



UNIVERSIDADE FEDERAL DO RIO GRANDE DO NORTE
CENTRO DE CIÊNCIAS EXATAS E DA TERRA
DEPARTAMENTO DE INFORMÁTICA E MATEMÁTICA APLICADA
PROGRAMA DE PÓS-GRADUAÇÃO EM SISTEMAS E COMPUTAÇÃO

Autômatos *Fuzzy* Hesitantes Típicos: Teoria e Aplicações

Valdigleis da Silva Costa

Natal, RN

08 de Dezembro de 2020.

Valdigleis da Silva Costa

Autômatos *Fuzzy* Hesitantes Típicos: Teoria e Aplicações

Esta tese foi submetida ao Programa de Pós-graduação em Sistemas e Computação da Universidade Federal do Rio Grande do Norte como requisito parcial, para obter o título de doutor em Ciência da Computação.

Linha de Pesquisa:

Fundamentos da Computação.

Orientador:

Dr. Benjamín René Callejas Bedregal.

Coorientador:

Dr. Regivan Hugo Nunes Santiago.

Universidade Federal do Rio Grande do Norte

Centro de Ciências Exatas e da Terra

Departamento de Informática e Matemática Aplicada

Programa de Pós-graduação em Sistemas e Computação

Natal, RN

08 de Dezembro de 2020.

Universidade Federal do Rio Grande do Norte - UFRN
Sistema de Bibliotecas - SISBI
Catalogação de Publicação na Fonte. UFRN - Biblioteca Setorial Prof. Ronaldo Xavier de Arruda - CCET

Costa, Valdigleis da Silva.

Autômatos fuzzy hesitantes típicos: teoria e aplicações /
Valdigleis da Silva Costa. - 2020.
178f.: il.

Tese (Doutorado) - Universidade Federal do Rio Grande do
Norte, Centro de Ciências Exatas e da Terra, Departamento de
Informática e Matemática Aplicada, Programa de Pós-graduação em
Sistemas e Computação. Natal, 2020.

Orientador: Benjamín René Callejas Bedregal.

Coorientador: Regivan Hugo Nunes Santiago.

1. Computação - Tese. 2. Autômatos fuzzy - Tese. 3. Autômatos
fuzzy hesitantes típicos - Tese. 4. Computação com incerteza -
Tese. 5. Linguagens fuzzy hesitantes típicas - Tese. I.
Bedregal, Benjamín René Callejas. II. Santiago, Regivan Hugo
Nunes. III. Título.

RN/UF/CCET

CDU 004

VALDIGLEIS DA SILVA COSTA

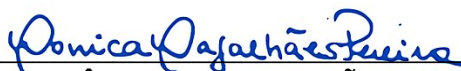
“Autômatos Fuzzy Hesitantes Típicos: Teoria e Aplicações”

Esta Tese foi julgada adequada para a obtenção do título de Doutor em Ciência da Computação e aprovada em sua forma final pelo Programa de Pós-Graduação em Sistemas e Computação do Departamento de Informática e Matemática Aplicada da Universidade Federal do Rio Grande do Norte.



Dr. BENJAMÍN RENÉ CALLEJAS BEDREGAL

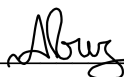
Presidente – Orientadora



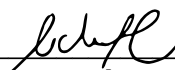
Prof.ª Dr.ª MÔNICA MAGALHÃES PEREIRA

Vice-Coordenadora do PPgSC

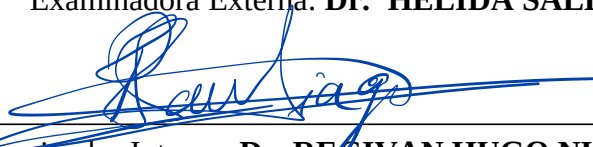
Banca Examinadora



Examinador Externo: **Dr. ANDERSON PAIVA CRUZ**



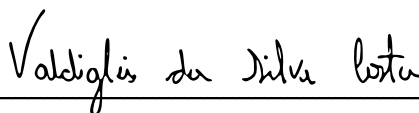
Examinadora Externa: **Dr.ª HÉLIDA SALLES SANTOS**



Examinador Interno: **Dr. REGIVAN HUGO NUNES SANTIAGO**



Examinadora Externa: **Dr.ª RENATA HAX SANDER REISER**



Discente: **VALDIGLEIS DA SILVA COSTA**

Dezembro, 2020

Este trabalho é dedicado, in memoriam, como todo meu amor e carinho a minha vó, e também para meu primo e grande amigo Yuri.

Agradecimentos

Aos meus pais, José Rodrigues da Costa Filho e Maria de Fátima da Silva Costa, por todo seu amor, carinho e dedicação. À minha amada vó, Rita Gomes (*in memoriam*) por sempre ter sido um poço de carinho e cuidado.

Ao meu orientador, Prof. Dr. Benjamín René Callejas Bedregal e ao meu coorientador Prof. Dr. Regivan Hugo Nunes Santiago, por me orientar e pela incomensurável contribuição para o meu desenvolvimento intelectual e amadurecimento social, vocês são duas pessoas diferentes mas igualmente únicas, e para mim representam o tipo de pessoa e de profissional que almejo ser.

Às professoras, Dra. Renata Reiser e Dra. Héliida Salles, e ao Prof. Dr. Anderson Paiva Cruz, por terem aceitado o convite para compor a banca da defesa deste trabalho e pelas maravilhosas observações e apontamentos feitos. Com respeito ao Prof. Anderson, também agradeço de todo coração por ter me iniciado neste maravilhoso universo recursivamente enumerável que é a computação formal.

Aos que fazem parte do grupo de pesquisa em Lógica, Linguagem, Informação, Teoria e Aplicações - LoLITA, especialmente à Annaxsuel, Rui, Vânia, Thadeu, Thiago, Huliane (que não sabe fazer café), Suene, Nicolas, Diego, André e Emanuelle, pelas boas risadas, cafés e pelo companheirismo durante esses últimos anos.

Por fim, a todos os funcionários do Departamento de Informática e Matemática Aplicada - DIMAp/UFRN, em especial ao Sr. Gaspar por tantas histórias e reflexões sobre a vida e a Daniel que sempre que preciso me ajudou junto as demandas da secretaria da pós-graduação. Finalizo agradecendo aos professores do Programa de Pós-Graduação em Sistemas e Computação - PPgSC/UFRN, por todo conhecimento que comigo foi compartilhado.

A Matemática existiu e existe desde antes da Criação. Ela é co-eterna com a mente de Deus. A Matemática forneceu a Deus o modelo da Criação. No fim, a Matemática é o próprio Deus.

Adaptado da famosa citação de
Johannes Kepler.

Resumo

Os autômatos *fuzzy* são uma generalização dos autômatos finitos, que podem modelar computação com incerteza. Nos últimos anos, devido às extensões dos conjuntos *fuzzy*, diferentes generalizações de autômatos *fuzzy* tais como autômatos *fuzzy* intervalares e autômatos *fuzzy* intuicionistas tem sido apresentadas. Os autômatos *fuzzy* são uma parte fundamental da teoria da computação *fuzzy*. Além disso, eles também têm sucesso em aplicações práticas, principalmente no reconhecimento de padrões. Este trabalho apresenta uma nova generalização de autômatos *fuzzy* baseada nas definições de conjuntos *fuzzy* hesitantes típicos, a ideia central por trás dessa generalização é trazer para o campo da computação a possibilidade de trabalhar simultaneamente com a incerteza e a hesitação. Assim, esta nova generalização visa possibilitar novas formas de enfrentar problemas que antes não eram facilmente modelados usando apenas incertezas. Além disso, este trabalho mostra maneiras de aplicar esses novos autômatos em problemas de classificação e processamento de imagens.

Palavras-chaves: Autômatos *Fuzzy*; Autômatos *Fuzzy* Hesitantes Típicos; Computação com incerteza; Linguagens *Fuzzy* Hesitantes Típicas.

Abstract

The fuzzy automata are a generalization of finite automata, which can model computation with uncertainty. Due to the fuzzy sets extensions, different generalizations of fuzzy automata such as interval-valued fuzzy automata and intuitionist fuzzy automata have been presented. Fuzzy automata are a fundamental part of the fuzzy computation theory. Moreover, they also have success in practical applications, mainly in pattern recognition. This work introduces a new generalization of fuzzy automata based on the definitions of typical hesitant fuzzy sets. The central idea behind this generalization is to bring to the field of computing the possibility of working simultaneously with uncertainty and hesitation. Thus, this new generalization aims to enable new ways to face problems that were not easily modeled before used only uncertainties. Besides, this work shows ways to apply these new automata in classification problems and image processing.

Key-words: Fuzzy Automata; Typical Hesitant Fuzzy Automata; Computing with uncertainty; Typical Hesitant Fuzzy Languages.

Lista de ilustrações

Figura 1 – Uma representação informal de autômato finito baseado na Figura 3 de (COSTA, 2016).	34
Figura 2 – Representação visual do AFD no Exemplo 2.1.1.	36
Figura 3 – Representação visual do AFN no Exemplo 2.1.3.	37
Figura 4 – Representação da máquina de estados <i>fuzzy</i> do Exemplo 3.2.1.	53
Figura 5 – Exemplo de um autômato <i>fuzzy</i> .	56
Figura 6 – Representação de um autômato <i>fuzzy</i> simplificado.	58
Figura 7 – Hierarquia entre as linguagens aceitas por autômatos <i>fuzzy</i> e suas extensões.	64
Figura 8 – Gráfico para a interpretação operacional da inf-combinação como um filtro passa baixa.	70
Figura 9 – Gráfico para a interpretação operacional da sup-combinação como um filtro passa alta.	73
Figura 10 – Representação de um AFHT.	79
Figura 11 – Um AFHT sobre o alfabeto $\{x, y\}$.	82
Figura 12 – AFHTD criado a partir do AFHT da Figura 10, pela aplicação dos Teoremas 4.2.1 e 4.2.2.	87
Figura 13 – Um AFHT sobre o alfabeto unitário $\{a\}$.	89
Figura 14 – Família de autômatos <i>fuzzy</i> gerada pela aplicação do Teorema 4.2.3 sobre o AFHT da Figura 13.	90
Figura 15 – Dois AFHT's sobre o alfabeto $\{a, b\}$.	93
Figura 16 – AFHT construído a partir dos AFHT's das Figuras 15a e 15b, seguindo os passos descritos na prova do Teorema 4.3.1.	94
Figura 17 – Hierarquia entre os conjuntos $\mathbb{T}$ e $\mathbb{T}_R$.	97
Figura 18 – Extensão da hierarquia de linguagens da Figura 7, considerando as LFHT $\mathbb{H}$ -regulares.	97
Figura 19 – Imagem dividida em quatro quadrantes indexados pelos elementos de Σ .	108
Figura 20 – AFD gerado aplicando o Algoritmo 2 sobre a imagem $\mathcal{I}$ do Exemplo 5.1.1.	111
Figura 21 – Representação visual do AFD gerado para o conjunto $\mathcal{W}_{X_d^n}$ do Exemplo 6.1.4.	123
Figura 22 – Processamento da imagem Img1 com proporção de 0.05 de sal.	149
Figura 23 – Processamento da imagem Img2 com proporção de 0.05 de sal.	149
Figura 24 – Processamento da imagem Img3 com proporção de 0.05 de sal.	150
Figura 25 – Processamento da imagem Img4 com proporção de 0.05 de sal.	150
Figura 26 – Processamento da imagem Img5 com proporção de 0.05 de sal.	150
Figura 27 – Processamento da imagem Img6 com proporção de 0.05 de sal.	151

