O Reconhecedor Gramatical Linguístico: Avanços em Desambiguação Sintática e Semântica

A. T. Contier, D. Padovani e J. José Neto

***Resumo*— Este trabalho faz uma breve revisão dos conceitos de Tecnologia Adaptativa, apresentando seu mecanismo de funcionamento e seus principais campos de aplicação, destacando o forte potencial de sua utilização no processamento de linguagens naturais. Em seguida, são apresentados os conceitos de processamento de linguagem natural, ressaltando seu intricado comportamento estrutural. Por fim, é apresentado o Linguístico, um reconhecedor gramatical que utiliza autômatos e gramáticas adaptativas como tecnologia subjacente. O presente artigo é uma continuação de trabalhos anteriores em que foi incluída uma nova sessão para apresentar os avanços em desambiguação sintática e semântica.**

***Palavras Chave* — Autômatos Adaptativos, Processamento de Linguagem Natural, Reconhecedores Gramaticais, Gramáticas Livres de Contexto, Gramáticas Adaptativas.**

I. Autômatos Adaptativos

O AUTÔMATO adaptativo é uma máquina de estados à qual são impostas sucessivas alterações resultantes da aplicação de ações adaptativas associadas às regras de transições executadas pelo autômato [1]. Dessa maneira, estados e transições podem ser eliminados ou incorporados ao autômato em decorrência de cada um dos passos executados durante a análise da entrada. De maneira geral, pode-se dizer que o autômato adaptativo é formado por um dispositivo convencional, não adaptativo, e um conjunto de mecanismos adaptativos responsáveis pela auto modificação do sistema.

O dispositivo convencional pode ser uma gramática, um autômato, ou qualquer outro dispositivo que respeite um conjunto finito de regras estáticas. Este dispositivo possui uma coleção de regras, usualmente na forma de cláusulas if-then, que testam a situação corrente em relação a uma configuração específica e levam o dispositivo à sua próxima situação. Se nenhuma regra é aplicável, uma condição de erro é reportada e a operação do dispositivo, descontinuada. Se houver uma única regra aplicável à situação corrente, a próxima situação do dispositivo é determinada pela regra em questão. Se houver mais de uma regra aderente à situação corrente do dispositivo, as diversas possíveis situações seguintes são tratadas em paralelo e o dispositivo exibirá uma operação não determinística. Os mecanismos adaptativos são formados por três tipos de ações adaptativas elementares: consulta (inspeção do conjunto de regras que define o dispositivo), exclusão (remoção de alguma regra) e inclusão (adição de uma nova regra).

Autômatos adaptativos apresentam forte potencial de aplicação ao processamento de linguagens naturais, devido à facilidade com que permitem representar fenômenos linguísticos complexos tais como dependências de contexto. Adicionalmente, podem ser implementados como um formalismo de reconhecimento, o que permite seu uso no pré-processamento de textos para diversos usos, tais como: análise sintática, verificação de sintaxe, processamento para traduções automáticas, interpretação de texto, corretores gramaticais e base para construção de sistemas de busca semântica e de aprendizado de línguas auxiliados por computador.

Diversos trabalhos confirmam a viabilidade prática da utilização de autômatos adaptativos para processamento da linguagem natural. É o caso, por exemplo, de [2], que mostra a utilização de autômatos adaptativos na fase de análise sintática; [3] que apresenta um método de construção de um analisador morfológico e [4], que apresenta uma proposta de autômato adaptativo para reconhecimento de anáforas pronominais segundo algoritmo de Mitkov.

II. Processamento da Linguagem Natural: Revisão da Literatura

O processamento da linguagem natural requer o desenvolvimento de programas que sejam capazes de determinar e interpretar a estrutura das sentenças em muitos níveis de detalhe. As linguagens naturais exibem um intricado comportamento estrutural visto que são profusos os casos particulares a serem considerados. Uma vez que as linguagens naturais nunca são formalmente projetadas, suas regras sintáticas não são simples nem tampouco óbvias e tornam, portanto, complexo o seu processamento computacional. Muitos métodos são empregados em sistemas de processamento de linguagem natural, adotando diferentes paradigmas, tais como métodos exatos, aproximados, pré-definidos ou interativos, inteligentes ou algorítmicos [5]. Independentemente do método utilizado, o processamento da linguagem natural envolve as operações de análise léxico-morfológica, análise sintática, análise semântica e análise pragmática [6]. A análise léxico-morfológica procura atribuir uma classificação morfológica a cada palavra da sentença, a partir das informações armazenadas no léxico [7]. O léxico ou dicionário é a estrutura de dados contendo os itens lexicais e as informações correspondentes a estes itens. Entre as informações associadas aos itens lexicais, encontram-se a categoria gramatical do item, tais como substantivo, verbo e adjetivo, e os valores morfossintático-semânticos, tais como gênero, número, grau, pessoa, tempo, modo, regência verbal ou nominal. Na etapa de análise sintática, o analisador verifica se uma sequência de palavras constitui uma frase válida da língua, reconhecendo-a ou não. O analisador sintático faz uso de um léxico e de uma gramática, que define as regras de combinação dos itens na formação das frases. Nos casos nos quais há a necessidade de interpretar o significado de um texto, a análise léxico-morfológica e a análise sintática não são

suficientes, sendo necessário realizar um novo tipo de operação, denominada análise semântica [7]. Na análise semântica procura-se mapear a estrutura sintática para o domínio da aplicação, fazendo com que a estrutura ganhe um significado. O mapeamento é feito identificando as propriedades semânticas do léxico e o relacionamento semântico entre os itens que o compõe [8]. Já a análise pragmática procura reinterpretar a estrutura que representa o que foi dito para determinar o que realmente se quis dizer. Inserem-se nessa categoria as relações anafóricas, correferências, determinações, focos ou temas, dêiticos e elipses [9].

Em [10] são apresentados trabalhos de pesquisas em processamento de linguagem natural para a Língua Portuguesa tais como o desenvolvido pelo Núcleo Interinstitucional de Linguística Aplicada (NILC) no desenvolvimento de ferramentas para processamento de linguagem natural; o projeto VISL – Visual Interactive Syntax Learning, sediado na Universidade do Sul da Dinamarca, que engloba o desenvolvimento de analisadores morfossintáticos para diversas línguas, entre as quais o português; e o trabalho de resolução de anáforas desenvolvido pela Universidade de Santa Catarina. A tecnologia adaptativa também tem contribuído com trabalhos em processamento da linguagem natural. Em [11], são apresentadas algumas das pesquisas desenvolvidas pelo Laboratório de Linguagens e Tecnologia Adaptativa da Escola Politécnica da Universidade de São Paulo: um etiquetador morfológico, um estudo sobre processos de análise sintática, modelos para tratamento de não determinismos e ambiguidades, e um tradutor texto voz baseado em autômatos adaptativos.

III. RECONHECEDOR ADAPTATIVO: SUPORTE TEÓRICO LINGUÍSTICO

A Moderna Gramática Brasileira de Celso Luft [12] foi escolhida como suporte teórico linguístico do reconhecedor aqui proposto. A escolha foi feita em função da forma clara e precisa com que Luft categoriza os diversos tipos de sentenças de língua portuguesa, diferenciando-se das demais gramáticas que priorizam a descrição da língua em detrimento da análise estrutural da mesma. Luft diz que a oração é moldada por padrões frasais ou oracionais, compostos por elementos denominados sintagmas (Tabela 1).

