Autômatos Adaptativos no Tratamento Sintático de Linguagem Natural

Célia Yumi Okano Taniwaki
João José Neto
Escola Politécnica da USP – Av. Prof. Luciano Gualberto, trav.3, n.158
CEP 05508-900 – São Paulo – SP – Fone: (0xx11) 3091-5402
e-mail: ctani@uol.com.br
joao.jose@poli.usp.br

RESUMO

Este trabalho tem como objetivo principal mostrar a viabilidade da utilização de Autômatos Adaptativos como modelo de representação de informações no processamento de linguagem natural. Para verificar a viabilidade da utilização dos Autômatos Adaptativos, no procedimento de análise sintática de linguagem natural, este trabalho apresenta propostas de algoritmos de mapeamento do formalismo ATN e da Gramática Baseada em Restrição para Autômato Adaptativo.

ABSTRACT

This work is intended to show that the Adaptive Automata may be used as a model of natural language processing knowledge representation. In order to verify the factibility of Adaptive Automata in natural language syntactic analysis, this work proposes algorithms that map ATN and Constraint-Based Grammars into Adaptive Automata.

1. Introdução

O processamento de linguagem natural é uma área complexa da Inteligência Artificial que tem sido objeto de pesquisas há várias décadas. É um campo que está intimamente ligado a algumas áreas fora do escopo da ciência da computação, tais como a lingüística e a psicologia.

Seu objetivo é conseguir produzir programas de computador capazes de "entender", ou seja, analisar e interpretar a língua humana, processando-a e gerando uma resposta, como ocorre, por exemplo, em sistemas de respostas a perguntas, sistemas de tradução automática ou sistemas de criação de sumários de texto. Atualmente, com a crescente expansão da Internet e da *World Wide Web*, uma aplicação interessante e de grande utilidade do processamento de linguagem natural é a mineração de informações. Existem inúmeros mecanismos de busca disponíveis na Internet, que, na sua quase totalidade, restringem-se à busca através de palavras-chave e não pelo significado.

Para o processamento de uma linguagem natural, primeiramente é preciso que haja alguma forma de se processar a entrada e a saída das informações no computador, bem como um meio de codificá-la internamente ao processador.

Entre a entrada e a saída das informações, o processamento da sentença envolve as seguintes operações [Rich-93]: análise morfológica, análise sintática, análise semântica e análise pragmática. A análise morfológica procura atribuir uma classificação morfológica a cada palavra da sentença (como por exemplo, artigo, substantivo, verbo, etc.). A análise sintática procura identificar os relacionamentos sintáticos entre as seqüências lineares de palavras, produzindo, assim, a estrutura sintática da sentença (identifica, por exemplo, qual o sujeito, o predicado, o objeto direto, etc). A análise semântica procura mapear a estrutura sintática para o domínio da aplicação, fazendo com que a estrutura ganhe um significado. A análise pragmática procura reinterpretar a estrutura que representa o que foi dito, para determinar o que realmente se quis dizer.

Cada uma dessas operações é essencial e imprescindível para o bom processamento da linguagem natural, servindo de base para a operação seguinte. Este trabalho concentra-se principalmente na segunda operação: a análise sintática da linguagem natural.

Um dos principais formalismos de análise sintática e de representação das estruturas de linguagem natural é o modelo *Augmented Transition Network* (Rede de Transição Aumentada), desenvolvido por William Woods, em 1970 [Woods-70]. Pela sua simplicidade e flexibilidade, foi facilmente aceito, tornando-se, na década de 70 e em grande parte da década de 80, o formalismo dominante na implementação de sistemas de linguagem natural [Bates-93].

Atualmente, os formalismos mais utilizados para a análise sintática e descrição de linguagens baseiam-se na unificação e no uso de restrições e são conhecidos como Gramáticas baseadas em unificação ou Gramáticas baseadas em restrições. Existem várias variantes e as mais conhecidas são PATR, PATR-II, LFG (*Lexical Functional Grammar* – Gramática Léxica Funcional), GPSG (*Generalized Phrase Structure Grammar* – Gramática de Estrutura de Frase Generalizada) e HPSG (*Head-driven Phrase Structure Grammar* – Gramática de Estrutura de Frase Direcionada Pelo Núcleo).

Este trabalho pretende mostrar que os autômatos adaptativos também podem ser utilizados para não apenas representar as informações, como também para o procedimento da análise sintática de linguagem natural. Os autômatos adaptativos foram criados para solucionar, principalmente, algumas deficiências existentes no projeto de reconhecedores sintáticos de linguagens de programação.

Os Autômatos Adaptativos surgiram a partir, entre outras, da necessidade de eliminar alguns desvios conceituais existentes na maioria dos compiladores de linguagens de programação dirigidos por sintaxe. Problemas como o de dependência de contexto, estruturas de blocos e escopo das variáveis, tratamento de macros, consistência de uso dos tipos das variáveis, entre outros, deveriam ser considerados no nível sintático e não no semântico.

Como tais compiladores se baseiam em máquina de estados e pilhas, surgiu então o conceito de Autômato Adaptativo [José-94]. O Autômato Adaptativo é um modelo de máquina de estados que se caracteriza por sua adaptabilidade, isto é, sua capacidade de se auto-modificar à medida que vai reconhecendo a cadeia de entrada. Essa característica dinâmica o torna capaz de reconhecer linguagens sensíveis ao contexto, podendo tratar os problemas acima mencionados no nível sintático, como é desejado.

Como os autômatos adaptativos podem potencialmente apresentar poder computacional equivalente ao da máquina de Turing [Iwai-00], podendo, portanto, representar linguagens de qualquer complexidade, intuitivamente, eles também podem ser empregados no processamento de linguagem natural.

1.1. Objetivo e motivação

O objetivo deste trabalho é verificar a viabilidade de os autômatos adaptativos serem utilizados como analisadores sintáticos e modelos de representação de informações no processamento de linguagem natural.

Este trabalho tem, como principal motivação, mostrar a viabilidade prática da utilização do autômato adaptativo na área de linguagens naturais, especificamente no que tange à fase da análise sintática, visto que tal formalismo é capaz de reconhecer e representar linguagens sensíveis ao contexto.

2. Conceitos

O tratamento sintático é uma das etapas do processamento da linguagem natural, destinada a levantar a estrutura do relacionamento entre as partes da sentença, bem como classificá-las. A análise sintática prévia pode reduzir consideravelmente a complexidade global do sistema [Rich-93].

Para representar e analisar sintaticamente a linguagem natural, diversos formalismos foram criados ao longo das últimas décadas. Os formalismos existentes de representação de linguagem baseiam-se, ou na gramática da linguagem, através da qual é possível gerar sentenças válidas da linguagem em questão, ou em dispositivos reconhecedores da linguagem, chamados de máquinas de estados ou autômatos, que contêm o conjunto de regras de aceitação de cadeias dessa linguagem, verificando assim, se uma determinada cadeia de símbolos é ou não uma sentença válida.

Descreve-se, sucintamente, a seguir, os formalismos utilizados neste trabalho: *Augmented Transition Network* (ATN), Gramática de Estrutura de Frase Generalizada (GPSG) e os Autômatos Adaptativos.

2.1. *Augmented Transition Network* (ATN)

O formalismo *Aumented Transition Network* (ATN) [Woods-70] [Bates-78] foi desenvolvido a partir do modelo de Redes de Transição Recursivas, estendendo-se sua funcionalidade para melhor representar e analisar a linguagem natural.

Em uma ATN, cada arco pode estar associado a uma condição, que deve ser satisfeita para que o arco seja transitado, e a um conjunto de ações de construção de estrutura que será executado caso o arco seja transitado.

Um modelo ATN consiste num conjunto de redes ATN, tendo cada rede um rótulo diferente. A definição formal de uma rede ATN pode ser encontrada em [Bates-78].

A seguir, um exemplo de ATN, para a língua portuguesa, especificado segundo uma linguagem utilizada para representar uma rede ATN [Bates-78] [Woods-70]:

```
((S/        (PUSH SS/ T
            (SETR SUJ *)
            (SETR TIPO "DCL")
            (TO Q1))
   (CAT AUX T
            (SETR AUX *)
            (SETR TIPO "Q")
            (TO Q2)))
(Q1         (CAT V T
            (SETR AUX NIL)
```

```
            (SETR V *)
            (TO Q4))
      (CAT AUX T
            (SETR AUX *)
            (TO Q3)))
(Q2         (PUSH SS/ T
            (SETR SUJ *)
            (TO Q3)))
(Q3         (CAT V T
            (SETR V *)
            (TO Q4)))
(Q4         (POP (BUILD (S +++ (SV +)) TIPO SUJ AUX V) T)
            (PUSH SS/ T
                  (SETR SV (BUILD (SV (V +) *) V))
                  (TO Q5)))
(Q5         (POP (BUILD (S ++++) TIPO SUJ AUX SV) T)))
```

que corresponde ao grafo da Figura 1:

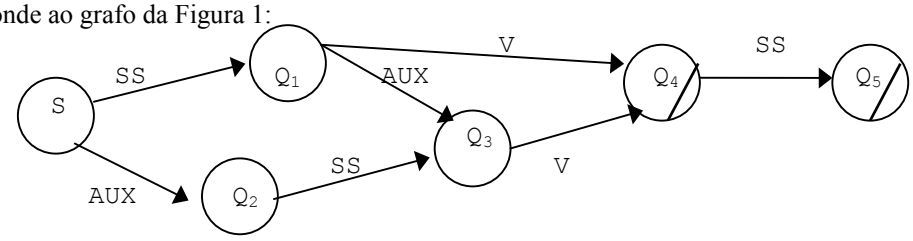


Figura 1 - Grafo de representação da ATN correspondente ao exemplo

sendo que SS quer dizer sintagma substantivo, AUX é o verbo auxiliar e V o verbo.

Após o reconhecimento da sentença 'O gato come o rato', os registradores conteriam os seguintes valores:

```
TIPO          DCL
SUJ           O gato
V             come
```

e a estrutura construída seria:

```
(S  DCL  (SS  O gato)  (SV  come o rato)))
```

O formalismo ATN provê, dessa forma, os mecanismos importantes para a análise sintática da linguagem natural: construção da árvore sintática durante o reconhecimento da sentença, imposição de condições para que se aplique a transição e o tratamento de dependência de contexto.

