UNIVERSIDADE DE SÃO PAULO
ESCOLA DE ARTES, CIÊNCIAS E HUMANIDADES
PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS DE INFORMAÇÃO

GILMAR PEREIRA DOS SANTOS

**Analisador sintático de Earley para gramáticas livres de contexto adaptativas
e sua aplicação na caracterização de famílias de RNAs com pseudonós**

São Paulo

2018

GILMAR PEREIRA DOS SANTOS

**Analizador sintático de Earley para gramáticas livres de contexto adaptativas
e sua aplicação na caracterização de famílias de RNAs com pseudonós**

Dissertação apresentada à Escola de Artes,
Ciências e Humanidades da Universidade de
São Paulo para obtenção do título de Mestre
em Ciências pelo Programa de Pós-graduação
em Sistemas de Informação.

Área de concentração: Metodologia e
Técnicas da Computação

Versão corrigida contendo as alterações
solicitadas pela comissão julgadora em 26 de
outubro de 2018. A versão original encontra-
se em acervo reservado na Biblioteca da
EACH-USP e na Biblioteca Digital de Teses
e Dissertações da USP (BDTD), de acordo
com a Resolução CoPGr 6018, de 13 de
outubro de 2011.

Orientador: Profa. Dra. Ariane Machado
Lima

São Paulo

2018

Autorizo a reprodução e divulgação total ou parcial deste trabalho, por qualquer meio convencional ou eletrônico, para fins de estudo e pesquisa, desde que citada a fonte.

CATALOGAÇÃO-NA-PUBLICAÇÃO

(Universidade de São Paulo. Escola de Artes, Ciências e Humanidades. Biblioteca)
CRB-8 4936

Santos, Gilmar Pereira dos

Analizador sintático de Earley para gramáticas livres de contexto adaptativas e sua aplicação na caracterização de famílias de RNAs com pseudônimos / Gilmar Pereira dos Santos ; orientadora, Ariane Machado Lima. – 2018.

172 f. : il.

Dissertação (Mestrado em Ciências) - Programa de Pós-Graduação em Sistemas de Informação, Escola de Artes, Ciências e Humanidades, Universidade de São Paulo.

Versão corrigida

1. Reconhecimento de padrões. 2. Análise sintática. 3. Linguagens formais. 4. Linguagens livres de contexto. 5. RNA. I. Lima, Ariane Machado, orient. II. Título.

CDD 22.ed.– 006.4

Dissertação de autoria de Gilmar Pereira dos Santos, sob o título “**Analisador sintático de Earley para gramáticas livres de contexto adaptativas e sua aplicação na caracterização de famílias de RNAs com pseudonós**”, apresentada à Escola de Artes, Ciências e Humanidades da Universidade de São Paulo, para obtenção do título de Mestre em Ciências pelo Programa de Pós-graduação em Sistemas de Informação, na área de concentração Metodologia e Técnicas da Computação, aprovada em 26 de outubro de 2018 pela comissão julgadora constituída pelos doutores:

Profa. Dra. Arianne Machado Lima

Universidade de São Paulo

Presidente

Prof. Dr. Amaury Antônio de Castro Junior

Universidade Federal de Mato Grosso do Sul

Prof. Dr. Luiz Carlos da Silva Rozante

Universidade Federal do ABC

Prof. Dr. Marcelo de Souza Lauretto

Universidade de São Paulo

Agradecimentos

Em primeiro lugar gostaria de agradecer a Deus, o maior arquiteto e engenheiro de todos os tempos. Sensacional o trabalho programando cada molécula desse nosso mundo complexo (e de sei lá quantos outros mundos que nem sabemos que existem).

Agradeço especialmente a minha orientadora, professora Ariane Machado Lima, pela paciência, dedicação e empolgação. Sempre disposta a ajudar a melhorar o texto, deixando desenvolver o trabalho com serenidade e sempre com os pés no chão.

Agradeço ao professor João José Neto, sempre humilde e parceiro, abriu para mim as portas do LTA e apresentou o mundo dos formalismos adaptativos.

Agradecimentos aos meus pais Francisco e Amélia, razões de minha vida e responsáveis pela pessoa que sou hoje.

À minha querida Bia, companheira sempre presente em cada momento alegre ou triste.

Muito obrigado a todos!

Resumo

SANTOS, Gilmar Pereira. **Analisador sintático de Earley para gramáticas livres de contexto adaptativas e sua aplicação na caracterização de famílias de RNAs com pseudonós**. 2018. 172 f. Dissertação (Mestrado em Ciências) – Escola de Artes, Ciências e Humanidades, Universidade de São Paulo, São Paulo, 2018.

A teoria das linguagens formais é amplamente utilizada nos processos de solução de problemas de naturezas diversas, uma vez que tem poder de lidar tanto com as linguagens artificiais quanto com as linguagens naturais. As gramáticas, formalismos capazes de sintetizar as linguagens, podem também ser utilizadas no âmbito do problema de reconhecimento de padrões por poderem modelar as hierarquias dos componentes da linguagem, decompondo padrões em subestruturas. Seguindo essa linha, o arcabouço GrammarLab, cujo objetivo é facilitar a implementação, geração e testes de diferentes classificadores de sequências baseados em gramáticas, permitia em sua implementação anterior o uso de gramáticas regulares e livres de contexto. No entanto, alguns problemas necessitam de formalismos presentes apenas em gramáticas de níveis superiores na hierarquia de Chomsky. O problema encontrado ao se subir a hierarquia de gramáticas é a complexidade de tempo necessária para a análise sintática. Enquanto o reconhecimento de sequências por gramáticas regulares e livres de contexto pode ser feito em tempo polinomial, o problema geral de reconhecimento por gramáticas sensíveis ao contexto é um problema NP-completo e o de gramáticas irrestritas é considerado indecidível no caso geral. No entanto, o uso de métodos adaptativos possibilita que uma gramática altere seu conjunto de regras de produção durante a geração de sentenças, adicionando sensibilidade ao contexto a gramáticas originalmente livres de contexto, sem prejudicar a complexidade de análise polinomial. Desta forma, este trabalho teve como foco a inserção de métodos adaptativos no arcabouço GrammarLab e a criação de uma versão adaptativa do algoritmo de Earley de análise sintática. Como forma de verificar sua aplicação em problemas reais, foi realizado um estudo preliminar do uso do arcabouço na caracterização de famílias funcionais de RNAs com estrutura conservada, incluindo pseudonós. Os pseudonós apresentam relações de dependências cruzadas entre os nucleotídeos de uma sequência de RNA, relação esta que exemplifica dependência de contexto, sendo portanto um bom caso para o uso do modelo com adaptatividade em sua constituição. Os resultados obtidos com duas famílias de RNAs com pseudonós mostraram que a abordagem é altamente promissora.

Palavras-chaves: Reconhecimento de Padrões. Métodos Sintáticos. Métodos Adaptativos. Gramáticas. Classificação. RNA. Pseudonós.

Abstract

SANTOS, Gilmar Pereira. **Earley's syntactic analyzer for adaptive context-free grammars and its application in the characterization of RNA families with pseudoknot**. 2018. 172 p. Dissertation (Master of Science) – School of Arts, Sciences and Humanities, University of São Paulo, São Paulo, 2018.

The theory of formal languages is widely used to solve problems of different natures as it can deal with artificial and natural languages. The grammars, formalisms able to synthesize languages, can also be used in pattern recognition problems due to the ability to model the language components hierarchies, decomposing patterns in substructures. Based on this idea, the framework GrammarLab was designed to facilitate the work involved in implementing, generating and testing different grammar based sequence classifiers, providing regular and context free grammar in the prior version. However, some problems need a formalism that can be found only in higher classes of grammars in the Chomsky hierarchy. The problem of using a higher class of grammar is the high computational time complexity for parsing. While the problem of recognizing sequences using regular and context free grammars is solved at polynomial time, the same problem in general case is NP-Complete for context sensitive grammars and undecidable for unrestricted grammars. Nevertheless, the use of adaptive methods allows a grammar to alter the set of production rules during sentences generation, including context sensitivity even to grammars that were designed to be context free, without increasing the polynomial parsing complexity. This work was focused in improving the GrammarLab framework by including the ability to deal with adaptive methods and in the creation of an adaptive version of Earley's algorithm. To test the solution in real world problems, it was conducted a preliminary study of the use of the framework in characterizing RNA functional families with conserved secondary structure, including pseudoknots. The pseudoknot pattern, represented by crossing dependences among RNA sequence nucleotides, is an example of context dependence, so it is a good test case for the use of a model that consider adaptability in the constitution. The obtained results with two families of RNAs with pseudoknots show that the approach is promising.

Keywords: Pattern Recognition. Syntactic Methods. Adaptive Methods. Grammars. Classification. RNA. Pseudoknot.

Lista de figuras

Figura 1 – Hierarquia de Chomsky	19
Figura 2 – Diagrama de estados de um autômato finito	21
Figura 3 – Exemplo de árvore sintática	22
Figura 4 – Diagrama de estados de um autômato com pilha	24
Figura 5 – Diagrama de funcionamento do GrammarLab	31
Figura 6 – Dependência cruzada.	38
Figura 7 – Estrutura de DNA x RNA	39
Figura 8 – Estrutura secundária x estrutura tridimensional de um RNA	40
Figura 9 – Elementos de uma estrutura secundária de RNA	41
Figura 10 – Tipos de pseudonós	42
Figura 11 – Alinhamento estrutural - formato stockholm utilizado no RFAM	43
Figura 12 – Comparação entre RNaseP do <i>Plasmodium vivax</i> e da <i>Entamoeba histolytica</i>	45
Figura 13 – Estrutura secundária do componente RNA da telomerase de ciliatos, vertebrados e leveduras	46
Figura 14 – Estrutura secundária de um RNA e árvore de derivação para uma gramática simples	47
Figura 15 – Exemplo de um conjunto de regras de produção e de uma derivação de uma sequência de RNA	49
Figura 16 – Árvore de derivação e estrutura secundária	50
Figura 17 – Alinhamento múltiplo e estrutura consenso	51
Figura 18 – Árvore guia do modelo de covariância	52
Figura 19 – Arquitetura completa do modelo de covariância	53
Figura 20 – Pseudonó desmembrado em duas gramáticas	54
Figura 21 – Exemplo de uma assinatura topológica de RNA	55
Figura 22 – Vírus da hepatite D representado em formato de grafos	56

Figura 23 – Estrutura secundária das famílias Alpha operon ribosome binding site (código no banco Rfam: RF00140) (A) e Hepatitis delta virus ribozyme (código no banco Rfam: RF00094) (B). Os números em (A) indicam as hélices. Os números em (B) indicam a posição relativa do nucleotídeo na sequência e as setas indicam a direção de leitura $5' \rightarrow 3'$	58
Figura 24 – Pseudonó de grau k	58
Figura 25 – Alinhamento entre duas estruturas, considerando triplas de bases . . .	59
Figura 26 – Exemplo de uma estrutura de RNA contendo pseudonós e sua modelagem em grafos. Cada S_i representa uma hélice. Cada vértice V_i representa uma hélice S_i . As indicações P, K e N representam hélices com relações paralelas (P), aninhadas (N) e hélices que se cruzam formando pseudonós (K)	61
Figura 27 – Codificação BEAR	63
Figura 28 – Arquitetura utilizada no algoritmo PMFastR	64
Figura 29 – R-PASS: Conversão entre alinhamento de estruturas e alinhamento de sequências	65
Figura 30 – GrammarLab - Gramática Livre de Contexto	69
Figura 31 – GrammarLab - Gramática Adaptativa	70
Figura 32 – GrammarLab - Modelagem de InputStream	71
Figura 33 – GrammarLab - Modelagem de GrammarFactoryInputStream	72
Figura 34 – GrammarLab - Algoritmo de Earley Adaptativo	84
Figura 35 – Processo de busca	90
Figura 36 – RydC RNA - Família RF00505	91
Figura 37 – RydC RNA - alinhamento múltiplo das sequências curadas	92
Figura 38 – Alinhamento múltiplo das sequências curadas com a sequência rejeitada da família RF00505	94
Figura 39 – Estrutura secundária do componente RNA da telomerase de vertebrados. O <i>template</i> é uma região não pareada utilizada como molde da polimerização das repetições teloméricas encontradas nas extremidades dos cromossomos.	95

Lista de algoritmos

Algoritmo 1 – Algoritmo de Earley para reconhecimento de gramáticas livres de contexto 30

Algoritmo 2 – Algoritmo de Earley Adaptativo para reconhecimento de gramáticas adaptativas 86

Lista de quadros

Quadro 1 – Notação BNF para gramáticas adaptativas	74
Quadro 2 – Exemplo de notação BNF para uma gramática adaptativa que descreve a linguagem $L(G) = \{a^n b^m c^n d^m\}$	76

Lista de tabelas

Tabela 1 – Função δ de um autômato finito	21
Tabela 2 – Famílias de RNAs com pseudonós na estrutura secundária	89
Tabela 3 – Desempenho da gramática adaptativa quando aplicada a sequências curadas de todas as famílias	93
Tabela 4 – Estatísticas do tamanho das sequências curadas. $\bar{x}$ e s são, respectiva- mente, a média e o desvio padrão da amostra	95
Tabela 5 – Desempenho da gramática adaptativa quando aplicada a sequências curadas de todas as famílias	96
Tabela 6 – Algoritmo de Earley adaptativo - tempo de CPU	96
Tabela 7 – Desempenho da validação cruzada simplificada realizada com a família RF00024	97

Sumário

1	INTRODUÇÃO	14
1.1	<i>Objetivos</i>	15
1.2	<i>Organização deste documento</i>	16
2	CONCEITOS FUNDAMENTAIS	17
2.1	<i>Linguagens Formais</i>	17
2.1.1	Hierarquia de Chomsky	18
2.1.2	Gramáticas estocásticas	25
2.1.3	Analisadores Sintáticos	26
2.1.4	Algoritmo de Earley	28
2.1.5	GrammarLab: Laboratório de geração de classificadores de sequências baseados em gramáticas	30
2.1.6	Dispositivos adaptativos	32
2.2	<i>RNA</i>	39
2.2.1	Estruturas primária, secundária e terciária	40
2.2.2	Famílias funcionais	43
2.2.3	Modelagem de RNAs com gramáticas	46
3	CARACTERIZAÇÃO COMPUTACIONAL DE SEQUÊNCIAS DE RNAs - UMA REVISÃO BIBLIOGRÁFICA	48
3.1	<i>Abordagens gramaticais</i>	48
3.1.1	Gramáticas desenhadas manualmente para caracterização de sequência e estrutura secundária	48
3.1.2	Modelos de covariância	50
3.1.3	Intersecção de gramáticas livres de contexto estocásticas para repre- sentação de pseudonós	54
3.2	<i>Abordagens não gramaticais</i>	54
3.2.1	Métodos que consideram pseudonós	55
3.2.2	Métodos que não consideram pseudonós	62
4	GRAMMARLAB ADAPTATIVO	68

4.1	<i>Gramáticas adaptativas</i>	68
4.2	<i>Notação BNF para gramáticas adaptativas</i>	71
4.3	<i>Algoritmo de Earley adaptativo</i>	75
4.3.1	Exemplo de aplicação do algoritmo para reconhecimento	77
4.3.2	Estratégia de implementação do algoritmo	83
5	APLICAÇÃO: CARACTERIZAÇÃO DE FAMÍLIAS DE RNAs COM PSEUDONÓS	88
5.1	<i>Estratégia de testes</i>	88
5.1.1	Seleção de famílias com pseudonós	88
5.1.2	Análise sintática das sequências do Rfam à procura das famílias selecionadas	89
5.2	<i>Família RF00505 - RycD</i>	91
5.2.1	Estratégia de Modelagem	92
5.2.2	Resultados	92
5.3	<i>Família RF00024 - Componente RNA da telomerase de vertebrados</i> .	94
5.3.1	Estratégia de Modelagem	95
5.3.2	Resultados	96
6	CONCLUSÃO	98
6.1	<i>Considerações finais</i>	98
6.2	<i>Principais contribuições</i>	99
6.3	<i>Trabalhos futuros</i>	100
6.4	<i>Artigos científicos</i>	101
	Referências¹	102
	 APÊNDICES	109
	Apêndice A – EXECUÇÃO - EARLEY ADAPTATIVO . . .	110
	Apêndice B – GRAMÁTICA - FAMÍLIA RF00505	115
	Apêndice C – GRAMÁTICA - FAMÍLIA RF00024	119

¹ De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

1 INTRODUÇÃO

A teoria das linguagens formais, elaborada com o objetivo de desenvolver teorias relacionadas com linguagens naturais, logo passou a ser notada como importante também para o estudo das linguagens artificiais (MENEZES, 2009). Assim, as linguagens formais passaram a ser utilizadas amplamente na análise sintática de linguagens de programação, na modelagem de circuitos e redes lógicas, em sistemas biológicos, sistemas de animação, hipertexto e outros (MENEZES, 2009).

O conjunto de cadeias que compõem uma linguagem pode ser exaustivamente sintetizado por gramáticas, que são sistemas formais baseados em regras de substituição (RAMOS; NETO; VEGA, 2009). Para cada classe de linguagem há uma gramática capaz de a identificar, seguindo uma hierarquia de acordo com a complexidade do formalismo (CHOMSKY, 1959).

As gramáticas também podem ser utilizadas para reconhecimento de padrões na abordagem sintática, uma vez que podem modelar a hierarquia dos componentes de uma linguagem, podendo ser traçado um paralelo entre esta hierarquia e a decomposição de padrões em subestruturas (JAIN; DUIN; MAO, 2000).

Foi justamente visando a essa finalidade que o arcabouço GrammarLab (MACHADO-LIMA, 2002) foi desenvolvido. Sua função é facilitar a implementação, geração e teste de diferentes classificadores de sequências baseados em gramáticas. Em sua primeira versão o arcabouço permitia o uso das duas classes gramaticais mais simples – regulares e livres de contexto (detalhes serão apresentados no Capítulo 2). Para a análise sintática de gramáticas livres de contexto o GrammarLab dispõe da implementação do algoritmo de Earley (EARLEY, 1970), pois este não exige que as regras de produção estejam em alguma forma (o que poderia afetar a informação estrutural fornecida pelas árvores sintáticas da gramática original), e é capaz de lidar com gramáticas ambíguas computando todas as árvores possíveis, o que de novo é importante quando se quer conhecer todas as estruturas possíveis ou para a caracterização estatística das linguagens geradas por gramáticas estocásticas. No entanto, alguns problemas necessitam de um formalismo presente apenas em níveis superiores da hierarquia de gramáticas (sensibilidade ao contexto). Sabe-se, porém, que o problema geral de reconhecimento de sequências por gramáticas sensíveis a contexto é NP-completo (BROWN; WILSON, 1995; SEARLS, 1997; RIVAS; EDDY, 2000).

No entanto, o uso de métodos adaptativos (RAMOS; NETO; VEGA, 2009) possibilita alterar dinamicamente as propriedades de uma gramática, viabilizando inserir sensibilidade ao contexto em uma gramática originalmente livre de contexto sem aumentar sua complexidade de análise.

Uma das aplicações de reconhecimento sintático de padrões é em bioinformática, particularmente destacado neste trabalho a caracterização de RNAs em famílias funcionais utilizando informação estrutural das sequências. Como a função que um RNA desempenha normalmente está muito mais relacionada com sua estrutura do que com sua sequência (NOVIKOVA; HENNELLY; SANBONMATSU, 2012b; DIXON; HILLIS, 1993; LANGE *et al.*, 2012; SEEMANN *et al.*, 2012; NOVIKOVA; HENNELLY; SANBONMATSU, 2012a), comparações estruturais e identificação de padrões comuns nestas estruturas se tornam um mecanismo importante para a caracterização funcional destas moléculas.

Alguns padrões estruturais complexos encontrados nas moléculas de RNAs, como os pseudonós, por apresentarem relações de dependências cruzadas (SEARLS, 1997), não podem ser representados por gramáticas livres de contexto, sendo necessário o uso de gramáticas sensíveis ao contexto (SEARLS, 1992). No entanto, como ao se caracterizar uma família específica de RNAs pode-se restringir ao reconhecimento de um tipo específico de pseudonó, o uso de gramáticas adaptativas para conferir ao modelo sensibilidade ao contexto parece ser uma alternativa promissora. No entanto, na revisão bibliográfica conduzida neste trabalho (apresentada no Capítulo 3) não foi encontrada nenhuma iniciativa nesse sentido.

1.1 Objetivos

O objetivo geral deste trabalho foi iniciar a inserção de métodos adaptativos no arcabouço GrammarLab por meio da modelagem de gramáticas adaptativas e de uma versão adaptativa do algoritmo de Earley de análise sintática, além de um estudo preliminar de sua potencial aplicação na caracterização de famílias funcionais de RNAs com estrutura conservada, incluindo pseudonós. Para isso, os seguintes objetivos específicos foram estabelecidos:

1. Estudo da tecnologia adaptativa, em particular de gramáticas adaptativas;

2. Levantar o estado da arte na caracterização computacional de RNAs com estrutura secundária, principalmente de abordagens gramaticais, a fim de identificar se métodos adaptativos já haviam sido prospectados e se o problema de caracterização de pseudonós ainda estava em aberto;
3. Criar uma versão adaptativa do algoritmo de Earley;
4. Propor uma adaptação da notação BNF¹ (reconhecida pelo GrammarLab) para a descrição de gramáticas adaptativas;
5. Evoluir a estrutura do arcabouço GrammarLab para manipular gramáticas adaptativas e implementar a versão adaptativa do algoritmo de Earley proposto no objetivo específico anterior;
6. Testar a nova versão do GrammarLab na caracterização e reconhecimento de RNAs com pseudonós.

1.2 Organização deste documento

Além desta introdução, este documento é dividido em mais cinco capítulos. No Capítulo 2 são apresentados conceitos fundamentais sobre linguagens formais, a versão inicial do arcabouço GrammarLab, dispositivos adaptativos (já como fruto do cumprimento do objetivo específico 1) e uma breve introdução sobre RNAs. No Capítulo 3 é apresentada uma revisão bibliográfica das técnicas computacionais de classificação de RNAs baseada em estruturas, como resultado do objetivo específico 2. No Capítulo 4 são apresentados os métodos e resultados da evolução do arcabouço, como resultado dos objetivos específicos 3, 4 e 5. No Capítulo 5 é apresentado um estudo de caso do novo GrammarLab na resolução de um problema de caracterização de estrutura secundária de RNAs com pseudonós, como resultado do objetivo específico 6. Por fim, no Capítulo 6 são apresentadas as considerações finais deste trabalho, incluindo trabalhos futuros.

¹ O Formalismo de Backus-Naur (BNF, do inglês Backus-Naur *Form* ou Backus *Normal Form*) é uma metassintaxe usada para expressar gramáticas livres de contexto

2 CONCEITOS FUNDAMENTAIS

Neste capítulo são apresentados os conceitos fundamentais sobre os temas relacionados com o foco de pesquisa desse trabalho. Serão apresentados os conceitos básicos sobre linguagens formais, a primeira versão do arcabouço GrammarLab (anterior a este trabalho), os conceitos sobre dispositivos adaptativos e, por fim, uma visão geral sobre RNAs e suas estruturas.

2.1 Linguagens Formais

Uma **linguagem formal** pode ser entendida como um conjunto finito de símbolos e regras de formação que são aplicadas para que estes símbolos formem sentenças (SHARMA, 2006).

As **gramáticas**, também conhecidas como dispositivos generativos, dispositivos de síntese ou dispositivos de geração de cadeias, são sistemas formais baseados em regras de substituição que podem sintetizar de forma exaustiva o conjunto de cadeias que compõem uma linguagem (RAMOS; NETO; VEGA, 2009).

As gramáticas das linguagens formais são descritas por notações matemáticas rigorosas, evitando assim dúvidas na interpretação.

Formalmente, uma **gramática** G pode ser definida como uma quádrupla $G = (V, \Sigma, P, S)$, na qual:

- V é o conjunto finito e não vazio de símbolos que representam o **vocabulário** da gramática;
- Σ é conjunto finito e não vazio de símbolos que representam o **alfabeto** da gramática, conhecido como símbolos terminais;
- P é o conjunto finito e não vazio de **produções** ou regras de substituição da gramática;
- S é o **símbolo inicial** da gramática, sendo um elemento de $V - \Sigma$.

Além dos elementos apresentados, também é definido $N = V - \Sigma$ como sendo o conjunto de **símbolos não terminais** da gramática. Os símbolos não terminais são símbolos intermediários que participam da estruturação e geração de sentenças, porém não fazem parte das mesmas, ao contrário dos símbolos terminais.

Os elementos do conjunto de produções P obedecem à forma geral $\alpha \rightarrow \beta$, sendo que α é uma cadeia constituída por elementos de V , estando presente pelo menos um símbolo não terminal, e β uma cadeia qualquer, mesmo vazia, de elementos de V . De maneira formal, $P = \{(\alpha, \beta) | (\alpha, \beta) \in V^*NV^* \times V^*\}$.

Forma sentencial é a denominação de uma cadeia $w \in V^*$ obtida pela aplicação recorrente das regras de substituição da gramática. Por definição, o símbolo inicial S é uma forma sentencial. Considerando $\alpha\rho\beta$ uma forma sentencial, sendo $\alpha \in V^*$ e $\beta \in V^*$, e sendo $\rho \rightarrow \gamma$ uma produção da gramática, a aplicação da produção à forma sentencial produz uma nova forma sentencial $\alpha\gamma\beta$.

A aplicação das regras de produção formando novas formas sentenciais é denominada **derivação**. No exemplo anterior temos uma **derivação direta**, que é uma substituição em que apenas uma produção é aplicada e que, formalmente, pode ser representada por $\alpha\rho\beta \Rightarrow_G \alpha\gamma\beta$. O índice G indica que a regra de substituição aplicada pertence ao conjunto de produções que define a gramática G . A aplicação de uma sequência de derivações diretas é representada pela notação $\Rightarrow_G^+$.

Uma cadeia $w \in \Sigma^*$ obtida pela aplicação de derivações iniciando no símbolo inicial S de uma gramática, além de ser uma forma sentencial, é também denominada **sentença**, sendo sua derivação formalmente denotada por $S \Rightarrow_G^+ w$.

O conjunto de todas as sentenças w geradas por uma gramática G é denominado **linguagem gerada** pela gramática G , ou simplesmente $L(G)$, sendo formalmente denotada por $L(G) = \{w \in \Sigma^* | S \Rightarrow_G^+ w\}$.

2.1.1 Hierarquia de Chomsky

A expressividade e os modelos para tratamento variam de linguagem para linguagem. Quanto maior a expressividade da linguagem, mais complexo será o formalismo necessário para o tratamento computacional (RAMOS; NETO; VEGA, 2009).

Chomsky classificou as linguagens e os formalismos que as tratam em quatro classes gramaticais que se relacionam em uma hierarquia (CHOMSKY, 1959). A Figura 1 apresenta a hierarquia de Chomsky organizada em ordem crescente de generalidade e complexidade de reconhecimento.

Figura 1 – Hierarquia de Chomsky

Fonte: MATSUNO (2006)

Gramáticas regulares

As linguagens do tipo 3 da hierarquia de Chomsky são geradas por gramáticas regulares.

Uma gramática é dita **regular** se ela for linear à direita ou linear à esquerda.

Uma gramática é **linear à direita** quando as produções possuem o seguinte formato:

- $A \rightarrow b, A \in N, b \in \Sigma$
- $A \rightarrow bC, A \in N, b \in \Sigma, C \in N$

Uma gramática é **linear à esquerda** quando as produções possuem o seguinte formato:

- $A \rightarrow b, A \in N, b \in \Sigma$
- $A \rightarrow Cb, A \in N, b \in \Sigma, C \in N$

Uma gramática linear à direita que gera a linguagem formada por qualquer cadeia composta por um número ímpar de 1s pode ser formalmente definida por $G = (\{q_1, q_2, 0, 1\}, \{0, 1\}, P, q_1)$, sendo P o conjunto de produções:

- $q_1 \rightarrow 0q_1$
- $q_1 \rightarrow 1q_2$
- $q_1 \rightarrow 1$
- $q_2 \rightarrow 0q_2$

- $q_2 \rightarrow 1q_1$
- $q_2 \rightarrow 0$

Uma linguagem regular é gerada por uma gramática regular e reconhecida por um autômato finito.

Um **autômato finito determinístico** pode ser definido formalmente por $M = (Q, \Sigma, \delta, q_0, F)$ (SIPSER, 2006), sendo que:

- Q é o conjunto finito de **estados** do autômato M ;
- Σ é conjunto finito de símbolos que compõem o **alfabeto** da linguagem;
- δ é a **função de transição** que descreve o conjunto de transições do autômato, sendo $\delta : Q \times \Sigma \longrightarrow Q$;
- q_0 é **estado inicial** do autômato, sendo $q_0 \in Q$;
- F é o conjunto de **estados de aceitação**, sendo $F \subseteq Q$.

Um **autômato finito** é um dispositivo teórico que representa uma máquina de estados, sendo que o estado seguinte do dispositivo é determinado pelo símbolo atual na entrada e pelo estado atual. Após a leitura e o processamento do último símbolo da cadeia de entrada, se o estado final do autômato for um estado de aceitação, a cadeia é dita aceita, caso contrário, a cadeia é rejeitada pois não faz parte da linguagem que o autômato foi desenhado para reconhecer.

Na Figura 2 é apresentado um exemplo de um diagrama de estados de um autômato finito que reconhece uma linguagem formada por qualquer cadeia composta por um número ímpar de 1s, sendo portanto capaz de reconhecer a linguagem gerada pela gramática linear à direita apresentada anteriormente nesta seção. Nesse modo de representação, os estados são representados por círculos, sendo os estados de aceitação representados por círculos duplos, o estado inicial indicado por um círculo que recebe uma seta que não parte de nenhum outro círculo, as setas representando as possíveis transições entre os estados e os símbolos sobre as setas representando o elemento do alfabeto que provoca a transição entre os estados.

Figura 2 – Diagrama de estados de um autômato finito

Fonte: Adaptado de Sipser (2006)

Formalmente, o autômato da Figura 2 é definido por $M = (\{q_1, q_2\}, \{0, 1\}, \delta, q_1, \{q_2\})$, sendo a função de transição δ apresentada na Tabela 1.

Tabela 1 – Função δ de um autômato finito

$Q \times \Sigma$	Q
$(q_1, 0)$	q_1
$(q_1, 1)$	q_2
$(q_2, 0)$	q_2
$(q_2, 1)$	q_1

Fonte: Gilmar Pereira dos Santos, 2018

Um algoritmo que implemente um autômato finito possui complexidade computacional $O(n)$, sendo n o tamanho da cadeia de entrada, uma vez que funciona de maneira linear e sequencial, alterando o estado do autômato de acordo com o estado atual e o símbolo seguinte da cadeia lida na entrada.

Linguagens em que todas as sentenças são palíndromos¹ (ex: $a_1a_1, a_2a_2, a_1a_2a_2a_1, a_2a_1a_1a_2, a_1a_2a_1a_1a_2a_1$ etc.) e linguagens em que todas as sentenças são compostas de duas partes idênticas (ex: $a_1a_1, a_1a_2a_1a_2, a_1a_1a_2a_1a_1a_2$ etc.) são exemplos de linguagens que não podem ser geradas por gramáticas regulares, não sendo portanto linguagens regulares.

Gramáticas livres de contexto

As linguagens do tipo 2 são geradas por gramáticas livres de contexto.

As **gramáticas livres de contexto** possuem produções no formato:

- $A \rightarrow \beta, A \in N, \beta \in V^*$

Um exemplo de gramática livre de contexto é a gramática a seguir que descreve a linguagem $L(G) = \{0^n \# 1^n | n \geq 0\}$:

¹ Palíndromos são sentenças que são iguais quando lidas da esquerda para a direita e da direita para a esquerda.

- $G = (\{A, B, 0, 1, \#\}, \{0, 1, \#\}, P, A)$

Sendo P o conjunto de produções:

- $A \rightarrow 0A1$
- $A \rightarrow B$
- $B \rightarrow \#$

As gramáticas livres de contexto podem gerar todas as linguagens regulares e muitas linguagens adicionais (SIPSER, 2006), como as linguagens formadas de palíndromos. No entanto, assim como as gramáticas regulares, as gramáticas livres de contexto não podem gerar linguagens caracterizadas por partes repetidas.

A sequência de derivação de uma gramática livre de contexto pode ser representada em uma estrutura conhecida como **árvore sintática**. Na Figura 3 é representada a árvore sintática da derivação da cadeia $000\#111$ gerada pela gramática definida anteriormente.

Figura 3 – Exemplo de árvore sintática

Fonte: Sipser (2006)

Quando uma mesma cadeia pode ser representada por mais de uma árvore sintática de uma mesma gramática, dizemos que a gramática é **ambígua**.

Para cada gramática livre de contexto pode ser desenhado um autômato com pilha que reconheça a linguagem gerada por tal gramática (SIPSER, 2006), sendo a complexidade de reconhecimento de ordem polinomial.

Um **autômato com pilha** pode ser definido formalmente por $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ (SIPSER, 2006), sendo:

- Q é o conjunto finito de **estados** do autômato M ;
- Σ é conjunto finito de símbolos que compõem o **alfabeto de entrada**;

- Γ é conjunto finito de símbolos que compõem o **alfabeto da pilha**;
- δ é a **função de transição** que descreve o conjunto de transições do autômato²
 $\delta : Q \times \Sigma_\epsilon \times \Gamma_\epsilon \longrightarrow \mathcal{P}(Q \times \Gamma_\epsilon)$, $\Sigma_\epsilon = \Sigma \cup \{\epsilon\}$, $\Gamma_\epsilon = \Gamma \cup \{\epsilon\}$ e ϵ representando uma cadeia vazia;
- q_0 é estado inicial do autômato, sendo $q_0 \in Q$;
- F é o conjunto de estados de aceitação, sendo $F \subseteq Q$.

Um autômato com pilha tem estruturação e funcionamento semelhante a um autômato finito. Sua constituição difere da de um autômato finito por apresentar uma pilha que, assim como o estado atual do dispositivo e o símbolo presente na entrada, determina o estado seguinte do autômato. A cada passo, de acordo com as regras de transição, o dispositivo pode alterar ou manter o estado atual, ler ou não o próximo símbolo da cadeia de entrada, ler ou não o topo da pilha, adicionar ou não um novo elemento no topo da pilha. O símbolo de cadeia vazia ϵ é utilizado na função de transição para permitir tais “pulos” nas leituras de cadeia de entrada e pilha ou escritas no topo da pilha.

Um autômato com pilha apresenta um comportamento não determinista, isto é, permite múltiplos estados seguintes possíveis. Esse não determinismo pode ser considerado como criação de múltiplas *threads*, cada uma contendo um autômato. Ao fim da leitura do último símbolo da cadeia de entrada, se algum dos autômatos estiver em um estado de aceitação, a cadeia de entrada é aceita como fazendo parte da linguagem definida pelo autômato.

A Figura 4 apresenta um exemplo de um autômato com pilha que reconhece a linguagem $L(G) = \{0^n 1^n | n \geq 0\}$. Neste exemplo, as notações sobre as setas seguem o formato $a, b \rightarrow c$, indicando que ao ser lido o símbolo a da cadeia de entrada, o símbolo b do topo da pilha pode ser substituído pelo símbolo c . Nos casos em que a é ϵ , o autômato pode realizar uma transição de estado sem ler um símbolo da entrada. Nos casos em que b é ϵ , o autômato pode realizar a transição de estado sem ler ou desempilhar um símbolo da pilha. Nos casos em que c é ϵ , o autômato não escreve nenhum símbolo na pilha ao realizar a transição de estado. Como não existe um controle de pilha vazia definido no formalismo de um autômato com pilha, neste exemplo o símbolo $\$$ é utilizado como artifício para que o autômato atinja este objetivo.

² O conjunto potência $\mathcal{P}(x)$ representa todos os subconjuntos do conjunto x . Nesta definição, $\mathcal{P}$ é usado para formalizar o não determinismo do autômato, isto é, seus múltiplos estados seguintes simultâneos possíveis resultantes de uma transição.

Figura 4 – Diagrama de estados de um autômato com pilha

Fonte: Sipser (2006)

Outra alternativa de reconhecimento são os algoritmos analisadores sintáticos de gramáticas, que serão apresentados na subseção 2.1.3.

Gramáticas sensíveis ao contexto

As linguagens do tipo 1 são geradas por gramáticas sensíveis ao contexto.

As **gramáticas sensíveis ao contexto** possuem produções no formato:

- $\alpha \rightarrow \beta, \alpha \in V^*NV^*, \beta \in V^*, |\alpha| \leq |\beta|$, sendo $|x|$ o número de símbolos da cadeia x .

As gramáticas sensíveis ao contexto podem gerar linguagens formadas por sentenças mais complexas que as geradas pelas linguagens livres de contexto, como por exemplo dependências cruzadas como as presentes em linguagens compostas de partes idênticas (cópias) (SEARLS, 1992). Um outro exemplo de linguagem com dependências cruzadas é o conjunto de sentenças formadas por símbolos a , b , c e d tais que $a^n b^m c^n d^m$, n e m inteiros não negativos.

Uma **máquina de Turing** é um dispositivo que basicamente possui uma fita infinita pela qual é feita a leitura da entrada e também na qual pode ser feita a escrita. A cabeça de leitura da fita pode se movimentar para a frente ou para trás, possibilitando ler uma informação escrita previamente. Diferentemente dos autômatos finitos, os estados de aceitação e rejeição fazem efeito assim que encontrados, rejeitando ou aceitando uma cadeia de entrada.

O reconhecimento de uma linguagem sensível ao contexto pode ser feito por uma **máquina de Turing com fita finita**, que é uma variação da máquina de Turing tradicional. O efeito da limitação do tamanho da fita reflete a restrição das gramáticas sensíveis ao contexto, que limita que o lado esquerdo das regras de produção não ultrapasse

o tamanho do lado direito. Isso faz com que haja um limite no número de derivações que serão analisadas, garantindo que haja um fim da computação em algum momento. O problema geral de computação de uma máquina de Turing com fita finita é um problema NP-Completo (DURBIN *et al.*, 1998).

Gramáticas irrestritas

As linguagens do tipo 0, também conhecidas como linguagens recursivamente enumeráveis, são geradas por gramáticas irrestritas.

As **gramáticas irrestritas** possuem produções no formato:

- $\alpha \rightarrow \beta, \alpha \in V^*NV^*, \beta \in V^*, \alpha \neq \epsilon$

As gramáticas irrestritas permitem que ambos os lados das regras de produções possuam quaisquer símbolos, sem a restrição de tamanho que há nas gramáticas sensíveis ao contexto. A única restrição é que haja pelo menos um símbolo não terminal do lado esquerdo de cada produção.