Figura 28 – Processamento da imagem Img1 com proporção de 0.10 de sal.	151
Figura 29 – Processamento da imagem Img2 com proporção de 0.10 de sal.	151
Figura 30 – Processamento da imagem Img3 com proporção de 0.10 de sal.	152
Figura 31 – Processamento da imagem Img4 com proporção de 0.10 de sal.	152
Figura 32 – Processamento da imagem Img5 com proporção de 0.10 de sal.	152
Figura 33 – Processamento da imagem Img6 com proporção de 0.10 de sal.	153
Figura 34 – Processamento da imagem Img1 com proporção de 0.15 de sal.	153
Figura 35 – Processamento da imagem Img2 com proporção de 0.15 de sal.	153
Figura 36 – Processamento da imagem Img3 com proporção de 0.15 de sal.	154
Figura 37 – Processamento da imagem Img4 com proporção de 0.15 de sal.	154
Figura 38 – Processamento da imagem Img5 com proporção de 0.15 de sal.	154
Figura 39 – Processamento da imagem Img6 com proporção de 0.15 de sal.	155
Figura 40 – Processamento da imagem Img1 com proporção de 0.20 de sal.	155
Figura 41 – Processamento da imagem Img2 com proporção de 0.20 de sal.	155
Figura 42 – Processamento da imagem Img3 com proporção de 0.20 de sal.	156
Figura 43 – Processamento da imagem Img4 com proporção de 0.20 de sal.	156
Figura 44 – Processamento da imagem Img5 com proporção de 0.20 de sal.	156
Figura 45 – Processamento da imagem Img6 com proporção de 0.20 de sal.	157
Figura 46 – Processamento da imagem Img1 com proporção de 0.25 de sal.	157
Figura 47 – Processamento da imagem Img2 com proporção de 0.25 de sal.	157
Figura 48 – Processamento da imagem Img3 com proporção de 0.25 de sal.	158
Figura 49 – Processamento da imagem Img4 com proporção de 0.25 de sal.	158
Figura 50 – Processamento da imagem Img5 com proporção de 0.25 de sal.	158
Figura 51 – Processamento da imagem Img6 com proporção de 0.25 de sal.	159

Lista de tabelas

Tabela 1	–	Agregações de cada quadrante $\xi(q)$ tal que $q \in \{q_0, q_1, q_2, q_3, q_4\}$.	115
Tabela 2	–	Resultados das similaridades entre as agregações de cada quadrante $\xi(q)$ tal que $q \in \{q_1, q_2, q_3, q_4\}$ e o quadrante $\xi(q_0)$.	115
Tabela 3	–	Resultados do cenário com proporção de 0.05 de ruído do tipo sal nas imagens.	119
Tabela 4	–	Resultados do cenário com proporção de 0.10 de ruído do tipo sal nas imagens.	119
Tabela 5	–	Resultados do cenário com proporção de 0.15 de ruído do tipo sal nas imagens.	119
Tabela 6	–	Resultados do cenário com proporção de 0.20 de ruído do tipo sal nas imagens.	119
Tabela 7	–	Resultados do cenário com proporção de 0.25 de ruído do tipo sal nas imagens.	119
Tabela 8	–	Base de dados baseada da Tabela 2.1 de (FACELI et al., 2011).	121
Tabela 9	–	Configurações usada do Algoritmo HeCA.	129
Tabela 10	–	Base de dados usadas nos experimentos.	129
Tabela 11	–	Acurácia média para as configurações do Algoritmo HeCA.	130
Tabela 12	–	média da métrica F1-measure para as configurações do Algoritmo HeCA.	130
Tabela 13	–	Média dos resultados para os algoritmo KNN e árvore de decisão.	130
Tabela 14	–	Valores de p e t do teste estatístico para a acurácia.	130
Tabela 15	–	Valores de p e t do teste estatístico para a $f1-measure$.	131

Lista de abreviaturas e siglas

AFD – Autômato Finito Determinístico.

AFN – Autômato Finito Não-determinístico.

SPO – Sistema parcialmente ordenado.

MMC – Mínimo Múltiplo Comum.

MDC – Máximo Divisor Comum.

CFH – Conjunto *Fuzzy* Hesitante.

CFHT – Conjunto *Fuzzy* Hesitante Típico.

EFHT – Elemento *Fuzzy* Hesitante Típico.

AFI – Autômato *Fuzzy* Intervalar.

AFN-EFI – Autômato Finitos Não-determinístico com Estados *Fuzzy* Intervalar.

AFMC – Autômato *Fuzzy* baseado em Multiconjunto.

AFN-TFI – Autômato Finito Não-determinístico como Transições *Fuzzy* Intuicionistas.

AFHT – Autômato *Fuzzy* Hesitante Típico.

AFHTD – Autômato *Fuzzy* Hesitante Típico Determinístico.

LFHT – Linguagem *Fuzzy* Hesitante Típica.

PAFHT – Pseudo Autômato *Fuzzy* Hesitante Típico.

PAFHTD – Pseudo Autômato *Fuzzy* Hesitante Típico Determinístico.

ReSPA – Algoritmo de remoção de ruído do tipo *sal* (salt) baseado em PAFHTD.

HeCA – Hesitant classification automaton.

KNN – k-nearest neighbors.

PSNR – Sigla em inglês para *peak signal to noise ratio*.

RNMSE – Sigla em inglês para *root normalized mean square error*.

MSE – Sigla em inglês para *mean squared error*.

RMSE – Sigla em inglês para *root mean square error*.

Lista de símbolos

Σ – Alfabeto.

Σ^* – Monoide livremente gerado por Σ .

λ – Palavra vazia.

$|w|$ – Tamanho da palavra w .

$\langle Q, \Sigma, \delta, q_0, F \rangle$ – Estrutura de um AFD A .

Q – Conjunto finito de estados.

δ – Função de transição.

$q_0 \in Q$ – Estado inicial.

$F \subseteq Q$ – Conjunto de estados finais.

$\widehat{\delta}$ – Extensão da função δ .

$L(A)$ – Linguagem aceita por um AFD (ou AFN) A .

L_{Reg} – Família das linguagens regulares.

$\langle Q, \Sigma, \delta_N, q_0, F \rangle$ – Estrutura de um AFN A .

δ_N – Função de transição para um AFN N .

$\widehat{\delta_N}$ – Extensão da função δ_N .

$\langle P, \preceq \rangle$ – SPO na notação algébrica.

$\mathbb{N}$ – O conjunto dos números naturais.

$\perp$ – Mínimo de um SPO.

$\top$ – Máximo de um SPO.

$\vee \mathbb{A}$ – Supremo do conjunto $\mathbb{A}$.

$\wedge \mathbb{A}$ – Ínfimo do conjunto $\mathbb{A}$.

$\langle \mathcal{L}, \vee, \wedge \rangle$ – Estrutura algébrica de um reticulado.

$\langle \mathcal{L}, \vee, \wedge, \perp, \top \rangle$ – Estrutura algébrica de um reticulado limitado.

$\mathbb{R}$ – Conjunto dos números reais.

$x \in A$ – x pertence ao conjunto A .

$x \notin A$ – x não pertence ao conjunto A .

$\mathbb{U}$ – Conjunto universo.

$\mu(x)$ – valor de pertinência do elemento x pertencer ao conjunto *fuzzy* μ .

$\tau(x)$ – Conjunto de valores de pertinência do elemento x pertence ao conjunto τ .

τ_Ω – CFH associado com o conjunto Ω .

$\mathbb{H}$ – Conjunto dos elementos *fuzzy* hesitantes.

$\#X$ – Cardinalidade do conjunto X .

$\mathbb{H}_1$ – Conjunto de todos os degenerados de $\mathbb{H}$.

$\mathbb{L}_1 + \mathbb{L}_2$ – União *fuzzy* de $\mathbb{L}_1$ e $\mathbb{L}_2$.

$\mathbb{L}_1 \cdot \mathbb{L}_2$ – Interseção *fuzzy* de $\mathbb{L}_1$ e $\mathbb{L}_2$.

$\mathbb{L}_1 \times \mathbb{L}_2$ – Produto Cartesiano das linguagens *fuzzy* $\mathbb{L}_1$ e $\mathbb{L}_2$.

$\mathbb{L}_1 \mathbb{L}_2^{-1}$ – Quociente à direita de $\mathbb{L}_1$ por $\mathbb{L}_2$.

$\mathbb{L}_2^{-1} \mathbb{L}_1$ – Quociente à esquerda de $\mathbb{L}_1$ por $\mathbb{L}_2$.

$\langle Q, \Sigma, \varphi \rangle$ – Estrutura de uma máquina de estado *fuzzy* M .

$\langle Q, \Sigma, \varphi, \mathbb{I}, \mathbb{F} \rangle$ – Estrutura de um autômato *fuzzy* M .

f_M – Linguagem *fuzzy* de um autômato *fuzzy* M .

$\langle Q, \Sigma, \varphi, q_0, \mathbb{F} \rangle$ – Estrutura de um autômato *fuzzy* simplificado M .

$Ling_F$ – Família das linguagens aceitas por autômatos *fuzzy*.

$\langle Q, \Sigma, \vartheta, S, F \rangle$ Estrutura de um AFI m .

$\langle Q, \delta_N, \vartheta, S, \tilde{F} \rangle$ Estrutura de um AFN-EFI m .

$Ling_{FI}$ – Família das linguagens aceitas por autômatos *fuzzy* intervalares.

$\langle Q, \overline{\Sigma}^\oplus, \beta, Q_I, Q_F \rangle$ – estrutura de um ACMF $\mathbb{M}$.

$\overline{\Sigma}^\oplus$ – Família de todos os multiconjuntos unitários sobre Σ .

$\Sigma^\oplus$ – Família de todos os multiconjuntos sobre Σ .

$Ling_{MF}$ – Família de linguagens aceitas por autômatos *fuzzy* baseados em multiconjuntos.

$\langle Q, \Sigma, \mathcal{T}, T, q_0, F \rangle$ – Estrutura de um AFN-TFI m .

$Ling_{FI n}$ – Família das linguagens aceitas por autômatos *fuzzy* intuicionistas.

$\langle \mathcal{L}, \vee, \wedge, \odot, \star \rangle$ – Estrutura algébrica de um reticulado completo residuado $\mathcal{L}$.

$Ling_{\mathcal{L}}$ – Família das linguagens aceitas por autômatos $\mathcal{L}$ -valorados.

$\preceq_{\mathbb{H}}$ – Ordem parcial sobre $\mathbb{H}$ baseada na ordem de inclusão.

$X \sqsubseteq Y$ – Ordem CBS sobre o conjunto $\mathbb{H}$ baseada na sup-combinação.

$\langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$ – Estrutura de um AFHT $\mathcal{M}$.

$\hat{f}_{\mathcal{M}}$ – Linguagem aceita por um AFHT $\mathcal{M}$.

$\mathcal{M} \sqsubseteq \mathcal{M}'$ – Relação de confiabilidade entre dois AFHT's.

$SUC_{\mathcal{M}}(q, a)$ – Sucessores do estado q com respeito ao símbolo a no AFHT $\mathcal{M}$.

π – Conjunto de todos os estados nulos em um AFHT.

$\mathbb{T}$ – Conjunto de todas as LFHT's.

$\mathbb{T}_R$ – Conjunto de todas as LFHT's que são $\mathbb{H}$ -regulares.

$\hat{f}_1 \uplus \hat{f}_2$ – $\mathbb{H}$ -união de $\hat{f}_1$ e $\hat{f}_2$.

$\mathcal{N} = \langle Q, \Sigma, \delta_{\mathcal{N}}, q_0, \mathcal{F} \rangle$ – Estrutura de um PAFHT $\mathcal{N}$.

$\hat{f}_1 \nabla \hat{f}_2$ – $\mathbb{H}$ -interseção de $\hat{f}_1$ e $\hat{f}_2$.

Σ^n – Conjunto das palavras sobre Σ com tamanho igual a n .

Pre_w – conjunto de todos os prefixos de uma palavra w

$\mathcal{E}_n$ – Enumeração do conjunto Σ^n .

ξ – Enumeração dos quadrantes de uma imagens.

$CP_{\mathcal{M}}$ – Computação dos prefixos em um PAFHTD $\mathcal{M}$.

$y \star X$ – Multiplicação por escalar de y por X .

$Mean(X)$ – Média aritmética de X .

X_d^n – Uma base de dados com n elementos e d atributos.

x_j^i – j -ésimo atributo para o i -ésimo elemento na base.

$Lis_{X_d^n}$ – Lista de todas as palavras que descrevem os elementos de X_d^n .

$Lis_{X_d^n}^y$ – Lista de todas as palavras sobre X_d^n com $x_d^i = y$.

$\mathcal{W}_{X_d^n}$ – Conjunto das palavras da base de dados X_d^n .

$\mathcal{W}_{X_d^n}^y$ – Conjunto das y -palavras da base de dados X_d^n .

$\bigoplus_{x \in \Sigma}$ – Somatório indexado por um alfabeto Σ .

$\langle \Sigma^{\oplus} \rangle$ – Conjunto de todas as listas ordenadas de forma crescente sobre $\Sigma^{\oplus}$.

$\mathbb{N}_G$ – Conjunto de todos os números de Gödel.

$FS(\Sigma^{\oplus})$ – família de todos os conjuntos *fuzzy* sobre $\Sigma^{\oplus}$.