2.2. Gramática de Estrutura de Frase Generalizada (GPSG)

A Gramática de Estrutura de Frase Generalizada foi formalmente especificada em 1985, por Gerald Gazdar, Ewan Klein, Geoffrey Pullum e Ivan Sag [Gazdar et al.-85]. Foi desenvolvido na tentativa de se obter um sistema que, apesar de ser formalmente restritivo, pudesse lidar com vários tipos de fenômenos sintáticos e semânticos do processamento da linguagem natural. Através da GPSG, por exemplo, pode-se tratar o caso de dependências irrestritas sem a necessidade de efetuar transformações de estruturas. Esse formalismo é utilizado neste trabalho, por já existir publicada uma especificação da língua portuguesa em GPSG [Chin-96].

Na GPSG, as estruturas de características são bastante restritas. Elas são denominadas categorias, sendo que cada uma delas representa um conjunto de especificações característica-valor.

Na GPSG, existem vários tipos de regras [Gazdar et al.-85]:

- regras ID (*immediate dominance*) – são semelhantes às da gramática livre de contexto, embora não especifiquem a ordem entre os vários constituintes. A regra ID é do tipo $C_0 \rightarrow C_1, C_2, ..., C_n$, sendo C_0 a categoria-mãe, que tem o domínio imediato sobre as categorias-filhas $C_1, C_2, ... C_n$. .
- regras LP (*linear precedence*) – especificam a ordem linear de todos os constituintes que aparecem nas regras ID. A regra LP é do tipo $C_i < C_j < \ldots < C_y$, sendo que o símbolo < indica a relação de precedência entre os termos relacionados.
- meta-regras – regras que definem novas regras, baseadas em regras ID existentes, e são utilizadas para expressar relações entre estruturas, como por exemplo, entre a voz ativa e a passiva.
- regras de especificação de defaults (FSD) – definem os valores que certas características devem assumir quando não são especificados através de alguma regra.

A GPSG também define algumas restrições:

- restrições sobre a simultaneidade de ocorrência de características (FCR) – determinam a simultaneidade de ocorrência de características.

Existem três classes de características: HEAD ou nucleares, FOOT ou não-nucleares, e de CONTROLE. Maiores detalhes sobre essas classes podem ser encontrados em [Gazdar et al.-85].

Diz-se que uma regra ID $C_0 \rightarrow C_1, \ldots, C_n$ admite uma árvore local se e somente se a raiz dessa árvore tiver domínio imediato sobre os nós filhos, e se cada um dos nós filhos for uma extensão de cada uma das categorias filhas da regra. As características que estão presentes nas categorias da árvore podem ser herdadas ou instanciadas. As herdadas são aquelas determinadas pela própria regra ID. As instanciadas são as características presentes na árvore, porém não na regra. A regra ID, por si só, é bastante permissiva, admitindo por exemplo, uma árvore em que os sintagmas não tivessem concordância. Para resolver problemas como esse, a GPSG determina que a árvore também deve satisfazer a determinados princípios de instanciação de características:

- HFC (Head Feature Convention) – convenção das características nucleares – exige que as características nucleares da categoria-mãe sejam idênticas às características nucleares da categoria-filha nuclear, desde que não seja uma das características nucleares da categoria-mãe já impostas pela regra ID ou por FCR.
- CAP (Control Agreement Principle) – princípio de controle de concordância – na GPSG, a concordância é analisada como função, que define o alvo e o controlador da concordância. A categoria que corresponde ao alvo da concordância deve ter a característica de controle AGR, cujo valor é a categoria controladora e as especificações de concordância que ela deve ter. O princípio CAP força a identidade entre as características de um controlador e do controlado, para que haja concordância sintática entre eles.
- FFP (Foot Feature Principle) – princípio de características não-nucleares – estabelece que as características do grupo FOOT instanciadas na categoria-mãe sejam iguais à unificação das características FOOT instanciadas nas categorias-filhas. Dessa forma, as características não-nucleares das categorias-filhas podem ser passadas para a categoria-mãe. Assim, fenômenos como as dependências de longa distância e a formação de cláusulas relativas, com os quais as características FOOT estão relacionadas, podem ser devidamente tratados.

As diversas regras e princípios de uma gramática GPSG devem ser satisfeitas por uma árvore de estrutura de frase, que é um conjunto das diversas árvores locais admitidas e correspondentes a cada uma das regras, que também satisfazem todos os princípios de instanciação de características.

2.3. Autômatos Adaptativos

O Autômato Adaptativo consta de uma máquina de estados que se utiliza de uma memória organizada em pilha e tem características adaptáveis, pois permite que a configuração da máquina seja alterada dinamicamente, em função das transições efetuadas pelo autômato.

Essa característica dinâmica os torna capazes de representar e lidar com linguagens sensíveis ao contexto. São formalismos poderosos, apresentando poder computacional equivalente ao da máquina de Turing [Iwai-00]. Dessa maneira, podem também ser empregados no processamento de linguagem natural.

Seu conceito foi formalmente introduzido em [José-94], e sua notação posteriormente aprimorada [Iwai-00], fruto de uma longa pesquisa que visava buscar uma nova solução para a automatização da elaboração de reconhecedores sintáticos para linguagens sensíveis ao contexto.

O Autômato Adaptativo tem sua estrutura baseada no autômato de pilha estruturado [José-94], acrescido de ações adaptativas, que podem estar associadas às regras de transição. Dessa forma, o autômato adaptativo ganha o poder de reconhecimento de dependências de contexto, fenômeno cujo tratamento não é possível no autômato de pilha estruturado.

As ações adaptativas que estão associadas às transições correspondem a chamadas de funções adaptativas. As funções adaptativas, por sua vez, são compostas de ações adaptativas elementares (apresentadas a seguir), declarações de variáveis e de geradores [Iwai-00], e eventuais chamadas de funções adaptativas anteriores e posteriores à execução de suas ações adaptativas elementares.

Existem três tipos de ações adaptativas elementares: ação de inspeção (pesquisa a regra de transição indicada no conjunto de regras do autômato), ação de eliminação (elimina a regra de transição indicada do conjunto de regras do autômato) e ação de inserção (insere a regra de transição indicada no conjunto de regras do autômato).

O seguinte exemplo de autômato adaptativo ilustra sua aplicação prática no reconhecimento de uma linguagem sensível ao contexto.

Seja a linguagem que aceita como sentenças válidas as expressões definidas a seguir, mas que só permite o uso das variáveis “a” e “b” se elas forem previamente declaradas.

$$\text{PROG} \rightarrow (a|b|a,b|b,a) : S_2$$

$$S_2 \rightarrow (a|b|<S_2>)((+|-|*|/)(a|b|<S_2>))^*$$

Essa linguagem pode ser representada pelo autômato adaptativo ilustrado na Figura 2:

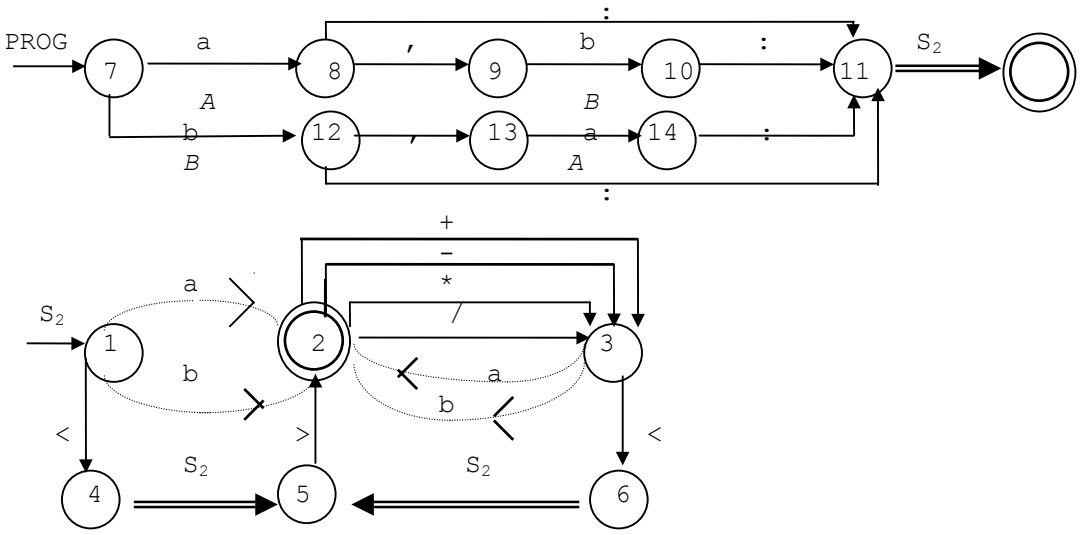


Figura 2 - Exemplo de autômato adaptativo

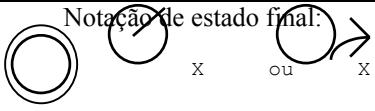
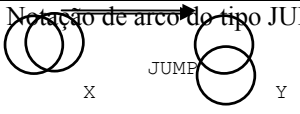
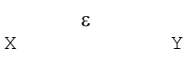
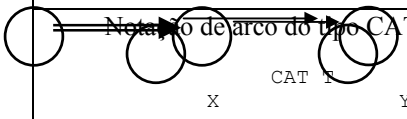
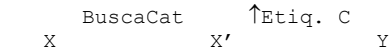
As transições representadas pelos arcos tracejados não existem na versão inicial do autômato adaptativo. Quando a variável 'a' é encontrada na cadeia de entrada, a ação adaptativa *A*, executada na transição 7-8 ou 13-14 cria os arcos 1→2 e 3→2 com a variável 'a'. Da mesma forma, quando a variável 'b' é encontrada na cadeia de entrada, a ação adaptativa *B*, executada na transição 9-10 ou 7-12 cria os arcos 1→2 e 3→2 com a variável 'b'. Assim, os símbolos 'a', 'b' ou ambos, quando declarados à esquerda do sinal ':' na sentença analisada, deverão ser aceitos nas expressões à direita de ':', e rejeitados caso contrário, caracterizando a dependência de contexto da linguagem.

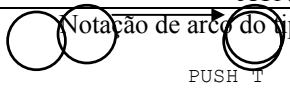
3. Mapeamento de Redes ATN para Autômato Adaptativo

Intuitivamente, pode-se perceber a existência de uma certa correspondência de estrutura entre ATN e Autômato Adaptativo, pois ambos se compõem de estados e transições, e possuem mecanismos para o tratamento sintático de casos de dependências de contexto.

Uma vez comprovada a existência de tal correspondência, pode-se concluir que o Autômato Adaptativo é capaz de representar a linguagem natural, à medida que o ATN tem tal capacidade, reconhecendo a linguagem segundo as suas regras sintáticas e determinando a estrutura sintática correspondente.

