REGINALDO INOJOSA DA SILVA FILHO

UMA NOVA FORMULAÇÃO ALGÉBRICA PARA O AUTÔMATO FINITO ADAPTATIVO DE SEGUNDA ORDEM APLICADA A UM MODELO DE INFERÊNCIA INDUTIVA

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Doutor em Ciências.

São Paulo
2011

REGINALDO INOJOSA DA SILVA FILHO

UMA NOVA FORMULAÇÃO ALGÉBRICA PARA O AUTÔMATO FINITO ADAPTATIVO DE SEGUNDA ORDEM APLICADA A UM MODELO DE INFERÊNCIA INDUTIVA

Tese apresentada à Escola Politécnica da Universidade de São Paulo para obtenção do Título de Doutor em Ciências.

Área de Concentração:

Sistemas Digitais

Orientador:

Prof. Dr. Ricardo Luis de Azevedo da Rocha

São Paulo
2011

Este exemplar foi revisado e alterado em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 02 de março de 2012.

Assinatura do autor _____

Assinatura do orientador _____

FICHA CATALOGRÁFICA

Silva Filho, Reginaldo Inojosa da
Uma nova formulação algébrica para o autômato finito adaptativo de segunda ordem aplicado a um modelo de inferência indutiva / R.I. da Silva Filho. -- ed. rev. -- São Paulo, 2012.
109 p.

Tese (Doutorado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1. Teoria dos autômatos 2. Aprendizado computacional 3. Linguagens formais I. Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II. t.

Para Patricia.

AGRADECIMENTOS

Agradeço a DEUS por me conceder a saúde, força e inspiração durante a realização do meu trabalho. Agradeço a todos que me ajudaram durante esse período:

- Meu orientador, Ricardo Luis de Azevedo da Rocha, pelo apoio e inestimável ajuda.
- Minha esposa, por estar sempre ao meu lado.
- A todos meus colegas, tanto os do laboratório de complexidade e sincronismo quanto os do Laboratório de linguagens e técnicas adaptativas, pelas muitas contribuições, pelo companheirismo e pelo auxílio nos momentos difíceis.
- Aos professores José Jaime da Cruz e Jorge Kinoshita.
- Ao meu filho Christian Inojosa, cujo sorriso me encoraja a enfrentar cada novo dia.

RESUMO

O objetivo deste trabalho é apresentar o modelo dos autômatos adaptativos de segunda ordem e mostrar a forte conexão desse modelo com o aprendizado indutivo no limite. Tal modelo é definido com a utilização de um conjunto de transformações sobre autômatos finitos não - determinísticos e a conexão com o aprendizado no limite é estabelecida usando o conceito de mutação composta, onde uma hipótese inicial dá inicio ao processo de aprendizagem, produzindo, após uma sequência de transformações sofridas por essa primeira hipótese, um modelo final que é o resultado correto do aprendizado. Será apresentada a prova de que um autômato adaptativos de segunda ordem, usado como um aprendiz, pode realizar o processo de aprendizado no limite.

O formalismo dos autômatos adaptativos de segunda ordem é desenvolvido sobre o modelo dos autômatos adaptativos de primeira ordem, uma extensão natural do modelo dos autômatos adaptativos clássicos. Embora tenha o mesmo poder computacional, o autômato adaptativo de primeira ordem apresenta uma notação mais simples e rigorosa que o seu antecessor, permitindo derivar novas propriedades. Uma dessas propriedades é justamente sua capacidade de aprendizado.

Como consequência, o modelo dos autômatos adaptativos de segunda ordem aumenta a expressividade computacional dos dispositivos adaptativos através da sua notação recursiva, e também através do seu potencial para o uso em aplicações de aprendizado de máquina, ilustrados nesta tese. Uma arquitetura de aprendizado de máquina usando os autômatos adaptativos de segunda ordem é proposto e um modelo de identificação no limite, aplicado em processos de inferência para linguagens livre de contexto, é apresentado.

ABSTRACT

The purpose of this work is to present the second-order adaptive automaton under an transformation automata approach and to show the strong connection of this model with learning in the limit. The connection is established using the adaptive mutations, in which any hypothesis can be used to start a learning process, and produces a correct final model following a step-by-step transformation of that hypothesis by a second-order adaptive automaton. Second-order adaptive automaton learner will be proved to acts as a learning in the limit.

The presented formalism is developed over the first-order adaptive automaton, a natural and unified extension of the classical adaptive automaton. First-order adaptive automaton is a new and better representation for the adaptive finite automaton and to also show that both formulations – the original and the newly created – have the same computational power. Afterwards both formulations show to be equivalent in representation and in computational power, but the new one has a highly simplified notation. The use of the new formulation actually allows simpler theorem proofs and generalizations, as can be verified in this work.

As results, the second-order adaptive automaton enhances the computational expressiveness of adaptive automaton through its recursive notation, and also its skills for the use in machine learning applications were illustrated here. An architecture of machine learning to use the adaptive technology is proposed and the model of identification in limit applied in inference processes for free-context languages.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SÍMBOLOS

Fin Ponteiro para o estado de chegada de uma transição, p. 16

$\mathscr{M}^0$ Classe de todos os AFND sob um alfabeto dado, p. 16

GLC Gramática Gramática livre de contexto, p. 17

SUMÁRIO

1 INTRODUÇÃO

A motivação deste trabalho partiu da observação de certos comportamentos presentes no aprendizado humano. Um indivíduo, quando frente a um problema, segue uma série de reaçoẽs estabelecidas por sua intuição, experiências e conhecimento [Damásio 1999]. Dentro desse contexto, é possível assumir que todo indivíduo possui uma representação interna do mundo que o cerca [Davis, Shrobe e Szolovits 1993]. Nesta representação, estão as "regras" que o indivíduo crê serem as responsáveis pelo funcionamento do mundo. Tal representação é dinâmica e os frutos dessa dinâmica são as mudanças sofridas pela representação ao longo do tempo, com o objetivo de adequá-la às regras que regem o ambiente onde o indivíduo se encontra.

Em neurociências, tais mudanças são associadas à plasticidade neural, que consiste justamente na capacidade de alteração adaptativa presente no sistema nervoso central, alteração essa decorrente de estímulos provenientes do ambiente externo. Considerando todo o sistema cérebro-espinhal como a estrutura que guarda a representação do mundo, admite-se que as mudanças mencionadas no parágrafo anterior buscam **adaptar** a representação do mundo às regras do ambiente por intermédio dos estímulos externos. Assim, a representação do mundo disponível ao indivíduo nunca será completa, mas sujeita a um processo de construção ao longo da vida desse indivíduo, restando ao mesmo "cobrir" a realidade que o cerca usando os recursos representacionais que possui.

Tal situação é ilustrada na figura 1. Nesse diagrama, a estrutura $\mathfrak{M}$ representa o conhecimento do indivíduo. Observa-se também uma linha diagonal que representa as alterações sofridas por $\mathfrak{M}$ em função dos estímulos externos, representados por $(e_1, e_2, \ldots e_n)$ e ocorridos em diferentes momentos no tempo. As setas $f^i_{\mathfrak{M}_i}$ representam a resposta do indivíduo frente a um estímulo e_i. Os estímulos provocam mudanças na representação interna $\mathfrak{M}_i$ do indivíduo. A estrutura $\mathfrak{M}$ guarda a representação $\mathfrak{R}$ das regras $\mathfrak{R} \subseteq \{R_l\}_{l=1}^{L}$ do ambiente. De forma figurada, se fosse possível "congelar" o tempo, antes de cada instância de $\mathfrak{M}$ sofrer as alterações geradas pelos estímulos, observaria-se as diferentes reações que indivíduo apresentaria nos diferentes momentos da sua história [Rocha 2011]. Assim, as linhas verticais indicam o comportamento de uma mesma instância de $\mathfrak{M}$ sujeita a diferentes estímulos, enquanto que as linhas horizontais indicam os comportamentos das diferentes instâncias de $\mathfrak{M}$ sujeitas a um mesmo tipo de estímulo.

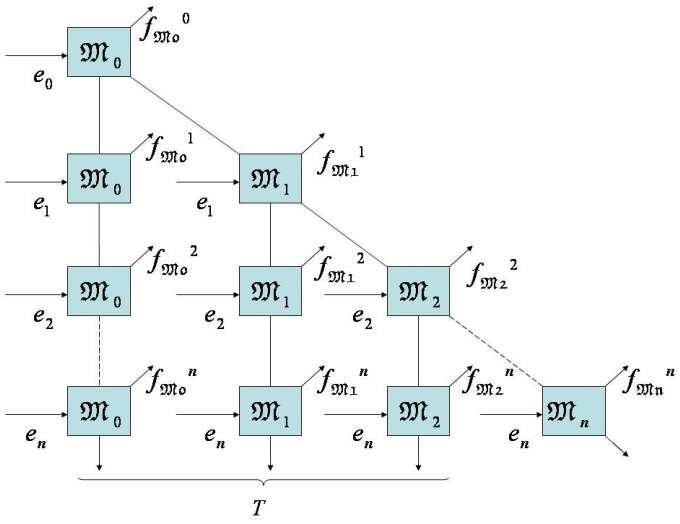

Figura 1: Comportamento de uma representação dinâmica do mundo.

Em consequência dessa dinâmica, o indivíduo infere e constrói, ao longo de sua vida, o conjunto de regras $\{R_l\}_{l=1}^{L}$ que governam o seu ambiente. Assim, dependendo do momento ao qual o indivíduo foi exposto a um determinado problema, sua reação pode ser diferente da reação que teria tomado em relação a

qualquer outro momento, tanto no passado quanto no futuro [Rocha 2011]. Tal fato leva à segunda característica relevante incorporada neste trabalho: o processo de construção da representação de mundo do indivíduo só termina com o fim da sua existência física.

Essa foi a exposição das idéias que inspiraram o presente trabalho. A partir desse quadro geral, serão estabelecidos as metas a serem alcançadas.

1.1 Organização

Este trabalho apresenta os seguintes capítulos:

(a) A presente **Introdução**, que mostra as motivações do trabalho e apresenta a tese.

(b) **Autômato Adaptativo de Primeira Ordem**. Descreve os autômatos adaptativos, sua formulação e comportamento, bem como as questões relacionadas com a evolução téorica do modelo, culminando na apresentação do modelo dos autômatos adaptativos de primeira ordem.

(c) O capítulo sobre **Inferência Indutiva e Aprendizado no Limite** apresenta o conceito de inferência da qual o aprendizado de linguagens formais no limite faz parte. Esse modelo possui particularidades que o relacionam com as motivações do trabalho, o que determinou sua escolha como paradigma de aprendizagem adotado neste trabalho.

(d) **Autômato Adaptativo de Segunda Ordem**. Nesse capítulo é discutida a necessidade da adaptatividade de segunda ordem, bem como o seu significado e aplicação no processo de aprendizagem.

(e) No capítulo **Resultados** se encontra descrito a relação entre os autômatos adaptativos de segunda ordem e o aprendizado no limite, sua aplicação para

aprendizagem de subclasses de linguagens livres de contexto, bem como um exemplo ilustrativo.

(f) Por último, o capítulo **Referências**, listando todas as obras que contribuiram na elaboração e que foram utilizadas neste trabalho.

1.2 Objetivos

Diferentemente do ponto de vista conexionista[Russell e Norvig 2003], não serão utilizados modelos inspirados nas estruturas que compõem o cérebro e não serão abordadas questões relacionadas à formação ou natureza da memória. Os modelos computacionais, nesta tese, não lembram os elementos anatômicos e neurofisiológicos relacionados com a cognição humana. Mesmo as características advindas da observação do "comportamento humano" são justificadas tão somente pelo senso comum. Não é objetivo da tese validar qualquer teoria pedagógica, psicanalítica ou biológica. Também não será explicado como o conhecimento teve seu início a partir do surgimento do indivíduo. Como hipótese, presume-se a existência de um conhecimento "inato" para, a partir dessa hipótese, prosseguir na análise de como a estrutura que representa tal conhecimento se comporta diante das ações descritas ao longo deste trabalho.

Para o estabelecimento de um escopo factível, o conceito de **conhecimento** será representado explicitamente pelas linguagens artificiais pertencentes à hierarquia de Chomsky [Lewis e Papadimitriou 1997]. Logo, o conceito de representação de conhecimento será sinônimo de linguagens formais. Tal escolha serve para delimitar uma fronteira de estudo razoável, pois tais linguagens podem ser tratadas tanto por humanos quanto por máquinas [Hopcroft, Motwani e Ullman 2000] e são objeto de ampla pesquisa a décadas [Lewis e Papadimitriou 1997, Hopcroft, Motwani e Ullman 2000].

1.2.1 Postulados

Partindo da apresentação das motivações apresentadas, são três os postulados assumidos como base deste trabalho:

- O aprendizado é um processo de adaptação frente a novos estímulos do ambiente.
- O aprendizado está baseada em um processo de inferência indutiva.
- A adaptação e a indução estão profundamente interligadas.

Com o fim de estabelecer o escopo deste trabalho, cabe esclarecer em qual contexto os termos "indutivo" e "adaptativo" serão utilizados. Todo processo dito, aqui, indutivo refere-se à **inferência indutiva**. Ray Solomonoff foi o pai da teoria geral da inferência indutiva [Solomonoff 2007], uma área frutífera de estudo que gerou muitos resultados em inteligência articial [Li e Vitányi 2008, Wallace 2004]. O objetivo da inferência indutiva é identificar um objeto desconhecido pela escolha de um elemento pertencente a um conjunto (geralmente infinito) de hipóteses para esse objeto [Chater e Vitányi 2007]. A hipótese é uma representação finita do objeto e deve ser consistente com a dados, apresentados de forma incremental na forma de segmentos crescentes de exemplos acerca do comportamento do objeto. Entretanto, como apontado por Wallace e Dowe, hoje, os rumos tomados pela linha de pesquisa aberta por Solomonoff "são, na verdade, mais voltados para a previsão do que para a indução"[Wallace 2004, Dowe 2011, Solomonoff 1996].

Por essa razão, a abordagem utilizada aqui é baseada no trabalho criado por Emil Mark Gold [Gold 1967], chamada de **aprendizado no limite**, ou também **identificação no limite**. Existem várias maneiras diferentes de escolher as hipóteses em um processo de inferência indutiva e cada uma, na prática, determina um modelo de aprendizagem diferente. As principais abordagens de

escolha são a probabilística [Li e Vitányi 2008, Wallace 2004] e a estratégia por enumeração [Li e Vitányi 1995]. A aprendizagem no limite utiliza a segunda estratégia. Responsável por essa ramificação dentro da área de estudo da inferência indutiva, Gold estudou o problema da aprendizagem de funções recursivas e linguagens formais. Em seu modelo, a inferência indutiva é um processo infinito, onde dispositivos chamados de **aprendizes** (do inglês ***learners***) identificam uma linguagem formal caso a geração de hipóteses convirja para uma única hipótese e nenhuma outra alteração ocorra, mesmo que novos exemplos da línguagem continuem a ser apresentados aos aprendizes, indefinidamente.

Já o termo adaptatividade é tratado dentro do escopo dos modelos computacionais auto-modificáveis [Rubinstein e Shutt 1994, Shutt 1995]. Os modelos computacionais auto-modificáveis são formados por todos os modelos computacionais cujas estruturas internas sofrem variações em função das etapas computacionais realizadas. Em particular, um dispositivo adaptativo é composto por um componente não-adaptativo, representado por um conjunto de regras, e um componente adaptativo, composto por um conjunto de regras que têm como objetivo modificar as regras do componente não-adaptativo. Também faz parte dessa arquitetura, uma unidade de entrada (ou leitura) que recebe os *inputs* que o dispositivo adaptativo processa. A modificação nas regras não-adaptativas, induzida pelas regras adaptativas, ocorre em função unicamente dos *inputs* enviados a essa unidade de entrada [Neto 2002].

Um **autômato adaptativo** consiste em um dispositivo adaptativo compostos por uma máquina de estados representada por um grafo orientado inicial. Tal máquina de estados é o componente não adaptativo em questão e é chamado de **dispositivo subjacente**. Atrelada a algumas das transições dessa máquina de estados, estão as regras que - quando da execução dessas transições - modificam esse grafo inicial, retirando transições presentes na máquina de es-

tados ou inserindo novas transições e estados na máquina de estados. Tais regras consistem no componente adaptativo do modelo e são chamadas de **ações adaptativas**. A operação do autômato adaptativo dá-se através de sucessivas transformações nesse grafo pelo processamento dos *inputs* lidos na entrada do autômato adaptativo que levam à execução das transições especiais, chamadas de t**ransições adaptativas**. Assim, se um determinado *input* levar à execução de uma transição adaptativa, as regras de transformação modificarão a topologia do autômato. Dessa forma, o próximo *input* será processado por uma máquina de estados completamente diferente [Rocha 2011].

1.2.2 A tese

A questão agora diz respeito à relação entre a adaptatividade e a inferência indutiva. Parte do trabalho de provar essa conexão conceitual implica em criar um formalismo que funcione como um *framework* que ajude a descrever tal ligação conceitual. Para autômatos que se modificam e aprendizes que mudam suas hipóteses, parece razoável considerar que exista tal formalismo unificador. A escolha de tal *framework* recaiu no uso de um conjunto de operações não-númericas sobre autômatos, que permitam transformações entre eles. Cabe salientar aqui que o autômato adaptativo usado será o **autômato finito adaptativo**.

Porém, para descrever, nesse novo *framework*, a relação entre os autômatos adaptativos e o aprendizado no limite, será necessário criar uma extensão para o modelo dos autômatos adaptativos que permita descrevê-los em termos desse novo formalismo. Logo, a formulação dos autômatos adaptativos usado neste trabalho não consiste na formulação original, mas em uma nova, que guarda semelhanças e diferenças com a formulação apresentada em [Neto 2002], sem fugir, porém, dos norteamentos que deram origem à formulação original. Tais mudanças permitem tratar algumas questões levantadas em [Neto e Bravo 2003], que já propunha al-

gumas mudanças na formulação original. Tais questões serão discutidas ao longo deste trabalho.

Se a relação entre o aprendizado no limite e adaptatividade é de fato tão forte, cabe então a pergunta: descrever a inferência indutiva em termos da adaptatividade implica em alguma nova propriedade ou novo comportamento para a primeira? Na introdução, foi dito que a origem deste trabalho estava na observação de certos comportamentos do aprendizado humano. As linguagens formais foram escolhidas para servir como representação do conhecimento, representação essa que pode ser processada tanto por humanos como por máquinas. Logo, se o inter-relacionamento entre o formalismo adaptativo e o aprendizado no limite existe e espelha alguma característica singular do aprendizado humano (particularmente no que diz respeito às linguagens), um modelo formal que englobe os dois conceitos, adaptatividade e aprendizado no limite, pode trazer novas características para algum aspecto do processo de aprendizado das linguagens formais.

Na área de aprendizado de máquina, a inferência de linguagens livres de contexto usando apenas exemplos positivos de sentenças é um problema muito explorado e ainda em aberto [Laxminarayana e Nagaraja 2003, Higuera 2010, Prajapati, Chaudhari e Chandwani 2008]. O problema da inferência de linguagens livres de contexto consiste em encontrar uma representação finita (gramática ou reconhecedor) para uma linguagem livre de contexto dada apenas uma sequência potencialmente infinita de palavras dessa linguagem - que são exatamente os seus exemplos positivos [Lee 1996]. Uma das razões para que esse problema seja tão importante consiste no fato das linguagens livres de contexto serem mais expressivas que as linguagens regulares [Lewis e Papadimitriou 1997], o que traz uma série de implicações. Do ponto de vista da linguistica, as linguagens livres de contexto são melhores aproximações para as linguagens naturais do que as linguagens regulares. Toda linguagem de programação utiliza os con-

ceitos das linguagens livres de contexto [Aho, Sethi e Ullman 1986]. Elas ainda surgem em uma série de estruturas diferentes, que vão dos documentos web à organização das proteinas. Entretanto, o aprendizado da classe de linguagens livres de contexto usando apenas exemplos positivos é impossível e tal resultado foi apresentado no artigo que deu origem ao aprendizado no limite [Gold 1967]. Porém os resultados alcançados por Angluin [Angluin 1980], provando que subclasses de linguagens regulares podem ser aprendidas apenas com exemplos positivos, gerou uma busca por resultados semelhantes para linguagens livres de contexto [Higuera 2010]. Segundo [Prajapati 2011], "a aprendizagem da classe das linguagens livres de contexto é intratável por qualquer modelo de aprendizagem. Apesar desse resultado negativo, existem resultados positivos para o aprendizado de subclasses de linguagens livres de contexto de forma eficiente". Para exercitar o modelo de inferência adaptativa deste trabalho, ele será utilizado para encontrar um resultado positivo para o aprendizado de uma subclasse de linguagem livres de contexto de forma eficiente. Como requisito, tal resultado, bem como a descrição da subclasse, têm que ser derivados de forma natural a partir do *framework* de aprendizado que será desenvolvido aqui.

Portanto, A tese defendida nesse trabalho é a seguinte:

Existe uma hierarquia de reconhecedores formais adaptativos onde os membros pertencentes à segunda ordem podem ser usados para aprender, no limite, subclasses de linguagens livres de contexto usando apenas exemplos positivos. Tais reconhecedores são uma extensão do modelo original dos autômatos adaptativos e são definidos em função de uma série de transformações de autômatos baseada em duas operações básicas: adição e subtração de transições internas do autômato.

Assim, para verificar as aplicações do modelo proposto nesta tese, um exemplo

de aprendizado no limite para linguagens livre de contexto usando somente a formalização definida será apresentado ao final deste trabalho.

1.2.3 Passos intermediários necessários

Dessa forma, para representar o aprendizado via inferência indutiva em termos da adaptatividade de segunda ordem, os seguintes objetivos devem ser alcançados:

- Demonstrar a existência de uma hierarquia de reconhecedores adaptativos.
- Demonstrar que os membros de segunda ordem dessa hierarquia podem aprender um subconjunto de linguagens livre de contexto, usando apenas exemplos positivos.
- Demonstrar que tal aprendizado é caracterízado como uma inferência indutiva, mais particularmente como um aprendizado no limite (que será apresentado nas seções seguintes).

