro formal. No caso de se encontrar um argumento homônimo ao parâmetro formal, são efetuadas as ações adaptativas A_{31} e A_{35}.

Estas funções adaptativas são ativadas na ocorrência de um argumento homônimo ao parâmetro formal. Observe-se na figura 18 que neste caso, o argumento não é memorizado na lista auxiliar. Daí, a necessidade de se efetuar uma cópia do parâmetro formal nesta lista auxiliar.

As funções adaptativas A_{31} e A_8

Inicialmente, a função adaptativa A_{31} desconecta o analisador léxico e executa a ação adaptativa A_8, a qual por sua vez efetivamente retira o último identificador.que foi armazenado entre os não-terminais N15 e N14 Seguem-se as descrições algébricas das ações adaptativas A_{31} e A_8

$A_{31}(W) =$

{ -[N110_Lex $\rightarrow$ N30_Lex]

-[N90 $\rightarrow$ W]

$A_8(1)$

}

Avaliação de Complexidade

Esta função adaptativa é ativada somente quando um argumento é homônimo ao parâmetro formal. O seu custo é igual ao da função adaptativa A_8, exceto que se deve reduzir em 2, o número de regras na gramática, bem como incluir o tempo de execução desta redução.

$A_8(f) = \{ X, Y, Z:$

-[N15 $\rightarrow$X]
/* remove qalquer símbolo ASCII, exceto aquele que inicializa novo argumento */
-[X $\leftarrow \theta$ Y] $\forall \theta \in$ ASCII – {"&"}
/* marca não-terminal associado a novo argumento em Z */
-[X $\leftarrow$ "&" Z]
/*reconecta extremidade */
+[N15 $\rightarrow$Y]
/* idem no caso ter eliminado "&" */
+[N15 $\rightarrow$Z]
/* caso contrário, volta a eliminar outro símbolo */
A_8(Y)

Avaliação de desempenho

Para cada símbolo a eliminar, são executadas ações para remover e reconectar o terminal N15 e um teste para verificar se se trata de um símbolo ASCII, distinto de "&". Ao final de n símbolos constituintes do argumento que não o homônimo mais recente, são removidas n+1 transições (os símbolos e aquela que consome "&") e o não terminal N15, adequadamente reconectado.

Após a ativação das ações adaptativas A_{31} e conseqüentemente de A_8, é ativada a função adaptativa A_{35} que dispara a execução das funções adaptativas A_5, A_6 e A_7 a fim de se obter uma uma reprodução do parâmetro formal. O transdutor em construção que fora

anteriormente desconectado do analisador léxico pela ativação de A_{31} é novamente conectado.

Seguem-se as descrições destas funções adaptativas.

Funções Adaptativas A_5 , A_6 e A_7

Estas funções adaptativas constituem um mecanismo adaptativo de cópias de regras que empilham símbolos na cadeia de entrada. Considere-se um conjunto de regras, como o ilustrado pelo exemplo seguinte:

$S \leftarrow \sigma_1 K_1$
$K_1 \leftarrow \sigma_2 K_2$
$K_2 \leftarrow \sigma_3 K3$
$R0 \rightarrow \{A\_SbAt(X)\}R1$
$R1 \rightarrow S$
$N15 \rightarrow U$
$Pont \rightarrow X$
$Pont \rightarrow R0$

A execução da ação adaptativa A5 e a primeira execução da ação adaptativa A6 permitem que se verifique a existência desse conjunto de regras.

As posteriores execuções da ação adaptativa A6 em conjunto com a ação adaptativa A7 permitem que se efetuem cópias dos n símbolos de empilhamento.

Observe-se que após a execução das ações adaptativas acima, é possível criar o seguinte conjunto de regras:

$N15 \rightarrow GV1$
$GV1 \leftarrow \sigma_1 GM_1$
$GM_1 \leftarrow \sigma_2 GM_2$
$GM_2 \leftarrow \sigma_3 GM_3$

A5 (X) =
{ Y, R1, S, U, *GV:
?[$P \rightarrow X$]

?[Y → X]

?[Y → R0]

?[R0 → R1 {A_SbAt(X)}]

?[R1 → S]

-[N15 → U]

+[N15 → GV]

+[GV → U] }

A6(S, GV, U) =

{X, σ:

?[(GS ← σ X]

A7(σ, GV, U, X) }

A7(σ, GV, U, X)=

{GM*:

-[(GV→ U]

+[GV ← σ GM]

+[GM → U]

A6 (X, GM, U) }

A figura seguinte ilustra o efeito da execução das ações adaptativas A_5, A_6 e A_7 sobre um conjunto de três transições de empilhamento dos símbolos σ_1, σ_2, e σ_3.