Pretende-se, assim, indicar uma forma de mapeamento do ATN para Autômato Adaptativo, para através disto provar a correspondência entre esses dois formalismos. Para cada possível construção básica de ATN, estabelece-se uma correspondência com a construção equivalente do Autômato Adaptativo, segundo a tabela a seguir:

ATN	Autômato Adaptativo
Notação de estado final:  	Notação de estado final:
Notação de arco do tipo JUMP:  	Notação de transição em vazio: 
Notação de arco do tipo CAT:  	 BuscaCat é uma submáquina que verifica qual a categoria léxica da palavra da cadeia de entrada e insere na cadeia de entrada uma etiqueta correspondente a essa categoria léxica.

ATN	Autômato Adaptativo
Notação de arco do tipo PUSH:  	 T é uma submáquina específica (ou seja, um autômato completo) capaz de reconhecer a classe sintática T.
Transição não especificada	Transição para um estado de erro

Estendendo-se o raciocínio para os testes e ações contidos nos comandos ATN, tem-se o seguinte mapeamento:

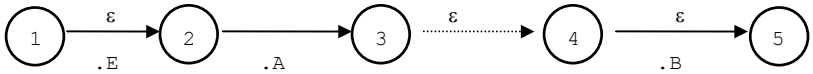
A maioria dos arcos possuem o formato:

```
(ARCO <informação> <teste> <ação>* (TO <próx-estado>))
```

O tipo de arco e sua <informação> (que depende do tipo de arco) devem ser mapeados para uma transição do autômato adaptativo, conforme já foi descrito.

A execução de <ação>* deve ser mapeada para uma função correspondente à ação adaptativa posterior associada a essa transição.

Quando <teste> não for apenas T (true), a execução de <teste> deve ser mapeada para uma função correspondente à ação adaptativa anterior associada a essa transição. Nesse caso, devem ser inseridas algumas transições intermediárias que preparam o autômato para prosseguir somente quando <teste> for verdadeiro. Esse mapeamento é representado através do diagrama a seguir:



Antes da transição que corresponde ao arco ATN (transição 2-3 do diagrama anterior), deve ser inserida uma transição em vazio (transição 1-2), associada a uma ação adaptativa posterior (`E`) que elimina a transição 3-4, caso ela exista. Em seguida, deve ser criada a transição que corresponde ao arco ATN propriamente dito (transição 2-3), associada a uma ação posterior (`A`) que corresponde à execução de <teste>. Se teste for verdadeiro, deve inserir a transição 3-4, como prosseguimento da transição que está sendo criada. No estado correspondente ao final da transição 3-4, deve ser criada uma transição intermediária (transição 4-5), em vazio, associada a uma ação adaptativa posterior (`B`) que corresponde à execução de <ação>.

Tanto em <teste> como em <ação>, podem existir chamadas de funções da linguagem LISP. Se existirem, essas funções também deverão ser mapeadas para Autômato Adaptativo. Como é possível escrever uma rede ATN sem a utilização de funções LISP, este trabalho concentrou-se apenas em mapear as funções e arcos do ATN propriamente dito. Para que qualquer rede ATN que contenha funções LISP pudesse ser mapeada para Autômato Adaptativo, o ideal seria que houvesse um interpretador LISP, cuja implementação se baseasse em Autômato Adaptativo, isto é, um interpretador LISP que utilizasse o Autômato Adaptativo como linguagem de máquina.

Cada entrada léxica ATN também deve ser mapeada para um fragmento de autômato adaptativo, de forma que o item léxico corresponda a um estado inicial, a partir do qual partam transições correspondentes a cada uma de suas características para estados que correspondam aos respectivos valores das características.

Por exemplo, seja a seguinte entrada léxica:

```
comeu –      CAT=V (Verbo), VFORM=FIN (Forma Finita), RAIZ=COMER, PESSOA=3,
             NUMERO=SG (Singular)
```

Essa entrada léxica deve ser mapeada para o seguinte fragmento de autômato:

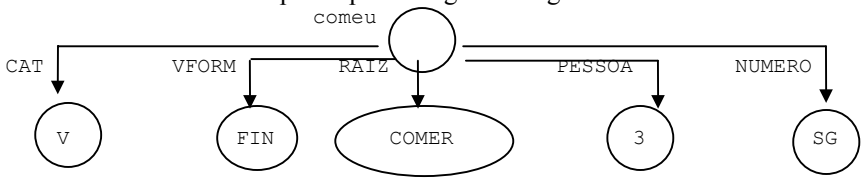


Figura 3 - Autômato correspondente à entrada léxica 'comeu'.

Utilizando-se o analisador e etiquetador morfológico desenvolvido em [Menezes-00], pode-se criar esses fragmentos de autômatos correspondentes a cada item léxico, à medida que se lê o texto gerado por esse analisador morfológico.

Esse texto consiste em palavras seguidas de etiquetas morfológicas. Assim, o algoritmo de mapeamento léxico pode ler cada palavra, criar seu fragmento de autômato correspondente contendo suas informações léxicas, e substituir na cadeia de entrada essa palavra e suas etiquetas por uma etiqueta que representa a palavra. Essa etiqueta está associada, através de uma tabela e através de uma lista ligada

implementada por intermédio de um autômato, à palavra que representa e ao início do fragmento do autômato que descreve suas informações léxicas.

Por exemplo, a sentença ‘O gato comeu o rato.’ é representada pela lista ligada ilustrada pela Figura 4.

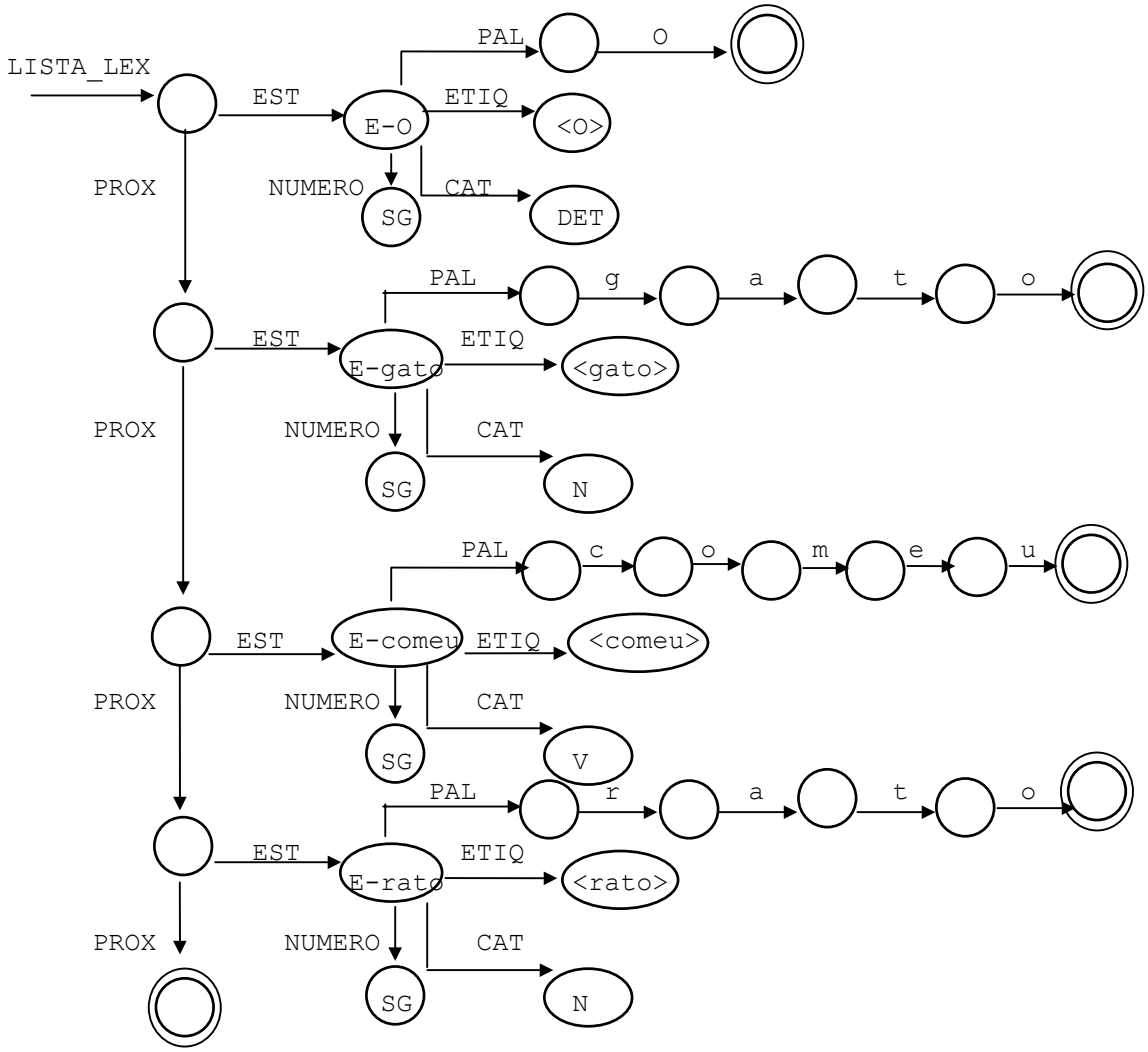


Figura 4 - Autômato que representa a lista léxica correspondente à sentença 'O gato comeu o rato.'

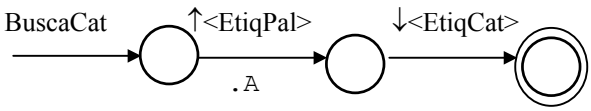
Nesse exemplo, percebe-se que houve o reaproveitamento da entrada léxica da palavra ‘o’, que aparece duas vezes na sentença que está sendo representada. O início do autômato é apontado por LISTA_LEX. A transição ‘EST’ aponta para um estado ‘E-palavra’, correspondente ao estado que contém as informações sobre a palavra. A transição ‘PROX’ aponta para a próxima entrada da lista. A transição ‘PAL’ aponta para a representação da palavra em si, a transição ‘ETIQ’ aponta para o estado que contém a representação da etiqueta da palavra (<palavra>) e as demais transições correspondem aos valores léxicos e sintáticos da palavra. Por questões de simplicidade e clareza, na Figura 4, a ilustração do estado ‘N’ (nome) foi repetida para cada substantivo da sentença. O mesmo ocorreu para o estado ‘SG’ (singular). Mas o autômato real terá apenas um estado ‘N’ e apenas um estado ‘SG’, que serão comuns a todas as palavras que os utilizam.

As rotinas relacionadas ao mapeamento de Rede ATN para Autômato Adaptativo são descritas a seguir, de forma sucinta. A descrição completa desses algoritmos encontra-se em [Taniwaki-01].

Rotina de mapeamento das informações léxicas.