São, portanto, os resultados necessários para se provar a tese.

1. Definir um **conjunto de transformações** para os autômatos finitos não-determinísticos, baseados em duas operações básicas: uma de inserção e outra de remoção para as transições de estados dos autômatos.
2. Defininir a classe dos autômatos adaptativos a partir das transformações de autômatos finitos. Tal extensão do modelo original é chamada de **autômatos adaptativos de primeira ordem**.
3. Definir um conjunto de transformações os autômatos adaptativos de primeira ordem. Como consequência, será possível definir **a classe dos**

autômatos adaptativos de segunda ordem seguindo o mesmo método usado no item anterior.

4. Definir a subclasse de linguagens livres de contexto que pode ser identificada no limite utilizando-se os autômatos adaptativos de segunda ordem apenas com exemplos positivos da linguagem.

1.3 Preliminares

Nesta seção será apresentado um sumário dos conceitos centrais que servem de base para o desenvolvimento da tese, muitos dos quais oriundos da teoria dos autômatos e da teoria dos conjuntos. As definições que envolvem tais teorias, apresentadas a seguir, se baseiam em [Hopcroft, Motwani e Ullman 2000], [Lewis e Papadimitriou 1997] e [Sipser 1996] que são textos clássicos sobre a teoria da computação e sobre linguagens formais. Neste trabalho, também foi utilizado [Ramos, Neto e Vega 2009]. As notações introduzidas aqui serão utilizadas em todos os demais capítulos.

Definição 1.1 (**Função de cardinalidade**). *Dado um conjunto Y (finito ou infinito) a função de cardinalidade $card(Y)$, aplicada ao conjunto Y , calcula o número de elementos de Y. Caso o conjunto Y seja infinito, o resultado da função será $card(Y) = \infty$.*

Definição 1.2 (**Classe**). *Uma **classe** é uma coleção de conjuntos (finitos ou infinitos) ou de sistemas formais que representem tais conjuntos. Uma classe pode ter um número finito de membros ou pode ter infinitos membros.*

Definição 1.3 (**Conjunto finito indexado**). *Um **conjunto finito indexado** é um conjunto para o qual existe uma função $i : M \to I$. onde $0 \leq M \leq m$ para todo $m \in \mathbb{N}$. Denota-se cada $i(m)$ por i_m, de modo a representar o conjunto finito indexado pela expressão:*

$$I = \{i_0, i_1, \ldots, i_m\} \tag{1.1}$$

Definição 1.4 (**Conjunto infinito indexado**). *Um* ***conjunto infinito indexado*** *é todo conjunto genérico para o qual existe uma função* $j : \mathbb{N} \to I$. *Da mesma forma que foi vista em 1.3, é possível representar cada* $j(n)$ *(com* $n \in \mathbb{N}$*) por* $(j_n)_{n\in\mathbb{N}}$, *de modo a representar o conjunto infinito indexado pela expressão:*

$$J = \{j_0, j_1, \ldots\} \tag{1.2}$$

Definição 1.5 (**Família**). *Uma* ***família*** *é uma classe (finita ou infinita) indexada.*

Definição 1.6 (**Função de remoção**). *Dado um conjunto indexado A qualquer, a* ***função de remoção*** *é definida como:*

$$\boldsymbol{rem}(A, x) = A - \{x\} \tag{1.3}$$

onde $x \in A$

Definição 1.7 (**Função de inserção**). *Dado um conjunto indexado A qualquer, a* ***função de inserção*** *é definida como:*

$$\boldsymbol{ins}\,(A, x) = \{a_0, a_1, \ldots, a_{m+1}\} \tag{1.4}$$

onde $a_{m+1} = x$

Definição 1.8 (**Alfabeto**). *Um* ***alfabeto*** *é um conjunto finito de símbolos. Neste trabalho, os elementos de um alfabeto serão sempre representados pelas letras minúsculas do alfabeto grego e qualquer alfabeto é representado pela letra maiúscula grega* Σ.

Definição 1.9 (**Cadeia de caracteres**). *Uma* ***cadeia de caracteres*** *é uma sequência justaposta finita, composta por símbolos de um alfabeto. Neste trabalho, as cadeias de caracteres são sempre representadas pelas últimas letras do alfabeto latino.*

Para efeitos de simplificação, será usado somente o termo cadeia para se referir às cadeias de caracteres.

Definição 1.10 (**Comprimento de uma cadeia**). *O* ***comprimeiro de uma cadeia*** *é o número natural que indica quantos símbolos essa cadeia possui. O comprimento dessa cadeia é indicado por $|w|$, para uma cadeia w qualquer.*

Cada símbolo de um alfabeto é considerado uma **cadeia de tamanho unitário**.

Definição 1.11 (**Cadeia vazia**). *A* ***cadeia vazia*** *é uma cadeia cujo comprimento é nulo. Neste trabalho, a cadeia vazia é representada pela letra grega ε.*

Definição 1.12 (**Operação de concatenação**). *Dadas duas cadeias quaisquer u e v, a* ***operação de concatenação*** *$u.v$ gera uma terceira cadeia, a cadeia $w = uv$, formada pela justaposição ordenada de todos os símbolos da cadeia u e da cadeia v.*

A expressão Σ^n, com $n \in \mathbb{N}$, denota todas as possíveis cadeias de tamanho n geradas através de operações de concatenação sobre um alfabeto Σ.

Definição 1.13 (**Fecho de Kleene**). *O* ***fecho de Kleene*** *é o fechamento reflexivo e transitivo sobre um alfabeto Σ. O fecho de Kleene é definido pela expressão*

$$\Sigma^* = \bigcup_{i=0}^{\infty} \Sigma^i \tag{1.5}$$

Definição 1.14 (**Linguagem**). *Uma **linguagem**, para um dado alfabeto* Σ*, é definida como:*

$$L \subseteq \Sigma^* \tag{1.6}$$

Definição 1.15 (**Autômato finito não-determinístico**). *Um **autômato finito não-determinístico**, (abreviado por AFND) para um dado alfabeto* Σ*, é definido pela quíntupla:*

$$M^0 = (Q, \Sigma, \partial, q_0, E) \tag{1.7}$$

onde:

- Q= $\{q_1, \ldots, q_n\}$ *é* ***o conjunto finito dos estados*** *do AFND* M^0.
- $q_0 \in Q$ *é o* ***estado inicial*** *do AFND* M^0.
- $E \subseteq Q$ *é o* ***conjunto dos estados de aceitação*** *do AFND* M^0.
- Σ *é o* ***alfabeto*** *do AFND.*
- $\partial \subseteq Q \times \{\Sigma \cup \{\varepsilon\}\} \times Q$ *é a* ***relação finita de transição de estados*** *do AFND* M^0.

A relação de transição de estados de um autômato M^0 estabelece o conjunto $\partial = \{\delta_1, \delta_2, \ldots, \delta_i\}$ de transições do AFND M^0, onde cada transição assume a forma δ=(q',$\alpha, q'')$ com $\{q', q''\} \subseteq Q$ e $\alpha \in \Sigma$.

Definição 1.16 (**Linguagem regular**). *Uma linguagem é denominada de uma linguagem regular se algum AFND a reconhece.*

Definição 1.17 (**Estrutura hierárquica escalar**). *Uma* ***estrutura hierárquica escalar****[Salthe e Matsuno 1995] é indicada pela notação* $\langle a_n \langle a_{n-1} \ldots \langle a_1 \langle a_0 \rangle \rangle \rangle \rangle$. *Tal notação apresenta a seguinte semântica: Se* a_{i+1} *é um sistema formal definido por uma ênupla, então* a_i *é um membro dessa enupla, para todo* $0 \leq i \leq (n-1)$. *Para isso, será utilizada a notação que utiliza os colchetes aninhados* $\langle . \langle . \rangle \rangle$.

A seguir, serão apresentadas definições de funções que têm como argumento um AFND. O objetivo de tais funções é a obtenção de algum tipo de informação sobre o autômato. Tais funções serão utilizadas neste trabalho como operações auxiliares.

Definição 1.18 (**Operação de busca interna em AFND**). *Dado um AFND* M^0, *existem sete* ***operações de busca interna*** , *denominadas* ***Sch***. *Com um AFND* M^0 *como parâmetro, tais funções tem por objetivo retornar a(s) transição(ões) no autômato que possua(m) determinados elementos que também são estipulados nos parâmetros da função.*

$$\boldsymbol{Sch}(q_i, a, q_j, M^0) = \{(q_i, a, q_j) : (q_i, a, q_j) \in \partial\} \tag{1.8}$$

$$\boldsymbol{Sch}(q_i, q_j, M^0) = \{(q_i, \alpha, q_j) : (q_i, \alpha, q_j) \in \partial\} \tag{1.9}$$

$$\boldsymbol{Sch}(q_i, M^0) = \{(q_i, \alpha, q'') : (q_i, \alpha, q'') \in \partial\} \tag{1.10}$$

$$\boldsymbol{Sch}(q_j, M^0) = \{(q', \alpha, q_j) : (q', \alpha, q_j) \in \partial\} \tag{1.11}$$

$$\boldsymbol{Sch}(q_i, a, M^0) = \{(q_i, a, q'') : q_i, a, q'') \in \partial\} \tag{1.12}$$

$$\boldsymbol{Sch}(q_j, a, M^0) = \{(q', a, q_j) : (q', a, q_j) \in \partial\} \tag{1.13}$$

$$\boldsymbol{Sch}(a, M^0) = \{(q', a, q'') : (q', a, q'') \in \partial\} \tag{1.14}$$

Definição 1.19 (**Operação de identificação de não-pertinência**). *Dado um AFND* M^0, *a* ***operação de identificação de não-pertinência****, representada*

por ***Pert*** *é utilizada para verificar se uma determinada transição não pertence a um autômato.*

$$\boldsymbol{Pert}(\delta, M^0) = \begin{cases} \delta & \Leftrightarrow \ \delta \notin \partial \\ \emptyset & \Leftrightarrow \ \delta \in \partial \end{cases} \tag{1.15}$$

Definição 1.20 (**Ponteiro para o estado de partida**). *Um* ***ponteiro para o estado de partida****, representado por* ***Start*** *é uma função que, dada uma transição (conforme a definição 1.15), retorna o primeiro estado da transição.*

$$\boldsymbol{Start}(\delta) = q' \tag{1.16}$$

Definição 1.21 (**Ponteiro para o estado de chegada de uma transição**). *Um* ***ponteiro para o estado de chegada****, representado por* ***Fin****, é uma função que, dada uma transição (conforme a definição 1.15), retorna o último estado da mesma.*

$$\boldsymbol{Fin}(\delta) = q'' \tag{1.17}$$

Definição 1.22 (**Classe $\mathscr{M}^0$**). *Dado um alfabeto Σ, a classe de todos os AFND possíveis de serem construídos a partir desse alfabeto é indicado pela expressão M^0.*

As próximas definições tratam das **gramáticas livres de contexto**. Antes, cabe uma introdução breve sobre o conceito de gramática formal.

Enquanto os AFNDs reconhecem cadeias de símbolos (isto é, eles processam as cadeias, símbolo por símbolo, e determinam se tais cadeias pertencem ou não a alguma linguagem), as gramáticas são formalismos ditos**geradores**, pois

elas "produzem" cadeias de uma determinada linguagem. As gramáticas, assim como os AFNDs são representações finitas de linguagens potêncialmente infinitas e o motivo da introdução de tal formalismo é a necessidade da definição das linguagens livres de contexto, pois tal classe de linguagens é definida usando-se o conceito de gramáticas formais. Como as gramáticas definem **regras** para a produção de cadeias, as linguagens livres de contexto são definidas por certos tipos de gramáticas com regras de produção bem específicas, chamadas de **gramáticas livres de contexto**.

Definição 1.23 (**Gramática formal**). *Dado um alfabeto* Σ*, uma* ***gramática formal*** *GLC consiste na quádrupla abaixo.*

$$G = (\Sigma, V, P, S) \tag{1.18}$$

onde:

- V *é o conjunto finito de* ***símbolos não-terminais****. O conjunto* V *e* Σ *são disjuntos. Todos os elementos do conjunto* V *são denotados por letras maiúsculas do alfabeto latino.*
- P *é o conjunto finito de* ***produções da gramática****, definido como:*

$$P \subset V \times (\Sigma \cup V)^* \tag{1.19}$$

- $S \in V$ *é o* ***símbolo não-terminal inicial****.*

Se u, v e w são cadeias constituídas de símbolos do alfabeto, pertencentes ao conjunto Σ, e símbolos não-terminais, pertencentes ao conjunto V, e $Z \rightarrow w$ é uma produção da gramática livre de contexto, diz-se que uZv **origina** uwv, escrito $uZv \Rightarrow_{GLC} uwv$ [Sipser 1996]. Usando a definição de originação, diz-se que u

deriva v, escrito $u \Rightarrow^*_{GLC} v$, se $u = v$ ou se existe uma sequência $u_1, u_2, \ldots, u_k$, com $k \geq 0$, para o qual [Sipser 1996]:

$$u \Rightarrow u_1 \Rightarrow u_2 \Rightarrow \ldots \Rightarrow u_k \Rightarrow v$$

Definição 1.24 (**Linguagem livre de contexto**). *Dada uma gramática livre de contexto GLC, a* ***linguagem livre de contexto*** *em função dessa gramática é definida como:*

$$L(GLC) = \{t \in \Sigma^* : S \Rightarrow^*_{GLC} t\} \tag{1.20}$$

O interesse nas linguagens livres de contexto não decorre apenas do fato da dificuldade do seu aprendizado utilizando apenas exemplos positivos. Como foi mencionado na subseção 1.2.2, as linguagens livres de contexto são utilizadas na formalização sintática das linguagens de programação de alto nível (tais como a linguagem C++ e a linguagem Java).

As linguagens de programação não são inteiramente livres de contexto, existem elementos nessas linguagens que são, sim, dependentes de contexto. Porém, a capacidade de representação de construções aninhadas por parte das linguagens de programação fazem parte das linguagens livres de contexto. O aninhamento de construções é a característica que distingue as linguagens livres de contexto das linguagens regulares, onde esse tipo de construção é impossível.

São vários os elementos sintáticos das linguagens de programação modelados através do uso de construções aninhadas, todos de suma importância no projeto de um compilador, que é o programa responsável por converter o arquivo do código-fonte em um arquivo executável. Em função disso, várias notações baseadas em GLCs foram desenvolvidas para facilitar a especificação das linguagens de programação. Uma das primeiras e mais importante é

a BNF (abreviatura do inglês *Backus-Naur Form*) [Aho, Sethi e Ullman 1986, Ramos, Neto e Vega 2009], usada ainda hoje em projeto de compiladores.

Exemplos de construções aninhadas ocorrem em linguagens de programação na construção de expressões aritméticas, em que subexpressões são delimitadas através do uso de parênteses. Na estruturação do fluxo de controle (através do uso de comandos de decisão ou de repetição, por exemplo), as expressões aninhadas ocorrem na utilização de comandos internos, inseridos como parte integrante de outros, esses externos, como, por exemplo, a utilização de comandos de repetição *while* dentro de comandos de repetição *while* em um programa escrito em linguagem C. Outros exemplos incluem o uso de funções e métodos empregados para criar diferentes escopos [Ramos, Neto e Vega 2009].

Tais características contribuíram para a escolha das linguagens livres de contexto como objeto de estudo para o aprendizado mediante inferência indutiva e sua relação com a adaptatividade, apresentado a seguir. Porém, para descrever a relação existente entre inferência indutiva e autômatos adaptativos, devemos primeiro descrever os dois. Os capítulos seguintes apresentam os respectivos modelos, bem como a relação entre eles.

2 AUTÔMATO ADAPTATIVO DE PRIMEIRA ORDEM

O termo adaptatividade tem um sentido bem definido dentro da área de estudo dos formalismos automodificáveis [Boden et al. 2004, Carmi 2002, Klein e Kutrib 2002, Boullier 1994, Rubinstein e Shutt 1994]. Nos autômatos adaptativos, a adaptatividade se manifesta como a capacidade de alteração (sem a interferência de qualquer agente externo) da própria estrutura do autômato em resposta às entradas apresentadas [Neto 2001]. Em outras palavras, toda e qualquer mudança possível deve ser contemplada pelo próprio dispositivo em reação aos estímulos externos.

2.1 Autômatos adaptativos, formulação original

Nesta seção será apresentada a formulação original dos autômatos adaptativos, tal como apresentada em [Neto 2001], [Neto 2002] e [Neto e Bravo 2003]. Embora traga o conceito de adaptatividade para os modelos computacionais, o modelo atual para os autômatos adaptativos apresenta deficiências. Tais deficiências dizem respeito tanto à capacidade de expressividade da formulação quanto a problemas conceituais na mesma. tais problemas não envolvem o conceito de adaptatividade em si, mas a maneira como ele é apresentado, que resulta em ambiguidades e imprecisões. Para resolver tais questões, foi criado o modelo dos autômatos adaptativos de primeira e de segunda ordem.

Os autômatos finitos, sejam eles determinísticos ou não, são modelos computacionais muito importantes na descrição de comportamentos dinâmicos e processos de uma maneira geral. Em qualquer momento, um autômato possui uma configuração que determina seu estado interno e,*associada aos símbolos provenientes da cadeia de entrada*, seus comportamentos futuros. Ele opera mudando sucessivamente de uma configuração para outra em resposta aos símbolos da cadeia que são consumidos como entrada. Assim, partindo de uma configuração inicial, o autômato finito consome toda a cadeia de entrada apresentada a ele e, após processá-la, termina em uma determinada configuração que determina se aquela cadeia foi aceita ou não [Hopcroft, Motwani e Ullman 2000].

O modelo original dos **autômatos finitos adaptativos**, apresenta um autômato finito composto de um conjunto Q de estados, um alfabeto Σ, uma *relação de transição* ∂, um estado inicial $q_0 \in Q$ e um conjunto $E \subseteq Q$ de estados finais. A função de transição mapeia o estado e o símbolo corrente em um novo estado. Em cada passo de execução do autômato, o estado corrente e o símbolo consumido determinam um conjunto de transições possíveis de serem executadas para que o próximo estado seja alcançado. Nos autômatos finitos determinísticos, esse conjunto contém um e somente um elemento. Já nos autômatos finitos não determinísticos, o conjunto de transições possíveis pode possuir mais de um elemento, ou seja, pode existir mais de uma possibilidade, ou nenhuma, nas opções possíveis de mudança de configuração.

Existem dois tipos de transições possíveis de um estado A para um estado B:

- transição (A, α, B), que consome o símbolo de entrada α.
- transição (A, ϵ, B), que muda o estado do autômato sem consumir qualquer símbolo (entrada vazia).

O comportamento adaptativo é obtido pelas **ações adaptativas** responsáveis

pela mudança no comportamento do autômato finito, seja ele determinístico ou não. Tal mudança é obtida pela alteração da relação de transição. Esta modificação é realizada pela inserção ou remoção das transições do autômato finito. O mecanismo adaptativo é definido pela associação de um par de ações adaptativas, $\mathcal{B}$ (do inglês *before*, ou *a priori*) e $\mathcal{A}$ (do inglês *after*, ou a *posteriori*) a uma transição do autômato finito - que, a partir de agora, poderá ser também referenciado como **dispositivo subjacente**. Tal par de ações é executado, uma ação antes (*before*) da realização da transição do autômato finito e a outra após (*after*) a transição. Essa transição, com seu par de ações adaptativas associadas, recebe o nome de transição adaptativa. A notação original para representar uma transição adaptativa com ações é apresentada abaixo:

$$(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A} \tag{2.1}$$

onde:

- $\{q, q'\} \subseteq Q$
- $\alpha \in \Sigma \cup \{\epsilon\}$
- $\mathcal{A}$ representa a ação adaptativa a *posteriori*.
- $\mathcal{B}$ representa a ação adaptativa a *priori*.

As funções adaptativas que descrevem as ações adaptativas $\mathcal{A}$ e $\mathcal{B}$ são declaradas à parte do autômato finito. A declaração de uma função adaptativa é composta pelo cabeçalho da função (onde constam o nome da função e seus parâmetros formais) e o corpo da função (onde se encontram os nomes das variáveis, geradores, bem como as ações adaptativas básicas). A notação para a função adaptativa é apresentada abaixo.

$\eta(\Omega)$ {

declaração de nomes (opcional)

declaração das ações adaptativas básicas

}

onde: $\Omega = \{\varphi_1, \ldots, \varphi_n\}$ é a lista ordenada de n-parâmetros, composto de elementos φ_i (para $1 \leq i \leq n$) todos eles passados por valor para a função η. Esses argumentos podem assumir qualquer papel coerente (*estados, símbolos, funções adaptativas,variáveis, etc.*) dentro do corpo da função. O corpo da função é delimitado pelos colchetes e formado pelas declarações dos nomes das variáveis e geradores, bem como por uma lista de ações adaptativas elementares, responsáveis pelas transformações a serem realizadas no autômato finito. As variáveis e os geradores são utilizados no escopo dessas ações. Particularmente, os geradores são utilizados sempre que existir a necessidade de incluir um *estado que não faça parte do conjunto de estados atual do dispositivo subjacente.* As variáveis, por sua vez, são associadas às ações básicas de inspeção. A ação de inspeção, bem como as demais ações adaptativas básicas são definidas abaixo:

1. $?[(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A}]$: a **ação adaptativa básica de inspeção** busca no conjunto atual de transições do autômato as transições que correspondem ao padrão definido na ação. Isso implica que *qualquer* elemento da transição pode ser uma variável dessa busca, inclusive as ações adaptativas. Assim, por exemplo, pode-se buscar "todas as transições que consomem o símbolo a de uma cadeia", ou "todas as transições do autômato que tem o estado q como o estado inicial da transição".