O dispositivo reconhecedor de uma linguagem recursivamente enumerável é uma máquina de Turing com fita infinita. Assim, o número de derivações que pode ser analisada pode crescer sem limites. O reconhecimento dessa classe de linguagem, no caso geral, é um problema indecidível (DURBIN *et al.*, 1998; RAMOS; NETO; VEGA, 2009).

2.1.2 Gramáticas estocásticas

As gramáticas podem também ser utilizadas no contexto de aprendizado estatístico, sendo neste caso utilizadas suas versões estocásticas. Uma gramática estocástica pode ser definida formalmente como uma quádrupla $G = (V, \Sigma, P, S)$, sendo que:

- V, Σ e S possuem os mesmos significados que em gramáticas não estocásticas;
- $P \subset \{\alpha \rightarrow \beta, p\}$, sendo $\alpha \in V^*NV^*$ e $\beta \in V^*$. Assim, a cada produção de P está associada uma probabilidade $p, 0 \leq p \leq 1$;
- Para cada $\alpha \in V^*NV^*$, considerando todas as produções $\{\alpha \rightarrow \beta_i, p_i\} \in P, \beta_i \in V^*$, $\sum_i p_i = 1$.

Uma gramática estocástica, ao invés de determinar se uma dada cadeia pertence à linguagem gerada pela gramática, atribui a ela uma probabilidade de pertencer à linguagem, ou seja, de ser gerada pela gramática. A probabilidade de cada árvore de derivação de uma cadeia consiste no produto das probabilidades de todas as produções utilizadas na derivação da árvore sintática da cadeia. Se a gramática for não ambígua só existe uma árvore de derivação para uma dada cadeia, e a probabilidade desta árvore é então a probabilidade da cadeia pertencer à linguagem. Se a gramática for ambígua, todas as árvores sintáticas da cadeia devem ser consideradas no cálculo desta probabilidade, somando-se as probabilidades dadas por cada árvore. Além disso, para gramáticas ambíguas, também é possível determinar qual a árvore sintática mais provável para uma dada cadeia, isto é, qual o padrão sintático mais provável associado à sua geração.

2.1.3 Analisadores Sintáticos

Os algoritmos analisadores sintáticos de gramáticas diferem dos autômatos a pilha por serem de propósito geral, isto é, não são desenhados para operar em uma linguagem livre de contexto específica.

Os analisadores sintáticos podem ser agrupados em três categorias gerais: algoritmos ascendentes (*bottom-up parsers*), algoritmos descendentes (*top-down parsers*) e algoritmos de uso universal (AHO *et al.*, 1986).

Os algoritmos analisadores descendentes recebem este nome por iniciarem a análise partindo do nó não terminal inicial (topo da árvore de derivação) e irem descendo até os nós terminais. Devido à estratégia adotada para evitar custo excessivo durante a análise (provocado pela necessidade de *backtracking*), estes algoritmos acabam sendo restritos às chamadas gramáticas LL.

Os algoritmos analisadores ascendentes, por sua vez, iniciam a análise partindo dos nós terminais e vão subindo na árvore de derivação. A estratégia utilizada por esta categoria de algoritmos faz com que sejam restritos às chamadas gramáticas LR que são mais gerais que as LL.

Os algoritmos ascendentes e descendentes são amplamente utilizados na construção de compiladores. No entanto, mesmo a gramática LR mais geral ainda é um subconjunto

restrito das gramáticas livres de contexto, fazendo com que os algoritmos ascendentes e descendentes acabem não sendo de uso geral para qualquer tipo de gramática.

Além de não serem de uso geral, os algoritmos ascendentes e descendentes lidam apenas com gramáticas não ambíguas. Isso representa um problema em alguns casos, como por exemplo na análise de sequências biológicas que, por natureza, são ambíguas e de interesse desta pesquisa.

Os algoritmos de uso universal utilizam programação dinâmica para ganhar eficiência no processo de percorrer o espaço de busca durante a procura das diferentes árvores de derivação. Embora não sejam tão eficientes quanto os algoritmos ascendentes e descendentes, possuem a vantagem de poderem lidar com gramáticas ambíguas e conseguirem analisar gramáticas livres de contexto gerais.

Dentre os algoritmos analisadores de gramáticas livres de contexto de uso universal, podemos destacar o algoritmo CYK (YOUNGER, 1967) e o algoritmo de Earley (EARLEY, 1970).

O algoritmo CYK utiliza uma estratégia de análise de baixo para cima (*bottom-up parser*) e programação dinâmica, resolvendo o problema de reconhecimento e determinação das diferentes árvores sintáticas com uma complexidade de tempo $O(n^3)$, sendo n o tamanho da sequência testada. Uma desvantagem do algoritmo CYK é a necessidade da gramática estar representada na forma normal de Chomsky. Uma gramática está na forma normal de Chomsky quando todas as suas regras de produção são de uma destas duas formas:

- $A \rightarrow BC, A \in N, B \in N, C \in N$
- $A \rightarrow \alpha, A \in N, \alpha \in \Sigma$

sendo N o conjunto de símbolos não terminais da gramática e Σ o conjunto finito e não vazio de símbolos que representa o alfabeto, os símbolos terminais, da gramática.

Toda gramática pode ser convertida para a forma normal de Chomsky (SIPSER, 2006). No entanto, essa normalização pode afetar a informação estrutural fornecida pelas árvores sintáticas da gramática original. Esta informação estrutural é particularmente importante, por exemplo, para a caracterização de estruturas secundárias de moléculas de RNA, como descrito ao longo deste trabalho.

2.1.4 Algoritmo de Earley

Diferentemente do algoritmo CYK, o algoritmo de Earley não necessita que a gramática esteja em uma forma específica, eliminando a necessidade de adaptação da gramática ou processamento adicional para normalizar uma gramática que se deseja analisar (EARLEY, 1970). Além disso, o algoritmo de Earley (EARLEY, 1970) apresenta a característica de identificar as diferentes árvores de derivação, possuindo uma complexidade computacional de ordem $O(n^3)$ para gramáticas ambíguas. Isso faz com que seja um algoritmo conveniente, por exemplo, para o uso em problemas de bioinformática relacionados com análise de sequências biológicas, que por natureza são linguagens ambíguas, e que demandam que o formato das regras de produção mimetizem a estrutura de pareamentos entre as bases nucleotídicas (símbolos terminais na gramática). Portanto, é de especial interesse neste trabalho.

Basicamente, o algoritmo de Earley percorre uma dada cadeia de entrada $X_1 \dots X_n$ da esquerda para a direita. Para cada símbolo X_i percorrido, um conjunto S_i de estados é construído, representando a condição do processo de reconhecimento no ponto atual. Cada estado $s \in S_i$ é um item de análise que é representado por uma quintupla, a qual é formada por: (i) uma regra de produção da gramática; (ii) uma posição que indica até que ponto desta regra o reconhecimento foi realizado com sucesso; (iii) o índice i do conjunto de estados S_i em que a regra de produção foi adicionada inicialmente, (iv) uma cadeia, conhecida como *lookahead*, formada pelos próximos k símbolos terminais seguintes à produção (geralmente é considerado apenas um) e (v) uma sequência formada por símbolos terminais e conjuntos de ponteiros para outros estados $s \in S_i$, utilizada para a construção das árvores de derivação.

Cada estado s é utilizado no desenvolvimento da análise sintática por meio da aplicação de uma dentre três possíveis operações do algoritmo: (i) *predictor*, que é a operação na qual são expandidos os símbolos não terminais das regras de produção, adicionando novos estados no conjunto S_i atual; (ii) *scanner*, operação na qual é verificado se o símbolo terminal sendo analisado da regra de produção do estado atual é o símbolo X_i atual, gerando um novo conjunto de estados S_{i+1} e seguindo para o próximo símbolo de entrada X_{i+1} e (iii) *completer*, operação aplicada quando a produção do estado s foi analisada completamente, sendo comparados os k símbolos candidatos sucessores (*lookahead*) aos

símbolos de entrada $X_{i+1}..X_{i+k}$, adicionando ao conjunto S_i atual os estados com as produções nas quais o símbolo completado aparece do lado direito.

Ao finalizar a varredura da cadeia de entrada, o algoritmo identifica se a cadeia pertence ou não à linguagem descrita pela gramática em questão e, caso afirmativo, pode retornar todas as árvores de derivação em uma forma fatorada.

O Algoritmo 1 apresenta os passos básicos do algoritmo reconhecedor, sendo adotadas as seguintes convenções e definições formais:

- As produções da gramática de entrada G são numeradas arbitrariamente de 1 a $d - 1$;
- Cada produção da gramática está na forma $D_p \rightarrow C_{p1}...C_{p\bar{p}}$ ($1 \leq p \leq d - 1$), sendo $\bar{p}$ o número de símbolos do lado direito da p -ésima produção;
- É adicionada a produção $D_0 \rightarrow R \dashv$ à gramática G , sendo R o símbolo não terminal inicial da gramática e $\dashv$ um símbolo terminal que não pertença à gramática G representando o fim da cadeia de entrada;
- Um **estado** é uma quintupla $\langle p, j, f, \alpha, B \rangle$, em que p, j e f são números inteiros, sendo p o número da produção ao qual o estado se refere ($0 \leq p \leq d - 1$), j a posição nesta produção até onde foi realizada a análise ($0 \leq j \leq \bar{p}$), f é o índice que indica o conjunto de estados S_f em que a produção p foi inserida no processo de análise inicialmente ($0 \leq f \leq n + 1$), α uma cadeia composta por k símbolos terminais (*lookahead*) e B é uma sequência formada por símbolos terminais e conjuntos de ponteiros para outros estados $s \in S_i$, utilizada para a construção das árvores de derivação³;
- Um **conjunto de estados** é um conjunto ordenado de *estados*, sendo elementos novos sempre inseridos no fim do conjunto;
- Um **estado final** s_{fim} é um estado em que $j = \bar{p}$;
- $H_k(\gamma) = \{\alpha | \alpha \in T, |\alpha| = k, \text{ e } \exists \beta \text{ tal que } \gamma \Rightarrow^* \alpha\beta, \beta \in T\}$. $H_k(\gamma)$ é o conjunto de todas as cadeias formadas por k símbolos terminais que iniciam uma cadeia derivada de γ . Esta função é utilizada para definir o *lookahead* α de cada estado.

³ Sempre que um novo estado $\langle p, j, f, \alpha, B_1...B_j \rangle$ for adicionado a um conjunto S_i e $s_n = \langle p, j, f, \alpha, B'_1...B'_j \rangle \in S_i$, então s_n deve ser substituído de forma que $s_n = \langle p, j, f, \alpha, (B_1 \cup B'_1)...(B_j \cup B'_j) \rangle$. Esse fato é omitido na descrição do algoritmo

Algoritmo 1 Algoritmo de Earley para reconhecimento de gramáticas livres de contexto

```

1: function EARLEYPARSER( $G, X_1 \dots X_n, k$ )
2:    $X_{n+i} = \perp$  ( $1 \leq i \leq k+1$ )
3:    $S_i = \emptyset$  ( $0 \leq i \leq n+1$ )
4:    $S_0 = \{ \langle 0, 0, 0, \perp^k, \emptyset \rangle \}$ 
5:   for  $i \leftarrow 0$  até  $n$  do
6:     for all  $s = \langle p, j, f, \alpha, B \rangle \in S_i$  do
7:       if  $s \langle \rangle s_{fim}$  e  $C_{p(j+1)} \in N$  then ▷ 1 - Predictor
8:         for all  $q \mid C_{p(j+1)} = D_q$  do
9:           for all  $\beta \in H_k(C_{p(j+2)} \dots C_{p\bar{p}}\alpha)$  do
10:             $S_i \leftarrow S_i + \{ \langle q, 0, i, \beta, \emptyset \rangle \}$ 
11:       if  $s = s_{fim}$  e  $\alpha = X_{i+1} \dots X_{i+k}$  then ▷ 2 - Completer
12:         for all  $\langle q, l, g, \beta, E \rangle \in S_f$  do
13:           if  $C_{q(l+1)} = D_p$  then
14:              $E \leftarrow B_1 \dots B_l$ 
15:              $E_{l+1} \leftarrow$  ponteiro para  $\langle p, \bar{p}, f, \alpha, B \rangle$ 
16:              $S_i \leftarrow S_i + \{ \langle q, l+1, g, \beta, E \rangle \}$ 
17:       if  $s \langle \rangle s_{fim}$  e  $C_{p(j+1)} \in T$  then ▷ 3 - Scanner
18:         if  $C_{p(j+1)} = X_{i+1}$  then
19:            $E \leftarrow B_1 \dots B_j$ 
20:            $E_{j+1} \leftarrow X_{i+1}$ 
21:            $S_{i+1} \leftarrow S_{i+1} + \{ \langle p, j+1, f, \alpha, E \rangle \}$ 
22:       if  $S_{i+1} = \emptyset$  then
23:         retornar Rejeição
24:       if  $i = n$  e  $S_{i+1} = \{ \langle 0, 2, 0, \perp \rangle \}$  then
25:         retornar Aceitação

```

 Fonte: Adaptado de Earley (1970)

2.1.5 GrammarLab: Laboratório de geração de classificadores de sequências baseados em gramáticas

O GrammarLab (MACHADO-LIMA, 2002) é um arcabouço desenvolvido em C++ com o objetivo de facilitar a implementação e geração de classificadores baseados em gramáticas, facilitando a pesquisa da aplicação de vários algoritmos de aprendizado de gramáticas estocásticas na modelagem, por exemplo, de sequências biológicas.

Na Figura 5 é apresentado o diagrama geral de funcionamento do processo utilizado no arcabouço para geração dos classificadores. Ele é composto por um módulo de aprendizado gramatical que, a partir de n conjuntos de sequências de treinamento, cada um representando uma classe, aprende uma gramática estocástica e gera um analisador sintático para cada um desses conjuntos. O aprendizado pode ser realizado em um ou em dois passos, sendo esses dois passos o de inferência das regras gramaticais e o de

estimação de probabilidades. Para a geração do analisador sintático é utilizado o algoritmo de Earley. Por questões de eficiência na execução, embora o algoritmo de Earley pudesse ser implementado de forma a ser independente da gramática sendo analisada, no arcabouço desenvolvido anteriormente a este trabalho foi adotada a estratégia de geração de um código compilado específico para uma determinada gramática. Por fim, é gerado um classificador multiclasse, considerando que cada gramática gerada irá representar uma classe distinta. Para uma determinada sequência de entrada, o classificador utiliza como regra de decisão classificar a sequência como pertencente à classe representada pela gramática que atribuir maior probabilidade a ela.

Figura 5 – Diagrama de funcionamento do GrammarLab

Fonte: Machado-Lima (2002)

É importante ressaltar que todo o funcionamento dos classificadores gerados utilizando o arcabouço é estático, uma vez que são gerados códigos compiláveis específicos para cada gramática e respectivos analisadores e classificadores.

O arcabouço desenvolvido é composto de três partes:

- Algoritmos de inferências gramatical e estimação de probabilidades;
- Suporte de implementação;
- Suporte de testes.

A primeira parte é constituída de classes abstratas de inferidores gramaticais e de estimadores de probabilidades. Fazem parte do arcabouço também as implementações de dois inferidores gramaticais (MÄKINEN, 1992; SAKAKIBARA, 1992) e de dois estimadores de probabilidade (FUYAMA, 1982; SAKAKIBARA *et al.*, 1994c).

A segunda parte é constituída de um conjunto de estruturas comuns encontradas nos algoritmos de inferidores e estimadores encontrados na literatura. Nesse módulo são encontradas estruturas que dão suporte à manipulação de gramáticas, autômatos a árvore, e uma estrutura conhecida como Trie. Além disso, nesse módulo existem conjuntos de classes que modelam *streams* de entrada e saída, possibilitando um canal de comunicação entre

os algoritmos. Também são encontradas classes para geração de analisadores sintáticos e um mecanismo integrador de todos os analisadores das gramáticas consideradas na classificação. O algoritmo de Earley (EARLEY, 1970) foi escolhido como implementação para o analisador sintático de gramáticas livres de contexto estocásticas no arcabouço pelo fato dele fornecer todas as árvores de derivação e não exigir nenhuma normalização da gramática.

A terceira parte é formada por programas Perl e C++ que tratam a geração dos classificadores, execução de testes e obtenção de resultados.

Vale ressaltar neste ponto que a estratégia de geração de códigos fontes (extensões .cpp e .h) para cada gramática inferida pelos algoritmos de aprendizado e seus analisadores sintáticos otimiza o tempo de execução, porém faz com que os analisadores possuam um comportamento totalmente estático, o que é o contrário do esperado em dispositivos adaptativos, como descrito na próxima seção.

2.1.6 Dispositivos adaptativos

Dispositivos adaptativos são dispositivos formais que podem ter seu comportamento alterado de forma dinâmica como resposta espontânea a estímulos de entrada (NETO, 2001).

Quaisquer alterações possíveis no comportamento de um dispositivo adaptativo devem ser conhecidas *a priori*. Assim, esses dispositivos são capazes de detectar as situações que disparam as modificações e devem ser auto-modificáveis para reagir de forma adequada, se adaptando à situação.

Um dispositivo adaptativo é formado pela incorporação de ações adaptativas às regras de um dispositivo não adaptativo subjacente. Assim, sempre que alguma dessas regras é aplicada, a ação adaptativa correspondente é acionada. Dessa forma, o dispositivo adaptativo resultante pode ser facilmente compreendido por todos que tenham familiaridade com o dispositivo subjacente (RAMOS; NETO; VEGA, 2009).

O uso de métodos adaptativos possibilita alterar dinamicamente as propriedades de uma gramática, viabilizando inserir sensibilidade ao contexto em uma gramática originalmente livre de contexto sem, com isso, aumentar excessivamente sua complexidade de análise (IWAI, 2000). Isto é possível pois as regiões dependentes de contexto de uma

cadeia podem ser confinadas em regiões controladas. Ou seja, é realizada uma separação da cadeia em regiões localmente regulares ou livres de contexto, regiões estas posteriormente integradas em uma análise global dependente de contexto. Tal separação de análise pode resultar em uma economia considerável de tempo de reconhecimento. Em (IWAI, 2000) é apresentado um formalismo para uma gramática adaptativa (apresentado a seguir) e é estabelecida uma equivalência com autômatos adaptativos (NETO, 1994).

Gramáticas adaptativas

Uma gramática adaptativa é um dispositivo adaptativo, conforme descrito em (IWAI, 2000), que é capaz de representar linguagens sensíveis ao contexto, sendo diferenciada das gramáticas tradicionais por possuir a capacidade de alterar seu conjunto de regras de produção e seu conjunto de símbolos não terminais durante a geração das sentenças.

Neste trabalho será considerado como dispositivo subjacente da gramática adaptativa uma gramática livre de contexto, de forma a reduzir, no geral, a complexidade de análise se comparada à complexidade de análise de gramáticas sensíveis ao contexto.

Uma gramática adaptativa é representada formalmente por $G_A = (G^0, T, R^0)$, na qual:

- G^0 é a gramática livre de contexto inicial;
- T é um conjunto finito, possivelmente vazio, de funções adaptativas, que podem ser descritas por conjuntos de ações adaptativas elementares, que quando forem aplicadas, devem ser obrigatoriamente executadas em uma sequência padronizada (NETO, 2001);
- R^0 é a relação entre as regras de produção da gramática G^0 e as funções adaptativas, sendo $R^0 \subseteq BA \times P^0 \times AA$, sendo:
 - P^0 o conjunto de produções da gramática G^0 ;
 - $\{\epsilon\}$ o conjunto que tem como único elemento ϵ , que representa uma função adaptativa nula;
 - $BA \subseteq (T \cup \{\epsilon\})$ é o conjunto de funções adaptativas executadas antes do acionamento da respectiva regra de produção $p \in P^0$;
 - $AA \subseteq (T \cup \{\epsilon\})$ é o conjunto de funções adaptativas executadas após o acionamento da respectiva regra de produção $p \in P^0$.

Durante a geração de uma sentença por uma gramática adaptativa, sempre que uma função adaptativa é ativada, uma nova gramática livre de contexto G^i é criada. Assim, uma sentença qualquer pertencente à linguagem definida por uma gramática adaptativa é gerada pela sequência de gramáticas $G^0, \dots, G^n$.

Cada gramática G^i criada pela ativação de uma função adaptativa pode apresentar um novo conjunto de símbolos não terminais N^i , novos conjuntos de regras de produções P^i e relações R^i entre as regras de produção e as funções adaptativas T , sendo i o indicador da quantidade de gramáticas criadas.

As regras de produção de uma gramática adaptativa G^i podem possuir um dos seguintes formatos:

- $A \rightarrow \alpha$, sendo $A \in N^i$ e $\alpha \in V^*$ – regra de produção em que nenhuma função adaptativa está associada;
- $A \rightarrow \{ba\}\alpha$, sendo $A \in N^i$, $ba \in T$ e $\alpha \in V^*$ – regra de produção com uma função adaptativa ba que é ativada antes da regra de produção;
- $A \rightarrow \alpha\{aa\}$, sendo $A \in N^i$, $\alpha \in V^*$ e $aa \in T$ – regra de produção com uma função adaptativa aa que é ativada após a regra de produção;
- $A \rightarrow \{ba\}\alpha\{aa\}$, sendo $A \in N^i$, $ba \in T$, $\alpha \in V^*$ e $aa \in T$ – regra de produção com uma função adaptativa ba que é ativada antes e com uma função adaptativa aa que é ativada após a regra de produção;
- $A \rightarrow \emptyset$, sendo $A \in N^i$ – regra de produção utilizada para declaração de um símbolo não terminal A que será definido posteriormente pela ativação de alguma função adaptativa.

Funções adaptativas

Uma função adaptativa é uma abstração genérica que define um determinado comportamento adaptativo, sendo um conjunto de ações adaptativas elementares.

Uma função adaptativa pode ser declarada da seguinte forma:

```
Nome da função(lista de parâmetros formais) = {
    lista de variáveis, lista de geradores (identificados pelo símbolo *) :
    função adaptativa opcional ao início
```

```

ação adaptativa elementar 1
...
ação adaptativa elementar n
função adaptativa opcional ao fim
}

```

Após o nome da função, entre parênteses, é informada uma lista de parâmetros formais separados por vírgulas. Os parâmetros são nomes simbólicos passados como argumentos para a função no momento de sua chamada.

As variáveis (N_{var}) são nomes simbólicos utilizados para armazenar valores resultantes de ações adaptativas elementares de pesquisa de regras (mencionadas à frente).

Os geradores (N_{ger}) são nomes simbólicos semelhantes às variáveis, porém têm seu valor atribuído automaticamente e de forma única no início da execução da função adaptativa. Para diferenciar das variáveis, na declaração os geradores são precedidos pelo símbolo *. Os geradores são utilizados para incluir novos símbolos não terminais na gramática, evoluindo o vocabulário.

Opcionalmente, pode haver uma chamada a uma função adaptativa que será executada antes das ações elementares definidas nesta função e a uma função adaptativa que será executada ao fim da execução.

Existem três tipos de ações adaptativas elementares:

- $?[A \rightarrow \{ba\}M^*\{aa\}]$, sendo $A \in N_{var} \cup N^i$, $M \in V^i \cup N_{var}$, $ba \in T \cup \{\epsilon\}$ e $aa \in T \cup \{\epsilon\}$ – Ações adaptativas elementares de **pesquisa** de regras, que são as ações que não modificam nenhuma regra, permitindo a inspeção das regras atuais em busca das que satisfaçam determinado padrão e preenchimento das variáveis N_{var} . A pesquisa por padrões é feita comparando todos os símbolos tanto do lado esquerdo quanto do lado direito da produção, sendo N_{var} preenchida com o símbolo que completa o padrão sendo comparado;
- $-[A \rightarrow \{ba\}M^*\{aa\}]$, sendo $A \in N_{var} \cup N^i$, $M \in V^i \cup N_{var}$, $ba \in T \cup \{\epsilon\}$ e $aa \in T \cup \{\epsilon\}$ – Ações adaptativas elementares de **eliminação** de regras, que são as ações que removem do conjunto de produções as regras que satisfazem a um determinado padrão;
- $+ [A \rightarrow \{ba\}M^*\{aa\}]$, sendo $A \in N_{var} \cup N_{ger} \cup N^i$, $M \in V^i \cup N_{var} \cup N_{ger}$, $ba \in T \cup \{\epsilon\}$ e $aa \in T \cup \{\epsilon\}$ – Ações adaptativas elementares de **inserção** de regras, que permitem

acrescentar uma regra específica ao conjunto de regras de produção. É importante notar aqui a possibilidade do uso de geradores para aumentar o vocabulário da gramática, adicionando um novo símbolo não terminal.

Existe uma ordem de precedência para a execução das ações adaptativas elementares, não importando sua ordem na declaração da função adaptativa: em primeiro lugar são executadas as pesquisas, na sequência são executadas as eliminações de regras e por fim as ações de inserção. As ações de mesma precedência, estas sim serão executadas respeitando a ordem de declaração na função.

Em (IWAI, 2000) são apresentados exemplos detalhados de gramáticas adaptativas, como a gramática para a linguagem $L(G) = \{a^n b^n c^n | n \geq 0\}$, que é um modelo clássico de uma linguagem sensível ao contexto.

Também em (IWAI, 2000) é apresentado um breve estudo informal da complexidade da gramática adaptativa que indica que o crescimento da gramática, isto é, a quantidade de regras de produção, no pior caso teórico, é linearmente proporcional ao tamanho da sequência e que o custo computacional para as substituições é de ordem quadrática, sendo portanto um modelo viável computacionalmente.

Os conceitos apresentados podem ser generalizados para outras classes de formalismos, possibilitando a criação de dispositivos adaptativos para diferentes formalismos subjacentes.

As definições para um autômato adaptativo são semelhantes às apresentadas para as gramáticas adaptativas. As ações adaptativas operam sobre o conjunto de transições de um autômato adaptativo, alterando sua topologia de forma análoga à alteração efetuada nas regras de produção de uma gramática adaptativa. No entanto, este trabalho tem como foco as gramáticas adaptativas, já que o objetivo principal é inserir adaptatividade no arcabouço GrammarLab, que é baseado em gramáticas.

Exemplo de Gramática Adaptativa

O seguinte exemplo, utilizando formalismo adaptativo (IWAI, 2000) (NETO, 2001), apresenta uma gramática simplificada que representa a linguagem $L(G) = \{a^n b^m c^n d^m\}$, que representa uma dependência cruzada na qual há uma relação entre os símbolos terminais a e c e entre os símbolos b e d .

$$G_A = (G^0, T, R^0)$$

$$G^0 = \{V^0, \Sigma, P^0, S\}$$

$$V^0 = \{S, K, a, b, c, d\}$$

$$\Sigma = \{a, b, c, d\}$$

$$P^0 = \{ \\ S \rightarrow K\{AdP(K)\} \\ K \rightarrow \phi \\ \}$$

$$T = \{ \\ AdP(X) = \{A'*, B'*, C'*, D'* : \\ +[X \rightarrow A'B'C'D'] \\ +[A' \rightarrow aA'\{Ad1(A', C', a, c)\}] \\ +[A' \rightarrow a\{Ad2(C', c)\}] \\ +[B' \rightarrow bB'\{Ad1(B', D', b, d)\}] \\ +[B' \rightarrow b\{Ad2(D', d)\}] \\ +[C' \rightarrow \phi] \\ +[D' \rightarrow \phi] \\ \}$$

$$Ad1(X, Y, x, y) = \{Y'* : \\ -[X \rightarrow xX\{Ad1(X, Y, x, y)\}] \\ -[X \rightarrow x\{Ad2(Y, y)\}] \\ +[X \rightarrow xN\{Ad1(X, Y', x, y)\}] \\ +[X \rightarrow x\{Ad2(Y', y)\}] \\ +[Y \rightarrow yY'] \\ +[Y' \rightarrow \phi] \\ \}$$

$$Ad2(Y, y) = \{ \\ +[Y \rightarrow y] \\ \}$$

$$\}$$

Neste exemplo, a regra de produção inicial $S \rightarrow K$ possui uma associação com uma função adaptativa $\{AdP(K)\}$, sendo que a produção $K \rightarrow \phi$ indica que o símbolo não terminal K será definido de forma dinâmica, pelo acionamento de alguma ação adaptativa. A função adaptativa $AdP(X)$, ao ser acionada, basicamente indica o início de um domínio de dependência cruzada desmembrado em quatro partes, sendo A^i relacionado com C^i e B^i relacionado com D^i . Cada vez que uma regra de produção associada com A^i ou B^i é aplicada, uma nova regra associada a C^i e/ou D^i é adicionada ao conjunto de regras de produção, forçando a dependência cruzada.

Abaixo, uma derivação de uma dependência cruzada utilizando a gramática adaptativa de exemplo, e na Figura 6 o destaque para a dependência apresentada.

$$\begin{aligned}
S &\Rightarrow_{G^0} K \\
&\Rightarrow_{G^1} A^1 B^1 C^1 D^1 \\
&\Rightarrow_{G^1} a A^1 B^1 C^1 D^1 \\
&\Rightarrow_{G^2} aa B^1 C^1 D^1 \\
&\Rightarrow_{G^3} aab B^1 C^1 D^1 \\
&\Rightarrow_{G^4} aabb B^1 C^1 D^1 \\
&\Rightarrow_{G^5} aabbb C^1 D^1 \\
&\Rightarrow_{G^6} aabbbc C^2 D^1 \\
&\Rightarrow_{G^6} aabbbcc D^1 \\
&\Rightarrow_{G^6} aabbbccd D^2 \\
&\Rightarrow_{G^6} aabbbccdd D^3 \\
&\Rightarrow_{G^6} aabbbccddd
\end{aligned}$$

Figura 6 – Dependência cruzada.

Fonte: Gilmar Pereira dos Santos, 2018

2.2 RNA

O ácido ribonucleico (RNA) é uma macromolécula que pode ser encontrada no núcleo ou espalhada por todo o citoplasma da célula. O RNA é constituído de nucleotídeos, que são polímeros formados por uma molécula de açúcar (ribose), um fosfato e uma base nitrogenada. As bases nitrogenadas se dividem em dois grupos: púricas (guanina e adenina) e pirimídicas (citosina e uracila) (JUNQUEIRA; CARNEIRO, 2015).

Diferentemente do ácido desoxirribonucleico (DNA) que tem a estrutura clássica conhecida de uma fita dupla (WATSON; CRICK, 2003), a maioria das moléculas de RNA são formadas por um filamento simples de nucleotídeos encadeados (CLANCY *et al.*, 2008), conforme visto na Figura 7.

Figura 7 – Estrutura de DNA x RNA

Fonte: Clancy *et al.* (2008)

Além dos RNAs mensageiros, transportadores e ribossomais, que possuem participação essencial no processo de síntese de proteínas (transcrição e tradução) (CLANCY; BROWN, 2008), nas últimas décadas foram descobertas várias outras famílias de RNAs não codificadores de proteínas que desempenham diversas funções celulares (MATTICK; MAKUNIN, 2006). Esses RNAs não codificantes também estão relacionados com várias doenças como câncer (COSTA, 2005; REIS *et al.*, 2004; REIS *et al.*, 2005), problemas cardíacos (ISHII *et al.*, 2006), Alzheimer (LUKIW *et al.*, 1992), esquizofrenia (MILLAR *et al.*, 2000; POLESSKAYA *et al.*, 2003) e doenças neurodegenerativas (MATTICK; MAKUNIN, 2006).

2.2.1 Estruturas primária, secundária e terciária

As sequências que formam as macromoléculas de RNA são representadas por caracteres que representam as bases nitrogenadas que formam os nucleotídeos, da mesma forma que ocorre com a representação tradicional das sequências de DNA. Assim, o alfabeto utilizado para representar as sequências de RNA é formado pelas letras A (adenina), G (guanina), C (citosina) e U (uracila). A fita de RNA possui duas extremidades livres, uma chamada de 3' e outra de 5', em uma referência aos átomos de carbono que ficam livres no açúcar que compõe cada nucleotídeo. Por convenção, a sequência é lida no sentido de 5' para 3', que é o sentido da polimerização da molécula.

As sequências que formam os RNAs (também chamadas de estruturas primárias) podem não se apresentar muito conservadas (similares) entre diferentes espécies. No entanto, o filamento de RNA pode se dobrar por meio do pareamento de bases complementares de nucleotídeos, formando estruturas secundárias (Figura 8). Os pareamentos geralmente ocorrem entre as bases G-C, A-U e, com menos frequência, G-U. Os elementos da estrutura secundária interagem entre si, formando estruturas terciárias ou tridimensionais complexas (Figura 8).

Figura 8 – Estrutura secundária x estrutura tridimensional de um RNA

Fonte: Adaptado de Clancy *et al.* (2008)

Frequentemente as estruturas estão mais relacionadas com a função que a molécula de RNA desempenha do que a sequência (NOVIKOVA; HENNELLY; SANBONMATSU, 2012a; DIXON; HILLIS, 1993; LANGE *et al.*, 2012; SEEMANN *et al.*, 2012; NOVIKOVA; HENNELLY; SANBONMATSU, 2012b). Para viabilizar estudos, geralmente são utilizadas

as estruturas secundárias, por serem mais simples de ser computadas e preditas do que as estruturas tridimensionais.

Uma estrutura secundária de um RNA pode ser decomposta em diferentes componentes estruturais (MACHADO-LIMA; PORTILLO; DURHAM, 2008), conforme visto na Figura 9:

Figura 9 – Elementos de uma estrutura secundária de RNA

Fonte: Adaptado de Machado-Lima, Portillo e Durham (2008)

- **Hélice ou *stem*:** empilhamento de bases pareadas;
- **Loop:** região de bases não pareadas;
- **Hairpin loop:** região não pareada no término de uma hélice;
- **Multi-loop:** região de *loop* da qual partem mais de duas hélices;
- **Loop interno simétrico:** um *loop* que ocorre dentro de uma hélice, sendo que a região não pareada em ambos os lados da hélice possui a mesma quantidade de nucleotídeos;
- **Loop interno assimétrico:** um *loop* que ocorre dentro de uma hélice, sendo que a região não pareada em ambos os lados da hélice possui quantidades diferentes de nucleotídeos;
- **Bojo:** um loop dentro de apenas um lado de uma hélice.

Esses componentes podem participar de relações entre si, formando outros componentes estruturais que, por isso, são por vezes considerados como já pertencentes à estrutura terciária da molécula:

- **Triplas de bases:** interações envolvendo três bases;
- **Pseudonó:** hélices que se cruzam. Existem cinco tipos conhecidos de cruzamentos que formam pseudonós (Figura 10) (WASHIETL *et al.*, 2012):
 - *H-Type*, que é o tipo de pseudonó mais simples, formado pelo cruzamento de duas hélices (Figura 10A);
 - *Three-chain* ou *kissing hairpin*, um pseudonó formado pela conexão de dois *hairpin loops* por um ou mais pares de bases (Figura 10B);
 - *Three-knot*, que é formado pelo cruzamento de três hélices entre si (Figura 10C);
 - *Closed four-chain*, uma estrutura complexa formada por um encadeamento quádruplo fechado por uma quinta hélice (Figura 10D);
 - *Canonical pseudoknot*, que é formado por duas hélices compostas apenas de pares de bases canônicas (A-U e G-C), não possuindo *loops* internos ou bojos e estendidos ao máximo (não podendo ser estendidos por pares de bases canônicas) (Figura 10E).

Figura 10 – Tipos de pseudonós

Fonte: Washietl *et al.* (2012)

2.2.2 Famílias funcionais

Em RNAs cuja função está relacionada com sua estrutura, normalmente sua estrutura é mais conservada filogeneticamente (entre espécies relacionadas evolutivamente) do que sua sequência. Desta forma, busca por similaridade de estrutura e sequência são úteis na classificação de RNAs em famílias funcionais.

Utilizando estruturas curadas (estruturas verificadas por especialistas) encontradas em diferentes bases de dados e um modelo computacional para automatizar a identificação de novos elementos homólogos ⁴, a base de dados RFAM (GRIFFITHS-JONES *et al.*, 2003) disponibiliza publicamente um repositório de RNAs agrupados em famílias funcionais anotadas.

Como base para realizar a classificação das famílias, o RFAM realiza um alinhamento estrutural entre as sequências curadas no qual é considerada a covariação das bases dentro das hélices (ou seja, a substituição de pares A-U e G-C). Desse alinhamento no nível de sequências e estruturas, é definida uma estrutura consenso, a qual é utilizada para a caracterização da família.

Na Figura 11 é apresentado um exemplo de uma alinhamento estrutural da família RF01380 do RFAM.

Figura 11 – Alinhamento estrutural - formato stockholm utilizado no RFAM

```
EF545108.1/38-56  GGCGACYGGUGAGUACGCC
S69386.1/283-301  GGCGACUGGUGAGUACGCC
K03455.1/736-754  GGCGACUGGUGAGUACGCC
M17449.1/732-750  GGCGACUGGUGAGUACGCC
M38429.1/737-755  GGCGACUGGUGAGUACGCC
U63632.1/59-77    GGCGACUGGUGAGUACGCC
M38431.1/736-754  GGCGACUGGUGAGUACGCC
U26942.1/114-132  GGCGACUGGUGAGUACGCC
M13136.1/734-752  GGCGACUGGUGAGUACGCC
M17451.1/250-268  GGCGAACGGUGAGUACGCC
K03454.1/282-300  AGCGACUGGUGAGUACGCU
M22639.1/736-754  AGCGACCGGUGAGUACGCU
M62320.1/201-219  GGCGACUGGUGAGUACGCC
AY118157.1/282-300 GGCGACUGGUGAGUACGCA
AF286227.1/83-101 GGCGACUGGUGAGUACGCU
AY451080.1/30-48  AGCGACUGGUGAGUACGCC
AF190128.1/57-75  CGCGACUGGUGAGUACGCC
X52154.1/752-770  GGCGACUGGUGAGUACGCA
DQ373063.1/283-301 GGCGACUGGUGAGUACGCC
AY169968.1/277-295 GGCGACUGGUGAGUACGCA
DQ373064.1/281-299 GGCGACUGGUGAGUACGCC
AF465242.1/809-827 AGCGACCGGUGAGUACGCU
#=GC SS_cons      <<<<<<____>>>->>>
#=GC RF           GGCGACUGGUGAGUACGCC
```

Fonte: <<http://rfam.xfam.org>>

⁴ O termo “homologia” deriva do grego *homos* (igual) e *logos* (relação) e refere-se à relação existente entre duas estruturas, de espécies diferentes, que partilham um ancestral comum

Neste alinhamento, a linha SS_cons representa a estrutura secundária consenso resultante do alinhamento estrutural, no padrão WUSS (*Washington University Secondary Structure notation*). Neste padrão, temos as seguintes notações (EDDY, 2003):

- **Bases pareadas:** são representadas pelos pares de símbolos <>, (), [] e {};
- ***Hairpin loop*:** os nucleotídeos não pareados são representados pelo símbolo de sublinhado (-);
- **Bojo e *loops* internos:** os nucleotídeos são representados por traços (-);
- ***Multi-loops*:** os nucleotídeos residuais são representados por vírgulas;
- **Resíduos externos:** os nucleotídeos residuais que não fazem parte de nenhuma estrutura, ficando nas extremidades, são representados pelo símbolo de dois pontos;
- **Pseudonós:** os pares de bases que representam pseudonós são representados por pares de letras maiúsculas e minúsculas. Exemplo: <<<<AAAA----->>>>aaaa.