A rotina MapeamentoLexico processa a cadeia de entrada e cria o autômato correspondente à lista léxica, substituindo na cadeia de entrada a palavra e sua etiquetas morfológicas pela etiqueta que representa essa palavra.

A submáquina BuscaCat é responsável por reconhecer a etiqueta da palavra na cadeia de entrada e procurá-la na lista léxica, inserindo na cadeia de entrada a etiqueta correspondente à sua categoria léxica. O diagrama a seguir ilustra o funcionamento da submáquina BuscaCat:



Rotina da ação adaptativa de BuscaCat:

Esta rotina percorre a lista léxica procurando a etiqueta da palavra e devolve a etiqueta da categoria correspondente e o estado ESTR_CORR (estrutura corrente) apontando para o estado correspondente à palavra que está sendo reconhecida.

Rotina de mapeamento de rede ATN para Autômato Adaptativo.

A rotina MapeamentoRedeATN-AA recebe uma rede ATN no seguinte formato:

```
(Nome-da-rede
 (Estado1      (Arco1)
               ....
               (ArcoM))
.....
 (EstadoN      (Arco1)
               ....
               (ArcoX)))
```

e cria o autômato adaptativo correspondente, mapeando devidamente cada um dos arcos ATN, e seus respectivos testes e ações, conforme explicado anteriormente.

Rotina que prepara o autômato para a execução de <teste>.

A rotina PreparaTeste prepara o autômato para a execução de <teste> e cria a ação adaptativa responsável pela execução de <teste>.

Rotina que trata <ação>.

A rotina TrataAção cria a ação adaptativa responsável pela execução de <ação>.

Rotina que trata <expressão> contida na <ação>.

A rotina TrataExpressão cria as ações adaptativas elementares que tratam <expressão> contida em <ação> ou em <teste>. Devolve o estado RetornoExpr apontando para o estado que representa o resultado de <expressão>.

Rotinas que tratam <teste> e <teste <tipo-do-constituinte>>.

As rotinas TrataTeste e TrataTesteTipoConstituinte criam as ações adaptativas elementares que tratam <teste> e <teste <tipo-do-constituinte>>, respectivamente. Devolvem o estado RetornoTeste apontando para o estado Verdadeiro ou Falso, dependendo do resultado do teste.

Como exemplo da aplicação dos algoritmos propostos, seja a seguinte Rede ATN, que define um sintagma substantivo da gramática da língua portuguesa. Essa rede foi obtida adaptando-se um exemplo de Rede ATN, que define o sintagma substantivo da gramática da língua inglesa, extraído de [Bates-78].

```
(NP/
 (REGS DET NUMDET GENDET N NUM GEN PP)
 (NP/
     (CAT DET T
          (SETR DET *)
          (SETR NUMDET (GETF NUMERO))
          (SETR GENDET (GETF GENERO))
          (TO NP/DET))
     (CAT N T
          (SETR N *)
          (SETR NUM (GETF NUMERO))
          (SETR GEN (GETF GENERO))
          (TO NP/N3)))
 (NP/DET
     (CAT N T
          (SETR N *)
          (SETR NUM (GETF NUMERO))
          (SETR GEN (GETF GENERO))
          (TO NP/N1)))
 (NP/N1
     (JUMP NP/N2 (AND       (AGREE (GETR NUMDET) (GETR NUM))
```

```
                              (AGREE (GETR GENDET) (GETR GEN))))
    (NP/N2
        (PUSH PP/ T (SETR PP *) (TO NP/PP1))
        (POP (BUILD (+ + + +) DET NUM GEN N) T))
    (NP/PP1
        (PUSH PP/ T (SETR PP (APPEND PP *)) (TO NP/PP1))
        (POP T (BUILD (+ + + + +) DET NUM GEN N PP)))
   (NP/N3
        (PUSH PP/ T (SETR PP *) (TO NP/PP2))
        (POP (BUILD (+ + +) NUM GEN N) T))
    (NP/PP2
        (PUSH PP/ T (SETR PP (APPEND PP *)) (TO NP/PP2))
        (POP T (BUILD (+ + + +) NUM GEN N PP))))
```

Essa rede pode ser representada pelo seguinte diagrama:

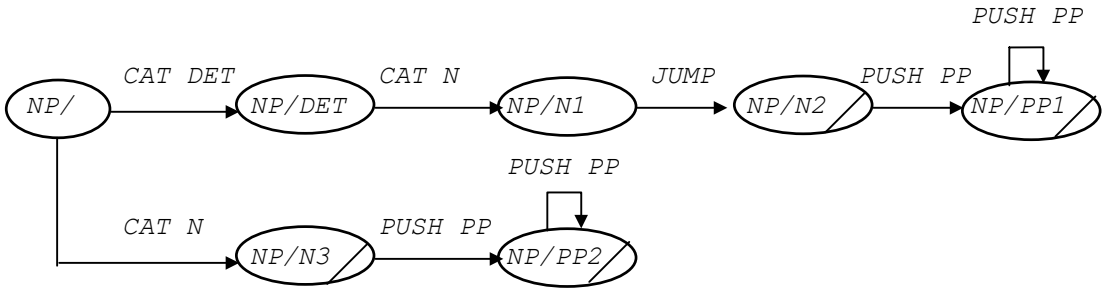


Figura 5 - Rede ATN correspondente à Rede NP/.

Submetendo-se essa rede ao algoritmo de mapeamento de Rede ATN para Autômato Adaptativo, obtém-se o autômato adaptativo ilustrado na Figura 6, a seguir. A simulação passo a passo desse mapeamento e a descrição completa das ações adaptativas são descritas em [Taniwaki-01].

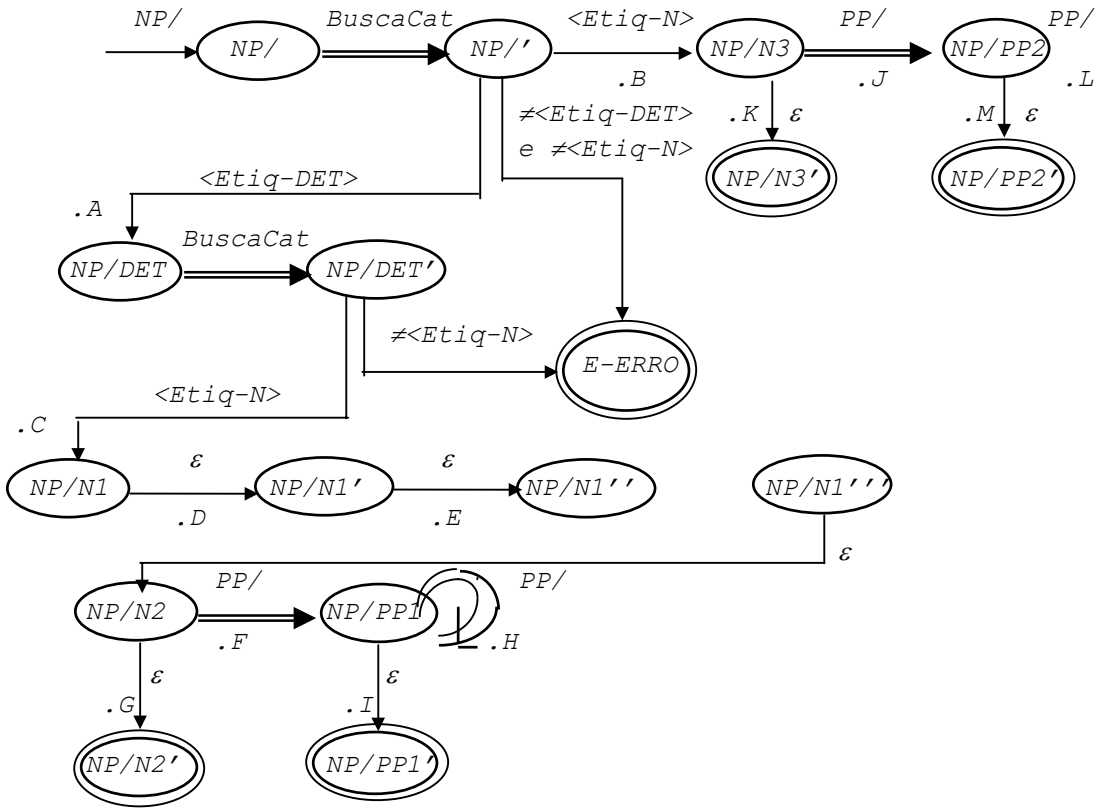


Figura 6 - Autômato adaptativo correspondente ao mapeamento da Rede NP/.

Ação adaptativa A insere a transição do estado E-DET para o estado correspondente ao Determinante, insere a transição do estado E-NUMDET para o estado correspondente à característica NUMERO do Determinante e insere a transição do estado E-GENDET para o estado correspondente à característica GENERO do Determinante.

Ação adaptativa B insere a transição do estado E-N para o estado correspondente ao Nome, insere a transição do estado E-NUM para o estado correspondente à característica

NUMERO de Nome e insere a transição do estado E-GEN para o estado correspondente à característica GENERO de Nome.

Ação adaptativa C insere a transição do estado E-N para o estado correspondente ao Nome, insere a transição do estado E-NUM para o estado correspondente à característica NUMERO de Nome e insere a transição do estado E-GEN para o estado correspondente à característica GENERO de Nome.

Ação adaptativa D elimina a transição de NP/N1'' para NP/N1'''.

Ação adaptativa E testa se E-NUMDET e E-NUM apontam para o mesmo estado e se E-GEN e E-GENDET apontam para o mesmo estado. Caso seja verdadeiro, insere a transição de NP/N1'' para NP/N1'''.

Ações adaptativas F, H, J e L inserem a transição de E-PP para o estado correspondente ao sintagma preposicional.

Ação adaptativa G insere as transições de EBI para E-DET, E-NUM, E-GEN, E-N, e a transição de ESTR_CORR para EBI.

Ação adaptativa I insere as transições de EBI para E-DET, E-NUM, E-GEN, E-N, E-PP, e a transição de ESTR_CORR para EBI.

Ação adaptativa K insere as transições de EBI para E-NUM, E-GEN, E-N, e a transição de ESTR_CORR para EBI.

Ação adaptativa M insere as transições de EBI para E-NUM, E-GEN, E-N, E-PP, e a transição de ESTR_CORR para EBI.

Seja também a seguinte Rede ATN, que especifica um sintagma preposicional da gramática da língua portuguesa. Essa rede foi obtida adaptando-se um exemplo equivalente à língua inglesa, extraído de [Bates-78]:

```
(PP/
 (REGS PREP NP)
 (PP/
    (CAT PREP T (SETR PREP *) (TO PP/PREP)))
 (PP/PREP
    (PUSH NP/ T (SETR NP *) (TO PP/NP)))
 (PP/NP
    (POP (BUILD (+ +) PREP NP) T)))
```

Essa rede pode ser representada pelo seguinte diagrama:

Figura 7 - Rede ATN correspondente à Rede PP/.