TABELA 1.
ELEMENTOS FORMADORES DE SINTAGMAS

Sintagmas	
Substantivo	Quantitativos+Pronomes Adjetivos+ Sintagma Adjetivo1+Substantivo+ Sintagma Adjetivo2+ Sintagma Preposicional+ Oração Adjetiva
Verbal	Pré-verbais+ Verbo Auxiliar+ Verbo Principal

TABELA 1.
CONTINUAÇÃO

Sintagmas	
Adjetivo	Advérbio de Intensidade+ Adjetivo+ Sintagma Preposicional
Adverbial	Advérbio de Intensidade+ Adverbio+ Sintagma Preposicional
Preposicional	Preposição+ Sintagma Substantivo

Sintagma é qualquer constituinte imediato da oração, podendo exercer papel de sujeito, complemento (objeto direto e indireto), predicativo e adjunto adverbial. É composto por uma ou mais palavras, sendo que uma é classificada como núcleo e as demais como dependentes. As palavras dependentes podem estar localizadas à esquerda ou à direita do núcleo. Luft utiliza os seguintes nomes e abreviaturas:

1. Sintagma substantivo (SS): núcleo é um substantivo;
2. Sintagma verbal (SV): núcleo é um verbo;
3. Sintagma adjetivo (Sadj): núcleo é um adjetivo;
4. Sintagma adverbial (Sadv): núcleo é um advérbio;
5. Sintagma preposicional (SP): é formado por uma preposição (Prep) mais um SS.
6. Vlig: verbo de ligação
7. Vi: verbo intransitivo
8. Vtd: verbo transitivo direto
9. Vti: verbo transitivo indireto
10. Vtdi: verbo transitivo direto e indireto
11. Vt-pred: verbo transitivo predicativo

Um padrão oracional é determinado pelos tipos de sintagmas e pela sequência em que aparecem. Por exemplo, o padrão oracional SS Vlig SS, indica que a frase é composta por um sintagma substantivo, seguido de um verbo de ligação e de outro sintagma substantivo. Os padrões são classificados em cinco tipos (Tabela 2):

1. Padrões pessoais nominais: Neste caso, existe sujeito e o núcleo do predicado é um nome (substantivo, adjetivo, advérbio) ou um pronome (substantivo, adjetivo, advérbio). O verbo, nesses casos, é chamado de verbo de ligação (Vlig).
2. Padrões pessoais verbais: São aqueles nos quais existe o sujeito e o núcleo do predicado é um verbo. O verbo pode ser transitivo direto (Vtd), transitivo indireto (Vti), transitivo direto e indireto (Vtdi), e intransitivo (Vi). Se o verbo for transitivo direto (Vtd), o complemento será um objeto direto; se o verbo for transitivo indireto (Vti), o complemento será um objeto indireto; se o verbo for transitivo direto e indireto (Vtdi), o complemento será um objeto direto e um indireto; se o verbo for intransitivo (Vi), não há complemento.
3. Padrões Pessoais Verbo-Nominais: Neste caso, existe o sujeito e o núcleo do predicado é um verbo transitivo predicativo (Vt-pred), cujo complemento é um objeto direto e um predicativo do objeto.

4. Padrões Impessoais Nominais: Ocorrem quando não existe sujeito e o núcleo do predicado é um nome (substantivo, adjetivo, advérbio) ou um pronome (substantivo, adjetivo, advérbio).
5. Padrões Impessoais Verbais: Neste caso, não existe sujeito e o núcleo do predicado é um verbo.

TABELA 2
PADRÕES ORACIONAIS DE LUFT

Padrões Pessoais Nominais				
SS	Vlig	SS		
SS	Vlig	Sadj		
SS	Vlig	Sadv		
SS	Vlig	SP		
Padrões Pessoais Verbais				
SS	Vtd	SS		
SS	Vti	SP		
SS	Vti	Sadv		
SS	Vti	SP	SP	
SS	Vtdi	SS	SP	
SS	Vtdi	SS	Sadv	
SS	Vtdi	SS	SP	SP
SS	Vi			
Padrões Pessoais Verbo-Nominais				
SS	Vtpred	SS	SS	
SS	Vtpred	SS	Sadj	
SS	Vtpred	SS	SP	
SS	Vtpred	SS	Sadv	
SS	Vtpred	SS		
SS	Vtpred	Sadj		
SS	Vtpred	SP		
Padrões Impessoais Nominais				
	Vlig	SS		
	Vlig	Sadj		
	Vlig	Sadv		
	Vlig	SP		
Padrões Impessoais Verbais				
	Vtd	SS		
	Vti	SP		
	Vi			

Os sintagmas e os padrões oracionais de Luft foram mapeados em uma gramática para que pudessem ser processados computacionalmente, ficando da seguinte forma:

F → [Conec] [SS] SV [Conec]
Conec → F
SS → [Sadj] SS [Sadj | SP]
SS → [Quant | PrA] (Sc | Sp | PrPes)
SV→ [Neg] [Aux | PreV] (Vlig | Vtd| Vti | Vtdi| Vi)
[SS | Sadj | Sadv | SP] [SS| Sadj| Sadv| SP] [SP]
SP → Prep (SS | Sadj)
Sadj → Sadj [SP]
Sadj →[Adv] Adj
Sadv → Sadv [SP]
Sadv → [Adv] Adv
PrA → Ind | ArtDef | ArtInd| Dem| Pos

Sendo:
F – Frase
SS – Sintagma substantivo
SV – Sintagma verbal
SP – Sintagma preposicional
SN – Sintagma nominal
Sadv – Sintagma adverbial
Sadj – Sintagma adjetivo
Adv – adverbio
Adj – adjetivo
ArtDef – artigo definido
ArtInd – artigo indefinido
Aux – Partícula auxiliar (apassivadora ou pré-verbal)
Conec – Conector (conjunção ou pronome relativo)
Dem – pronome demonstrativo indefinido
Ind – pronome indefinido
Neg – partícula (negação)
PrA – pronome adjetivo
PrPes – pronome pessoal
Prep – preposição
Quant – numeral
Sc – substantivo comum
Sp – substantivo próprio
V– verbo
Vlig – verbo de ligação
Vi – verbo intransitivo
Vtd – verbo transitivo direto
Vti – verbo transitivo indireto
Vtdi – verbo transitivo direto e indireto

IV. PROPOSTA DE UM RECONHECEDOR GRAMATICAL

O Linguístico é uma proposta de reconhecedor gramatical composto de cinco módulos sequenciais que realizam cada qual um processamento especializado, enviando o resultado obtido para o módulo seguinte, tal como ocorre em uma linha de produção, até que o texto esteja completamente analisado (Fig.1).

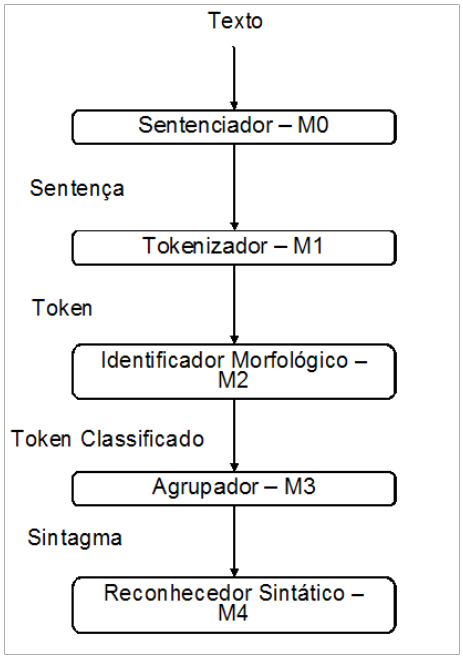

Figura 1. Estrutura do Linguístico.

O primeiro módulo, denominado Sentenciador, recebe um texto e realiza um pré-processamento, identificando os caracteres que possam indicar final de sentença, palavras abreviadas e palavras compostas, e eliminando aspas simples e duplas. Ao final, o Sentenciador divide o texto em supostas sentenças, para análise individual nas etapas seguintes.