2. $-[(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A}]$: a **ação adaptativa básica de eliminação** retira do conjunto atual de transições do autômato todas as transições que correspodem à forma de transição definida na ação.

3. $+[(q, \alpha) : \mathcal{B} \rightarrow q' : \mathcal{A}]$: a **ação adaptativa básica de inserção** insere no conjunto atual de transições do autômato a transição que correspode à forma de transição definida na ação.

Assim, os autômatos adaptativos têm sua operação fundamentada na evolução estrutural *pré-definida (programada nas ações adaptativas) e governada pela própria sequência de transições executadas a cada símbolo da cadeia de entrada* de reconhecedores hierarquicamente estruturados, que efetuam, portanto, o reconhecimento da linguagem que representam como se percorressem uma trajetória no espaço de máquinas de estados, evoluindo de um ponto para outro neste espaço a cada alteração estrutural ocorrida [Rocha 2011].

Ao final dessa seção, algumas considerações se fazem necessárias. Em [Neto e Bravo 2003] os autores defendem a necessidade de realizar mudanças no modelo para simplificar e estabelecer uma "formulação mais efetiva" dos autômatos adaptativos. Nesse sentido, eles levantam dois pontos a serem tratados:

1. limitar, em uma transição adaptativa, o número de ações adaptativas. De modo que cada transição possua apenas ações *a priori* ou *a posteriori*.

2. Restringir o número de parâmetros permitidos em cada função adaptativa.

Sobre o segundo item, os autores comentam:

"(...)is it really needed to allow an arbitrary number of parameters? Should it be better to limit the number of parameters to a minimum? What should this minimum be?"

A essas questões, somam-se outras três que não constam em [Neto e Bravo 2003] e que foram levantadas em [Filho e Rocha 2011]:

- A notação em estilo de "pseudocódigo" da especificação das funções adaptativas. Embora sejam interessantes para alguém envolvido na implementação desse modelo em alguma linguagem de programação, tal abordagem deixa claro que o conceito de "função" usado nesse caso está associado ao conceito de "procedimento" ou "sub-rotina", que trazem uma série de questões (tipagem de dados, sintaxe) que não são esclarecidos no modelo.

- A formulação dos autômatos adaptativos possui conceitos ambíguos. Não é possível saber qual a natureza das variáveis e dos geradores. Não é possível saber em qual ordem os estados e os símbolos são passados como parâmetros para o corpo de uma função adaptativa.

- A notação gráfica para a representação do processamento de uma cadeia por parte do autômato, embora intuitiva e visual, é pouco prática e pouco geral para ser utilizada em outra situação que não seja a apresentação de exemplos bem simples de tais processamentos.

Para resolver esses problemas, a solução encontrada foi propor uma nova formulação para o modelo dos autômatos adaptativos usando um formalismo que descrevesse qual é a natureza das ações de auto-modificação realizadas pelos autômatos adaptativos. Tal formalismo é apresentado na seção seguinte.

2.2 Transformações adaptativas

A reformulação dos autômatos adaptativos começa pela escolha de um *framework* formal para sua descrição. Entretanto, a tarefa de reescrever os autômatos adaptativos vai além de resolver os problemas mencionados no final da seção anterior. Tal tarefa também deve ser vista como uma iniciativa para buscar compreender mais detalhadamente as propriedades referentes à capacidade de auto-modificação dos automatos adaptativos. Era sa-

bido que, durante sua execução, as alterações nos dispositivos subjacentes dos autômatos adaptativos poderiam ser vistos como uma trajetória no espaço dos autômatos finitos [Rocha 2007]. Portanto, um dos tópicos deste trabalho foi justamente propor a definição de um conjunto de transformações não-numéricas [Denecke e Wismath 2002][Mikolajczak 1991] sobre o conjunto dos autômatos finitos não-determinísticos. Esse conjunto é composto de dois operadores que mimetizam o comportamento das ações adaptativas básicas de inserção e remoção de transições no dispositivo subjacente.

Usando esses operadores, define-se uma variante da formulação dos autômatos adaptativos (a formulação ou modelo original), chamada de **autômatos adaptativos de primeira ordem**. A escolha desse nome tem uma razão: de posse da definição dos autômatos adaptativos de primeira ordem, foi possível aplicar a mesma idéia e definir um segundo conjunto de operadores, agora sobre o conjunto dos autômatos adaptativos de primeira ordem. Assim, um M^2 é um autômato adaptativo com as operações de inserção e remoção de transições adaptativas de primeira ordem, ou seja, um **autômato adaptativo de segunda ordem**. Neste tipo de autômato, os dispositivos subjacentes são os autômatos adaptativos de primeira ordem.

Definição 2.1 (**Transição própria**). *Para cada AFND M^0 da classe $\mathscr{M}^0$, definido em 1.22, sobre um alfabeto Σ, uma transição própria é definida como:*

$$\delta_{pro} = \delta = (q', \alpha, q'') : \delta \in \partial \tag{2.2}$$

Onde ∂ é a relação de transição de estados do AFND M^0.

Definição 2.2 (**Transição estrangeira**). *Para a mesma classe $\mathscr{M}^0$ da definição 2.1, uma transição estrangeira de M^0 é definida como:*

$$\delta_{for} = \delta = (q', \alpha, q'') : \delta \notin \partial (2.3)$$

Onde ∂ é a relação de transição de estados do AFND M^0.

Definição 2.3 (**Sequência positiva**). *Uma* ***sequência positiva*** *é uma sequência definida como:*

$$\lambda_{pro} = (\delta_{pro_1}, \ldots, \delta_{pro_m}) \tag{2.4}$$

composta de transições próprias de um dado M^0.

Definição 2.4 (**Sequência negativa**). *Uma* ***sequência negativa*** *é uma sequência definida como:*

$$\lambda_{for} = (\delta_{for_1}, \ldots, \delta_{for_n}) \tag{2.5}$$

composta de transições estrangeiras para um dado M^0.

Definição 2.5 (**Par de transformações de primeira ordem**). *Dada uma sequência negativa λ_{for} e uma sequência positiva λ_{pro}, ambas para o mesmo AFND M^0, a sequência*

$$\phi = (\lambda_{for}, \lambda_{pro}) \tag{2.6}$$

é chamada de ***par de transformações de primeira ordem****.*

2.2.1 Operadores Sintáticos

Utilizando os conceitos de transição própria, transição estrangeira, bem como as definições de função de remoção e função de inserção, é possível definir opera-

dores que alteram as relações de transição de membros da classe $\mathscr{M}^0$, bem como seu conjunto de estados. Existem dois operadores, f^- e f^+, que, juntos, são denominados de **operadores sintáticos**, pois alteram a linguagem reconhecida pelo AFND M^0 modificado.

Definição 2.6 (**δ-remoção**). *Dado um membro M^0 da classe $\mathscr{M}^0$ e utilizando-se a função apresentada na equação 1.3, para qualquer transição própria de M^0, a δ-remoção é definida como*

$$f^- M^0 = f^-(\delta_{pro}, M^0) = (Q, q_0, E, \Sigma, \boldsymbol{rem}(\partial, \delta_{pro})) \tag{2.7}$$

Definição 2.7 (**δ-inserção**). *Dado um membro M^0 da família $\mathscr{M}^0$ e utilizando-se a função apresentada na equação 1.4, para qualquer transição estrangeira de M^0, a δ-inserção é definida como:*

$$f^+ M^0 = f^+(\delta_{for}, M^0) = (\boldsymbol{ins}(\boldsymbol{ins}(Q, q'), q''), q_0, E, \Sigma, \boldsymbol{ins}(\partial, \delta_{for})) \tag{2.8}$$

2.2.2 Transformações Sintáticas

As operações de inserção e remoção (definições 2.6 e 2.7) providenciam a base para a construção de transformações complexas sobre os AFNDs da classe $\mathscr{M}^0$. Essas transformações são definidas pela combição dos operadores de inserção ou de remoção, usando sequências positivas e negativas como parâmetros. Tais transformações são denominadas de **transformações sintáticas** e serão definidas a seguir.

Definição 2.8 (**Transformação expansiva de primeira ordem**). *Dada uma sequência negativa λ_{for} referente a um AFND M^0, a **transformação expansiva de primeira ordem** é definida como:*

$$F^{+}_{\lambda_{for}} M^0 \triangleq F^{+}(\lambda_{for}, M^0) = (f^{+}_{n} \circ f^{+}_{n-1} \circ \ldots \circ f^{+}_{2} \circ f^{+}_{1}) M^0 \tag{2.9}$$

Definição 2.9 (**Transformação contrativa de primeira ordem**). *Dada uma sequência positiva λ_{pro} referente a um AFND M^0, a* ***transformação contrativa de primeira ordem*** *é definida como:*

$$F^{-}_{\lambda_{pro}} M^0 \triangleq F^{-}(\lambda_{pro}, M^0) = (f^{-}_{m} \circ f^{-}_{m-1} \circ \ldots \circ f^{-}_{2} \circ f^{-}_{1}) M^0 \tag{2.10}$$

O termo expansivo e o termo contrativo são utilizados para nomear as transformações por causa do efeito que causam na relação de transição ∂ dos AFNDs, hora diminuindo o número de transições pela remoção das mesmas (transformação contrativa), hora aumentando o número pela inserção de transições não contempladas inicialmente pelo autômato (transformação expansiva).

De posse das definições de mutação expansiva e contrativa, é possível definir um operador que realiza, simultâneamente, esses dois tipos de mutação sobre um mesmo AFND M^0, usando um par de transformações de primeira ordem como argumento.

Definição 2.10 (**Mutação composta de primeira ordem**). *Dado um par de transformações de primeira ordem ϕ para um autômato M^0, uma mutação composta de primeira ordem é definida como:*

$$\mathbb{F}_{\phi} M^0 \triangleq \mathbb{F}(\phi, M^0) = F^{-}_{\lambda_{pro}} F^{+}_{\lambda_{for}} M^0 \tag{2.11}$$

Existem casos especiais que podem ocorrer no cálculo das mutações compostas de primeira ordem que envolvem os pares de transformações de primeira ordem presentes nessas mutações e dizem respeito às formas que as sequências positivas e negativas desses pares podem assumir. Tais situações são descritas abaixo:

- Situação 1: a sequência positiva λ_{pro} do par de transformações de primeira ordem ϕ é uma sequência vazia.
- Situação 2: a sequência negativa λ_{for} do par de transformações de primeira ordem ϕ é uma sequência vazia.
- Situação 3: Tanto a a sequência positiva λ_{pro} quanto a a sequência negativa λ_{for} do par de transformações de primeira ordem ϕ são sequências vazias.

A situação três é um caso extremo. Nela, as duas sequências não possuem qualquer elemento. Devido à sua singularidade, esse caso leva a definição de um par de transformações com características peculiares.

Definição 2.11 (**Par vazio de primeira ordem**). *O par de transformações de primeira ordem onde tanto a sequência positiva λ_{pro} quanto a sequência negativa λ_{for} são sequências vazias é denominado de par vazio de primeira ordem, sendo representado pela notação $\phi^{\emptyset}$.*

A ocorrência de sequências vazias em um par de par de transformações de primeira ordem altera o comportamento da mutação composta de primeira ordem de diferentes maneiras; cada uma delas é descrita a seguir.

- $\mathbb{F}_{\phi} M^0 = F^{+}_{\lambda_{for}} M^0$; para a situação 1.
- $\mathbb{F}_{\phi} M^0 = F^{-}_{\lambda_{pro}} M^0$; para a situação 2.
- $\mathbb{F}_{\phi^{\emptyset}} M^0 = M^0$; para a situação 3.

Assim, é possível verificar que as diferentes ocorrências de sequências vazias nos pares de transformação de primeira ordem reduzem a mutação composta a cada uma das transformações sintáticas, tanto as contrativas como as expansivas. No caso extremo, utilizando o par vazio de primeira ordem (onde tanto a sequência

positiva quanto a negativa são vazias), a mutação composta não afeta o autômato M^0, ou seja, a aplicação da mutação composta retorna o próprio AFND M^0.

2.2.3 Autômato adaptativo de primeira ordem

Agora, de posse dos conceitos conceitos de pares de transformação, pares vazios, mutação composta e utilizando a classe $\mathscr{M}^0$ dos autômatos finitos não-determísticos, é possível definir o autômato adaptativo de primeira ordem.

Definição 2.12 (**Autômato adaptativo de primeira ordem**). *Um autômato adaptativo de primeira ordem (abreviado por AAPO) é uma quádrupla*

$$M^1 = (M^0, \Phi, \phi_0, \partial^1) \tag{2.12}$$

Onde:

- $M^0 \in \mathscr{M}^0$ *é chamado de* ***dispositivo subjacente de primeira ordem****.*
- Φ *é um conjunto de pares de transformação de primeira ordem, chamado de* ***conjunto de comportamentos de primeira ordem****.*
- *O elemento* $\phi_0 \in \Phi$ *é um par vazio de primeira ordem, chamado de* ***comportamento nulo****. Todo AAPO possui um comportamento nulo em seu conjunto de comportamentos de primeira ordem.*
- *Set* ∂^1 *é a* ***relação de transição adaptativa de primeira ordem****. Cada elemento de* ∂^1 *assume a forma:*

$$\delta^1_{i,k} = (\delta_i, \langle M^1 \langle \mathbb{F}_{\phi_k} M^0 \rangle \rangle) \tag{2.13}$$

sendo chamado de ***transição adaptativa de primeira ordem*** *onde* ϕ_k

pertence ao conjunto de comportamentos Φ *e* $\delta_i \in \partial$ *é uma transição pertencente à relação* ∂ *de transição do dispositivo subjacente.*

À primeira vista, a definição do autômato adaptativo de primeira ordem parece diferente daquela apresentada em [Neto 2002]. Embora diferentes na forma, as duas formulações são equivalentes e esse fato será provado formalmente. Entretanto, antes, é preciso mostrar como os elementos pertinentes a teoria clássica dos autômatos se apresentam no modelo dos AAPOs. Qualquer extensão no conceito original de autômato implica em uma nova forma de expressão para os elementos que formam a teoria. Assim, os elementos tradicionais da teoria dos autômatos devem ser tratados dentro do novo modelo. Esses elementos são listados abaixo:

- Configuração
- Relação de mudança de configuração
- Fechamento reflexivo e transitivo da relação de mudança de configuração
- Linguagem reconhecida pelo autômato

Como todo reconhecedor, o AAPO realiza sua tarefa aceitando todas as cadeias de uma determinada linguagem e rejeitando as demais. Tal reconhecimento se faz processando cada cadeia. O processamento se dá consumindo cada símbolo que compõe a cadeia de entrada, começando pelo símbolo mais à esquerda e dando prosseguimento em direção aos símbolos à direita, até que todos os símbolos sejam consumidos e a cadeia processada, estando o AAPO em um estado final. Já a rejeição ocorre quando o próximo símbolo a ser processado leva a uma situação onde não existe uma transição viável no autômato ou quando toda a cadeia é processada, mas o autômato não termina tal processamento em um estado final.Os detalhes desse processo serão mostrados nas definições a seguir.

Definição 2.13 (**Configuração**). *A **configuração** de um AAPO é a dupla*

$$(q', c) \in Q \times \Sigma^* \tag{2.14}$$

onde Q é o conjunto de estados do dispositivo subjacente do AAPO e c é a cadeia de entrada em alguma etapa do seu processamento.

Definição 2.14 (**Configuração inicial**). *A **Configuração inicial** de um AAPO é representado pela dupla*

$$(q_0, t) \tag{2.15}$$

onde q_0 é o estado inicial do dispositivo subjacente do AAPO e t é a cadeia de entrada a ser analisada, ou seja, nenhum simbolo de t foi consumido pelo AAPO.

Definição 2.15 (**Relação de mudança de configuração**). *A **Relação de mudança de configuração** descreve como o AAPO muda de uma configuração para outra*

$$(q', t) \vdash_{[\mathbb{F}_{\phi_k} M^0]} (q'', w) \Leftrightarrow \exists(\alpha \in \Sigma \ ou \ \alpha = \epsilon) : \alpha w = t$$

Onde q'' é um estado de $\mathbb{F}_{\phi_k} M^0$ e $((q', \alpha, q''), \langle M^1 \langle \mathbb{F}_{\phi_k}(M^0) \rangle \rangle) \in \partial^1$ para $\phi_k \in \Phi$.

Definição 2.16 (**Fechamento transitivo e reflexivo da relação de mudança de configuração**). *O **fechamento transitivo e reflexivo da relação de mudança de configuração** de um AAPO é definido como segue:*

$$(q', t) \vdash^*_{[\mathbb{F}_{\phi_{k_j}} \ldots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0]} (q'', w) \tag{2.16}$$

se e somente se:

- $(q' = q'')$ *e* $(w = t)$
- *ou caso as regras 1, 2 e 3, definidas abaixo, são satisfeitas:*

1. $t = a_0 a_1 \dots a_j w$ *com* $a_i \in \Sigma$ *para* $0 \leq i \leq j$
2. $\exists(\phi_{k_1}, \phi_{k_2}, \dots \phi_{k_{j+1}})$ *com* $\phi_{k_i} \in \Phi$ *para* $1 \leq i \leq j$
3. $\exists\ p_1, p_2 \dots p_j \in Q$ *onde* Q *é o conjunto de estados do dispositivo subjacente do AAPO a cada mutação, de forma que,*
   para $j \in \mathbb{N}$*:*

$$\begin{aligned}(q', t) \ &\vdash_{[\mathbb{F}_{\phi_{k_1}} M^0]} (p_1, a_1 a_2 a_3 \dots a_j w) \\ &\vdash_{[\mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0]} (p_2, a_2 a_3 \dots a_j w) \vdash_{[\mathbb{F}_{\phi_{k_3}} \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0]} \dots \\ &\vdash_{[\mathbb{F}_{\phi_{k_j}} \dots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0]} (p_j, a_j w) \\ &\vdash_{[\mathbb{F}_{\phi_{k_{j+1}}} \mathbb{F}_{\phi_{k_j}} \dots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0]} (q'', w)\end{aligned}$$

Definição 2.17 (**Linguagem reconhecida por um AAPO**). *A* ***linguagem definida por um AAPO*** *é definida como*

$$L(M^1) = \{t : (q_0, t) \vdash^*_{[\mathbb{F}_{\phi_{k_j}} \dots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0]} (q_f, \varepsilon)\} \tag{2.17}$$

Na situação especial em que o conjunto de comportamentos possui apenas o comportamento nulo, ou seja, $\Phi = \{\phi_0\}$, as condições necessárias para que uma cadeia seja aceita pelo AAPO passsam a ser:

$$(q_0, t) \vdash^*_{[\mathbb{F}_{\phi^\emptyset} \dots \mathbb{F}_{\phi^\emptyset} \mathbb{F}_{\phi^\emptyset} M^0]} (q_f, \varepsilon) = (q_0, t) \vdash^*_{M^0} (q_f, \varepsilon)$$

Em tal situação, o reconhecimento da cadeia t é realizado pelo dispositivo subjacente do AAPO. Assim, o AAPO age como um autômato finito não-determinístico, reconhecendo, portanto, apenas linguagens regulares, dentro da hierarquia de Chomsky.

2.3 Equivalência entre os modelos

O modelo dos AAPO foi criado para resolver os problemas apresentados em [Neto e Bravo 2003] e [Filho e Rocha 2011], acerca dos autômatos adaptativos. Para isso, foi usada uma abordagem diferente da original apresentada em [Neto 2002], com o intuito de tornar mais rigosas as definições que se referem aos comportamentos de auto-modificação e de adaptatividade. A consequência disso foi o surgimento de elementos que não estão presentes no modelo original, tais como as operações sintáticas, as transformações sintáticas, a mutação composta de primeira ordem e o conjunto de comportamentos de primeira ordem. Isso poderia se tornar um problema, caso a expressividade dos modelos fosse diferente. Nesta seção, a equivalência dos dois modelos será provada.

Em uma primeira análise, é possível observar a inexistência de ações adaptativas *a priori* no novo modelo. Porém, a própria formulação original resolve essa questão. A proposição 1 em [Rocha e Neto 2005] mostra que qualquer transição adaptativa, na formulação original, que use ações adaptativas *a priori* e *a posteriori* pode ser substituída por uma transição adaptativa equivalente somente *a priori* ou somente *a posteriori*. Assim, cada transição possuirá apenas uma ação adaptativa. O lema abaixo apresenta esse resultado.

Lema 2.1. *Os autômatos adaptativos podem ser modificados de forma a possuirem* (a) *apenas transições adaptativas com ações adaptativas a priori com nenhuma ação adaptativa a posteriori ou* (b) *transições adaptativas apenas com ações adaptativas a posteriori e com nenhuma ação adaptativa a priori.*

O lema 2.1, em conjunção com os lemas 2.2 e 2.3, mostra que qualquer transição de um autômato adaptativo construído usando-se a modelagem original pode ser mapeada para a nova representação.

Lema 2.2. *A seguinte equivalência existe para qualquer ação adaptativa elemen-*

tar da formulação original que modifique um AFND.