Submetendo-se a rede PP/ ao algoritmo MapeamentoRedeATN-AA, obtém-se o autômato ilustrado pela Figura 8.

As ações adaptativas especificadas na Figura 8 são descritas sucintamente a seguir (a descrição completa encontra-se em [Taniwaki-01]):

Ação adaptativa N insere a transição do estado E-PREP para o estado correspondente à Preposição.

Ação adaptativa O insere a transição de E-NP para o estado correspondente ao sintagma substantivo.

Ação adaptativa P insere as transições de EBI para E-PREP, E-NP, e a transição de ESTR_CORR para EBI.

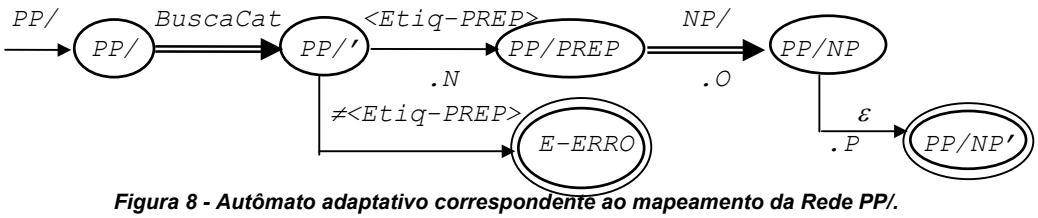


Figura 8 - Autômato adaptativo correspondente ao mapeamento da Rede PP/.

Submetendo-se o sintagma substantivo 'a destruição da cidade' ao analisador e etiquetador morfológico de [Menezes-00], obtém-se:

```
a /D-F destruição /N da /P+D-F cidade/N
```

Submetendo-se a saída do analisador e etiquetador morfológico referente ao sintagma substantivo 'a destruição da cidade' à rotina MapeamentoLexico, e submetendo-se, em seguida, a saída da rotina MapeamentoLéxico referente ao sintagma substantivo 'a destruição da cidade' aos autômatos adaptativos gerados pelo mapeamento das redes ATN NP/ e PP/, obtém-se a estrutura ilustrada na Figura 9, correspondente à análise sintática do sintagma.

Observa-se neste exemplo, que houve a verificação da concordância entre o determinante e o substantivo em 'a destruição' e em 'a cidade'.

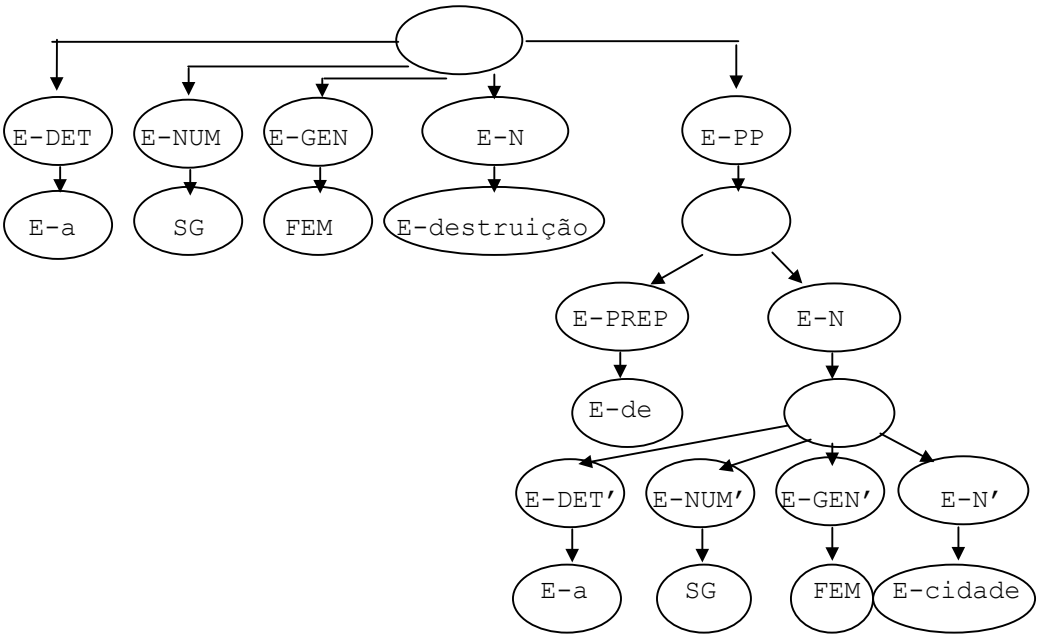


Figura 9 - Estrutura obtida como resultado do reconhecimento do sintagma substantivo 'a destruição da cidade', pelos autômatos adaptativos correspondentes às Redes NP/ e PP/.

Comprova-se, assim, que os autômatos adaptativos resultantes do mapeamento das Redes ATN NP/ e PP/ para Autômato Adaptativo são capazes de analisar sintaticamente um sintagma substantivo e criar uma estrutura sintática em decorrência dessa análise.

Novos experimentos podem ser realizados, mapeando-se os demais sintagmas de uma sentença, e assim, obter um analisador sintático representado através de Autômato Adaptativo, capaz de reconhecer sentenças da língua portuguesa.

Através dos algoritmos aqui propostos, é possível, assim, mapear uma rede ATN para um Autômato Adaptativo e, dessa forma, atinge-se o objetivo proposto de se verificar que o Autômato Adaptativo pode ser usado na representação e análise sintática do processamento de linguagem natural.

4. Mapeamento de GPSG para Formalismo Adaptativo

O GPSG foi o formalismo escolhido para representar a classe de formalismos baseados em restrições, que correspondem aos formalismos mais empregados recentemente para representar a linguagem natural. Apesar das suas limitações, o GPSG foi eleito para ser utilizado no desenvolvimento da proposta devido, principalmente, ao fato de já existir publicada uma especificação da gramática da língua portuguesa nessa notação.

Assim como foi feito para o ATN e o Autômato Adaptativo, pretende-se indicar uma forma de mapeamento da GPSG para um Formalismo Adaptativo, no caso, o Autômato Adaptativo, e, assim, provar que qualquer especificação GPSG de uma determinada linguagem natural pode ser mapeada para uma representação em Autômato Adaptativo. Uma vez comprovada a existência desse mapeamento, pode-se concluir que o Autômato Adaptativo é capaz de representar a linguagem natural, à medida que a GPSG tem tal capacidade, reconhecendo a linguagem segundo as suas regras sintáticas e determinando a estrutura sintática correspondente.

Primeiramente, cada entrada léxica GPSG deve ser mapeada para um fragmento de autômato adaptativo, de forma que o item léxico corresponda a um estado inicial, a partir do qual partam transições correspondentes a cada uma de suas características para estados que correspondam aos respectivos valores das características.

Por exemplo, seja a seguinte entrada léxica:

```
preferiu [N-, V+, BAR 0, SUBCAT n, PAST+,
          VFORM FIN, AGR NP[PER 3, PLU-]]
```

Essa entrada léxica deve ser mapeada para o fragmento de autômato, ilustrado na Figura 10. Observa-se, no exemplo, que no mapeamento léxico, há o aproveitamento de estados que correspondem a valores repetidos de características, como é o caso do estado que representa o valor '-' e é compartilhado pelas transições 'N' e 'PLU'.

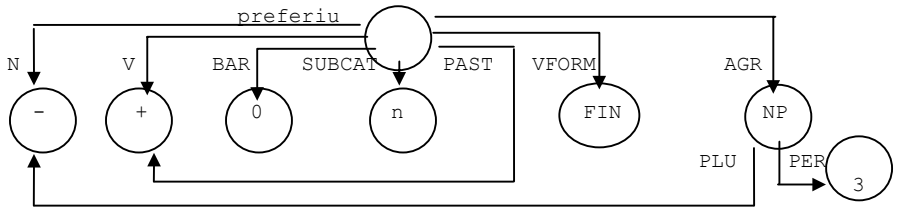


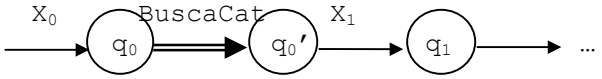
Figura 10 - Autômato correspondente à entrada léxica 'preferiu'.

Como foi explicado no mapeamento de Rede ATN para Autômato Adaptativo, utilizando-se o analisador e etiquetador morfológico desenvolvido em [Menezes-00], pode-se criar esses fragmentos de autômatos correspondentes a cada item léxico conforme se lê o texto gerado por esse analisador morfológico.

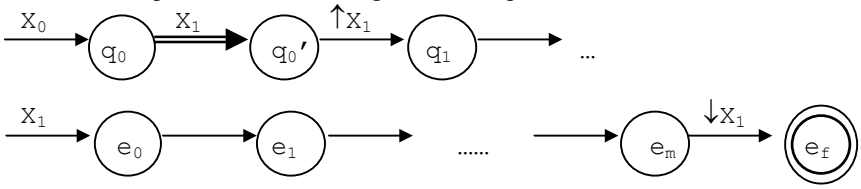
Pode-se, assim, utilizar um algoritmo semelhante ao de mapeamento léxico (Rotina MapeamentoLexico), descrito no item anterior, para realizar o mapeamento léxico da sentença que será submetida ao autômato adaptativo resultante do mapeamento da especificação GPSG.

Uma regra ID $X_0 \rightarrow X_1, \ldots, X_n$ (básica ou derivada de uma meta-regra) de uma especificação GPSG pode ser mapeada para Autômato Adaptativo da seguinte forma:

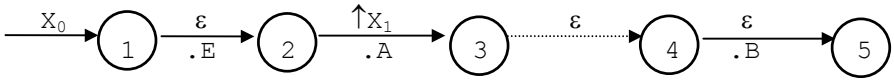
- a categoria-mãe X_0 corresponderá a um autômato adaptativo, com um estado inicial q_0.
- cada categoria-filha X_i corresponderá a duas transições desse autômato.
  - se X_i corresponder a um item léxico, então ela será mapeada para duas transições: a primeira será uma chamada da submáquina BuscaCat e a segunda tem como estímulo a etiqueta correspondente à categoria léxica de X_i. A submáquina BuscaCat é a mesma que é utilizada no mapeamento de Rede ATN para Autômato Adaptativo, descrita anteriormente neste trabalho.