Devido às diversidades entre os organismos, algumas famílias possuem integrantes muito divergentes. Um exemplo de diversidade entre organismos é a RNase P de *Plasmodium vivax*, apresentada em (PICCINELLI; ROSENBLAD; SAMUELSSON, 2005) que, em contraste com a RNase P encontrada na *Entamoeba histolytica*, é uma estrutura de hélices bem mais alongadas, conforme Figura 12.

Figura 12 – Comparação entre RNaseP do *Plasmodium vivax* e da *Entamoeba histolytica*

Fonte: Piccinelli, Rosenblad e Samuelsson (2005)

Outro exemplo é encontrado em (CHEN; GREIDER, 2004) no qual é apresentada a divergência entre as estruturas do componente RNA da telomerase de ciliados, vertebrados e leveduras, conforme Figura 13. Apesar da aparente divergência, todas apresentam uma organização estrutural semelhante.

Figura 13 – Estrutura secundária do componente RNA da telomerase de ciliatos, vertebrados e leveduras

Fonte: Chen e Greider (2004)

2.2.3 Modelagem de RNAs com gramáticas

Como já mencionado, as moléculas de RNA são compostas de sequências compostas pelos nucleotídeos adenina (a), guanina (g), citosina (c) e uracila (u). Utilizando a teoria de linguagens formais, podemos considerar que a linguagem do RNA é formada por sentenças cujo alfabeto é composto pelos símbolos a, g, c e u: $\Sigma = \{a, g, c, u\}$.

Gramáticas regulares podem ser utilizadas para modelagem de sequências simples de RNAs, fornecendo uma visão simplista das moléculas. Para a modelagem considerando estrutura secundária, no entanto, é necessária uma gramática que represente dependências entre bases pareadas, sendo que estas dependências apresentam distâncias arbitrárias. As gramáticas livres de contexto podem representar estas dependências, o que não é possível com as gramáticas regulares. Assim, a árvore de derivação de uma cadeia de RNA

segundo uma gramática livre de contexto pode descrever a estrutura secundária desta cadeia (Figura 14).

Figura 14 – Estrutura secundária de um RNA e árvore de derivação para uma gramática simples

Fonte: Adaptado de Machado-Lima (2002)

Observando apenas a sequência linear de um RNA podemos notar várias possibilidades de dobramentos, formando diferentes estruturas secundárias possíveis. Assim, temos que uma gramática que descreve uma molécula de RNA necessita ser ambígua. Para lidar com as incertezas de estruturas e também para tratar possíveis mutações encontradas na natureza (inserções, deleções ou substituições de nucleotídeos específicos) é necessário o uso de gramáticas estocásticas.

A maioria das estruturas secundárias de RNAs encontradas na natureza podem ser representadas por gramáticas livres de contexto, porém algumas estruturas encontradas nestas moléculas necessitam de gramáticas mais complexas. Um exemplo são os pseudonós, que são caracterizados por dependências cruzadas (SEARLS, 1997). Dependências cruzadas podem ser descritas por gramáticas sensíveis ao contexto (SEARLS, 1992). O grande problema das linguagens sensíveis ao contexto é a complexidade da análise sintática envolvida, sendo um problema NP-completo (BROWN; WILSON, 1995; SEARLS, 1997; RIVAS; EDDY, 2000).

3 CARACTERIZAÇÃO COMPUTACIONAL DE SEQUÊNCIAS DE RNAs - UMA REVISÃO BIBLIOGRÁFICA

Nesse capítulo será apresentada uma visão geral dos métodos utilizados para caracterização de sequências de RNA com estrutura secundária, permitindo a busca de novas sequências.

Um RNA pode ser classificado em uma determinada família por meio da comparação entre sua estrutura secundária e as estruturas consenso conhecidas de diversas famílias. Assim, a comparação entre a estrutura de um RNA, ou entre a estrutura consenso de uma família, e uma base de assinaturas estruturais anotadas funcionalmente pode ser utilizada para inferir a funcionalidade desta molécula de RNA ou desta família. (MACHADO-LIMA, 2006; MACHADO-LIMA; PORTILLO; DURHAM, 2008).

3.1 *Abordagens gramaticais*

3.1.1 Gramáticas desenhadas manualmente para caracterização de sequência e estrutura secundária

Sakakibara e seus colaboradores utilizaram em seus trabalhos gramáticas livres de contexto estocásticas para caracterizar famílias de RNAs (SAKAKIBARA *et al.*, 1993; SAKAKIBARA *et al.*, 1994a; SAKAKIBARA *et al.*, 1994b; SAKAKIBARA *et al.*, 1994c). Nestes trabalhos os autores basearam-se no conhecimento *a priori* sobre a estrutura secundária da família de interesse para desenhar manualmente a parte não estocástica da gramática e também desenvolveram um algoritmo baseado em árvores (Tree-Grammar EM) para o cálculo dos parâmetros probabilísticos da gramática, tornando-a estocástica.

Os autores utilizaram para modelagem de estruturas secundárias de RNAs, produções nos seguintes formatos: $S \rightarrow SS$, $S \rightarrow aSa$, $S \rightarrow aS$, $S \rightarrow S$ e $S \rightarrow a$, sendo S algum símbolo não terminal e a um dos símbolos terminais A, U, G ou C que representa algum dos nucleotídeos adenina, uracila, guanina ou citosina. Produções do tipo $S \rightarrow SS$ são utilizadas na modelagem de bifurcações. Produções do tipo $S \rightarrow aSa$ são utilizadas na emissão de pares de bases que formam hélices. Para a modelagem de regiões não pareadas são utilizadas produções do tipo $S \rightarrow aS$ e $S \rightarrow a$. Por fim, produções do tipo $S \rightarrow S$ são utilizadas no contexto de alinhamentos múltiplos para deleção de um nucleotídeo em uma posição específica.

No exemplo da Figura 15 é apresentado um conjunto de regras de produção e uma derivação de uma sequência utilizando um modelo simples de gramática livre de contexto. Podemos observar na figura que algumas regras de produção permitem que áreas pareadas e não pareadas possam ter tamanhos variados e, mesmo assim, ser aceitas pela gramática por meio da repetição de um mesmo símbolo não terminal do lado direito e do lado esquerdo. É o caso da produção $S_7 \rightarrow US_7$, que permite a presença de uma região formada por uma sequência de nucleotídeos U não pareados sem limite, e da produção $S_{10} \rightarrow CS_{10}G$, que permite uma região de hélice formada por um empilhamento de bases de pares C e G também sem limite de tamanho.

Figura 15 – Exemplo de um conjunto de regras de produção e de uma derivação de uma sequência de RNA

$$\begin{aligned}
 P = \{ & S_0 \rightarrow S_1, & S_7 \rightarrow G S_8, \\
 & S_1 \rightarrow C S_2 G, & S_8 \rightarrow G, \\
 & S_1 \rightarrow A S_2 U, & S_8 \rightarrow U, \\
 & S_2 \rightarrow A S_3 U, & S_9 \rightarrow A S_{10} U, \\
 & S_3 \rightarrow S_4 S_9, & S_{10} \rightarrow C S_{10} G, \\
 & S_4 \rightarrow U S_5 A, & S_{10} \rightarrow G S_{11} C, \\
 & S_5 \rightarrow C S_6 G, & S_{11} \rightarrow A S_{12} U, \\
 & S_6 \rightarrow A S_7, & S_{12} \rightarrow U S_{13}, \\
 & S_7 \rightarrow U S_7, & S_{13} \rightarrow C \quad \quad \quad \} \\
 \\
 S_0 \Rightarrow & S_1 \Rightarrow C S_2 G \Rightarrow C A S_3 U G \Rightarrow C A S_4 S_9 U G \\
 \Rightarrow & C A U S_5 A S_9 U G \Rightarrow C A U C S_6 G A S_9 U G \\
 \Rightarrow & C A U C A S_7 G A S_9 U G \Rightarrow C A U C A G S_8 G A S_9 U G \\
 \Rightarrow & C A U C A G G G A S_9 U G \Rightarrow C A U C A G G G A A S_{10} U U G \\
 \Rightarrow & C A U C A G G G A A G S_{11} C U U G \\
 \Rightarrow & C A U C A G G G A A G A S_{12} U C U U G \\
 \Rightarrow & C A U C A G G G A A G A U S_{13} U C U U G \\
 \Rightarrow & C A U C A G G G A A G A U C U C U U G.
 \end{aligned}$$

Fonte: Sakakibara *et al.* (1994a)

Na modelagem de RNAs por meio de gramáticas, cada árvore de derivação representa uma possível estrutura secundária. Isto é, uma estrutura sintática de uma sequência produzida por uma gramática que modela um RNA representa uma possível estrutura secundária da molécula, conforme exemplo da Figura 16.

Figura 16 – Árvore de derivação e estrutura secundária

Fonte: Sakakibara *et al.* (1994a)

Após a gramática inicial ser definida, manualmente, de modo a representar a estrutura secundária da família de RNA de interesse, sequências de treinamento sem informação estrutural (não alinhadas e não anotadas) são utilizadas para estimar as probabilidades associadas a cada regra de produção. Para isso, é utilizado o algoritmo Tree-Grammar EM. O algoritmo basicamente consiste em aplicar a gramática às sequências de treinamento, estimando as estruturas secundárias mais prováveis e, formando assim, um conjunto de treinamento com informação estrutural. Na sequência, o conjunto de estruturas secundárias é utilizado para reestimar as probabilidades associadas com cada regra de produção da gramática atual. Após isso, as estruturas secundárias são reestimadas utilizando a gramática livre de contexto estocástica recém ajustada, e o processo se repete até atingir um estado de convergência.

3.1.2 Modelos de covariância

Em paralelo e de forma independente dos estudos do grupo de Sakakibara, Eddy e Durbin propuseram os modelos de covariância, que são um tipo especial de gramáticas livres de contexto estocásticas (EDDY; DURBIN, 1994). Atualmente os modelos de covariância são a base do pacote de programas INFERNAL (NAWROCKI; KOLBE; EDDY, 2009; NAWROCKI; EDDY, 2013), pacote este bastante utilizado para busca de sequências similares de uma determinada família com estrutura secundária conservada (NAWROCKI, 2014; KORBI *et al.*, 2014; BARQUIST; BURGE; GARDNER, 2016).

Um grande diferencial entre os trabalhos de Sakakibara e Eddy e Durbin é a capacidade de inferência não só dos parâmetros, mas também das regras de produção da gramática que os programas do pacote INFERNAL disponibilizam. Para isso, são

utilizadas informações de alinhamentos múltiplos estruturais para caracterizar uma família de RNAs. A gramática inferida é então utilizada para a realização de buscas de sequências similares. Assim, os modelos de covariância são modelos probabilísticos que caracterizam tanto a sequência quanto a estrutura secundária de RNAs, utilizando para isso informações de alinhamentos estruturais com anotação de estruturas secundárias.

Os modelos de covariância são uma ampliação dos modelos ocultos de Markov (HMM) de forma a permitir a emissão de símbolos pareados, sendo assim capaz de representar um subconjunto de gramáticas estocásticas livres de contexto. Um modelo de covariância é composto por estados, símbolos de emissão, probabilidades de transição entre os estados e probabilidades de emissão de símbolos em cada estado. Os símbolos de emissão representam os nucleotídeos e os estados representam a estrutura secundária (pareamento, bojos, loops internos, bifurcações, começo e fim de estrutura etc.). Existem também estados que representam inserções de nucleotídeo à direita, inserções à esquerda e deleções tanto de um único nucleotídeo quanto de um par dos mesmos em relação à estrutura secundária consenso. (EDDY, 2003).

O primeiro passo para a construção de um modelo de covariância é a definição da estrutura consenso a partir de um alinhamento estrutural (Figura 17). Somente participam da estrutura consenso as colunas que possuem menos que 50% de *gaps*¹. Tais colunas podem corresponder a uma base pareada ou não pareada dependendo do símbolo presente na respectiva coluna, na linha que descreve a estrutura secundária: será considerada uma base pareada se o símbolo for “<” ou “>” e não pareada caso contrário.

Figura 17 – Alinhamento múltiplo e estrutura consenso

Fonte: Eddy (2003)

O segundo passo é, partindo da estrutura consenso, a construção de uma árvore guia formada por “nós” (Figura 18). A árvore guia possui oito tipos de nós, representando os elementos básicos da estrutura consenso: o nó ROOT marca o início da árvore; o nó

¹ Um *gap*, normalmente representado por um “.” ou “-”, representa um espaço que foi inserido para permitir o alinhamento das sequências.

END marca o fim de uma ramificação; o nó BIF indica o início de uma bifurcação, sendo seguido pelos nós BEGL e BEGR que marcam os inícios das ramificações esquerda e direita, respectivamente; os nós MATP, MATL e MATR representam, respectivamente, duas bases pareadas (MATP), colunas não pareadas à esquerda (MATL) e colunas não pareadas à direita (MATR).

Figura 18 – Árvore guia do modelo de covariância

Fonte: Eddy (2003)

No terceiro passo, após a montagem da árvore guia, o modelo é adaptado convertendo-se cada nó em um conjunto pré-estabelecido de estados e suas respectivas transições, formando a arquitetura completa do modelo de covariância (Figura 19). Para os estados possíveis correspondentes a cada nó da árvore guia é feita uma divisão em dois grupos, sendo o primeiro obrigatório (*“split set”*) e composto de um a quatro estados possíveis. O segundo grupo (*“inserts”*) é composto de estados de inserção, podendo ter de zero a dois elementos. O grupo *“split”* é composto pelos estados MP (emissão de pareamento de bases), ML (inserção de nucleotídeo não pareado à esquerda), MR (inserção de nucleotídeo não pareado à direita), D (deleção), B (bifurcação), S (marcação de início) e E (fim de ramificação). O grupo *“inserts”* é composto pelos estados IL (inserção de nucleotídeo não pareado à esquerda) e IR (inserção de nucleotídeo não pareado à direita). As transações podem ocorrer de cada estado do grupo *“split”* para todos os estados do grupo *“split”* do nó seguinte, de cada estado do grupo *“split”* para todos os estados do grupo *“inserts”* do próprio nó, do estado IR para o estado IL do próprio nó, do estado IL para ele mesmo (inserção recorrente), do estado IR para ele mesmo, dos estados IL e IR para cada estado do grupo *“split”* do nó seguinte.

Figura 19 – Arquitetura completa do modelo de covariância

Fonte: Eddy (2003)

Após a construção do modelo de covariância considerando todos os estados possíveis, a probabilidade de cada transição entre os estados é estimada utilizando o alinhamento

estrutural que serviu de base para a construção do modelo. A estimação é feita por máxima *a posteriori*, utilizando como *priori* uma mistura de Dirichlet (MACHADO-LIMA, 2006).

3.1.3 Intersecção de gramáticas livres de contexto estocásticas para representação de pseudonós

A derivação de uma gramática livre de contexto pode ser representada em forma de uma árvore hierárquica. Como um pseudonó é uma estrutura cruzada, não pode ser representada por uma gramática livre de contexto (SEARLS, 2002). Para contornar esse problema, Brown e Wilson (1995) modelaram separadamente as hélices componentes de um pseudonó, conforme Figura 20, descrevendo em seu trabalho como calcular as probabilidades de derivação de cada hélice separadamente, mantendo uma análise sintática com complexidades de tempo e memória na ordem de $O(n^3)$ e $O(n^2)$, respectivamente.

Figura 20 – Pseudonó desmembrado em duas gramáticas

Fonte: Brown e Wilson (1995)

3.2 Abordagens não gramaticais

Os métodos que podem ser utilizadas para caracterização de sequências de RNA com estrutura secundária que não utilizam abordagens baseadas em gramáticas podem ser reunidos em dois grupos principais, sendo um grupo composto pelos métodos que consideram pseudonós e o outro pelos que não consideram. Além disso, dentro de cada

um desses grupos são identificadas três divisões básicas: (i) estudos de propósito geral que propõem novas representações para estruturas secundárias de RNAs, (ii) algoritmos que tratam do problema de alinhamento entre uma sequência de RNA com estrutura secundária informada e uma ou mais sequências de RNA com estrutura desconhecida e (iii) algoritmos que tratam do problema de alinhamento entre duas ou mais sequências com estruturas secundárias informadas.

3.2.1 Métodos que consideram pseudonós

Representações alternativas de estruturas secundárias

O artigo em (HUANG; LI; GRIBSKOV, 2016) apresenta uma abordagem utilizando grafos para a conversão de uma estrutura secundária de RNA em uma assinatura de padrões topológicos, considerando inclusive pseudonós. Para isso, foi desenvolvida uma biblioteca considerando padrões compostos de 1 a 7 hélices, formando grafos não isomorfos que são utilizados como componentes para a construção de estruturas completas, formando assim as assinaturas estruturais. Na Figura 21 é apresentado um exemplo de uma assinatura de uma estrutura de RNA formada por 6 vértices, sendo decomposta em subgrafos de 3 vértices cada.

Figura 21 – Exemplo de uma assinatura topológica de RNA

Fonte: Huang, Li e Gribskov (2016)

Na Figura 22 é apresentado o RNA do vírus de hepatite D apresentado em forma de grafos.

Figura 22 – Vírus da hepatite D representado em formato de grafos

Fonte: Huang, Li e Gribskov (2016)

A busca de grafos, utilizando força bruta, apresenta complexidade $O(nm^m)$, sendo n é o número de grafos na biblioteca e m o número de arestas no grafo de busca.

O método proposto se mostrou capaz de ser utilizado para caracterização de estruturas incompletas, estruturas inferidas por métodos computacionais e estruturas contendo ou não pseudonós, atingindo um índice de área abaixo da curva ROC maior que 0.95 nos testes. Foram utilizados nos testes estruturas curadas de RNAs das famílias tRNA, tmRNA, rnasep, g1 introns, 16S rRNA, 5S rRNA, g2 introns e 23S rRNA, não sendo especificadas em detalhes as fontes.

Os programas utilizados estão disponíveis em (<https://github.rcac.purdue.edu/mgribsko/XIOS.RNA.fingerprint>).

Uma nova forma de representação gráfica da estrutura secundária de um RNA e sua utilização para medição de similaridade e classificação estrutural de RNAs é apresentada em (ZHANG *et al.*, 2016). No artigo é proposta uma forma de utilização da estrutura secundária associada às características químicas das bases para formar um gráfico sobre um plano tridimensional que representa um RNA por meio da distribuição de bases isoladas e

pares de bases em sua estrutura secundária, formando um vetor de características de 36 dimensões, sendo consideradas estruturas com pseudonós.

No artigo, os nucleotídeos não pareados são representados como A, U, G e C, enquanto as bases presentes em pares de bases são representadas por A', U', G' e C'.

Como características químicas das bases, são consideradas suas divisões em três grupos: (i) amino $M=\{A,C\}$ e keto $K=\{G,U\}$, (ii) purina $R=\{A,G\}$ e pirimidina $Y=\{C,U\}$ e (iii) ligação fraca de hidrogênio $W=\{A,U\}$ e ligação forte de hidrogênio $S=\{G,C\}$.

Para uma estrutura de RNA são definidos três conjuntos de pontos (x_{1i}, y_{1i}, z_{1i}) , (x_{2i}, y_{2i}, z_{2i}) e (x_{3i}, y_{3i}, z_{3i}) , $i = 1, 2, \dots, n$, sendo n é o tamanho da estrutura.

Para os testes foram utilizados três conjuntos de dados, tendo como origens nove vírus reportados em (REUSKEN; BOL, 1996), 17 estruturas da base RNase P (BROWN, 1998), 16 estruturas secundárias contendo pseudonós da base Pseud Base++ (BATENBURG *et al.*, 2000), 60 RNAs não codificantes de proteínas da base RNA STRAND (ANDRONESCU *et al.*, 2008) e 60 sequências selecionadas aleatoriamente de 18 famílias presentes na base Rfam. O método proposto apenas falhou na caracterização de um membro da família 5S ribossomal RNA (código no banco Rfam: RF00001), demonstrando um bom desempenho.

O programa desenvolvido está disponível como um suplemento do próprio artigo.

Algoritmos para alinhamento estrutura x sequência

O trabalho apresentado em (WONG *et al.*, 2011) tem como objetivo realizar o alinhamento estrutural entre uma sequência de busca com estrutura secundária conhecida e uma sequência alvo com estrutura não conhecida. O estudo apresentado é voltado para a identificação de estruturas contendo pseudonós em que três hélices se cruzam mutuamente (Figura 23A) e pseudonós que apresentam um nível de recursividade² (Figura 23B).

² Pseudonós com recursividade são pseudonós que apresentam outros pseudonós ou estruturas simples dentro de sua estrutura

O algoritmo proposto tem complexidade de tempo $O(kmn^{k+1})$, sendo k o grau do pseudonó, m o tamanho da sequência de busca com estrutura conhecida e n o tamanho da sequência sendo alinhada.

Para a realização de testes, foram consideradas as sequências de sete famílias obtidas do banco Rfam: RF00140, RF00094, RF00622, RF01084, RF01075, RF01085 e RF00176. Para cada família, um membro semente foi considerado como a sequência de busca, sendo as demais sequências inseridas em regiões arbitrárias de uma longa sequência aleatória gerada. O resultado do alinhamento entre a sequência de busca e sua estrutura e as regiões no genoma fictício foram comparados com os resultados de buscas realizadas com o BLAST (que considera apenas as sequências). Enquanto o BLAST apresentou uma taxa de acerto de 40% nos testes, o método proposto acertou 96.67% da amostra. Não é disponibilizado o programa desenvolvido no artigo.

O algoritmo proposto em (WONG; YIU, 2012) apresenta uma abordagem semelhante à apresentada em (WONG *et al.*, 2011), ou seja, tem como objetivo o alinhamento entre uma sequência com estrutura secundária conhecida e uma sequência de estrutura não conhecida, porém considerando a presença de triplas de bases. São consideradas ligações presentes somente dentro de pseudonós simples. Um exemplo de alinhamento é apresentado na Figura 25.

Figura 25 – Alinhamento entre duas estruturas, considerando triplas de bases

Fonte: Wong e Yiu (2012)

O algoritmo se baseia na maximização de três funções, uma de similaridade de sequências, uma de similaridade entre pares de bases que não se envolvem em uma interação

de triplas de bases e uma terceira função que apresenta uma similaridade entre interações envolvendo triplas de bases.

O algoritmo apresenta complexidade de tempo $O(mn^3)$, sendo m o tamanho da sequência sendo buscada e n o tamanho da sequência alvo.

Para os testes foram utilizadas as famílias RF00024, RF01050 e RF01074 do banco Rfam, sendo a estrutura secundária consenso contendo as relações de triplas de base deduzida de trabalhos correlatos. De cada família foi extraída uma região contendo a presença de triplas de base para ser a sequência de busca, sendo todas as sequências da mesma família inseridas arbitrariamente em uma longa sequência aleatória de um genoma simulado. Os resultados das buscas pelo algoritmo proposto foram comparados com os resultados encontrados utilizando o BLAST. O BLAST apresentou uma taxa de acerto de 93.85% nos testes, enquanto o método proposto acertou 97.76% da amostra. Não é disponibilizado o programa desenvolvido no artigo.

Algoritmos para alinhamento estrutura x estrutura

O algoritmo RKalign (SONG *et al.*, 2015) apresenta um método para alinhamento estrutural de duas sequências de RNA com estruturas secundárias conhecidas, contendo pseudonós. O princípio do algoritmo é a utilização de uma metodologia baseada em funções de partição para calcular as probabilidades *a posteriori* dos alinhamentos entre as bases ou pares de bases. Após o cálculo das probabilidades, é calculada a acurácia esperada do alinhamento entre as estruturas, devendo o resultado deste cálculo ser maximizado.

Para um alinhamento simples, o algoritmo tem uma complexidade de tempo $O(m^2n^2)$ e complexidade $O(mn)$ de espaço, sendo m e n os tamanhos das sequências sendo alinhadas. Em sua versão para alinhamentos múltiplos, o algoritmo tem complexidade de tempo $O(k^2n^4)$ e complexidade de espaço $O(k^2n^2)$, sendo k é o número de estruturas no alinhamento e n é o tamanho máximo das estruturas.

Para os testes, foram extraídas algumas estruturas de RNAs contendo pseudonós das bases PDB (BERMAN *et al.*, 2000) e RNA STRAND. O algoritmo RKalign apresentou uma média de erros de 43.52% no alinhamento de bases.

O programa desenvolvido está disponível em <http://bioinformatics.njit.edu/RKalign>.

O algoritmo PSMalign (CHIU; CHEN, 2015) apresenta uma abordagem baseada em grafos para resolver os problemas de edição e alinhamento entre estruturas contendo pseudonós. O problema de edição calcula o menor número de modificações necessárias para a conversão de uma estrutura em outra, utilizando uma série de operações de edição pré-definidas. O problema de alinhamento encontra uma estrutura consenso, sendo que o total de edições para que ambas as estruturas se tornem a estrutura consenso é minimizado. Na Figura 26 é apresentado um exemplo de modelagem de uma estrutura contendo pseudonós por meio de um grafo no formato utilizado pelo algoritmo proposto.

Figura 26 – Exemplo de uma estrutura de RNA contendo pseudonós e sua modelagem em grafos. Cada S_i representa uma hélice. Cada vértice V_i representa uma hélice S_i . As indicações P, K e N representam hélices com relações paralelas (P), aninhadas (N) e hélices que se cruzam formando pseudonós (K)

Fonte: Chiu e Chen (2015)

Para os testes, foram utilizadas famílias SRP e 16S rRNA da base BRASERO (ALLALI; SAGOT, 2008) e algumas famílias da base RNA STRAND, removendo os pares de bases não canônicos. O algoritmo apresentou um índice de área abaixo da curva ROC de 0.764 para a família SRP e 0.931 para a família 16s rRNA.

O programa desenvolvido está disponível em <http://homepage.cs.latrobe.edu.au/ypchen/psmalign>.

O *web server* CARNA (SORESCU *et al.*, 2012) apresenta uma ferramenta para alinhamento de múltiplas sequências de RNAs. A entrada do algoritmo é um conjunto de sequências associadas ou não a matrizes de probabilidades de pareamentos. Se forem informadas apenas as sequências, para cada uma é associada uma matriz de probabilidades

de pares de bases, utilizando para isso o programa RNAfold do pacote Vienna (GRUBER *et al.*, 2008) ou o programa PAIRS (DIRKS; PIERCE, 2004) do pacote NUPACK, o qual considera pseudonós. De posse das matrizes, é realizado um alinhamento entre pares de sequências de forma progressiva, utilizando uma estratégia todos contra todos.

Para os testes do *web server* foi utilizada a base BRAliBase 2.1 (WILM; MAINZ; STEGER, 2006), sendo medido o índice SPS³. Os alinhamentos resultantes tiveram suas estruturas secundárias previstas pelo programa RNAalifold do pacote Vienna, sendo as estruturas resultantes comparadas com as estruturas referência pelo índice de similaridade Matthews *correlation coefficient* (MCC) (MATTHEWS, 1975). Para sequências apresentando 40% de similaridade, o algoritmo apresentou um índice SPS em torno de 0.7 e um índice MCC de 0.75.

O *web server* está disponibilizado em <http://rna.informatik.uni-freiburg.de/CARNA>.

3.2.2 Métodos que não consideram pseudonós

Representações alternativas de estruturas secundárias

O trabalho apresentado em (MATTEI *et al.*, 2014) apresenta uma nova abordagem de representação de estruturas secundárias de RNAs. A notação BEAR (*Brand nEw Alphabet for RNA*) representa cada nucleotídeo por um carácter que representa o componente estrutural a que o mesmo faz parte (*loop*, hélice, bojo etc.) e o tamanho da estrutura. Não são considerados pseudonós no artigo. Na Figura 27A é apresentado o alfabeto utilizado na codificação BEAR. Na Figura 27B é apresentado um exemplo de alguns componentes estruturais, é apresentado também o alfabeto BEAR e a equivalência entre sequência, estrutura secundária e codificação BEAR. Na Figura 27C é apresentada uma conversão entre uma estrutura secundária de um RNA extraída da base Rfam e a codificação BEAR.

³ o índice *Sum-of-pairs score* (SPS) é a relação entre os pares alinhados na estrutura referência e os pares preditos

Algoritmo para alinhamento estrutura x sequência

O algoritmo PMFastR (DEBLASIO; BRUAND; ZHANG, 2012) tem como objetivo realizar um alinhamento estrutural entre múltiplos RNAs de uma mesma família, sendo a complexidade de memória reduzida de forma a permitir o alinhamento de sequências longas, desconsiderando pseudonós.

Na Figura 28 é apresentada uma visão geral do algoritmo PMFastR. A entrada do algoritmo é uma sequência com estrutura conhecida e uma base de dados de sequências pertencentes à mesma família, porém sem anotação estrutural. A primeira parte do algoritmo é responsável por construir um *profile* inicial, resultante do alinhamento entre a sequência com estrutura conhecida e uma sequência selecionada da base de dados. Após a criação do *profile* inicial, cada sequência da base de dados é alinhada com o mesmo de forma incremental, sendo considerada a distribuição de probabilidade em cada coluna como fator de escore para a realização dos alinhamentos. O alinhamento resultante, após o processamento de todas as sequências, possui um número alto de “gaps”, sendo necessária a aplicação de um passo extra para refinamento do alinhamento, utilizando para isso o CMBuild do pacote Infernal, descrito na Seção 3.1.2.

Figura 28 – Arquitetura utilizada no algoritmo PMFastR

Fonte: DeBlasio, Bruand e Zhang (2012)

O algoritmo PMFastR apresenta complexidade de tempo $O(n^3)$ e complexidade de espaço teórica $O(n)$.

Para a realização de testes, foram utilizadas todas as famílias da base Rfam 8.1, além da base de dados de *benchmark* BRAliBase 2.1. Foram também realizados alinhamentos entre sequências de 16S rRNA encontradas na base de dados CRW (CANNONE *et al.*,

2002). O algoritmo apresentou para sequências com 40% de similaridade um índice SPS de 0.95 e um índice SCR⁴ de 0.75.

O programa desenvolvido está disponibilizado em <http://genome.ucf.edu/PMFastR>.

Algoritmos para alinhamento estrutura x estrutura

O algoritmo R-PASS (JIANG *et al.*, 2011) apresenta uma forma de alinhamento entre duas sequências anotadas estruturalmente, sem considerar pseudonós. O algoritmo realiza comparações no nível estrutural, criando vetores de características que consideram o padrão estrutural (loops, hélices etc.), a quantidade de nucleotídeos em cada padrão, a posição relativa dentro da estrutura total (considerando começo e fim relativo).

Como forma de ganhar velocidade, o alinhamento é realizado em basicamente duas etapas. Na primeira é aplicada uma técnica baseada em grafos para comparação entre os padrões estruturais e, dentro dos padrões alinhados, é realizado na sequência o alinhamento entre as sequências (Figura 29).

Figura 29 – R-PASS: Conversão entre alinhamento de estruturas e alinhamento de sequências

Fonte: Jiang *et al.* (2011)

O algoritmo apresenta complexidade de tempo $O(nm)$.

Para a realização de testes foram utilizadas as famílias RF00004, RF00009 e RF00010 da Rfam 10.1, as famílias g2intron, U5 spliceosomal RNA, tRNA e 5S rRNA da base

⁴ O índice *Structure Conservation Rate* (SCR) calcula a fração de pares de bases conservados no alinhamento, utilizando como referência as estruturas das bases de *benchmark*

BRAlBase 2.1 e as famílias 5S rRNA e 16S rRNA da base CRW. Nos testes, o algoritmo R-PASS demonstrou acurácia acima de 85% para sequências com 60% de similaridade. Não é disponibilizado o programa desenvolvido no artigo.

O algoritmo ERA (ZHONG; ZHANG, 2013) tem como objetivo realizar um alinhamento entre duas sequências de estruturas secundárias conhecidas. O foco do trabalho é no desenvolvimento de um algoritmo com bom desempenho de tempo, sem abrir mão da qualidade do alinhamento final. Para isso, é utilizada uma técnica de programação dinâmica esparsa para implementar um algoritmo baseado em distâncias de edição de duas sequências e é utilizada uma técnica de poda para melhorar o tempo de execução real. A similaridade entre as sequências é determinada por uma função que considera tanto a sequência quanto a similaridade entre as estruturas, não sendo considerados pseudonós.

O algoritmo proposto apresenta uma complexidade de tempo na ordem $O(n^3)$, sendo n é o número médio de pares de base nas estruturas de RNA.

Para os testes do ERA foram utilizadas algumas famílias do Rfam e do BRAlBase. No artigo são apresentadas as famílias tRNA, Gly riboswitch, U12 spliceosome, Phage.pRNA, tmRNA, biocoid.3UTR, snR86 e Sacc.telomerase. Para sequências com 40% de similaridade, o algoritmo ERA obteve um índice SPS de 0.89 e um índice SCI⁵ de 1.075.

O programa está disponível em <http://genome.ucf.edu/ERA>.

Em (MATTEI *et al.*, 2015) é apresentado o Web-Beagle, uma ferramenta web de busca de similaridades estruturais e alinhamentos entre estruturas de RNAs utilizando a codificação BEAR para a representação da estrutura secundária de RNAs e a matriz de substituição MBR para guiar os alinhamentos. Além do algoritmo Needleman-Wunsch, que realiza um alinhamento global, o *web server* implementa o algoritmo Smith-Waterman (SMITH; WATERMAN, 1981) que realiza alinhamento local.

O web server Web-Beagle apresenta complexidade de tempo $O(n^2)$ ao alinhar estruturas secundárias de RNAs.

⁵ O índice *Structure Conservation Index* (SCI) é definido pela fração entre a mínima energia livre da estrutura consenso e a média de mínima energia livre das sequências

Para os testes, foram preparados quatro conjuntos de teste, formados por RNAs extraídos das bases RNA STRAND, RNAspa (HORESH *et al.*, 2007), RNASTAR (WIDMANN *et al.*, 2012) e BRAlibase. Considerando sequências com 50% de similaridade, o algoritmo apresentou um índice SPS de 0.75.

O web server está disponível em <http://beagle.bio.uniroma2.it>.

4 GRAMMARLAB ADAPTATIVO

Uma vez que o arcabouço GrammarLab possui toda uma infraestrutura que atende às principais demandas para reconhecimento de padrões sintáticos para o problema alvo de caracterização de famílias de RNAs, como a implementação do algoritmo de Earley e várias classes de apoio para a implementação de classificadores, o mesmo foi selecionado como objeto de aplicação do estudo desta pesquisa.

Serão apresentados neste capítulo os métodos e resultados da implementação da evolução do arcabouço de forma que o mesmo possa lidar com gramáticas sensíveis ao contexto. Para isso, conforme visto na seção 2.1.6, foram utilizados dispositivos adaptativos, de forma que a complexidade de análise sintática não seja aumentada excessivamente ao ponto de inviabilizar seu uso na prática.

O foco do presente trabalho é a implementação da análise sintática baseada no algoritmo de Earley para gramáticas adaptativas, permitindo assim a classificação de sequências de estruturas que apresentam elementos que necessitem de uma gramática no mínimo sensível ao contexto, a exemplo dos pseudonós encontrados nas estruturas secundárias de alguns RNAs.

A seção 4.1 descreve como o conjunto de classes que implementam gramáticas e seus componentes foi alterado para implementar gramáticas adaptativas e não adaptativas. A seção 4.2 descreve a notação BNF-*like* expandida para representar gramáticas adaptativas, útil para leitura e escrita de gramáticas pelo arcabouço. Finalmente a seção 4.3 descreve a versão adaptativa proposta neste trabalho para o algoritmo de Earley, um exemplo de execução deste novo algoritmo no reconhecimento de uma linguagem sensível ao contexto, e as alterações realizadas no GrammarLab para a implementação do algoritmo de Earley adaptativo.

4.1 Gramáticas adaptativas

Atualmente os componentes do GrammarLab são estruturados ao redor de uma modelagem de classes capaz de representar uma gramática, conforme Figura 30.

Nessa modelagem, a classe *Grammar* é uma classe abstrata que representa o comportamento geral de uma gramática. Sua subclasse *ConcreteGrammar* implementa

Figura 30 – GrammarLab - Gramática Livre de Contexto

Fonte: Gilmar Pereira dos Santos, 2018

uma gramática composta por símbolos terminais (classe *Terminal*) e não terminais (classe *NonTerminal*) e por regras de produção (classe *Rule*). As regras de produção são limitadas a apenas possuírem um símbolo não terminal do lado esquerdo e um conjunto ordenado de símbolos (classe *Symbol*) do lado direito, que podem ser terminais ou não terminais. Pode assim ser observado que o escopo do arcabouço está fixo em gramáticas no máximo livres de contexto.

Para permitir a inclusão de ações adaptativas associadas às regras de produção de uma gramática livre de contexto, a modelagem foi alterada para permitir regras que estejam de acordo com o formalismo apresentado na seção 2.1.6, conforme apresentado na Figura 31. Para isso, sete novas classes foram criadas: *AdaptiveGrammar*, *AdaptiveRule*, *AdaptiveFunction*, *AdaptiveAction*, *SearchAdaptiveAction*, *InsertAdaptiveAction* e *DeleteAdaptiveAction*.

A classe *AdaptiveGrammar* foi criada como uma extensão da classe *ConcreteGrammar*, apresentando como grande diferencial em sua composição um conjunto de funções adaptativas (*AdaptiveFunction*) e a substituição da classe *Rule* pela classe *AdaptiveRule* como componente representativo das regras de produção.

A classe *AdaptiveRule*, por sua vez, é a responsável por ampliar as funcionalidades da classe *Rule* de tal forma que seja possível associar o disparo de funções adaptativas às

Figura 31 – GrammarLab - Gramática Adaptativa

Fonte: Gilmar Pereira dos Santos, 2018

regras de produção da gramática, podendo haver uma função que é executada antes da aplicação da regra de produção e uma outra função após.

A classe *AdaptiveFunction*, responsável pela inclusão do comportamento adaptativo, possui uma ou mais ações adaptativas elementares (*AdaptiveAction*) associadas, e também pode disparar uma função adaptativa ao iniciar seu processamento e uma outra ao término do processamento de suas ações adaptativas.

As ações adaptativas elementares (*AdaptiveAction*), conforme visto na seção 2.1.6, sempre estarão associadas a uma determinada regra de produção adaptativa (*AdaptiveRule*), podendo ser ações adaptativas de pesquisa (*SearchAdaptiveAction*), de inserção (*InsertAdaptiveAction*) ou de deleção (*DeleteAdaptiveAction*). A regra de ordem de ativação

das ações adaptativas foi implementada na classe *AdaptiveFunction*, que é por sua vez executada durante a análise sintática, como será apresentado na seção 4.3.