- *ação adaptativa básica de inspeção* $\longrightarrow$ *operação de busca interna em AFND*
- *ação adaptativa básica de eliminação* $\longrightarrow$ δ*-remoção*
- *ação adaptativa básica de inserção* $\longrightarrow$ δ*-inserção*

Prova 2.1 (do Lema 2.2). *A prova é obtida diretamente das definições.*

Equivalencia da ação básica de inspeção com a operação de busca interna em AFND: *As ações de inspeção buscam, na relação de transição do AFND, pela transição correspondente ao padrão dado. As variáveis do padrão de transição dado para a busca, na ação básica de inspeção, podem ser representadas pelos parâmetros presentes na definição de uma das sete operações de busca interna nos AFNDs (definição 1.18). Assim, existem sete possibilidades de representação para um padrão de transição da formulação original com os parâmetros das operações de busca interna. Para cada uma das possibilidades, existe uma operação de busca interna que, por definição, realiza exatamente a mesma busca.*

- $?[(v_k, v_l) \rightarrow v_m : \mathcal{A}]$ *é equivalente a* $\boldsymbol{Sch}(q_i, a, q_j, M^0)$
- $?[(v_k, \alpha) \rightarrow v_m : \mathcal{A}]$ *é equivalente a* $\boldsymbol{Sch}(q_i, q_j, M^0)$
- $?[(v_k, \alpha) \rightarrow q' : \mathcal{A}]$ *é equivalente a* $\boldsymbol{Sch}(q_i, M^0)$
- $?[(q, \alpha) : \mathcal{B} \rightarrow v_m : \mathcal{A}]$ *é equivalente a* $\boldsymbol{Sch}(q_j, M^0)$
- $?[(v_k, v_l) \rightarrow q' : \mathcal{A}]$*é equivalente a* $\boldsymbol{Sch}(q_i, a, M^0)$
- $?[(q, v_l) : \mathcal{B} \rightarrow v_m : \mathcal{A}]$ *é equivalente a* $\boldsymbol{Sch}(q_j, a, M^0)$
- $?[(q, v_l) \rightarrow q' : \mathcal{A}]$ *é equivalente a* $\boldsymbol{Sch}(a, M^0)$

onde:

- *A variável v_k é mapeada para o parâmetro q_i*
- *A variável v_l é mapeada para o parâmetro a*
- *A variável v_m é mapeada para o parâmetro q_j*

***Equivalência entre a ação básica de eliminação e a operação de δ-remoção**: prova imediata a partir das definições.*

***Equivalência entre a ação básica de inserção e a operação de δ-inserção**: Se a ação básica de inserção introduz, no conjunto de transições do AFND, uma transição sem ações adaptativas associadas, então a equivalência é imediata, pelas definições. Por outro lado, se a transição adicionada é uma transição adaptativa, então mais de uma δ-remoção é necessária para se obter a equivalência procurada. Assim, a ação de inserção $+[(q, \alpha) \rightarrow q' : \mathcal{A}]$ pode ser substituida por duas δ-inserções com um efeito de modificação sobre o AFND equivalente, realizado agora em duas etapas na nova formulação.*

Sem perda de generalidade, considere que toda ação adaptativa, na formulação original, possa ser definida a priori*, então, no novo modelo, elas podem ser definidas como transições adaptativas de primeira ordem com símbolo vazio e "desconectadas" do restante do autômato, isto é, não existe qualquer transição a partir do estado inicial que atinja tais transições.*

Qualquer ação adaptativa $\mathcal{A}_i$ definida usando a formulação original pode ser representada por uma mutação composta de primeira ordem $\mathbb{F}_{\phi_i} M^0$ sobre o dispositivo subjacente (a prova disto está no lema 2.3).

Para obter o mesmo comportamento de incluir uma transição com uma ação adaptativa $\mathcal{A}_i$, deve-se ter, na nova formulação, uma transição adaptativa de primeira ordem com o formato $((q_j, \epsilon, q_k), \langle M^1 \langle \mathbb{F}_{\phi_i} M^0 \rangle \rangle)$ presente no conjunto de transições adaptativas de primeira ordem, porém "desconectada" das demais

transições, ou seja: não deve existir transição possível para qualquer estado de $((q_j, \epsilon, q_k), \langle M^1\langle \mathbb{F}_{\phi_i} M^0\rangle\rangle)$.

O objetivo dessa transição é tornar a ação adaptativa $\mathcal{A}_i$*, agora na forma da mutação composta de primeira ordem* $\mathbb{F}_{\phi_i} M^0$*, presente a priori no AAPO. Com a transição contendo a mutação composta presente no conjunto de transições do AAPO, o procedimento equivalente, na nova formulação, à inserção de uma transição que possua uma ação adaptativa é realizado da seguinte forma:*

- *Primeiro passo: inserir a transição* (q, α, q_i).
- *Segundo passo: inserir a transição* (q_i, ϵ, q_j).

Assim, a ordem de execução da nova transição e da ação adaptativa a posteriori (representada pela mutação composta de primeira ordem) é preservada no novo modelo. O efeito final é equivalente à inserção de uma transição adaptativa na formulação original. □

Lema 2.3. *Qualquer ação adaptativa* $\mathcal{A}_i$*, definida usando a formulação original, pode ser representada por uma mutação composta de primeira ordem* $\mathbb{F}_{\phi_i} M^0$.

Prova 2.2 (do lema 2.3). *Usando os lemas 2.1 e 2.2 é possível realizar essa prova. Pelo lema 2.1 é possível transformar qualquer autômato na formulação original em outro equivalente onde todas as funções adaptativas presentes são funções adaptativas a priori.*

Toda ação adaptativa $\mathcal{A}_i$ *é composta por uma lista de ações adaptativas básicas, por definição. Assim, uma ação adaptativa* $\mathcal{A}_i$ *pode ser descrita por três conjuntos de ações adaptativas básicas (ações elementares de inspeção, ações elementares de remoção e ações elementares de inserção). Usando o lemma 2.2, todas as ações básicas podem ser mapeadas em operações de busca interna, δ-remoções ou δ-inserções do novo formalismo. Dessa forma, existe uma mutação*

composta de primeira ordem $\mathbb{F}_{\phi_i}M^0$ que, usando o lemma 2.2, mapeia as mesmas tarefas executadas pela ação adaptativa original $\mathcal{A}_i$. Portanto, a mutação composta de primeira ordem $\mathbb{F}_{\phi_i}M^0$ é computacionalmente equivalente à ação adaptativa $\mathcal{A}_i$ original. □

Até o momento foi provada a equivalência entre as ações adaptativas e as mutações compostas de primeira ordem. Tais provas são a base para mostrar que ambos os formalismos de representação, o original e a versão desta tese, são, de fato, equivalentes. Dessa forma, o novo formalismo pode substituir o original sem causar qualquer perda de expressividade.

Teorema 2.1. *Qualquer autômato adaptativo descrito na forma original possui um AAPO equivalente.*

Prova 2.3 (**do teorema 2.1**). *Essa prova é possível por indução no número de transições adaptativas.*

***Base da indução**: Pelo lema 2.3, para cada transição adaptativa original $(q', \alpha) \rightarrow q'' : \mathcal{A}_i$, existe uma mutação composta de primeira ordem $\mathbb{F}_{\phi_i}$ tal que $(q', \alpha t) \vdash_{[\mathbb{F}_{\phi_i}M^0]} (q'', t)$ e a ação adaptativa $\mathcal{A}$ é mapeada diretamente.*

***Hipótese Indutiva**: Suponha que existe uma sequência de n transições adaptativas da formulação original, onde $0 \leq n \leq \ell$:*

$$(q^k, \alpha^k) \rightarrow q^{k+1} : \mathcal{A}_j$$
$$\ldots$$
$$(q^{k+n}, \alpha^{k+n}) \rightarrow q^{k+(n+1)} : \mathcal{A}_{j+n}$$

então, existe uma sequência $\mathbb{F}_{\phi_{j+n}} \ldots \mathbb{F}_{\phi_j}M^0$, de mutações compostas, na nova formulação, tal que cada $\mathbb{F}_{\phi_i}$, para $1 \leq i \leq n$, é equivalente a $\mathcal{A}_i$ and $(q, \alpha^k \ldots \alpha^{(k+n)}t) \vdash^*_{[\mathbb{F}_{\phi_{j+n}} \ldots \mathbb{F}_{\phi_j}M^0]} (q^{k+(n+1)}, t)$.

***Passo indutivo**: Suponha que exista uma sequência de $n+1$ transições*

adaptativas da formulação original:

$$
\begin{aligned}
&(q, \alpha) \rightarrow q' : \mathcal{A}_1 \\
&(q', \alpha') \rightarrow q'' : \mathcal{A}_2 \\
&\ldots \\
&(q^{(n-1)}, \alpha^{(n-1)}) \rightarrow q^{(n)} : \mathcal{A}_n \\
&(q^{(n)}, \alpha^{(n)}) \rightarrow q^{(n+1)} : \mathcal{A}_{n+1}
\end{aligned}
$$

então, pela hipótese indutiva, existe uma sequência de n transições possíveis $(q, \alpha'\alpha'' \ldots \ \alpha^{(n-1)}t) \ \vdash^*_{[\mathbb{F}_{\phi_n} \ldots \mathbb{F}_{\phi_2} \mathbb{F}_{\phi_1} M^0]} \ (q^{(n)}, t)$ *que realizam a sequência de n transições adaptativas. Basedo nesse resultado e levando em conta a base da indução, é possível construir* $(q^{(n)}, \alpha w) \vdash_{[\mathbb{F}_{\phi_{n+1}} M^0]} (q^{(n+1)}, w)$ *que realiza a mesma tarefa da ação adaptativa* $\mathcal{A}_{n+1}$ *original. Assim, combinando esses resultados, fica claro que qualquer sequência de* $(n + 1)$ *transições adaptativas são devidamente mapeadas por uma sequência de* $(n + 1)$ *transições adaptativas da nova formulação.*

□

3 INFERÊNCIA INDUTIVA E APRENDIZADO NO LIMITE

Uma criança que cresce observando um número cada vez maior de sentenças da sua língua nativa desenvolve uma gramática, ou seja: um conjunto de regras da linguagem, que converge para um gerador e reconhecedor de sentenças válidas para essa linguagem [Yang 2004]. Tal situação ilustra bem os conceitos apresentados no diagrama da figura 1. Do ponto de vista dessa figura, seria possível observar que as reações e o discurso da criança iriam se sofisticando à medida que ela fosse submetida a novas construções sintáticas e a novos exemplos do vocabulário, de modo a inferir **indutivamente** as regras pertinentes à linguagem nativa. A inferência indutiva consiste justamente no processo de geração e escolha de hipóteses sobre as regras que governam um determinado domínio a partir de um fluxo de exemplos disponíveis ao longo do tempo. Assim, a inferência indutiva trabalha com informação eventualmente incompleta em um determinado instante, mas abundante ao longo do tempo. Sua origem histórica se encontra no trbalho de filósofos como Hume [Hume e Chittom 2004], entretanto, nas últimas três décadas, ela tem recebido atenção da área da ciência da computação. Atualmente, a inferência indutiva pode ser considerada como uma forma de aprendizado de máquina com aplicações em inteligência artificial [Russel e Norvig 2003]. Assim, ela é um *framework* de aprendizado, onde a palavra "aprendizado" é entendida como a capacidade de identificar um determinado objeto dentro de uma classe de escolhas possíveis. Para tanto, é necessário investigar as propriedades

de convergência das hipótesese envolvidas em direção a uma que "explique" os exemplos apresentados [Solomonoff 1967].

3.1 Inferência indutiva

Tal processo é realizado por entidades matemáticas denominadas **"aprendizes"**. Podemos utilizar a criança mencionada anteriormente como um exemplo do conceito de aprendiz. Na inferência indutiva, o aprendiz é o responsável por receber, como sua entrada, um fluxo de exemplos a ele apresentados. De posse desses dados ele, de tempos em tempos, gera e modifica suas hipóteses acerca das regras gerais que "explicam" os exemplos.

Formalmente, o Problema da Inferência Indutiva, (ou PII) é visto como a quádrupla apresentada a seguir [Zilles 2001]:

$$I = (\mathfrak{R}, \mathcal{H}, \mathfrak{T}, \mathfrak{I}) \tag{3.1}$$

onde

- $\mathfrak{R}$ é a classe dos conceitos a serem aprendidos. Cada membro dessa classe guarda as regras desconhecidas que precisam ser identificadas. Neste trabalho, uma linguagem formal $L \subseteq \Sigma^*$ será a representação válida de um conceito.

- $\mathcal{H} = (h_1, h_2, h_3, \ldots)$ é o espaço de hipóteses, consistindo de uma família - não necessáriamente finita - de modelos finitos tal que, para cada conceito da classe $\mathfrak{R}$, existe pelo menos uma hipótese que o represente nesse espaço.

- O conjunto $\mathfrak{T}$ é o fluxo de dados ao qual o aprendiz tem acesso. Existem dois tipos de fluxos: os exemplos positivos são aqueles pertinentes à representação da regra desconhecida que se quer aprender, enquanto que os

exemplos negativos não são pertinentes.

- Por fim, $\mathfrak{I}$ é um critério de aprendizagem e refere-se à estratégia de produções das hipóteses pelo aprendiz em relação aos exemplos apresentados. Um critério de aprendizagem se refere ao modo como a sequência de hipóteses de saída será construída em relação ao exemplos recebidos até então.

A inferência indutiva fornece uma estrutura formal para a análise do problemas de aprendizagem e aquisição para uma série de domínios diferentes, inclusive para as linguagens formais. Esse é o caso da **aprendizagem no limite**, também conhecida como **identificação no limite**, introduzida por Emil Mark Gold [Gold 1967].

3.2 Aprendizado no Limite

Gold observou que o fato de um aprendiz "saber" quando deve parar de interpretar os exemplos que recebe (ou seja, um aprendiz que decide quando a hipótese correta foi encontrada) é uma característica muito restritiva para o processo de aprendizagem. Assim, quando uma nova sentença da linguagem apresentada torna a hipótese atual acerca da mesma insustentável, o aprendiz muda essa mesma hipótese para outra e espera que essa nova hipótese seja coerente com a nova setença apresentada, além de sustentar todas as setenças já aceitas anteriormente. Tal procedimento é denominado de **mudança de opinião** (traduzido do inglês ***mindchange***) e ocorre enquanto durar a apresentação de novas sentenças. É facil concluir que para uma linguagem com infinitas sentenças sempre existirá a possibilidade da ocorrência de uma mudança de opinião, daí a importância de se estudar as questões de convergência para essas situações. Tal fato vai de encontro ao segundo postulado, apresentado na seção 1.2.1

A Teoria Algorítmica do Aprendizado, área que estuda o aprendizado no limite, formaliza a figura do aprendiz na forma de um dispositivo recursivo, chamado de **Máquina de Inferência Indutiva** (ou MII). Uma MII recebe como entrada os elementos de $\mathfrak{T}$. De tempos em tempos, a MII apresenta um elemento do conjunto $\mathcal{H}$ como saída. Uma MII pode ser uma função parcial ou total. Na aprendizagem de linguagens formais, não é necessário que a MII seja capaz de aprender toda e qualquer linguagem, mas ela deve ser capaz de lidar com alguma classe de linguagens de interesse.

Uma classe de linguagens pode ser aprendida se e somente se existe um aprendiz tal que, para um fluxo de exemplos apresentados, MII gera as hipóteses corretas sobre o comportamento da linguagem pertencente à classe. Grande parte dos esforços realizados no âmbito do estudo da Teoria Algorítmica do Aprendizado está na análise das classes de linguagens. Existem três tipos de classes envolvidas:

- Classe recursiva. A classe é descrita como uma lista $\mathbb{C}_{L_r} = (L_0, L_1, \ldots)$ de linguagens onde $\{(e, x) : x \in L_e\}$ é recursiva.
- Classe recursivamente enumerável. A classe é descrita como uma lista $\mathbb{C}_{L_{re}} = (L_0, L_1, \ldots)$ de linguagens onde $\{(e, x) : x \in L_e\}$ é recursivamente enumerável.
- Classe geral. A classe é descrita como uma lista $\mathbb{C}_{L_g} = (L_0, L_1, \ldots)$ de linguagens.

A **aprendizagem no limite** de Gold se caracteriza por apresentar o critério $\mathfrak{I}$ verificado na expressão abaixo.

$$\lim_{i \to \infty} h_{i,n} = L_n \tag{3.2}$$

Onde L_n é uma linguagem pertencente a uma determinada classe. Sua representação finita (gramática ou reconhecedor) é desconhecida e ela só pode ser representada pelos seus exemplos, ou seja, L_n é a linguagem que tem de ser identificada pela MII. A lista abaixo menciona as duas principais formas pela qual a convergência desse limite ocorre, a literatura apresenta outras variações [Case e Moelius 2008].

- **Aprendizado Explanatório**: A MII gera uma sequência de hipóteses que converge para uma hipótese de índice i,. Tal hipótese é a resposta correta, ou seja, a partir da i-nésima hipótese, todas as hipóteses geradas serão iguais. Esse tipo de convergência é chamada de convergência sintática [Gold 1967].

- **Aprendizado Comportamentalmente Correto**: Esse critério exige que a semântica (e não a sintaxe) das hipóteses convirjam corretamente no limite. Nesse tipo de convergência, o processo de inferência pode continuar modificando as hipóteses de saída. A MII gera uma sequência infinita de hipóteses que, após um determinado índice i, são todas descrições corretas de L_n para os sub-conjunto de exemplos apresentados entre duas mudanças de opinião consecutivas.

A diferença entre entre os dois tipos de convergência consiste no fato de que, no primeiro tipo, as hipóteses param de mudar e a linguagem L_n é computada pela i-ésima hipótese, enquanto que, no segundo tipo, após uma i-ésima hipótese, as hipóteses geradas não param de mudar, mas reconhecem os exemplos da linguagem apresentados entre duas mudanças de hipóteses consecutivas [Case e Lynes 1982]. Podemos verificar, portanto, que a garantia de convergência é extremamente importante. Assim, é necessário que a MMI convirja sintaticamente ou semanticamente em função dos exemplos apresentados. Neste trabalho,

o tipo de convergência escolhida é a convergência sintática, logo, o escopo dos resultados aqui apresentados estão limitados ao aprendizado explanatório.

Logo abaixo, serão apresentadas definições que formalizam os demais elementos-base da aprendizagem no limite para linguagens formais.

Definição 3.1 (**Texto**). *Um texto consiste em um fluxo de dados composto por uma sequência* $\theta = (t_1, t_2, ...)$ *potencialmente infinita de cadeias de uma linguagem* $L \in \Sigma^*$. *define-se* $t_k \in \theta$ *como um exemplo positivo de* L. *A expressão* $texto(L)$ *denota a classe de todos os textos possíveis de uma linguagem.*

Definição 3.2 (**Segmentos iniciais de texto com comprimento n**). *A sequência finita* $\theta[n] = (t_1, t_2, ...t_n)$ *- onde* $\theta[n]$ *é uma subsequência de* θ *- é chamada de segmento inicial de texto com comprimento n, onde* $n \in \mathbb{N}$. *A classe* $seq(\theta)$ *representa todos os segmentos iniciais possíveis a partir de um determinado* θ.

Definição 3.3 (**Máquina de inferência indutiva**). *Uma máquina de inferência indutiva (abreviado por* MII*) é definida como um procedimento efetivo que computa, segundo um critério* $\mathfrak{I}$ *estabelecido, mapeamentos* $MII \subseteq seq(\theta) \times \mathcal{H}$ *totais ou parciais de exemplos positivos de textos* $\theta \in text(L)$ *em elementos de um espaço de hipóteses* $\mathcal{H}$.

Definição 3.4 (**Função de mudança de opinião**). *Uma* MII *muda sua opinião quando duas hipóteses de saída consecutivas são diferentes. Assim, a função de mudança de opinião é definida como o número de vezes que a* MII *mudou sua saída para um determinado texto. De modo formal, a função de mudança de opinião é expressa por:*

$$mdo(MII, \theta) = card(\{m : \ MII(\theta[m]) \neq MII(\theta[m+1])\}) \tag{3.3}$$

Definição 3.5 (**Critério de Convergência**). *Dado uma máquina de inferência*

indutiva MII e um texto θ de uma linguagem L, $MII(\theta\,[n])$ denota a última hipótese gerada pela MII e a expressão:

$$MII(\theta) \downarrow = h \Leftrightarrow (\exists h \in \mathcal{H})(\forall^{\infty} n)[MII(\theta[n])] = h \tag{3.4}$$

significa que a MII converge sintáticamente em θ para $h \in \mathcal{H}$.

Definição 3.6 (**Identificação no Limite**)**.** *Dada uma família indexada $\mathcal{L}$ de linguagens formais e um espaço de hipóteses $\mathcal{H}$, então:*

$$MII \textbf{ \textit{Lim-identifica} } \mathcal{L} \Leftrightarrow (\forall L \in \mathcal{L})(\exists h \in \mathcal{H} : L(h) = L)[MII(\theta) \downarrow = h] \tag{3.5}$$

significa que a MII é capaz de realizar o aprendizado explanatório de todas as linguagens $L \in \mathcal{L}$.

4 AUTÔMATO ADAPTATIVO DE SEGUNDA ORDEM

Como foi visto na seção 2.1, na formulação original, a ocorrência de uma transição adaptativa pode acarretar a inserção e/ou a remoção tanto das transições pertencentes ao dispositivo subjacente (AFND) quanto das transições adaptativas. Tal possibilidade é uma contradição, pois as ações adaptativas deveriam alterar apenas o conjunto de transições dos dispositivos subjacentes. Para resolver mais essa imprecisão, decidiu-se tonar a regra de alteração das transições dos dispositivos subjacentes mais restritivas e rigorosamente definida. Assim, um AAPO pode alterar somente as transições do seu dispositivo subjacente.

Mas o que acontece caso o AAPO precise inserir ou retirar transições adaptativas? Como foi visto na prova do lema 2.1, existe uma maneira de realizar tal alteração desde que, no caso da inserção, exista uma transição em vazio do AFND, desconectada de todas as demais, associada a uma mutação na forma de uma transição adaptativa de primeira ordem. Porém, tal solução na prática pode ser computacionalmente mais complexa e custosa em termos de tempo e espaço, pois se existir a necessidade de inserir n novas transições adaptativas de primeira ordem, deverão existir n transições em vazio desconectadas, pertencentes ao AFND.

Assim, é possível observar que tornar rígida a regra de permitir modificações apenas nos dispositivos subjacentes não é um mero preciosismo. Entretanto, qual solução deve ser dada quando surge a necessidade de alterar as transições

adaptativas de primeira ordem, mas não se deseja "sobrecarregar" o dispositivo subjcente com transições em vazio? A resposta é: cria-se um novo reconhecedor adaptativo que respeite de forma rigorosa a regra de modificar apenas seu dispositivo subjacente. Dessa forma, deve-se criar um reconhecedor adaptativo que possua, como dispositivo subjacente, um AAPO, de modo a ter a capacidade de modificar as regras modificadoras desse. Tal reconhecedor será chamado de **autômato adaptativo de segunda ordem**.