  - se X_i não corresponder a um item léxico, então ela será mapeada para duas transições: a primeira será uma chamada de uma sub-máquina do autômato adaptativo que trata essa categoria e que, em caso de sucesso, insere na cadeia de entrada uma etiqueta correspondente à categoria tratada; e a segunda transição terá como estímulo a etiqueta da cadeia de entrada correspondente a essa categoria, desempilhando-a da cadeia de entrada.



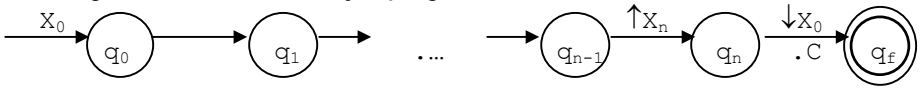
- cada transição com estímulo corresponderá a uma ação adaptativa posterior, que processará um teste para verificar se realmente a transição poderá ser processada. Esse teste consiste em conferir se as características do item léxico ou da categoria são compatíveis com a categoria-filha X_i correspondente. Se tais características forem compatíveis, então essa transição poderá ser processada. O processamento dessa transição representa o reconhecimento da categoria-filha X_i pelo autômato. Para que esse trecho do algoritmo possa ser executado várias vezes, sempre que houver teste a ser executado, devem ser inseridas algumas transições intermediárias que preparam o autômato para prosseguir somente quando o resultado do teste for verdadeiro. Nesse caso, esse mapeamento é representado através do diagrama a seguir:



  Antes da transição que corresponde à categoria-filha X_1, deve ser inserida uma transição em vazio (transição 1-2), associada a uma ação adaptativa posterior (`E`) que elimina a transição 3-4, caso ela exista. Em seguida, deve ser criada a transição que corresponde à categoria-filha X_1 propriamente dita, associada a uma ação posterior (`A`) que corresponde à execução do teste. Se teste for verdadeiro, deve inserir a transição 3-4, como prosseguimento da transição que está sendo criada. No estado correspondente ao final da transição 3-4, deve ser criada uma transição intermediária (transição 4-5).
- Cada transição com estímulo corresponderá a uma ação adaptativa posterior (`B`), que criará um estado correspondente à categoria reconhecida e uma transição partindo desse estado para o estado inicial do item sendo tratado. Além disso, verificará se existe alguma característica não-

nuclear instanciada para a categoria correspondente. Em caso afirmativo, cria uma transição, a partir do estado correspondente a essa categoria, para o estado correspondente a essa característica. No diagrama, essa ação posterior é associada à última transição em vazio (transição 4-5), que é executada após a verificação de que o resultado do teste é verdadeiro.

- Após o reconhecimento da última categoria-filha X_n da regra, cria-se mais uma transição em vazio para um estado final, cuja ação posterior consistirá em:



  - Criar um estado que corresponda à categoria X_0, e criar transições entre X_0 e X_1, X_0 e X_2, e assim sucessivamente, até X_0 e X_n.
  - Verificar se existe concordância sintática entre as categorias da regra, ou seja, verificar se a regra satisfaz o princípio CAP: para cada par de categorias-filhas X_i e X_j, tais que X_i controla X_j, unificar as características de concordância de X_i e X_j.
  - Verificar se a regra satisfaz o princípio FFP: para cada característica não-nuclear instanciada de X_i e não definida de X_0, acrescentar essa característica a X_0, e seu valor será a unificação dos valores dessa característica para todo X_i que tiver essa característica instanciada.
  - Verifica se a regra satisfaz o princípio HFC: unificar cada característica nuclear da categoria-filha nuclear X_h, com as características nucleares da categoria-mãe X_0.

A aplicação dos princípios CAP, FFP e HFC, assim como a aplicação do FSD no mapeamento sugerido por este trabalho baseia-se no mapeamento do formalismo GPSG para o formalismo PATR, desenvolvido em [Shieber-88].

No mapeamento sugerido neste trabalho, por questões de simplicidade, supõe-se que as categorias-filhas da regra ID devem ocorrer na ordem em que estão dispostas na regra. Não é difícil alterar posteriormente o mapeamento para que aceite que as categorias-filhas ocorram em qualquer ordem.
Também não estão sendo consideradas, neste mapeamento, as restrições FCR (restrições sobre a simultaneidade de ocorrência de características), uma vez que é possível realizar um pré-processamento das regras GPSG para que essas restrições sejam impostas.

Haverá apenas um autômato adaptativo para cada categoria-mãe X_0, ou seja, se houver mais de uma regra ID correspondente à mesma categoria-mãe X_0, então todas essas regras serão mapeadas para um único autômato adaptativo.

São descritas sucintamente, a seguir, as diversas rotinas relacionadas ao mapeamento de uma regra ID para autômato adaptativo. A descrição completa dos algoritmos encontra-se em [Taniwaki-01].

Rotina de mapeamento de uma regra ID para autômato adaptativo.

Seja uma regra ID $X_0 \rightarrow X_1, \ldots, X_n$ (básica ou derivada de uma meta-regra) de uma especificação GPSG, sendo que uma categoria X_i ($1 \leq i \leq n$) denotada entre parênteses significa que essa categoria é opcional, e uma categoria X_i ($1 \leq i \leq n$) denotada entre chaves significa que essa categoria pode ser repetida uma ou mais vezes. Essa regra pode ser mapeada para autômato adaptativo através da rotina MapeamentoRegraID-AA, que cria o autômato adaptativo correspondente, mapeando devidamente cada uma das categorias da regra, conforme explicado anteriormente.

Rotina que prepara o autômato para a execução do teste.

A rotina PreparaTesteGPSG prepara o autômato para a execução do teste e cria a ação adaptativa responsável pela execução dos testes.

Rotina que cria a ação adaptativa correspondente ao teste das características da categoria-filha X_i de uma regra ID.

A rotina TrataTesteGPSG cria as ações adaptativas elementares que processam o teste das características de X_i (categoria sendo reconhecida). Devolve o estado RetornoTeste apontando para o estado Verdadeiro ou Falso, dependendo do resultado do teste.

Rotina que cria a ação adaptativa posterior da transição que corresponde ao reconhecimento da categoria-filha X_i de uma regra ID.

A rotina TrataAçãoGPSG cria a ação adaptativa correspondente ao reconhecimento da categoria-filha X_i (cria um estado correspondente à categoria reconhecida e uma transição partindo desse estado para o estado inicial do item sendo tratado). Além disso, verifica se há alguma característica não-nuclear instanciada para a categoria correspondente. Se houver, cria uma transição a partir do estado correspondente a essa categoria para o estado correspondente a essa característica.

Rotina que cria a ação adaptativa posterior da transição que corresponde ao final do reconhecimento de uma regra ID.

A rotina ReconheceRegra cria a ação adaptativa correspondente ao reconhecimento da categoria-mãe X_0. Esta ação monta a estrutura final, contendo as informações da categoria-mãe X_0 que está sendo reconhecida (cria um estado correspondente à categoria reconhecida e uma transição partindo desse estado para o estado correspondente a X_1, uma transição do estado correspondente a X_0 para o estado correspondente a X_2, e assim por diante). Além disso, verifica se a regra satisfaz aos princípios CAP, FFP e HFC.

Rotinas UnifiqueConcordância, Unifique e UnifiqueValor.

Estas rotinas realizam a unificação de características, utilizadas para a verificação dos princípios CAP, FFP e HFC. Devolvem o estado Verdadeiro ou Falso, dependendo do resultado da unificação e, em caso de sucesso, devolvem a unificação das características.

Mediante os algoritmos descritos, pode-se converter, assim, uma especificação GPSG para Autômato Adaptativo.

Sejam, por exemplo, as seguintes regras ID:

(1) a. N1 → H[1.0]
    b. N1 → H[1.1], P2[de]

O autômato adaptativo resultante do mapeamento dessas regras é (a simulação passo a passo dos algoritmos de mapeamento GPSG para Autômato Adaptativo encontra-se em [Taniwaki-01]):

As ações adaptativas que aparecem na Figura 11 são descritas sucintamente a seguir e sua descrição detalhada encontra-se em [Taniwaki-01]:

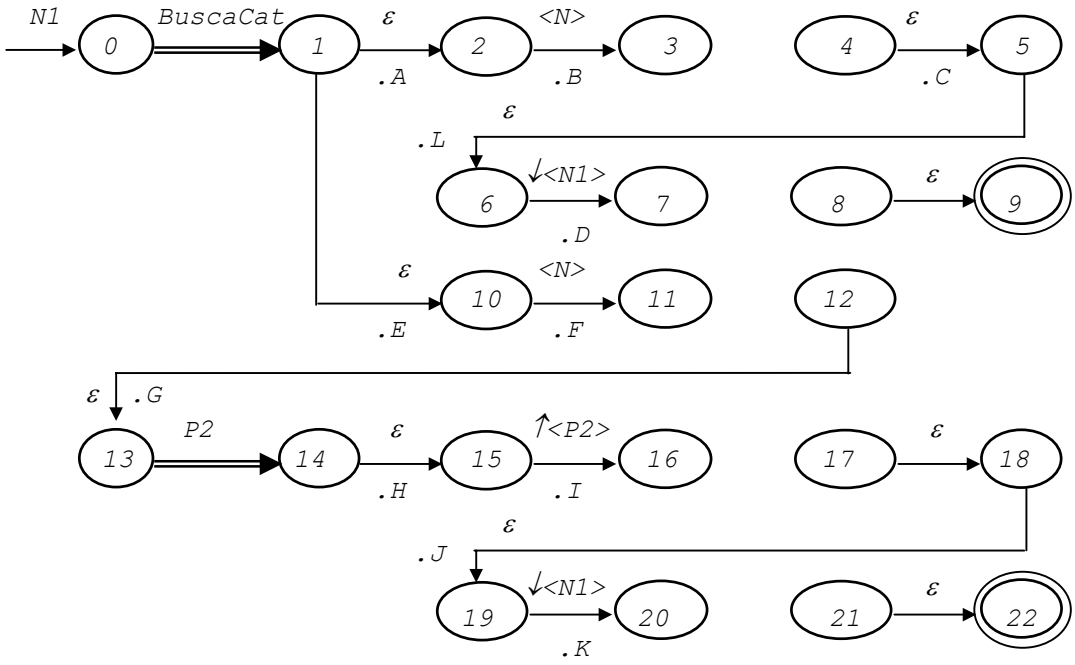


Figura 11 - Autômato Adaptativo correspondente ao mapeamento das regras ID 1a e 1b.

A ação adaptativa A elimina a transição entre os estados 3 e 4.

A ação adaptativa B deve testar se as características de N[1.0] são válidas, ou seja, se o item léxico lido da cadeia de entrada tem as seguintes características e valores: N+, V-, BAR 0, SUBCAT 1.0. Esse teste pode ser processado, conferindo-se as transições e estados do fragmento de autômato correspondente ao item léxico lido. Se o resultado de execução do teste for verdadeiro, cria a transição entre os estados 3 e 4.