Assim, pode ser observado que a modelagem implementada estabelece a estrutura necessária para uma gramática livre de contexto com comportamento adaptativo inserido em seu formalismo, sendo a base para atingir o objetivo proposto.

4.2 Notação BNF para gramáticas adaptativas

O GrammarLab disponibiliza em sua implementação uma hierarquia de classes que modela uma arquitetura baseada em fluxos de dados, ou *streams*, que permite que os componentes do arcabouço, independentemente de sua implementação, possam se comunicar de uma forma abstrata e permitindo armazenamento de resultados intermediários.

O fluxo por onde chegam os dados é modelado pela estrutura representada na Figura 32. Desta estrutura, para este trabalho, merecem destaques as classes *InputStream*, *GrammarInputStream* e *BNFGrammarInputStream*, por serem as responsáveis por permitir que o arcabouço interprete gramáticas descritas em uma notação BNF-like.

Figura 32 – GrammarLab - Modelagem de InputStream

Fonte: Machado-Lima (2002)

A classe abstrata *InputStream* é a raiz da hierarquia, tendo como papel a modelagem de um fluxo de dados vindos de uma fonte genérica.

A classe abstrata *GrammarInputStream* modela um fluxo de entrada, sendo os dados uma representação de uma gramática com seus símbolos terminais, não terminais, produções e probabilidades associadas.

A classe concreta *BNFGrammarInputStream* implementa a funcionalidade de leitura de gramáticas descritas em um formato BNF-like.

Para que o arcabouço possa trabalhar com descrições de gramáticas adaptativas, foram feitas alterações na estrutura de *InputStream*, sendo implementadas as classes concretas *BNFAdaptGrammarInputStream* e *GrammarFactoryInputStream* (Figura 33). A classe *BNFAdaptGrammarInputStream* contém todas as regras necessárias para reconhecimento dos componentes de uma gramática adaptativa. A classe *GrammarFactoryInputStream* tem o objetivo de funcionar como uma fábrica de gramáticas que, de acordo com a identificação do tipo de gramática informada na descrição, instancia e inicializa a classe ideal para o tipo de gramática descrita. Neste trabalho foi implementada a geração de gramáticas adaptativas que, dada uma entrada com a descrição da gramática adaptativa em uma notação BNF-like especialmente desenvolvida neste trabalho, instancia toda a hierarquia de classes representada na Figura 31.

Figura 33 – GrammarLab - Modelagem de GrammarFactoryInputStream

Fonte: Gilmar Pereira dos Santos, 2018

Para representar as gramáticas adaptativas utilizadas na nova implementação do arcabouço GrammarLab, foi adotada uma variação da notação BNF com a inclusão de

definição de funções adaptativas e associação entre tais funções e regras de produção de gramáticas (Quadro 1).

A definição das funções adaptativas na notação, apresentada no Quadro 1, segue o padrão apresentado na seção 2.1.6, sendo utilizadas as seguintes definições:

- **<GrammarType>** é um valor que identifica o tipo de gramática sendo descrita. Neste trabalho foi adicionado o tipo **5** como identificador de uma gramática livre de contexto adaptativa;
- **<NonTerminal>** representa um símbolo não terminal da gramática e marca o início da definição de uma regra de produção adaptativa;
- **<Vocabulary>** representa uma sequência composta de símbolos terminais, não terminais, variáveis ou geradores;
- **\ba{AdaptiveFunction(parameter,...)}** representa uma função adaptativa que é executada antes da ativação da produção em que está associada;
- **\aa{AdaptiveFunction(parameter,...)}** representa uma função adaptativa que é executada após a ativação da produção em que está associada;
- O símbolo de barra invertida isolado (“\”) representa uma continuidade de linha;
- As funções adaptativas são demarcadas pela marcação **\BeginFunction** seguida de nome e parâmetros e **\EndFunction**, que marca o fim da declaração da função;
- **\DefineVar** indica a declaração de uma variável (que deve ser definida em ações de pesquisa);
- **\DefineGenerator** indica a declaração de um símbolo gerador;
- **\BeforeFunction** indica uma função adaptativa que deve ser executada antes da função atual;
- **\AfterFunction** indica uma função adaptativa que deve ser executada após a execução da função atual;
- As ações adaptativas de pesquisa são delimitadas pelas marcações **\BeginSearchActions** e **\EndSearchActions**;
- As ações adaptativas de deleção são delimitadas pelas marcações **\BeginDeleteActions** e **\EndDeleteActions**;
- As ações adaptativas de inserção são delimitadas pelas marcações **\BeginInsertActions** e **\EndInsertActions**;

Quadro 1 – Notação BNF para gramáticas adaptativas

```

<GrammarType>

;produções adaptativas
<NonTerminal> ::= [\ba{AdaptiveFunction(parameter,...)}] \
                  <Vocabulary> \
                  [\aa{AdaptiveFunction(parameter,...)}]
[...] ;Outras produções

[\BeginFunction <AdaptiveFunctionName>(parameter,...)
  [\DefineVar <var>] ;variável que deve ser definida em ações de pesquisa
  [...] ;outras variáveis

  [\DefineGenerator <varGen>] ;símbolo não terminal gerado dinamicamente
  [...] ;outros símbolos geradores

;variáveis e geradores podem ser utilizados em parâmetros
;e/ou no vocabulário de ações adaptativas

;Função Adaptativa que é executada antes da função atual
[\BeforeFunction <AdaptiveFunctionName>(parameter,...)]

[\BeginSearchActions
  <NonTerminal> ::= [\ba{AdaptiveFunction(parameter,...)}] \
                    <Vocabulary> \
                    [\aa{AdaptiveFunction(parameter,...)}]
  [...] ;outras ações adaptativas
\EndSearchActions]

[\BeginDeleteActions
  <NonTerminal> ::= [\ba{AdaptiveFunction(parameter,...)}] \
                    <Vocabulary> \
                    [\aa{AdaptiveFunction(parameter,...)}]
  [...] ;outras ações adaptativas
EndDeleteActions]

[\BeginInsertActions
  <NonTerminal> ::= [\ba{AdaptiveFunction(parameter,...)}] \
                    <Vocabulary> \
                    [\aa{AdaptiveFunction(parameter,...)}]
  [...] ;outras ações adaptativas
\EndInsertActions]

;Função Adaptativa que é executada após a função atual
[\AfterFunction <AdaptiveFunctionName>(parameter,...)]
\EndFunction]

```

Exemplo de gramática na notação BNF para gramáticas adaptativas

O Quadro 2 apresenta a gramática adaptativa utilizada como demonstração ao longo desse trabalho (que gera a linguagem $L(G) = \{a^n b^m c^n d^m\}$, como descrito na seção 2.1.6) representada na notação BNF elaborada.

4.3 Algoritmo de Earley adaptativo

Conforme apresentado na seção 2.1.4, a análise sintática realizada pelo algoritmo de Earley é baseada na aplicação das operações *predictor*, *completer* ou *scanner* sobre cada um dos itens de análise.

A operação *predictor* é a responsável pela análise de símbolos não terminais nas regras de produção, provocando o comportamento de descida na árvore de derivação, adicionando novas regras de produção ao conjunto de estados que será analisado no ciclo do algoritmo. É ela quem inicializa o campo de conjunto (de estados) de origem de um estado, já que este campo indica em qual conjunto de estados um não terminal foi aberto. Por ser a primeira operação aplicada em cada regra de produção que é analisada, a operação *predictor* é onde devem ser ativadas as funções adaptativas associadas a cada regra. Como a ativação de uma função adaptativa resulta em uma nova gramática, surge a necessidade da gramática resultante ser adicionada como um novo elemento na definição de cada estado de análise do algoritmo.

A operação *scanner* é a responsável pela análise de símbolos terminais nas regras de produção, comparando-os com o símbolo da cadeia de entrada que está sendo analisado no passo atual do algoritmo. Como as produções em que esta operação é aplicada foram obrigatoriamente fruto de uma operação *predictor*, não devem ser ativadas na operação *scanner* as ações adaptativas (pois isso já ocorreu na operação *predictor*).

A operação *completer* indica que a análise de um determinado símbolo não terminal (iniciada na operação *predictor*) foi finalizada em uma derivação coerente com a cadeia de entrada até o ponto atual em que o algoritmo se encontra na análise. Sendo assim, também não deve ser ativada nenhuma função adaptativa nesta operação.

Nesta seção será apresentado um passo a passo da aplicação do algoritmo de Earley modificado para contemplar gramáticas adaptativas e na seção 4.3.2 será apresen-

Quadro 2 – Exemplo de notação BNF para uma gramática adaptativa que descreve a linguagem $L(G) = \{a^n b^m c^n d^m\}$

5

```

<S> ::= <K> \aa{AdP(<K>)}

\BeginFunction AdP(X)
  \DefineGenerator <A'>
  \DefineGenerator <B'>
  \DefineGenerator <C'>
  \DefineGenerator <D'>

  \BeginInsertActions
    X ::= <A'><B'><C'><D'>
    <A'> ::= a<A'> \aa{Ad1(<A'>,<C'>,a,c)}
    <A'> ::= a \aa{Ad2(<C'>,c)}
    <B'> ::= b<B'> \aa{Ad1(<B'>,<D'>,b,d)}
    <B'> ::= b \aa{Ad2(<D'>,d)}
  \EndInsertActions
\EndFunction

\BeginFunction Ad1(X,Y,x,y)
  \DefineGenerator <Y'>

  \BeginDeleteActions
    X ::= xX \aa{Ad1(X,Y,x,y)}
    X ::= x \aa{Ad2(Y,y)}
  \EndDeleteActions

  \BeginInsertActions
    X ::= xX \aa{Ad1(X,<Y'>,x,y)}
    X ::= x \aa{Ad2(<Y'>,y)}
    Y ::= y<Y'>
  \EndInsertActions
\EndFunction

\BeginFunction Ad2(Y,y)
  \BeginInsertActions
    Y ::= y
  \EndInsertActions
\EndFunction

```

tada a estratégia de implementação e o algoritmo em pseudocódigo, complementando o entendimento da abordagem adotada neste trabalho.

4.3.1 Exemplo de aplicação do algoritmo para reconhecimento

A seguir, um exemplo de uma análise sintática considerando a gramática adaptativa e a cadeia de exemplo apresentadas na seção 2.1.6.

Inicialmente, a cadeia de entrada é complementada com $k + 1$ símbolos $\dashv$ não pertencentes ao alfabeto da gramática que simbolizam o término da cadeia de entrada.

$$X = aabbbccddd \dashv \dashv$$

Passo $i = 0$, conjunto de estados S_0 e $X_1 = a$:

Como passo inicial $i = 0$, o estado inicial é adicionado ao conjunto de estados S_0 .

$$S_0 : \quad \phi \rightarrow .S \dashv \dashv 0 \ G^0 \quad (1)$$

Nessa representação de um estado, $\phi \rightarrow S \dashv$ é a produção, o “.” indica até que ponto o reconhecimento já foi realizado com sucesso, o símbolo $\dashv$ representa a cadeia de símbolos terminais seguintes à produção (ou seja, neste caso apenas o símbolo de término de cadeia), 0 é o número do conjunto de estados atual, no qual a produção foi trazida com o “.” logo no início do lado direito (como uma operação *predictor* faz) e G^0 representa a gramática resultante do acionamento da produção.

Como o símbolo após “.” é o símbolo não terminal S , a operação *predictor* é aplicada à produção inicial original da gramática G^0 , gerando o estado 2 ainda no conjunto de estados S_0 , ativando a função adaptativa associada a tal produção e gerando uma nova gramática G^1 .

$$S_0 : \quad S \rightarrow .K\{AdP(K)\} \dashv 0 \ G^1 \quad (2)$$

Novamente a operação *predictor* é aplicada ao novo estado, adicionando um novo estado no conjunto S_0 , utilizando desta vez a produção da gramática G^1 . Como não há uma função adaptativa associada com a produção, desta vez a gramática não evolui.

$$S_0 : \quad K \rightarrow .A^1B^1C^1D^1 \quad \vdash 0 \ G^1 \quad (3)$$

A operação *predictor* é aplicada mais uma vez. Como o símbolo seguinte a A^1 é o não terminal B^1 , o mesmo é avaliado a fim de identificar os próximos símbolos não terminais possíveis como a cadeia seguinte à produção (*lookahead*), adicionando então os seguintes estados ao conjunto S_0 .

$$S_0 : \quad A^1 \rightarrow .aA^1\{Ad1(A^1, C^1, a, c)\} \quad b \ 0 \ G^{2.1} \quad (4)$$

$$S_0 : \quad A^1 \rightarrow .a\{Ad2(C^1, c)\} \quad b \ 0 \ G^{2.2} \quad (5)$$

A operação *predictor* não pode ser aplicada a nenhum dos últimos estados adicionados, pois em todos eles o símbolo à direita do ponto é um símbolo terminal. Neste caso, a operação *scanner* é aplicada, adicionando ao conjunto de estados S_{i+1} os estados em que o símbolo terminal seguinte ao ponto for igual ao símbolo $X_{i+1} = a$ da cadeia de entrada. Os estados são copiados, tendo como alteração apenas a posição do ponto, que é deslocado um símbolo para a direita, indicando que o símbolo terminal anterior foi processado.

Assim, o conjunto de estados S_1 , após a aplicação da operação *scanner* em todos os últimos estados de S_0 , possui os seguintes estados:

$$S_1 : \quad A^1 \rightarrow a.A^1\{Ad1(A^1, C^1, a, c)\} \quad b \ 0 \ G^{2.1} \quad (6)$$

$$S_1 : \quad A^1 \rightarrow a\{Ad2(C^1, c)\} \cdot \quad b \ 0 \ G^{2.2} \quad (7)$$

Passo $i = 1$, conjunto de estados S_1 e $X_2 = a$:

Como não existem mais estados a serem processados no conjunto S_0 , o algoritmo evolui para o passo $i = 1$ e o conjunto S_1 se torna o conjunto a ser processado.

Como o símbolo não terminal A^1 está logo após o ponto no estado 6, a operação *predictor* é aplicada, adicionando ao conjunto S_1 os estados 8 e 9:

$$S_1 : \quad A^1 \rightarrow .aA^1\{Ad1(A^1, C^2, a, c)\} \quad b \ 1 \ G^{3.1} \quad (8)$$

$$S_1 : \quad A^1 \rightarrow .a\{Ad2(C^2, c)\} \quad b \ 1 \ G^{3.2} \quad (9)$$

A operação *completer* é aplicada nos estados em que o ponto se encontra no fim da produção, sendo comparados os k símbolos da cadeia posterior do estado com os símbolos $X_{i+1}..X_{i+k}$ da cadeia de entrada. O estado 7 possui uma cadeia posterior divergente da cadeia de entrada, sendo portanto descartado.

Após a aplicação da operação *scanner* nos últimos estados de S_1 (estados 8 e 9), os seguintes estados são adicionados ao conjunto de estados S_2 :

$$S_2 : \quad A^1 \rightarrow a.A^1\{Ad1(A^1, C^2, a, c)\} \quad b \ 1 \ G^{3.1} \quad (10)$$

$$S_2 : \quad A^1 \rightarrow a\{Ad2(C^2, c)\}. \quad b \ 1 \ G^{3.2} \quad (11)$$

O algoritmo evolui para o passo $i = 2$ na sequência.

Passo $i = 2$, conjunto de estados S_2 e $X_2 = b$:

A operação *predictor* é aplicada ao estado 10, adicionando ao conjunto S_2 os estados:

$$S_2 : \quad A^1 \rightarrow .aA^1\{Ad1(A^1, C^3, a, c)\} \quad b \ 2 \ G^{4.1} \quad (12)$$

$$S_2 : \quad A^1 \rightarrow .a\{Ad2(C^3, c)\} \quad b \ 2 \ G^{4.2} \quad (13)$$

A operação *completer* é aplicada ao estado 11 e, como o símbolo posterior à produção (*lookahead*) é igual ao próximo símbolo de entrada $X_3 = b$, devem ser trazidos para o conjunto de estados atual (S_2) todos os estados do conjunto de estados S_j (sendo j o número do conjunto de estados no qual a produção foi originada via operação *predictor*, neste caso também o conjunto de estados S_2) em que o símbolo à direita do ponto seja o símbolo não terminal do lado esquerdo da produção do estado 11 (em que a operação

completer foi aplicada), sendo o ponto movido para a direita, indicando que o símbolo anterior foi processado. A gramática do estado em que a operação *completer* foi aplicada é mantida nos novos estados. O seguinte estado é então adicionado ao conjunto S_2 :

$$S_2 : \quad A^1 \rightarrow aA^1\{Ad1(A^1, C^1, a, c)\}. \quad b \ 0 \ G^{3.2} \quad (14)$$

Os estados 12 e 13 são descartados pela operação *scanner*.

A operação *completer* é aplicada no estado 14, adicionando o novo estado:

$$S_2 : \quad K \rightarrow A^1.B^1C^1D^1 \quad \vdash \ 0 \ G^{3.2} \quad (15)$$

A operação *predictor* é aplicada na sequência, adicionando os seguintes estados a S_2 :

$$S_2 : \quad B^1 \rightarrow .bB^1\{Ad1(B^1, D^1, b, d)\} \quad c \ 2 \ G^{4.3} \quad (16)$$

$$S_2 : \quad B^1 \rightarrow .b\{Ad2(D^1, d)\} \quad c \ 2 \ G^{4.4} \quad (17)$$

As três operações continuam sendo aplicadas estado a estado, sendo os conjuntos de estados de cada passo do algoritmo apresentados resumidamente na sequência. Para não sobrecarregar a notação, já que todos os estados de cada conjunto serão descritos conjuntamente, será omitida a informação do conjunto de estados ao qual ele foi adicionado (“ $S_i : \quad$ ”)

Passo $i = 3$, conjunto de estados S_3 e $X_4 = b$:

$$B^1 \rightarrow b.B^1\{Ad1(B^1, D^1, b, d)\} \quad c \ 2 \ G^{4.3} \quad (18)$$

$$B^1 \rightarrow b\{Ad2(D^1, d)\}. \quad c \ 2 \ G^{4.4} \quad (19)$$

$$B^1 \rightarrow .bB^1\{Ad1(B^1, D^2, b, d)\} \quad c \ 3 \ G^{5.1} \quad (20)$$

$$B^1 \rightarrow .b\{Ad2(D^2, d)\} \quad c \ 3 \ G^{5.2} \quad (21)$$

Passo $i = 4$, conjunto de estados S_4 e $X_5 = b$:

$$B^1 \rightarrow b.B^1\{Ad1(B^1, D^2, b, d)\} \quad c \ 3 \ G^{5.1} \quad (22)$$

$$B^1 \rightarrow b\{Ad2(D^2, d)\}. \quad c \ 3 \ G^{5.2} \quad (23)$$

$$B^1 \rightarrow .bB^1\{Ad1(B^1, D^3, b, d)\} \quad c \ 4 \ G^{6.1} \quad (24)$$

$$B^1 \rightarrow .b\{Ad2(D^3, d)\} \quad c \ 4 \ G^{6.2} \quad (25)$$

Passo $i = 5$, conjunto de estados S_5 e $X_6 = c$:

$$B^1 \rightarrow b.B^1\{Ad1(B^1, D^3, b, d)\} \quad c \ 4 \ G^{6.1} \quad (26)$$

$$B^1 \rightarrow b\{Ad2(D^3, d)\}. \quad c \ 4 \ G^{6.2} \quad (27)$$

$$B^1 \rightarrow .bB^1\{Ad1(B^1, D^4, b, d)\} \quad c \ 5 \ G^{7.1} \quad (28)$$

$$B^1 \rightarrow .b\{Ad2(D^4, d)\} \quad c \ 5 \ G^{7.2} \quad (29)$$

$$B^1 \rightarrow bB^1\{Ad1(B^1, D^2, b, d)\}. \quad c \ 3 \ G^{6.2} \quad (30)$$

$$B^1 \rightarrow bB^1\{Ad1(B^1, D^1, b, d)\}. \quad c \ 2 \ G^{6.2} \quad (31)$$

$$K \rightarrow A^1B^1.C^1D^1 \quad \dashv \ 0 \ G^{6.2} \quad (32)$$

$$C^1 \rightarrow .cC^2 \quad d \ 5 \ G^{6.2} \quad (33)$$

Passo $i = 6$, conjunto de estados S_6 e $X_7 = c$:

$$C^1 \rightarrow c.C^2 \quad d \ 5 \ G^{6.2} \quad (34)$$

$$C^2 \rightarrow .c \quad d \ 6 \ G^{6.2} \quad (35)$$

Passo $i = 7$, conjunto de estados S_7 e $X_8 = d$:

$$C^2 \rightarrow c. \quad d \ 6 \ G^{6.2} \quad (36)$$

$$C^1 \rightarrow cC^2. \quad d \ 5 \ G^{6.2} \quad (37)$$

$$K \rightarrow A^1B^1C^1.D^1 \quad \vdash \ 0 \ G^{6.2} \quad (38)$$

$$D^1 \rightarrow .dD^2 \quad \vdash \ 7 \ G^{6.2} \quad (39)$$

Passo $i = 8$, conjunto de estados S_8 e $X_9 = d$:

$$D^1 \rightarrow d.D^2 \quad \vdash \ 7 \ G^{6.2} \quad (40)$$

$$D^2 \rightarrow .dD^3 \quad \vdash \ 8 \ G^{6.2} \quad (41)$$

Passo $i = 9$, conjunto de estados S_9 e $X_{10} = d$:

$$D^2 \rightarrow d.D^3 \quad \vdash \ 8 \ G^{6.2} \quad (42)$$

$$D^3 \rightarrow .d \quad \vdash \ 9 \ G^{6.2} \quad (43)$$

Passo $i = 10$, conjunto de estados S_{10} e $X_{11} = \vdash$:

$$D^3 \rightarrow d. \quad \vdash \quad 9 \quad G^{6.2} \quad (44)$$

$$D^2 \rightarrow dD^3. \quad \vdash \quad 8 \quad G^{6.2} \quad (45)$$

$$D^1 \rightarrow dD^2. \quad \vdash \quad 7 \quad G^{6.2} \quad (46)$$

$$K \rightarrow A^1 B^1 C^1 D^1. \quad \vdash \quad 0 \quad G^{6.2} \quad (47)$$

$$S \rightarrow K\{AdP(K)\}. \quad \vdash \quad 0 \quad G^{6.2} \quad (48)$$

$$\phi \rightarrow S. \quad \vdash \quad \vdash \quad 0 \quad G^{6.2} \quad (49)$$

Ao aplicar a operação *scanner* no último estado não processado, estado 49, é adicionado ao conjunto de estados S_{11} o seguinte estado:

$$\phi \rightarrow S \vdash. \quad \vdash \quad 0 \quad G^{6.2} \quad (50)$$

Ao chegar ao fim do processamento de um conjunto de estados resultando em um estado S_{i+1} contendo apenas o estado 50, o algoritmo se encerra indicando a aceitação da cadeia de entrada.

4.3.2 Estratégia de implementação do algoritmo

A implementação anterior do algoritmo de Earley no GrammarLab possuía muito acoplamento com outras funções disponibilizadas no arcabouço, como a consideração de probabilidades associadas com as produções (o que não está sendo considerado neste momento), uso de variáveis globais e dependência de comportamentos estáticos da gramática.

Como a evolução do arcabouço está planejada para ser realizada de forma gradual, sem comprometimento das funcionalidades, foi adotada nesta pesquisa a estratégia de

implementação de uma versão não probabilística do algoritmo, conforme proposta original (EARLEY, 1970) em um novo conjunto de classes, conforme Figura 34.

Figura 34 – GrammarLab - Algoritmo de Earley Adaptativo

Fonte: Gilmar Pereira dos Santos, 2018

A classe *AdaptiveEarleyParser* é responsável pela preparação da cadeia de entrada, inicialização e processamento de cada um dos conjuntos de estados (implementados pela classe *AdaptiveParserState*) e pela verificação, ao término do processamento de todos os conjuntos de estado, se a sequência de entrada foi aceita ou rejeitada pela gramática.

A classe *AdaptiveParserState* modela um conjunto de estados e é onde são implementadas as operações *predictor*, *scanner* e *completer*. Esta classe é composta por 0 ou mais de estados, modelados pela classe *AdaptiveParserStateItem*.

A classe *AdaptiveParserStateItem*, que modela um estado, contém uma referência para a gramática adaptativa atual (classe *AdaptiveGrammar*), para a uma regra de produção (classe *AdaptiveRule*) e para a classe *AdaptiveParserStateNode*.

A classe *AdaptiveParserStateNode* é a responsável por armazenar as informações necessárias para retornar as possíveis árvores de derivação ao término do processamento do algoritmo. Esta classe contém a informação de derivação de um estado de análise, podendo indicar a derivação de um símbolo terminal (representado por uma instância da classe *Terminal*, de um símbolo não terminal (representado por uma instância de uma classe *AdaptiveParserStateItem*). Quando existe mais de uma árvore de derivação

possível, ou seja, quando o objeto da análise sintática é uma gramática ambígua, a classe *AdaptiveParserStateNode* possui uma outra instância do tipo *AdaptiveParserStateNode* para cada derivação alternativa partindo do estado de análise atual.

Analisando a estratégia de implementação anterior do algoritmo e confrontando tais dados com a proposta, o primeiro problema encontrado é o cálculo antecipado da cadeia *lookahead*. Em uma gramática livre de contexto, as possíveis derivações de cada símbolo não terminal podem ser calculadas antes mesmo do processo de análise iniciar, podendo o resultado ser armazenado em um conjunto de ponteiros indexados pelos símbolos não terminais. Em uma gramática adaptativa, no entanto, como as regras de produção podem ser incluídas ou removidas dependendo da ordem em que forem sendo ativadas as funções adaptativas associadas, o processo de identificação das cadeias *lookahead* se torna muito custoso.

Como o objetivo da inclusão de regras adaptativas é a resolução de cenários de análise mais complexos não atendidos pelas gramáticas livres de contexto, o algoritmo resultante deste trabalho não considera o cálculo das sequências de símbolos *lookahead* (ou seja, é considerado $k = 0$ fixo). Isso implica, conforme descrito em (EARLEY, 1970), que o algoritmo resultante não resolverá gramáticas em um tempo linear, ficando a complexidade, no caso geral, em $O(n^3)$.

O Algoritmo 2 apresenta a versão adaptativa do algoritmo de Earley proposto e implementado nesta pesquisa. São consideradas as seguintes notações adicionais, considerando uma regra de produção no formato $S \rightarrow \{pre(...)\}\gamma\{pos(...)\}$, sendo $\{pre(...)\}$ uma função adaptativa que é executada antes da ativação da regra de produção $S \rightarrow \gamma$ e $\{pos(...)\}$ a função adaptativa que é executada após a ativação de tal regra:

- $Pre_n(G)$ representa a aplicação da função adaptativa $\{pre(...)\}$ presente na regra de produção n sobre a gramática G^i , gerando uma gramática G^{i+1}
- $Pos_n(G)$ representa a aplicação da função adaptativa $\{pos(...)\}$ presente na regra de produção n sobre a gramática G^i , gerando uma gramática G^{i+1}

Algoritmo 2 Algoritmo de Earley Adaptativo para reconhecimento de gramáticas adaptativas

```

1: function ADAPTIVEEARLEYPARSER( $G^0, X_1 \dots X_n$ )
2:    $X_{n+1} = \perp$ 
3:    $S_i = \emptyset$  ( $0 \leq i \leq n + 1$ )
4:    $S_0 = \{ \langle 0, 0, 0, G^0, \emptyset \rangle \}$ 
5:   for  $i \leftarrow 0$  até  $n$  do
6:     for all  $s = \langle p, j, f, G, B \rangle \in S_i$  do
7:       if  $s \neq s_{fim}$  e  $C_{p(j+1)} \in N$  then ▷ 1 - Predictor
8:         for all  $q \mid C_{p(j+1)} = D_q$  do
9:            $New_G = Pos_q(Pre_q(G))$ 
10:           $S_i \leftarrow S_i + \{ \langle q, 0, i, New_G, \emptyset \rangle \}$ 
11:        if  $s = s_{fim}$  then ▷ 2 - Completer
12:          for all  $\langle q, l, g, G, E \rangle \in S_f$  do
13:            if  $C_{q(l+1)} = D_p$  then
14:               $E \leftarrow B_1 \dots B_l$ 
15:               $E_{l+1} \leftarrow$  ponteiro para  $\langle p, \bar{p}, f, G, B \rangle$ 
16:               $S_i \leftarrow S_i + \{ \langle q, l + 1, g, G, E \rangle \}$ 
17:          for all  $s = \langle p, j, f, G \rangle \in S_i$  do
18:            if  $s \neq s_{fim}$  e  $C_{p(j+1)} \in T$  then ▷ 3 - Scanner
19:              if  $C_{p(j+1)} = X_{i+1}$  then
20:                 $E \leftarrow B_1 \dots B_j$ 
21:                 $E_{j+1} \leftarrow X_{i+1}$ 
22:                 $S_{i+1} \leftarrow S_{i+1} + \{ \langle p, j + 1, f, G, E \rangle \}$ 
23:          if  $S_{i+1} = \emptyset$  then
24:            retornar Rejeição
25:          if  $i = n$  e  $S_{i+1} = \{ \langle 0, 2, 0, (\text{qualquer } G), (\text{qualquer } B) \rangle \}$  then
26:            retornar Aceitação

```

Fonte: Gilmar Pereira dos Santos, 2018

Resultado da execução do algoritmo

O apêndice A descreve uma saída gerada pela execução do algoritmo de Earley Adaptativo desenvolvido neste trabalho, utilizando como entrada a gramática adaptativa e a cadeia de exemplo, ambas apresentadas na seção 2.1.6, o mesmo exemplo apresentado de forma teórica na seção 4.3.1. A notação utilizada para representar cada estado analisado é uma adaptação da notação BNF para gramáticas adaptativas (seção 4.2).

Na sequência de execução pode ser notada a ausência do elemento *lookahead* e a presença de estados de análise adicionais (além dos apresentados na demonstração teórica da seção 4.3.1). Também são apresentadas as evoluções da gramática nos passos da análise

sintática. A árvore sintática, também apresentada nesse apêndice, foi gerada percorrendo os ponteiros B do estado final (Algoritmo 2).

O exemplo seguinte mostra a habilidade do algoritmo em identificar todas as árvores de derivação possíveis de uma sequência para uma gramática ambígua. Neste exemplo, foi considerada a gramática livre de contexto $G = (\{S, a\}, \{a\}, P, S)$, sendo P o seguinte conjunto de produções:

- $S \rightarrow a$
- $S \rightarrow Sa$
- $S \rightarrow aSa$

Como teste, foi utilizada a sequência aaa . Esta sequência pode ser gerada de duas formas: (i) $S \rightarrow aSa, S \rightarrow a$ e (ii) $S \rightarrow Sa, S \rightarrow Sa$ e $S \rightarrow a$.

1: aaa => ACEITA

Árvore Sintática:

<@>

variante 0

<S>

a

<S>

a

a

variante 1

<S>

<S>

<S>

a

a

a

#

5 APLICAÇÃO: CARACTERIZAÇÃO DE FAMÍLIAS DE RNAs COM PSEUDONÓS

Como forma de verificar o potencial do uso da nova versão do arcabouço GrammarLab em problemas práticos, foi realizado neste trabalho um estudo preliminar no contexto do problema de busca de famílias funcionais de RNAs com pseudonós em sua estrutura secundária, tema que motivou a inclusão de adaptatividade no GrammarLab. As famílias de interesse serão descritas por gramáticas adaptativas livres de contexto na notação BNF elaborada na seção 4.2. A busca por estas famílias foi realizada na base de dados pública RFAM (KALVARI *et al.*, 2017) (<http://rfam.xfam.org>), como descrito na seção 5.1.1.

Neste capítulo será apresentada a estratégia utilizada para a execução dos testes e serão comentados os resultados obtidos na busca por duas famílias de RNAs.

5.1 Estratégia de testes

Como estratégia de testes, foram seguidos os seguintes passos:

- Seleção de famílias de RNAs com pseudonós anotados na estrutura secundária;
- Modelagem da estrutura consenso de cada uma das famílias selecionadas na forma de uma gramática adaptativa, de forma que sejam capturadas as características estruturais das sequências curadas destas famílias;
- Realização da busca de uma determinada família em bases de dados compostas por sequências curadas e preditas pelo software do RFAM.

5.1.1 Seleção de famílias com pseudonós

Para a seleção das famílias de interesse, foi desenvolvido um programa em python com o objetivo de percorrer todas as estruturas secundárias presentes na base RFAM em busca de anotações de pseudonós, conforme padrão apresentado na seção 2.2.2. O resultado das famílias encontradas com pseudonós é apresentado na Tabela 2, contendo na primeira coluna o código RFAM da família, na segunda coluna sua descrição e na terceira coluna o tamanho da estrutura consenso da família em termos do número de caracteres.

Das famílias encontradas, foram selecionadas para o estudo de caso a família RF00505, por apresentar uma estrutura bem definida de um pseudonó e baixa complexidade

Tabela 2 – Famílias de RNAs com pseudonós na estrutura secundária

Código	Descrição	Tamanho
RF00505	RydC RNA	64
RF02679	Pistol ribozyme	70
RF00622	Mammalian CPEB3 ribozyme	78
RF00094	Hepatitis delta virus ribozyme	91
RF01775	RNA S.aureus Orsay G	148
RF01807	GIR1 branching ribozyme	199
RF00030	RNase MRP	266
RF00009	Nuclear RNase P	303
RF00373	Archaeal RNase P	303
RF01850	Betaproteobacteria transfer-messenger RNA	331
RF01849	Alphaproteobacteria transfer-messenger RNA	338
RF00023	transfer-messenger RNA	354
RF00011	Bacterial RNase P class B	366
RF00010	Bacterial RNase P class A	367
RF00024	Vertebrate telomerase RNA	436
RF01577	Plasmodium RNase P	626
RF01050	Saccharomyces telomerase	1208

Fonte: Gilmar Pereira dos Santos, 2018

estrutural e a família RF00024, selecionada pela alta complexidade estrutural típica do componente RNA da telomerase, famílias estas destacadas em negrito na Tabela 2.

5.1.2 Análise sintática das sequências do Rfam à procura das famílias selecionadas

Para a realização das buscas por famílias específicas, foi elaborada uma linha de execução de acordo com a Figura 35. Este processo recebe como entradas um arquivo com a definição da gramática adaptativa que descreve o objeto da busca, isto é, a família alvo, e um arquivo contendo uma lista de sequências que serão confrontadas com a gramática informada.

Como passo inicial, a lista de sequências (base de dados) é dividida em n partes iguais, sendo n configurável, viabilizando assim o paralelismo da execução e garantia de liberação de memória após o processamento de n sequências. Após o particionamento, cada fração do arquivo é processada, sendo cada uma das sequências avaliadas como fazendo ou não parte da linguagem descrita pela gramática informada.

O RFAM possui, para cada família, um conjunto de sequências curadas e um conjunto de sequências não curadas. As sequências curadas são aquelas que passaram por algum tipo de validação que comprovem que são realmente daquela família, enquanto as

Figura 35 – Processo de busca

Fonte: Gilmar Pereira dos Santos, 2018

não curadas são aquelas provenientes de predição computacional pelo software do RFAM. As gramáticas representando as duas famílias selecionadas foram criadas utilizando como base as sequências curadas. A fim de permitir o cálculo de estimativas relacionadas às taxas de falso positivo e falso negativo, a análise sintática foi executada em três cenários:

- busca nas sequências curadas da família alvo da busca (amostra positiva confiável);
- busca sequências preditas da família alvo da busca (a fim de estimar a intersecção de resultados positivos entre a gramática adaptativa e o software do RFAM);
- busca na base de todas as sequências curadas de todas as demais famílias (amostra negativa confiável).

Foi ainda aplicada a técnica de validação cruzada *2-fold* de estimação de desempenho para uma das famílias selecionadas, como forma de verificar o quão sensível é a abordagem utilizada para modelagem em relação ao conjunto de treinamento selecionado.

Para avaliar os resultados, foram utilizadas as métricas de taxa de erro (E), precisão (P) e revocação ou sensibilidade (R). Considerando:

- FP – Quantidade de falsos positivos (número de sequências curadas de outras famílias aceitas pela gramática alvo)
- FN – Quantidade de falsos negativos (número de sequências curadas da família alvo rejeitadas pela gramática alvo)

5.2.1 Estratégia de Modelagem

Para realizar a modelagem da gramática adaptativa que descreve a família RF00505, foi levada em consideração a estrutura secundária consenso do RNA disponibilizada no RFAM (Figura 37), sendo utilizadas as sequências curadas da família para validação dos pareamentos de bases.

Figura 37 – RydC RNA - alinhamento múltiplo das sequências curadas

```
U00096.2/1489530-1489467 CUUCCGAUGUAGACCCGU-AUUCUUCGCCUGUACCACGGGUC-GGUUUUAGUACAGGCGUUUUCU
AL627270.1/77020-76955 CUUCCGAUGUAGACCCGUCCUCCUUCGCCUGCGUCACGGGUCUGGUUAGACGCAGGCGUUUUCUG
CP000038.1/1826318-1826381 CUUCCGAUGUAGACCCGU-AUUCUUCGCCUGUACCACGGGUC-GGUUUUAGUACAGGCAUUUUCU
CU928161.2/1508420-1508357 CUUCCGAUGUAGGCCCGU-AUUCUUCGCCUGUACCACGGGUC-GGUUUUAGUACAGGCGUUUUCU
CP001113.1/1726026-1726091 CUUCCGAUGUAGACCCGUCCUCCUUCGCCUGCGUCACGGGUCUUGGUUAGACGCAGGCGUUUUCUG
#=GC SS_cons :::::::::: <<<<<<. _____AAAAAAAA_>>>>>>.: ::::: aaaaaaaaaa: :::::
```

Fonte: <http://rfam.xfam.org>

A gramática adaptativa modelada é apresentada no Apêndice B. Nessa gramática é utilizada uma função adaptativa *Ad1* que adiciona uma regra de produção do tipo $X \rightarrow y$, sendo X um símbolo não terminal passado por parâmetro para a função e y um símbolo terminal da gramática, também passado por parâmetro. Tal função adaptativa permite que sejam modeladas dependências entre nucleotídeos em distâncias arbitrárias, inclusive com cruzamentos, permitindo modelar os pseudonós (semelhante ao exemplo apresentado na seção 4.3.1).