4.1 Identificação no limite e a tecnologia adaptativa

Embora o aprendiz tenha sido visto como uma caixa-preta, existe um método universal para implementá-lo, denominado identificação por enumeração [Angluin e Smith 1983]. Ele se baseia na geração de uma enumeração computável de todas as descrições $h_1, h_2, h_3, \ldots$ das linguagens L_n de uma classe $\mathcal{L}$, de modo que cada linguagem da classe tenha uma descrição presente nessa enumeração.

Dado um texto θ dessa linguagem, o método irá buscar, na enumeração, a primeira descrição h_i compatível com os exemplos dados. No caso de um reconhecedor, por exemplo, a compatibilidade é verificada se as cadeias do texto apresentado são todas reconhecidas por h_i. Logo, a verificação dessa compatibilidade também deve ser computável. O critério de convergência para esse método se baseia em duas condições:

- A primeira condição é que uma hipótese correta é sempre compatível com os exemplos dados.
- A segunda condição é que uma hipótese incorreta é incompatível com uma coleção suficientemente grande de exemplos para todos os textos dados [Angluin e Smith 1983].

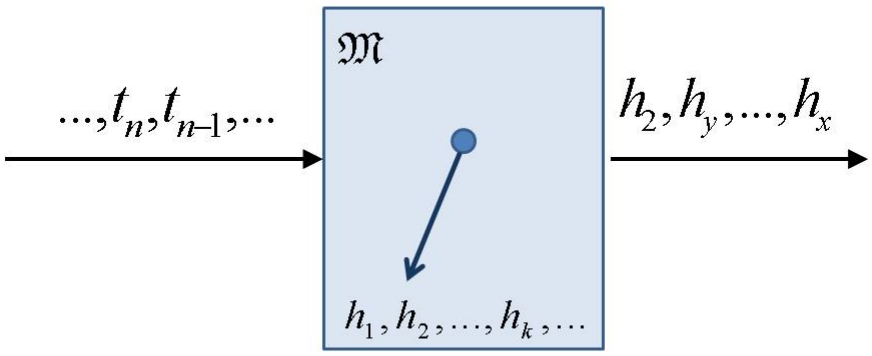

Figura 2: Método de identificação por enumeração

Apesar de genérica e poderosa, a identificação por enumeração pode se tornar impraticável dependendo do tamanho da classe de linguagens envolvida.

Porém, suponha-se a existência de uma estrutura computacional U que possua, internamente, um reconhecedor R_i para uma determinada linguagem L_n, dado um alfabeto Σ, de forma que para cada modificação efetuada, tal estrutura altere R_i, gerando um novo reconhecedor para uma linguagem diferente da anterior, mas pertencente à mesma classe $\mathcal{L}$.

Suponha-se também que tal estrutura U possua internamente um componente associado; um módulo que será chamado ζ. Entre suas atribuições, esse módulo possui a capacidade de "ler" os exemplos positivos $t_i \in \theta$ de uma linguagem e induzir as modificações em R_i.

Neste ponto, é possível fazer a seguinte pergunta: a estrutura U é um dispositivo adaptativo? A resposta envolve o comportamento do módulo ζ. Se o módulo induz as alterações no reconhecedor R_i em função dos exemplos lidos, então, pela

própria definição, a resposta será sim.

Reconhecedores são dispositivos que aceitam cadeias de caracteres como entrada. Tais cadeias fazem com que o reconhecedor percorra uma série de configurações até atingir uma configuração onde está presente um estado de aceitação ou de rejeição. Explorando essa característica, se ao invés de uma cadeia de caracteres, o dispositivo adaptativo U fosse submetido a um fluxo potencialmente infinito de cadeias de uma mesma linguagem, seria possível utilizar as características adaptativas para identificar linguagens por meio de exemplos.

Como foi visto antes, independentemente do critério de aprendizagem ou da classe de linguagens, as hipóteses geradas são definidas como uma enumeração das representações finitas de uma linguagem (reconhecedores são representações finitas). Assim, utilizando a equivalência entre autômatos adaptativos e as Máquinas de Turing mencionadas em [Rocha e Neto 2000], é possível definir o conjunto das hipóteses geradas por um aprendiz como sendo a lista $\mathcal{H} = (M_1^1, M_2^1, M_3^1, \ldots)$, composta por uma enumeração dos autômatos adaptativos. Dessa forma, sem perda de generalidade, os autômatos adaptativos podem ser utilizados como uma representação do conjunto de hipóteses em um processo de identificação no limite.

Voltando à descrição da estrutura U, nada se afirmou sobre as características das modificações induzidas por ζ e o conjunto de auto-modificações induzido por esse módulo poderia ser arbitrário. Porém, será estabelecido que toda modificação aplicada em R_i irá gerar reconhecedores para linguagens diferentes, porém pertencentes a uma mesma classe de linguagens. Se o exemplo apresentado para U não for reconhecido pela atual configuração da estrutura, o módulo ζ requisita uma modificação de R_i que, ao se "adaptar", passa a aceitar ou não o exemplo apresentado como pertencente à linguagem que representa. Enquanto o exemplo apresentado não for reconhecido, ζ continuará requisitando modificações. Quando o exemplo é aceito, a estrutura U apresenta como saída o reconhecedor R_i adaptado

para reconhecer todos os exemplos anteriormente apresentados, mais o último que foi responsável pela sua transformação. A figura 4.1 apresenta um diagrama de blocos para a estrutura U.

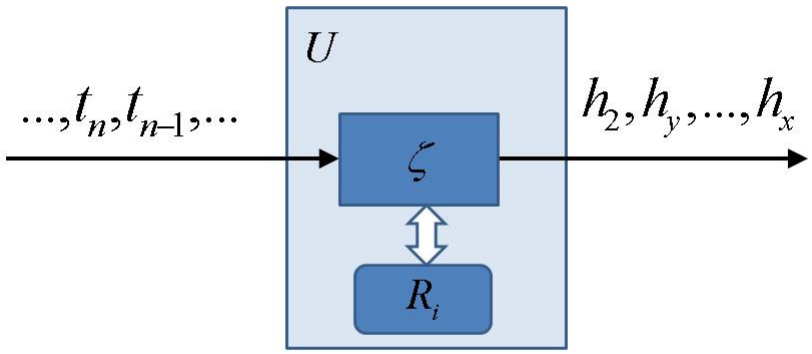

Figura 3: Diagrama de blocos da Estrutura U proposta

Já que U é um dispositivo adaptativo por definição, a seguinte pergunta poderia ser feita: é possível reproduzir o comportamento de U utilizando puramente o modelo dos autômatos adaptativos original? Caso não seja possível, quais novos elementos teríamos que incluir para obter uma solução que se comporte como a estrutura U idealizada?

Nada impede que o reconhecedor R_i seja um autômato adaptativo. Sendo assim, tal fato não tornaria o módulo ζ desnecessário? Via de regra, um autômato adaptativo é construído para reconhecer cadeias de uma determinada linguagem. Poderiam as ações adaptativas mudar a linguagem reconhecida pelo autômato caso esse não reconhecesse uma cadeia de entrada?

Para responder essa pergunta é possível imaginar o seguinte cenário: um

programador possui um arquivo em um determinado formato. Esse arquivo possui a seguinte série de cadeias:

1000000001

1100000011

1110000111

1111001111

1111111111

O programador é orientado a desenvolver um reconhecedor para a linguagem representada pelas cadeias presentes no arquivo com uma diretiva bem explícita: o reconhecedor deverá ser um autômato adaptativo. O resultado do seu trabalho será guardado em um repositório, onde o programador coloca o código-fonte e o código executável produzido. Caso a versão guardada lá funcione para todas as *cadeias* contidas no arquivo dado, então um novo conjunto de exemplos será fornecido em um outro arquivo.

Para facilitar o trabalho do desenvolvedor, são dadas duas informações: todas as *cadeias* contidas nos arquivos são exemplos válidos da linguagem dada e a linguagem em questão é $L = 1^n0^*1^n$ com $n \geq 0$. Com essas informações, o programador parte para o desenvolvimento do reconhecedor. Obviamente, de posse dessas informações, ele irá conseguir projetar o autômato adaptativo para tratar essa situação.

Porém, o que aconteceria se ele não recebesse a descrição da linguagem, de maneira a ter somente os exemplos dados no arquivo? Existiria uma maneira do programador descobrir a linguagem somente com base nos exemplos e assim construir o seu reconhecedor ?

Na situação em que carece de informações, o programador provavelmente analisaria, de alguma forma, os exemplos fornecidos, já que essa é a única informação que ele possui. Em algum momento, ele poderia fazer uma suposição, algo como "*provavelmente*, a linguagem representada pelas cadeias é: $L = ww^{-1}$." Implementando o autômato adaptativo baseado nessa suposição, ele verifica que de fato seu palpite foi muito bom, pois todas as *cadeias* do arquivo foram processadas com sucesso. Assim, ele armazena seu reconhecedor no repositório e aguarda o próximo arquivo.

O segundo arquivo chega com os novos dados:

0000000000

00000000

0000

00

0

O programador fornece as cadeias ao seu reconhecedor, anteriormente bem-sucedido. Entretanto, durante o processamento dos exemplos, a última cadeia composta pelo símbolo "0"não é aceita. Caso ele não mude de suposição, ou seja, não implemente outro autômato adaptativo com um conjunto de funções adaptativas diferente, seu programa não reconhecerá por completo todo o conjunto de novos exemplos dados.

Diante desse impasse, uma informação adicional e intermediária é dada ao programador: "a linguagem em questão pertence a uma determinada classe de linguagens." Essa pode não ser a melhor das informações. Dependendo da classe em questão, o número de linguagens-membro pode ser muito grande, podendo

ser infinito.

Porém, para uma classe que possua um número finito e não demasiadamente grande de linguagens-membro, o programador "percebe" que pode implementar um autômato adaptativo para cada linguagem-membro, formando assim uma pequena "biblioteca" de reconhecedores. Dessa forma, caso uma cadeia seja rejeitada por um dos autômatos de sua biblioteca, o programador poderia escolher outro de sua coleção de implementações e iniciar uma nova tentativa, mudando as funções adaptativas do seu autômato. A cada nova suposição, uma nova atualização chegaria no repositório. Ao final, quando o programador "acertasse" a linguagem representada pelas cadeias contidas nos arquivos, o repositório teria uma enumeração de todas as versões dos reconhecedores usados. Assim, ao encontrar o reconhecedor certo para os exemplos que recebeu por via dos arquivos, pode-se afirmar que o programador "aprendeu" com qual linguagem ele está trabalhando.

Neste momento, a atitude do programador frente ao problema de encontrar o reconhecedor correto é muito parecida com o comportamento do módulo ζ na figura 4.1. Apesar do módulo ζ ser descrito como um dispositivo capaz de gerar modificações no reconhecedor R_i em função das entradas recebidas, se fosse possível que cada modificação tivesse o mesmo efeito das trocas realizadas pelo programador com sua "biblioteca", então o programador poderia ser substituído pelo módulo ζ.

Assim, caso uma cadeia do texto não seja aceita pelo dispositivo R_i subjacente (um autômato adaptativo), uma ação adaptativa oriunda de ζ - nesse caso uma **ação adaptativa oriunda de ordem superior, ou seja, uma ação adaptativa de segunda ordem** - é ativada e o dispositivo adaptativo subjacente é alterado e submetido novamente ao fluxo de cadeias acrescido de uma nova cadeia. Se todas as cadeias do fluxo forem aceitas, o dispositivo adaptativo subjacente

não precisa ser alterado pelas ações adaptativas de segunda ordem geradas por ζ. A idéia central dessa abordagem é que a sequência de ações adaptativas de segunda ordem gerem, a partir do dispositivo subjacente inicial, novas hipóteses acerca da linguagem representada pelos exemplos que compõem o fluxo de entrada de cadeias. Após uma série de execuções de ações adaptativas de segunda ordem, pelo menos mais de uma linguagem Turing-computável foi representada pelo autômato adaptativo de primeira ordem, pois o mesmo já foi alterado mais de uma vez. Dessa forma, para um fluxo de cadeias, fornecido pelo texto θ de uma linguagem desconhecida, a estrutura U com suas ações adaptativas de segunda ordem se auto-modificaria até convergir para a linguagem-alvo.

No modelo atual, um autômato adaptativo, por si só, não pode alterar a linguagem ao qual ele está codificado para reconhecer. As funções adaptativas podem alterar o autômato finito subjacente de modo a este reconhecer cadeias as quais ele não possuía configurações de aceitação *a priori*, mas todas essas palavras pertecem a uma mesma linguagem fixada. Mudar a linguagem aceita pelo autômato adaptativo implica em, no mínimo, alterar o conjunto de funções adaptativas presente no autômato.

Mas qual a vantagem de usar essa estrutura adaptativa de segunda ordem como aprendiz para identificação de linguagens formais no limite? Como foi dito anteriormente, a estratégia de identificação por enumeração pode ser inviável por necessitar de uma lista com uma enumeração de todas as linguagens-membro da classe que está sendo investigada. Porém, no caso da estrutura U, tal lista é construída à medida que as cadeias da linguagem são processadas. Assim, seria possível tratar classes de linguagens mais extensas até eventualmente atingir a situação em que a linguagem representada pelos exemplos apresentados nos arquivos seja aprendida no limite, no sentido dado por Gold (e presente na definição 3.6), sem a necessidade de ter todas as linguagens enumeradas *a priori*.

O modelo dos Autômatos Adaptativos não é o suficiente para modelarmos toda a estrutura U. Para ser mais específico, não podemos reduzir toda a estrutura U ao modelo dos Autômatos Adaptativos. Embora o módulo ζ da arquitetura não pertença ao modelo dos autômatos adaptativos, as instâncias R_i que são as funções reconhecedoras podem ser Autômatos Adaptativos. Dessa forma, as ações adaptativas em cada R_i também estariam sujeitas a um processo adaptativo, pois o módulo ζ teria a capacidade de alterar as funções adaptativas em cada R_i com o objetivo de gerar novas hipóteses.

4.2 Autômato adaptativo de segunda ordem

Assim, existe um conceito novo com que trabalhar: a Adaptatividade de Segunda Ordem. Nela, temos dispositivos adaptativos que também estão sujeitos a sofrer auto-modificações. A adaptatividade de segunda auxilia na identificação de linguagens onde a única informação disponível é o conjunto dos exemplos apresentados a um aprendiz. Na adaptatividade de segunda ordem, cada auto-modificação gera um Autômato Adaptativo para uma nova linguagem, com o intuito de se adaptar aos exemplos apresentados e convergir para a linguagem representada por esses mesmos exemplos.

Definição 4.1 (**Classe $\mathscr{M}^1$**). *Dado um alfabeto Σ, define-se $\mathscr{M}^1$ como a classe de todos os AAPOs possíveis de serem construídos sobre Σ.*

Definição 4.2 (**Transição adaptativa própria**). *Dada a classe $\mathscr{M}^1$ de todos os AAPO sobre um alfabeto Σ, para cada membro M^1 dessa classe, uma* ***transição adaptativa própria*** *é definida como:*

$$\delta^1_{pro} = \delta^1 : \delta^1 \in \partial^1 \tag{4.1}$$

Onde ∂^1 é a relação de transição adaptativa de primeira ordem do AAPO

M^1.

Definição 4.3 (**Transição adaptativa estrangeira**). *Uma* ***transição adaptativa estrangeira*** *de* M^1 *é definida como:*

$$\delta^1_{for} = \delta^1 : \delta^1 \notin \partial^1 \tag{4.2}$$

Onde ∂^1 *é a relação de transição adaptativa de primeira ordem do membro* M^1.

Definição 4.4 (**Sequência positiva de primeira ordem**). *Uma* ***sequência positiva de primeira ordem*** *é definida como:*

$$\lambda^1_{pro} = (\delta^1_{pro_1}, \ldots, \delta^1_{pro_m}) \tag{4.3}$$

ou seja: uma sequência positiva de primeira ordem é uma sequência composta de transições adaptativas próprias pertencentes um dado M^1.

Definição 4.5 (**Sequência negativa de primeira ordem**). *Uma* ***sequência negativa de primeira ordem*** *é definida como:*

$$\lambda^1_{for} = (\delta^1_{for_1}, \ldots, \delta^1_{for_n}) \tag{4.4}$$

ou seja: uma sequência negativa de primeira ordem é uma sequência composta de transições estrangeiras para um dado M^1.

Definição 4.6 (**Par de transformações de segunda ordem**). *Dada uma sequência negativa de primeira ordem* λ^1_{for} *e uma sequência positiva de primeira ordem* λ^1_{pro}, *ambas para o mesmo autômato* M^1, *um* ***par de transformações de segunda ordem*** *é definido como:*

$$\psi \triangleq (\lambda^1_{for}, \lambda^1_{pro}) \tag{4.5}$$

4.2.1 Operadores Sintáticos de primeira ordem

Tomando as seguintes definições:

- Transição própria de primeira ordem
- Transição estrangeira de primeira ordem
- Função de remoção
- Função de inserção

é possível definir operadores que alteram a relação de transição adaptativa de membros da família $\mathscr{M}^1$, bem como seus conjuntos de estados, de maneira análoga ao que foi feito com os AFND pertencentes à família $\mathscr{M}^0$. De forma semelhante aos operadores sintáticos definidos para os AFNDs, existem dois operadores e são denominados **operadores sintáticos de primeira ordem**, pois alteram a linguagem reconhecida pelo autômato M^1 modificado por eles.

Definição 4.7 (**δ^1-remoção**). *Dado um AAPO M^1 pertencente à classe $\mathscr{M}^1$ e utilizando-se a função definida na equação-1.6, para qualquer transição adaptativa própria de M^1, uma δ-remoção é definida como:*

$$g^- M^1 = g^-(\delta^1_{pro}, M^1) = (f^- M^0, \Phi, \phi_0, \boldsymbol{rem}(\partial^1, \delta^1_{pro})) \tag{4.6}$$

Definição 4.8 (**δ-inserção**). *Dado um AAPO M^1 e utilizando-se a função definida na equação-1.7, para qualquer transição adaptativa estrangeira de M^1, uma δ-inserção é definida como:*

$$g^+M^1 = g^+(\delta^1_{for}, M^1) = (f^+M^0, \boldsymbol{ins}(\Phi, \phi), \phi_0, \boldsymbol{ins}(\partial^1, \delta^1_{for})) \tag{4.7}$$

4.2.2 Transformações Sintáticas de primeira ordem

As expressões 4.6 e 4.7 das operações de inserção e remoção podem ser combinadas para gerar transformações mais complexas para os elementos da família $\mathscr{M}^1$. Essas transformações utilizam a definição 4.5, bem como a definição 4.6. A idéia é análoga àquela aplicada aos AFNDs: criar transformações capazes de inserir e remover uma sequência de transições (no caso dos AAPOs, transições adaptativas). Por essa razão, tais transformações são denominadas **transformações sintáticas de primeira ordem**.

Definição 4.9 (**Transformação expansiva de segunda ordem**)**.** *Dada uma sequência negativa de primeira ordem λ^1_{for} para o AAPO M^1, a* ***transformação expansiva de primeira ordem*** *é uma trasformação sintática de primeira ordem definida como:*

$$G^+_{\lambda^1_{for}} M^1 \triangleq G^+(\lambda^1_{for}, M^1) = (g^+_n \circ g^+_{n-1} \circ \ldots \circ g^+_2 \circ g^+_1)M^1 \tag{4.8}$$

Definição 4.10 (**Transformação contrativa de segunda ordem**)**.** *Dada uma sequência positiva de primeira ordem λ^1_{pro} para o AAPO M^1, a* ***transformação contrativa de primeira ordem*** *é definida como:*

$$G^-_{\lambda^1_{pro}} M^1 \triangleq G^-(\lambda^1_{pro}, M^1) = (g^-_m \circ g^-_{m-1} \circ \ldots \circ g^-_2 \circ g^-_1)M^1 \tag{4.9}$$

O próximo passo natural nessa série de construções de transformações é exatamente definir o análogo, para os AAPOs, da mutação composta definida para os AFNDs.

Definição 4.11 (**Mutação composta de segunda ordem**). *Dado um par de transformações de segunda ordem ψ para um autômato M^1, uma mutação composta de segunda ordem é definida como:*

$$\mathbb{G}_{\psi} \triangleq \mathbb{G}(\psi, M^1) = G^{-}_{\lambda^1_{pro}} G^{+}_{\lambda^1_{for}} M^1 \tag{4.10}$$

Assim como a mutação composta de primeira ordem, existem casos especiais que podem ocorrer no cálculo das mutações compostas de segunda ordem que envolvem os pares de transformações de segunda ordem presentes nessas mutações. Tais situações são três:

- Situação 1: a sequência positiva de primeira ordem λ^1_{pro} do par de transformações de segunda ordem ψ é uma sequência vazia.
- Situação 2: a sequência negativa de primeira ordem λ^1_{for} do par de transformações de segunda ordem ψ é uma sequência vazia.
- Situação 3: Tanto a a sequência positiva λ^1_{pro} quanto a a sequência negativa λ^1_{for} do par de transformações de segunda ordem ψ são sequências vazias.

Tal como no caso da mutação composta de primeira ordem, as peculiaridades da situação três, com as duas sequências vazias, leva a definição de um novo par de transformação.

Definição 4.12 (**Par vazio de segunda ordem**). *O par de transformação de segunda ordem onde tanto a sequência positiva de primeira ordem λ^1_{pro} quanto a sequência negativa de primeira ordem λ^1_{for} são sequências vazias é denominado de par vazio de segunda ordem, sendo representado pela notação $\psi^{\emptyset}$.*

Dessa forma, a mutação composta de segunda ordem assume os seguintes comportamentos frente às três situações especiais envolvendo as ocorrências de sequências vazias nos pares de transformação de segunda ordem:

- $\mathbb{G}_{\psi} M^1 = G^{+}_{\lambda^1_{for}} M^1$ para a situação 1.
- $\mathbb{G}_{\psi} M^1 = G^{-}_{\lambda^1_{pro}} M^1$ para a situação 2.
- $\mathbb{G}_{\psi_\emptyset} M^1 = M^1$ para a situação 3.