A ação adaptativa C cria um estado correspondente a N[1.0] e uma transição desse estado para o estado correspondente ao item léxico reconhecido. Se houver alguma característica não-nuclear instanciada nessa categoria, cria uma transição a partir do estado correspondente a essa categoria para o estado correspondente a essa característica.

A ação adaptativa D'' elimina a transição entre os estados 7 e 8.

A ação adaptativa D cria um estado correspondente à categoria N1 e uma transição entre esse estado e o estado correspondente a N[1.0]. Além disso, se a regra satisfizer aos princípios CAP, FFP e HFC, cria a transição entre os estados 7 e 8.

A ação adaptativa E elimina a transição entre os estados 11 e 12.

A ação adaptativa F deve testar se as características de N[1.1] são válidas, ou seja, se o item léxico lido da cadeia de entrada tem as seguintes características e valores: N+, V-, BAR 0, SUBCAT 1.1. Se o resultado de execução do teste for verdadeiro, cria a transição entre os estados 11 e 12.

A ação adaptativa G cria um estado correspondente a N[1.1] e uma transição desse estado para o estado correspondente ao item léxico reconhecido. Se houver alguma característica não-nuclear instanciada nessa categoria, cria uma transição a partir do estado correspondente a essa categoria para o estado correspondente a essa característica.

A ação adaptativa H elimina a transição entre os estados 16 e 17.

A ação adaptativa I deve testar se as características de P2[de] são válidas, ou seja, se o item reconhecido na cadeia de entrada tem as seguintes características e valores: N-, V-, BAR 2, PFORM de. Se o resultado de execução do teste for verdadeiro, cria a transição entre os estados 16 e 17.

A ação adaptativa K'' elimina a transição entre os estados 20 e 21.

A ação adaptativa K cria um estado correspondente à categoria N1, uma transição entre esse estado e o estado correspondente a N[1.1], e uma transição entre esse estado e o correspondente a P2[de]. Além disso, se a regra satisfizer aos princípios CAP, FFP e HFC, cria a transição entre os estados 20 e 21.

Sejam também as seguintes regras ID:

(1) c. N2 → H1
d. N2 → [SUBCAT D], H2
e. P1 → H[3.0], N2
f. P2 → H1

Submetendo-se essas regras ao algoritmo de mapeamento de regra ID para Autômato Adaptativo (MapeamentoRegraID-AA), tem-se (a simulação e a descrição completa das ações adaptativas encontram-se em [Taniwaki-01]):

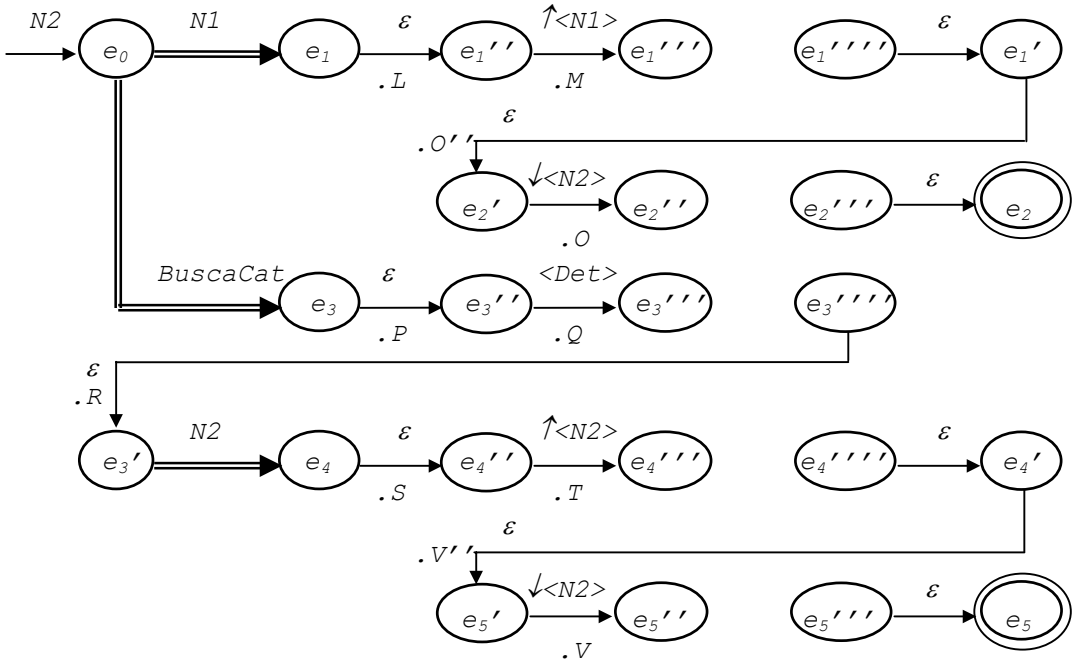


Figura 12 - Autômato adaptativo correspondente ao mapeamento das Regras ID 1c e 1d.

Ação adaptativa L elimina a transição de e_1''' para e_1''''.

Ação adaptativa M testa se as características de N1 são válidas. Se o resultado do teste for verdadeiro, cria a transição entre os estados e_1''' para e_1''''.

Ação adaptativa O'' elimina a transição de e_2'' para e_2'''.

Ação adaptativa O cria um estado correspondente à categoria N2, uma transição entre esse estado e o estado correspondente a N1. Se a regra satisfizer aos princípios CAP, FFP e HFC, cria a transição entre os estados e_2'' para e_2'''.

Ação adaptativa P elimina a transição de e_3''' para e_3''''.

Ação adaptativa Q testa se as características de [SUBCAT D] são válidas e se forem, cria a transição entre os estados e_3''' para e_3''''.

Ação adaptativa R cria um estado correspondente a [SUBCAT D] e uma transição desse estado para o estado correspondente ao item léxico reconhecido. Verifica também se há alguma característica não-nuclear instanciada nessa categoria. Se houver, cria uma transição a partir do estado correspondente a essa categoria para o estado correspondente a essa característica.

Ação adaptativa S elimina a transição de e_4''' para e_4''''.

Ação adaptativa T testa se as características de N2 são válidas e se forem, cria a transição entre os estados e_4''' para e_4''''.

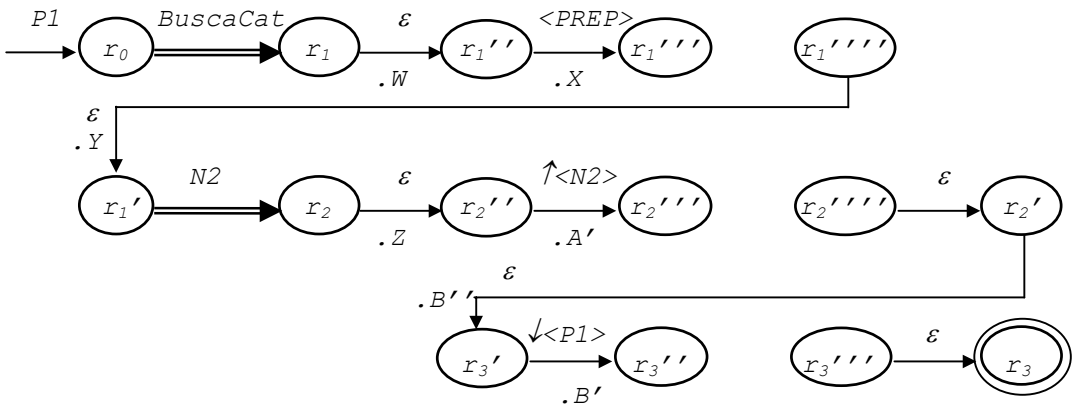


Figura 13 - Autômato adaptativo correspondente ao mapeamento da Regra ID 1e.

Ação adaptativa V′′ elimina a transição de e_5'' para e_5'''.

Ação adaptativa V cria um estado correspondente à categoria N2, uma transição entre esse estado e o estado correspondente a [SUBCAT D], e uma transição entre esse estado e o correspondente a N2. Se a regra satisfizer aos princípios CAP, FFP e HFC, cria a transição entre os estados e_5'' para e_5'''.

Ação adaptativa W elimina a transição de r_1''' para r_1''''.

Ação adaptativa X testa se as características de H[3.0] são válidas, e se forem, cria a transição entre os estados r_1''' para r_1''''.

Ação adaptativa Y cria um estado correspondente a H[3.0] e uma transição desse estado para o estado correspondente ao item léxico reconhecido. Verifica também se há alguma característica não-nuclear instanciada nessa categoria. Se houver, cria uma transição a partir do estado correspondente a essa categoria para o estado correspondente a essa característica.

Ação adaptativa Z elimina a transição de r_2''' para r_2''''.

Ação adaptativa A′ testa se as características de N2 são válidas e se forem, cria a transição entre os estados r_2'' para r_2''''.

Ação adaptativa B′′ elimina a transição de r_3'' para r_3'''.

Ação adaptativa B′ cria um estado correspondente à categoria P1, uma transição entre esse estado e o estado correspondente a H[3.0], e uma transição entre esse estado e o correspondente a N2. Se a regra satisfizer aos princípios CAP, FFP e HFC, cria a transição entre os estados r_3'' para r_3'''.

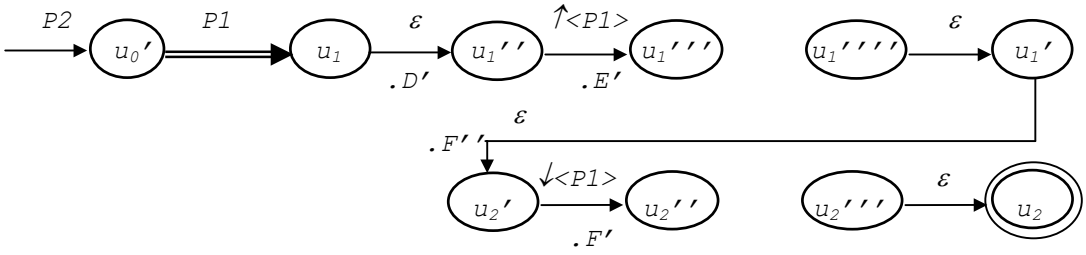


Figura 14 - Autômato adaptativo correspondente ao mapeamento da Regra ID 1f.

Ação adaptativa D′ elimina a transição de u_1''' para u_1''''.

Ação adaptativa E′ testa se as características de H1 são válidas, e se forem, cria a transição entre os estados u_1''' e u_1''''.

Ação adaptativa F′′ elimina a transição de u_2'' para u_2'''.