O segundo módulo, denominado Tokenizador, recebe as sentenças identificadas na etapa anterior e as divide em *tokens*, considerando, neste processo, abreviaturas, valores monetários, horas e minutos, numerais arábicos e romanos, palavras compostas, nomes próprios, caracteres especiais e de pontuação final. Os *tokens* são armazenados em estruturas de dados (*arrays*) e enviados um a um para análise do módulo seguinte.

O terceiro módulo, chamado Identificador Morfológico, foi concebido para utilizar tecnologias adaptativas (Fig.2.1). O Identificador Morfológico é composto por um Autômato Mestre e um conjunto de submáquinas especialistas que acessam bases de dados com regras de acesso às bases de dados de classificações morfológicas. A prioridade é obter as classificações do corpus Bosque [13]; caso o termo procurado não seja encontrado, o Identificador Morfológico procura por substantivos, adjetivos e verbos, no formato finito e infinito (flexionados e não flexionados), através de uma submáquina de formação e identificação de palavras, que usa, como base, o vocabulário do TeP2.0 [14]; por fim, o Identificador Morfológico procura por termos invariáveis, ou seja, termos cuja classificação morfológica é considerada estável pelos linguistas, tais como, conjunções, preposições e pronomes, no caso, extraídos no léxico do Portal São Francisco [15].

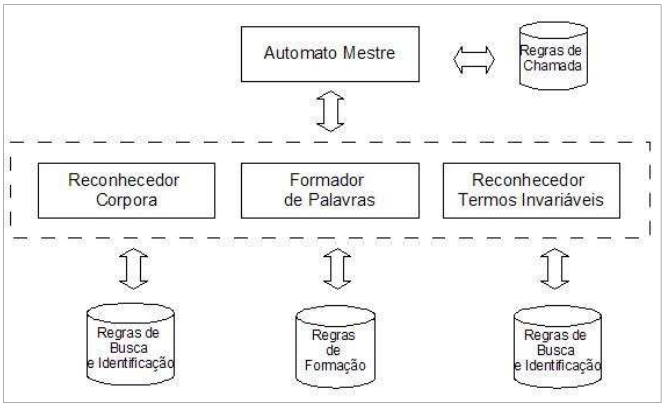

Figura 2.1. Arquitetura do Identificador Morfológico.

O Autômato Mestre (Fig.2.2) é responsável pelo sequenciamento das chamadas às submáquinas, de acordo com um conjunto de regras cadastradas em base de dados. Ele inicia o processamento recebendo o token da coleção de tokens criada pela classe Texto. Em seguida, antes de processá-lo, o autômato se modifica através de uma função adaptativa, criando uma submáquina de processamento (M1), uma transição entre o estado 1 e M1, e uma transição que aguarda o estado final da submáquina M1. O token é passado para M1 e armazenado em uma pilha. Quando M1 chega ao estado final, o autômato se modifica novamente, criando uma nova submáquina M2, o estado 2 e as transições correspondentes. Caso o estado final de M1 seja de aceitação, o processo é finalizado no estado 2, caso contrário a máquina M2 é chamada, passando o token armazenado na pilha.

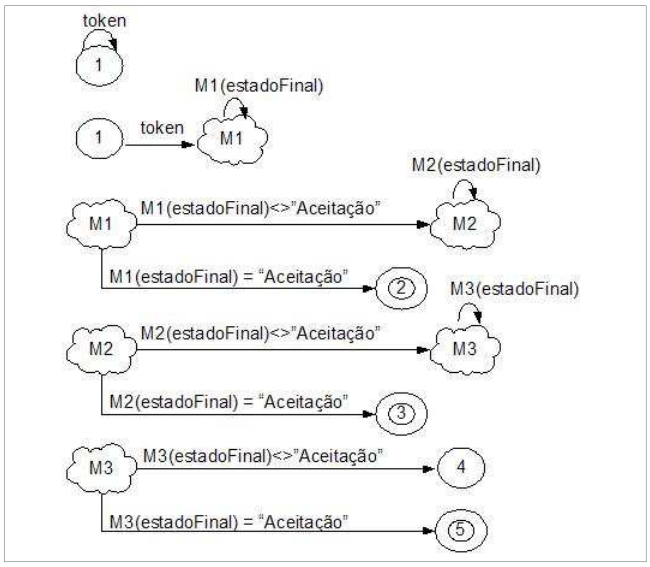

Figura 2.2. Estrutura Adaptativa do Autômato Mestre.

O processo se repete quando M2 chega ao final do processamento, com a criação da submáquina M3, do estado 3 e das transições correspondentes. Um novo ciclo se repete, e se o estado final de M3 é de aceitação, o autômato transiciona para o estado 5, de aceitação, caso contrário, ele vai para o estado 4 de não aceitação. M1, M2 e M3 representam, respectivamente, as submáquinas do Reconhecedor de Corpus, do Formador de Palavras e do Reconhecedor de Termos Invariáveis. Portanto, caso o Autômato Mestre encontre a classificação morfológica ao final de M1, ele não chama M2; caso encontre em M2, não chama M3 e, caso também não encontre em M3, ele informa aos demais módulos do Linguístico que não há classificação morfológica para o termo analisado.

As submáquinas M1, M2 e M3 também foram projetadas de acordo com a tecnologia adaptativa. A Máquina M1 usa um autômato adaptativo que se auto modifica de acordo com o tipo de token que está sendo analisado: palavras simples, palavras compostas, números, valores e símbolos. A Fig. 2.3 apresenta a estrutura adaptativa de M1.

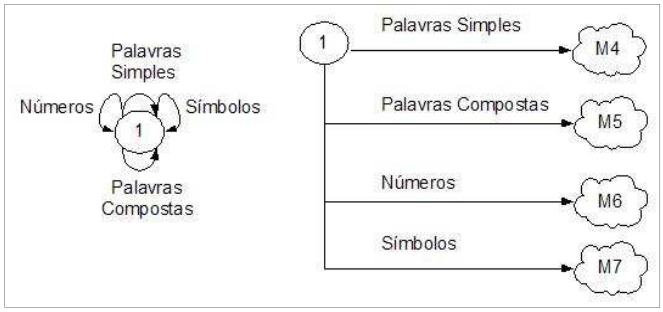

Figura 2.3. Estrutura Adaptativa do Reconhecedor de Corpus M1.

Inicialmente o autômato é composto por um único estado e por transições para ele mesmo (Fig.2.3, à esquerda). Ao identificar o tipo de token que será analisado (obtido no processo de tokenização), o autômato cria submáquinas e as transições correspondentes. As alternativas de configuração

são apresentadas na Fig.2.3, à direita. As submáquinas M4, M5, M6 e M7 reconhecem os tokens através de outro tipo de autômato que processa os tokens byte a byte (Fig. 2.4).

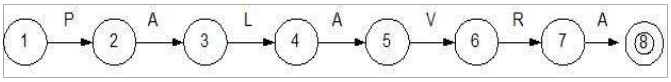

Figura 2.4. Reconhecedor de Palavras Simples.

No exemplo apresentado na Fig.2.4, o token “Palavra” é processado pela máquina M4; se o processamento terminar em um estado de aceitação, o token é reconhecido. As submáquinas M4, M5, M6 e M7 são criadas previamente por um programa que lê o Corpus e o converte em autômatos finitos determinísticos. A estrutura de identificação morfológica é composta por um par [chave, valor], no qual a chave é o estado de aceitação do elemento lexical e, o valor, sua classificação morfológica. No exemplo apresentado, a chave do item lexical “palavra” seria o estado ”8”, e, o valor, a classificação morfológica, H+n (substantivo, núcleo de sintagma nominal), proveniente do Corpus Bosque.