É possível observar que o comportamento da mutação composta de segunda ordem se mostra semelhante ao de primeira no que diz respeito às diferentes ocorrências de sequências vazias nos pares de transformação de segunda ordem. A mutação composta se reduz a cada uma das transformações sintáticas, tanto a contrativa quanto a expansiva. Novamente, no caso extremo, quando utilizado o par vazio de segunda ordem, a mutação composta não afeta o AAPO M^1.

4.2.3 Autômato adaptativo de segunda ordem

Todas as definições apresentadas até o momento são mais do que exercícios sobre as possibilidades de contrução de transformações para alterar AAPOs. A meta é construir uma nova família de reconhecedores, porém, agora com outras propriedades. Assim como os AAPOs possuíam um dispositivo subjacente que sofria alterações toda vez que uma transição adaptativa de primeira ordem era realizada, esse novo reconhecedor também possuirá essa característica. Agora, entretanto, o dispositivo subjecente a ser modificado não é um AFND, mas sim um AAPO, ou seja: o novo reconhecedor modificará as regras (transições adaptativas de primeira ordem) do AAPO de forma análoga ao modo através do qual o próprio AAPO modificava as regras (transições de estado) do AFND.

Definição 4.13 (**Autômato adaptativo de segunda ordem**). *Um autômato adaptativo de segunda ordem (AASO) é uma quádrupla*

$$M^2 = (M^1, \Psi, \psi_0, \partial^2) \tag{4.11}$$

Onde:

- $M^1 \in \mathscr{M}^1$ *é o* ***dispositivo subjacente do AASO****.*
- Ψ *é um conjunto de pares de transformações de segunda ordem, chamado de* ***conjunto de comportamentos de segunda ordem****.*
- *O elemento* $\psi_0 \in \Psi$ *é um par vazio de segunda ordem, chamado de* ***comportamento nulo de segunda ordem****. Todo conjunto de comportamentos de segunda ordem de um AASO possui um comportamento nulo.*
- ∂^2 *é a* ***relação de transição adaptativa de segunda ordem****. Cada elemento de* ∂^2 *assume a forma:*

$$\delta^2_{i,k,j} = (\delta^1_{i,k}, \langle M^2 \langle \mathbb{G}_{\psi_j} M^1 \rangle \rangle) \tag{4.12}$$

sendo chamado de transição adaptativa de segunda ordem, com $\psi_j \in \Psi$ *e* $\delta^1_{i,k} \in \partial^1$*, onde* ∂^1 *é a relação de transição do dispositivo subjacente de segunda ordem.*

Como foi feito na descrição dos AAPOs, é necessário, agora, apresentar os conceitos pertinentes à computação de uma cadeia por parte dos AASO, conceitos esse que, novamente, são baseados na teoria clássica dos autômatos. Dessa forma, os conceitos listados abaixo vão ser definidos para os AASO.

- Configuração
- Relação de mudança de configuração
- Fechamento transitivo e reflexivo da relação de mudança de configuração

- Linguagem reconhecida pelo autômato

Definição 4.14 (**Configuração**). *A **configuração** de um AASO é a dupla*

$$(q', c) \in Q \times \Sigma^* \tag{4.13}$$

onde Q é o conjunto de estados do dispositivo subjacente do AASO e c é a cadeia de entrada em alguma etapa do seu processamento.

Definição 4.15 (**Configuração inicial**). *A **Configuração inicial** de um AASO é representado pela dupla (q_0, t), onde q_0 é o estado inicial do dispositivo subjacente do AASO e t é a cadeia de entrada a ser analisada.*

Neste momento, a seguinte pergunta poderia surgir: "Qual é o estado inicial do dispositivo subjacente de segunda ordem?" A resposta é simples: o estado do dispositivo subjacente de segunda ordem é o estado inicial do dispositivo subjacente de primeira ordem, ou seja, continua sendo o estado inicial do AFND "contido" no AAPO que agora serve de dispositivo subjacente para o AASO. O mesmo ocorre com cada configuração do AASO.

Definição 4.16 (**Relação de mudança de configuração**). *A **Relação de mudança de configuração**, que descreve como o AASO muda de uma configuração para outra, é definida como:*

$$(q', t) \vdash_{[\langle \mathbb{G}_{\psi_j} M^1 \langle \mathbb{F}_{\phi_k} M^0 \rangle \rangle]} (q'', w) \Leftrightarrow \exists \alpha \in \Sigma : \alpha w = t \tag{4.14}$$

onde

- q'' *é um estado de* $\mathbb{F}_{\phi_k} M^0$
- $\phi_k \in \Phi$

- $\psi_j \in \Psi$

Definição 4.17 (**Fechamento transitivo e reflexivo da relação de mudança de configuração**). *O **fechamento transitivo e reflexivo da relação de mudança de configuração** de um AASO é definido como segue:*

$$(q', t) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_s}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1 \langle \mathbb{F}_{\phi_{k_s}} \ldots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0 \rangle \rangle]} (q'', w) \tag{4.15}$$

se e somente se:

- $(q' = q'')$ *e* $(w = t)$ *ou*
- *as regras 1 a 4 definidas abaixo são satisfeitas:*

1. $t = a_0 a_1 \ldots a_s w$ *com* $a_z \in \Sigma$ *para* $0 \leq z \leq s$
2. $\exists(\phi_{k_1}, \phi_{k_2}, \ldots \phi_{k_{s+1}})$ *com* $\phi_{k_z} \in \Phi$ *para* $1 \leq z \leq s$
3. $\exists(\psi_{j_1}, \psi_{j_2}, \ldots \psi_{j_{s+1}})$ *com* $\psi_{j_z} \in \Psi$ *para* $1 \leq z \leq s$
4. $\exists\ p_1, p_2 \ldots p_s \in Q$, *onde o conjunto* Q *é conjunto de estados do dispositivo subjacente do AAPO (que por sua vez é o dispositivo subjacente do AASO) a cada mutação , de modo que:*

$$\begin{aligned}
(q', t) \quad & \vdash_{[\langle \mathbb{G}_{\psi_{j_1}} M^1 \langle \mathbb{F}_{\phi_{k_1}} M^0 \rangle \rangle]} (p_1, a_1 a_2 a_3 \ldots a_s w) \\
& \vdash_{[\langle \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1 \langle \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0 \rangle]} (p_2, a_2 a_3 \ldots a_s w) \\
& \vdash_{[\langle \mathbb{G}_{\psi_{j_3}} \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} \langle \mathbb{F}_{\phi_{k_3}} \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0 \rangle \rangle]} \ldots \\
& \vdash_{[\langle \mathbb{G}_{\psi_{j_s}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1 \langle \mathbb{F}_{\phi_{k_s}} \ldots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0 \rangle \rangle]} (p_s, a_s w) \\
& \vdash_{[\langle \mathbb{G}_{\psi_{j_{s+1}}} \mathbb{G}_{\psi_{j_s}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1 \langle \mathbb{F}_{\phi_{k_{s+1}}} \mathbb{F}_{\phi_{k_s}} \ldots \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0 \rangle \rangle]} (q'', w)
\end{aligned}$$

Definição 4.18 (**Linguagem reconhecida por um AASO**). *A **linguagem reconhecida por um AASO** é definida como*

$$L(M^2) = \{t : (q_0, t) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_s}} ... \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1 \langle \mathbb{F}_{\phi_{k_j}} ... \mathbb{F}_{\phi_{k_2}} \mathbb{F}_{\phi_{k_1}} M^0 \rangle \rangle]} (q_f, \varepsilon)\} \tag{4.16}$$

Na situação especial em que o conjunto de comportamentos possui apenas o comportamento nulo, ou seja, $\Psi = \{\psi_0\}$, as condições necessárias para que uma cadeia seja aceita pelo AAPO passsam a ser:

$$(q_0, t) \vdash^*_{[\mathbb{F}_{\phi\emptyset} ... \mathbb{F}_{\phi\emptyset} \mathbb{F}_{\phi\emptyset} M^0]} (q_f, \varepsilon) = (q_0, t) \vdash^*_{M^0} (q_f, \varepsilon)$$

Nessa situação, o reconhecimento da cadeia t passa a ser feito pelo dispositivo subjacente inicial do AASO. Assim, o AASO assume o comportamento de um AAPO.

5 RESULTADOS

Se um AAPO tem seu poder computacional equivalente à máquina de Turing, qual é a vantagem em utilizar um AASO? Por definição, um AASO transforma AAPOs pela aplicação de mutações compostas de segunda ordem. Ao comparar esse comportamento ao processo de *mindchange* descrito na definição, pode-se observar uma conexão entre os dois conceitos. O AASO se auto-modifica em função dos novos estímulos externos, enquanto que o aprendiz, na inferência indutiva, muda sua hipótese acerca da linguagem quando confrontado com um estímulo que não pode ser sustentado pela hipótese atual. Tomando essa sequência de transformações, é possível demonstrar que os AASO podem ser utilizados para realizar a aprendizagem no limite de Gold para linguagens formais. Uma vez que os AAPOs são Turing-equivalentes, eles podem representar uma classes de linguagens formais dentro da hierarquia de Chomsky e essa característica é importante para estabelecer a ponte entre adaptatividade e aprendizado.

5.1 AASO e o aprendizado no limite

Antes de tratar da conexão entre os AASOs e o aprendizado no limite , é necessário mostrar como representar o processamento não de uma cadeia, mas de um fluxo delas. Como foi visto na seção 3.1, um aprendiz processa um segmento de texto sempre crescente para uma determinada linguagem. Assim, é necessário criar uma representação para o processamento desse "fluxo" por parte de um

AASO, bem como uma maneira de permitir o uso dos AASO para a geração de hipóteses, que é o objetivo de um dispositivo aprendiz. Em segundo lugar, é necessário mostrar como é possível derivar definições de linguagens usando exclusivamente as características dos AASO e como tais definições se relacionam com o conceito de aprendizagem no limite. Assim, todos os elementos necessários para descrever o aprendizado no limite em termos de adaptatividade de segunda ordem estarão disponíveis. Durante o seu processamento, o AASO altera seu dispositivo subjacente. Se antes de sofrer a próxima alteração uma cópia desse dispositivo for guardada em uma lista, uma sequência de todas as "versões" dos AAPOs internos ao AASO estará disponível. Percebe-se ai um processo semelhante à geração de hipóteses por parte de um aprendiz no sentido de Gold, o que seria muito interessante, já que os AAPOs são Turing-equivalentes, ou seja, o espaço de hipóteses poderia ser, *a priori* composto por subconjuntos de funções Turing-computáveis.

Definição 5.1 (**Classe $\mathscr{M}^2$**). *Dada a classe $\mathscr{M}^1$ de todos os AAPOS, define-se $\mathscr{M}^2$ como a classe de todos os AASOs possíveis de serem construídos sobre $\mathscr{M}^1$.*

Definição 5.2 (**Função de reação**). *Dado um AASO M^2 e um texto θ qualquer de uma determinada linguagem L, a* ***função de reação Reac*** $: \mathscr{M}^2 \times seq(\theta) \rightarrow \mathscr{M}^2$ *retorna o dispositivo subjacente (AAPO) de um AASO M^1 após processar um segmento de texto $\theta[m]$ e após alterar (ou não) o dispositivo subjacente original de M^2.*

Definição 5.3 (**Classe alomórfica**). *Dada a classe $\mathscr{G}$ de todas as mutações compostas de segunda ordem para todos os AAPOs em $\mathscr{M}^1$, tome uma subclasse finita $\mathcal{G} \subset \mathscr{G}$. Tal classe é denominada* ***classe alomórfica****.*

Definição 5.4 (**Composição alomórfica**). *Dado uma classe alomórfica $\mathcal{G}$, a* ***composição alomórfica*** *é indicada por $\mathcal{G}^i$, para $i \in \mathbb{N}$, e é definido indutivamente da seguinte forma:*

1. $\mathcal{G}^0 = \{\mathbb{G}_{\psi^0}\}$,

2. $\mathcal{G}^1 = \mathcal{G}$,

3. $\forall(\mathbf{g} \in \mathcal{G}^1)\forall(\mathbf{g}' \in \mathcal{G}^n)$, $\mathcal{G}^{n+1} = \mathbf{g} \circ \mathbf{g}'$

Definição 5.5 (**Fechamento reflexivo transitivo sobre $\mathcal{G}$**). *Dado uma classe alomórfica $\mathcal{G}$, o fechamento reflexivo transitivo sobre essa classe alomórfica é definido como:*

$$\mathcal{G}^* = \bigcup_{i=0}^{\infty} \mathcal{G}^i \tag{5.1}$$

A seguir, será mostrado como a definição do fechamento reflexivo transitivo sobre $\mathcal{G}$ pode ser utilizado para definir classes de linguagens formais.

Definição 5.6 (**Família alomórfica**). *Uma família alomórfica é uma classe alomórfica indexada, representada por $\{\mathcal{G}_i\}_{i\in\mathbb{N}}$.*

Definição 5.7 (**Função de escolha**). *Dado uma família alomórfica $\{\mathcal{G}_i\}_{i\in\mathbb{N}}$, uma função computável $e : \mathbb{N} \to \mathcal{G}$ é chamada de **função de escolha**.*

Definição 5.8 (**Família radicular**). *Dada uma família alomórfica $\{\mathcal{G}_i\}_{i\in\mathbb{N}}$, um AAPO M^1 qualquer e uma função de escolha, uma **família radicular** é indicada por $\mathcal{R}$ e definida indutivamente da seguinte forma:*

1. ***Base da indução**: O AAPO M^1 é o elemento m_0 de $\mathcal{R}$.*

2. ***Primeira cláusula indutiva**: Se m_i é membro de $\mathcal{R}$ então*

$$m_{i+1} = e(i)m_i \tag{5.2}$$

também é elemento de $\mathcal{R}$. A indexação dos membros de $\mathcal{R}$ é induzida pela função de escolha, o qual determina uma enumeração para os membros da família radicular.

3. ***Segunda cláusula indutiva**: Não existem outras maneiras de construir membros para uma família radicular.*

Definição 5.9 (**Família $\mathcal{L}_R$ de linguagens**). *Dada uma família alomórfica $\{\mathcal{G}_i\}_{i\in\mathbb{N}}$, um AAPO M^1 qualquer, uma função de escolha e a família radicular $\mathcal{R}$ definida por esses elementos, a **Família $\mathcal{L}_R$ de linguagens** é definida como a família das linguagens $\mathcal{L}_R = L(\mathcal{R})$ definidas sobre os AAPO gerados na família radicular $\mathcal{R}$.*

Definição 5.10 (**Problema adaptativo confinado**). *Dado um AAPO M^1 e uma linguagem L, onde $L \neq L(M^1)$, tome uma sequência $C_\propto = (\mathbb{G}_{\psi_i}, \ldots, \mathbb{G}_{\psi_j}, \ldots, \mathbb{G}_{\psi_k})$ cujos elementos são mutações compostas de segunda ordem. Caso a igualdade a seguir se verifique:*

$$L = L((\mathbb{G}_{\psi_i} \ldots (\mathbb{G}_{\psi_j} \ldots (\mathbb{G}_{\psi_k}(M^1)) \ldots) \ldots)) \tag{5.3}$$

*então, L é um **problema adaptativo confinado** e M^1 é uma **semente para** L.*

Definição 5.11 (**Sequência de alcançabilidade**). *Dado um problema adaptativo confinado L, a sequência $C_\propto$ que torna a igualdade 5.3 verdadeira é chamada de **sequência de alcançabilidade**.*

Definição 5.12 (**Família dos problemas adaptativos confinados lineares**). *Dada uma classe alomórfica $\mathcal{G}_L$, a família $\mathcal{L}_{C_\propto} = \{L_n\}_{n\in\mathbb{N}}$ de problemas adaptativos confinados baseados na mesma semente, onde a sequência de alcançabilidade de L_i é uma subsequência da sequência de alcançabilidade de L_{i+1} e os membros da sequência $C_{\propto_i}$ de qualquer linguagem da família pertencem a $\mathcal{G}_L$, é chamada de **família dos problemas adaptativos confinados lineares**.*

Teorema 5.1. *Se L é um problema adaptativo confinado, então, para qualquer texto θ de L, existe um AASO M^2 e um número $m > 0$ (com $m \in \mathbb{N}$) tal que:*

$$\begin{cases} L \neq L(M^2) & para\ \theta[n] \\ L = L(M^2) & para\ \theta[n+1] \end{cases}$$

Prova 5.1 (**do teorema 5.1 (por construção)**). *Tome um AASO no qual o dispositivo subjacente M^1 é uma semente para um problema adaptativo confinado L. Tome também um conjunto de comportamentos de segunda ordem Ψ, onde todos os elementos da sequência de alcançabilidade $C_\propto$ de L estão presentes em Ψ. Assim, pela definição 5.10 é possível definir uma relação de transição adaptativa de segunda ordem onde a computação do texto θ assume a forma:*

$$(q_0, t_1) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_s}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1 \rangle]} (q', w')$$

$$(q_0, t_2) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_t}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)' \rangle]} (q'', w'')$$

$$\ldots$$

$$(q_0, t_{n-1}) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_u}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^{(n-2)} \rangle]} (q^{n-1}, w^{n-1})$$

$$(q_0, t_n) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_w}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^{(n-1)} \rangle]} (q^n, w^n)$$

$$(q_0, t_{n+1}) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_p}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^n \rangle]} (q_f, \epsilon)$$

$$(q_0, t_{n+2}) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_v}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^{(n+1)} \rangle]} (q_f, \epsilon)$$

$$\ldots$$

$$(q_0, t_{n+k}) \vdash^*_{[\langle \mathbb{G}_{\psi_{j_l}} \ldots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^{(n+(k-1))} \rangle]} (q_f, \epsilon)$$

para $q', q'', \ldots, q^{n-1}, q^n$ diferentes de q_f; $w'; ew'', \ldots, w^{n-1}, w^n$ diferentes

de ϵ *e*

$$(M^1)' = \mathbb{G}_{\psi_{j_s}} \dots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1$$

$$(M^1)'' = \mathbb{G}_{\psi_{j_t}} \dots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)'$$

$$\dots$$

$$(M^1)^{n-1} = \mathbb{G}_{\psi_{j_u}} \dots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^{n-2}$$

$$(M^1)^{n} = \mathbb{G}_{\psi_{j_w}} \dots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^{n-1}$$

$$(M^1)^{n+1} = \mathbb{G}_{\psi_{j_p}} \dots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} (M^1)^{n}$$

$$\dots$$

de forma que a execução das transições adaptativas gera a sequência de transformações adaptativas descritas abaixo:

$$L = L(\mathbb{G}_{\psi_{j_p}} \dots \mathbb{G}_{\psi_{j_w}} \dots \mathbb{G}_{\psi_{j_u}} \dots \mathbb{G}_{\psi_{j_t}} \dots \mathbb{G}_{\psi_{j_s}} \dots \mathbb{G}_{\psi_{j_2}} \mathbb{G}_{\psi_{j_1}} M^1)$$

e, para qualquer t_{n+k} *para* $k > 1$*, a seguinte expressão é verdadeira*

$$(M^1)^{(n+(k-1))} = (M^1)^{(n+1)}$$

Assim, para $n > 0$*, existe um AASO para o problema adaptativo confinado* L *tal que:*

$$\begin{cases} L \neq L(M^2) & para\ (t_1, t_2, \dots, t_n) \\ L = L(M^2) & para\ (t_{n+1}, \dots) \end{cases}$$

□

Teorema 5.2. *Existe um AASO M^2 para o qual, dado qualquer texto θ_k de uma linguagem L_i membro de uma família $\mathcal{L}_{C_\propto}$, a função* ***Reac***(M^2, θ_k) *é uma MII onde:*

$$\textbf{\textit{Reac}}(M^2, \theta_k)\ \textbf{\textit{Lim-identifica}}\mathcal{L}_{C_\propto} \tag{5.4}$$

Prova 5.2 (do teorema 5.2 (por construção)). *Como foi provado no teorema 5.1, para qualquer problema adaptativo confinado L é possível construir com AASO M^2 tal que:*

$$\begin{cases} L \neq L(M^2) & para\ \theta[n] \\ L = L(M^2) & para\ \theta[n+1] \end{cases}$$

Portanto, é possível afirmar que:

$$\exists(n \in \mathbb{N})\exists(M^1 \in \mathscr{M}^1 : L(M^1) = L)(\forall m \geq n)[\textbf{\textit{Reac}}(M^2, \theta[m]) = M^1]$$

em outras palavras:

$$\exists(M^1 \in \mathscr{M}^1 : L(M^1) = L)[\textbf{\textit{Reac}}(M^2, \theta[m]) \downarrow = M^1]$$

significando que a função ***Reac***(M^2, θ) ***Lim-identifica*** L.

Pela definição 5.12, a sequência de alcançabilidade de L_i (com $i \geq 0$) de $\mathcal{L}_{C_\propto}$ é uma subsequência de alcançabilidade para a sequência de alcançabilidade de L_z, com $z \geq i$. Assim, usando o teorema 5.2, é possível mostrar que

para L_0 $\exists(\boldsymbol{Reac}(M_0^2,.) : \boldsymbol{Reac}(M_0^2,.)$ $\boldsymbol{Lim\text{-}identifica}$ (L_0)

para L_1 $\exists(\boldsymbol{Reac}(M_1^2,.) : \boldsymbol{Reac}(M_1^2,.)$ $\boldsymbol{Lim\text{-}identifica}$ (L_0, L_1)

para L_2 $\exists(\boldsymbol{Reac}(M_2^2,.) : \boldsymbol{Reac}(M_2^2,.)$ $\boldsymbol{Lim\text{-}identifica}$ (L_0, L_1, L_2)

...

para L_i $\exists(\boldsymbol{Reac}(M_i^2,.) : \boldsymbol{Reac}(M_i^2,.)$ $\boldsymbol{Lim\text{-}identifica}$ $(L_0, L_1, L_2, \ldots, L_i)$

...

para L_n $\exists(\boldsymbol{Reac}(M_n^2,.) : \boldsymbol{Reac}(M_n^2,.)$ $\boldsymbol{Lim\text{-}identifica}$ $\boldsymbol{\mathcal{L}}_{C_\infty}$

□

Tal conclusão traz um resultado imediato para as classes de hipóteses que podem ser utilizadas.