Ação adaptativa F′ cria um estado correspondente à categoria P2, uma transição entre esse estado e o estado correspondente a H1. Se a regra satisfizer aos princípios CAP, FFP e HFC, cria a transição entre os estados u_2' e u_2'''.

Como nos experimentos referentes ao mapeamento de redes ATN para Autômato Adaptativo, submetendo-se o sintagma substantivo ‘a destruição da cidade’ ao analisador e etiquetador morfológico de [Menezes-00], obtém-se:

```
a /D-F destruição /N da /P+D-F cidade/N
```

Submetendo-se a saída do analisador e etiquetador morfológico referente ao sintagma substantivo ‘a destruição da cidade’ à rotina MapeamentoLexico, e submetendo-se, em seguida, a saída da rotina MapeamentoLéxico referente ao sintagma substantivo ‘a destruição da cidade’ aos autômatos adaptativos gerados pelo mapeamento das regras ID 1a a 1f, obtém-se a estrutura ilustrada na Figura 15, correspondente à análise sintática do sintagma.

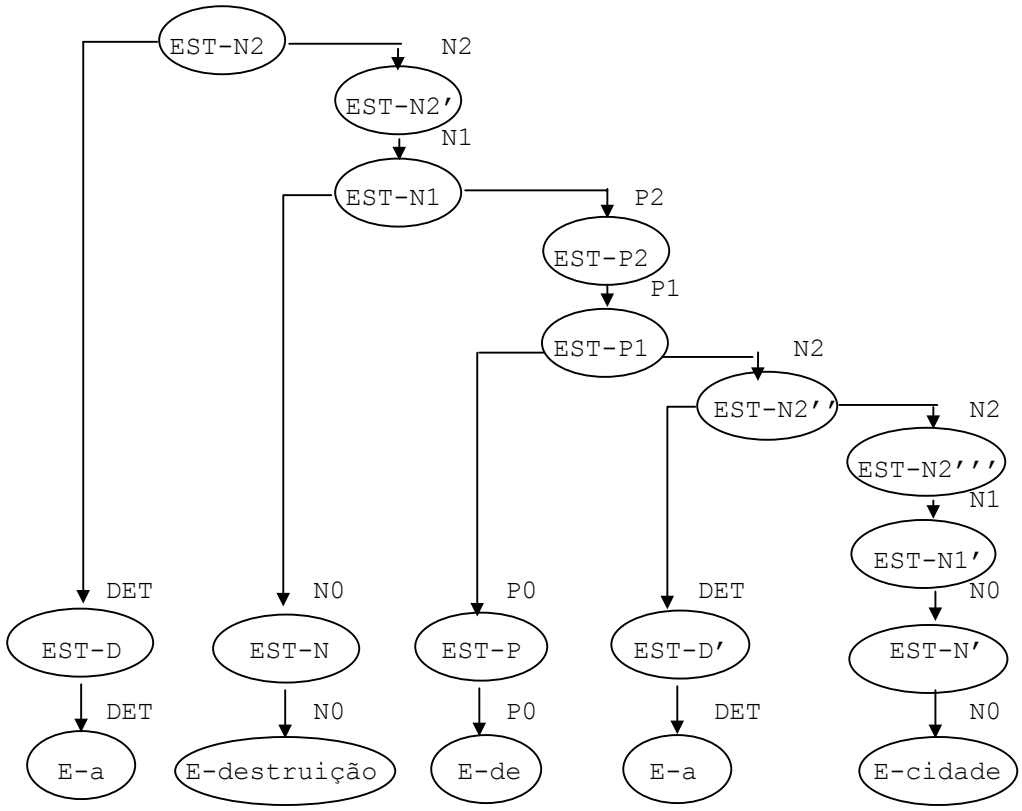


Figura 15 - Estrutura obtida como resultado do reconhecimento do sintagma substantivo 'a destruição da cidade', pelos autômatos adaptativos correspondentes às Regras ID 1a a 1f.

Neste exemplo, através da aplicação do princípio HFC, os estados EST-N1' e EST-N2''' recebem os valores [PER3, PLU- , MASC-] , referentes à palavra 'cidade'. Aplicando-se o princípio CAP, verifica-se a concordância entre os estados EST-D' e EST-N2''' (ambos apresentam-se no singular e no gênero feminino).

Pela aplicação do princípio HFC, os estados EST-P1 e EST-P2 recebem o valor [PFORM de], o que é essencial para que se possa aplicar a regra 1b (`N1 → H[1.1], P2[de]`).

Também pela aplicação do princípio HFC, os estados EST-N1 e EST-N2' recebem os valores [PER3, PLU-, MASC-], referentes à palavra 'destruição'. Aplicando-se o princípio CAP, verifica-se a concordância entre os estados EST-D e EST-N2' (ambos apresentam-se no singular e no gênero feminino).

Constata-se, assim, que os autômatos adaptativos resultantes do mapeamento das Regras GPSG 1a a 1f são capazes de reconhecer e analisar sintaticamente um sintagma substantivo da língua portuguesa, criando uma estrutura sintática em decorrência dessa análise.

Dessa forma, pode-se comprovar a funcionalidade dos algoritmos de mapeamento de GPSG para Autômato Adaptativo.

Da mesma forma como se mapeou as regras 1a a 1f, pode-se mapear as demais regras da especificação da gramática da língua portuguesa em GPSG, definidas em [Chin-96]. O mapeamento completo de todas as regras resultará numa especificação da gramática superficial da língua portuguesa em Autômato Adaptativo e num analisador sintático da língua portuguesa implementado através de Autômato Adaptativo.

5. Avaliação

Pelos experimentos realizados, pode-se concluir que os Autômatos Adaptativos podem ser empregados no processamento de linguagem natural, especificamente no que se refere à representação de informações lingüísticas e à análise sintática.

Os algoritmos propostos mostraram-se funcionais, e os métodos utilizados possibilitaram demonstrar a viabilidade de utilização prática dos Autômatos Adaptativos na análise sintática da linguagem natural.

O mapeamento do GPSG para Autômato Adaptativo foi bem menos direto do que o do ATN, resultando em algoritmos extensos, bem como em ações adaptativas extensas.

Embora não se tenha registrado, neste trabalho, experimentos mais significativos, que possibilitariam uma melhor avaliação dos resultados, pode-se dizer, de maneira geral, que o objetivo proposto foi atingido.

5.1. Contribuições

A contribuição deste trabalho é apresentar uma utilização prática do Autômato Adaptativo, à medida que se verifica a sua capacidade de uso como ferramenta para a representação de informações e análise sintática no processamento de linguagem natural.

Através do desenvolvimento deste trabalho, são apresentadas formas de mapeamento, para Autômato Adaptativo, de duas notações clássicas de representação de linguagem natural: as redes *Augmented Transition Network* e as especificações GPSG (*Generalized Phrase Structure Grammar*).

Uma contribuição importante decorrente desses experimentos é a utilização da especificação da gramática da língua portuguesa em GPSG, desenvolvida em [Chin-96], o que representa a evolução de um trabalho lingüístico e o aproveitamento de esforços já despendidos, possibilitando a geração de futuros trabalhos sobre o assunto, a respeito do qual pouco existe disponível na literatura.

Através do desenvolvimento do mapeamento, também foi possível constatar que o Autômato Adaptativo é capaz de resolver diversos problemas de linguagens complexas, tais como especificações incompletas e dependências de contexto, utilizando apenas recursos sintáticos.

5.2. Trabalhos Futuros

Há a necessidade de ensaios, envolvendo uma implementação completa dos algoritmos propostos neste trabalho, bem como da especificação da gramática da língua portuguesa, para que se possa avaliar melhor o alcance das técnicas apresentadas. Disso resultaria um analisador sintático para a língua portuguesa, implementado através de Autômato Adaptativo.

Para dar continuidade a esse trabalho, o que pode ser realizado futuramente é a especificação da representação da linguagem e de um analisador sintático através do Autômato Adaptativo, sem que haja mapeamentos intermediários de outros formalismos, sendo possível, assim, melhor aproveitar os recursos de adaptabilidade dos Autômatos Adaptativos.

6. Referências

BATES, M. The Theory and Practice of Augmented Transition Network Grammars. **Natural language communication with computer.** Berlim, 1978. Lecture Notes in Computer Science, 63, p. 191-259.

BATES, M.; BOBROW, R.J.; WEISCHEDEL, R.M. Critical challenges for natural language processing, p. 3-34. In BATES, M.; WEISCHEDEL, R.M. (Eds.) **Challenges in Natural Language Processing (Studies in Natural Language Processing).** Cambridge University Press, 1993.

CHIN, E. **Tradução por computador: dicionário e componentes de análise e transferência.** 1996. 330p. Tese (Doutorado) - Departamento de Lingüística da Faculdade de Filosofia, Letras e Ciências Humanas da Universidade de São Paulo. São Paulo.

GAZDAR G.; KLEIN E.; PULLUM G.; SAG I. **Generalized Phrase Structure Grammar.** Cambridge: Harvard University Press, 1985.

IWAI, M.K. **Um formalismo gramatical adaptativo para linguagens dependentes de contexto.** 2000. 191p. Tese (Doutorado) - Departamento de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo. São Paulo.

JOSÉ NETO, J. Adaptive automata for context-dependent languages. **ACM SIGPLAN Notices**, v.29, n.9, p.115-24, 1994.

JOSÉ NETO, J. Adaptive rule-driven devices – general formulation and case study. In: CONFERENCE OF IMPLEMENTATIONA AND APPLICATIONS OF AUTOMATA, Pretoria, 2001. **Proceedings of the 6th conference on implementationa and applications of automata.** p.158-176.

MENEZES, C.E.D. **Um método para a construção de analisadores morfológicos, aplicado à língua portuguesa, baseado em autômatos adaptativos.** 2000. 117p. Dissertação (Mestrado) – Departamento de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo. São Paulo.

RICH, E.; KNIGHT, K. **Inteligência Artificial,** 2. Ed. São Paulo: Makron Books, 1993.

SHIEBER, S.M. Separating Linguistic Analyses from Linguistic Theories. **Natural Language Parsing and Linguistic Theories**, p. 33-68, D. Reidel Publishing Company, 1988.

TANIWAKI, C.Y.O. **Formalismos adaptativos na análise sintática de linguagem natural.** 2001. 210p. Dissertação (Mestrado) - Departamento de Computação e Sistemas Digitais da Escola Politécnica da Universidade de São Paulo. São Paulo.

WOODS, W. A. Transition Network Grammars for Natural Language Analysis. **Communications of the ACM**, 13, n. 10, Out. 1970, p. 591-606.