O Formador de Palavras, submáquina M2, também é montado previamente por um programa construtor, levando em consideração as regras de formação de palavras do Português do Brasil, descritas por Margarida Basílio em [16]. A submáquina M2 utiliza o vocabulário do TeP2.0 (formado por substantivos, adjetivos e verbos), como léxico de formas já feitas e o conjunto de regras de prefixação, sufixação e regressão verbal descrito pela autora para construir novas formas. No entanto, são necessários alguns cuidados para evitar a criação de estruturas que aceitem palavras inexistentes. A Fig. 2.5 apresenta um exemplo de autômato no qual são aplicados os prefixos “a” e “per”, e os sufixos “ecer” e “ar” (derivação parassintética) ao radical “noit” do substantivo noite, que faz parte do TeP2.0.

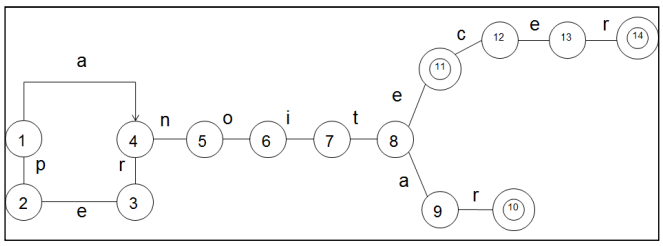

Figura 2.5. Autômato Formador de Palavras.

No exemplo apresentado, as palavras “anoitecer”, “pernoitar” e “anoitar” (sinônimo de “anoitecer”) existem no léxico do português do Brasil. Já pernoitecer é uma combinação que não existe na língua portuguesa. Margarida Basílio diz que algumas combinações não são aceitas simplesmente porque já existem outras construções consagradas pelo uso [17]. Para reduzir o risco de aceitar derivações inexistentes, o processo de construção do autômato restringe as possíveis formações, utilizando apenas as regras que a autora destaca como sendo mais prováveis. É o caso de nominalização de verbos com o uso dos sufixos –ção , –mento e – da. Em [16], Basílio cita que o sufixo –ção é responsável por 60% das formações regulares, enquanto o sufixo –mento é responsável por 20% destas formações. Já o sufixo –da é, via de regra, usado em nominalizações de verbos de movimento, tais como, entrada, saída, partida, vinda, etc.

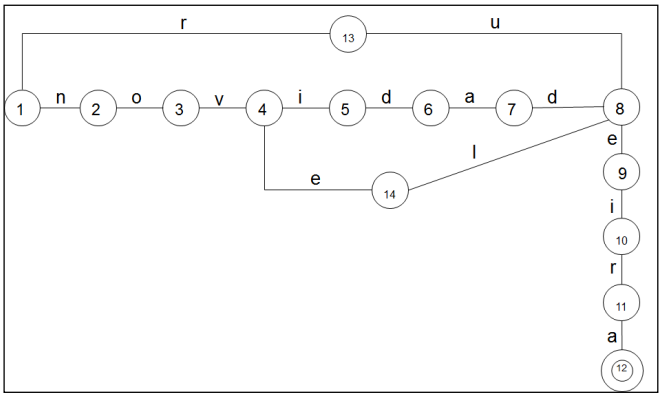

Figura 2.6. Autômato Formador de Palavras.

Outra característica do construtor do autômato é representar os prefixos e sufixos sempre pelo mesmo conjunto de estados e transições, evitando repetições que acarretariam o consumo desnecessário de recursos computacionais. A Fig. 2.6 apresenta exemplo de palavras formadas por reutilização de estados e transições na derivação das palavras rueira, novidadeira e noveleira. Os estados e transições usados para representar o sufixo “eira”, usado para designar são os mesmos nas três derivações.

A submáquina M2 também reconhece palavras flexionadas, obtidas, no caso de substantivos e adjetivos, através da aplicação de sufixos indicativos de gênero, número e grau aos radicais do vocabulário TeP2.0. A Fig. 2.7 apresenta um exemplo de autômato usado para a formação das formas flexionadas do substantivo menino. Foram adicionados ao radical “menin” as flexões “o(s)”, “a(s)”, “ão”, “ões”, “onona(s)”, “inho(s)” e“inha(s)”.

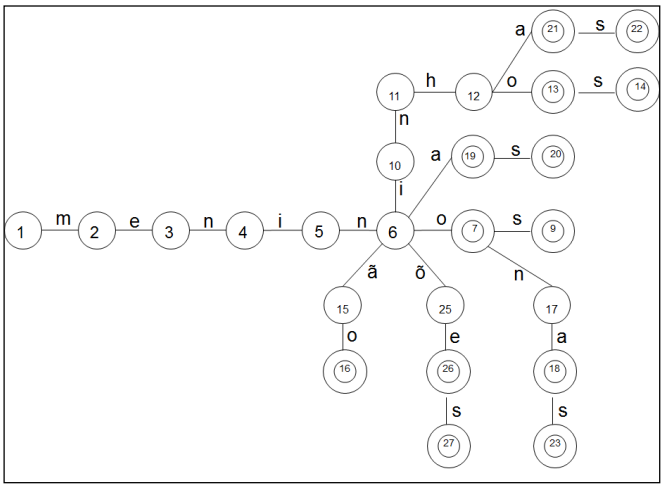

Figura 2.7. Autômato Formador de Flexões Nominais.

No caso de verbos, foi criada uma estrutura de estados e transições para representar as flexões de tempo, modo, voz e pessoa, obtendo-se, assim, as respectivas conjugações.

A Fig. 2.8 apresenta um exemplo de autômato usado para a formação das formas flexionadas do presente do indicativo do

verbo andar. Foram adicionados ao radical “and” as flexões “o”, “as”, “a”, “andamos”, “ais” e “andam”.

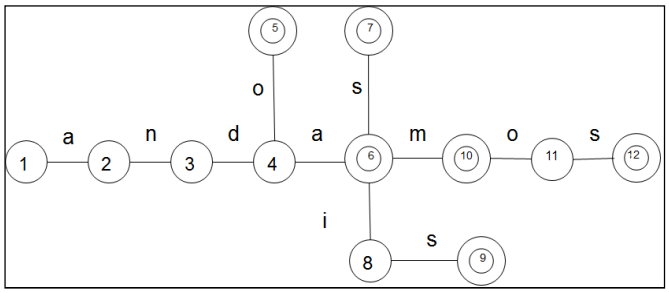

Figura 2.8. Autômato Formador de Flexões Verbais.

A estrutura de armazenamento da submáquina M2 também é composta por um par [chave, valor], no qual a chave é o estado de aceitação do elemento lexical e, o valor, sua classificação morfológica, acrescida da origem do termo, indicando que ele foi gerado pelo construtor. Por exemplo, o termo “novidadeira” seria classificado como H+n+C – substantivo, núcleo de sintagma nominal, gerado pelo construtor.

Já a submáquina M3 é um autômato que varia em função do tipo de termo (conjunções, preposições e pronomes) e utiliza uma estrutura arbórea similar a M4. A Fig. 2.9 apresenta um exemplo de autômato usado no reconhecimento de termos deste domínio.

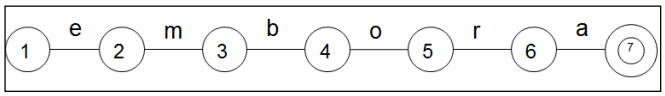

Figura 2.9. Autômato Reconhecedor de Conjunções, Preposições e Pronomes.