$FS(\Sigma^*)$ – família de todos os conjuntos *fuzzy* sobre Σ^* .

$Mat(A, m, n)$ – Conjunto de todas as matrizes de elementos de A com dimensão m por n .

$dim(\mathcal{I})$ – Dimensão da imagem $\mathcal{I}$.

$\mathcal{Q}$ – Quadrante em uma imagem $\mathcal{I}$.

$\theta_{\mathcal{I}}$ – Conjunto de todos os quadrantes da imagem $\mathcal{I}$.

$\lll$ – Relação de ordem parcial definida sobre o conjunto $\theta_{\mathcal{I}}$.

$MAX_{\mathcal{I}}$ – valor máximo da intensidade que pode ocorrer nos *pixels* de uma imagem $\mathcal{I}$.

S_n – Sinal discreto de dimensão n .

$S_{n,k}$ – k -ésimo elemento do sinal S_n .

$\mathbb{S}_{2^j}$ – Conjunto de todos os sinais de dimensão 2^j .

Sumário

1	INTRODUÇÃO	27
1.1	Objetivos	29
1.2	Organização do Texto	29
I	FUNDAMENTOS	31
2	CONCEITOS FUNDAMENTAIS	33
2.1	Autômatos Finitos	33
2.2	Ordens Parciais e Reticulados	39
2.3	Conjuntos <i>Fuzzy</i>	42
2.4	Conjuntos <i>Fuzzy</i> Hesitantes e Hesitantes Típicos	45
2.5	Considerações Finais	47
3	AUTÔMATOS <i>FUZZY</i> ESTADO ATUAL	49
3.1	Linguagens <i>Fuzzy</i>	49
3.2	Autômatos <i>Fuzzy</i>	50
3.3	Extensões da Teoria de Autômatos <i>Fuzzy</i> e uma Hierarquia de Lin- guagens	59
3.4	Aplicações de Autômatos <i>Fuzzy</i>	63
3.5	Considerações Finais	65
II	CONTRIBUIÇÕES	67
4	UMA TEORIA DE AUTÔMATOS <i>FUZZY</i> HESITANTES TÍPICOS	69
4.1	Combinações e Ordem sobre EFHT's	69
4.2	Autômatos <i>Fuzzy</i> Hesitantes Típicos	78
4.3	Caracterização das Linguagens $\mathbb{H}$ -Regulares.	91
4.4	Pseudo Autômatos <i>Fuzzy</i> Hesitante Típicos	98
4.5	Considerações finais	104
5	RUÍDO TIPO SAL E PSEUDO AUTÔMATOS <i>FUZZY</i> HESITAN- TES TÍPICOS	107
5.1	Imagens e PAFHTD's	108
5.2	O Algoritmo ReSPA	115
5.3	Resultados Experimentais	118

5.4	Considerações Finais	120
6	CLASSIFICAÇÃO DE DADOS E PSEUDO AUTÔMATOS <i>FUZZY</i> HESITANTES TÍPICOS	121
6.1	Transformando a Base de Exemplos em um PAFHTD	121
6.2	Generalizando os PAFHTD	125
6.3	O Algoritmo HeCA	127
6.4	Análise experimental	129
6.5	Considerações Finais	131
7	CONCLUSÃO	133
7.1	Resultados Teóricos	133
7.2	Resultados Práticos	134
7.3	Trabalhos Futuros	135

APÊNDICES 137

APÊNDICE A – O ISOMORFISMO ENTRE OS CONJUNTOS *FUZZY* SOBRE $\Sigma^{\oplus}$ E SOBRE Σ^* 139

A.1	Um pouco de Teoria dos Multiconjuntos	139
A.2	A Bijeção entre $\Sigma^{\oplus}$ e Σ^*	140
A.3	O Isomorfismo $Map_{\mathcal{Z}}$	142

APÊNDICE B – FORMALIZAÇÕES PARA O PROCESSAMENTO DIGITAL DE IMAGENS. 145

APÊNDICE C – IMAGENS PRODUZIDAS PELOS EXPERIMEN- TOS DO CAPÍTULO 5 149

APÊNDICE D – UMA FORMULAÇÃO ALGORÍTMICA DA TRANS- FORMADA *WAVELET* DISCRETA DE HAAR . . 161

REFERÊNCIAS 165

1 Introdução

Zadeh em seu trabalho seminal (ZADEH, 1965) introduziu a teoria dos conjuntos *fuzzy* como uma generalização da teoria clássica de conjuntos (COSTA, 2016). Com sua teoria, Zadeh apresentou de forma rigorosa uma ferramenta matemática para se trabalhar com incertezas.

Com o passar das décadas diversas generalizações para a teoria dos conjuntos *fuzzy* tem sido apresentadas (BUSTINCE et al., 2016). Entre essas generalizações encontradas na literatura estão: os conjuntos *fuzzy* intuicionistas de Atanassov (ATANASSOV, 1994), os conjuntos *fuzzy* tipo 2 (MENDEL; JOHN, 2002), muti-conjuntos *fuzzy* (MIYAMOTO, 2000), conjuntos *fuzzy* intervalares (SAMBUC, 1975), os conjuntos *fuzzy* n -dimensionais (SHANG; YUAN; LEE, 2010; BEDREGAL et al., 2012), os conjuntos *fuzzy* multi-dimensionais (LIMA et al., 2020), os conjuntos *fuzzy* hesitantes (TORRA; NARUKAWA, 2009; TORRA, 2010) e os conjuntos *fuzzy* hesitantes típicos (BEDREGAL et al., 2014a).

Cada uma dessas generalizações tenta capturar nuances a respeito de incertezas e (ou) imprecisões que não são naturalmente capturadas pela teoria de Zadeh. Em especial, os conjuntos *fuzzy* hesitantes (e hesitantes típicos) como dito por Torra (TORRA; NARUKAWA, 2009; TORRA, 2010), tentam capturar a ideia de hesitação na atribuição dos valores de pertinência com que um elemento pertence a algum conjunto (em uma teoria conjuntista) ou a hesitação de definir o nível de pertinência de um evento acontecer na modelagem de algum problema. Vale salientar que, como dito em (RODRÍGUEZ et al., 2016), os conjuntos *fuzzy* hesitantes são conceitualmente análogos à definição dos *Set-Valued Fuzzy Sets* apresentada em (BEDREGAL et al., 2012).

Em anos recentes tem aumentado o interesse nos modelos teóricos de computação com incerteza, isto é, os modelos de computação *fuzzy* (SYROPOULOS, 2014; ZADEH, 1996), computação probabilística (PAZ, 2014) e computação quântica (NIELSEN; CHUANG, 2005). A explicação para esse interesse crescente nos modelos de computação com incerteza se dá como consequência da promessa de extrapolar a tese de Church (FARIAS et al., 2016c), para o caso da computação *fuzzy* esta extrapolação foi comprovada pelo trabalho de Wiedermann (WIEDERMANN, 2004). Em relação a teoria da computação *fuzzy*, o seu ramo mais desenvolvido atualmente é o estudo dos autômatos *fuzzy* (MORDESON; MALIK, 2002), tais autômatos foram inicialmente definidos por Wee em sua tese de doutorado (WEE, 1967).

Como dito em (COSTA; BEDREGAL, 2020), um autômato *fuzzy* pode ser visto como um computador que é capaz de tolerar erros durante seu funcionamento, ou também

como um programa em que cada instrução tem associado a si um valor de pertinência de ser executada de forma precisa.

Em autômatos *fuzzy* dizer que uma instrução é executada de forma precisa é o mesmo que dizer que, ela será executada em sua totalidade como foi escrita, assim se existe uma margem de incerteza com respeito a uma instrução, significa que partes da instrução podem não ser executadas, ou que a instrução apresentará um comportamento parcialmente diferente das ações que ela descreve.

Desde que foram introduzidos por Wee (WEE, 1967), os autômatos *fuzzy* atraem muita atenção por suas grandes possibilidades de uso em aplicações práticas, nas área de monitoramento inteligente (STEIMANN; ADLASSNIG, 1994), telecomunicações e sistemas tolerantes a falhas (RIGATOS, 2009), reconhecimento de padrões (KO; HAM; NAM, 2011), problemas de busca (GIRIJAMMA, A., 2010), e etc, mostrando assim o valor da teoria dos autômatos *fuzzy* como campo que fornece poderosas ferramentas para aplicações fora do escopo teórico.

Como dito em (COSTA; BEDREGAL, 2020), a questão de definir os níveis de pertinência em um autômato (ou programa) *fuzzy* é uma grande problemática para aplicações práticas, pois tais valores tem que ser definidos *a priori*, muitas vezes antes mesmo de se conhecer todas as características e relações existentes no problema que se esta modelando.

Note que, quando um autômato *fuzzy* está sendo modelado (ou definido) existem no mínimo duas incertezas que deveriam ser tratadas pela definição do autômato *fuzzy*.

- (1) As incertezas de que as instruções serão executadas de forma precisa.
- (2) A hesitação de se escolher um único valor exato de pertinência para as instruções.

Entretanto, todas as definições padrão de autômatos *fuzzy* (WEE, 1967; SANTOS; WEE, 1968; CHAUDHARI; KOMEJWAR, 2011), só levam em consideração em sua definição formal o item (1), como explicado em (COSTA; BEDREGAL, 2020), ficando o item (2) negligenciado.

Motivado por essa questão de modelar a incerteza e a hesitação simultaneamente, este trabalho apresenta uma nova extensão da teoria dos autômatos *fuzzy*. Ela é baseada nos conjuntos *fuzzy* hesitantes típicos e será nomeada como teoria dos autômatos *fuzzy* hesitantes típicos¹.

A ideia é substituir os valores de pertinência exatos das instruções serem executadas por subconjuntos finitos e não vazios do intervalo $[0,1]$, esses conjuntos são chamados

¹ A escolha deste nome foi feita para manter conformidade com a nomenclatura encontrada na atual teoria dos autômatos *fuzzy*.

elementos *fuzzy* hesitantes típicos (RODRÍGUEZ et al., 2014). Tais conjuntos generalizam a ideia de valor de pertinência dos conjuntos *fuzzy* convencionais, e portanto, esse novo tipo de autômato modela o item (1), através do princípio da generalização (RODRÍGUEZ et al., 2016). Além disso, a cardinalidade do elemento *fuzzy* hesitante típico ligada a uma instrução pode ser interpretada como sendo o nível de hesitação para se definir um único valor exato de pertinência para tal instrução, ou seja, o item (2) não é mais negligenciado.

Dessa forma os autômatos *fuzzy* hesitantes típicos são modelos de máquinas capazes de lidar com a incertezas apresentadas pelas instruções *fuzzy* (ZADEH, 1996), e que também possuem a capacidade de modelar a hesitação dos processos envolvidos na escolha dos valores de pertinência das instruções.

1.1 Objetivos

O principal objetivo deste trabalho é apresentar e discutir uma nova extensão da teoria dos autômatos *fuzzy* baseada na teoria dos conjuntos *fuzzy* hesitantes típicos. Dessa forma, tem-se como objetivos específicos:

- Apresentar uma formalização de uma estrutura algébrica para o conjunto dos elementos *fuzzy* hesitantes típicos, que sirva como espaço de valoração para os autômatos *fuzzy* hesitantes típicos.
- Apresentar os autômatos *fuzzy* hesitantes típicos e as noções ligadas à computação *fuzzy* hesitante típica.
- Investigar a diferença entre níveis de poder de computação para os autômatos *fuzzy* hesitantes típicos e os autômatos *fuzzy*.
- Caracterizar a classe de linguagens *fuzzy* hesitantes típicas que são computadas por autômatos *fuzzy* hesitantes típicos.
- Apresentar aplicações dos autômatos *fuzzy* hesitantes típicos nas áreas de processamento digital de imagens e classificação de dados.

1.2 Organização do Texto

Para atingir os objetivos apresentados anteriormente, este trabalho se encontra organizado nos seguintes capítulos:

Capítulo 1. Esta Introdução.

Capítulo 2. Conceitos Fundamentais: apresentando as noções básicas e fundamentais para o desenvolvimento do trabalho, o mesmo sendo subdividido nas seguintes seções: Autômatos Finitos, Ordens Parciais e Reticulados, Conjuntos *Fuzzy*, Conjuntos *Fuzzy* Hesitantes e Hesitantes Típicos.

Capítulo 3. Autômatos *Fuzzy* Estado Atual: neste capítulo é apresentado a noção de autômato *fuzzy* que será adotada como padrão para este trabalho, também são apresentados algumas generalizações encontradas dos mesmos no atual estado da arte da área. Por fim, são apresentados aplicações dos autômatos *fuzzy*.