5.2.2 Resultados

A busca pelos elementos da família RF00505 realizada na amostra positiva, ou seja, no conjunto formado pelas sequências curadas da própria família (total de cinco sequências), resultou na identificação correta de todas as sequências positivas confiáveis, isto é, $VP = 5$ e $FN = 0$. Já a busca realizada na amostra negativa, ou seja, no conjunto de todas as sequências curadas de todas as demais famílias presentes na base RFAM, resultou na rejeição das mesmas, ou seja, $VN = 47853$ e $FP = 0$. Considerando este conjunto de teste, a gramática adaptativa obteve o desempenho apresentado na Tabela 3.

Para verificar a capacidade da gramática adaptativa elaborada de analisar sequências não curadas (preditas pelo software de detecção do RFAM) e comparar seus resultados com os do RFAM, foi realizada uma busca considerando como base de dados o conjunto

Tabela 3 – Desempenho da gramática adaptativa quando aplicada a sequências curadas de todas as famílias

	Medida (%)
Taxa de Erro	0
Precisão	100
Revocação	100

Fonte: Gilmar Pereira dos Santos, 2018

de cinco sequências preditas¹, classificadas como pertencentes à família RF00505 na base RFAM. Considerando este novo conjunto, 4 sequências foram aceitas, o que corresponde a 80% do total. Analisando a estrutura secundária consenso, as sequências curadas e as sequências preditas que foram aceitas, pode ser verificado que é esperado que tamanho da sequência varie entre 64 e 66 nucleotídeos, o que justamente não ocorre na sequência que foi rejeitada, a qual possui um tamanho de 63 nucleotídeos.

Na Figura 38 é apresentado um alinhamento entre a sequência que foi rejeitada e as sequências curadas (sequência rejeitada na última linha, abaixo da linha referente à estrutura consenso), sendo destacados os nucleotídeos (ou ausências) que provocaram a rejeição da sequência pela gramática adaptativa. Dentre estes, nota-se a alteração de três nucleotídeos, dois deles afetando uma região pareada. A ausência de dois nucleotídeos e a alteração de um outro ocorreram em posições altamente conservadas. É importante ressaltar que essa é uma sequência predita e não curada, que poderia não representar uma molécula real. Embora não validada, a sequência provavelmente é um elemento pertencente à família, devido à sua alta similaridade com as sequências curadas desta família tanto no nível de sequência quanto no nível estrutural. Tal sequência foi rejeitada devido à estratégia conservadora de modelagem adotada neste trabalho inicial, em que posições altamente conservadas só aceitariam os nucleotídeos apresentados pelas sequências curadas naquelas respectivas posições e regiões pareadas consenso seriam obrigatórias. Tal estratégia, utilizada neste momento apenas para uma avaliação preliminar da aplicabilidade de gramáticas adaptativas neste contexto de RNAs com pseudonós, será revista conforme abordado na seção 6.3.

Se considerarmos positivas todas as cinco sequências preditas, somando-se às cinco sequências curadas, a gramática adaptativa teria alcançado 90% de revocação e 100% de precisão.

¹ Na base RFAM são encontradas 38 sequências preditas para a família RF00505, porém existem duplicidades, resultando em cinco sequências únicas.

Figura 39 – Estrutura secundária do componente RNA da telomerase de vertebrados. O *template* é uma região não pareada utilizada como molde da polimerização das repetições teloméricas encontradas nas extremidades dos cromossomos.

Fonte: Adaptado de Chen e Greider (2004)

5.3.1 Estratégia de Modelagem

A estratégia adotada para a modelagem manual de uma gramática adaptativa que descrevesse a família RF00024 foi semelhante à apresentada na modelagem da família RF00505. Na Tabela 4 são apresentadas estatísticas sobre o tamanho das sequências curadas dos TERCs.

Tabela 4 – Estatísticas do tamanho das sequências curadas. $\bar{x}$ e s são, respectivamente, a média e o desvio padrão da amostra

Quantidade	Tamanho mínimo	Tamanho máximo	$\bar{x}$ tamanho	s tamanho
37	382	559	444,65	37,59

Fonte: Gilmar Pereira dos Santos, 2018

A gramática adaptativa modelada que representa a estrutura consenso da família é apresentada no Apêndice C.

Para a realização da validação cruzada as sequências curadas foram divididas em dois conjuntos A e B , sendo os elementos de cada conjunto selecionados aleatoriamente e a estrutura consenso resultante de cada um dos conjuntos modelada individualmente, gerando assim mais duas gramáticas adaptativas. Cada uma dessas gramáticas também representa a família RF00024. Assim, a gramática modelada para o conjunto A foi utilizada para realizar a busca pela família RF00024 nos elementos do conjunto B , e a gramática modelada para o conjunto B foi utilizada para realizar a busca nos elementos do conjunto A . Ambas as gramáticas foram utilizadas para analisar a amostra negativa (demais famílias)

5.3.2 Resultados

De forma análoga ao procedimento realizado com a família RF00505, a busca pelos elementos da família RF00024 no conjunto formado pelas sequências curadas da própria família (total de 37 sequências) resultou na aceitação de todas as sequências curadas desta família e na rejeição de todas as sequências curadas das demais famílias. Assim, o desempenho da gramática adaptativa elaborada obteve o desempenho apresentado na Tabela 5.

Tabela 5 – Desempenho da gramática adaptativa quando aplicada a sequências curadas de todas as famílias

	Medida (%)
Taxa de Erro	0
Precisão	100
Revocação	100

Fonte: Gilmar Pereira dos Santos, 2018

Na tabela 6 é apresentado o tempo de CPU gasto pelo algoritmo de Earley durante a execução da busca.

Tabela 6 – Algoritmo de Earley adaptativo - tempo de CPU

Amostra	Quantidade	Média Tempo (s)	Mín. Tempo (s)	Máx. Tempo (s)
POSITIVA	37	1254,359	536, 657	1525,662
NEGATIVA	47821	0,303	0,007	10,206

Fonte: Gilmar Pereira dos Santos, 2018

Foi também realizada uma busca considerando como base de dados um conjunto de 48 sequências preditas², classificadas como pertencentes à família RF00024 na base RFAM. Desta busca, 22 sequências foram aceitas, o que corresponde a 46% do total.

Analisando a Tabela 4, pode ser verificado que é esperado (dada a estratégia de modelagem manual da gramática) que o tamanho da sequência varie entre 382 e 559 nucleotídeos. Comparando este dado com os tamanhos das 25 sequências preditas que foram rejeitadas pela gramática, são encontradas 16 sequências que caem fora desse intervalo.

Como as sequências utilizadas no teste são preditas e não validadas, não pode ser afirmado com certeza que houve 54% de erro no teste na amostra, pois algumas sequências podem realmente ser falsas. No entanto, é possível que muitas destas, ou até todas elas, sejam verdadeiras e não tenham sido aceitas devido à abordagem conservadora da modelagem das gramáticas, conforme discutido na seção 5.2.2. Mas mesmo ao considerarmos positivas todas as 48 sequências preditas, somando-se às 37 sequências curadas, a gramática adaptativa teria alcançado 69% de revocação e 100% de precisão, apesar da estratégia conservadora da modelagem.

A tabela 7 apresenta o resultado da validação cruzada, mostrando que uma pequena alteração no conjunto de treinamento pode afetar consideravelmente a taxa de revocação devido ao *overfitting* causado pela estratégia de modelagem conservadora.

Tabela 7 – Desempenho da validação cruzada simplificada realizada com a família RF00024

	Gramática <i>A</i> (%)	Gramática <i>B</i> (%)
Taxa de Erro	32	24
Precisão	100	100
Revocação	68	76

Fonte: Gilmar Pereira dos Santos, 2018

² Na base RFAM são encontradas 65 sequências preditas para a família RF00024, porém existem duplicidades e anotações diferentes do alfabeto gênico, resultando em 48 sequências únicas.

6 CONCLUSÃO

6.1 Considerações finais

O GrammarLab, um arcabouço de geração de classificadores de sequências baseado em gramáticas, em sua versão anterior a este trabalho, somente podia lidar com problemas limitados às restrições impostas pelas gramáticas livres de contexto estocásticas, que eram a base do arcabouço. Neste trabalho foi evoluída a hierarquia de classes utilizadas para modelar as gramáticas do arcabouço, inserindo adaptatividade em sua constituição. A estratégia adotada para a implementação de gramáticas adaptativas não inviabiliza o uso das demais gramáticas não adaptativas, pois a adaptatividade foi inserida por meio da especialização das classes existentes (heranças) e pela criação de novas classes para os novos comportamentos (funções adaptativas associadas a produções).

Para representar a nova classe de gramáticas, foi desenvolvida nesse trabalho uma notação, inspirada na notação BNF, para descrever tanto as regras quanto os comportamentos adaptativos (funções adaptativas) das gramáticas livres de contexto estocásticas, permitindo o uso de descrições textuais das gramáticas em arquivos externos.

Para converter uma gramática descrita nesta notação em uma instância de uma gramática adaptativa, foi também evoluída a hierarquia de classes *stream*, sendo desenvolvido um interpretador para a notação BNF adaptativa. Como forma de generalizar o uso do arcabouço, foi desenvolvida uma classe *Factory* que, dada a classe gramatical informada no cabeçalho do arquivo BNF, instancia uma gramática da classe adequada, seja ela adaptativa ou não.

Para permitir que o GrammarLab realize análise sintática com a nova classe de gramáticas, foi desenvolvida uma versão adaptativa do algoritmo de Earley, permitindo a análise sintática com complexidade da ordem de $O(n^3)$ mesmo de gramáticas que apresentam elementos com dependência de contexto em sua constituição.

Como forma de realizar uma análise preliminar do potencial uso da nova versão do arcabouço em problemas reais, foi utilizado o problema de bioinformática de caracterização de famílias funcionais de RNAs com pseudonós, problema que motivou este trabalho. Para isto foram utilizadas as famílias RF00505 e RF00024 do banco de dados RFAM como casos de estudo.

Para cada família, a gramática adaptativa modelada manualmente e de forma bastante conservadora, foi capaz de aceitar todas as sequências curadas de suas respectivas famílias e rejeitar todas as sequências curadas das demais famílias, o que significa uma taxa de 100% tanto de precisão quanto de revocação. Cabe no entanto uma observação importante nesta taxa estimada de revocação. Como esta métrica foi calculada com base nos resultados da análise sintática das sequências curadas, as mesmas utilizadas para modelar a gramática, essa alta taxa de sucesso em identificar casos positivos foi superestimada.

Como forma de contornar este problema, foram analisadas as sequências preditas pelo software do RFAM. No teste envolvendo as sequências preditas para a família RF00505, que possui baixa complexidade estrutural, foi atingida uma taxa de aceitação de 80% das sequências, sendo a rejeição provocada pela ausência de um nucleotídeo em uma região de hélice pois, como o mesmo está presente em todas as sequências curadas, a gramática modelada exige tal presença. No teste envolvendo sequências preditas para a família RF00024 (componente RNA da telomerase), de maior complexidade estrutural, foi atingida uma taxa de 46% de aceitação, novamente devido à estratégia conservadora adotada na modelagem da gramática. Isso, porém, não significa que houve uma taxa de erro de 54%, pois como as sequências são preditas e não validadas, não há garantia que sejam todas as sequências verdadeiras. Estes resultados mostram que um pequeno ajuste na modelagem da gramática já poderia aumentar a taxa de revocação.

Considerando que os testes realizados representam uma avaliação preliminar do potencial de uso dos dispositivos adaptativos no problema de caracterização de famílias de RNAs com pseudonós, os resultados demonstram que o uso do arcabouço GrammarLab Adaptativo é muito promissor.

Vale ressaltar que os conceitos desenvolvidos neste trabalho, aliados à consolidação de informações em um local único sobre linguagens formais, dispositivos adaptativos e a demonstração do potencial de uso em problemas reais torna este trabalho uma possível referência para um leque de possíveis pesquisas futuras.

6.2 Principais contribuições

Em resumo, as principais contribuições deste trabalho foram:

- Desenvolvimento de uma versão adaptativa do algoritmo de Earley;

- Evolução do arcabouço GrammarLab para inclusão de adaptatividade;
- Desenvolvimento de uma notação BNF-*like* para representar gramáticas adaptativas;
- Testes e obtenção de bons resultados no uso de gramáticas adaptativas na caracterização de pseudonós;
- Revisão bibliográfica dos métodos computacionais de caracterização de famílias de RNAs.

6.3 *Trabalhos futuros*

Como a modelagem manual de gramáticas que representem uma determinada família funcional de RNA pode se tornar um processo extremamente trabalhoso e propenso a erros, principalmente em famílias maiores, torna-se interessante a ideia do desenvolvimento de uma **ferramenta que automatize o processo de geração de gramáticas adaptativas**, utilizando tanto as referências de estruturas secundárias documentadas quanto, principalmente, o conhecimento de especialistas. Assim, um trabalho futuro é o desenvolvimento de uma linguagem que permita que o especialista descreva uma família de RNAs por meio de informações estruturais e subsequências conservadas, e de um programa que, a partir desta informação textual alto-nível, gere a gramática adaptativa correspondente no formato BNF proposto neste trabalho.

A implementação atual do algoritmo de Earley Adaptativo gera as árvores de derivação para uma gramática ambígua em uma forma fatorada. Deve ser explorada uma forma de evoluir esta característica para **permitir a geração de todas as árvores de derivação na íntegra** (por completo, individualmente) sem, no entanto, prejudicar o desempenho de execução.

Como forma de verificar formalmente a capacidade e limites do algoritmo proposto e desenvolvido, a **análise de complexidade** com foco no **estudo do determinismo** do algoritmo de Earley Adaptativo também é um trabalho futuro que deve ser desenvolvido.

Para lidar com a rigidez imposta por uma modelagem baseada em um conjunto limitado de sequências curadas, um outro trabalho futuro derivado desta pesquisa é a **conversão da gramática adaptativa** aqui desenvolvida **em sua versão estocástica**. Com isto pode-se relaxar substancialmente as regras sintáticas uma vez que pode-se contar com a caracterização da linguagem por meio das probabilidades associadas a essas regras de produção.

Além disso, a **comparação de desempenho** entre o algoritmo de Earley Adaptativo e demais *parsers* sensíveis ao contexto também é um estudo futuro.

6.4 Artigos científicos

O artigo “Algoritmo de Earley Adaptativo para Reconhecimento de Padrões Sintáticos”, descrevendo a proposta do analisador de Earley adaptativo, foi publicado no WTA 2018 (Workshop de Tecnologias Adaptativas).

O artigo com os resultados totais do trabalho, incluindo os testes com famílias de RNAs, foi submetido ao periódico *Theoretical Computer Science*, Qualis A1 em Ciência da Computação.

O capítulo de revisão bibliográfica também apresenta potencial para ser evoluído e servir de base para um artigo.

Referências¹

- AHO, A. V.; LAM, M. S.; SETHI, R.; ULLMAN, J. D. Compilers: Principles, Tools, and Techniques. [S.l.]: Addison-Wesley, 1986. v. 2. Citado na página 26.
- ALLALI, J.; SAGOT, M.-F. A multiple layer model to compare RNA secondary structures. Software: Practice and Experience, v. 38, n. 8, p. 775–792, 2008. Citado na página 61.
- ANDRONESCU, M.; BEREG, V.; HOOS, H. H.; CONDON, A. RNA STRAND: the RNA secondary structure and statistical analysis database. BMC bioinformatics, BioMed Central, v. 9, n. 1, p. 340, 2008. Citado na página 57.
- ANTAL, M.; BORDEAU, V.; DOUCHIN, V.; FELDEN, B. A small bacterial RNA regulates a putative ABC transporter. Journal of Biological Chemistry, ASBMB, v. 280, n. 9, p. 7901–7908, 2005. Citado na página 91.
- BARQUIST, L.; BURGE, S. W.; GARDNER, P. P. Studying RNA homology and conservation with infernal: from single sequences to RNA families. Current Protocols in Bioinformatics, Wiley Online Library, p. 12–13, 2016. Citado na página 50.
- BATENBURG, F. V.; GULTYAEV, A. P.; PLEIJ, C.; NG, J.; OLIEHOEK, J. PseudoBase: a database with RNA pseudoknots. Nucleic Acids Research, Oxford Univ Press, v. 28, n. 1, p. 201–204, 2000. Citado na página 57.
- BERMAN, H. M.; WESTBROOK, J.; FENG, Z.; GILLILAND, G.; BHAT, T. N.; WEISSIG, H.; SHINDYALOV, I. N.; BOURNE, P. E. The protein data bank. Nucleic acids research, Oxford Univ Press, v. 28, n. 1, p. 235–242, 2000. Citado na página 60.
- BROWN, J. W. The ribonuclease P database. Nucleic acids research, Oxford University Press, v. 26, n. 1, p. 351–352, 1998. Citado na página 57.
- BROWN, M.; WILSON, C. RNA pseudoknot modeling using intersections of stochastic context free grammars with applications to database search. In: Pacific Symposium on Biocomputing. Pacific Symposium on Biocomputing. [S.l.: s.n.], 1995. p. 109–125. Citado 3 vezes nas páginas 14, 47 e 54.
- CANNONE, J. J.; SUBRAMANIAN, S.; SCHNARE, M. N.; COLLETT, J. R.; D’SOUZA, L. M.; DU, Y.; FENG, B.; LIN, N.; MADABUSI, L. V.; MÜLLER, K. M. *et al.* The comparative RNA web (crw) site: an online database of comparative sequence and structure information for ribosomal, intron, and other RNAs. BMC bioinformatics, BioMed Central, v. 3, n. 1, p. 2, 2002. Citado na página 65.
- CHEN, J.-L.; GREIDER, C. W. An emerging consensus for telomerase RNA structure. Proceedings of the National Academy of Sciences of the United States of America, National Acad Sciences, v. 101, n. 41, p. 14683–14684, 2004. Citado 3 vezes nas páginas 45, 46 e 95.
- CHIU, J. K.; CHEN, Y. P. Pairwise RNA secondary structure alignment with conserved stem pattern. Bioinformatics, v. 31, n. 24, p. 3914–21, 2015. ISSN 1367-4803. Citado na página 61.

¹ De acordo com a Associação Brasileira de Normas Técnicas. NBR 6023.

- CHOMSKY, N. On certain formal properties of grammars. Information and control, Elsevier, v. 2, n. 2, p. 137–167, 1959. Citado 2 vezes nas páginas 14 e 18.
- CLANCY, S.; BROWN, W. Translation: DNA to mRNA to protein. Nature Education, v. 1, n. 1, p. 101, 2008. Citado na página 39.
- CLANCY, S. *et al.* Chemical structure of RNA. Nature Education, v. 1, n. 1, p. 223, 2008. Citado 2 vezes nas páginas 39 e 40.
- COSTA, F. F. Non-coding RNAs: new players in eukaryotic biology. Gene, Elsevier, v. 357, n. 2, p. 83–94, 2005. Citado na página 39.
- DEBLASIO, D.; BRUAND, J.; ZHANG, S. A memory efficient method for structure-based RNA multiple alignment. IEEE/ACM Trans Comput Biol Bioinform, v. 9, n. 1, p. 1–11, 2012. ISSN 1545-5963. Citado na página 64.
- DIRKS, R. M.; PIERCE, N. A. An algorithm for computing nucleic acid base-pairing probabilities including pseudoknots. Journal of computational chemistry, Wiley Online Library, v. 25, n. 10, p. 1295–1304, 2004. Citado na página 62.
- DIXON, M. T.; HILLIS, D. M. Ribosomal RNA secondary structure: compensatory mutations and implications for phylogenetic analysis. Molecular Biology and Evolution, SMOE, v. 10, n. 1, p. 256–267, 1993. Citado 2 vezes nas páginas 15 e 40.
- DURBIN, R.; EDDY, S. R.; KROGH, A.; MITCHISON, G. Biological sequence analysis: probabilistic models of proteins and nucleic acids. [S.l.]: Cambridge university press, 1998. Citado na página 25.
- EARLEY, J. An efficient context-free parsing algorithm. Communications of the ACM, ACM, v. 13, n. 2, p. 94–102, 1970. Citado 7 vezes nas páginas 14, 27, 28, 30, 32, 84 e 85.
- EDDY, S. Infernal user's guide. Disponível em <http://infernal.janelia.org>, 2003. Citado 4 vezes nas páginas 44, 51, 52 e 53.
- EDDY, S. R.; DURBIN, R. RNA sequence analysis using covariance models. Nucleic acids research, Oxford Univ Press, v. 22, n. 11, p. 2079–2088, 1994. Citado na página 50.
- FUYAMA, S. Syntactic Pattern Recognition and Application. [S.l.]: New Jersey Prentice-Hall, Inc, 1982. Citado na página 31.
- GRIFFITHS-JONES, S.; BATEMAN, A.; MARSHALL, M.; KHANNA, A.; EDDY, S. R. Rfam: an RNA family database. Nucleic acids research, Oxford Univ Press, v. 31, n. 1, p. 439–441, 2003. Citado na página 43.
- GRUBER, A. R.; LORENZ, R.; BERNHART, S. H.; NEUBÖCK, R.; HOFACKER, I. L. The vienna RNA websuite. Nucleic acids research, Oxford Univ Press, v. 36, n. suppl 2, p. W70–W74, 2008. Citado na página 62.
- HORESH, Y.; DONIGER, T.; MICHAELI, S.; UNGER, R. RNAspa: a shortest path approach for comparative prediction of the secondary structure of ncRNA molecules. BMC bioinformatics, BioMed Central, v. 8, n. 1, p. 366, 2007. Citado 2 vezes nas páginas 63 e 67.

HUANG, J.; LI, K.; GRIBSKOV, M. Accurate classification of RNA structures using topological fingerprints. PLoS One, v. 11, n. 10, p. e0164726, 2016. ISSN 1932-6203. Citado 2 vezes nas páginas 55 e 56.

ISHII, N.; OZAKI, K.; SATO, H.; MIZUNO, H.; SAITO, S.; TAKAHASHI, A.; MIYAMOTO, Y.; IKEGAWA, S.; KAMATANI, N.; HORI, M. *et al.* Identification of a novel non-coding RNA, MIAT, that confers risk of myocardial infarction. Journal of human genetics, Springer, v. 51, n. 12, p. 1087–1099, 2006. Citado na página 39.

IWAI, M. K. Um formalismo gramatical adaptativo para linguagens dependentes de contexto. Departamento de Computação e Sistemas Digitais (PCS)-Escola Politécnica, Tese de Doutorado, Escola Politécnica, Universidade de São Paulo (USP), São Paulo, SP (in portuguese), 2000. Citado 3 vezes nas páginas 32, 33 e 36.

JAIN, A. K.; DUIN, R. P. W.; MAO, J. Statistical pattern recognition: A review. IEEE Transactions on pattern analysis and machine intelligence, IEEE, v. 22, n. 1, p. 4–37, 2000. Citado na página 14.

JIANG, Y.; XU, W.; THOMPSON, L. P.; GUTELL, R. R.; MIRANKER, D. P. R-PASS: A fast structure-based RNA sequence alignment algorithm. Proceedings (IEEE Int Conf Bioinformatics Biomed), v. 2011, p. 618–622, 2011. ISSN 2156-1125 (Print) 2156-1125. Citado na página 65.

JUNQUEIRA, L. C.; CARNEIRO, J. Biologia celular e molecular. In: Biologia Celular e Molecular. [S.l.]: Guanabara Koogan, 2015. Citado na página 39.

KALVARI, I.; ARGASINSKA, J.; QUINONES-OLVERA, N.; NAWROCKI, E. P.; RIVAS, E.; EDDY, S. R.; BATEMAN, A.; FINN, R. D.; PETROV, A. I. Rfam 13.0: shifting to a genome-centric resource for non-coding RNA families. Nucleic acids research, Oxford University Press, v. 46, n. D1, p. D335–D342, 2017. Citado na página 88.

KORBI, A. E.; OUELLET, J.; NAGHDI, M. R.; PERREAULT, J. Finding instances of riboswitches and ribozymes by homology search of structured RNA with Infernal. Therapeutic Applications of Ribozymes and Riboswitches: Methods and Protocols, Springer, p. 113–126, 2014. Citado na página 50.

LANGE, S. J.; MATICZKA, D.; MÖHL, M.; GAGNON, J. N.; BROWN, C. M.; BACKOFEN, R. Global or local? predicting secondary structure and accessibility in mRNAs. Nucleic acids research, Oxford Univ Press, p. gks181, 2012. Citado 2 vezes nas páginas 15 e 40.

LUKIW, W.; HANDLEY, P.; WONG, L.; MCLACHLAN, D. C. BC200 RNA in normal human neocortex, non-alzheimer dementia (NAD), and senile dementia of the alzheimer type (AD). Neurochemical research, Springer, v. 17, n. 6, p. 591–597, 1992. Citado na página 39.

MACHADO-LIMA, A. Laboratório de geração de classificadores de seqüências. Dissertação (Mestrado) — Universidade de São Paulo, 2002. Citado 5 vezes nas páginas 14, 30, 31, 47 e 71.

MACHADO-LIMA, A. Predição de RNAs não codificantes e sua aplicação na busca do componente RNA da telomerase. Tese (Doutorado) — Universidade de São Paulo, 2006. Citado 2 vezes nas páginas 48 e 54.

MACHADO-LIMA, A.; PORTILLO, H. A. D.; DURHAM, A. M. Computational methods in noncoding RNA research. Journal of mathematical biology, Springer, v. 56, n. 1-2, p. 15–49, 2008. Citado 2 vezes nas páginas 41 e 48.

MÄKINEN, E. On the structural grammatical interference problem for some classes of context-free grammars. Information Processing Letters, Elsevier North-Holland, Inc., v. 42, n. 1, p. 1–5, 1992. Citado na página 31.

MATSUNO, I. P. Um Estudo do Processo de Inferência de Gramáticas Regulares e Livres de Contexto Baseados em Modelos Adaptativos. Dissertação (Mestrado) — M. Sc. Dissertation, Escola Politécnica, Universidade de São Paulo, 2006. Citado na página 19.

MATTEI, E.; AUSIELLO, G.; FERRE, F.; HELMER-CITTERICH, M. A novel approach to represent and compare RNA secondary structures. Nucleic Acids Res, v. 42, n. 10, p. 6146–57, 2014. ISSN 0305-1048. Citado 2 vezes nas páginas 62 e 63.

MATTEI, E.; PIETROSANTO, M.; FERRE, F.; HELMER-CITTERICH, M. Web-Beagle: a web server for the alignment of RNA secondary structures. Nucleic Acids Res, v. 43, n. W1, p. W493–7, 2015. ISSN 0305-1048. Citado na página 66.

MATTHEWS, B. W. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. Biochimica et Biophysica Acta (BBA)-Protein Structure, Elsevier, v. 405, n. 2, p. 442–451, 1975. Citado na página 62.

MATTICK, J. S.; MAKUNIN, I. V. Non-coding RNA. Human molecular genetics, Oxford Univ Press, v. 15, n. suppl 1, p. R17–R29, 2006. Citado na página 39.

MENEZES, P. B. Linguagens Formais e Autômatos: Volume 3 da Série Livros Didáticos Informática UFRGS. [S.l.]: Bookman Editora, 2009. Citado na página 14.

MILLAR, J. K.; WILSON-ANNAN, J. C.; ANDERSON, S.; CHRISTIE, S.; TAYLOR, M. S.; SEMPLE, C. A.; DEVON, R. S.; CLAIR, D. M. S.; MUIR, W. J.; BLACKWOOD, D. H. *et al.* Disruption of two novel genes by a translocation co-segregating with schizophrenia. Human molecular genetics, Oxford Univ Press, v. 9, n. 9, p. 1415–1423, 2000. Citado na página 39.

NAWROCKI, E. P. Annotating functional RNAs in genomes using infernal. RNA Sequence, Structure, and Function: Computational and Bioinformatic Methods, Springer, p. 163–197, 2014. Citado na página 50.

NAWROCKI, E. P.; EDDY, S. R. Infernal 1.1: 100-fold faster RNA homology searches. Bioinformatics, Oxford Univ Press, v. 29, n. 22, p. 2933–2935, 2013. Citado na página 50.

NAWROCKI, E. P.; KOLBE, D. L.; EDDY, S. R. Infernal 1.0: inference of RNA alignments. Bioinformatics, Oxford Univ Press, v. 25, n. 10, p. 1335–1337, 2009. Citado na página 50.

NEEDLEMAN, S. B.; WUNSCH, C. D. A general method applicable to the search for similarities in the amino acid sequence of two proteins. Journal of molecular biology, Elsevier, v. 48, n. 3, p. 443–453, 1970. Citado na página 63.

NETO, J. J. Adaptive automata for context-dependent languages. ACM Sigplan Notices, ACM, v. 29, n. 9, p. 115–124, 1994. Citado na página 33.

- NETO, J. J. Adaptive rule-driven devices-general formulation and case study. In: SPRINGER. International conference on implementation and application of automata. [S.l.], 2001. p. 234–250. Citado 3 vezes nas páginas 32, 33 e 36.
- NOVIKOVA, I. V.; HENNELLY, S. P.; SANBONMATSU, K. Y. Sizing up long non-coding RNAs: do lncRNAs have secondary and tertiary structure? Bioarchitecture, Taylor & Francis, v. 2, n. 6, p. 189–199, 2012. Citado 2 vezes nas páginas 15 e 40.
- NOVIKOVA, I. V.; HENNELLY, S. P.; SANBONMATSU, K. Y. Structural architecture of the human long non-coding RNA, steroid receptor RNA activator. Nucleic acids research, Oxford Univ Press, v. 40, n. 11, p. 5034–5051, 2012. Citado 2 vezes nas páginas 15 e 40.
- PICCINELLI, P.; ROSENBLAD, M. A.; SAMUELSSON, T. Identification and analysis of ribonuclease P and MRP RNA in a broad range of eukaryotes. Nucleic acids research, Oxford Univ Press, v. 33, n. 14, p. 4485–4495, 2005. Citado 2 vezes nas páginas 44 e 45.
- POLESSKAYA, O. O.; HAROUTUNIAN, V.; DAVIS, K. L.; HERNANDEZ, I.; SOKOLOV, B. P. Novel putative nonprotein-coding RNA gene from 11q14 displays decreased expression in brains of patients with schizophrenia. Journal of neuroscience research, Wiley Online Library, v. 74, n. 1, p. 111–122, 2003. Citado na página 39.
- RAMOS, M. V. M.; NETO, J. J.; VEGA, Í. S. Linguagens formais: teoria, modelagem e implementação. [S.l.]: Bookman Editora, 2009. Citado 6 vezes nas páginas 14, 15, 17, 18, 25 e 32.
- REIS, E. M.; LOURO, R.; NAKAYA, H. I.; VERJOVSKI-ALMEIDA, S. As antisense RNA gets intronic. Omics: a journal of integrative biology, Mary Ann Liebert, Inc. 2 Madison Avenue Larchmont, NY 10538 USA, v. 9, n. 1, p. 2–12, 2005. Citado na página 39.
- REIS, E. M.; NAKAYA, H. I.; LOURO, R.; CANAVEZ, F. C.; FLATSCHART, Á. V.; ALMEIDA, G. T.; EGIDIO, C. M.; PAQUOLA, A. C.; MACHADO, A. A.; FESTA, F. *et al.* Antisense intronic non-coding RNA levels correlate to the degree of tumor differentiation in prostate cancer. Oncogene, Nature Publishing Group, v. 23, n. 39, p. 6684–6692, 2004. Citado na página 39.
- REUSKEN, C. B.; BOL, J. F. Structural elements of the 3'-terminal coat protein binding site in alfalfa mosaic virus RNAs. Nucleic acids research, Oxford Univ Press, v. 24, n. 14, p. 2660–2665, 1996. Citado na página 57.
- RIVAS, E.; EDDY, S. R. The language of RNA: a formal grammar that includes pseudoknots. Bioinformatics, Oxford Univ Press, v. 16, n. 4, p. 334–340, 2000. Citado 2 vezes nas páginas 14 e 47.
- SAKAKIBARA, Y. Efficient learning of context-free grammars from positive structural examples. Information and Computation, Academic Press, Inc., v. 97, n. 1, p. 23–60, 1992. Citado na página 31.
- SAKAKIBARA, Y.; BROWN, M.; HUGHEY, R.; MIAN, I. S. The application of stochastic context-free grammars to folding, aligning and modeling homologous RNA sequences. Report, UC Santa Cruz, Citeseer, 1993. Citado na página 48.

SAKAKIBARA, Y.; BROWN, M.; HUGHEY, R.; MIAN, S.; SJÖLANDER, K.; UNDERWOOD, R.; HAUSSLER, D. Recent methods for RNA modeling using stochastic context-free grammars. In: SPRINGER. Combinatorial Pattern Matching. [S.l.], 1994. p. 289–306. Citado 3 vezes nas páginas 48, 49 e 50.

SAKAKIBARA, Y.; BROWN, M.; HUGHEY, R.; MIAN, I. S.; SJÖLANDER, K.; UNDERWOOD, R. C.; HAUSSLER, D. Stochastic context-free grammars for tRNA modeling. Nucleic acids research, Oxford Univ Press, v. 22, n. 23, p. 5112–5120, 1994. Citado na página 48.

SAKAKIBARA, Y.; BROWN, M.; UNDERWOOD, R.; MIAN, I.; HAUSSLER, D. Stochastic context-free grammars for modeling RNA. In: 1994 Proceedings of the Twenty-Seventh Hawaii International Conference on System Sciences. [S.l.: s.n.], 1994. Citado 2 vezes nas páginas 31 e 48.

SEARLS, D. B. The linguistics of DNA. American Scientist, JSTOR, v. 80, n. 6, p. 579–591, 1992. Citado 3 vezes nas páginas 15, 24 e 47.

SEARLS, D. B. Linguistic approaches to biological sequences. Computer applications in the biosciences: CABIOS, Oxford Univ Press, v. 13, n. 4, p. 333–344, 1997. Citado 3 vezes nas páginas 14, 15 e 47.

SEARLS, D. B. The language of genes. Nature, Nature Publishing Group, v. 420, n. 6912, p. 211–217, 2002. Citado na página 54.

SEEMANN, S. E.; SUNKIN, S. M.; HAWRYLYCZ, M. J.; RUZZO, W. L.; GORODKIN, J. Transcripts with in silico predicted RNA structure are enriched everywhere in the mouse brain. BMC genomics, BioMed Central, v. 13, n. 1, p. 214, 2012. Citado 2 vezes nas páginas 15 e 40.

SHARMA, A. Theory of automata and formal languages. [S.l.]: Firewall Media, 2006. Citado na página 17.

SIPSER, M. Introduction to the Theory of Computation. [S.l.]: Thomson Course Technology Boston, 2006. v. 2. Citado 5 vezes nas páginas 20, 21, 22, 24 e 27.

SMITH, T. F.; WATERMAN, M. S. Identification of common molecular subsequences. Journal of molecular biology, Elsevier, v. 147, n. 1, p. 195–197, 1981. Citado na página 66.

SONG, Y.; HUA, L.; SHAPIRO, B. A.; WANG, J. T. Effective alignment of RNA pseudoknot structures using partition function posterior log-odds scores. BMC Bioinformatics, v. 16, p. 39, 2015. ISSN 1471-2105. Citado na página 60.

SORESCU, D. A.; MOHL, M.; MANN, M.; BACKOFEN, R.; WILL, S. CARNA—alignment of RNA structure ensembles. Nucleic Acids Res, v. 40, n. Web Server issue, p. W49–53, 2012. ISSN 0305-1048. Citado na página 61.

WASHIETL, S.; WILL, S.; HENDRIX, D. A.; GOFF, L. A.; RINN, J. L.; BERGER, B.; KELLIS, M. Computational analysis of noncoding RNAs. Wiley Interdisciplinary Reviews: RNA, Wiley Online Library, v. 3, n. 6, p. 759–778, 2012. Citado na página 42.

WATSON, J.; CRICK, F. Molecular structure of nucleic acids: a structure for deoxyribose nucleic acid. American Journal of Psychiatry, Am Psychiatric Assoc, v. 160, n. 4, p. 623–624, 2003. Citado na página 39.

WIDMANN, J.; STOMBAUGH, J.; MCDONALD, D.; CHOCHOLOUSOVA, J.; GARDNER, P.; IYER, M. K.; LIU, Z.; LOZUPONE, C. A.; QUINN, J.; SMIT, S. *et al.* RNASAR: an RNA structural alignment repository that provides insight into the evolution of natural and artificial RNAs. Rna, Cold Spring Harbor Lab, v. 18, n. 7, p. 1319–1327, 2012. Citado 2 vezes nas páginas 63 e 67.

WILM, A.; MAINZ, I.; STEGER, G. An enhanced RNA alignment benchmark for sequence alignment programs. Algorithms for molecular biology, BioMed Central, v. 1, n. 1, p. 19, 2006. Citado na página 62.

WONG, T. K.; LAM, T. W.; SUNG, W. K.; CHEUNG, B. W.; YIU, S. M. Structural alignment of RNA with complex pseudoknot structure. J Comput Biol, v. 18, n. 1, p. 97–108, 2011. ISSN 1066-5277. Citado 3 vezes nas páginas 57, 58 e 59.

WONG, T. K.; YIU, S. M. Structural alignment of RNA with triple helix structure. J Comput Biol, v. 19, n. 4, p. 365–78, 2012. ISSN 1066-5277. Citado na página 59.

YOUNGER, D. H. Recognition and parsing of context-free languages in time n^3 . Information and control, Elsevier, v. 10, n. 2, p. 189–208, 1967. Citado na página 27.

ZHANG, Y.; HUANG, H.; DONG, X.; FANG, Y.; WANG, K.; ZHU, L.; HUANG, T.; YANG, J. A dynamic 3D graphical representation for RNA structure analysis and its application in non-coding RNA classification. PLoS One, v. 11, n. 5, p. e0152238, 2016. ISSN 1932-6203. Citado na página 56.

ZHONG, C.; ZHANG, S. Efficient alignment of RNA secondary structures using sparse dynamic programming. BMC Bioinformatics, v. 14, p. 269, 2013. ISSN 1471-2105. Citado na página 66.