Corolário 5.1. *A classe $\mathscr{M}^1$ dos AAPO é um espaço de hipóteses admissível.*

5.2 Exemplo

Para o alfabeto $\Sigma = \{a, b\}$, tome-se a linguagem:

$$L = a^n b^n \tag{5.5}$$

A linguagem acima é uma linguagem livre de contexto. É possivel tomar essa expressão denotacional e definir uma classe de linguagens apenas estendendo a definição do alfabeto utilizado. Assim, generalizando a expressão de Σ tem-se:

$$\Sigma = \{\xi_{(i)} | max(i) = Max\} \tag{5.6}$$

Para o alfabeto acima, considere-se então a família de linguagens :

$$\mathcal{L} = \xi_{(x)} \ldots \xi_{(y)} \alpha^i \xi_{(z)} \ldots \xi_{(t)} \beta^i \xi_{(w)} \ldots \xi_{(k)} \tag{5.7}$$

onde:

- $i > 0$.
- os símbolos α e β pertencem a Σ.
- $\xi_{(y)} \neq \alpha$ e $\xi_{(z)} \neq \alpha$.
- $\xi_{(t)} \neq \beta$ e $\xi_{(w)} \neq \beta$

A característica que torna essa escolha interessante, reside no aninhamento sintático entre os símbolos α e β presentes nas cadeias de todas as linguagens que são membros dessa família. Para cada cadeia pertencente a uma linguagem da classe $\mathcal{L}$, o aninhamento sintático garante que, para uma subcadeia com i repetições do simbolo α, deve existir, na mesma cadeia, uma subcadeia com mesmo número i de repetições do simbolo β. O Aninhamento sintático é a característica fundamental de uma série de linguagens formais importantes, tais como as linguagens livres de contexto e as linguagens de Dyck [Mott, Kandel e Baker 1986]. Tal regra de aninhamento, de agora em diante, será representado pela expressão:

$$\wp(\alpha, \beta) \tag{5.8}$$

Mas o aninhamento sintático não é a única característica relevante nessa classe de linguagens. Cada linguagem-membro de $\mathcal{L}$ possui também uma cadeia de menor tamanho, diferente da cadeia vazia. A estrutura e tamanho dessa cadeia depende da maneira pela qual os símbolos do alfabeto foram combinados durante a geração das cadeias, no que diz respeito à ordem e sequência estabelecidas. Será utilizada a expressão abaixo:

$$\omega_{min} \;=\; \xi_{(x)} \ldots \xi_{(y)} \alpha \xi_{(z)} \ldots \xi_{(t)} \beta \xi_{(w)} \ldots \xi_{(k)} \tag{5.9}$$

para representar genericamente todas as cadeias mínimas das linguagens pertencentes à família $\mathcal{L}$. Note-se que a expressão 5.9 consiste na expressão geral 5.7 para o caso onde $i = 1$.

Assim, conhecendo-se o alfabeto, o aninhamento sintático e a menor cadeia diferente da cadeia vazia, é possível determinar univocamente uma linguagem dentro da família $\mathcal{L}$. Dessa forma, é possível caracterizar cada linguagem-membro pela expressão:

$$L^{\Sigma,\omega_{min}}_{\wp(\alpha,\beta)} \tag{5.10}$$

Como a cadeia mínima se mantém constante para cada linguagem, a única mudança de uma linguagem-membro para outra são os símbolos, presentes na cadeia mínima, que estão relacionados entre si pela dependência sintática. Isso implica que uma linguagem-membro se diferencia de outra pelo par de símbolos presentes na dependência sintática $\wp(\alpha, \beta)$. Assim, é possível não só caracterizar todos os elementos da família $\mathcal{L}$, como enumerá-los, usando como critério os diferentes pares de dependência sintática das linguagens-membros. Logo, $\mathcal{L}$ é uma família indexável de linguagens recursivas e pode também ser definida em função de suas linguagens-membro pela expressão:

$$\mathcal{L} = \bigcup_{\alpha,\beta \;\in\; \omega_{min}} (L^{\Sigma,\omega_{min}}_{\wp(\alpha,\beta)}) \tag{5.11}$$

Usar um AASO no reconhecimento de membros dessa família por meio de exemplos positivos implica em responder se é possível construir pelo menos um

desses dispositivos de modo que o dispositivo subjacente de primeira ordem, ou seja, o AAPO, convirja para a linguagem representada pelos textos da linguagem.

Assim, considere-se a seguinte situação: existe um texto θ que pertence a uma linguagem desconhecida X. A única informação conhecida sobre essa linguagem reside no fato que X pertence à classe $\mathcal{L}$ para um dado Σ. Logo, a questão se resume em saber se existe um AASO que identifica a linguagem X representada pela sequência θ.

Como isso pode ser verificado? Lembrando o resultado do teorema 5.2, se $\mathcal{L}$ for uma família de problemas adaptativos confinados lineares, então esse AASO existe. Assim, para responder se esse AASO existe, é necessário verificar se $\mathcal{L}$ é uma família de problemas adaptativos confinados lineares ou não.

O primeiro passo para isso, é estabelecer uma enumeração para os elementos da classe $\mathcal{L}$, de modo a torná-la uma família. Como foi mencionado, é possível criar uma enumeração aproveitando as características da dependência sintática, de modo a estabelecer uma ordem entre as linguagens membro da classe de modo a indexá-los. Para isso, realiza-se o seguinte procedimento: toma-se a cadeia mínima da classe $\mathcal{L}$ reescrita da seguinte forma:

$$\omega_{min} = a_1 a_1 \dots a_k \tag{5.12}$$

Para uma cadeia de tamanho mínimo k, onde $a_1, a_1, \dots, a_k \in \Sigma$. O critério de ordem escolhido, utilizando o conceito de depêndencia sintática, é o descrito a seguir:

$$L_0 = L^{\Sigma,\omega_{min}}_{\wp(a_1,a_2)} = a_1^i a_2^i \dots a_k$$

$$L_1 = L^{\Sigma,\omega_{min}}_{\wp(a_1,a_3)} = a_1^i a_2 a_3^i \dots a_k$$

...

$L_{k-2} = L^{\Sigma,\omega_{min}}_{\wp(a_1,a_k)} = a_1^i a_2 a_3 \ldots a_k^i$

$L_{k-1} = L^{\Sigma,\omega_{min}}_{\wp(a_2,a_3)} = a_1 a_2^i a_3^i \ldots a_k$

$L_k = L^{\Sigma,\omega_{min}}_{\wp(a_2,a_4)} = a_1 a_2^i a_3 a_4^i \ldots a_k$

...

$L_{C_{k,2}} = L^{\Sigma,\omega_{min}}_{\wp(a_{k-1},a_k)} = a_1 a_2 a_3 a_4 \ldots a_{k-1}^i a_k^i$

Ou seja, as linguagens cujos símbolos pertencentes à dependência sintática estão mais à esquerda têm os índices menores, enquanto que aqueles que possuem os símbolos da dependência sintática mais à direita da palavra mínima possuem os índices mais altos. Como essa família é finita, o total de membros também será determinado pela dependência sintática, sendo correspondente ao número de todas combinações possíveis para os pares de símbolos.

Dada essa indexação, a linguagem $L_0 = L^{\Sigma,\omega_{min}}_{\wp(a_1,a_2)}$ é a linguagem de menor índice. Uma das condições para que família $\mathcal{L}$ seja uma família de problemas adaptativos confinados lineares é que L_0 seja uma semente para todas as demais membros de $\mathcal{L}$. Porém, antes de verificar isso, é preciso mostrar que existe um AAPO que reconheça a linguagem L_0.

Nos trabalhos referentes à formulação original já foram apresentados exemplos de autômatos adaptativos para linguagens livres de contexto bem semelhantes à linguagem L_0, um exemplo pode ser visto em [Neto 2001]. Usando o resultado do teorema 2.1, poder-se-ia admitir por equivalência com os exemplos já apresentados na formulação original, a existência de um AAPO para L_0. Entretanto, construir um AAPO para L_0 servirá para exercitar a nova formulação.

Deve-se começar, portanto, procurando qual é o dispositivo subjacente M^0 desse AAPO.

Uma maneira de se conseguir isso é verificando o seguinte fato: todas as linguagens de L_0 possuem a mesma cadeia mínima, logo, um AFND que reconheça essa cadeia é um reconhecedor que funciona para todas as linguagens de $\mathcal{L}$ quando $i = 1$, portanto é um reconhecedor conveniente para a cadeia mínima de L_0. Porém, como tratar as demais cadeias da linguagem? A resposta a essa pergunta está na relação de dependência sintática $\wp(a_1, a_2)$.

Para reconhecer qualquer cadeia de $\mathcal{L}$ com $i > 0$ adaptando o AFND que reconhece a cadeia de menor tamanho é necessário converter a relação de depêndencia sintática em uma mutação composta que adapte o AFND a novas cadeias onde $i > 0$. Para tanto, a regra de dependência sintática deve ser reescrita dentro de um ponto de vista mais voltado para o processamento das cadeias por parte do reconhecedor.

"*Dada uma cadeia qualquer de L_0, com $i > 1$ repetições de a_1, para cada símbolo a_1 processado, o AFND que reconhece a cadeia mínima deverá possuir uma transição que reconheça um símbolo a_2*".

A frase acima pode parecer óbvia, porém sua interpretação é sutil: para cada símbolo a_1 processado, o AFND deverá possuir uma transição que reconheça um símbolo a_2, **mesmo que inicialmente não a possua**. Assim, é possivel reescrever a frase acima da seguinte forma:

"*Após o processamento da primeira ocorrência de a_1, para cada símbolo a_1 processado, uma nova transição $(q', a_2, q")$ deve ser inserida no AFND que reconhece a cadeia mínima*".

Portanto, a primeira modificação a ser feita no AFND que reconhece a menor cadeia para adequá-lo ao caso geral é permitir o reconhecimento da primeira ocorrência de i e das demais $i-1$ ocorrências, de modo a permitir o processamento de qualquer cadeia de L_0.

Esse novo AFND será, dessa forma, o dispositivo subjacente M^0 do AAPO procurado.

$$M^0 = (Q, q_0, E, \Sigma, \partial) \tag{5.13}$$

onde:

$-Q = \{q_0, q_1, q_2, \ldots, q_{k-1}, q_f\}$

$-E = \{q_f\}$

$-\partial = \{\delta_1, \delta_2, \delta_3, \delta_4, \ldots, \delta_{k+1}\}$

com

$\delta_1 = (q_0, a_1, q_1)$

$\delta_2 = (q_1, a_1, q_1)$

$\delta_3 = (q_1, a_2, q_2)$

$\delta_4 = (q_2, a_3, q_3)$

$\ldots$

$\delta_{k+1} = (q_{k-1}, a_k, q_f)$

Agora, é preciso descrever a transformação que o dispositivo subjacente M^0 sofre para se adaptar a cada nova cadeia de entrada pertencente a L_0. Como foi declarado anteriormente: *...para cada símbolo a_1 processado, uma nova transição $(q', a_2, q")$ deve ser inserida no AFND que reconhece a cadeia mínima.* A mutação composta de primeira ordem $\mathbb{F}_{\phi_1}$ que cumpre tal requisito tem o seu par de tansformações adaptativas descrito por:

$$\phi = (\lambda_{for_1}, \lambda_{pro_1}) \tag{5.14}$$

para:

$$\lambda_{for_1} = ((\mathbf{Start}(\mathbf{Sch}(b, q_2, M^0)), b, q'), (q', b, q_2))$$

$$\lambda_{pro_1} = ((\mathbf{Start}(\mathbf{Sch}(b, q_2, M^0)), b, q_2))$$

Lembrando que a função **Start** foi definida em 1.20 como o ponteiro que, dada uma transição de um AFND, retorna o primeiro estado da transição. Já a função **Sch** faz a busca da transição cujos elementos foram passados como argumentos da função.

Agora, para completar o AAPO falta apenas definir as transições adaptativas, ou seja, estabelecer quais as transições do dispositivo subjacente que estarão associadas à mutação composta de primeira ordem. Dado que uma nova transição para a_2 deve ser inserida sempre que um novo processamento de um símbolo a_1 é realizado, a mutação composta deve ser associada à transição $\delta_2 = (q_1, a_1, q_1)$. As demais transições de M^0 não possuem transições adaptativas e, portanto, serão associadas a mutações compostas de primeira ordem com pares de tansformações vazios.

Assim, obtém-se um dispositivo subjacente, o conjunto de comportamentos de primeira ordem e uma relação de transição adaptativa de primeira ordem, estabelecendo por completo todos os componentes do AAPO que reconhece a linguagem L_0.

$$M^1_{L_0} = (M^0, \Phi, \phi_0, \partial^1) \tag{5.15}$$

definido com o conjunto de comportamentos de primeira ordem:

$$\Phi = \{\phi_0, \phi_1\}$$

bem como a relação de transição adaptativa de primeira ordem abaixo:

$$\partial^1 = \left\{\delta^1_{1,0}, \delta^1_{2,1}, \delta^1_{3,0}, \delta^1_{4,0}, \ldots, \delta^1_{(k+1),0}\right\}$$

contendo as transições adaptativas mostradas a seguir.

$\delta^1_{1,0} = (\delta_1, \langle M^1 \langle \mathbb{F}_{\phi_0} M^0 \rangle \rangle)$

$\delta^1_{2,1} = (\delta_2, \langle M^1 \langle \mathbb{F}_{\phi_1} M^0 \rangle \rangle)$

$\delta^1_{3,0} = (\delta_3, \langle M^1 \langle \mathbb{F}_{\phi_0} M^0 \rangle \rangle)$

$\delta^1_{4,0} = (\delta_4, \langle M^1 \langle \mathbb{F}_{\phi_0} M^0 \rangle \rangle)$

...

$\delta^1_{(k+1),0} = (\delta_{k+1}, \langle M^1 \langle \mathbb{F}_{\phi_0} M^0 \rangle \rangle)$

Com o AAPO para L_0, o próximo passo para descobrir se $\mathcal{L}$ é uma família de problemas adaptativos confinados é buscar uma sequência de alcançabilidade $C_\propto$ que, aplicada a L_0, gere $L_{C_{k,2}}$ de forma que as sequências de alcançabilidade de todas as linguagens cujos índices estejam entre 0 e $C_{k,2}$ sejam subsequências de $C_\propto$.

A indexação das linguagens de $\mathcal{L}$ seguiram uma orientação determinada pelas dependências sintáticas dos pares de símbolos da cadeia mínima mais à esquerda para os pares mais à direita.

$L_0 = L^{\Sigma,\omega_{min}}_{\wp(a_1,a_2)}$

$L_1 = L^{\Sigma,\omega_{min}}_{\wp(a_1,a_3)}$

...

$L_{k-2} = L^{\Sigma,\omega_{min}}_{\wp(a_1,a_k)}$

$L_{k-1} = L^{\Sigma,\omega_{min}}_{\wp(a_2,a_3)}$

$L_k = L^{\Sigma,\omega_{min}}_{\wp(a_2,a_4)}$

...

$L_{C_{k,2}} = L^{\Sigma,\omega_{min}}_{\wp(a_{k-1},a_k)}$

Tomando, então, apenas as dependências sintáticas, tem-se que:

$\wp(a_1, a_2)$

$\wp(a_1, a_3)$

...

$\wp(a_1, a_k)$

$\wp(a_2, a_3)$

$\wp(a_2, a_4)$

...

$\wp(a_{k-1}, a_k)$

Como foi visto na construção do AAPO da expressão 5.15, a dependência sintática é implementada no autômato através da transição adaptativa de primeira ordem. Tal transição é formada pela mutação composta de primeira ordemn que está associada à transição que processa as $i-1$ ocorrências do primeiro símbolo do par da dependência sintática. Assim, para modificar a linguagem reconhecida pelo AAPO, é necessário modificar essa transição adaptativa e a mutação composta que implementa a transformação do dispositivo subjacente.

Observando-se os membros de $\mathcal{L}$ do ponto de vista das dependências

sintáticas, verifica-se que é possível converter um membro i em um membro $i+1$ mudando o par de símbolos das respectivas dependências sintáticas. Dessa forma, tal mudança pode ser expressa por:

$$\wp(a_i, a_{i+1}) \overset{a_{i+1}|a_{i+2}}{\rightarrow} \wp(a_i, a_{i+2}) \tag{5.16}$$

ou:

$$\wp(a_i, a_{i+2}) \overset{a_i|a_{i+1}}{\rightarrow} \wp(a_{i+1}, a_{i+2}) \tag{5.17}$$

e também, no caso mais geral, por:

$$\wp(a_i, a_{i+1}) \overset{a_i|a_{i+1};a_{i+1}|a_{i+2}}{\rightarrow} \wp(a_{i+1}, a_{i+2}) \tag{5.18}$$

Assim, uma sequência de alcançabilidade de L_0 para $L_{C_{k,2}}$ pode ser expressa por:

$\wp(a_1, a_2) \overset{a_2|a_3}{\rightarrow} \wp(a_1, a_3)$

$\wp(a_1, a_3) \overset{a_3|a_4}{\rightarrow} \wp(a_1, a_4)$

...

$\wp(a_1, a_{k-1}) \overset{a_{k-1}|a_k}{\rightarrow} \wp(a_1, a_k)$

$\wp(a_1, a_k) \overset{a_1|a_2;a_k|a_3}{\rightarrow} \wp(a_2, a_3)$

$\wp(a_2, a_3) \overset{a_3|a_4}{\rightarrow} \wp(a_2, a_4)$

...

$\wp(a_{k-2}, a_k) \overset{a_{k-2}|a_{k-1}}{\rightarrow} \wp(a_{k-1}, a_k)$

Cada linguagem da família $\mathcal{L}$ pode ser representada por seu respectivo AAPO. A questão de provar se a sequência de alcançabilidade existe se reduz ao trabalho de mostrar que é possível construir as transformações acima para cada AAPO que reconhece uma das linguagens membro, de forma que cada dependência sintática, descrita na forma de uma transição adaptativa, possa ser alterada. Portanto, para compor a sequência de alcançabilidade e alterar as transições adaptativas de primeira ordem, tais trasformações devem ser descritas como mutações compostas de segunda ordem.

Caso seja possível provar que tais mutações compostas de segunda ordem existem para o caso genérico:

$$\wp(a_i, a_{i+1}) \overset{a_i|a_{i+1};a_{i+1}|a_{i+2}}{\rightarrow} \wp(a_{i+1}, a_{i+2})$$

então tem-se que a sequência de alcançabilidade $C_\propto$ existe e toda sequência de alcançabilidade dos demais membros da família serão subsequências de $C_\propto$.

O par de transformações de segunda ordem que determina a solução genérica para a mutação composta de segunda ordem $\mathbb{G}_{\psi_{\wp(a_i,a_{i+1})\rightarrow\wp(a_{i+1},a_{i+2})}}$ é apresentado abaixo. Essa mutação implementa a transformação apresentada na expressão 5.18, provando que a sequência $C_\propto$ existe.

$$\psi_{\wp(a_i,a_{i+1})\rightarrow\wp(a_{i+1},a_{i+2})} = (\lambda^1_{for_{a_{i+1},a_{i+2}}}, \lambda^1_{pro_{a_i,a_{i+1}}}) \tag{5.19}$$

com:

$$\lambda^1_{for_{\wp(a_{i+1},a_{i+2})}} = (\delta^1_{for_{\wp(a_{i+1},a_{i+2})}})$$

$$\lambda^1_{pro_{\wp(a_i,a_{i+1})}} = (\delta^1_{pro_{\wp(a_i,a_{i+1})}})$$

onde

$$\delta^1_{for_{\wp(a_{i+1},a_{i+2})}} = \left((q_{i+1}, a_{i+1}, q_{i+1}), \left\langle M^1 \left\langle \mathbb{F}_{\phi_{i+2}} M^0 \right\rangle \right\rangle\right)$$

$$\delta^1_{pro_{\wp(a_i,a_{i+1})}} = \left((q_i, a_i, q_i), \left\langle M^1 \left\langle \mathbb{F}_{\phi_{i+1}} M^0 \right\rangle \right\rangle\right)$$

Os pares de transformações de primeira ordem ϕ_{i+1} e ϕ_{i+2} são generalizações do par de transformações apresentado na expressão 5.14.

$$\phi_{i+1} = (\lambda_{for_{i+1}}, \lambda_{pro_{i+1}}) \tag{5.20}$$

para:

$$\lambda_{for_{i+1}} = ((\mathbf{Start}(\mathbf{Sch}(a_{i+1}, q_{i+1}, M^0)), a_{i+1}, q'), (q', a_{i+1}, q_{i+1}))$$

$$\lambda_{pro_{i+1}} = ((\mathbf{Start}(\mathbf{Sch}(a_{i+1}, q_{i+1}, M^0)), a_{i+1}, q_{i+1}))$$

bem como o segundo par de transformações de primeira ordem:

$$\phi_{i+2} = (\lambda_{for_{i+2}}, \lambda_{pro_{i+2}}) \tag{5.21}$$

com:

$$\lambda_{for_{i+2}} = ((\mathbf{Start}(\mathbf{Sch}(a_{i+2}, q_{i+2}, M^0)), a_{i+2}, q'), (q', a_{i+2}, q_{i+2}))$$

$$\lambda_{pro_{i+2}} = ((\mathbf{Start}(\mathbf{Sch}(a_{i+2}, q_{i+2}, M^0)), a_{i+2}, q_{i+2}))$$

6 CONSIDERAÇÕES FINAIS

Assim, com a apresentação do exemplo da seção 5.2, cumpriram-se as metas estabelecidas na tese. Com a apresentação dos AAPOs no capítulo 2 e dos AASOs, no capítulo 4, bem como com a apresentação da relação entre os conjuntos $\mathscr{M}^1$ e $\mathscr{M}^2$, definiu-se a hierarquia de reconhecedores formais adaptativos. Mostrou-se também, no capítulo 5, que para se trabalhar com identificação no limite apenas com exemplos positivos, os AASOs são suficientes e necessários. Tanto os AAPOs quanto os AASOs foram definidos em função unicamente dos AFNDs e dos conjuntos de transformações sobre autômatos, definidos nas subseções 2.2.2 e 4.2.2.