Capítulo 4. Uma Teoria de Autômatos *Fuzzy* Hesitantes: neste capítulo é apresentada a primeira contribuição deste trabalho, aqui são expostos os resultados algébricos relacionados ao conjunto de elementos *fuzzy* hesitantes típicos. Também é neste capítulo que são apresentados os resultados teóricos relacionados aos autômatos *fuzzy* hesitantes típicos. Este capítulo é subdividido nas seções: Combinações e Ordem sobre EFHT's, Autômatos *Fuzzy* Hesitantes Típicos e Pseudo Autômatos *Fuzzy* Hesitantes Típicos.

Capítulo 5. Ruído tipo sal e Pseudo Autômatos *Fuzzy* Hesitantes Típicos: neste capítulo é apresentada a segunda contribuição deste trabalho, aqui são apresentados e explicados os algoritmos que compõem o método de tratamento de ruídos do tipo sal em imagens escala de cinza.

Capítulo 6. Classificação de Dados e Pseudo Autômatos *Fuzzy* Hesitantes Típicos: neste capítulo é apresentada a terceira contribuição deste trabalho, aqui são apresentados e explicados os algoritmos que compõem o método de classificação de dados.

Capítulo 7. Conclusão: discute de forma sucinta as contribuições desta tese e propõem trabalhos futuros.

Parte I

Fundamentos

2 Conceitos Fundamentais

Este capítulo revisa brevemente os conceitos de autômatos finitos, reticulados, conjuntos *fuzzy* e conjuntos *fuzzy* hesitantes típicos. Estes conceitos são necessários para o desenvolvimento dos outros capítulos desta tese, assim os leitores já familiarizados com estes conceitos básicos podem prosseguir para os próximos capítulos.

2.1 Autômatos Finitos

A teoria dos autômatos finitos, ou simplesmente teoria dos autômatos, foi estabelecida entre as décadas de 40 e 60 do século XX pelos trabalhos de McCulloch e Pitts (MCCULLOCH; PITTS, 1943), Kleene (KLEENE, 1951), Mealy (MEALY, 1955), Moore (MOORE, 1956), Rabin e Scott (RABIN; SCOTT, 1959; RABIN, 1963). Os autômatos finitos são os mais simples modelos abstratos de máquinas de computação (FARIAS et al., 2016b), sendo vistos como simplificações limitadas das máquinas de Turing (TURING, 1937).

Antes de definir precisamente o conceito de autômato finito, é conveniente recorrer a uma definição informal apresentada em diversos livros sobre o tema, tais como (BEDREGAL; ACIÓLY; LYRA, 2010; HOPCROFT; MOTWANI; ULLMAN, 2008; LINZ, 2006). Nesta visão informal os autômatos finitos são descritos como sendo dispositivos ou computadores constituídos de três partes fundamentais (ver a Figura 1):

- Uma **fita** ou **memória** (somente de leitura) finita e dividida em **cédulas**, cada **cédula** sendo capaz de armazenar um único **símbolo** por vez.
- Um **cabeçote** de leitura capaz de ler uma única **cédula** por vez, e que, se movimenta na fita apenas no sentido da esquerda para à direita, acessando uma única **cédula** por vez.
- Uma **unidade de controle** que administra o **estado** interno do autômato.

Como dito em (BEDREGAL; ACIÓLY; LYRA, 2010), os autômatos finitos podem ser vistos como sendo computadores que operam em tempo discreto. Assim em qualquer momento t no tempo, a **unidade de controle** estará em algum **estado** interno possível e o **cabeçote** de leitura estará sobre uma das **cédulas** da **fita**. No próximo momento $t + 1$, o **estado** interno do autômato será determinado em função apenas do **estado** interno no momento t e do **símbolo** na **cédula** em que o **cabeçote** se encontrava no momento t . O par ordenado (**estado**, **símbolo**) costuma ser chamado de configuração.

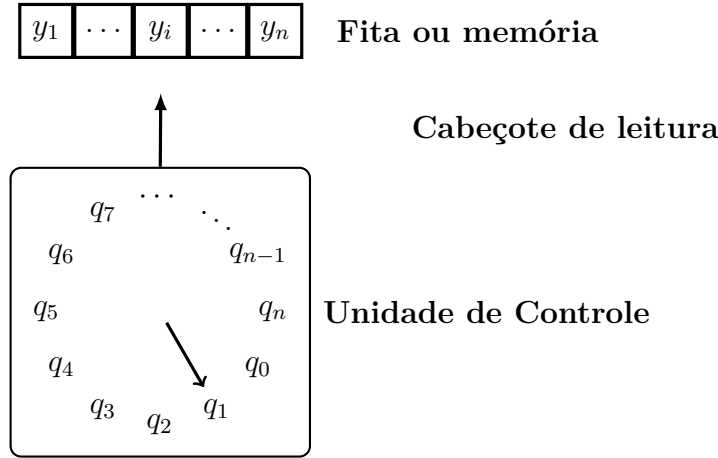


Figura 1 – Uma representação informal de autômato finito baseado na Figura 3 de (COSTA, 2016).

Como explicado por Hopcroft *et al.* em (HOPCROFT; MOTWANI; ULLMAN, 2008), os autômatos finitos estão intimamente ligados a ideia de linguagem, e portanto, antes de definir formalmente o conceito de autômato finito é interessante introduzir a noção de linguagem.

Definição 2.1.1. (COSTA; BEDREGAL, 2020) Um alfabeto Σ é um conjunto não vazio, cujo os elementos são chamados de símbolos. O conjunto Σ^* corresponde ao monoide livremente gerado por Σ com respeito a operação de concatenação de palavras. O elemento neutro de Σ^* é a palavra vazia, denotada por λ . O tamanho de uma palavra w é denotado por $|w|$ e é definido como $|w| = 0$ se w é a palavra vazia e $|w| = n$ se $w = a_1 \cdots a_n$ com $a_i \in \Sigma, 1 \leq i \leq n$ e $n \in \mathbb{Z}_+^*$. Além disso, é comum assumir o conjunto $\Sigma^+ = \Sigma^* - \{\lambda\}$.

Definição 2.1.2. (LINZ, 2006) Uma linguagem L é qualquer subconjunto de Σ^* .

Formalmente existem dois tipos de autômatos finitos: Autômatos Finitos Determinísticos e Autômatos Finitos Não-determinísticos. Seguindo as ideias apresentadas por Hopcroft *et al.* (HOPCROFT; MOTWANI; ULLMAN, 2008), uma definição formal para a noção de Autômato Finito Determinístico (AFD) é dado como se segue.

Definição 2.1.3. (HOPCROFT; MOTWANI; ULLMAN, 2008) Um AFD é uma quintupla $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ tal que Q é um conjunto finito e não vazio de estados, Σ é um alfabeto, $\delta : Q \times \Sigma \rightarrow Q$ é a função de transição entre estados, $q_0 \in Q$ é o estado inicial e $F \subseteq Q$ é o conjunto de estados finais¹.

Exemplo 2.1.1. A estrutura $A = \langle \{q_0, q_1\}, \{a\}, \delta, q_0, \{q_1\} \rangle$ onde a função de transição é definida por: $\delta(q_0, a) = q_1$ e $\delta(q_1, a) = q_0$, é um AFD.

¹ Em alguns textos também é usado o termo conjunto de estados de aceitação (HIGUERA, 2010).

Observação 2.1.1. *Para simplificar a escrita, neste trabalho será adicionado um apóstrofo e um s no final de uma sigla para referenciar o plural da mesma, por exemplo, AFD's será o plural de AFD.*

A função de transição δ descreve o comportamento da mudança entre os estados do AFD, isto é, quando escrevemos $\delta(q, x) = p$ significa que, o AFD muda do estado atual q para o estado p quando o cabeçote de leitura lê o símbolo x na fita. Como explicado em (HOPCROFT; MOTWANI; ULLMAN, 2008), δ pode ser estendida em uma função $\hat{\delta} : Q \times \Sigma^* \rightarrow Q$ para $a \in \Sigma$ e $w \in \Sigma^*$ usando recursividade como se segue:

$$\hat{\delta}(q, \lambda) = q \quad (2.1)$$

$$\hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a) \quad (2.2)$$

A noção de computação em um AFD consiste da tarefa de aceitar ou rejeitar palavras de entrada (MARTIN, 1991). Neste sentido, qualquer AFD é uma máquina capaz de resolver certos problemas de decisão. Para mais detalhes sobre isto ver (BÜCHI, 1960; CHURCH, 1962).

Definição 2.1.4. (BEDREGAL; ACIÓLY; LYRA, 2010) *Sejam $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ um AFD e $w \in \Sigma^*$. A palavra w é aceita por A sempre que $\hat{\delta}(q_0, w) \in F$ e é rejeitada por A em qualquer outro caso.*

É bem conhecido que $\hat{\delta}(q_0, w) \in F$ com $w = a_1 a_2 \cdots a_n$ se, e somente se, existir uma sequência finita de estados $(q_1, q_2, \cdots, q_{n-1}, q_n)$ com $q_n \in F$ tal que $\delta(q_0, a_1) = q_1, \delta(q_1, a_2) = q_2, \delta(q_2, a_3) = q_3, \cdots, \delta(q_{n-1}, a_n) = q_n$. Em particular tem-se que $\hat{\delta}(q_0, \lambda) \in F$ se, e somente se, $q_0 \in F$.

Definição 2.1.5. (BEDREGAL; ACIÓLY; LYRA, 2010) *A linguagem aceita por um AFD $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ é denotada por $L(A)$ e corresponde ao conjunto de todas as palavras sobre Σ que são aceitas por A . Em notação formal isto é escrito como $L(A) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$.*

Exemplo 2.1.2. *Seja A o AFD do Exemplo 2.1.1, é fácil mostrar por indução que a linguagem de A corresponde ao conjunto $L(A) = \{w \in \Sigma^* \mid |w| \text{ é ímpar}\}$.*

Observação 2.1.2. (RABIN, 1963) *A família² de todas as linguagens que são aceitas por AFD's e chamada de família das linguagens regulares, denotada por L_{Reg} . E tal família é enumerável.*

É comum representar os AFD's usando grafos de transição. Em um grafo de transição os vértices são representados por círculos, que neste caso são usados para

² Em alguns textos é usada a palavra conjunto (ou classe) para se referir a uma família de linguagens.

representar os elementos de Q , e as arestas (q_i, q_j) são rotuladas por x representando a equação $\delta(q_i, x) = q_j$, os estados que pertencem a F costumam ser representados por vértices desenhados como um círculo duplo, um exemplo deste tipo de representação é esboçado na Figura 2.

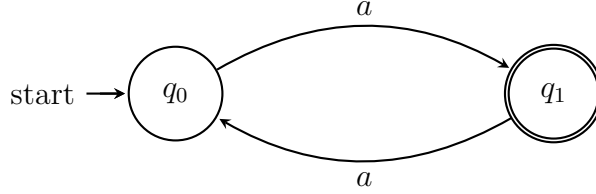


Figura 2 – Representação visual do AFD no Exemplo 2.1.1.

Para introduzir formalmente a noção de Autômato Finito Não-determinístico (AFN) é necessário antes explicar a diferença com o modelo determinístico. De acordo com Peter Linz (LINZ, 2006), a característica que diferencia um AFN de um AFD é exatamente sua função de transição. Em qualquer AFN a função de transição possui duas características que não são apresentadas pela função de transição de qualquer AFD, sendo essas características:

- (1) No AFN as transições entre dois estados distintos podem ocorrer sem o uso de um símbolo do alfabeto, isto é, usando a palavra vazia e
- (2) em um AFD a função de transição mapeia uma configuração (estado, símbolo) em um único estado, já a função de transição de um AFN mapeia uma configuração (estado, símbolo) em um conjunto de estados.

Definição 2.1.6. (BEDREGAL; ACIÓLY; LYRA, 2010) Um AFN é uma quintupla $A = \langle Q, \Sigma, \delta_N, q_0, F \rangle$ assumindo as mesmas definições para Q, Σ, q_0 e F encontradas na Definição 2.1.3 e sendo $\delta_N : Q \times (\Sigma \cup \{\lambda\}) \rightarrow 2^Q$ a função de transição entre estados do AFN.