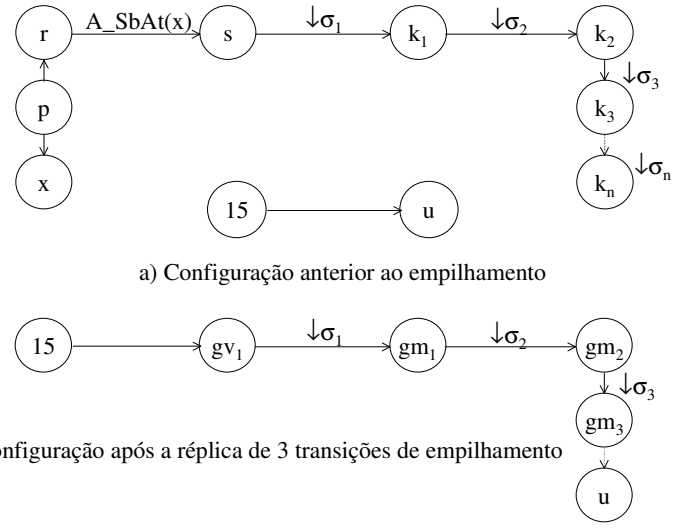


Fig. 24 - Resultado da ativação das funções A_5, A_6 e A_7 para a réplica de 3 símbolos

Avaliação de complexidade:

Se houver n regras de inserção de símbolos de contexto, tem-se que a gramática se expande em n regras (não adaptativas). São criados (n+1) símbolos não-terminais. A obtenção do novo conjunto de regras, implica na aplicação de (8 + n*4) regras.

Função adaptativa A_{26}:

Esta função adaptativa é ativada quando se detecta o fim da cadeia de símbolos que identificam um argumento, ou seja na ocorrência do símbolo delimitador (θ_1). A transição que a acionou é substituída por aquela inicial marcada com o símbolo "•", removida anteriormente pela ocorrência do mesmo símbolo. O símbolo "&" é passa a ser interpretado como delimitante de um parâmetro formal. A cadeia de inserção de símbolos concernentes ao argumento é associada ao seu respectivo parâmetro formal através da ligação de uma transição entre o estado onde é finalizado o identificador de parâmetro e aquele apontado pelo estado 15. Dessa forma, a extração de um parâmetro da cadeia de entrada, cujo identificador é finalizado no não-terminal W, no modo uso é sucedida pela inserção dos símbolos que constituem o identificador do argumento que lhe corresponde. Observe-se a figura 25.

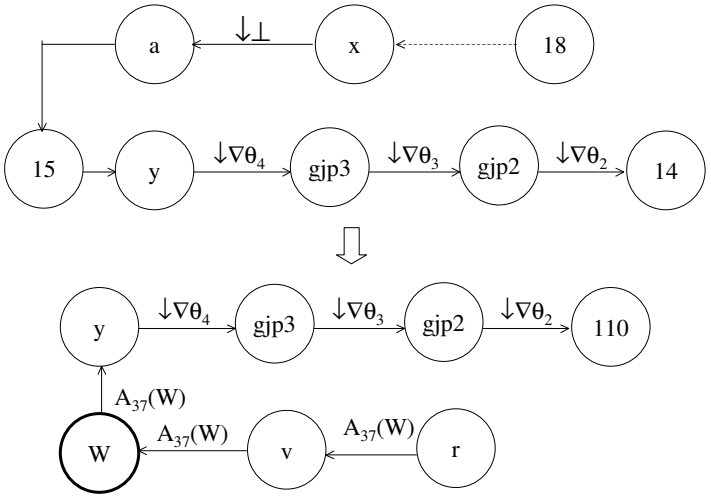


Figura 25 – Configuração de trecho do autômato após a execução da ação adaptativa A_{26}.