A estrutura de armazenamento da submáquina M3 é composta por um par [chave, valor], no qual a chave é o estado de aceitação do elemento lexical e, o valor, sua classificação morfológica, acrescida da origem do termo, indicando que ele faz parte da base de dados de apoio do Portal São Francisco. Por exemplo, o termo “embora” seria classificado como cj+Ba –conjunção da base de dados de apoio.

O Autômato Mestre também é responsável por desambiguar as classificações morfológicas e informar ao Agrupador apenas as que forem mais adequadas. Como as palavras podem ter mais do que uma classificação morfológica, é necessário utilizar uma técnica para selecionar aquela que é mais apropriada para o contexto analisado. Por exemplo, a palavra “casa” pode ser classificada como substantivo comum, feminino, singular ou como verbo flexionado na 3ª pessoa do singular no tempo presente, modo indicativo. No entanto, tendo em vista o contexto em que palavra se encontra, é possível escolher a classificação mais provável. Por exemplo, se a palavra “casa” vier precedida de um artigo definido, é mais provável que ela seja um substantivo; já se a palavra antecessora for um substantivo próprio, é mais provável que “casa” seja um verbo.

Collins [18] apresenta um modelo estatístico que leva em consideração 2 classificações anteriores à palavra analisada para fazer a desambiguação, criando distribuições nas quais a etiqueta de máxima probabilidade é o resultado da função:

P = max p(E,S), sendo:
E = etiquetas
S = palavras da sentença
p = probabilidade de E, dado S
max = máxima probabilidade

Por exemplo, para etiquetar a frase “O advogado entrevista a testemunha”, a função receberia as palavras da sentença e as sequências de etiquetas possíveis para elas (obtidas a partir de um Corpus de testes), calculando, então, a sequência de maior probabilidade. No entanto, Collins alerta que a complexidade para executar tal função inviabiliza sua utilização, pois ela cresce exponencialmente em função do número de palavras da sentença (Em uma sentença de “n” palavras e “K” possíveis classificações, existiriam $|K|^n$ possíveis etiquetas de sequencia).

Para evitar o problema da complexidade da execução, Collins propõe o uso do algoritmo Viterbi [19]. O algoritmo Viterbi é usado para encontrar a sequência mais provável de estados ocultos que resultam da observação de uma sequência de eventos. No caso da identificação das etiquetas morfológicas, os eventos são as palavras do texto analisado e os estados são as etiquetas de identificação morfológica. O algoritmo Viterbi pode ser implementado por meio de um autômato finito determinístico. No entanto, um autômato com estas características poderia ficar muito grande e, consequentemente, lento, devido ao tamanho do léxico usado para montá-lo. Uma alternativa para evitar este tipo de problema, é usar a tecnologia adaptativa. A Fig. 2.10 apresenta a estrutura do autômato que implementa o algoritmo Viterbi usado pelo desambiguador morfológico do Linguístico. O autômato recebe como parâmetro de entrada a palavra que está sendo analisada e monta dinamicamente os estados e transições de acordo com as possíveis etiquetas e as respectivas probabilidades, identificando, ao final do processamento, a etiqueta mais provável.

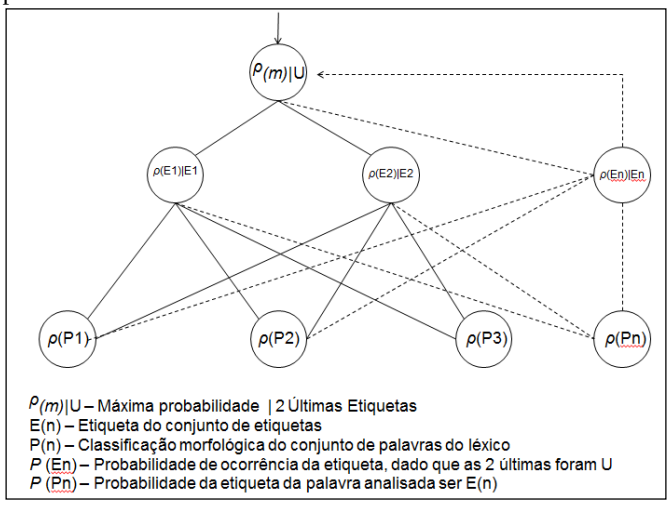

Figura 2.10. Autômato Adaptativo Viterbi.

O quarto módulo, denominado Agrupador é composto de um autômato, responsável pela montagem dos sintagmas a partir de símbolos terminais da gramática e um bigrama, responsável pela montagem dos sintagmas a partir de não-terminais (Fig.3). Inicialmente, o Agrupador recebe do Identificador as classificações morfológicas dos *tokens* e as agrupa em sintagmas de acordo com as regras de Luft. Neste processo são identificados sintagmas nominais, verbais, preposicionais, adjetivos e adverbiais Para isso, o Agrupador utiliza um

autômato adaptativo cuja configuração completa é definida da seguinte forma:
Estados = {1, 2, 3, 4, SS, SP, V, Sadj, Sadv, A},
Onde:
1,2,3 e 4 = Estados Intermediários
SS, SP, V Sadj, Sadv = Estados nos quais houve formação de sintagmas, sendo:
SS= Sintagma substantivo
SP = Sintagma preposicional
V = Verbo ou locução verbal
Sadj = Sintagma adjetivo
Sadv = Sintagma adverbial
A = Estado após o processamento de um ponto final
Tokens = {art, num, n, v, prp, pron, conj, adj, adv, rel, pFinal, sClass}, onde:
art = artigo, num = numeral
n = substantivo, v = verbo
prp = preposição, pron = pronome
conj = conjunção, adj = adjetivo
adv = advérbio, rel = pronome relativo
pFinal = ponto final, sClass = sem classificação
Estados de Aceitação = {SS, SP, V, Sadj, Sadv, A}
Estado Inicial = {1}
Função de Transição = {(Estado, *Token*)→Estado}, sendo:
{(1, art)→2, (2, art)→2, (3, art)→3
(1, num)→Sadv, (2, num)→2, (3, num)→3
(1, n)→SS, (2, n)→SS, (3, n)→SP
(1, v)→SV, (2, v)→ SV, (3, v)→SP
(1, prp)→3, (2, prp)→2, (3, prp)→3
(1, prop)→SS, (2, prop)→SS, (3, prop)→SP
(1, pron)→SS, (2, pron)→SS, (3, pron)→SP
(1, conj)→conj, (2, conj)→Ø, (3, conj)→ Ø
(1, adj)→Sadj, (2, adj)→Sadj, (3, adj)→3
(1, adv)→Sadv, (2, adv)→2, (3, adv)→3
(1, rel)→conj, (2, rel)→ Ø, (3, rel)→conj
(1, pFinal)→A, (2, pFinal)→ Ø, (3, pFinal)→ Ø}

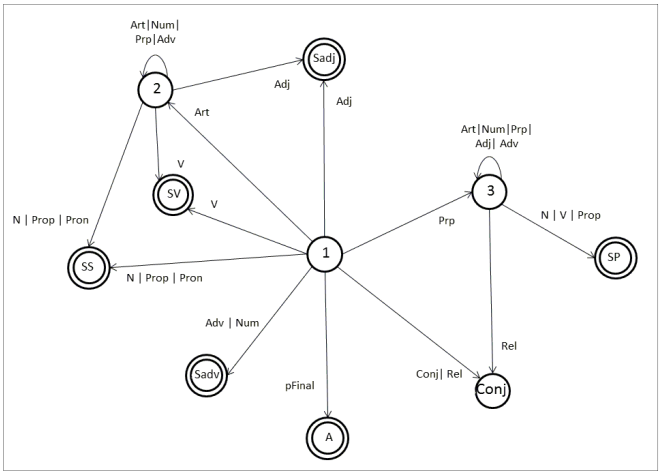

Figura 3. Configuração Completa do Autômato Construtor de Sintagmas.