Observação 2.1.3. Um fato bem conhecido na literatura sobre autômatos finitos é que as transições rotuladas por λ , também chamadas λ -transições³, não incrementam o poder computacional dos AFN's (MENEZES, 1998), assim aqui será considerado que o domínio da função δ_N é apenas $Q \times \Sigma$ em vez de $Q \times (\Sigma \cup \{\lambda\})$.

Como para o caso determinístico, a função de transição dos AFN's, isto é, a função δ_N , pode ser estendida para uma função $\widehat{\delta}_N : Q \times \Sigma^* \rightarrow 2^Q$ usando recursividade. A função $\widehat{\delta}_N$ é definida para $a \in \Sigma, w \in \Sigma^*$ e $q \in Q$ como se segue:

$$\widehat{\delta}_N(q, \lambda) = \{q\} \quad (2.3)$$

$$\widehat{\delta}_N(q, wa) = \bigcup_{q' \in \widehat{\delta}_N(q, w)} \delta_N(q', a) \quad (2.4)$$

³ λ -transições são transições da forma $\delta_N(q, \lambda)$ para algum $q \in Q$.

Exemplo 2.1.3. A estrutura $A = \langle \{q_0, q_1, q_2, q_3\}, \{a, b\}, \delta_N, q_0, \{q_3\} \rangle$ com δ_N sendo definido pela matriz de transição a seguir é um AFN.

	a	b
q_0	$\{q_0, q_1\}$	$\{q_0, q_2\}$
q_1	$\{q_3\}$	$\{q_1\}$
q_2	$\{q_2\}$	$\{q_3\}$
q_3	$\{q_3\}$	$\{q_3\}$

Da mesma maneira que acontece com os AFD's, é possível representar os AFN's usando grafos de transição. Novamente os elementos do conjunto de estados Q serão os vértices e cada aresta (q_i, q_j) é rotulada por x se, e somente se, $q_j \in \delta(q_i, x)$. Um exemplo de AFN nesta representação pode ser visto na Figura 3.

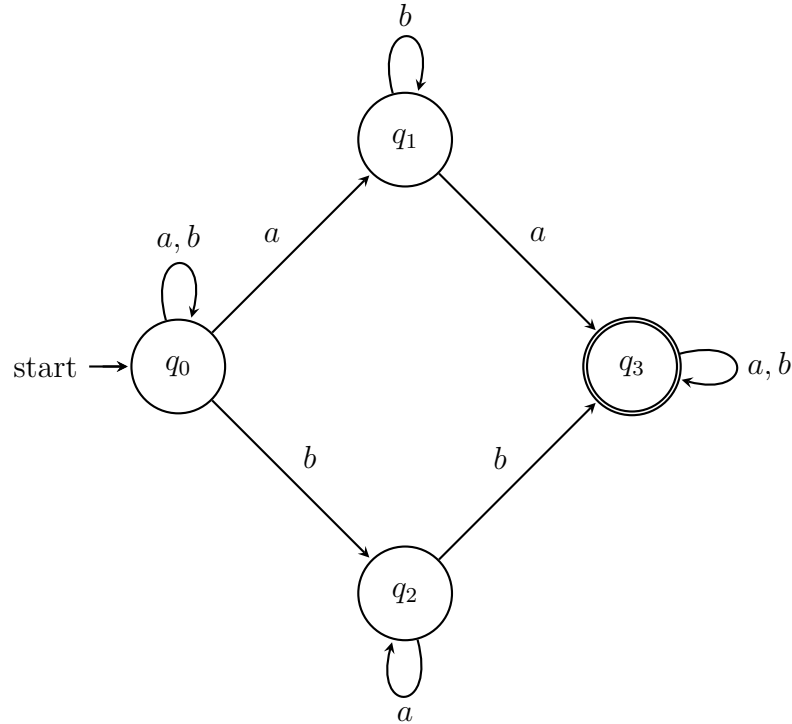


Figura 3 – Representação visual do AFN no Exemplo 2.1.3.

Definição 2.1.7. (BEDREGAL; ACIÓLY; LYRA, 2010) Sejam $A = \langle Q, \Sigma, \delta_N, q_0, F \rangle$ um AFN e $w \in \Sigma^*$. A palavra w é aceita pelo AFN A sempre que $\widehat{\delta}_N(q_0, w) \cap F \neq \emptyset$ e é rejeitada por A em qualquer outro resultado desta interseção.

Definição 2.1.8. (SIPSER, 2012) A linguagem aceita por um AFN $A = \langle Q, \Sigma, \delta_N, q_0, F \rangle$ é o conjunto $L(A) = \{w \in \Sigma^* \mid \widehat{\delta}(q_0, w) \cap F \neq \emptyset\}$.

Observação 2.1.4. Observe que dado um AFN A uma palavra $a_1 \cdots a_n \in L(A)$ implica na existência de uma sequência de conjuntos $X_1, \dots, X_n$, com $X_i \subseteq Q$ para qualquer $1 \leq i \leq n$ tal que $\delta_N(q_0, a) = X_1, \dots, \delta_N(X_{n-1}, a) = X_n$ e $X_n \cap F \neq \emptyset$.

Teorema 2.1.1. (*BEDREGAL; ACIÓLY; LYRA, 2010*) Se A é um AFN, então existe um AFD A' tal que $L(A) = L(A')$.

Corolário 2.1.1. (*LINZ, 2006*) Seja L uma linguagem sobre Σ . L está em L_{Reg} se, e somente se, existe um AFN A tal que $L = L(A)$.

O Corolário 2.1.1 diz que AFD's e AFN's são equivalentes em termos da família de linguagens que estas máquinas computam (ou aceitam). Dessa forma, qualquer resultado válido para AFD's e suas linguagens também valerá para os AFN's e suas respectivas linguagens. A família L_{Reg} apresenta as seguintes propriedades.

Teorema 2.1.2. (*BEDREGAL; ACIÓLY; LYRA, 2010*) Se L_1 e L_2 são duas linguagens regulares, então $L_1 \cup L_2$ é uma linguagem regular.

Teorema 2.1.3. (*BEDREGAL; ACIÓLY; LYRA, 2010*) Se L_1 e L_2 são duas linguagens regulares, então $L_1 \cap L_2$ é uma linguagem regular.

Teorema 2.1.4. (*BEDREGAL; ACIÓLY; LYRA, 2010*) Se L é uma linguagem regular, então o complemento de L (denotado por $\bar{L}$) é também uma linguagem regular.

Os resultados acima enunciados foram primeiramente demonstrados por Rabin e Scott em seu trabalho (*RABIN; SCOTT, 1959*), que apresenta estas operações como operadores de fechamento para uma álgebra de linguagens regulares. Eles apresentaram o seguinte resultado.

Teorema 2.1.5. (*RABIN; SCOTT, 1959*) A família L_{Reg} é uma álgebra booleana de conjuntos.

Em adicional a este resultado, Rabin e Scott também provaram que a família L_{Reg} é fechada sobre as operações de concatenação de linguagens e fecho de Kleene (ou fecho estrela), para detalhes ver (*BEDREGAL; ACIÓLY; LYRA, 2010; HOPCROFT; MOTWANI; ULLMAN, 2008; RABIN; SCOTT, 1959*). Outro significativo resultado provado por eles relaciona a operação de interseção de linguagens regulares com a operação de produto direto de autômatos.

Definição 2.1.9. Seja $A = \langle Q', \Sigma, \delta', q'_0, F' \rangle$ e $B = \langle Q'', \Sigma, \delta'', q''_0, F'' \rangle$ dois AFD's. O produto direto $A \times B$ é um AFD $\mathcal{U} = \langle Q' \times Q'', \Sigma, \delta, (q'_0, q''_0), F' \times F'' \rangle$ onde δ é definido para qualquer $(q, r) \in (Q' \times Q'')$ e $a \in \Sigma$ como sendo:

$$\delta((q, r), a) = (\delta'(q, a), \delta''(r, a)) \quad (2.5)$$

Teorema 2.1.6. (*RABIN; SCOTT, 1959*) Se A e B são AFD's. Então $L(A \times B) = L(A) \cap L(B)$.

Definição 2.1.10. (*BEDREGAL; ACIÓLY; LYRA, 2010*) Sejam Σ_1 e Σ_2 dois alfabetos e $H : \Sigma_1 \rightarrow \Sigma_2^*$ uma função. A função $\widehat{H} : \Sigma_1^* \rightarrow \Sigma_2^*$ definida recursivamente para cada $w \in \Sigma_1^*$ e $a \in \Sigma_1$ por:

$$\widehat{H}(\lambda) = \lambda \quad (2.6)$$

$$\widehat{H}(a) = H(a) \quad (2.7)$$

$$\widehat{H}(wa) = \widehat{H}(w)H(a) \quad (2.8)$$

é chamada de homomorfismo⁴ (ou homomorfismo de linguagem).

Teorema 2.1.7. (*BEDREGAL; ACIÓLY; LYRA, 2010*) A família L_{Reg} é fechada sobre a operação de homomorfismo.

Nesta seção foram expostos apenas os autômatos finitos com a arquitetura padrão, isto é, com uma memória (somente de leitura) finita e um único cabeçote de leitura. Para leitores interessados em autômatos com uma memória adicional de escrita ver os trabalhos (*MOORE, 1956; MEALY, 1955; FARIAS et al., 2016b*) e para autômatos com múltiplos cabeçotes (*COSTA, 2016; COSTA; BEDREGAL, 2018; BEDREGAL, 2011; BEDREGAL, 2015; BIRD, 1973; PELLETIER; SAKAROVITCH, 1999; SUDBOROUGH, 1973*). E para autômatos com múltiplas fitas (somente de leitura) ver (*HARJU; KARHUMÄKI, 1990; HARJU; KARHUMÄKI, 1991; KOBAYASHI; SEKIGUCHI, 1966*).

2.2 Ordens Parciais e Reticulados

Uma vez que reticulados são casos especiais de sistemas parcialmente ordenados (*MEZZOMO, 2013*), para introduzir as noções ligadas às estruturas de reticulados antes é necessário apresentar os conceitos fundamentais da teoria das ordens parciais.

Definição 2.2.1. (*DAVEY; PRIESTLEY, 2002*) Seja P um conjunto não vazio, uma ordem parcial sobre P é uma relação binária $\preceq$ tal que para todo $x, y, z \in P$ as seguintes condições são satisfeitas:

- (i) $x \preceq x$ (reflexividade);
- (ii) Se $x \preceq y$ e $y \preceq x$, então $x = y$ (anti-simetria);
- (iii) Se $x \preceq y$ e $y \preceq z$, então $x \preceq z$ (transitividade).

O objeto fundamental no estudo da teoria das ordens parciais é a noção de sistema parcialmente ordenado⁵, ou simplesmente SPO (*ABE; PAPAVERO, 1991*). A seguir é definido formalmente este conceito básico.

⁴ Homomorfismo do monoide Σ_1^* para o monoide Σ_2^* .

⁵ Alguns textos usam o termo conjunto parcialmente ordenado.

Definição 2.2.2. (*ABE; PAPAVERO, 1991*) Um SPO é um par $\langle P, \preceq \rangle$ onde P é um conjunto não vazio e $\preceq$ é uma relação de ordem parcial sobre P .

Exemplo 2.2.1. (*ABE; PAPAVERO, 1991*) Seja $\mathbb{N}$ o conjunto dos números naturais e seja $|$ a relação binária sobre $\mathbb{N}$ definida como sendo:

$$x|y \iff x \text{ divide } y$$

é fácil verificar que o par $\langle \mathbb{N}, | \rangle$ é um SPO.

Exemplo 2.2.2. (*PAIVA, 2019*) Seja $X = \{a, b, c\}$ um conjunto finito, a estrutura $\langle 2^X, \subseteq \rangle$ é um SPO.

Como explicado por Mezzomo em (*MEZZOMO, 2013*), alguns SPO's possuem elementos distintos (ou destacados) que satisfazem propriedades específicas. Os mais famosos destes elementos são chamados de mínimo e máximo, que são formalmente definidos como segue.

Definição 2.2.3. Um elemento $\perp \in P$ é o menor elemento ou mínimo do SPO $\langle P, \preceq \rangle$ se satisfaz a condição, $\perp \preceq x$ para qualquer $x \in P$. Dualmente, um elemento $\top \in P$, é o maior elemento ou máximo do SPO $\langle P, \preceq \rangle$ se satisfaz a condição, $x \preceq \top$ para qualquer $x \in P$.

Exemplo 2.2.3. Para o SPO do Exemplo 2.2.2 tem-se que o elemento $\emptyset$ é o mínimo e o elemento $\{a, b, c\}$ corresponde ao máximo, isto é $\perp = \emptyset$ e $\top = \{a, b, c\}$.