A mutação composta de primeira ordem, expressa pela transformação $\mathbb{F}_\phi M^0$, restringiu a natureza e o número de parâmetros permitidos para as ações adaptativas para apenas uma: o par de transformação de primeira ordem. Assim, usar as transformações adaptativas para redefinir o conceito de ação adaptativa permitiu resolver os dois problemas mencionados em [Neto e Bravo 2003]. Como resultado secundário, a notação dos autômatos adaptativos se tornou mais compacta.

Um resultado importante deste trabalho foi conseguir mostrar a forte conexão entre aprendizado no limite e os AASOs provando ser possível representar um processo de aprendizagem usando o AASO como um aprendiz em um processo de identificação no limite de linguagens formais. Na prática, foi criada uma nova forma de geração de hipóteses que vai além da simples enumeração. Com as

mutações compostas de segunda ordem, é possível gerar uma nova hipótese "reciclando" a hipótese antiga, representada por um AAPO. Assim, a partir deste ponto de vista, qualquer hipótese pode ser usada para iniciar um processo de aprendizagem, e, após uma transformação passo-a-passo dessa hipótese por um AASO, produz um modelo correto final, resultando em uma aprendizagem computacional eficaz. Dessa forma, a inferência indutiva pode ser vista sob uma outra perspectiva. O resultado que determina a subclasse $\mathcal{L} = \bigcup_{\alpha,\beta \in \omega_{min}} (L^{\Sigma,\omega_{min}}_{\wp(\alpha,\beta)})$ de linguagens livres de contexto como sendo identificáveis no limite por um AASO também é importante na medida que várias linguagens da classe livre de contexto podem ser descritas nessa formulação.

Na seção 3.1, apresentou-se a formulação geral do problema de inferência indutiva, caracterizada pela quádrupla:

$$PII = (\mathfrak{R}, \mathcal{H}, \mathfrak{T}, \mathfrak{I})$$

Dessa forma, ao final da apresentação deste trabalho, é possível particularizar essa formulação, definindo a formulação geral do Problema da Inferência Indutiva Adaptativa (ou PIIA) para , através da quádrupla:

$$PIIA = (\mathcal{L}_{C_\propto}, \mathcal{H} \subseteq \mathscr{M}^1, texto(\mathcal{L}_{C_\propto}), \mathbf{Reac}(M^2, \theta_k)\ \mathbf{Lim\text{-}identifica}\mathcal{L}_{C_\propto}) \quad (6.1)$$

Cabe lembrar que uma dada família de problemas adaptativos confinados lineares $\mathcal{L}_{C_\propto}$ existe em função de uma classe alomórfica dada. Portanto, existem infinitas famílias $\mathcal{L}_{C_\propto}$, cada uma com infinitas linguagens. Mas para todas essas famílias existe um AASO $M^2 \in \mathscr{M}^2$ capaz de identificar uma linguagem membro no limite utilizando apenas exemplos positivos. As linguagens livre de contexto pertencentes à $\mathcal{L} = \bigcup_{\alpha,\beta \in \omega_{min}} (L^{\Sigma,\omega_{min}}_{\wp(\alpha,\beta)})$ são apenas um exemplo de famílias de

problemas adaptativos confinados lineares possíveis.

6.1 Resultados

Aqui, os objetivos propostos nas subseções 1.2.3 e 1.2.2 são revisitados de modo a mostrar que foram atingidos. Dessa forma as contribuições desse trabalho são discriminadas a seguir.

- Subresultados atingidos:

Meta	**Resultado**
conjunto de transformações para os AFNDs	Seção2.2
conjunto de transformações para os AAPOs	Subseção4.2.2

Tabela 1: Resultados intermediários.

- Propostas da tese atingidas:

Meta	**Resultado**
AAPO	Definição2.12
equivalência do AAPO com a formulação original	Teorema2.1
AASO	Definição4.13
AASO como um aprendiz no limite	Teorema 5.2
Aplicação para linguagens livre de contexto	Seção 5.2

Tabela 2: Metas da tese.

6.2 Trabalhos Futuros

A inclusão das transformações adaptativas abriu uma nova linha de pesquisa no estudo dos formalismos adaptativos. Estudar as características algébricas e funcionais de tais transformações é um próximo trabalho a ser realizado. Tais características incluem uma análise mais aprofundada das propriedades apresentadas por tais transformações.

Com respeito às classes $\mathscr{M}^0$ de AFNDs, $\mathscr{M}^1$ de AAPOs e $\mathscr{M}^2$ de AASO utilizadas neste trabalho, uma questão interessante a ser analisada são as suas

características topológicas, sua estrutura e geometria. Com respeito a isso, existem vários tópicos a serem explorados, como, por exemplo, que tipo de métrica é possível associar a cada um dos espaços.

Outra questão diz respeito à hierarquia dos autômatos adaptativos. Um trabalho a ser realizado é estudar a necessidade e a possibilidade da existência de classes de autômatos adaptativos de ordem superior à segunda, ou seja: a classe de autômatos adaptativos que tenham os AASOs como dispositivos subjacentes. Um estudo das características de tais ordens mais elevadas, caso existam, é um trabalho futuro importante e instigante a ser feito. Tal investigação se relaciona com a necessidade de definir todas as limitações do modelo computacional adaptativo e, em seguida, definir as limitações de aprendizado neste modelo. As restrições e limites intrínsecos à hierarquia de autômatos adaptativos, caso existam, têm de ser formalmente definidos. Para os fins deste trabalho, a segunda ordem foi necessária e suficiente.

Como foi visto na definição 5.8, os membros da família radicular são construídos através da expressão:

$$m_{i+1} = e(i)m_i$$

que também representa o comportamento de um **sistema dinâmico discreto**. Neste ponto, o PIIA faz intersecção com uma área que, a princípio, parece não ter relacionamento algum com os modelos computacionais: os sistemas dinâmicos. Embora o mais comum seja o estudo de simulações de sistemas dinâmicos utilizando modelos computacionais, o que se propõe aqui como trabalho futuro é o estudo de modelos computacionais como sistemas dinâmicos. Especificamente falando, como o comportamento adaptativo para a geração de hipóteses pode ser analisado como um sistema dinâmico e quais as consequências desse fato para as

questões pertinentes ao processo de aprendizado.

Mas não é apenas na área de aprendizado de máquina que o estudo dos autômatos adaptativos como sistemas dinâmicos pode apresentar resultados interessantes. Em [Chaitin 2010], é apresentado o conceito de "DNA como programa". Nesse sentido, se a estrutura e a dinâmica do DNA podem ser representados computacionalmente, seria interessante pesquisar como os AASO podem ser usados para modelar os fenômenos ligados à evolução e conceitos tais como a mutação e a recombinação genética.

Outra fonte de pesquisas futuras diz respeito à aplicação da adaptatividade de segunda ordem na teoria dos jogos, com o objetivo de estudar o aprendizado em jogos onde as regras mudam dinâmicamente. Imaginando um jogo não-colaborativo onde um dos jogadores não conhece as regras do jogo, um observador externo poderia questionar-se se tal jogador poderia aprender (e, consequentemente, ganhar) um jogo contra um adversário que estabelece regras "móveis" para as partidas.

As idéias apresentadas nesta seção mostram que os resultados obtidos neste trabalho podem servir de ponto de partida para vários novos estudos. Ainda não foi tratada a questão da possibilidade do uso do aprendizado comportalmente correto, definido na seção 3.2, e será necessário um estudo mais detalhado sobre as propriedades das famílias de problemas adaptativos confinados lineares que não envolvam somente as questões relacionadas com aprendizado.

6.3 Conclusão

Esta tese se caracterizou por ser um trabalho fortemente teórico. Entretanto, existe uma série de aplicações práticas para as quais os resultados aqui apresentados podem contribuir. É importante frisar que os resultados téoricos aqui apresentados foram todos publicados. Fora o prosseguimento das linhas de pesquisas

aqui apontadas. As aplicações da inferência adaptativa de segunda ordem podem alcançar diversas áreas técnicas, todas aquelas em que um processo de inferência atua com mecanismo de automatização e otimização de processos. São áreas que vão desde a modelagem médica até aplicações em sistemas de software corporativos, hardware adaptativo, *Cloud Computing*, *On-line learning*, computação autonomica (do inglês *Autonomic Computing*) e simulação.

REFERÊNCIAS BIBLIOGRÁFICAS

[Aho, Sethi e Ullman 1986]AHO, A. V.; SETHI, R.; ULLMAN, J. D. *Compilers: principles, techniques, and tools*. Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc., 1986. ISBN 0-201-10088-6.

[Angluin 1980]ANGLUIN, D. Inductive inference of formal languages from positive data. *Information and Control*, v. 45(2), p. 117–135, 1980.

[Angluin e Smith 1983]ANGLUIN, D.; SMITH, C. H. Inductive Inference: Theory and Methods. *ACM Comput. Surv.*, v. 15, n. 3, p. 237–269, 1983.

[Boden et al. 2004]BODEN, M. et al. Cost-efficient implementation of adaptive finite state machines. In: *Proceedings of the Digital System Design, EUROMICRO Systems*. Washington, DC, USA: IEEE Computer Society, 2004. (DSD '04), p. 144–151. ISBN 0-7695-2203-3. Disponível em: <http://dx.doi.org/10.1109/DSD.2004.35>.

[Boullier 1994]BOULLIER, P. *Dynamic Grammars and Semantic Analysis*. Paris, France, August 1994.

[Carmi 2002]CARMI, A. Adapser: An LALR(1) Adaptive Parser. In: *The Israeli Workshop on Programming Languages & Development Environments*. Haifa, Israel: [s.n.], 2002. (I, 1), p. 1–3.

[Case e Lynes 1982]CASE, J.; LYNES, C. Machine Inductive Inference and Language Identification. In: *Proceedings of the 9th Colloquium on Automata, Languages and Programming*. London, UK: Springer-Verlag, 1982. p. 107–115. ISBN 3-540-11576-5.

[Case e Moelius 2008]CASE, J.; MOELIUS, S. Optimal Language Learning. In: FREUND; THOMAS, Z. (Ed.). *Algorithmic Learning Theory*. [S.l.]: Springer Berlin / Heidelberg, 2008, (Lecture Notes in Computer Science, v. 5254). p. 419–433. ISBN 978-3-540-87986-2. 10.1007/978-3-540-87987-9$_3$4.

[Chaitin 2010]CHAITIN, G. *To a mathematical theory of evolution and biological creativity*. [S.l.], September 2010.

[Chater e Vitányi 2007]CHATER, N.; VITáNYI, P. "Ideal learning" of natural language: positive results about learning from positive evidence. *Journal of Mathematical Psychology*, Elsevier, v. 51, n. 3, p. 135–163, 2007. Disponível em: <http://eprints.pascal-network.org/archive/00002798/>.

[Damásio 1999]DAMÁSIO, A. O. *O Erro de Descartes: emoção, razão e o cérebro humano*. Trad. Dora Vicente e Georgina Segurado. São Paulo, SP, Brasil: Companhia das Letras, 1999.

[Davis, Shrobe e Szolovits 1993]DAVIS, R.; SHROBE, H.; SZOLOVITS, P. What Is a Knowledge Representation? *AI Magazine*, AAAI, Menlo Park, CA, USA, v. 14, n. 1, p. 17–33, 1993.

[Denecke e Wismath 2002]DENECKE, K.; WISMATH, S. L. *Universal Algebra and Applications in Theoretical Computer Science*. New York: CRC/C&H, 2002. ISBN 1584882549.

[Dowe 2011]DOWE, D. L. Handbook of the Philosophy of Science - (HPS Volume 7) Philosophy of Statistics. In: _____. [S.l.]: Elsevier, 2011. cap. MML, hybrid Bayesian network graphical models, statistical consistency, invariance and uniqueness, p. 901–982.

[Filho e Rocha 2011]FILHO, R. I. d. S.; ROCHA, R. L. d. A. d. International Conference on Adaptive and Natural Computing Algorithms, ICANNGA 2011, Ljubljana, Slovenia, April 14-16, 2011, Revised Papers. In: LNCS. Ljubljana, Slovenia: Springer-Verlag, 2011. (Lecture Notes in Computer Science, v. 6594), cap. Adaptive Finite Automaton: a New Algebraic Approach, p. 275–284.

[Gold 1967]GOLD, E. Language identification in the limit. *Information and Control*, v. 10, n. 5, p. 447–474, 1967.

[Higuera 2010]HIGUERA, C. de la. *Grammatical Inference: Learning Automata and Grammars*. New York, NY, USA: Cambridge University Press, 2010. ISBN 0521763169, 9780521763165.

[Hopcroft, Motwani e Ullman 2000]HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. *Introduction to automata theory, languages, and computation*. 2. ed. Boston, MA, USA: Addison-Wesley, 2000.

[Hume e Chittom 2004]HUME, D.; CHITTOM, T. *An Enquiry Concerning Human Understanding (Barnes & Noble Library of Essential Reading): And Selections from A Treatise of Human Nature*. New York, NY, USA: BARNES & NOBLE, 2004. (Barnes & Noble Library of Essential Reading). ISBN 9780760755921.

[Klein e Kutrib 2002]KLEIN, A.; KUTRIB, M. Self-assembling finite automata. In: *Proceedings of the 8th Annual International Conference on Computing and Combinatorics*. London, UK: Springer-Verlag, 2002. (COCOON '02), p. 310–319. ISBN 3-540-43996-X. Disponível em: <http://dl.acm.org/citation.cfm?id=646720.702403>.

[Laxminarayana e Nagaraja 2003]LAXMINARAYANA, J. A.; NAGARAJA, G. Inference of a Subclass of Context Free Grammars Using Positive Samples. In: *ECML Workshop on Learning Contex-Free Grammars*. Cavtat-Dubrovnik, Croatia: [s.n.], 2003. v. 1, p. 29–40.

[Lee 1996]LEE, L. *Learning of Context-Free Languages: A Survey of the Literature*. Cambridge, MA, 1996.

[Lewis e Papadimitriou 1997]LEWIS, H.; PAPADIMITRIOU, C. *Elements of the Theory of Computation*. New York: Prentice-Hall, 1997.

[Li e Vitányi 1995]LI, M.; VITáNYI, P. Computational Machine Learning in Theory and Praxis. 1995.

[Li e Vitányi 2008]LI, M.; VITáNYI, P. *An introduction to Kolmogorov complexity and its applications*. Third. [S.l.]: Springer Publishing Company, Incorporated, 2008.

[Mikolajczak 1991]MIKOLAJCZAK, B. (Ed.). *Algebraic and Structural Automata Theory*. Heidelberg: North-Holland, 1991.

[Mott, Kandel e Baker 1986]MOTT, J. L.; KANDEL, A.; BAKER, T. P. *Discrete mathematics for computer scientists & mathematicians (2nd ed.)*. Upper Saddle River, NJ, USA: Prentice-Hall, Inc., 1986. ISBN 0-8359-1391-0.

[Neto 2001]NETO, J. a. J. Solving complex problems efficiently with adaptive automata. In: *Revised Papers from the 5th International Conference on Implementation and Application of Automata*. London, UK: Springer-Verlag, 2001. (CIAA '00), p. 340–342. ISBN 3-540-42491-1. Disponível em: <http://dl.acm.org/citation.cfm?id=647267.721571>.

[Neto 2002]NETO, J. a. J. Adaptive rule-driven devices - general formulation and case study. In: *Revised Papers from the 6th International Conference on Implementation and Application of Automata*. London, UK: Springer-Verlag, 2002. (CIAA '01), p. 234–250. ISBN 3-540-00400-9. Disponível em: <http://dl.acm.org/citation.cfm?id=647268.721714>.

[Neto e Bravo 2003]NETO, J. J.; BRAVO, C. Adaptive automata: a revisited proposal. In: *Proceedings of the 7th international conference on Implementation and application of automata*. Berlin, Heidelberg: Springer-Verlag, 2003. (CIAA'02), p. 158–168. ISBN 3-540-40391-4. Disponível em: <http://dl.acm.org/citation.cfm?id=1756384.1756401>.

[Prajapati 2011]PRAJAPATI, G. L. Advances in learning formal languages. In: *Lecture Notes in Engineering and Computer Science*. Hong Kong: Newswood and International Association of Engineers, 2011. p. 118–126. ISSN 20780966.

[Prajapati, Chaudhari e Chandwani 2008]PRAJAPATI, G. L.; CHAUDHARI, N. S.; CHANDWANI, M. Efficient incremental model for learning context-free grammars from positive structural examples. In: *Proceedings of the 5th Hellenic conference on Artificial Intelligence: Theories, Models and Applications*. Berlin, Heidelberg: Springer-Verlag, 2008. (SETN '08), p. 250–262. ISBN 978-3-540-87880-3.

[Ramos, Neto e Vega 2009]RAMOS, M. V. M.; NETO, J. J.; VEGA, Í. S. *Linguagens Formais: Teoria, Modelagem e Implementação*. 1. ed. Porto Alegre: Bookman, 2009. 656p p. ISBN 978-85-7780-453-5.

[Rocha e Neto 2000]ROCHA, R. L. d. A. d.; NETO, J. a. J. Autômato adaptativo, limites e complexidade em comparação com máquina de turing. In: FACULDADE SENAC DE CIêNCIAS EXATAS E TECNOLOGIA. *Proceedings of the I Congress of Logic Applied to Technology - LAPTEC 2000*. São Paulo, 2000. p. 33–48.

[Rocha e Neto 2005]ROCHA, R. L. d. A. d.; NETO, J. a. J. An Adaptive Finite-State Automata Application to the Problem of Reducing the Number of States in Approximate String Matching. In: *XI Congreso Argentino de Ciencias de la Computación*. Concordia, Entre Ríos, Argentina: [s.n.], 2005. (CACIC 2005, 1), p. 17–21.

[Rocha 2007]ROCHA, R. L. de Azevedo da. An attempt to express the semantics of the adaptive devices. In: LAMBERT-TORRES, G. et al. (Ed.). *Advances in Technological Applications of Logical and Intelligent Systems, Selected Papers from the Sixth Congress on Logic Applied to Technology, LAPTEC 2007, Unisanta, Santa Cecilia University, Santos, Brazil, November 21-23, 2007*. [S.l.]: IOS Press, 2007. (Frontiers in Artificial Intelligence and Applications, v. 186), p. 13–27. ISBN 978-1-58603-936-3.

[Rocha 2011]ROCHA, R. L. de Azevedo da. *Adaptatividade*. São Paulo, SP, Brasil: Ed. Blucher, 2011. ISBN 9788580390124.

[Rubinstein e Shutt 1994]RUBINSTEIN, R. S.; SHUTT, J. N. Self-modifying finite automata. In: PEHRSON, B.; SIMON, I. (Ed.). *Proceedings of the 13th IFIP World Computer Congress*. Amsterdam: North-Holland, 1994. (Technology and Foundations: Information Processing '94, I), p. 493–498.

[Russel e Norvig 2003]RUSSEL, S.; NORVIG, P. *Artificial Intelligence – A Modern Approach*. [S.l.]: Prentice Hall, 2003.

[Russell e Norvig 2003]RUSSELL, S. J.; NORVIG, P. *Artificial Intelligence – A Modern Approach*. 2. ed. Englewood Cliffs, NJ, USA: Prentice Hall, 2003.

[Salthe e Matsuno 1995]SALTHE, S.; MATSUNO, K. Self-Organization in Hierarchical Systems. *Journal of Social and Evolutionary Systems*, v. 18, n. 4, p. 327–338, 1995.

[Shutt 1995]SHUTT, J. N. Self-Modifying Finite Automata: Power and Limitations. *Technical Report*, Worcester Polytechnic Institute, n. WPI-CS-TR-95-4, 1995.

[Sipser 1996]SIPSER, M. *Introduction to the Theory of Computation*. 1st. ed. [S.l.]: International Thomson Publishing, 1996. ISBN 053494728X.

[Solomonoff 1967]SOLOMONOFF, R. J. Inductive Inference Research Status, Spring 1967. Rockford Research, Inc., Cambridge, MA, USA, 1967.

[Solomonoff 1996]SOLOMONOFF, R. J. Does algorithmic probability solve the problem of induction? In: DOWE, D. L.; KORB, K. B.; OLIVER, J. J. (Ed.). *Proceedings of the Information, Statistics and Induction in Science (ISIS) Conference*. Melbourne, Australia: World Scientific, 1996. p. 7–8. ISBN ISBN 981-02-2824-4.

[Solomonoff 2007]SOLOMONOFF, R. J. Machine Learning – Past and Future. 2007. Disponível em: <http://world.std.com/ rjs/pubs.html>.

[Wallace 2004]WALLACE, C. S. *Statistical and Inductive Inference by Mininum Message Length*. [S.l.]: Springer, 2004.

[Yang 2004]YANG, C. D. Universal Grammar, statistics or both? *Trends in Cognitive Sciences*, v. 8, n. 10, p. 451–456, out. 2004.

[Zilles 2001]ZILLES, S. On the Synthesis of Strategies Identifying Recursive Functions. In: *Proceedings of the 14th Annual Conference on Computational Learning Theory and and 5th European Conference on Computational Learning Theory*. London, UK: Springer-Verlag, 2001. (COLT 01/EuroCOLT 01), p. 160–176. ISBN 3-540-42343-5.