Por exemplo, no caso de sintagmas substantivos teríamos:
SS → [Quant | PrA] (Sc | Sp | PrPes)
Pela regra acima, o conjunto de *tokens* "A" e "casa" formam um sintagma substantivo, da seguinte forma:
PrA = "A" (artigo definido)
Sc = "casa" (substantivo comum)
Da direita para esquerda, são realizadas as seguintes transições:
PrA Sc → SS; A Sc → SS; A casa → SS
Já o Agrupador recebe o *token* "A", identificado pelo Tokenizador como artigo definido, e se movimenta do estado 1 para o estado 2. Ao receber o *token* "casa", identificado como substantivo comum, ele se movimenta do estado 2 para o estado SS, que é um estado de aceitação. Neste momento o Agrupador armazena a cadeia "A casa" e o símbolo "SS" em uma pilha e reinicializa o autômato preparando-o para um novo reconhecimento. Em um passo seguinte, o Agrupador usa o bigrama para comparar um novo sintagma com o último sintagma formado, visando identificar elementos mais altos na hierarquia. Para isso ele usa a matriz apresentada na Tabela 3. A primeira coluna da matriz indica o último sintagma formado (US) e a primeira linha, o sintagma atual (SA). A célula resultante apresenta o novo nó na hierarquia.

TABELA 3.
MATRIZ DE AGRUPAMENTO DE SINTAGMAS

SA / US	SS	SP	V	Sadv	Sadj	Conj
SS	SS	SS	-	-	SS	-
SP	SP	-	-	-	-	-
V	-	-	V	-	-	-
Sadv	-	-	-	Sadv	Sadj	-
Sadj	SS	Sadj	-	-	Sadj	-
Conj	-	-	-	-	-	Conj

Esta técnica foi usada para tratar as regras gramaticais nas quais um sintagma é gerado a partir da combinação de outros, como é o caso da regra de formação de sintagmas substantivos: SS → [Sadj] SS [Sadj | SP]. Por esta regra, os sintagmas substantivos são formados por outros sintagmas substantivos precedidos de um sintagma adjetivo e seguidos de um sintagma adjetivo ou um sintagma preposicional. No exemplo anterior, supondo que os próximos 2 *tokens* fossem "de" e "madeira", após a passagem pelo autômato, o Agrupador formaria um sintagma SP. Considerando que na pilha ele tinha armazenado um SS, após a passagem pelo bigrama, e de acordo com a Tabela 3, o sintagma resultante seria um SS e o conteúdo que o compõe seria a combinação dos textos de cada sintagma que o originou. Caso não haja agrupamentos possíveis, o Agrupador envia o último sintagma formado para análise do Reconhecedor Sintático e movimenta o sintagma atual para a posição de último sintagma no bigrama, repetindo o processo com o próximo sintagma.

O quinto e último módulo, denominado Reconhecedor Sintático, recebe os sintagmas do módulo anterior e verifica se estão sintaticamente corretos de acordo com padrões oracionais de Luft. O Reconhecedor Sintático utiliza um autômato adaptativo que faz chamadas recursivas sempre que recebe conjunções ou pronomes relativos, armazenando, em uma estrutura de pilha, o estado e a cadeia de sintagmas reconhecidos até o momento da chamada. Caso o Reconhecedor Sintático não consiga se movimentar a partir do sintagma recebido, ele gera um erro e retorna o ponteiro para o

último sintagma reconhecido, finalizando a instância do autômato recursivo e retornando o processamento para aquela que a inicializou. Esta, por sua vez, retoma posição em que se encontrava antes da chamada e continua o processamento até o final da sentença ou até encontrar uma nova conjunção, situação na qual o processo se repete.

A configuração completa do autômato é definida da seguinte forma:

Estados = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27}

Tokens = {SS, SP, Vli, Vi, Vtd, Vti, Vtdi, Sadj, Sadv, Conj, A}

Estados de Aceitação = { 4, 5, 6, 9, 12, 13,14, 15, 17, 18, 19, 21, 22, 24, 25, 26, 27}

Estado Inicial = {1}

Função de Transição = {(Estado, *Token*)→Estado}, sendo:

{(1, SS)→2, (2, Vti)→3, (3, SP)→4, (4, SP)→4,
(3, Sadv)→5, (2, Vi)→6, (2, Vtdi)→7
(7, SS)→8, (8, SP)→9, (9, SP)→9, (8, Sadv)→10,
(2, Vlig)→11, (11, SP)→12, (11, Sadv)→13,
(11, Sadj)→ 14, (11, SS)→15, (2, Vtd)→16, (16, SS)→17,
(2, Vtpred)→18, (18, SP)→19, (18, Sadj)→20
(18, SS)→ 21, (21, SS)→22, (21, Sadj)→23, (21, Sadv)→24,
(21, SP)→25}

Pilha = {[Texto, Sintagma, Estado]}

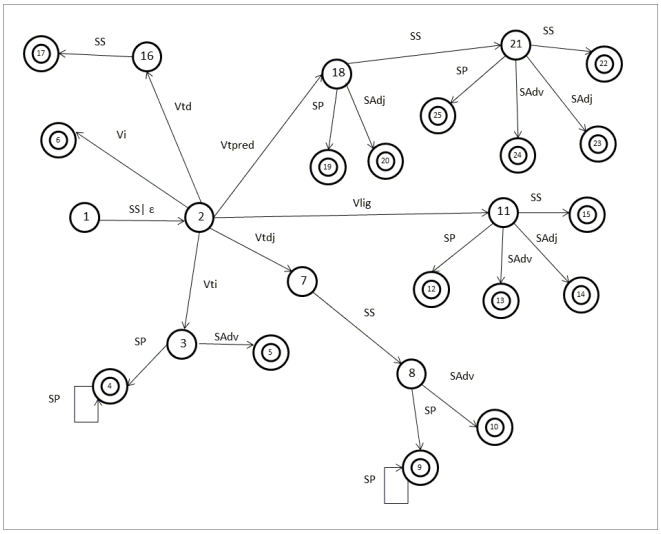

Figura 4.1. Configuração Completa do Reconhecedor Sintático.

No entanto, para que a análise sintática seja feita, não são necessárias todas as ramificações da configuração completa do autômato (Fig. 4.1). Por exemplo, quando se transita um verbo de ligação a partir do estado 2, o autômato vai para o estado 11 e todas as demais ramificações que partem deste estado para os estados 3, 7, 16 e 18, não são usadas. Com a tecnologia adaptativa, é possível criar dinamicamente os estados e transições do autômato em função dos tipos de verbos, evitando manter ramificações que não são usadas.

A Fig. 4.2 apresenta a configuração inicial do autômato adaptativo equivalente ao autômato de pilha apresentado anteriormente. No estado 1, o autômato recebe os *tokens* e transita para o estado 2 quando processa um sintagma substantivo (SS) ou quando transita em vazio. No estado 2, o autômato transita para si mesmo quando recebe qualquer tipo de verbo: Vi, Vtd, Vlig, Vtpred, Vtdi e Vti. Todas as outras ramificações são criadas por meio de funções adaptativas chamadas em função do tipo de verbo processado.

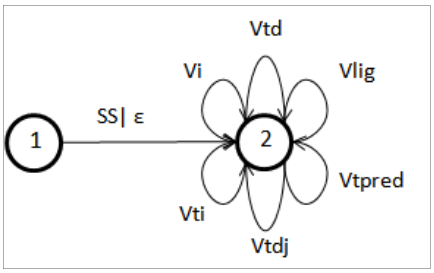

Figura 4.2. Configuração Inicial do Reconhecedor Gramatical.