Observação 2.2.1. Existem SPO's que não possuem maior e (ou) menor elemento, um exemplo disto é o SPO $\langle 2^X - \{\emptyset, X\}, \subseteq \rangle$.

Exemplo 2.2.4. (*ABE; PAPAVERO, 1991*) Seja $\langle \mathbb{N}, | \rangle$ o SPO do Exemplo 2.2.1, é fácil ver que os números 0 e 1 são respectivamente o máximo e o mínimo do SPO, ou seja, vale as igualdades $\perp = 1$ e $\top = 0$.

Como dito em (*ABE; PAPAVERO, 1991*), sempre que um SPO possui mínimo e máximo, esses elementos são únicos, e neste caso o SPO é chamado de SPO limitado.

Definição 2.2.4. (*ROMAN, 2008*) Seja $\langle P, \preceq \rangle$ um SPO e $\mathbb{A} \subseteq P$. Se existe um elemento $y \in P$ tal que:

(i) $y \preceq x$, para todo $x \in \mathbb{A}$.

(ii) Se $z \in P$ é tal que $z \preceq x$ para qualquer $x \in \mathbb{A}$, então $z \preceq y$.

então y chama-se ínfimo de $\mathbb{A}$.

Definição 2.2.5. (*ROMAN, 2008*) Seja $\langle P, \preceq \rangle$ um SPO e $\mathbb{A} \subseteq P$. Se existe um elemento $y \in P$ tal que:

(i) $x \preceq y$, para todo $x \in \mathbb{A}$.

(ii) Se $z \in P$ é tal que $x \preceq z$ para qualquer $x \in \mathbb{A}$, então $y \preceq z$.

então y chama-se supremo de $\mathbb{A}$.

Exemplo 2.2.5. (*ABE; PAPAVERO, 1991*) Considere o SPO $\langle \mathbb{N}, | \rangle$. É bem conhecido que para qualquer subconjunto finito e não vazio $X \subseteq \mathbb{N}$, as operações de mínimo múltiplo comum (ou MMC) e o máximo divisor comum (ou MDC) fornecem respectivamente o supremo e o ínfimo de X .

Proposição 2.2.1. (*ROMAN, 2008*) Se $\langle \mathcal{L}, \preceq \rangle$ é um SPO limitado, então $\top$ e $\perp$ são respectivamente o supremo e o ínfimo do conjunto $\mathcal{L}$.

Usando a mesma notação que o trabalho (*FARIAS et al., 2016b*), o supremo de um subconjunto $\mathbb{A}$ quando existe, é denotado por $\bigvee \mathbb{A}$, e em particular, sempre que $\mathbb{A} = \{x, y\}$, o supremo de $\mathbb{A}$ é escrito como sendo $x \vee y$. Dualmente, o ínfimo quando existir, é denotado por $\bigwedge \mathbb{A}$, e no caso de $\mathbb{A} = \{x, y\}$, o ínfimo de $\mathbb{A}$ é escrito como sendo $x \wedge y$.

Observação 2.2.2. Vale salientar que, $x \wedge y = y \wedge x$ e $x \vee y = y \vee x$. Além disso, $x \wedge x = x \vee x = x$.

Definição 2.2.6. (*SYROPOULOS, 2014*) Um SPO $\langle P, \preceq \rangle$ é chamado de reticulado quando para todo par $x, y \in P$ o conjunto $\{x, y\}$ possui supremo e ínfimo.

Todo reticulado $\langle \mathcal{L}, \preceq \rangle$ possui uma versão algébrica em que ínfimo e supremo são vistos como operações binárias sobre o conjunto $\mathcal{L}$, isto é, o ínfimo e o supremo passam a ser funções com a seguinte assinatura $\wedge, \vee : \mathcal{L} \times \mathcal{L} \rightarrow \mathcal{L}$. Em geral esta versão algébrica é denotada por $\langle \mathcal{L}, \vee, \wedge \rangle$, para mais detalhes, incluído as propriedades das funções $\vee$ e $\wedge$ ver a referência (*ABE; PAPAVERO, 1991*).

Observação 2.2.3. (*ROMAN, 2008*) Sempre que, $\langle \mathcal{L}, \preceq \rangle$ é um SPO limitado, o reticulado definido sobre este SPO é chamado de reticulado limitado, e é denotado por $\langle \mathcal{L}, \vee, \wedge, \perp, \top \rangle$.

Proposição 2.2.2. (*ROMAN, 2008*) Seja $\langle \mathcal{L}, \vee, \wedge, \perp, \top \rangle$ um reticulado limitado, então para todo $x \in \mathcal{L}$ as seguintes igualdades são satisfeitas:

$$(\mathcal{L}1) \quad x \vee \perp = \perp \vee x = x.$$

$$(\mathcal{L}2) \quad x \wedge \top = \top \wedge x = x.$$

$$(\mathcal{L}3) \quad x \vee \top = \top \vee x = \top.$$

$$(\mathcal{L}4) \quad x \wedge \perp = \perp \wedge x = \perp.$$

$$(\mathcal{L}5) \quad x \wedge x = x \vee x = x.$$

Um reticulado é dito ser completo quando todo subconjunto (incluindo o conjunto vazio) possui supremo e ínfimo. Note que como dito em (ABE; PAPAVERO, 1991), sempre que o reticulado for completo implica que ele será limitado.

Definição 2.2.7. (ROMAN, 2008) Um reticulado $\langle \mathcal{L}, \vee, \wedge, \perp, \top \rangle$ é chamado de distributivo quando para todo $x, y, z \in \mathcal{L}$:

$$(\mathcal{L}6) \quad x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z).$$

$$(\mathcal{L}7) \quad x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z).$$

Exemplo 2.2.6. A estrutura $\langle [0, 1], \vee, \wedge, 0, 1 \rangle$ é um reticulado distributivo, neste caso $[0, 1]$ é ordenado pelo ordem usual $\leq$ sobre o conjunto dos números reais $\mathbb{R}$. Para este caso particular tem-se que $\vee = \max$ and $\wedge = \min$.

Observação 2.2.4. Sempre que o contexto for claro, será referenciado o reticulado apenas por seu conjunto base, ficando sua estrutura algébrica subentendida.

2.3 Conjuntos Fuzzy

De acordo com (COSTA; BEDREGAL, 2018), a teoria dos conjuntos *fuzzy* introduzida por Zadeh (ZADEH, 1965), é uma generalização da teoria clássica de conjuntos proposta pelo matemático Georg Cantor em 1884. Na teoria de Cantor, a natureza da relação de “pertencer”, existente entre elementos e conjuntos de um universo do discurso é bi-valorada, isto é, para qualquer elemento x e conjunto A em algum universo do discurso apenas uma das sentenças:

(a) x pertence a A , ou em notação formal $x \in A$.

(b) x não pertence a A , ou em notação formal $x \notin A$.

pode ser verdadeira. Assim a teoria dos conjuntos de Cantor preserva o princípio lógico do terceiro excluído. Em contraste à teoria de Cantor, Zadeh em sua teoria dos conjuntos *fuzzy*, admite que para qualquer conjunto universo $\mathbb{U}$, os elementos de $\mathbb{U}$ pertence a um subconjunto A com um certo nível de certeza (ou adequação), uma interpretação semântica (não inteiramente correta) disso é dizer que a teoria de Zadeh ignora o princípio do terceiro excluído, permitindo que elementos pertençam parcialmente a um subconjunto A , e portanto, também pertencem parcialmente ao complemento de A .

Definição 2.3.1. (*ZADEH, 1965*) Seja $\mathbb{U}$ um conjunto não vazio, chamado de universo⁶. Um conjunto fuzzy μ sobre $\mathbb{U}$ é uma função $\mu : \mathbb{U} \rightarrow [0, 1]$, onde $\mu(x)$ denota o valor⁷ de pertinência (certeza ou adequação) do elemento x em relação ao conjunto fuzzy μ .

Exemplo 2.3.1. Seja $\mathbb{N}$ o conjunto dos números naturais e $k \in \mathbb{N}$. O conjunto fuzzy de todos os números muito distantes de k , denotado por μ_{D_k} , pode ser definido para todo $x \in \mathbb{N}$ como sendo $\mu_{D_k}(x) = 1 - \frac{1}{2^{|k-x|}}$.

Alguns autores alternativamente usam uma representação por pares ordenados para representar conjuntos fuzzy (*COSTA; BEDREGAL, 2018; MEZZOMO, 2013*), nesta notação de pares ordenados um conjunto fuzzy T sobre $\mathbb{U}$ é escrito como sendo,

$$T = \{(x, \mu(x)) \mid x \in \mathbb{U}\} \quad (2.9)$$

é comum nessa notação indexar a função μ pelo símbolo usado para denotar o conjunto fuzzy, para o exemplo acima seria escrito μ_T .

Sobre conjuntos fuzzy é comum se adotar a interpretação sobre as “bordas” do conjunto fuzzy, no sentido clássico da teoria dos conjuntos. Como discutido em (*BARROS; BASSANEZI, 2015*) dado um conjunto fuzzy μ tem-se que $\mu(x) = 1$ significa total certeza da pertinência do elemento x com respeito ao conjunto fuzzy μ , e $\mu(x) = 0$, representa a total certeza da não pertinência do elemento x com respeito ao conjunto fuzzy μ .

Desde que foram introduzidos por Zadeh, diversos conceitos e operadores ligados a ideia de conjuntos fuzzy foram introduzidos. Entre os operadores alguns de destaque são: t-normas (*KLEMENT; MESIAR; PAP, 2013*), t-conormas (*KLEMENT; MESIAR; PAP, 2013*), negações fuzzy (*LIMA; BEDREGAL; MEZZOMO, 2020*), implicações fuzzy (*BEDREGAL, 2010; FODOR, 1991; PINHEIRO et al., 2018*), uninormas (*BAETS, 1999; BAETS; FODOR, 1999; YAGER, 2001*), funções de agregação fuzzy (*GRABISCH et al., 2009*), *overlaps* (*DIMURO et al., 2019*) e funções *grouping* (*BEDREGAL et al., 2013*).

Além dos operadores, como dito por Torra (*TORRA, 2010*), varias extensões para os conjuntos fuzzy foram propostas, algumas destas extensões são: os conjuntos fuzzy intuicionistas de Atanassov (*ATANASSOV, 1994*), conjuntos fuzzy tipo 2 (*MENDEL; JOHN, 2002*), conjuntos fuzzy multidimensionais (*LIMA et al., 2020*) e os muti-conjuntos fuzzy (*MIYAMOTO, 2000*). Para leitores interessados em mais extensões de conjuntos fuzzy, é recomendado ler (*BUSTINCE et al., 2016*), e para detalhes sobre as relações de inclusão entre as extensões da teoria de conjuntos fuzzy ver (*LIMA, 2019*).

As definições das extensões de conjuntos fuzzy anteriormente mencionadas assumem que os valores de pertinência dos conjuntos fuzzy sempre são valores no intervalo $[0, 1]$ e a partir disto constroem suas próprias definições. Joseph Amadee Goguen generalizou o

⁶ Em algumas obras também é usado o termo conjunto de referência ou universo do discurso.

⁷ Na literatura em língua portuguesa também são usados os termos nível e grau.

conceito de conjunto *fuzzy* em (GOGUEN, 1967), sua ideia é que os valores de pertinência podem ser representados por elementos de um reticulado limitado, ou pelo menos de um SPO limitado, isto é Goguen substitui o conjunto $[0, 1]$ por um reticulado qualquer. Goguen nomeou esses novos conjuntos *fuzzy* como conjuntos $\mathcal{L}$ -fuzzy⁸.

Definição 2.3.2. (SYROPOULOS, 2014) *Um conjunto $\mathcal{L}$ -fuzzy μ sobre $\mathbb{U}$ é uma função $\mu : \mathbb{U} \rightarrow \mathcal{L}$, tal que $\mathcal{L}$ é um reticulado limitado.*

Usando então a estrutura de reticulado é comum generalizar as operações sobre conjuntos como se segue.

Definição 2.3.3. *Seja $\mu : \mathbb{U} \rightarrow \mathcal{L}$ e $\mu' : \mathbb{U} \rightarrow \mathcal{L}$ dois conjuntos $\mathcal{L}$ -fuzzy, a seguir são definidas para todo $x \in \mathbb{U}$ as operações de $\mathcal{L}$ -união e $\mathcal{L}$ -interseção que são denotadas respectivamente por, $(\mu + \mu')$ e $(\mu \star \mu')$.*

- $(\mu + \mu')(x) = \mu(x) \vee \mu'(x).$
- $(\mu \star \mu')(x) = \mu(x) \wedge \mu'(x).$

Outra ideia que passa pelo processo de extensão para a teoria dos conjuntos $\mathcal{L}$ -fuzzy é a ideia de subconjunto.