A_{26}(J, σ, K, W) =

{ X, Y, Z, R, S, t, V:

/*restaura transição em laço com o consumo do símbolo delimitador */

+[J → {A_{12} (σ), A_{23}( )} σ J] }

/*restaura transição consumindo símbolo •*/

-[J → {A_{26} (J, σ, K, W)} σ K]

+[J → {A_{26} (J, •, K, W)} • K]

/*restaura consumo do símbolo "&"

-[X → "&" N1]

+[N1 → "&" X]

-[S → {A_{22}( )} "&" N1]

+[N1 → {A_{22}( )} "&" S]

-["&" N80 → {A_{22}( )} e_com Z]

/*argumento associado ao parâmetro é desconectado da lista auxiliar:

é conectado ao identificador da regra e ao analisador léxico */

-[N15 → Y]

-[W → {A_{37}(W)} V]

-[R → {A_{37}(W)} W]

+[W → {A_{37}(W)} Y]

/*o identificador de parâmetros é apontado por uma lista de argumentos*/

+[V → {A_{37}(W)} W]

+[R → {A_{37}(W)} V]

-[W → {A_{37}(W)} V]

/*conexão ao léxico */

-[Z ← t N14]

+[Z ← t N110] }

Avaliação de Complexidade:

Esta função adaptativa é ativada sempre que for detectado o símbolo delimitador que finaliza o identificador de um argumento. Apresenta 8 ações elementares de inserção e 8 ações elementares de remoção e portanto, não altera o número de regras da gramática. A

execução da mesma consome o tempo de 16 ações elementares para cada parâmetro presente em uma regra.

Função Adaptativa A_{37}

Esta função adaptativa equivale a A_SbAt, ou seja mantém um ponteiro para o estado final de um identificador , que em A_{37} diz respeito a um parâmetro formal de regra. Tem-se:

$A_{37}(w) = \{X:$
$-[N90 \rightarrow X]$
$+[N90 \rightarrow W]\}$

Avaliação de Complexidade

$A_{37}(w)$ não altera o número de regras. O tempo de execução é o de duas regras elementares.

Com a conclusão de A_{26}, é finalizado também o reconhecimento do argumento. Sucede-se a inserção do corpo da regra na cadeia de entrada. Por outro lado, dado que a sintaxe das regras permite que estas se encontrem ora coordenadas entre si ora numa relação de subordinação, .prevê-se um mecanismo de contagem de regras em andamento associado ao corpo da regra., conforme a Fig. 12. Tal mecanismo adaptativo de contagem está associado às ações adaptativas A_{16} e A_{33}

As funções adaptativas A_{16}, A_{21} e A_{20}

A função adaptativa A_{16} é ativada em tempo de expansão, ou seja, sempre que no modo uso das regras ocorrer uma referência a uma determinada regra, esta é substituída pelo texto que lhe corresonde. Neste caso, o contador de regras em andamento é incrementado. A_{16}

o que o faz, mediante a ativação da função adaptativa A_{20}. Na primeira vez, A_{16} também executa A_{21}, responsável por configurar o analisador léxico para extrair nomes dos argumentos da regra.

Segue-se a descrição algébrica destas funções adaptativas.

A_{16}(J, K) = { f:

?[N100 →f N100]

{ A_{21} (f, j, k), A_{20}( ) } }

A_{21}(f, J, K) = {

-[N1 → {A_{22}( )} "&" N17]

+[N1 → "&" N70]

/* as duas alterações seguintes substituem a regra associada à execução da ação adaptativa A_{33}, por uma de inserção do meta-símbolo ■, associada à mesma ação. Este símbolo indica que não há nenhuma outra regra em expansão */

-[J → {A_{33}(J, K) } K]

+[J ← {A_{33}(J, K) } ■ K]

}

A_{20}( ) = { X, GJ*:

-[N100 → X]

+[N100 → GJ]

+[GJ → X]}

Avaliação de complexidade:

Estas ações acrescentam uma produção à gramática. Se se tratar da primeira ativação de A_{16}, o tempo de execução é o de 8 produções elementares, senão consome o tempo de exeucção de 4 produções elementares.

Função adaptativa A_{33}

O símbolo ■ deve ser inserido, para cada regra a ser processada, apenas uma vez na cadeia de entrada.. A ação adaptativa A_{33} substitui a transição responsável pela inserção do símbolo na cadeia de entrada, por uma transição em vazio. Tem-se:

$A_{33}(J, K) = \{ \sigma :$

$-[J \leftarrow \{A_{33}(J, K)\ \sigma\ K]$

$+ [J \rightarrow \{A_{33}(J, K)\ \ K]$

Avaliação de Complexidade:

A ação adaptativa A_{33} não altera o número de produções presentes na gramática. Consome o tempo de execução de duas ações elementares. É executada, apenas ao final do processamento de uma regra.

Para as situações de fim da substituição de um parâmetro por seu argumento, fim do tratamento da regra corrente e fim do processamento da regra em andamento, são inseridos na cadeia de entrada, para marcarem tais eventos, os seguintes símbolos: ⊥, ● e ■, a serem consumidos pelo analisador léxico e devidamente interpretados através das ações adaptativas, conforme o que se expõe a seguir.