Por exemplo, se o verbo é de ligação (Vlig), o autômato utiliza as funções adaptativas α (j) e β (o), definidas da seguinte forma:

α (j): { o* :
- [(j, Vlig)]
+[(j, Vlig) :→ o, β (o)]
}

β (o): { t*u*v*x*:
+ [(o, SP) :→ t]
+ [(o, Sadv) :→ u]
+ [(o, Sadj) :→ v]
+ [(o, SS) :→ x]
}

A função adaptativa α (j) é chamada pelo autômato antes de processar o *token*, criando o estado 11 e a produção que leva o autômato do estado 2 ao novo estado criado. Em seguida, o autômato chama a função β (o), criando os estados 12, 13, 14 e 15 e as produções que interligam o estado 11 aos novos estados. A Fig. 4.3 mostra a configuração do autômato após o processamento do verbo de ligação. Neste exemplo, o autômato criou apenas os estados 11, 12, 13, 14 e 15 e as respectivas transições, evitando alocar recursos que seriam necessários para criar o autômato completo, conforme apresentado na Fig. 4.1.

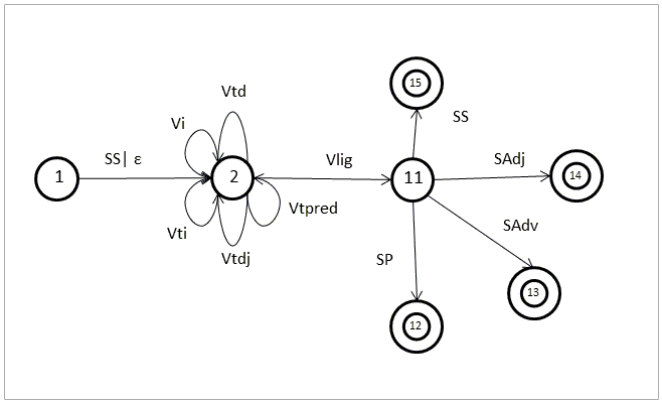

Figura 4.3. Configuração do autômato após o processamento do verbo de ligação.

V. DESAMBIGUAÇÃO SINTÁTICA E SEMÂNTICA

Iwai [20] apresenta um formalismo, denominado Gramáticas Adaptativas, no qual as regras gramaticais são criadas dinamicamente a partir do processamento da cadeia de entrada e de informações relacionadas ao contexto em que se apresentam, permitindo a identificação e resolução de ambiguidades semânticas. A autora demonstra, ainda, a

equivalência computacional entre as Gramáticas Adaptativas e os autômatos adaptativos, o que permite que frases da linguagem gerada por uma gramática adaptativa sejam reconhecidas por autômatos adaptativos.

O Linguístico incorpora o formalismo proposto pela autora, através do mapeamento dos sintagmas e padrões oracionais de Luft para uma gramática adaptativa. As ações adaptativas são usadas para modificar as regras de produção em função do contexto definido pelo tipo do verbo encontrado no texto e pela presença de conjunções ou pronomes relativos. O resultado final é o seguinte:

$G = (G^0, T, R^0)$, onde:

$G^0 = (VN^0, VT, VC, PL^0, PD^0, S)$

R^0 = relação que associa regras de produção às ações adaptativas.

VN0 = { S, con, SS, SV, SS, Sadj, SP,num, PrA, Sc, Sp, PrPes, Neg, Aux, PreV,V, Vlig ,Vtda,Vtdna , Vtdi, Vtpred ,Vti ,Vi, Sadv, Prep, Ind, ArtDef, ArtInd, Dem, Pos, PrRel, Adv, Adj, pont }

VT = {{léxico},{,}, ; , . ,:,!,?,1, 2, 3, 4, 5, 6, 7,8, 9, 0} V_C = ∅, P_{D0} = ∅

PL0 = { S → [con] [PrRel] [SS] SV pont [S]

SS → [Sadj] SS [Sadj | SP] [S] | S

SS → (([num | PrA] Sc) | Sp | PrPes)

SV→ [Neg] [Aux | PreV] V [SS | Sadj | Sadv | SP] | ((SS SP) | (SS Sadj) | (SP SP) | (SS SS) | (SS Sadv) | (SS SP SP))

V → (Vlig | Vtda | Vtdna | Vtdi | Vtpred |Vti |Vi)

SP → Prep (SS | Sadj)

Sadj → Sadj [SP] [S] | S

Sadj → [Adv] Adj [S]

Sadv → Sadv [SP] [S] | S

Sadv → [Adv] Adv [S]

PrA → [PrA] (Ind | ArtDef | ArtInd| Dem| Pos)

con → (aditiva | adversativa | alternativa | conclusiva| explicativa | integrante | causal | comparativa | concessiva | condicional | conformativa | consecutiva | final | proporcional | temporal)

num→ (0 |1 |2 | 3 |4 |5 |6 |7 |8 |9)| (cardinal| ordinal| multiplicativo| fracionario)

pont → (,| ; | . |: |! |?)}

Sendo:

F – Frase
SS – Sintagma substantivo
SV – Sintagma verbal
SP – Sintagma preposicional
SN – Sintagma nominal
Sadv – Sintagma adverbial
Sadj – Sintagma adjetivo
Adv – adverbio
Adj – adjetivo
ArtDef – artigo definido
ArtInd – artigo indefinido
Aux – Partícula auxiliar (apassivadora ou pré-verbal)
Conec – Conector (conjunção ou pronome relativo)
Dem – pronome demonstrativo indefinido
Ind – pronome indefinido
Neg – partícula (negação)
PrA – pronome adjetivo
PrPes – pronome pessoal
Prep – preposição
Quant – numeral
Sc – substantivo comum
Sp – substantivo próprio
V– verbo
Vlig – verbo de ligação
Vi – verbo intransitivo
Vtd – verbo transitivo direto
Vti – verbo transitivo indireto
Vtdi – verbo transitivo direto e indireto

A análise semântica das sentenças é feita de acordo com os termos processados, que podem gerar novas regras de produção e exclusão de regras existentes. Por exemplo, quando o Linguístico encontra um verbo transitivo indireto, ele dispara uma ação adaptativa A (Vti) que elimina regras iniciais definidas na gramática G^0 e cria outras, relacionadas aos complementos oracionais exigidos por este tipo de verbo. A representação destas alterações é feita da seguinte forma:

$$A\ (t = Vti) = L\ (t) = \{ + [R: {}_G{}^0 P_{26\text{-}66,\ 69,77\text{-}78,\ 87,\ 93,\ 101\text{-}102,\ 113\text{-}114}: {}_G{}^1 V \leftarrow \chi \backslash V] + [{}_{G1}\chi V \rightarrow Vti] - [R: {}_G{}^0 P_{67\text{-}68,70\text{-}76,79\text{-}86,\ 88\text{-}100,103\text{-}112,115\text{-}120}: {}_G{}^1 \varnothing]$$

A sintaxe da instrução R é a seguinte:

R: $<{}_G{}^n> <P_{I\text{-}F}> : <{}_G{}^{n+1}> P'$ (4)

onde:

R: replace

${}_G{}^n$: gramática anterior à atualização

$P_{I\text{-}F}$: regras de produção para atualização, sendo I = Inicio de intervalo e F= fim de intervalo. Acrescenta-se mais de um grupo de regras usando a vírgula para separação.