Definição 2.3.4. (SYROPOULOS, 2014) *Seja $\mu : \mathbb{U} \rightarrow \mathcal{L}$ e $\mu' : \mathbb{U} \rightarrow \mathcal{L}$ dois conjuntos $\mathcal{L}$ -fuzzy, tem-se que μ é um $\mathcal{L}$ -subconjunto de μ' se para todo $x \in \mathbb{U}$ a desigualdade $\mu(x) \preceq \mu'(x)$ é satisfeita, com $\preceq$ sendo a relação de ordem sobre a qual o reticulado $\mathcal{L}$ é definido.*

Observação 2.3.1. *Deste ponto em diante, o termo conjunto fuzzy será considerado como sendo um conjunto $\mathcal{L}$ -fuzzy, em que $\mathcal{L}$ irá corresponder ao reticulado distributivo $\langle [0, 1], \vee, \wedge, 0, 1 \rangle$.*

Apesar dos conjuntos *fuzzy* serem bastante adequados para modelar as incertezas associadas aos elementos em um universo do discurso (COSTA, 2016; BARROS; BAS-SANEZI, 2015), eles geram um novo tipo de incerteza, para ilustrar esse novo tipo de incerteza imagine que é preciso definir sobre as pessoas de uma determinada cidade um conjunto *fuzzy* das pessoas jovens, assuma a seguinte definição para esse conjunto *fuzzy*:

$$\mu_{jovens}(x) = \begin{cases} 1, & \text{se a idade de } x \text{ é menor ou igual que } 20 \\ 0.5, & \text{se a idade de } x \text{ é maior que } 20 \text{ e menor que } 45 \\ 0, & \text{qualquer outro caso} \end{cases} \quad (2.10)$$

Agora note que uma pessoa x com a idade de 21 anos não seria considerada uma pessoa jovem, pois $\mu(x) = 0.5$, o que na realidade não faria sentido visto que, com 21 anos

⁸ Em alguns trabalhos é comum encontrar o termo $\mathcal{L}$ -valorado (QIU, 2001).

as pessoas são sem dúvidas jovens, assim essa definição para o conjunto *fuzzy* não está precisa. Seria possível então redefinir a função μ_{jovens} para atribuir valor de pertinência 1 para uma pessoa com idade de 21 anos, mas assim pode-se recursivamente questionar a situação para uma pessoa com idade de 22 anos.

Esse exemplo revela a grande dificuldade ou **hesitação** que existe para se definir a função que representa o conjunto *fuzzy*, ou seja, a função para atribuir os valores de pertinência dos elementos em relação ao conjunto *fuzzy*. Diversas extensões de conjuntos *fuzzy* entre elas os conjuntos *fuzzy* intervalar (PEKALA, 2018; GRZEGORZEWSKI, 2004) e os conjuntos *fuzzy* hesitante (TORRA, 2010) (que será o tema da próxima seção), oferecem formas para contornar o problema que foi ilustrado acima.

2.4 Conjuntos Fuzzy Hesitantes e Hesitantes Típicos

Torra em (TORRA, 2010), apresentou uma nova extensão para os conjuntos *fuzzy*, que ele nomeou como conjuntos *fuzzy* hesitantes, onde o objetivo principal é modelar a incerteza produzida pela dúvida do ser humano ao extrair informações (RODRÍGUEZ et al., 2016).

Os conjuntos *fuzzy* hesitantes, além de, lidar com a questão de representar o valor de pertinência com que os elementos de um certo universo pertencem a um determinado subconjunto, também são capazes de modelar a hesitação gerada pela necessidade de se definir precisamente a função que representa os conjuntos *fuzzy* convencionais. Isto é possível porque, como destacado em (RODRÍGUEZ et al., 2014), em vez de determinar um único valor de pertinência para cada elemento do universo em relação a um subconjunto, os conjuntos *fuzzy* hesitantes associam cada elemento a um conjunto de valores de pertinência possíveis, quanto maior esse conjunto de valores, maior é a hesitação de se definir a pertinência do elemento em relação ao subconjunto analisado (TORRA; NARUKAWA, 2009).

Definição 2.4.1. (BEDREGAL et al., 2014b) *Seja $\mathbb{U}$ um conjunto universo não vazio. Um conjunto fuzzy hesitante (ou simplesmente CFH) τ sobre $\mathbb{U}$ é definido por uma função $\tau : \mathbb{U} \rightarrow 2^{[0,1]}$. $\tau(x)$ denota o conjunto de valores de pertinência do elemento x em relação ao conjunto τ .*

Exemplo 2.4.1. *A função $\tau : [0, 1] \rightarrow 2^{[0,1]}$ definida por,*

$$\tau(x) = \bigcup \{1 - y \mid 0 < y < x\} \quad (2.11)$$

é um CFH.

O aumento no interesse pelos conjuntos *fuzzy* hesitantes se dá pelo fato de que, tais conjuntos possuem um grande potencial prático, uma das principais áreas onde eles

levam vantagens sobre conjuntos *fuzzy* convencionais, é na área de tomada de decisão (XIA; XU, 2011; WEI, 2012; XIA; XU; CHEN, 2013; FARHADINIA, 2013), em especial em problemas que exigem a opinião de múltiplos especialistas, como foi destacado em (SANTOS et al., 2015).

Definição 2.4.2. (TORRA, 2010) *Seja $\Omega = \{\mu_i : \mathbb{U} \rightarrow [0, 1] \mid 1 \leq i \leq n\}$ uma família com n conjuntos *fuzzy*. O CFH associado a Ω , denotado por τ_Ω , é definido como sendo:*

$$\tau_\Omega(x) = \bigcup_{i=1}^n \{\mu_i(x)\} \quad (2.12)$$

para todo $x \in \mathbb{U}$.

A Definição 2.4.2 é uma forma apresentada em (TORRA, 2010) para obter conjuntos *fuzzy* hesitantes usando conjuntos *fuzzy* convencionais. Assim como para os conjuntos *fuzzy* convencionais, os conjuntos *fuzzy* hesitantes também podem ser enxergados como sendo um conjunto de pares ordenados na forma

$$K = \{(x, \tau_K(x)) \mid x \in \mathbb{U}\} \quad (2.13)$$

Esta notação foi primeiro usada por Xia e Xu (XIA; XU, 2011), sendo que por questões de conveniência, Xia e Xu não escrevem o símbolo K no índice da função τ .

Uma vez que os conjuntos *fuzzy* hesitantes são também generalizações de conjuntos clássicos, como destacado em (RODRÍGUEZ et al., 2014), as operações mais fundamentais sobre conjuntos *fuzzy* hesitantes são as generalizações da união e da interseção, que são formalmente definidas a seguir.

Definição 2.4.3. (TORRA, 2010) *Sejam $\tau_1 : \mathbb{U} \rightarrow 2^{[0,1]}$ e $\tau_2 : \mathbb{U} \rightarrow 2^{[0,1]}$ dois conjuntos *fuzzy* hesitantes, a união de ambos é o CFH $\tau_{\tau_1 \cup \tau_2} : \mathbb{U} \rightarrow 2^{[0,1]}$ definido como:*

$$\tau_{\tau_1 \cup \tau_2}(x) = \bigcup_{X \in \tau_1(x), X' \in \tau_2(x)} \{\max(X, X')\} \quad (2.14)$$

Definição 2.4.4. (TORRA, 2010) *Sejam $\tau_1 : \mathbb{U} \rightarrow 2^{[0,1]}$ e $\tau_2 : \mathbb{U} \rightarrow 2^{[0,1]}$ dois conjuntos *fuzzy* hesitantes, a interseção de ambos é o CFH $\tau_{\tau_1 \cap \tau_2} : \mathbb{U} \rightarrow 2^{[0,1]}$ definido como:*

$$\tau_{\tau_1 \cap \tau_2}(x) = \bigcap_{X \in \tau_1(x), X' \in \tau_2(x)} \{\min(X, X')\} \quad (2.15)$$

Além destas duas operações, há diversos outros operadores definidos sobre conjuntos *fuzzy* hesitantes como dito em (PEI; YI, 2015), entre estes operadores estão: o complemento (TORRA, 2010), os operadores de bordas superiores e inferiores (RODRÍGUEZ et al., 2014), funções de agregação *fuzzy* hesitantes (BEDREGAL et al., 2014a), t-normas (GARMENDIA; CAMPO; RECASENS, 2017; SANTOS et al., 2015), implicações *fuzzy*

hesitantes (WEN; ZHAO; XU, 2019), e também medidas de distância, similaridade e consenso (XU; XIA, 2011; ZHANG et al., 2017; MATZENAUER et al., 2019).

Recentemente, Bedregal *et al.* em seu trabalho (BEDREGAL et al., 2014a), realizaram estudos em um caso particular de CFH. Este caso particular, que eles chamaram de conjunto *fuzzy* hesitante típico (ou simplesmente CFHT), considera algumas restrições sobre a definição de CFH.

Definição 2.4.5. (BEDREGAL et al., 2014a) *Seja $\mathbb{H}$ o conjunto de todos os subconjuntos finitos e não vazios do intervalo $[0, 1]$. Um CFHT sobre $\mathbb{U}$ é definido por uma função $\tau : \mathbb{U} \rightarrow \mathbb{H}$.*

Como mencionado em (BEDREGAL et al., 2014a; SANTOS et al., 2015), cada $X \in \mathbb{H}$ é chamado de Elemento *Fuzzy* Hesitante Típico (EFHT), logo $\mathbb{H}$ é o conjunto de todos os elementos *fuzzy* hesitantes típicos. A cardinalidade de qualquer $X \in \mathbb{H}$ é denotada usando a notação padrão de teoria dos conjuntos $\#X$.

Observação 2.4.1. *A cardinalidade de qualquer EFHT é também chamada de nível de hesitação.*

Os conjuntos unitários $\{x\} \in \mathbb{H}$ são chamados de EFHT's degenerados, ou simplesmente degenerados. O conjunto de todos os degenerados de $\mathbb{H}$ neste trabalho será denotado por $\mathbb{H}_1$. Outro ponto a se destacar é que o CFH τ_Ω construído na Equação (2.12) é na verdade um CFHT, uma vez que Ω é um conjunto finito.

Observação 2.4.2. (RODRÍGUEZ et al., 2014) *Em diversos trabalhos os conceitos de $CFH(T)$ e $EFH(T)$ são usados indistintamente, sendo que ambos os conceitos são diferentes. Um $CFH(T)$ é uma função τ de um conjunto $\mathbb{U}$ no conjunto $2^{[0,1]}(\mathbb{H})$, já um $EFH(T)$ é o resultado da aplicação da função τ em um elemento particular $x \in \mathbb{U}$.*

2.5 Considerações Finais

Neste capítulo, foi introduzido ao leitor toda a gama de conhecimentos básicos necessários para prosseguir na leitura deste trabalho. Foram abordados aqui os tópicos relacionados à teoria dos autômatos finitos (HOPCROFT; MOTWANI; ULLMAN, 2008), onde foram destacados as noções ligadas ao processo de computação e de linguagens para tais modelos abstratos de máquinas de computação. Logo em seguida, foram apresentados os conceitos acerca da teoria das ordens parciais (ABE; PAPAVERO, 1991) e dos reticulados (MEZZOMO, 2013; ROMAN, 2008).

Este capítulo também apresentou uma introdução sucinta da teoria dos conjuntos *fuzzy* (ZADEH, 1965) e sua generalização *$\mathcal{L}$ -fuzzy* (GOGUEN, 1967). Depois, foram

apresentados as definições básicas para as generalizações da teoria dos conjuntos *fuzzy*, chamadas de teoria dos conjuntos *fuzzy* hesitantes (TORRA, 2010) e teoria dos conjuntos *fuzzy* hesitantes típicos (BEDREGAL et al., 2014b), onde foi dado mais enfoque para os conjuntos *fuzzy* hesitantes típicos, os quais tiveram algumas de suas operações apresentadas aqui. Por fim, foi apresentado a ideia de elementos *fuzzy* hesitantes típicos e foi discutido brevemente a diferença destes com os conjuntos *fuzzy* hesitantes típicos.