Função adaptativa A_{32}

Quando é finalizado tratamento da regra corrente, há apenas que se decrementar o contador de macros em andamento.

$A_{32}(\ ) =$

$\{ X, y:$

$-[N100\ \rightarrow X]$

- [X → Y]

+[N100 → Y] }

Avaliação de Complexidade:

A função adaptativa A_{32} diminui em 1 o número de regras da gramática e seu desempenho é o de 3 regras elementares. É executada para toda a regra empregada ao longo da análise da sentença de entrada.

Função Adaptativa A_{18}

Esta função adaptativa é ativada pelo consumo do símbolo ⊥, indicador que foi finalizada a substituição de um parâmetro pelo seu respectivo argumento. A_{18} remove o parâmetro e o argumento de suas respectivas pilhas. Observe-se a Fig. 26

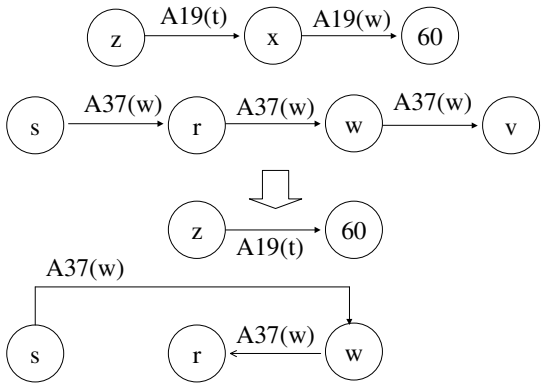


Fig. 26 – Funalização da Substituição de um Parâmetro por seu argumento

Segue-se a descrição algébrica de A_{18}

$A_{18}( )$ = { X, Z, R, S, t, V, W:

/*remove da pilha de ponteiros de parâmetros, o último parâmetro */

-[X → {A_{19} (W)} N60]

-[Z → {A_{19} (T)} X]

+[Z → {A_{19} (T)} N60]

/*remove da pilha de argumentos homônimos associado ao parâmetro W, o último argumento*/

-[W → {A_{37} (W)} V]

-[R → {A_{37} (W)} W]

-[S → {A_{37} (W)} R]

+[W → {A_{37}(W)} R]

+[S → {A_{37} (W)} W]

}

Avaliação de Complexidade:

Esta função adaptativa é executada para cada argumento presente no corpo da regra. A cada execução, o número de regras da gramática é decrementado em 2 e o custo no tempo é o de 8 ações elementares

Função Adaptativa A_{34}

Uma vez finalizado o processamento de uma regra, o próximo símbolo que ocorrer na cadeia de entrada deverá marcar o início de um parâmetro formal e portanto deverá ser mapeado como e_com pelo analisador léxico. A_{34} configura o analisador léxico para esta situação. Tem-se:

A_{34}( ) = {

-[1 → "&" N70]

+[1 ⇒ {A_{22}( ) } "&" N17]

}

3.3 Conclusões

Neste item teceram-se considerações sobre os desdobramentos da utilização dos mecanismos de expansão apresentados em (NETO, 1993) e empregados para o processamento da especificação da Linguagem Natural no tratamento de Dependências de Contexto.

A avaliação de complexidade de cada uma das técnicas aqui apresentadas revela desempenho sempre proporcional ao comprimento do texto de entrada, número de argumentos parâmetros e ocorrências de determinados símbolos.

Trata-se de mecanismo que efetua sucessivas cópias de regras originais, fundamentado na existência de um analisador léxico original e de uma máquina reconhecedora da sintaxe das regras de representação. O uso iterativo destas duas máquinas permitem a expansão do analisador léxico, criando uma camada capaz de tratar as ocorrências no modo uso de um conjunto de regras originais, cujo aprendizado foi anteriormente efetuado.

A utilização destes mecanismos permitem a implementação do tratamento de não-determinismos e ambigüidades apresentados em (NETO; MORAES, 2002). Por outro lado se a esta época a descrição da gramática a partir de uma aproximação livre de contexto, nos mecanismos aqui estudados, os atributos das estruturas e a concordância entre os mesmos são facilmente representados e processados automáticamente.

Como já afirmado na introdução deste capítulo, a conversão ***automática*** da representação de uma estrutura em uma seqüência de símbolos, permite inferir que a representação interna destas estruturas sejam processadas com o mesmo desempenho indicado no capítulo 2.

Levando-se em consideração os resultados apresentados neste capítulo e no que lhe antecede, o próximo tece algumas considerações adicionais sobre a proposta de uma arquitetura adaptativa para processamento de Linguagem Natural.