${}_G{}^{n+1}$: gramática posterior à atualização

P': nova regra de produção

∅ : símbolo que indica que as regras assinaladas em $_{PI\text{-}F}$ serão removidas

O formalismo apresentado por Iwai é bastante prático e fornece suporte para incluir características adicionais que o parser pode utilizar para resolver problemas em que o contexto seja importante para definição das regras de produção. Por exemplo, o parser pode verificar se a frase analisada está correta ou resolver possíveis ambiguidades caso tenha à disposição informações sobre o tipo do verbo, tempo, modo e pessoa do verbo analisado. No exemplo abaixo, a ação adaptativa A utiliza como parâmetros de entrada a identificação do tipo do verbo, modo e pessoa para definição das funções adaptativas.

T = { A (verbo=Vlig, tempo = Presente, modo = Singular, pessoa = Primeira) = F (verbo, tempo, modo, pessoa) =
{ + [R: ${}_G{}^n P_{inicio\text{-}fim}$: ${}_G{}^n$+1V ← χV]
+ [${}_G{}^{n+1}$χV → Vlig, Presente, Singular, Primeira]
+ [R: ${}_G{}^n P_{inicio\text{-}fim}$: ${}_G{}^{n+1}$ ∅] }

Analogamente, é possível incluir regras para análise da concordância nominal ou qualquer outro tipo de conteúdo semântico, sem necessidade de usar qualquer outro elemento

fora do formalismo. As gramáticas adaptativas também podem ser usadas para representar o uso de informações probabilísticas na seleção das regras de produção. Tal técnica é usada quando existe mais de uma regra de produção aplicável e não existem informações sintáticas e semânticas suficientes para desambiguá-las. Neste caso, é possível usar a probabilidade de ocorrência das regras como fator de escolha e o formalismo de Iwai pode ser usado representar este tipo de informação contextual. No exemplo abaixo, a ação adaptativa A utiliza como parâmetro de entrada a probabilidade de ocorrência das regras de produção avaliadas e uma indicação para usar a regra de máxima probabilidade.

SVprob → V SP 10%
SVprob → V (SS SP) 90%

T = { A (t=SV, u = prob, v=max) = F (SV, prob, max) =
{ + [R: $_G^n$ $P_{inicio-fim}$: $_G^{n+1}$SV ← χSVprob]
+ [$_G^{n+1}$ $\chi SV_{prob} \rightarrow \chi SV_{max}$]
? [$_G^{n+1}$ χSV_{max}]
+ [$_G^{n+1}$ χSV_{max} → V (SS SP)]
+ [R: $_G^n$ $P_{inicio-fim}$: $_G^{n+1}$ ∅] }

VI. CONSIDERAÇÕES FINAIS

Este artigo apresentou uma revisão dos conceitos de Tecnologia Adaptativa e de Processamento da Linguagem Natural. Em seguida, foi apresentado o Linguístico, uma proposta de reconhecedor gramatical que utiliza autômatos adaptativos como tecnologia subjacente. Por fim, procurou-se apresentar cenários em que a gramática utilizada pelo Linguístico é estendida, incorporando critérios semânticos para auxiliar na resolução de ambiguidades.

O Linguístico está fase de desenvolvimento e testes preliminares confirmaram a tecnologia adaptativa como uma alternativa válida para o processamento da linguagem natural, proporcionando economia de recursos computacionais e flexibilidade para implantação de novas regras, o que nos motiva a aprofundar a pesquisa, provendo embasamento quantitativo para chegar a respostas mais conclusivas.

REFERÊNCIAS

[1] http://www.pcs.usp.br/~lta/
[2] C. Taniwaki. Formalismos adaptativos na análise sintática de Linguagem Natural . Dissertação de Mestrado, EPUSP, São Paulo, 2001.
[3] C. E. Menezes. Um método para a construção de analisadores morfológicos, aplicado à língua portuguesa, baseado em autômatos adaptativos. Dissertação de Mestrado, Escola Politécnica da Universidade de São Paulo, 2000.
[4] D. Padovani. Uma proposta de autômato adaptativo para reconhecimento de anáforas pronominais segundo algoritmo de Mitkov. Workshop de Tecnologias Adaptativas –WTA 2009, 2009.
[5] M. Moraes. Alguns aspectos de tratamento sintático de dependência de contexto em linguagem natural empregando tecnologia adaptativa, Tese de Doutorado, Escola Politécnica da Universidade de São Paulo, 2006.
[6] E. Rich and K. Knight. Inteligência Artificial, 2. Ed. São Paulo: Makron Books, 1993.
[7] R. Vieira and V. Lima. Linguística computacional: princípios e aplicações. IX Escola de Informática da SBC-Sul, 2001.
[8] C. Fuchs and P. Le Goffic. Les Linguistiques Contemporaines.
[9] M. G. V.Nunes et al. Introdução ao Processamento das Línguas Naturais. Notas didáticas do ICMC Nº 38, São Carlos, 88p, 1999. Paris, Hachette, 1992. 158p.
[10] T. B. Sardinha, A Língua Portuguesa no Computador. 295p. Mercado de Letras, 2005.
[11] R.L.A. Rocha. Tecnologia Adaptativa Aplicada ao Processamento Computacional de Língua Natural. Workshop de Tecnologias Adaptativas – WTA 2007, 2007.
[12] C. Luft. Moderna Gramática Brasileira. 2ª. Edição Revista e Atualizada. 265p. Editora Globo, 2002.
[13] http://www.linguateca.pt/
[14] http://www.nilc.icmc.usp.br/tep2/
[15] http://www.portalsaofrancisco.com.br/alfa/materias/index-lingua-portuguesa.php/
[16] M. Basilio. Formação e Classes de Palavras no Português do Brasil. Ed.Contexto, 2004.
[17] M. Basilio. Teoria Lexical. Ed.Atica, 1987.
[18] http://www.cs.columbia.edu/~mcollins/hmms-spring2013.pdf/
[19] G.D. J. Forney. IEEE. Proceedings of the IEEE, Volume 61 , Issue 3,1973.
[20] M., Iwai: Um Formalismo Gramatical para Linguagens Dependentes de Contexto. São Paulo, 2000. Escola Politécnica da Universidade de São Paulo. Tese de Doutorado (2000).

Ana Teresa Contier formou-se em Letras-Português pela Universidade de São Paulo (2001) e em publicidade pela PUC-SP (2002). Em 2007 obteve o título de mestre pela Poli-USP com a dissertação: "Um modelo de extração de propriedades de textos usando pensamento narrativo e paradigmático".

Djalma Padovani formou-se em administração de empresas pela Faculdade de Economia e Administração da Universidade de São Paulo, em 1987 e obteve o mestrado em engenharia de software pelo Instituto de Pesquisas Tecnológicas de São Paulo - IPT, em 2008. Trabalhou em diversas empresas nas áreas de desenvolvimento de software e tecnologia de informação e atualmente é atua no Laboratório de Dados da Serasa Experían.

João José Neto graduado em Engenharia de Eletricidade (1971), mestrado em Engenharia Elétrica (1975) e doutorado em Engenharia Elétrica (1980), e livre-docência (1993) pela Escola Politécnica da Universidade de São Paulo. Atualmente é professor associado da Escola Politécnica da Universidade de São Paulo, e coordena o LTA - Laboratório de Linguagens e Tecnologia Adaptativa do PCS - Departamento de Engenharia de Computação e Sistemas Digitais da EPUSP. Tem experiência na área de Ciência da Computação, com ênfase nos Fundamentos da Engenharia da Computação, atuando principalmente nos seguintes temas: dispositivos adaptativos, tecnologia adaptativa, autômatos adaptativos, e em suas aplicações à Engenharia de Computação, particularmente em sistemas de tomada de decisão adaptativa, análise e processamento de linguagens naturais, construção de compiladores, robótica, ensino assistido por computador, modelagem de sistemas inteligentes, processos de aprendizagem automática e inferências baseadas em tecnologia adaptativa.