Apêndices

Apêndice A – EXECUÇÃO - EARLEY ADAPTATIVO

Gramática adaptativa na notação BNF

5

```

<S> ::= <K> \aa{AdP(<K>)}

\BeginFunction AdP(X)
  \DefineGenerator <A'>
  \DefineGenerator <B'>
  \DefineGenerator <C'>
  \DefineGenerator <D'>

  \BeginInsertActions
    X ::= <A'><B'><C'><D'>
    <A'> ::= a<A'> \aa{Ad1(<A'>,<C'>,a,c)}
    <A'> ::= a \aa{Ad2(<C'>,c)}
    <B'> ::= b<B'> \aa{Ad1(<B'>,<D'>,b,d)}
    <B'> ::= b \aa{Ad2(<D'>,d)}
  \EndInsertActions
\EndFunction

\BeginFunction Ad1(X,Y,x,y)
  \DefineGenerator <Y'>

  \BeginDeleteActions
    X ::= xX \aa{Ad1(X,Y,x,y)}
    X ::= x \aa{Ad2(Y,y)}
  \EndDeleteActions

  \BeginInsertActions
    X ::= xX \aa{Ad1(X,<Y'>,x,y)}
    X ::= x \aa{Ad2(<Y'>,y)}
    Y ::= y<Y'>
  \EndInsertActions
\EndFunction

\BeginFunction Ad2(Y,y)
  \BeginInsertActions
    Y ::= y
  \EndInsertActions
\EndFunction

```

Resultado da execução do algoritmo

1: aabbbccddd => ACEITA

Parser States:

State: 0 input: a

$\langle 0 \rangle ::= .\langle S \rangle \# \mid f: 0 \mid g: 0$
 $\langle S \rangle ::= .\langle K \rangle \{ \text{AdP}(\langle K \rangle) \} \mid f: 0 \mid g: 0 \rightarrow 1$
 $\langle K \rangle ::= .\langle A'1 \rangle \langle B'2 \rangle \langle C'3 \rangle \langle D'4 \rangle \mid f: 0 \mid g: 1$
 $\langle A'1 \rangle ::= .a \langle A'1 \rangle \{ \text{Ad1}(\langle A'1 \rangle, \langle C'3 \rangle, a, c) \} \mid f: 0 \mid g: 1 \rightarrow 2$
 $\langle A'1 \rangle ::= .a \{ \text{Ad2}(\langle C'3 \rangle, c) \} \mid f: 0 \mid g: 1 \rightarrow 2$

State: 1 input: a

$\langle A'1 \rangle ::= a.\langle A'1 \rangle \{ \text{Ad1}(\langle A'1 \rangle, \langle C'3 \rangle, a, c) \} \mid f: 0 \mid g: 1 \rightarrow 2$
 $\langle A'1 \rangle ::= a. \{ \text{Ad2}(\langle C'3 \rangle, c) \} \mid f: 0 \mid g: 1 \rightarrow 2$
 $\langle A'1 \rangle ::= .a \langle A'1 \rangle \{ \text{Ad1}(\langle A'1 \rangle, \langle X'5 \rangle, a, c) \} \mid f: 1 \mid g: 2 \rightarrow 3$
 $\langle A'1 \rangle ::= .a \{ \text{Ad2}(\langle X'5 \rangle, c) \} \mid f: 1 \mid g: 2 \rightarrow 3$
 $\langle K \rangle ::= \langle A'1 \rangle . \langle B'2 \rangle \langle C'3 \rangle \langle D'4 \rangle \mid f: 0 \mid g: 1 \rightarrow 2$
 $\langle B'2 \rangle ::= .b \langle B'2 \rangle \{ \text{Ad1}(\langle B'2 \rangle, \langle D'4 \rangle, b, d) \} \mid f: 1 \mid g: 2 \rightarrow 3$
 $\langle B'2 \rangle ::= .b \{ \text{Ad2}(\langle D'4 \rangle, d) \} \mid f: 1 \mid g: 2 \rightarrow 3$

State: 2 input: b

$\langle A'1 \rangle ::= a.\langle A'1 \rangle \{ \text{Ad1}(\langle A'1 \rangle, \langle X'5 \rangle, a, c) \} \mid f: 1 \mid g: 2 \rightarrow 3$
 $\langle A'1 \rangle ::= a. \{ \text{Ad2}(\langle X'5 \rangle, c) \} \mid f: 1 \mid g: 2 \rightarrow 3$
 $\langle A'1 \rangle ::= .a \langle A'1 \rangle \{ \text{Ad1}(\langle A'1 \rangle, \langle X'6 \rangle, a, c) \} \mid f: 2 \mid g: 3 \rightarrow 4$
 $\langle A'1 \rangle ::= .a \{ \text{Ad2}(\langle X'6 \rangle, c) \} \mid f: 2 \mid g: 3 \rightarrow 4$
 $\langle A'1 \rangle ::= a \langle A'1 \rangle . \{ \text{Ad1}(\langle A'1 \rangle, \langle C'3 \rangle, a, c) \} \mid f: 0 \mid g: 1 \rightarrow 3$
 $\langle K \rangle ::= \langle A'1 \rangle . \langle B'2 \rangle \langle C'3 \rangle \langle D'4 \rangle \mid f: 0 \mid g: 1 \rightarrow 3$
 $\langle B'2 \rangle ::= .b \langle B'2 \rangle \{ \text{Ad1}(\langle B'2 \rangle, \langle D'4 \rangle, b, d) \} \mid f: 2 \mid g: 3 \rightarrow 4$
 $\langle B'2 \rangle ::= .b \{ \text{Ad2}(\langle D'4 \rangle, d) \} \mid f: 2 \mid g: 3 \rightarrow 4$

State: 3 input: b

$\langle B'2 \rangle ::= b.\langle B'2 \rangle \{Ad1(\langle B'2 \rangle, \langle D'4 \rangle, b, d)\} \mid f: 2 \mid g: 3 \rightarrow 4$
 $\langle B'2 \rangle ::= b. \{Ad2(\langle D'4 \rangle, d)\} \mid f: 2 \mid g: 3 \rightarrow 4$
 $\langle B'2 \rangle ::= .b\langle B'2 \rangle \{Ad1(\langle B'2 \rangle, \langle X'9 \rangle, b, d)\} \mid f: 3 \mid g: 4 \rightarrow 5$
 $\langle B'2 \rangle ::= .b \{Ad2(\langle X'9 \rangle, d)\} \mid f: 3 \mid g: 4 \rightarrow 5$
 $\langle K \rangle ::= \langle A'1 \rangle \langle B'2 \rangle . \langle C'3 \rangle \langle D'4 \rangle \mid f: 0 \mid g: 1 \rightarrow 4$
 $\langle C'3 \rangle ::= .c\langle X'5 \rangle \mid f: 3 \mid g: 4$

State: 4 input: b

$\langle B'2 \rangle ::= b.\langle B'2 \rangle \{Ad1(\langle B'2 \rangle, \langle X'9 \rangle, b, d)\} \mid f: 3 \mid g: 4 \rightarrow 5$
 $\langle B'2 \rangle ::= b. \{Ad2(\langle X'9 \rangle, d)\} \mid f: 3 \mid g: 4 \rightarrow 5$
 $\langle B'2 \rangle ::= .b\langle B'2 \rangle \{Ad1(\langle B'2 \rangle, \langle X'10 \rangle, b, d)\} \mid f: 4 \mid g: 5 \rightarrow 6$
 $\langle B'2 \rangle ::= .b \{Ad2(\langle X'10 \rangle, d)\} \mid f: 4 \mid g: 5 \rightarrow 6$
 $\langle B'2 \rangle ::= b\langle B'2 \rangle . \{Ad1(\langle B'2 \rangle, \langle D'4 \rangle, b, d)\} \mid f: 2 \mid g: 3 \rightarrow 5$
 $\langle K \rangle ::= \langle A'1 \rangle \langle B'2 \rangle . \langle C'3 \rangle \langle D'4 \rangle \mid f: 0 \mid g: 1 \rightarrow 5$
 $\langle C'3 \rangle ::= .c\langle X'5 \rangle \mid f: 4 \mid g: 5$

State: 5 input: c

$\langle B'2 \rangle ::= b.\langle B'2 \rangle \{Ad1(\langle B'2 \rangle, \langle X'10 \rangle, b, d)\} \mid f: 4 \mid g: 5 \rightarrow 6$
 $\langle B'2 \rangle ::= b. \{Ad2(\langle X'10 \rangle, d)\} \mid f: 4 \mid g: 5 \rightarrow 6$
 $\langle B'2 \rangle ::= .b\langle B'2 \rangle \{Ad1(\langle B'2 \rangle, \langle X'11 \rangle, b, d)\} \mid f: 5 \mid g: 6 \rightarrow 7$
 $\langle B'2 \rangle ::= .b \{Ad2(\langle X'11 \rangle, d)\} \mid f: 5 \mid g: 6 \rightarrow 7$
 $\langle B'2 \rangle ::= b\langle B'2 \rangle . \{Ad1(\langle B'2 \rangle, \langle X'9 \rangle, b, d)\} \mid f: 3 \mid g: 4 \rightarrow 6$
 $\langle B'2 \rangle ::= b\langle B'2 \rangle . \{Ad1(\langle B'2 \rangle, \langle D'4 \rangle, b, d)\} \mid f: 2 \mid g: 3 \rightarrow 6$
 $\langle K \rangle ::= \langle A'1 \rangle \langle B'2 \rangle . \langle C'3 \rangle \langle D'4 \rangle \mid f: 0 \mid g: 1 \rightarrow 6$
 $\langle C'3 \rangle ::= .c\langle X'5 \rangle \mid f: 5 \mid g: 6$

State: 6 input: c

$\langle C'3 \rangle ::= c.\langle X'5 \rangle \mid f: 5 \mid g: 6$
 $\langle X'5 \rangle ::= .c \mid f: 6 \mid g: 6$

State: 7 input: d

$\langle X'5 \rangle ::= c. \mid f: 6 \mid g: 6$
 $\langle C'3 \rangle ::= c\langle X'5 \rangle . \mid f: 5 \mid g: 6$

$\langle K \rangle ::= \langle A'1 \rangle \langle B'2 \rangle \langle C'3 \rangle . \langle D'4 \rangle \mid f: 0 \mid g: 1 \rightarrow 6$
 $\langle D'4 \rangle ::= .d \langle X'9 \rangle \mid f: 7 \mid g: 6$

State: 8 input: d

$\langle D'4 \rangle ::= d . \langle X'9 \rangle \mid f: 7 \mid g: 6$
 $\langle X'9 \rangle ::= .d \langle X'10 \rangle \mid f: 8 \mid g: 6$

State: 9 input: d

$\langle X'9 \rangle ::= d . \langle X'10 \rangle \mid f: 8 \mid g: 6$
 $\langle X'10 \rangle ::= .d \mid f: 9 \mid g: 6$

State: 10 input: #

$\langle X'10 \rangle ::= d . \mid f: 9 \mid g: 6$
 $\langle X'9 \rangle ::= d \langle X'10 \rangle . \mid f: 8 \mid g: 6$
 $\langle D'4 \rangle ::= d \langle X'9 \rangle . \mid f: 7 \mid g: 6$
 $\langle K \rangle ::= \langle A'1 \rangle \langle B'2 \rangle \langle C'3 \rangle \langle D'4 \rangle . \mid f: 0 \mid g: 1 \rightarrow 6$
 $\langle S \rangle ::= \langle K \rangle . \{ \text{AdP}(\langle K \rangle) \} \mid f: 0 \mid g: 0 \rightarrow 6$
 $\langle @ \rangle ::= \langle S \rangle . \# \mid f: 0 \mid g: 0 \rightarrow 6$

State: 11 input:

$\langle @ \rangle ::= \langle S \rangle \# . \mid f: 0 \mid g: 0 \rightarrow 6$

Árvore sintática

```

<@>
  <S>
    <K>
      <A'1>
        a
        <A'1>
          a
      <B'2>
        b
        <B'2>
          b
          <B'2>
            b
      <C'3>
        c
        <X'5>
          c
      <D'4>
        d
        <X'9>
          d
          <X'10>
            d
    #

```

Apêndice B – GRAMÁTICA - FAMÍLIA RF00505

5	<8> ::= A<9>
<S> ::= <1>	<8> ::= U<9>
<1> ::= A<2>	<8> ::= G<9>
<1> ::= U<2>	<8> ::= C<9>
<1> ::= G<2>	<9> ::= A<10>
<1> ::= C<2>	<9> ::= U<10>
<2> ::= A<3>	<9> ::= G<10>
<2> ::= U<3>	<9> ::= C<10>
<2> ::= G<3>	<10> ::= A<11>
<2> ::= C<3>	<10> ::= U<11>
<3> ::= A<4>	<10> ::= G<11>
<3> ::= U<4>	<10> ::= C<11>
<3> ::= G<4>	<11> ::= A<12>
<3> ::= C<4>	<11> ::= U<12>
<4> ::= A<5>	<11> ::= G<12>
<4> ::= U<5>	<11> ::= C<12>
<4> ::= G<5>	<12> ::= <13><37>
<4> ::= C<5>	<13> ::= A<14>U
<5> ::= A<6>	<13> ::= U<14>A
<5> ::= U<6>	<13> ::= G<14>C
<5> ::= G<6>	<13> ::= C<14>G
<5> ::= C<6>	<13> ::= G<14>U
<6> ::= A<7>	<13> ::= U<14>G
<6> ::= U<7>	<14> ::= A<15>U
<6> ::= G<7>	<14> ::= U<15>A
<6> ::= C<7>	<14> ::= G<15>C
<7> ::= A<8>	<14> ::= C<15>G
<7> ::= U<8>	<14> ::= G<15>U
<7> ::= G<8>	<14> ::= U<15>G
<7> ::= C<8>	<15> ::= A<16>U

$\langle 15 \rangle ::= U\langle 16 \rangle A$	$\langle 20 \rangle ::= \langle 21 \rangle$
$\langle 15 \rangle ::= G\langle 16 \rangle C$	$\langle 21 \rangle ::= A\langle 22 \rangle$
$\langle 15 \rangle ::= C\langle 16 \rangle G$	$\langle 21 \rangle ::= U\langle 22 \rangle$
$\langle 15 \rangle ::= G\langle 16 \rangle U$	$\langle 21 \rangle ::= G\langle 22 \rangle$
$\langle 15 \rangle ::= U\langle 16 \rangle G$	$\langle 21 \rangle ::= C\langle 22 \rangle$
$\langle 16 \rangle ::= A\langle 17 \rangle U$	$\langle 22 \rangle ::= A\langle 23 \rangle$
$\langle 16 \rangle ::= U\langle 17 \rangle A$	$\langle 22 \rangle ::= U\langle 23 \rangle$
$\langle 16 \rangle ::= G\langle 17 \rangle C$	$\langle 22 \rangle ::= G\langle 23 \rangle$
$\langle 16 \rangle ::= C\langle 17 \rangle G$	$\langle 22 \rangle ::= C\langle 23 \rangle$
$\langle 16 \rangle ::= G\langle 17 \rangle U$	$\langle 23 \rangle ::= A\langle 24 \rangle$
$\langle 16 \rangle ::= U\langle 17 \rangle G$	$\langle 23 \rangle ::= U\langle 24 \rangle$
$\langle 17 \rangle ::= A\langle 18 \rangle U$	$\langle 23 \rangle ::= G\langle 24 \rangle$
$\langle 17 \rangle ::= U\langle 18 \rangle A$	$\langle 23 \rangle ::= C\langle 24 \rangle$
$\langle 17 \rangle ::= G\langle 18 \rangle C$	$\langle 24 \rangle ::= A\langle 25 \rangle$
$\langle 17 \rangle ::= C\langle 18 \rangle G$	$\langle 24 \rangle ::= U\langle 25 \rangle$
$\langle 17 \rangle ::= G\langle 18 \rangle U$	$\langle 24 \rangle ::= G\langle 25 \rangle$
$\langle 17 \rangle ::= U\langle 18 \rangle G$	$\langle 24 \rangle ::= C\langle 25 \rangle$
$\langle 18 \rangle ::= A\langle 19 \rangle U$	$\langle 25 \rangle ::= A\langle 26 \rangle$
$\langle 18 \rangle ::= U\langle 19 \rangle A$	$\langle 25 \rangle ::= U\langle 26 \rangle$
$\langle 18 \rangle ::= G\langle 19 \rangle C$	$\langle 25 \rangle ::= G\langle 26 \rangle$
$\langle 18 \rangle ::= C\langle 19 \rangle G$	$\langle 25 \rangle ::= C\langle 26 \rangle$
$\langle 18 \rangle ::= G\langle 19 \rangle U$	$\langle 26 \rangle ::= A\langle 27 \rangle$
$\langle 18 \rangle ::= U\langle 19 \rangle G$	$\langle 26 \rangle ::= U\langle 27 \rangle$
$\langle 19 \rangle ::= A\langle 20 \rangle U$	$\langle 26 \rangle ::= G\langle 27 \rangle$
$\langle 19 \rangle ::= U\langle 20 \rangle A$	$\langle 26 \rangle ::= C\langle 27 \rangle$
$\langle 19 \rangle ::= G\langle 20 \rangle C$	$\langle 27 \rangle ::= A\langle 28 \rangle$
$\langle 19 \rangle ::= C\langle 20 \rangle G$	$\langle 27 \rangle ::= U\langle 28 \rangle$
$\langle 19 \rangle ::= G\langle 20 \rangle U$	$\langle 27 \rangle ::= G\langle 28 \rangle$
$\langle 19 \rangle ::= U\langle 20 \rangle G$	$\langle 27 \rangle ::= C\langle 28 \rangle$
$\langle 20 \rangle ::= A\langle 21 \rangle$	$\langle 28 \rangle ::= A\langle 29 \rangle \setminus \text{aa}\{\text{Ad1}(\langle 28' \rangle, U)\}$
$\langle 20 \rangle ::= U\langle 21 \rangle$	$\langle 28 \rangle ::= U\langle 29 \rangle \setminus \text{aa}\{\text{Ad1}(\langle 28' \rangle, A)\}$
$\langle 20 \rangle ::= G\langle 21 \rangle$	$\langle 28 \rangle ::= G\langle 29 \rangle \setminus \text{aa}\{\text{Ad1}(\langle 28' \rangle, C)\}$
$\langle 20 \rangle ::= C\langle 21 \rangle$	$\langle 28 \rangle ::= C\langle 29 \rangle \setminus \text{aa}\{\text{Ad1}(\langle 28' \rangle, G)\}$

<28> ::= G<29> \aa{Ad1(<28'>,U)}	<34> ::= U<35> \aa{Ad1(<34'>,A)}
<28> ::= U<29> \aa{Ad1(<28'>,G)}	<34> ::= G<35> \aa{Ad1(<34'>,C)}
<29> ::= A<30> \aa{Ad1(<29'>,U)}	<34> ::= C<35> \aa{Ad1(<34'>,G)}
<29> ::= U<30> \aa{Ad1(<29'>,A)}	<34> ::= G<35> \aa{Ad1(<34'>,U)}
<29> ::= G<30> \aa{Ad1(<29'>,C)}	<34> ::= U<35> \aa{Ad1(<34'>,G)}
<29> ::= C<30> \aa{Ad1(<29'>,G)}	<35> ::= A<36> \aa{Ad1(<35'>,U)}
<29> ::= G<30> \aa{Ad1(<29'>,U)}	<35> ::= U<36> \aa{Ad1(<35'>,A)}
<29> ::= U<30> \aa{Ad1(<29'>,G)}	<35> ::= G<36> \aa{Ad1(<35'>,C)}
<30> ::= A<31> \aa{Ad1(<30'>,U)}	<35> ::= C<36> \aa{Ad1(<35'>,G)}
<30> ::= U<31> \aa{Ad1(<30'>,A)}	<35> ::= G<36> \aa{Ad1(<35'>,U)}
<30> ::= G<31> \aa{Ad1(<30'>,C)}	<35> ::= U<36> \aa{Ad1(<35'>,G)}
<30> ::= C<31> \aa{Ad1(<30'>,G)}	<36> ::= A
<30> ::= G<31> \aa{Ad1(<30'>,U)}	<36> ::= U
<30> ::= U<31> \aa{Ad1(<30'>,G)}	<36> ::= G
<31> ::= A<32> \aa{Ad1(<31'>,U)}	<36> ::= C
<31> ::= U<32> \aa{Ad1(<31'>,A)}	<37> ::= A<38>
<31> ::= G<32> \aa{Ad1(<31'>,C)}	<37> ::= U<38>
<31> ::= C<32> \aa{Ad1(<31'>,G)}	<37> ::= G<38>
<31> ::= G<32> \aa{Ad1(<31'>,U)}	<37> ::= C<38>
<31> ::= U<32> \aa{Ad1(<31'>,G)}	<37> ::= <38>
<32> ::= A<33> \aa{Ad1(<32'>,U)}	<38> ::= A<39>
<32> ::= U<33> \aa{Ad1(<32'>,A)}	<38> ::= U<39>
<32> ::= G<33> \aa{Ad1(<32'>,C)}	<38> ::= G<39>
<32> ::= C<33> \aa{Ad1(<32'>,G)}	<38> ::= C<39>
<32> ::= G<33> \aa{Ad1(<32'>,U)}	<39> ::= A<40>
<32> ::= U<33> \aa{Ad1(<32'>,G)}	<39> ::= U<40>
<33> ::= A<34> \aa{Ad1(<33'>,U)}	<39> ::= G<40>
<33> ::= U<34> \aa{Ad1(<33'>,A)}	<39> ::= C<40>
<33> ::= G<34> \aa{Ad1(<33'>,C)}	<40> ::= A<41>
<33> ::= C<34> \aa{Ad1(<33'>,G)}	<40> ::= U<41>
<33> ::= G<34> \aa{Ad1(<33'>,U)}	<40> ::= G<41>
<33> ::= U<34> \aa{Ad1(<33'>,G)}	<40> ::= C<41>
<34> ::= A<35> \aa{Ad1(<34'>,U)}	<41> ::= A<42>

<41> ::= U<42>	<55> ::= A<56>
<41> ::= G<42>	<55> ::= U<56>
<41> ::= C<42>	<55> ::= G<56>
<42> ::= A<43>	<55> ::= C<56>
<42> ::= U<43>	<56> ::= A<57>
<42> ::= G<43>	<56> ::= U<57>
<42> ::= C<43>	<56> ::= G<57>
<43> ::= A<44>	<56> ::= C<57>
<43> ::= U<44>	<57> ::= A<58>
<43> ::= G<44>	<57> ::= U<58>
<43> ::= C<44>	<57> ::= G<58>
<44> ::= A<45>	<57> ::= C<58>
<44> ::= U<45>	<58> ::= A<59>
<44> ::= G<45>	<58> ::= U<59>
<44> ::= C<45>	<58> ::= G<59>
<45> ::= <35'><46>	<58> ::= C<59>
<46> ::= <34'><47>	<59> ::= A<60>
<47> ::= <33'><48>	<59> ::= U<60>
<48> ::= <32'><49>	<59> ::= G<60>
<49> ::= <31'><50>	<59> ::= C<60>
<50> ::= <30'><51>	<60> ::= A
<51> ::= <29'><52>	<60> ::= U
<52> ::= <28'><53>	<60> ::= G
<53> ::= A<54>	<60> ::= C
<53> ::= U<54>	
<53> ::= G<54>	\BeginFunction Ad1(X,y)
<53> ::= C<54>	\BeginInsertActions
<54> ::= A<55>	X ::= y
<54> ::= U<55>	\EndInsertActions
<54> ::= G<55>	\EndFunction
<54> ::= C<55>	

Apêndice C – GRAMÁTICA - FAMÍLIA RF00024

5	<10> ::= G<11>
<S> ::= <1>	<10> ::= <11>
<1> ::= G<2>	<11> ::= A<12>
<1> ::= <2>	<11> ::= U<12>
<2> ::= A<3>	<11> ::= G<12>
<2> ::= U<3>	<11> ::= C<12>
<2> ::= <3>	<11> ::= <12>
<3> ::= A<4>	<12> ::= U<13>
<3> ::= G<4>	<12> ::= G<13>
<3> ::= <4>	<12> ::= C<13>
<4> ::= U<5>	<12> ::= <13>
<4> ::= C<5>	<13> ::= A<14>
<4> ::= <5>	<13> ::= U<14>
<5> ::= G<6>	<13> ::= G<14>
<5> ::= C<6>	<13> ::= <14>
<5> ::= <6>	<14> ::= A<15>
<6> ::= A<7>	<14> ::= G<15>
<6> ::= G<7>	<14> ::= C<15>
<6> ::= <7>	<14> ::= <15>
<7> ::= A<8>	<15> ::= A<16>
<7> ::= G<8>	<15> ::= U<16>
<7> ::= <8>	<15> ::= G<16>
<8> ::= U<9>	<15> ::= C<16>
<8> ::= C<9>	<15> ::= <16>
<8> ::= <9>	<16> ::= A<17>
<9> ::= A<10>	<16> ::= G<17>
<9> ::= U<10>	<16> ::= <17>
<9> ::= G<10>	<17> ::= U<18>
<9> ::= <10>	<17> ::= G<18>
<10> ::= U<11>	<17> ::= C<18>

<17> ::= <18>	<25> ::= C<26>
<18> ::= A<19>	<25> ::= <26>
<18> ::= U<19>	<26> ::= U<27>
<18> ::= G<19>	<26> ::= G<27>
<18> ::= C<19>	<26> ::= C<27>
<18> ::= <19>	<26> ::= <27>
<19> ::= A<20>	<27> ::= A<28>
<19> ::= U<20>	<27> ::= U<28>
<19> ::= G<20>	<27> ::= G<28>
<19> ::= C<20>	<27> ::= C<28>
<19> ::= <20>	<27> ::= <28>
<20> ::= A<21>	<28> ::= A<29>
<20> ::= U<21>	<28> ::= U<29>
<20> ::= G<21>	<28> ::= G<29>
<20> ::= C<21>	<28> ::= C<29>
<20> ::= <21>	<28> ::= <29>
<21> ::= A<22>	<29> ::= A<30>
<21> ::= G<22>	<29> ::= U<30>
<21> ::= C<22>	<29> ::= G<30>
<21> ::= <22>	<29> ::= C<30>
<22> ::= G<23>	<29> ::= <30>
<22> ::= C<23>	<30> ::= A<31>
<22> ::= <23>	<30> ::= U<31>
<23> ::= A<24>	<30> ::= G<31>
<23> ::= G<24>	<30> ::= C<31>
<23> ::= C<24>	<30> ::= <31>
<23> ::= <24>	<31> ::= A<32>
<24> ::= A<25>	<31> ::= U<32>
<24> ::= U<25>	<31> ::= G<32>
<24> ::= G<25>	<31> ::= C<32>
<24> ::= <25>	<31> ::= <32>
<25> ::= U<26>	<32> ::= A<33>
<25> ::= G<26>	<32> ::= U<33>

<32> ::= G<33>	<39> ::= A<40>
<32> ::= C<33>	<39> ::= U<40>
<32> ::= <33>	<39> ::= G<40>
<33> ::= A<34>	<39> ::= C<40>
<33> ::= U<34>	<39> ::= <40>
<33> ::= G<34>	<40> ::= A<41>
<33> ::= C<34>	<40> ::= U<41>
<33> ::= <34>	<40> ::= G<41>
<34> ::= A<35>	<40> ::= C<41>
<34> ::= U<35>	<40> ::= <41>
<34> ::= G<35>	<41> ::= A<42>
<34> ::= C<35>	<41> ::= U<42>
<34> ::= <35>	<41> ::= G<42>
<35> ::= A<36>	<41> ::= C<42>
<35> ::= U<36>	<41> ::= <42>
<35> ::= G<36>	<42> ::= A<43>
<35> ::= C<36>	<42> ::= U<43>
<35> ::= <36>	<42> ::= G<43>
<36> ::= A<37>	<42> ::= C<43>
<36> ::= U<37>	<42> ::= <43>
<36> ::= G<37>	<43> ::= A<44>
<36> ::= C<37>	<43> ::= U<44>
<36> ::= <37>	<43> ::= G<44>
<37> ::= A<38>	<43> ::= C<44>
<37> ::= U<38>	<43> ::= <44>
<37> ::= G<38>	<44> ::= A<45>
<37> ::= C<38>	<44> ::= U<45>
<37> ::= <38>	<44> ::= G<45>
<38> ::= A<39>	<44> ::= C<45>
<38> ::= U<39>	<44> ::= <45>
<38> ::= G<39>	<45> ::= A<46>
<38> ::= C<39>	<45> ::= U<46>
<38> ::= <39>	<45> ::= G<46>

<45> ::= C<46>	<52> ::= U<53>
<45> ::= <46>	<52> ::= G<53>
<46> ::= A<47>	<52> ::= C<53>
<46> ::= U<47>	<52> ::= <53>
<46> ::= G<47>	<53> ::= A<54>
<46> ::= C<47>	<53> ::= U<54>
<46> ::= <47>	<53> ::= G<54>
<47> ::= A<48>	<53> ::= C<54>
<47> ::= U<48>	<53> ::= <54>
<47> ::= G<48>	<54> ::= A<55>
<47> ::= C<48>	<54> ::= U<55>
<47> ::= <48>	<54> ::= G<55>
<48> ::= A<49>	<54> ::= C<55>
<48> ::= U<49>	<54> ::= <55>
<48> ::= G<49>	<55> ::= A<56>
<48> ::= C<49>	<55> ::= U<56>
<48> ::= <49>	<55> ::= G<56>
<49> ::= A<50>	<55> ::= C<56>
<49> ::= U<50>	<55> ::= <56>
<49> ::= G<50>	<56> ::= A<57>
<49> ::= C<50>	<56> ::= U<57>
<49> ::= <50>	<56> ::= G<57>
<50> ::= A<51>	<56> ::= C<57>
<50> ::= U<51>	<56> ::= <57>
<50> ::= G<51>	<57> ::= A<58>
<50> ::= C<51>	<57> ::= U<58>
<50> ::= <51>	<57> ::= G<58>
<51> ::= A<52>	<57> ::= C<58>
<51> ::= U<52>	<57> ::= <58>
<51> ::= G<52>	<58> ::= A<59>
<51> ::= C<52>	<58> ::= U<59>
<51> ::= <52>	<58> ::= G<59>
<52> ::= A<53>	<58> ::= C<59>

<58> ::= <59>	<71> ::= <72>
<59> ::= A<60>	<72> ::= A<73>
<59> ::= U<60>	<72> ::= U<73>
<59> ::= G<60>	<72> ::= G<73>
<59> ::= C<60>	<72> ::= C<73>
<59> ::= <60>	<72> ::= <73>
<60> ::= A<61>	<73> ::= A<74>
<60> ::= U<61>	<73> ::= U<74>
<60> ::= G<61>	<73> ::= G<74>
<60> ::= C<61>	<73> ::= C<74>
<60> ::= <61>	<73> ::= <74>
<61> ::= C<62>	<74> ::= A<75>
<62> ::= U<63>	<74> ::= U<75>
<63> ::= A<64>	<74> ::= G<75>
<64> ::= A<65>	<74> ::= C<75>
<65> ::= C<66>	<74> ::= <75>
<66> ::= C<67>	<75> ::= A<76>
<67> ::= C<68>	<75> ::= U<76>
<68> ::= U<69>	<75> ::= G<76>
<69> ::= A<70>	<75> ::= C<76>
<69> ::= U<70>	<75> ::= <76>
<69> ::= G<70>	<76> ::= A<77>
<69> ::= C<70>	<76> ::= U<77>
<69> ::= <70>	<76> ::= G<77>
<70> ::= A<71>	<76> ::= C<77>
<70> ::= U<71>	<76> ::= <77>
<70> ::= G<71>	<77> ::= A<78>
<70> ::= C<71>	<77> ::= U<78>
<70> ::= <71>	<77> ::= G<78>
<71> ::= A<72>	<77> ::= C<78>
<71> ::= U<72>	<77> ::= <78>
<71> ::= G<72>	<78> ::= A<79>
<71> ::= C<72>	<78> ::= U<79>

<78> ::= G<79>	<85> ::= A<86>
<78> ::= C<79>	<85> ::= U<86>
<78> ::= <79>	<85> ::= G<86>
<79> ::= A<80>	<85> ::= C<86>
<79> ::= U<80>	<85> ::= <86>
<79> ::= G<80>	<86> ::= A<87>
<79> ::= C<80>	<86> ::= U<87>
<79> ::= <80>	<86> ::= G<87>
<80> ::= A<81>	<86> ::= C<87>
<80> ::= U<81>	<86> ::= <87>
<80> ::= G<81>	<87> ::= A<88>
<80> ::= C<81>	<87> ::= U<88>
<80> ::= <81>	<87> ::= G<88>
<81> ::= A<82>	<87> ::= C<88>
<81> ::= U<82>	<87> ::= <88>
<81> ::= G<82>	<88> ::= A<89>
<81> ::= C<82>	<88> ::= U<89>
<81> ::= <82>	<88> ::= G<89>
<82> ::= A<83>	<88> ::= C<89>
<82> ::= U<83>	<88> ::= <89>
<82> ::= G<83>	<89> ::= A<90>
<82> ::= C<83>	<89> ::= U<90>
<82> ::= <83>	<89> ::= G<90>
<83> ::= A<84>	<89> ::= C<90>
<83> ::= U<84>	<89> ::= <90>
<83> ::= G<84>	<90> ::= A<91>
<83> ::= C<84>	<90> ::= U<91>
<83> ::= <84>	<90> ::= G<91>
<84> ::= A<85>	<90> ::= C<91>
<84> ::= U<85>	<90> ::= <91>
<84> ::= G<85>	<91> ::= A<92>
<84> ::= C<85>	<91> ::= U<92>
<84> ::= <85>	<91> ::= G<92>

<91> ::= C<92>	<98> ::= U<99>
<91> ::= <92>	<98> ::= G<99>
<92> ::= A<93>	<98> ::= C<99>
<92> ::= U<93>	<98> ::= <99>
<92> ::= G<93>	<99> ::= A<100>
<92> ::= C<93>	<99> ::= U<100>
<92> ::= <93>	<99> ::= G<100>
<93> ::= A<94>	<99> ::= C<100>
<93> ::= U<94>	<99> ::= <100>
<93> ::= G<94>	<100> ::= A<101>
<93> ::= C<94>	<100> ::= U<101>
<93> ::= <94>	<100> ::= G<101>
<94> ::= A<95>	<100> ::= C<101>
<94> ::= U<95>	<100> ::= <101>
<94> ::= G<95>	<101> ::= A<102>
<94> ::= C<95>	<101> ::= U<102>
<94> ::= <95>	<101> ::= G<102>
<95> ::= A<96>	<101> ::= C<102>
<95> ::= U<96>	<101> ::= <102>
<95> ::= G<96>	<102> ::= A<103>
<95> ::= C<96>	<102> ::= U<103>
<95> ::= <96>	<102> ::= G<103>
<96> ::= A<97>	<102> ::= C<103>
<96> ::= U<97>	<102> ::= <103>
<96> ::= G<97>	<103> ::= A<104>
<96> ::= C<97>	<103> ::= U<104>
<96> ::= <97>	<103> ::= G<104>
<97> ::= A<98>	<103> ::= C<104>
<97> ::= U<98>	<103> ::= <104>
<97> ::= G<98>	<104> ::= A<105>
<97> ::= C<98>	<104> ::= U<105>
<97> ::= <98>	<104> ::= G<105>
<98> ::= A<99>	<104> ::= C<105>

$\langle 104 \rangle ::= \langle 105 \rangle$	$\langle 111 \rangle ::= A\langle 112 \rangle U$
$\langle 105 \rangle ::= A\langle 106 \rangle$	$\langle 111 \rangle ::= U\langle 112 \rangle A$
$\langle 105 \rangle ::= U\langle 106 \rangle$	$\langle 111 \rangle ::= G\langle 112 \rangle C$
$\langle 105 \rangle ::= G\langle 106 \rangle$	$\langle 111 \rangle ::= C\langle 112 \rangle G$
$\langle 105 \rangle ::= C\langle 106 \rangle$	$\langle 111 \rangle ::= G\langle 112 \rangle U$
$\langle 105 \rangle ::= \langle 106 \rangle$	$\langle 111 \rangle ::= U\langle 112 \rangle G$
$\langle 106 \rangle ::= A\langle 107 \rangle$	$\langle 112 \rangle ::= A\langle 113 \rangle U$
$\langle 106 \rangle ::= U\langle 107 \rangle$	$\langle 112 \rangle ::= U\langle 113 \rangle A$
$\langle 106 \rangle ::= G\langle 107 \rangle$	$\langle 112 \rangle ::= G\langle 113 \rangle C$
$\langle 106 \rangle ::= C\langle 107 \rangle$	$\langle 112 \rangle ::= C\langle 113 \rangle G$
$\langle 106 \rangle ::= \langle 107 \rangle$	$\langle 112 \rangle ::= G\langle 113 \rangle U$
$\langle 107 \rangle ::= A\langle 108 \rangle$	$\langle 112 \rangle ::= U\langle 113 \rangle G$
$\langle 107 \rangle ::= U\langle 108 \rangle$	$\langle 113 \rangle ::= A\langle 114 \rangle U$
$\langle 107 \rangle ::= G\langle 108 \rangle$	$\langle 113 \rangle ::= U\langle 114 \rangle A$
$\langle 107 \rangle ::= C\langle 108 \rangle$	$\langle 113 \rangle ::= G\langle 114 \rangle C$
$\langle 107 \rangle ::= \langle 108 \rangle$	$\langle 113 \rangle ::= C\langle 114 \rangle G$
$\langle 108 \rangle ::= A\langle 109 \rangle$	$\langle 113 \rangle ::= G\langle 114 \rangle U$
$\langle 108 \rangle ::= U\langle 109 \rangle$	$\langle 113 \rangle ::= U\langle 114 \rangle G$
$\langle 108 \rangle ::= G\langle 109 \rangle$	$\langle 114 \rangle ::= A\langle 115 \rangle$
$\langle 108 \rangle ::= C\langle 109 \rangle$	$\langle 114 \rangle ::= U\langle 115 \rangle$
$\langle 108 \rangle ::= \langle 109 \rangle$	$\langle 114 \rangle ::= G\langle 115 \rangle$
$\langle 109 \rangle ::= A\langle S110 \rangle$	$\langle 114 \rangle ::= C\langle 115 \rangle$
$\langle 109 \rangle ::= U\langle S110 \rangle$	$\langle 114 \rangle ::= \langle 115 \rangle$
$\langle 109 \rangle ::= G\langle S110 \rangle$	$\langle 115 \rangle ::= A\langle 116 \rangle$
$\langle 109 \rangle ::= C\langle S110 \rangle$	$\langle 115 \rangle ::= U\langle 116 \rangle$
$\langle 109 \rangle ::= \langle S110 \rangle$	$\langle 115 \rangle ::= G\langle 116 \rangle$
$\langle S110 \rangle ::= \langle 110 \rangle \langle 194 \rangle$	$\langle 115 \rangle ::= C\langle 116 \rangle$
$\langle 110 \rangle ::= A\langle 111 \rangle U$	$\langle 115 \rangle ::= \langle 116 \rangle$
$\langle 110 \rangle ::= U\langle 111 \rangle A$	$\langle 116 \rangle ::= A\langle 117 \rangle$
$\langle 110 \rangle ::= G\langle 111 \rangle C$	$\langle 116 \rangle ::= U\langle 117 \rangle$
$\langle 110 \rangle ::= C\langle 111 \rangle G$	$\langle 116 \rangle ::= G\langle 117 \rangle$
$\langle 110 \rangle ::= G\langle 111 \rangle U$	$\langle 116 \rangle ::= C\langle 117 \rangle$
$\langle 110 \rangle ::= U\langle 111 \rangle G$	$\langle 116 \rangle ::= \langle 117 \rangle$

<117> ::= A<118>	<123> ::= C<124>
<117> ::= U<118>	<123> ::= <124>
<117> ::= G<118>	<124> ::= A<125>
<117> ::= C<118>	<124> ::= U<125>
<117> ::= <118>	<124> ::= G<125>
<118> ::= A<119>	<124> ::= C<125>
<118> ::= U<119>	<124> ::= <125>
<118> ::= G<119>	<125> ::= A<126>
<118> ::= C<119>	<125> ::= U<126>
<118> ::= <119>	<125> ::= G<126>
<119> ::= A<120>	<125> ::= C<126>
<119> ::= U<120>	<125> ::= <126>
<119> ::= G<120>	<126> ::= A<127>
<119> ::= C<120>	<126> ::= U<127>
<119> ::= <120>	<126> ::= G<127>
<120> ::= A<121>	<126> ::= C<127>
<120> ::= U<121>	<126> ::= <127>
<120> ::= G<121>	<127> ::= A<128>
<120> ::= C<121>	<127> ::= U<128>
<120> ::= <121>	<127> ::= G<128>
<121> ::= A<122>	<127> ::= C<128>
<121> ::= U<122>	<127> ::= <128>
<121> ::= G<122>	<128> ::= A<129>
<121> ::= C<122>	<128> ::= U<129>
<121> ::= <122>	<128> ::= G<129>
<122> ::= A<123>	<128> ::= C<129>
<122> ::= U<123>	<128> ::= <129>
<122> ::= G<123>	<129> ::= A<130>
<122> ::= C<123>	<129> ::= U<130>
<122> ::= <123>	<129> ::= G<130>
<123> ::= A<124>	<129> ::= C<130>
<123> ::= U<124>	<129> ::= <130>
<123> ::= G<124>	<130> ::= A<131>

<143> ::= G<144>	<150> ::= A<S151>
<143> ::= C<144>	<150> ::= C<S151>
<143> ::= <144>	<150> ::= U<S151>
<144> ::= A<145>	<S151> ::= <151><184>
<144> ::= U<145>	<151> ::= A<152>U
<144> ::= G<145>	<151> ::= U<152>A
<144> ::= C<145>	<151> ::= G<152>C
<144> ::= <145>	<151> ::= C<152>G
<145> ::= A<146>	<151> ::= G<152>U
<145> ::= U<146>	<151> ::= U<152>G
<145> ::= G<146>	<152> ::= A<153>U
<145> ::= C<146>	<152> ::= U<153>A
<145> ::= <146>	<152> ::= G<153>C
<146> ::= A<147>	<152> ::= C<153>G
<146> ::= U<147>	<152> ::= G<153>U
<146> ::= G<147>	<152> ::= U<153>G
<146> ::= C<147>	<152> ::= C<153>A
<146> ::= <147>	<152> ::= C<153>U
<147> ::= A<148>	<153> ::= A<154>U
<147> ::= U<148>	<153> ::= U<154>A
<147> ::= G<148>	<153> ::= G<154>C
<147> ::= C<148>	<153> ::= C<154>G
<147> ::= <148>	<153> ::= G<154>U
<148> ::= A<149>	<153> ::= U<154>G
<148> ::= U<149>	<153> ::= A<154>G
<148> ::= G<149>	<153> ::= G<154>G
<148> ::= C<149>	<153> ::= G<154>A
<148> ::= <149>	<154> ::= A<155>U
<149> ::= A<150>	<154> ::= U<155>A
<149> ::= U<150>	<154> ::= G<155>C
<149> ::= G<150>	<154> ::= C<155>G
<149> ::= C<150>	<154> ::= G<155>U
<149> ::= <150>	<154> ::= U<155>G

<155> ::= A<156>U	<161> ::= U<162>
<155> ::= U<156>A	<161> ::= G<162>
<155> ::= G<156>C	<161> ::= C<162>
<155> ::= C<156>G	<162> ::= A<163>
<155> ::= G<156>U	<162> ::= U<163>
<155> ::= U<156>G	<162> ::= G<163>
<156> ::= A<157>U	<162> ::= C<163>
<156> ::= U<157>A	<163> ::= A<164>
<156> ::= G<157>C	<163> ::= U<164>
<156> ::= C<157>G	<163> ::= G<164>
<156> ::= G<157>U	<163> ::= C<164>
<156> ::= U<157>G	<164> ::= A<165>
<157> ::= A<158>U	<164> ::= U<165>
<157> ::= U<158>A	<164> ::= G<165>
<157> ::= G<158>C	<164> ::= C<165>
<157> ::= C<158>G	<164> ::= <165>
<157> ::= G<158>U	<165> ::= U<166>
<157> ::= U<158>G	<165> ::= C<166>
<158> ::= A<159>U	<166> ::= A<167>
<158> ::= U<159>A	<166> ::= U<167>
<158> ::= G<159>C	<166> ::= G<167>
<158> ::= C<159>G	<166> ::= C<167>
<158> ::= G<159>U	<167> ::= G<168> \aa{Ad1(<167'>,C)}
<158> ::= U<159>G	<168> ::= C<169> \aa{Ad1(<168'>,G)}
<159> ::= A<160>	<168> ::= U<169> \aa{Ad1(<168'>,G)}
<159> ::= U<160>	<169> ::= U<170> \aa{Ad1(<169'>,A)}
<159> ::= G<160>	<170> ::= A<171> \aa{Ad1(<170'>,U)}
<159> ::= C<160>	<170> ::= G<171> \aa{Ad1(<170'>,C)}
<160> ::= A<161>	<171> ::= A<172> \aa{Ad1(<171'>,U)}
<160> ::= U<161>	<171> ::= G<172> \aa{Ad1(<171'>,C)}
<160> ::= G<161>	<171> ::= C<172> \aa{Ad1(<171'>,G)}
<160> ::= C<161>	<171> ::= G<172> \aa{Ad1(<171'>,U)}
<161> ::= A<162>	<172> ::= C<173> \aa{Ad1(<172'>,G)}

<173> ::= U<174> \aa{Ad1(<173'>,A)}	<194> ::= U<195>
<174> ::= U<175> \aa{Ad1(<174'>,A)}	<194> ::= G<195>
<175> ::= U \aa{Ad1(<175'>,A)}	<194> ::= C<195>
<175> ::= C \aa{Ad1(<175'>,A)}	<195> ::= A<196>
<184> ::= C<185>	<195> ::= U<196>
<184> ::= G<185>	<195> ::= G<196>
<184> ::= U<185>	<195> ::= C<196>
<184> ::= A<185>	<196> ::= A<197>
<185> ::= A<186>	<196> ::= U<197>
<185> ::= U<186>	<196> ::= G<197>
<185> ::= G<186>	<196> ::= C<197>
<185> ::= C<186>	<196> ::= <197>
<185> ::= <186>	<197> ::= A<198>
<186> ::= A<187>	<197> ::= U<198>
<186> ::= U<187>	<197> ::= G<198>
<186> ::= G<187>	<197> ::= C<198>
<186> ::= C<187>	<197> ::= <198>
<186> ::= <187>	<198> ::= A<199>
<187> ::= A<188>	<198> ::= U<199>
<187> ::= U<188>	<198> ::= G<199>
<187> ::= G<188>	<198> ::= C<199>
<187> ::= C<188>	<198> ::= <199>
<187> ::= <188>	<199> ::= A<200>
<188> ::= A<189>	<199> ::= U<200>
<188> ::= U<189>	<199> ::= G<200>
<188> ::= G<189>	<199> ::= C<200>
<188> ::= C<189>	<199> ::= <200>
<188> ::= <189>	<200> ::= A<201>
<189> ::= A	<200> ::= U<201>
<189> ::= U	<200> ::= G<201>
<189> ::= G	<200> ::= C<201>
<189> ::= C	<200> ::= <201>
<194> ::= A<195>	<201> ::= A<202>

<201> ::= U<202>	<207> ::= <208>
<201> ::= G<202>	<208> ::= A<209>
<201> ::= C<202>	<208> ::= U<209>
<201> ::= <202>	<208> ::= G<209>
<202> ::= A<203>	<208> ::= C<209>
<202> ::= U<203>	<208> ::= <209>
<202> ::= G<203>	<209> ::= A<210>
<202> ::= C<203>	<209> ::= U<210>
<202> ::= <203>	<209> ::= G<210>
<203> ::= A<204>	<209> ::= C<210>
<203> ::= U<204>	<209> ::= <210>
<203> ::= G<204>	<210> ::= A<211>
<203> ::= C<204>	<210> ::= U<211>
<203> ::= <204>	<210> ::= G<211>
<204> ::= A<205>	<210> ::= C<211>
<204> ::= U<205>	<210> ::= <211>
<204> ::= G<205>	<211> ::= A<212>
<204> ::= C<205>	<211> ::= U<212>
<204> ::= <205>	<211> ::= G<212>
<205> ::= A<206>	<211> ::= C<212>
<205> ::= U<206>	<211> ::= <212>
<205> ::= G<206>	<212> ::= A<213>
<205> ::= C<206>	<212> ::= U<213>
<205> ::= <206>	<212> ::= G<213>
<206> ::= A<207>	<212> ::= C<213>
<206> ::= U<207>	<212> ::= <213>
<206> ::= G<207>	<213> ::= A<214>
<206> ::= C<207>	<213> ::= U<214>
<206> ::= <207>	<213> ::= G<214>
<207> ::= A<208>	<213> ::= C<214>
<207> ::= U<208>	<213> ::= <214>
<207> ::= G<208>	<214> ::= A<215>
<207> ::= C<208>	<214> ::= U<215>

<214> ::= G<215>	<221> ::= A<222>
<214> ::= C<215>	<221> ::= U<222>
<214> ::= <215>	<221> ::= G<222>
<215> ::= A<216>	<221> ::= C<222>
<215> ::= U<216>	<221> ::= <222>
<215> ::= G<216>	<222> ::= A<223>
<215> ::= C<216>	<222> ::= U<223>
<215> ::= <216>	<222> ::= G<223>
<216> ::= A<217>	<222> ::= C<223>
<216> ::= U<217>	<222> ::= <223>
<216> ::= G<217>	<223> ::= A<224>
<216> ::= C<217>	<223> ::= U<224>
<216> ::= <217>	<223> ::= G<224>
<217> ::= A<218>	<223> ::= C<224>
<217> ::= U<218>	<223> ::= <224>
<217> ::= G<218>	<224> ::= A<225>
<217> ::= C<218>	<224> ::= U<225>
<217> ::= <218>	<224> ::= G<225>
<218> ::= A<219>	<224> ::= C<225>
<218> ::= U<219>	<224> ::= <225>
<218> ::= G<219>	<225> ::= A<226>
<218> ::= C<219>	<225> ::= U<226>
<218> ::= <219>	<225> ::= G<226>
<219> ::= A<220>	<225> ::= C<226>
<219> ::= U<220>	<225> ::= <226>
<219> ::= G<220>	<226> ::= A<227>
<219> ::= C<220>	<226> ::= U<227>
<219> ::= <220>	<226> ::= G<227>
<220> ::= A<221>	<226> ::= C<227>
<220> ::= U<221>	<226> ::= <227>
<220> ::= G<221>	<227> ::= A<228>
<220> ::= C<221>	<227> ::= U<228>
<220> ::= <221>	<227> ::= G<228>

<227> ::= C<228>	<234> ::= U<235>
<227> ::= <228>	<234> ::= G<235>
<228> ::= A<229>	<234> ::= C<235>
<228> ::= U<229>	<234> ::= <235>
<228> ::= G<229>	<235> ::= A<236>
<228> ::= C<229>	<235> ::= U<236>
<228> ::= <229>	<235> ::= G<236>
<229> ::= A<230>	<235> ::= C<236>
<229> ::= U<230>	<235> ::= <236>
<229> ::= G<230>	<236> ::= A<237>
<229> ::= C<230>	<236> ::= U<237>
<229> ::= <230>	<236> ::= G<237>
<230> ::= A<231>	<236> ::= C<237>
<230> ::= U<231>	<236> ::= <237>
<230> ::= G<231>	<237> ::= A<238>
<230> ::= C<231>	<237> ::= U<238>
<230> ::= <231>	<237> ::= G<238>
<231> ::= A<232>	<237> ::= C<238>
<231> ::= U<232>	<237> ::= <238>
<231> ::= G<232>	<238> ::= A<239>
<231> ::= C<232>	<238> ::= U<239>
<231> ::= <232>	<238> ::= G<239>
<232> ::= A<233>	<238> ::= C<239>
<232> ::= U<233>	<238> ::= <239>
<232> ::= G<233>	<239> ::= A<240>
<232> ::= C<233>	<239> ::= U<240>
<232> ::= <233>	<239> ::= G<240>
<233> ::= A<234>	<239> ::= C<240>
<233> ::= U<234>	<239> ::= <240>
<233> ::= G<234>	<240> ::= A<241>
<233> ::= C<234>	<240> ::= U<241>
<233> ::= <234>	<240> ::= G<241>
<234> ::= A<235>	<240> ::= C<241>

<240> ::= <241>	<247> ::= G<248>
<241> ::= A<242>	<247> ::= C<248>
<241> ::= U<242>	<247> ::= <248>
<241> ::= G<242>	<248> ::= A<249>
<241> ::= C<242>	<248> ::= U<249>
<241> ::= <242>	<248> ::= G<249>
<242> ::= A<243>	<248> ::= C<249>
<242> ::= U<243>	<248> ::= <249>
<242> ::= G<243>	<249> ::= A<250>
<242> ::= C<243>	<249> ::= U<250>
<242> ::= <243>	<249> ::= G<250>
<243> ::= A<244>	<249> ::= C<250>
<243> ::= U<244>	<249> ::= <250>
<243> ::= G<244>	<250> ::= A<251>
<243> ::= C<244>	<250> ::= U<251>
<243> ::= <244>	<250> ::= G<251>
<244> ::= A<245>	<250> ::= C<251>
<244> ::= U<245>	<250> ::= <251>
<244> ::= G<245>	<251> ::= A<252>
<244> ::= C<245>	<251> ::= U<252>
<244> ::= <245>	<251> ::= G<252>
<245> ::= A<246>	<251> ::= C<252>
<245> ::= U<246>	<251> ::= <252>
<245> ::= G<246>	<252> ::= A<253>
<245> ::= C<246>	<252> ::= U<253>
<245> ::= <246>	<252> ::= G<253>
<246> ::= A<247>	<252> ::= C<253>
<246> ::= U<247>	<252> ::= <253>
<246> ::= G<247>	<253> ::= A<254>
<246> ::= C<247>	<253> ::= U<254>
<246> ::= <247>	<253> ::= G<254>
<247> ::= A<248>	<253> ::= C<254>
<247> ::= U<248>	<253> ::= <254>

<254> ::= A<255>	<260> ::= C<261>
<254> ::= U<255>	<260> ::= <261>
<254> ::= G<255>	<261> ::= A<262>
<254> ::= C<255>	<261> ::= U<262>
<254> ::= <255>	<261> ::= G<262>
<255> ::= A<256>	<261> ::= C<262>
<255> ::= U<256>	<261> ::= <262>
<255> ::= G<256>	<262> ::= A<263>
<255> ::= C<256>	<262> ::= U<263>
<255> ::= <256>	<262> ::= G<263>
<256> ::= A<257>	<262> ::= C<263>
<256> ::= U<257>	<262> ::= <263>
<256> ::= G<257>	<263> ::= A<264>
<256> ::= C<257>	<263> ::= U<264>
<256> ::= <257>	<263> ::= G<264>
<257> ::= A<258>	<263> ::= C<264>
<257> ::= U<258>	<263> ::= <264>
<257> ::= G<258>	<264> ::= A<265>
<257> ::= C<258>	<264> ::= U<265>
<257> ::= <258>	<264> ::= G<265>
<258> ::= A<259>	<264> ::= C<265>
<258> ::= U<259>	<264> ::= <265>
<258> ::= G<259>	<265> ::= A<266>
<258> ::= C<259>	<265> ::= U<266>
<258> ::= <259>	<265> ::= G<266>
<259> ::= A<260>	<265> ::= C<266>
<259> ::= U<260>	<265> ::= <266>
<259> ::= G<260>	<266> ::= A<267>
<259> ::= C<260>	<266> ::= U<267>
<259> ::= <260>	<266> ::= G<267>
<260> ::= A<261>	<266> ::= C<267>
<260> ::= U<261>	<266> ::= <267>
<260> ::= G<261>	<267> ::= A<268>

<267> ::= U<268>	<273> ::= <274>
<267> ::= G<268>	<274> ::= A<275>
<267> ::= C<268>	<274> ::= U<275>
<267> ::= <268>	<274> ::= G<275>
<268> ::= A<269>	<274> ::= C<275>
<268> ::= U<269>	<274> ::= <275>
<268> ::= G<269>	<275> ::= A<276>
<268> ::= C<269>	<275> ::= U<276>
<268> ::= <269>	<275> ::= G<276>
<269> ::= A<270>	<275> ::= C<276>
<269> ::= U<270>	<275> ::= <276>
<269> ::= G<270>	<276> ::= A<277>
<269> ::= C<270>	<276> ::= U<277>
<269> ::= <270>	<276> ::= G<277>
<270> ::= A<271>	<276> ::= C<277>
<270> ::= U<271>	<276> ::= <277>
<270> ::= G<271>	<277> ::= A<278>
<270> ::= C<271>	<277> ::= U<278>
<270> ::= <271>	<277> ::= G<278>
<271> ::= A<272>	<277> ::= C<278>
<271> ::= U<272>	<277> ::= <278>
<271> ::= G<272>	<278> ::= A<279>
<271> ::= C<272>	<278> ::= U<279>
<271> ::= <272>	<278> ::= G<279>
<272> ::= A<273>	<278> ::= C<279>
<272> ::= U<273>	<278> ::= <279>
<272> ::= G<273>	<279> ::= A<280>
<272> ::= C<273>	<279> ::= U<280>
<272> ::= <273>	<279> ::= G<280>
<273> ::= A<274>	<279> ::= C<280>
<273> ::= U<274>	<279> ::= <280>
<273> ::= G<274>	<280> ::= A<281>
<273> ::= C<274>	<280> ::= U<281>

<280> ::= G<281>	<287> ::= A<288>
<280> ::= C<281>	<287> ::= U<288>
<280> ::= <281>	<287> ::= G<288>
<281> ::= A<282>	<287> ::= C<288>
<281> ::= U<282>	<287> ::= <288>
<281> ::= G<282>	<288> ::= A<289>
<281> ::= C<282>	<288> ::= U<289>
<281> ::= <282>	<288> ::= G<289>
<282> ::= A<283>	<288> ::= C<289>
<282> ::= U<283>	<288> ::= <289>
<282> ::= G<283>	<289> ::= A<290>
<282> ::= C<283>	<289> ::= U<290>
<282> ::= <283>	<289> ::= G<290>
<283> ::= A<284>	<289> ::= C<290>
<283> ::= U<284>	<289> ::= <290>
<283> ::= G<284>	<290> ::= A<291>
<283> ::= C<284>	<290> ::= U<291>
<283> ::= <284>	<290> ::= G<291>
<284> ::= A<285>	<290> ::= C<291>
<284> ::= U<285>	<290> ::= <291>
<284> ::= G<285>	<291> ::= A<292>
<284> ::= C<285>	<291> ::= U<292>
<284> ::= <285>	<291> ::= G<292>
<285> ::= A<286>	<291> ::= C<292>
<285> ::= U<286>	<292> ::= A<293>
<285> ::= G<286>	<292> ::= U<293>
<285> ::= C<286>	<292> ::= G<293>
<285> ::= <286>	<292> ::= C<293>
<286> ::= A<287>	<293> ::= A<294>
<286> ::= U<287>	<293> ::= U<294>
<286> ::= G<287>	<293> ::= G<294>
<286> ::= C<287>	<293> ::= C<294>
<286> ::= <287>	<293> ::= <294>

<294> ::= A<295>	<303> ::= A<304>
<294> ::= U<295>	<303> ::= U<304>
<294> ::= G<295>	<303> ::= G<304>
<294> ::= C<295>	<303> ::= C<304>
<294> ::= <295>	<304> ::= <172'><305>
<295> ::= A<296>	<305> ::= <171'><306>
<295> ::= U<296>	<306> ::= <170'><307>
<295> ::= G<296>	<307> ::= A<308>
<295> ::= C<296>	<307> ::= U<308>
<295> ::= <296>	<307> ::= G<308>
<296> ::= A<297>	<307> ::= C<308>
<296> ::= U<297>	<307> ::= <308>
<296> ::= G<297>	<308> ::= A<309>
<296> ::= C<297>	<308> ::= U<309>
<296> ::= <297>	<308> ::= G<309>
<297> ::= A<298>	<308> ::= C<309>
<297> ::= U<298>	<308> ::= <309>
<297> ::= G<298>	<309> ::= <169'><310>
<297> ::= C<298>	<310> ::= <168'><311>
<297> ::= <298>	<311> ::= <167'><312>
<298> ::= A<299>	<312> ::= A<313>
<298> ::= U<299>	<312> ::= U<313>
<298> ::= G<299>	<312> ::= G<313>
<298> ::= C<299>	<312> ::= C<313>
<298> ::= <299>	<312> ::= <313>
<299> ::= A<300>	<313> ::= A<314>
<299> ::= U<300>	<313> ::= U<314>
<299> ::= G<300>	<313> ::= G<314>
<299> ::= C<300>	<313> ::= C<314>
<299> ::= <300>	<313> ::= <314>
<300> ::= <175'><301>	<314> ::= A<315>
<301> ::= <174'><302>	<314> ::= U<315>
<302> ::= <173'><303>	<314> ::= G<315>

<314> ::= C<315>	<321> ::= G<322>
<315> ::= A<316>	<321> ::= C<322>
<315> ::= U<316>	<321> ::= <322>
<315> ::= G<316>	<322> ::= A<323>
<315> ::= C<316>	<322> ::= U<323>
<315> ::= <316>	<322> ::= G<323>
<316> ::= A<317>	<322> ::= C<323>
<316> ::= U<317>	<322> ::= <323>
<316> ::= G<317>	<323> ::= A<324>
<316> ::= C<317>	<323> ::= U<324>
<316> ::= <317>	<323> ::= G<324>
<317> ::= A<318>	<323> ::= C<324>
<317> ::= U<318>	<323> ::= <324>
<317> ::= G<318>	<324> ::= A<325>
<317> ::= C<318>	<324> ::= U<325>
<317> ::= <318>	<324> ::= G<325>
<318> ::= A<319>	<324> ::= C<325>
<318> ::= U<319>	<324> ::= <325>
<318> ::= G<319>	<325> ::= A<326>
<318> ::= C<319>	<325> ::= U<326>
<318> ::= <319>	<325> ::= G<326>
<319> ::= A<320>	<325> ::= C<326>
<319> ::= U<320>	<325> ::= <326>
<319> ::= G<320>	<326> ::= A<327>
<319> ::= C<320>	<326> ::= U<327>
<319> ::= <320>	<326> ::= G<327>
<320> ::= A<321>	<326> ::= C<327>
<320> ::= U<321>	<326> ::= <327>
<320> ::= G<321>	<327> ::= A<328>
<320> ::= C<321>	<327> ::= U<328>
<320> ::= <321>	<327> ::= G<328>
<321> ::= A<322>	<327> ::= C<328>
<321> ::= U<322>	<327> ::= <328>

<328> ::= A<329>	<334> ::= C<335>
<328> ::= U<329>	<334> ::= <335>
<328> ::= G<329>	<335> ::= A<336>
<328> ::= C<329>	<335> ::= U<336>
<328> ::= <329>	<335> ::= G<336>
<329> ::= A<330>	<335> ::= C<336>
<329> ::= U<330>	<335> ::= <336>
<329> ::= G<330>	<336> ::= A<337>
<329> ::= C<330>	<336> ::= U<337>
<329> ::= <330>	<336> ::= G<337>
<330> ::= A<331>	<336> ::= C<337>
<330> ::= U<331>	<336> ::= <337>
<330> ::= G<331>	<337> ::= A<338>
<330> ::= C<331>	<337> ::= U<338>
<330> ::= <331>	<337> ::= G<338>
<331> ::= A<332>	<337> ::= C<338>
<331> ::= U<332>	<337> ::= <338>
<331> ::= G<332>	<338> ::= A<339>
<331> ::= C<332>	<338> ::= U<339>
<331> ::= <332>	<338> ::= G<339>
<332> ::= A<333>	<338> ::= C<339>
<332> ::= U<333>	<338> ::= <339>
<332> ::= G<333>	<339> ::= A<340>
<332> ::= C<333>	<339> ::= U<340>
<332> ::= <333>	<339> ::= G<340>
<333> ::= A<334>	<339> ::= C<340>
<333> ::= U<334>	<339> ::= <340>
<333> ::= G<334>	<340> ::= A<341>
<333> ::= C<334>	<340> ::= U<341>
<333> ::= <334>	<340> ::= G<341>
<334> ::= A<335>	<340> ::= C<341>
<334> ::= U<335>	<340> ::= <341>
<334> ::= G<335>	<341> ::= A<342>

<341> ::= U<342>

<341> ::= G<342>

<341> ::= C<342>

<341> ::= <342>

<342> ::= A<343>

<342> ::= U<343>

<342> ::= G<343>

<342> ::= C<343>

<342> ::= <343>

<343> ::= A<344>

<343> ::= U<344>

<343> ::= G<344>

<343> ::= C<344>

<343> ::= <344>

<344> ::= A<345>

<344> ::= U<345>

<344> ::= G<345>

<344> ::= C<345>

<344> ::= <345>

<345> ::= A<346>

<345> ::= U<346>

<345> ::= G<346>

<345> ::= C<346>

<345> ::= <346>

<346> ::= A<347>

<346> ::= U<347>

<346> ::= G<347>

<346> ::= C<347>

<346> ::= <347>

<347> ::= A<348>

<347> ::= U<348>

<347> ::= G<348>

<347> ::= C<348>

<347> ::= <348>

<348> ::= A<349>

<348> ::= U<349>

<348> ::= G<349>

<348> ::= C<349>

<348> ::= <349>

<349> ::= A<350>

<349> ::= U<350>

<349> ::= G<350>

<349> ::= C<350>

<349> ::= <350>

<350> ::= A<351>

<350> ::= U<351>

<350> ::= G<351>

<350> ::= C<351>

<350> ::= <351>

<351> ::= A<352>

<351> ::= U<352>

<351> ::= G<352>

<351> ::= C<352>

<351> ::= <352>

<352> ::= A<353>

<352> ::= U<353>

<352> ::= G<353>

<352> ::= C<353>

<352> ::= <353>

<353> ::= A<354>

<353> ::= U<354>

<353> ::= G<354>

<353> ::= C<354>

<353> ::= <354>

<354> ::= A<355>

<354> ::= U<355>

<354> ::= G<355>	<361> ::= A<362>
<354> ::= C<355>	<361> ::= U<362>
<354> ::= <355>	<361> ::= G<362>
<355> ::= A<356>	<361> ::= C<362>
<355> ::= U<356>	<361> ::= <362>
<355> ::= G<356>	<362> ::= A<363>
<355> ::= C<356>	<362> ::= U<363>
<355> ::= <356>	<362> ::= G<363>
<356> ::= A<357>	<362> ::= C<363>
<356> ::= U<357>	<362> ::= <363>
<356> ::= G<357>	<363> ::= A<364>
<356> ::= C<357>	<363> ::= U<364>
<356> ::= <357>	<363> ::= G<364>
<357> ::= A<358>	<363> ::= C<364>
<357> ::= U<358>	<363> ::= <364>
<357> ::= G<358>	<364> ::= A<365>
<357> ::= C<358>	<364> ::= U<365>
<357> ::= <358>	<364> ::= G<365>
<358> ::= A<359>	<364> ::= C<365>
<358> ::= U<359>	<364> ::= <365>
<358> ::= G<359>	<365> ::= A<366>
<358> ::= C<359>	<365> ::= U<366>
<358> ::= <359>	<365> ::= G<366>
<359> ::= A<360>	<365> ::= C<366>
<359> ::= U<360>	<365> ::= <366>
<359> ::= G<360>	<366> ::= A<367>
<359> ::= C<360>	<366> ::= U<367>
<359> ::= <360>	<366> ::= G<367>
<360> ::= A<361>	<366> ::= C<367>
<360> ::= U<361>	<366> ::= <367>
<360> ::= G<361>	<367> ::= A<368>
<360> ::= C<361>	<367> ::= U<368>
<360> ::= <361>	<367> ::= G<368>

<367> ::= C<368>	<374> ::= C<375>G
<367> ::= <368>	<374> ::= C<375>C
<368> ::= A<369>	<374> ::= U<375>G
<368> ::= U<369>	<375> ::= U<376>A
<368> ::= G<369>	<375> ::= C<376>G
<368> ::= C<369>	<375> ::= C<376>A
<368> ::= <369>	<375> ::= U<376>G
<369> ::= A<370>	<376> ::= A<S377>
<369> ::= U<370>	<376> ::= U<S377>
<369> ::= G<370>	<376> ::= G<S377>
<369> ::= C<370>	<376> ::= C<S377>
<369> ::= <370>	<S377> ::= <377><668>
<370> ::= A<371>	<S377> ::= <377>
<370> ::= U<371>	<377> ::= A<378>A
<370> ::= G<371>	<377> ::= G<378>C
<370> ::= C<371>	<377> ::= G<378>U
<370> ::= <371>	<378> ::= A<379>U
<371> ::= A<372>	<378> ::= U<379>A
<371> ::= U<372>	<378> ::= G<379>C
<371> ::= G<372>	<378> ::= C<379>G
<371> ::= C<372>	<378> ::= G<379>U
<371> ::= <372>	<378> ::= U<379>G
<372> ::= A<373>	<379> ::= A<380>
<372> ::= U<373>	<379> ::= U<380>
<372> ::= G<373>	<379> ::= G<380>
<372> ::= C<373>	<379> ::= C<380>
<372> ::= <373>	<379> ::= <380>
<373> ::= A<S374>	<380> ::= A<381>
<373> ::= U<S374>	<380> ::= U<381>
<373> ::= G<S374>	<380> ::= G<381>
<373> ::= C<S374>	<380> ::= C<381>
<373> ::= <S374>	<380> ::= <381>
<S374> ::= <374><671>	<381> ::= A<382>

<381> ::= U<382>	<388> ::= U<389>
<381> ::= G<382>	<388> ::= G<389>
<381> ::= C<382>	<388> ::= C<389>
<382> ::= A<383>	<388> ::= <389>
<382> ::= U<383>	<389> ::= A<390>
<382> ::= G<383>	<389> ::= U<390>
<382> ::= C<383>	<389> ::= G<390>
<383> ::= A<384>	<389> ::= C<390>
<383> ::= U<384>	<389> ::= <390>
<383> ::= G<384>	<390> ::= A<391>
<383> ::= C<384>	<390> ::= U<391>
<383> ::= <384>	<390> ::= G<391>
<384> ::= A<385>	<390> ::= C<391>
<384> ::= U<385>	<390> ::= <391>
<384> ::= G<385>	<391> ::= A<392>
<384> ::= C<385>	<391> ::= U<392>
<384> ::= <385>	<391> ::= G<392>
<385> ::= A<386>	<391> ::= C<392>
<385> ::= U<386>	<391> ::= <392>
<385> ::= G<386>	<392> ::= A<393>
<385> ::= C<386>	<392> ::= U<393>
<385> ::= <386>	<392> ::= G<393>
<386> ::= A<387>	<392> ::= C<393>
<386> ::= U<387>	<392> ::= <393>
<386> ::= G<387>	<393> ::= A<394>
<386> ::= C<387>	<393> ::= U<394>
<386> ::= <387>	<393> ::= G<394>
<387> ::= A<388>	<393> ::= C<394>
<387> ::= U<388>	<393> ::= <394>
<387> ::= G<388>	<394> ::= A<395>
<387> ::= C<388>	<394> ::= U<395>
<387> ::= <388>	<394> ::= G<395>
<388> ::= A<389>	<394> ::= C<395>

<394> ::= <395>	<401> ::= G<402>
<395> ::= A<396>	<401> ::= C<402>
<395> ::= U<396>	<401> ::= <402>
<395> ::= G<396>	<402> ::= A<403>
<395> ::= C<396>	<402> ::= U<403>
<395> ::= <396>	<402> ::= G<403>
<396> ::= A<397>	<402> ::= C<403>
<396> ::= U<397>	<402> ::= <403>
<396> ::= G<397>	<403> ::= A<404>
<396> ::= C<397>	<403> ::= U<404>
<396> ::= <397>	<403> ::= G<404>
<397> ::= A<398>	<403> ::= C<404>
<397> ::= U<398>	<403> ::= <404>
<397> ::= G<398>	<404> ::= A<405>
<397> ::= C<398>	<404> ::= U<405>
<397> ::= <398>	<404> ::= G<405>
<398> ::= A<399>	<404> ::= C<405>
<398> ::= U<399>	<404> ::= <405>
<398> ::= G<399>	<405> ::= A<406>
<398> ::= C<399>	<405> ::= U<406>
<398> ::= <399>	<405> ::= G<406>
<399> ::= A<400>	<405> ::= C<406>
<399> ::= U<400>	<405> ::= <406>
<399> ::= G<400>	<406> ::= A<407>
<399> ::= C<400>	<406> ::= U<407>
<399> ::= <400>	<406> ::= G<407>
<400> ::= A<401>	<406> ::= C<407>
<400> ::= U<401>	<406> ::= <407>
<400> ::= G<401>	<407> ::= A<408>
<400> ::= C<401>	<407> ::= U<408>
<400> ::= <401>	<407> ::= G<408>
<401> ::= A<402>	<407> ::= C<408>
<401> ::= U<402>	<407> ::= <408>

<408> ::= A<409>	<414> ::= C<415>
<408> ::= U<409>	<414> ::= <415>
<408> ::= G<409>	<415> ::= A<416>
<408> ::= C<409>	<415> ::= U<416>
<408> ::= <409>	<415> ::= G<416>
<409> ::= A<410>	<415> ::= C<416>
<409> ::= U<410>	<415> ::= <416>
<409> ::= G<410>	<416> ::= A<417>
<409> ::= C<410>	<416> ::= U<417>
<409> ::= <410>	<416> ::= G<417>
<410> ::= A<411>	<416> ::= C<417>
<410> ::= U<411>	<416> ::= <417>
<410> ::= G<411>	<417> ::= A<418>
<410> ::= C<411>	<417> ::= U<418>
<410> ::= <411>	<417> ::= G<418>
<411> ::= A<412>	<417> ::= C<418>
<411> ::= U<412>	<417> ::= <418>
<411> ::= G<412>	<418> ::= A<419>
<411> ::= C<412>	<418> ::= U<419>
<411> ::= <412>	<418> ::= G<419>
<412> ::= A<413>	<418> ::= C<419>
<412> ::= U<413>	<418> ::= <419>
<412> ::= G<413>	<419> ::= A<420>
<412> ::= C<413>	<419> ::= U<420>
<412> ::= <413>	<419> ::= G<420>
<413> ::= A<414>	<419> ::= C<420>
<413> ::= U<414>	<419> ::= <420>
<413> ::= G<414>	<420> ::= A<421>
<413> ::= C<414>	<420> ::= U<421>
<413> ::= <414>	<420> ::= G<421>
<414> ::= A<415>	<420> ::= C<421>
<414> ::= U<415>	<420> ::= <421>
<414> ::= G<415>	<421> ::= A<422>

<421> ::= U<422>	<427> ::= <428>
<421> ::= G<422>	<428> ::= A<429>
<421> ::= C<422>	<428> ::= U<429>
<421> ::= <422>	<428> ::= G<429>
<422> ::= A<423>	<428> ::= C<429>
<422> ::= U<423>	<428> ::= <429>
<422> ::= G<423>	<429> ::= A<430>
<422> ::= C<423>	<429> ::= U<430>
<422> ::= <423>	<429> ::= G<430>
<423> ::= A<424>	<429> ::= C<430>
<423> ::= U<424>	<429> ::= <430>
<423> ::= G<424>	<430> ::= A<431>
<423> ::= C<424>	<430> ::= U<431>
<423> ::= <424>	<430> ::= G<431>
<424> ::= A<425>	<430> ::= C<431>
<424> ::= U<425>	<430> ::= <431>
<424> ::= G<425>	<431> ::= A<432>
<424> ::= C<425>	<431> ::= U<432>
<424> ::= <425>	<431> ::= G<432>
<425> ::= A<426>	<431> ::= C<432>
<425> ::= U<426>	<431> ::= <432>
<425> ::= G<426>	<432> ::= A<433>
<425> ::= C<426>	<432> ::= U<433>
<425> ::= <426>	<432> ::= G<433>
<426> ::= A<427>	<432> ::= C<433>
<426> ::= U<427>	<432> ::= <433>
<426> ::= G<427>	<433> ::= A<434>
<426> ::= C<427>	<433> ::= U<434>
<426> ::= <427>	<433> ::= G<434>
<427> ::= A<428>	<433> ::= C<434>
<427> ::= U<428>	<433> ::= <434>
<427> ::= G<428>	<434> ::= A<435>
<427> ::= C<428>	<434> ::= U<435>

<434> ::= G<435>	<441> ::= A<442>
<434> ::= C<435>	<441> ::= U<442>
<434> ::= <435>	<441> ::= G<442>
<435> ::= A<436>	<441> ::= C<442>
<435> ::= U<436>	<441> ::= <442>
<435> ::= G<436>	<442> ::= A<443>
<435> ::= C<436>	<442> ::= U<443>
<435> ::= <436>	<442> ::= G<443>
<436> ::= A<437>	<442> ::= C<443>
<436> ::= U<437>	<442> ::= <443>
<436> ::= G<437>	<443> ::= A<444>
<436> ::= C<437>	<443> ::= U<444>
<436> ::= <437>	<443> ::= G<444>
<437> ::= A<438>	<443> ::= C<444>
<437> ::= U<438>	<443> ::= <444>
<437> ::= G<438>	<444> ::= A<445>
<437> ::= C<438>	<444> ::= U<445>
<437> ::= <438>	<444> ::= G<445>
<438> ::= A<439>	<444> ::= C<445>
<438> ::= U<439>	<444> ::= <445>
<438> ::= G<439>	<445> ::= A<446>
<438> ::= C<439>	<445> ::= U<446>
<438> ::= <439>	<445> ::= G<446>
<439> ::= A<440>	<445> ::= C<446>
<439> ::= U<440>	<445> ::= <446>
<439> ::= G<440>	<446> ::= A<447>
<439> ::= C<440>	<446> ::= U<447>
<439> ::= <440>	<446> ::= G<447>
<440> ::= A<441>	<446> ::= C<447>
<440> ::= U<441>	<446> ::= <447>
<440> ::= G<441>	<447> ::= A<448>
<440> ::= C<441>	<447> ::= U<448>
<440> ::= <441>	<447> ::= G<448>

<447> ::= C<448>
 <447> ::= <448>
 <448> ::= A<449>
 <448> ::= U<449>
 <448> ::= G<449>
 <448> ::= C<449>
 <448> ::= <449>
 <449> ::= A<450>
 <449> ::= U<450>
 <449> ::= G<450>
 <449> ::= C<450>
 <449> ::= <450>
 <450> ::= A<451>
 <450> ::= U<451>
 <450> ::= G<451>
 <450> ::= C<451>
 <450> ::= <451>
 <451> ::= A<452>
 <451> ::= U<452>
 <451> ::= G<452>
 <451> ::= C<452>
 <451> ::= <452>
 <452> ::= A<453>
 <452> ::= U<453>
 <452> ::= G<453>
 <452> ::= C<453>
 <452> ::= <453>
 <453> ::= A<454>
 <453> ::= U<454>
 <453> ::= G<454>
 <453> ::= C<454>
 <453> ::= <454>
 <454> ::= A<455>

<454> ::= U<455>
 <454> ::= G<455>
 <454> ::= C<455>
 <454> ::= <455>
 <455> ::= A<456>
 <455> ::= U<456>
 <455> ::= G<456>
 <455> ::= C<456>
 <455> ::= <456>
 <456> ::= A<457>
 <456> ::= U<457>
 <456> ::= G<457>
 <456> ::= C<457>
 <456> ::= <457>
 <457> ::= A<458>
 <457> ::= U<458>
 <457> ::= G<458>
 <457> ::= C<458>
 <457> ::= <458>
 <458> ::= A<459>
 <458> ::= U<459>
 <458> ::= G<459>
 <458> ::= C<459>
 <458> ::= <459>
 <459> ::= A<460>
 <459> ::= U<460>
 <459> ::= G<460>
 <459> ::= C<460>
 <459> ::= <460>
 <460> ::= A<461>
 <460> ::= U<461>
 <460> ::= G<461>
 <460> ::= C<461>

<460> ::= <461>	<467> ::= G<S468>
<461> ::= A<462>	<467> ::= C<S468>
<461> ::= U<462>	<467> ::= <S468>
<461> ::= G<462>	<S468> ::= <468><593>
<461> ::= C<462>	<S468> ::= <468>
<461> ::= <462>	<468> ::= C<469>G
<462> ::= A<463>	<468> ::= G<469>C
<462> ::= U<463>	<468> ::= U<469>G
<462> ::= G<463>	<468> ::= G<469>U
<462> ::= C<463>	<468> ::= G<469>A
<462> ::= <463>	<468> ::= C<469>C
<463> ::= A<464>	<468> ::= G<469>G
<463> ::= U<464>	<469> ::= A<470>U
<463> ::= G<464>	<469> ::= U<470>A
<463> ::= C<464>	<469> ::= G<470>C
<463> ::= <464>	<469> ::= C<470>G
<464> ::= A<465>	<470> ::= A<471>
<464> ::= U<465>	<470> ::= U<471>
<464> ::= G<465>	<470> ::= G<471>
<464> ::= C<465>	<470> ::= C<471>
<464> ::= <465>	<470> ::= <471>
<465> ::= A<466>	<471> ::= A<472>
<465> ::= U<466>	<471> ::= U<472>
<465> ::= G<466>	<471> ::= G<472>
<465> ::= C<466>	<471> ::= C<472>
<465> ::= <466>	<471> ::= <472>
<466> ::= A<467>	<472> ::= A<473>
<466> ::= U<467>	<472> ::= U<473>
<466> ::= G<467>	<472> ::= G<473>
<466> ::= C<467>	<472> ::= C<473>
<466> ::= <467>	<472> ::= <473>
<467> ::= A<S468>	<473> ::= A<474>
<467> ::= U<S468>	<473> ::= U<474>

<473> ::= G<474>	<480> ::= C<481>
<473> ::= C<474>	<480> ::= <481>
<473> ::= <474>	<481> ::= A<482>
<474> ::= A<475>	<481> ::= U<482>
<474> ::= U<475>	<481> ::= G<482>
<474> ::= G<475>	<481> ::= C<482>
<474> ::= C<475>	<481> ::= <482>
<474> ::= <475>	<482> ::= A<483>
<475> ::= A<476>	<482> ::= U<483>
<475> ::= U<476>	<482> ::= G<483>
<475> ::= G<476>	<482> ::= C<483>
<475> ::= C<476>	<482> ::= <483>
<475> ::= <476>	<483> ::= A<484>
<476> ::= A<477>	<483> ::= U<484>
<476> ::= U<477>	<483> ::= G<484>
<476> ::= G<477>	<483> ::= C<484>
<476> ::= C<477>	<484> ::= A<485>
<476> ::= <477>	<484> ::= U<485>
<477> ::= A<S478>	<484> ::= G<485>
<477> ::= U<S478>	<484> ::= C<485>
<477> ::= G<S478>	<485> ::= A<486>
<477> ::= C<S478>	<485> ::= U<486>
<477> ::= <S478>	<485> ::= G<486>
<S478> ::= <478><581>	<485> ::= C<486>
<478> ::= U<479>A	<485> ::= <486>
<478> ::= C<479>G	<486> ::= A<487>
<478> ::= U<479>G	<486> ::= U<487>
<479> ::= G<480>C	<486> ::= G<487>
<479> ::= C<480>G	<486> ::= C<487>
<479> ::= G<480>U	<486> ::= <487>
<480> ::= A<481>	<487> ::= A<488>
<480> ::= U<481>	<487> ::= U<488>
<480> ::= G<481>	<487> ::= G<488>

<487> ::= C<488>	<494> ::= U<495>
<487> ::= <488>	<494> ::= G<495>
<488> ::= A<489>	<494> ::= C<495>
<488> ::= U<489>	<495> ::= G<496>
<488> ::= G<489>	<496> ::= G<497>
<488> ::= C<489>	<497> ::= C<S498>
<488> ::= <489>	<S498> ::= <498><552>
<489> ::= A<490>	<498> ::= A<499>U
<489> ::= U<490>	<498> ::= U<499>A
<489> ::= G<490>	<498> ::= G<499>C
<489> ::= C<490>	<498> ::= C<499>G
<489> ::= <490>	<498> ::= G<499>U
<490> ::= A<491>	<498> ::= U<499>G
<490> ::= U<491>	<499> ::= A<500>C
<490> ::= G<491>	<499> ::= G<500>C
<490> ::= C<491>	<499> ::= G<500>U
<490> ::= <491>	<500> ::= A<501>U
<491> ::= A<492>	<500> ::= U<501>A
<491> ::= U<492>	<500> ::= G<501>C
<491> ::= G<492>	<500> ::= C<501>G
<491> ::= C<492>	<500> ::= G<501>U
<491> ::= <492>	<500> ::= U<501>G
<492> ::= A<493>	<501> ::= A<502>U
<492> ::= U<493>	<501> ::= A<502>C
<492> ::= G<493>	<501> ::= G<502>C
<492> ::= C<493>	<502> ::= A<503>U
<492> ::= <493>	<502> ::= U<503>A
<493> ::= A<494>	<502> ::= G<503>C
<493> ::= U<494>	<502> ::= C<503>G
<493> ::= G<494>	<502> ::= G<503>U
<493> ::= C<494>	<502> ::= U<503>G
<493> ::= <494>	<503> ::= A<504>
<494> ::= A<495>	<503> ::= U<504>

<503> ::= G<504>	<509> ::= <510>
<503> ::= C<504>	<510> ::= A<511>
<503> ::= <504>	<510> ::= U<511>
<504> ::= U<505>G	<510> ::= G<511>
<504> ::= C<505>G	<510> ::= C<511>
<504> ::= C<505>A	<510> ::= <511>
<505> ::= A<506>U	<511> ::= A<512>
<505> ::= U<506>A	<511> ::= U<512>
<505> ::= G<506>C	<511> ::= G<512>
<505> ::= C<506>G	<511> ::= C<512>
<505> ::= C<506>A	<511> ::= <512>
<505> ::= G<506>U	<512> ::= A<513>
<505> ::= U<506>G	<512> ::= U<513>
<506> ::= A<507>U	<512> ::= G<513>
<506> ::= U<507>A	<512> ::= C<513>
<506> ::= G<507>C	<512> ::= <513>
<506> ::= C<507>G	<513> ::= A<514>
<506> ::= G<507>U	<513> ::= U<514>
<506> ::= U<507>G	<513> ::= G<514>
<507> ::= A<508>U	<513> ::= C<514>
<507> ::= U<508>A	<513> ::= <514>
<507> ::= G<508>C	<514> ::= A<515>
<507> ::= C<508>G	<514> ::= U<515>
<507> ::= G<508>U	<514> ::= G<515>
<507> ::= U<508>G	<514> ::= C<515>
<508> ::= A<509>	<514> ::= <515>
<508> ::= U<509>	<515> ::= A<516>
<508> ::= G<509>	<515> ::= U<516>
<508> ::= C<509>	<515> ::= G<516>
<509> ::= A<510>	<515> ::= C<516>
<509> ::= U<510>	<515> ::= <516>
<509> ::= G<510>	<516> ::= A<517>
<509> ::= C<510>	<516> ::= U<517>

<516> ::= G<517>	<523> ::= A<524>
<516> ::= C<517>	<523> ::= U<524>
<516> ::= <517>	<523> ::= G<524>
<517> ::= A<518>	<523> ::= C<524>
<517> ::= U<518>	<523> ::= <524>
<517> ::= G<518>	<524> ::= A<525>
<517> ::= C<518>	<524> ::= U<525>
<517> ::= <518>	<524> ::= G<525>
<518> ::= A<519>	<524> ::= C<525>
<518> ::= U<519>	<524> ::= <525>
<518> ::= G<519>	<525> ::= A<526>
<518> ::= C<519>	<525> ::= U<526>
<518> ::= <519>	<525> ::= G<526>
<519> ::= A<520>	<525> ::= C<526>
<519> ::= U<520>	<525> ::= <526>
<519> ::= G<520>	<526> ::= A<527>
<519> ::= C<520>	<526> ::= U<527>
<519> ::= <520>	<526> ::= G<527>
<520> ::= A<521>	<526> ::= C<527>
<520> ::= U<521>	<526> ::= <527>
<520> ::= G<521>	<527> ::= A<528>
<520> ::= C<521>	<527> ::= U<528>
<520> ::= <521>	<527> ::= G<528>
<521> ::= A<522>	<527> ::= C<528>
<521> ::= U<522>	<527> ::= <528>
<521> ::= G<522>	<528> ::= A<529>
<521> ::= C<522>	<528> ::= U<529>
<521> ::= <522>	<528> ::= G<529>
<522> ::= A<523>	<528> ::= C<529>
<522> ::= U<523>	<528> ::= <529>
<522> ::= G<523>	<529> ::= A<530>
<522> ::= C<523>	<529> ::= U<530>
<522> ::= <523>	<529> ::= G<530>

<529> ::= C<530>	<536> ::= U<537>
<529> ::= <530>	<536> ::= G<537>
<530> ::= A<531>	<536> ::= C<537>
<530> ::= U<531>	<536> ::= <537>
<530> ::= G<531>	<537> ::= A<538>
<530> ::= C<531>	<537> ::= U<538>
<530> ::= <531>	<537> ::= G<538>
<531> ::= A<532>	<537> ::= C<538>
<531> ::= U<532>	<537> ::= <538>
<531> ::= G<532>	<538> ::= A<539>
<531> ::= C<532>	<538> ::= U<539>
<531> ::= <532>	<538> ::= G<539>
<532> ::= A<533>	<538> ::= C<539>
<532> ::= U<533>	<538> ::= <539>
<532> ::= G<533>	<539> ::= A<540>
<532> ::= C<533>	<539> ::= U<540>
<532> ::= <533>	<539> ::= G<540>
<533> ::= A<534>	<539> ::= C<540>
<533> ::= U<534>	<539> ::= <540>
<533> ::= G<534>	<540> ::= A<541>
<533> ::= C<534>	<540> ::= U<541>
<533> ::= <534>	<540> ::= G<541>
<534> ::= A<535>	<540> ::= C<541>
<534> ::= U<535>	<540> ::= <541>
<534> ::= G<535>	<541> ::= A<542>
<534> ::= C<535>	<541> ::= U<542>
<534> ::= <535>	<541> ::= G<542>
<535> ::= A<536>	<541> ::= C<542>
<535> ::= U<536>	<542> ::= A
<535> ::= G<536>	<542> ::= U
<535> ::= C<536>	<542> ::= G
<535> ::= <536>	<542> ::= C
<536> ::= A<537>	<552> ::= A<S553>

<S553> ::= <553><568>

<553> ::= A<554>U

<553> ::= U<554>A

<553> ::= G<554>C

<553> ::= C<554>G

<553> ::= G<554>U

<553> ::= U<554>G

<554> ::= A<555>U

<554> ::= U<555>A

<554> ::= G<555>C

<554> ::= C<555>G

<554> ::= G<555>U

<554> ::= U<555>G

<555> ::= A<556>U

<555> ::= U<556>A

<555> ::= G<556>C

<555> ::= C<556>G

<555> ::= G<556>U

<555> ::= U<556>G

<556> ::= A<557>U

<556> ::= U<557>A

<556> ::= G<557>C

<556> ::= C<557>G

<556> ::= G<557>U

<556> ::= U<557>G

<557> ::= A<558>

<557> ::= U<558>

<557> ::= G<558>

<557> ::= C<558>

<557> ::= <558>

<558> ::= A<559>

<558> ::= U<559>

<558> ::= G<559>

<558> ::= C<559>

<558> ::= <559>

<559> ::= A<560>

<559> ::= U<560>

<559> ::= G<560>

<559> ::= C<560>

<560> ::= A<561>

<560> ::= U<561>

<560> ::= G<561>

<560> ::= C<561>

<561> ::= A<562>

<561> ::= U<562>

<561> ::= G<562>

<561> ::= C<562>

<561> ::= A

<561> ::= U

<561> ::= G

<561> ::= C

<562> ::= A<563>

<562> ::= U<563>

<562> ::= G<563>

<562> ::= C<563>

<562> ::= <563>

<563> ::= A

<563> ::= U

<563> ::= G

<563> ::= C

<568> ::= A<569>

<568> ::= U<569>

<568> ::= G<569>

<568> ::= C<569>

<569> ::= A<570>

<569> ::= U<570>

<569> ::= G<570>	<576> ::= G<577>
<569> ::= C<570>	<576> ::= C<577>
<570> ::= A<571>	<576> ::= <577>
<570> ::= U<571>	<577> ::= A<578>
<570> ::= G<571>	<577> ::= U<578>
<570> ::= C<571>	<577> ::= G<578>
<571> ::= A<572>	<577> ::= C<578>
<571> ::= U<572>	<577> ::= <578>
<571> ::= G<572>	<578> ::= A
<571> ::= C<572>	<578> ::= U
<572> ::= A<573>	<578> ::= G
<572> ::= U<573>	<578> ::= C
<572> ::= G<573>	<581> ::= A<582>
<572> ::= C<573>	<581> ::= U<582>
<573> ::= A<574>	<581> ::= G<582>
<573> ::= U<574>	<581> ::= C<582>
<573> ::= G<574>	<582> ::= A<583>
<573> ::= C<574>	<582> ::= U<583>
<574> ::= A<575>	<582> ::= G<583>
<574> ::= U<575>	<582> ::= C<583>
<574> ::= G<575>	<582> ::= A
<574> ::= C<575>	<582> ::= U
<574> ::= A	<582> ::= G
<574> ::= U	<582> ::= C
<574> ::= G	<583> ::= A<584>
<574> ::= C	<583> ::= U<584>
<575> ::= A<576>	<583> ::= G<584>
<575> ::= U<576>	<583> ::= C<584>
<575> ::= G<576>	<583> ::= <584>
<575> ::= C<576>	<584> ::= A<585>
<575> ::= <576>	<584> ::= U<585>
<576> ::= A<577>	<584> ::= G<585>
<576> ::= U<577>	<584> ::= C<585>

<584> ::= <585>	<593> ::= C<594>
<585> ::= A<586>	<593> ::= <594>
<585> ::= U<586>	<594> ::= A<595>
<585> ::= G<586>	<594> ::= U<595>
<585> ::= C<586>	<594> ::= G<595>
<585> ::= <586>	<594> ::= C<595>
<586> ::= A<587>	<594> ::= <595>
<586> ::= U<587>	<595> ::= A<596>
<586> ::= G<587>	<595> ::= U<596>
<586> ::= C<587>	<595> ::= G<596>
<586> ::= <587>	<595> ::= C<596>
<587> ::= A<588>	<595> ::= <596>
<587> ::= U<588>	<596> ::= A<597>
<587> ::= G<588>	<596> ::= U<597>
<587> ::= C<588>	<596> ::= G<597>
<587> ::= <588>	<596> ::= C<597>
<588> ::= A<589>	<596> ::= <597>
<588> ::= U<589>	<597> ::= A<598>
<588> ::= G<589>	<597> ::= U<598>
<588> ::= C<589>	<597> ::= G<598>
<588> ::= <589>	<597> ::= C<598>
<589> ::= A<590>	<597> ::= <598>
<589> ::= U<590>	<598> ::= A<599>
<589> ::= G<590>	<598> ::= U<599>
<589> ::= C<590>	<598> ::= G<599>
<589> ::= <590>	<598> ::= C<599>
<590> ::= A	<598> ::= <599>
<590> ::= U	<599> ::= A<600>
<590> ::= G	<599> ::= U<600>
<590> ::= C	<599> ::= G<600>
<593> ::= A<594>	<599> ::= C<600>
<593> ::= U<594>	<599> ::= <600>
<593> ::= G<594>	<600> ::= A<601>

<600> ::= U<601>	<606> ::= <607>
<600> ::= G<601>	<607> ::= A<608>
<600> ::= C<601>	<607> ::= U<608>
<600> ::= <601>	<607> ::= G<608>
<601> ::= A<602>	<607> ::= C<608>
<601> ::= U<602>	<607> ::= <608>
<601> ::= G<602>	<608> ::= A<609>
<601> ::= C<602>	<608> ::= U<609>
<601> ::= <602>	<608> ::= G<609>
<602> ::= A<603>	<608> ::= C<609>
<602> ::= U<603>	<608> ::= <609>
<602> ::= G<603>	<609> ::= A<610>
<602> ::= C<603>	<609> ::= U<610>
<602> ::= <603>	<609> ::= G<610>
<603> ::= A<604>	<609> ::= C<610>
<603> ::= U<604>	<609> ::= <610>
<603> ::= G<604>	<610> ::= A<611>
<603> ::= C<604>	<610> ::= U<611>
<603> ::= <604>	<610> ::= G<611>
<604> ::= A<605>	<610> ::= C<611>
<604> ::= U<605>	<610> ::= <611>
<604> ::= G<605>	<611> ::= A<612>
<604> ::= C<605>	<611> ::= U<612>
<604> ::= <605>	<611> ::= G<612>
<605> ::= A<606>	<611> ::= C<612>
<605> ::= U<606>	<611> ::= <612>
<605> ::= G<606>	<612> ::= A<613>
<605> ::= C<606>	<612> ::= U<613>
<605> ::= <606>	<612> ::= G<613>
<606> ::= A<607>	<612> ::= C<613>
<606> ::= U<607>	<612> ::= <613>
<606> ::= G<607>	<613> ::= A<614>
<606> ::= C<607>	<613> ::= U<614>

<613> ::= G<614>	<620> ::= A<621>
<613> ::= C<614>	<620> ::= U<621>
<613> ::= <614>	<620> ::= G<621>
<614> ::= A<615>	<620> ::= C<621>
<614> ::= U<615>	<620> ::= <621>
<614> ::= G<615>	<621> ::= A<622>
<614> ::= C<615>	<621> ::= U<622>
<614> ::= <615>	<621> ::= G<622>
<615> ::= A<616>	<621> ::= C<622>
<615> ::= U<616>	<621> ::= <622>
<615> ::= G<616>	<622> ::= A<623>
<615> ::= C<616>	<622> ::= U<623>
<615> ::= <616>	<622> ::= G<623>
<616> ::= A<617>	<622> ::= C<623>
<616> ::= U<617>	<622> ::= <623>
<616> ::= G<617>	<623> ::= A<624>
<616> ::= C<617>	<623> ::= U<624>
<616> ::= <617>	<623> ::= G<624>
<617> ::= A<618>	<623> ::= C<624>
<617> ::= U<618>	<623> ::= <624>
<617> ::= G<618>	<624> ::= A<625>
<617> ::= C<618>	<624> ::= U<625>
<617> ::= <618>	<624> ::= G<625>
<618> ::= A<619>	<624> ::= C<625>
<618> ::= U<619>	<624> ::= <625>
<618> ::= G<619>	<625> ::= A<626>
<618> ::= C<619>	<625> ::= U<626>
<618> ::= <619>	<625> ::= G<626>
<619> ::= A<620>	<625> ::= C<626>
<619> ::= U<620>	<625> ::= <626>
<619> ::= G<620>	<626> ::= A<627>
<619> ::= C<620>	<626> ::= U<627>
<619> ::= <620>	<626> ::= G<627>

<626> ::= C<627>	<633> ::= U<634>
<626> ::= <627>	<633> ::= G<634>
<627> ::= A<628>	<633> ::= C<634>
<627> ::= U<628>	<633> ::= <634>
<627> ::= G<628>	<634> ::= A<635>
<627> ::= C<628>	<634> ::= U<635>
<627> ::= <628>	<634> ::= G<635>
<628> ::= A<629>	<634> ::= C<635>
<628> ::= U<629>	<634> ::= <635>
<628> ::= G<629>	<635> ::= A<636>
<628> ::= C<629>	<635> ::= U<636>
<628> ::= <629>	<635> ::= G<636>
<629> ::= A<630>	<635> ::= C<636>
<629> ::= U<630>	<635> ::= <636>
<629> ::= G<630>	<636> ::= A<637>
<629> ::= C<630>	<636> ::= U<637>
<629> ::= <630>	<636> ::= G<637>
<630> ::= A<631>	<636> ::= C<637>
<630> ::= U<631>	<636> ::= <637>
<630> ::= G<631>	<637> ::= A<638>
<630> ::= C<631>	<637> ::= U<638>
<630> ::= <631>	<637> ::= G<638>
<631> ::= A<632>	<637> ::= C<638>
<631> ::= U<632>	<637> ::= <638>
<631> ::= G<632>	<638> ::= A<639>
<631> ::= C<632>	<638> ::= U<639>
<631> ::= <632>	<638> ::= G<639>
<632> ::= A<633>	<638> ::= C<639>
<632> ::= U<633>	<638> ::= <639>
<632> ::= G<633>	<639> ::= A<640>
<632> ::= C<633>	<639> ::= U<640>
<632> ::= <633>	<639> ::= G<640>
<633> ::= A<634>	<639> ::= C<640>

<639> ::= <640>	<646> ::= G<647>
<640> ::= A<641>	<646> ::= C<647>
<640> ::= U<641>	<646> ::= <647>
<640> ::= G<641>	<647> ::= A<648>
<640> ::= C<641>	<647> ::= U<648>
<640> ::= <641>	<647> ::= G<648>
<641> ::= A<642>	<647> ::= C<648>
<641> ::= U<642>	<647> ::= <648>
<641> ::= G<642>	<648> ::= A<649>
<641> ::= C<642>	<648> ::= U<649>
<641> ::= <642>	<648> ::= G<649>
<642> ::= A<643>	<648> ::= C<649>
<642> ::= U<643>	<648> ::= <649>
<642> ::= G<643>	<649> ::= A<650>
<642> ::= C<643>	<649> ::= U<650>
<642> ::= <643>	<649> ::= G<650>
<643> ::= A<644>	<649> ::= C<650>
<643> ::= U<644>	<649> ::= <650>
<643> ::= G<644>	<650> ::= A<651>
<643> ::= C<644>	<650> ::= U<651>
<643> ::= <644>	<650> ::= G<651>
<644> ::= A<645>	<650> ::= C<651>
<644> ::= U<645>	<650> ::= <651>
<644> ::= G<645>	<651> ::= A<652>
<644> ::= C<645>	<651> ::= U<652>
<644> ::= <645>	<651> ::= G<652>
<645> ::= A<646>	<651> ::= C<652>
<645> ::= U<646>	<651> ::= <652>
<645> ::= G<646>	<652> ::= A<653>
<645> ::= C<646>	<652> ::= U<653>
<645> ::= <646>	<652> ::= G<653>
<646> ::= A<647>	<652> ::= C<653>
<646> ::= U<647>	<652> ::= <653>

<653> ::= A<654>	<659> ::= C<660>
<653> ::= U<654>	<659> ::= <660>
<653> ::= G<654>	<660> ::= A<661>
<653> ::= C<654>	<660> ::= U<661>
<653> ::= <654>	<660> ::= G<661>
<654> ::= A<655>	<660> ::= C<661>
<654> ::= U<655>	<660> ::= <661>
<654> ::= G<655>	<661> ::= A<662>
<654> ::= C<655>	<661> ::= U<662>
<654> ::= <655>	<661> ::= G<662>
<655> ::= A<656>	<661> ::= C<662>
<655> ::= U<656>	<661> ::= <662>
<655> ::= G<656>	<662> ::= A<663>
<655> ::= C<656>	<662> ::= U<663>
<655> ::= <656>	<662> ::= G<663>
<656> ::= A<657>	<662> ::= C<663>
<656> ::= U<657>	<662> ::= <663>
<656> ::= G<657>	<663> ::= A<664>
<656> ::= C<657>	<663> ::= U<664>
<656> ::= <657>	<663> ::= G<664>
<657> ::= A<658>	<663> ::= C<664>
<657> ::= U<658>	<663> ::= <664>
<657> ::= G<658>	<664> ::= A<665>
<657> ::= C<658>	<664> ::= U<665>
<657> ::= <658>	<664> ::= G<665>
<658> ::= A<659>	<664> ::= C<665>
<658> ::= U<659>	<664> ::= <665>
<658> ::= G<659>	<665> ::= A
<658> ::= C<659>	<665> ::= U
<658> ::= <659>	<665> ::= G
<659> ::= A<660>	<665> ::= C
<659> ::= U<660>	<668> ::= A
<659> ::= G<660>	<668> ::= U

<668> ::= G	<678> ::= G<679>
<668> ::= C	<678> ::= C<679>
<671> ::= A<672>	<679> ::= A<680>
<671> ::= U<672>	<679> ::= U<680>
<671> ::= G<672>	<679> ::= G<680>
<671> ::= C<672>	<679> ::= C<680>
<671> ::= <672>	<680> ::= A<681>
<672> ::= A<673>	<680> ::= U<681>
<672> ::= U<673>	<680> ::= G<681>
<672> ::= G<673>	<680> ::= C<681>
<672> ::= C<673>	<680> ::= <681>
<673> ::= A<674>	<681> ::= A<682>
<673> ::= U<674>	<681> ::= U<682>
<673> ::= G<674>	<681> ::= G<682>
<673> ::= C<674>	<681> ::= C<682>
<674> ::= A<675>	<681> ::= <682>
<674> ::= U<675>	<682> ::= A<683>
<674> ::= G<675>	<682> ::= U<683>
<674> ::= C<675>	<682> ::= G<683>
<675> ::= A<676>	<682> ::= C<683>
<675> ::= U<676>	<682> ::= <683>
<675> ::= G<676>	<683> ::= A<684>
<675> ::= C<676>	<683> ::= U<684>
<676> ::= A<677>	<683> ::= G<684>
<676> ::= U<677>	<683> ::= C<684>
<676> ::= G<677>	<683> ::= <684>
<676> ::= C<677>	<684> ::= A<685>
<677> ::= A<678>	<684> ::= U<685>
<677> ::= U<678>	<684> ::= G<685>
<677> ::= G<678>	<684> ::= C<685>
<677> ::= C<678>	<684> ::= <685>
<678> ::= A<679>	<685> ::= A<686>
<678> ::= U<679>	<685> ::= U<686>

<685> ::= G<686>	<690> ::= U<691>G
<685> ::= C<686>	<691> ::= A<692>U
<685> ::= <686>	<691> ::= U<692>A
<686> ::= A<687>	<691> ::= G<692>C
<686> ::= U<687>	<691> ::= C<692>G
<686> ::= G<687>	<691> ::= G<692>U
<686> ::= C<687>	<691> ::= G<692>A
<686> ::= <687>	<691> ::= U<692>G
<687> ::= A<S688>	<692> ::= A<693>
<687> ::= U<S688>	<692> ::= U<693>
<687> ::= G<S688>	<692> ::= G<693>
<687> ::= C<S688>	<692> ::= C<693>
<687> ::= <S688>	<692> ::= <693>
<S688> ::= <688><770>	<693> ::= A<694>
<688> ::= A<689>U	<693> ::= U<694>
<688> ::= U<689>A	<693> ::= G<694>
<688> ::= G<689>C	<693> ::= C<694>
<688> ::= C<689>G	<693> ::= <694>
<688> ::= G<689>U	<694> ::= A<695>
<688> ::= U<689>G	<694> ::= U<695>
<689> ::= A<690>U	<694> ::= G<695>
<689> ::= U<690>A	<694> ::= C<695>
<689> ::= G<690>C	<694> ::= <695>
<689> ::= C<690>G	<695> ::= A<696>
<689> ::= G<690>U	<695> ::= U<696>
<689> ::= U<690>G	<695> ::= G<696>
<690> ::= A<691>A	<695> ::= C<696>
<690> ::= A<691>U	<695> ::= <696>
<690> ::= A<691>C	<696> ::= A<697>
<690> ::= U<691>A	<696> ::= U<697>
<690> ::= G<691>C	<696> ::= G<697>
<690> ::= C<691>G	<696> ::= C<697>
<690> ::= G<691>U	<696> ::= <697>

<697> ::= A<698>	<703> ::= <704>
<697> ::= U<698>	<704> ::= A<705>
<697> ::= G<698>	<704> ::= U<705>
<697> ::= C<698>	<704> ::= G<705>
<697> ::= <698>	<704> ::= C<705>
<698> ::= A<699>	<704> ::= <705>
<698> ::= U<699>	<705> ::= A<706>
<698> ::= G<699>	<705> ::= U<706>
<698> ::= C<699>	<705> ::= G<706>
<699> ::= A<700>	<705> ::= C<706>
<699> ::= U<700>	<705> ::= <706>
<699> ::= G<700>	<706> ::= A<707>
<699> ::= C<700>	<706> ::= U<707>
<699> ::= <700>	<706> ::= G<707>
<700> ::= A<701>	<706> ::= C<707>
<700> ::= U<701>	<706> ::= <707>
<700> ::= G<701>	<707> ::= A<708>
<700> ::= C<701>	<707> ::= U<708>
<700> ::= <701>	<707> ::= G<708>
<701> ::= A<702>	<707> ::= C<708>
<701> ::= U<702>	<707> ::= <708>
<701> ::= G<702>	<708> ::= A<709>
<701> ::= C<702>	<708> ::= U<709>
<701> ::= <702>	<708> ::= G<709>
<702> ::= A<703>	<708> ::= C<709>
<702> ::= U<703>	<708> ::= <709>
<702> ::= G<703>	<709> ::= A<710>
<702> ::= C<703>	<709> ::= U<710>
<702> ::= <703>	<709> ::= G<710>
<703> ::= A<704>	<709> ::= C<710>
<703> ::= U<704>	<709> ::= <710>
<703> ::= G<704>	<710> ::= A<711>
<703> ::= C<704>	<710> ::= U<711>

<710> ::= G<711>	<717> ::= U<718>A
<710> ::= C<711>	<717> ::= G<718>C
<711> ::= A<712>	<717> ::= C<718>G
<711> ::= U<712>	<717> ::= G<718>U
<711> ::= G<712>	<717> ::= U<718>G
<711> ::= C<712>	<718> ::= A<719>U
<712> ::= A<713>	<718> ::= U<719>A
<712> ::= U<713>	<718> ::= G<719>C
<712> ::= G<713>	<718> ::= C<719>G
<712> ::= C<713>	<718> ::= G<719>U
<712> ::= <713>	<718> ::= U<719>G
<713> ::= A<714>	<719> ::= A<720>
<713> ::= U<714>	<719> ::= U<720>
<713> ::= G<714>	<719> ::= G<720>
<713> ::= C<714>	<719> ::= C<720>
<713> ::= <714>	<719> ::= <720>
<714> ::= A<715>	<720> ::= A<721>
<714> ::= U<715>	<720> ::= U<721>
<714> ::= G<715>	<720> ::= G<721>
<714> ::= C<715>	<720> ::= C<721>
<714> ::= <715>	<720> ::= <721>
<715> ::= A<716>	<721> ::= A<722>
<715> ::= U<716>	<721> ::= U<722>
<715> ::= G<716>	<721> ::= G<722>
<715> ::= C<716>	<721> ::= C<722>
<715> ::= <716>	<721> ::= <722>
<716> ::= A<S717>	<722> ::= A<723>
<716> ::= U<S717>	<722> ::= U<723>
<716> ::= G<S717>	<722> ::= G<723>
<716> ::= C<S717>	<722> ::= C<723>
<716> ::= <S717>	<722> ::= <723>
<S717> ::= <717><750>	<723> ::= A<724>
<717> ::= A<718>U	<723> ::= U<724>

<723> ::= G<724>	<728> ::= <729>
<723> ::= C<724>	<729> ::= A<730>
<723> ::= <724>	<729> ::= U<730>
<724> ::= A<S725>	<729> ::= G<730>
<724> ::= U<S725>	<729> ::= C<730>
<724> ::= G<S725>	<730> ::= A<731>
<724> ::= C<S725>	<730> ::= U<731>
<724> ::= <S725>	<730> ::= G<731>
<S725> ::= <725>	<730> ::= C<731>
<S725> ::= <725><741>	<731> ::= A<732>
<725> ::= A<726>U	<731> ::= U<732>
<725> ::= A<726>G	<731> ::= G<732>
<725> ::= U<726>A	<731> ::= C<732>
<725> ::= G<726>C	<732> ::= A<733>
<725> ::= C<726>G	<732> ::= U<733>
<725> ::= G<726>U	<732> ::= G<733>
<725> ::= U<726>G	<732> ::= C<733>
<726> ::= A<727>U	<733> ::= A<734>
<726> ::= U<727>A	<733> ::= U<734>
<726> ::= G<727>C	<733> ::= G<734>
<726> ::= C<727>G	<733> ::= C<734>
<726> ::= G<727>U	<734> ::= A<735>
<726> ::= U<727>G	<734> ::= U<735>
<727> ::= A<728>U	<734> ::= G<735>
<727> ::= U<728>A	<734> ::= C<735>
<727> ::= G<728>C	<735> ::= A<736>
<727> ::= C<728>G	<735> ::= U<736>
<727> ::= G<728>U	<735> ::= G<736>
<727> ::= U<728>G	<735> ::= C<736>
<728> ::= A<729>	<736> ::= A<737>
<728> ::= U<729>	<736> ::= U<737>
<728> ::= G<729>	<736> ::= G<737>
<728> ::= C<729>	<736> ::= C<737>

<736> ::= A	<746> ::= A<747>
<736> ::= U	<746> ::= U<747>
<736> ::= G	<746> ::= G<747>
<736> ::= C	<746> ::= C<747>
<737> ::= A	<746> ::= <747>
<737> ::= U	<747> ::= A
<737> ::= G	<747> ::= U
<737> ::= C	<747> ::= G
<741> ::= A<742>	<747> ::= C
<741> ::= U<742>	<750> ::= A<751>
<741> ::= G<742>	<750> ::= U<751>
<741> ::= C<742>	<750> ::= G<751>
<741> ::= <742>	<750> ::= C<751>
<742> ::= A<743>	<750> ::= <751>
<742> ::= U<743>	<751> ::= A<752>
<742> ::= G<743>	<751> ::= U<752>
<742> ::= C<743>	<751> ::= G<752>
<742> ::= <743>	<751> ::= C<752>
<743> ::= A<744>	<751> ::= <752>
<743> ::= U<744>	<752> ::= A<753>
<743> ::= G<744>	<752> ::= U<753>
<743> ::= C<744>	<752> ::= G<753>
<743> ::= <744>	<752> ::= C<753>
<744> ::= A<745>	<752> ::= <753>
<744> ::= U<745>	<753> ::= A<754>
<744> ::= G<745>	<753> ::= U<754>
<744> ::= C<745>	<753> ::= G<754>
<744> ::= <745>	<753> ::= C<754>
<745> ::= A<746>	<753> ::= <754>
<745> ::= U<746>	<754> ::= A<755>
<745> ::= G<746>	<754> ::= U<755>
<745> ::= C<746>	<754> ::= G<755>
<745> ::= <746>	<754> ::= C<755>

<754> ::= <755>	<761> ::= A<762>
<755> ::= A<756>	<761> ::= U<762>
<755> ::= U<756>	<761> ::= G<762>
<755> ::= G<756>	<761> ::= C<762>
<755> ::= C<756>	<761> ::= <762>
<755> ::= <756>	<762> ::= A<763>
<756> ::= A<757>	<762> ::= U<763>
<756> ::= U<757>	<762> ::= G<763>
<756> ::= G<757>	<762> ::= C<763>
<756> ::= C<757>	<762> ::= <763>
<756> ::= <757>	<763> ::= A<764>
<757> ::= A<758>	<763> ::= U<764>
<757> ::= U<758>	<763> ::= G<764>
<757> ::= G<758>	<763> ::= C<764>
<757> ::= C<758>	<763> ::= <764>
<757> ::= <758>	<764> ::= A<765>
<758> ::= A<759>	<764> ::= U<765>
<758> ::= U<759>	<764> ::= G<765>
<758> ::= G<759>	<764> ::= C<765>
<758> ::= C<759>	<764> ::= <765>
<759> ::= A<760>	<765> ::= A
<759> ::= U<760>	<765> ::= U
<759> ::= G<760>	<765> ::= G
<759> ::= C<760>	<765> ::= C
<759> ::= A	<770> ::= A<771>
<759> ::= U	<770> ::= U<771>
<759> ::= G	<770> ::= G<771>
<759> ::= C	<770> ::= C<771>
<760> ::= A<761>	<771> ::= A<772>
<760> ::= U<761>	<771> ::= U<772>
<760> ::= G<761>	<771> ::= G<772>
<760> ::= C<761>	<771> ::= C<772>
<760> ::= <761>	<772> ::= A<773>

```

<772> ::= U<773>
<772> ::= G<773>
<772> ::= C<773>
<773> ::= A<774>
<773> ::= U<774>
<773> ::= G<774>
<773> ::= C<774>
<774> ::= A<775>
<774> ::= U<775>
<774> ::= G<775>
<774> ::= C<775>
<775> ::= A<776>
<775> ::= U<776>
<775> ::= G<776>
<775> ::= C<776>

```

```

<776> ::= A<777>
<776> ::= U<777>
<776> ::= G<777>
<776> ::= C<777>
<777> ::= A
<777> ::= U
<777> ::= G
<777> ::= C

\BeginFunction Ad1(X,y)
    \BeginInsertActions
        X ::= y
    \EndInsertActions
\EndFunction

```