De posse desses conhecimentos o leitor pode seguir então ao próximo capítulo onde será detalhado o estado atual da arte na teoria dos autômatos *fuzzy*.

3 Autômatos *Fuzzy* Estado Atual

Uma vez que esta tese se propõe a apresentar uma nova generalização para os autômatos finitos *fuzzy*, ou simplesmente autômatos *fuzzy*, este capítulo apresentará os aspectos do atual estado da arte na teoria dos autômatos *fuzzy*. Além disso, serão apresentadas algumas de suas generalizações e também algumas aplicações. Antes de definir formalmente os conceitos ligados aos autômatos *fuzzy*, é conveniente apresentar algumas noções referentes às linguagens *fuzzy*.

3.1 Linguagens *Fuzzy*

Lee e Zadeh introduziram em (LEE; ZADEH, 1969), os conceitos de linguagens e gramáticas *fuzzy* como sendo extensões de suas respectivas contra-partes clássicas. Formalmente uma linguagem *fuzzy* é definida como se segue.

Definição 3.1.1. (COSTA, 2016) *Seja Σ um alfabeto. Uma linguagens fuzzy $\mathbb{L}$ sobre Σ é um conjunto fuzzy $\mathbb{L} : \Sigma^* \rightarrow [0, 1]$.*

Após o trabalho inicial de Lee e Zadeh (LEE; ZADEH, 1969), as linguagens *fuzzy* tem sido estudadas extensivamente sob diferentes ângulos, para maiores detalhes ver os trabalhos (ASVELD, 2003; ASVELD, 2005; ASVELD, 2005; YU, 2001; CHAUDHARI; KOMEJWAR, 2011; MORDESON; MALIK, 2002). Na literatura são encontradas várias operações definidas para as linguagens *fuzzy* (MORDESON; MALIK, 2002), a seguir são apresentadas as operações de união e interseção *fuzzy*, produto Cartesiano, homomorfismo e quociente para a teoria das linguagens *fuzzy*.

Definição 3.1.2. (LEE; ZADEH, 1969) *Sejam $\mathbb{L}_1$ e $\mathbb{L}_2$ duas linguagens fuzzy sobre Σ , a união e interseção (respectivamente as Equações (3.1) e (3.2)) de $\mathbb{L}_1$ e $\mathbb{L}_2$, denotadas por $\mathbb{L}_1 + \mathbb{L}_2$ e $\mathbb{L}_1 \cdot \mathbb{L}_2$, são as linguagens fuzzy definidas por,*

$$\mathbb{L}_1 + \mathbb{L}_2(x) = \mathbb{L}_1(x) \vee \mathbb{L}_2(x) \quad (3.1)$$

e

$$\mathbb{L}_1 \cdot \mathbb{L}_2(x) = \mathbb{L}_1(x) \wedge \mathbb{L}_2(x) \quad (3.2)$$

Já o produto Cartesiano de linguagens *fuzzy* é formalmente definido como sendo o Cartesiano usual de funções.

Definição 3.1.3. (KOMEJWAR, 2012) *Sejam $\mathbb{L}_1$ e $\mathbb{L}_2$ duas linguagens fuzzy sobre Σ , o produto Cartesiano de $\mathbb{L}_1$ e $\mathbb{L}_2$, denotado por $\mathbb{L}_1 \times \mathbb{L}_2$, é uma função com a assinatura*

$\mathbb{L}_1 \times \mathbb{L}_2 : \Sigma^* \rightarrow [0, 1]^2$ e definida pela equação a seguir,

$$\mathbb{L}_1 \times \mathbb{L}_2(x) = (\mathbb{L}_1(x), \mathbb{L}_2(x)) \quad (3.3)$$

Ainda em seu trabalho Komejwar também apresenta a versão *fuzzy* do conceito de homomorfismo de linguagens.

Definição 3.1.4. (*KOMEJWAR, 2012*) Seja $\widehat{H} : \Sigma_1^* \rightarrow \Sigma_2^*$ um homomorfismo estendido e seja $\mathbb{L} : \Sigma_2^* \rightarrow [0, 1]$ uma linguagem fuzzy. A linguagem fuzzy $\mathbb{L}^{\widehat{H}} : \Sigma_1^* \rightarrow [0, 1]$, definida como sendo,

$$\mathbb{L}^{\widehat{H}}(x) = \mathbb{L}(\widehat{H}(x)) \quad (3.4)$$

é chamada de linguagem fuzzy homomórfica de $\mathbb{L}$.

Definição 3.1.5. (*KOMEJWAR, 2012*) Sejam $\mathbb{L}_1$ e $\mathbb{L}_2$ duas linguagens fuzzy sobre o alfabeto Σ . O quociente à direita de $\mathbb{L}_1$ por $\mathbb{L}_2$, denotado por $\mathbb{L}_1 \mathbb{L}_2^{-1}$, é uma linguagem fuzzy definida sobre Σ dada por,

$$\mathbb{L}_1 \mathbb{L}_2^{-1}(x) = \bigvee_{y \in \Sigma^*} \left\{ \mathbb{L}_1(xy) \wedge \mathbb{L}_2(y) \right\} \quad (3.5)$$

já o quociente à esquerda, denotado por $\mathbb{L}_2^{-1} \mathbb{L}_1$, é dado por,

$$\mathbb{L}_2^{-1} \mathbb{L}_1(x) = \bigvee_{y \in \Sigma^*} \left\{ \mathbb{L}_1(yx) \wedge \mathbb{L}_2(y) \right\} \quad (3.6)$$

3.2 Autômatos Fuzzy

Existem diversas formalizações e interpretações para a noção de autômatos *fuzzy* (*COSTA, 2016*), antes de definir formalmente os autômatos *fuzzy*, neste texto serão apresentadas duas interpretação dos mesmos.

A interpretação mais simples sobre autômatos *fuzzy* diz que, os autômatos *fuzzy* podem ser descritos como sendo AFN's em que, a todo momento seu estado interno corresponde a uma sobreposição de todos os estados possíveis simultaneamente, sendo que nesta sobreposição existe um valor¹ de pertinência para cada estado individual (*MACIEL, 2005; COSTA, 2016*).

Outra interpretação (esta baseada em um viés mais funcional) para autômatos *fuzzy*, é a descrição dos autômatos *fuzzy* como sendo formalizações de máquinas capazes de modelar falhas durante seu funcionamento, nessa interpretação os autômatos *fuzzy* podem ser vistos como sendo computadores ou como algoritmos *fuzzy*² (*ZADEH, 1996*).

¹ Em textos que relacionam autômatos *fuzzy* com autômatos quânticos é usado o termo amplitude.

² Um algoritmo *fuzzy* consiste de um programa em que as instruções podem conter um certo teor de ambiguidade, além disso, cada instrução tem associada a si um valor de pertinência de ser executada.

Observação 3.2.1. *Neste trabalho será considerado a segunda interpretação dos autômatos fuzzy.*

Na segunda interpretação dos autômatos *fuzzy* as falhas que os autômatos *fuzzy* são capazes de modelar são divididas em três categorias: erros de inicialização, erros de finalização e erros de transição.

Os erros de inicialização, são erros ligados às instruções de inicialização dos algoritmos *fuzzy*, isto é, instruções ligadas ao armazenamento das variáveis de entrada. Os autômatos *fuzzy* que modelam esse tipo de erro são computadores que funcionam admitindo a existência de incertezas na ação de armazenar os dados de entrada na memória de forma correta. Esses autômatos *fuzzy* são caracterizados por possuírem um conjunto *fuzzy* de estados iniciais (MALIK; MORDESON, 1996; CHAUDHARI; KOMEJWAR, 2011; SANTOS, 1968a; SANTOS, 1972b; COSTA; BEDREGAL, 2018; BĚLOHLÁVEK, 2002).

A interpretação para um conjunto *fuzzy* de estados iniciais é que, o valor de pertinência de cada estado q_i ser um estado inicial, corresponde à valoração do seguinte predicado:

$$\left(\forall x \in \text{Entrada } \text{Stored}(x) \right) \wedge \text{start}(M, q_i)$$

onde $\text{Stored}(x)$ e $\text{start}(M, q_i)$ correspondem respectivamente as proposições:

- x está armazenado na memória.
- A máquina M começa a execução no estado q_i .

Já os erros de finalização, são erros ligados às instruções de encerramento dos algoritmos *fuzzy*, isto é, instruções ligadas à ação de interromper ou encerrar a execução do algoritmo. Os autômatos *fuzzy* que modelam esse tipo de erro são computadores que funcionam admitindo a existência de incertezas na ação de se interromper (ou finalizar) a execução do algoritmo. Esses autômatos possuem a característica de terem um conjunto *fuzzy* de estados finais (MALIK; MORDESON, 1996; CHAUDHARI; KOMEJWAR, 2011; SANTOS, 1968a; SANTOS, 1972b; COSTA; BEDREGAL, 2018; BĚLOHLÁVEK, 2002; COSTA, 2016; COSTA; BEDREGAL, 2020).

A interpretação para um conjunto *fuzzy* de estados finais é que, o valor de pertinência de cada estado q_i ser um estado final, corresponde à valoração do seguinte predicado:

$$\text{Stopped}(M, q_i) \wedge \text{AllExec}(M)$$

onde $\text{Stopped}(M, q_i)$ e $\text{AllExec}(M)$ correspondem respectivamente às proposições:

- A máquina M parou de funcionar no estado q_i .

- M executou todos os comandos do algoritmo.

Os erros de transição, são erros que estão ligados às instruções intermediárias dos algoritmos *fuzzy*, ou seja, instruções que não são de inicialização e nem de finalização. Os autômatos *fuzzy* que modelam esse tipo de erro, são aqueles onde a função de transição entre estados é vista como sendo um conjunto *fuzzy*, a interpretação semântica para um conjunto *fuzzy* de transições será detalhada mais adiante neste texto. Para leitores interessados em autômatos *fuzzy* com a função de transição sendo vista como um conjunto *fuzzy*, ver os trabalhos (MENDÍVIL; GARITAGOITIA, 2015; FARIAS et al., 2016b; BĚLOHLÁVEK, 2002; DHURE, 2014; CHAUDHARI; KOMEJWAR, 2011; FARIAS et al., 2016c; COSTA; BEDREGAL, 2020).

A primeira definição formal para os autômatos *fuzzy* foi apresentada por Wee em sua tese de doutorado de 1967 (WEE, 1967), depois junto com Santos, Wee apresentou uma nova formulação para os autômatos *fuzzy*, sendo essa formulação ainda mais geral que a definição inicial por ele apresentada (SANTOS; WEE, 1968). Depois Santos (SANTOS, 1968a; SANTOS, 1968b; SANTOS, 1972a; SANTOS, 1972b; SANTOS, 1976) e mais recentemente outros pesquisadores têm apresentados novos resultados sobre os autômatos *fuzzy* (QIU, 2001; QIU, 2004; FARIAS et al., 2016b), o que tornou a teoria dos autômatos *fuzzy* um campo de estudo cada vez mais sólido.

Neste trabalho será considerada a definição de autômato *fuzzy* da mesma maneira que apresentada por Mordeson e Malik em (MORDESON; MALIK, 2002), ou seja, será considerada a definição de autômatos *fuzzy* como sendo uma variação da noção de máquinas de estados *fuzzy*, dessa forma para definir formalmente a noção de autômatos *fuzzy* é necessário antes apresentar a definição formal das máquinas de estados *fuzzy*, e isso é feito como se segue.

Definição 3.2.1. (MORDESON; MALIK, 2002) *Uma máquina de estados fuzzy é uma tripla $M = \langle Q, \Sigma, \varphi \rangle$ onde,*

- Q é um conjunto finito e não vazio de estados;
- Σ é um alfabeto e
- $\varphi : Q \times \Sigma \times Q \rightarrow [0, 1]$ é um conjunto de transições (ou instruções) *fuzzy*.

Para qualquer máquina de estados *fuzzy*, os valores de pertinência das transições *fuzzy* podem ser interpretados como sendo as valorações de predicados da forma:

$$\left(In(M, q) \wedge Head(x) \right) \Rightarrow Switch(q, p)$$

onde $In(M, q)$, $Head(x)$ e $Switch(q, p)$ correspondem respectivamente às proposições:

- A máquina M está no estado q .
- O cabeçote³ está sobre uma cédula com um “ x ” na memória.
- A máquina muda do estado q para o estado p .

Por conta desta natureza *fuzzy* de suas transições, fica implícito que máquinas de estado *fuzzy* podem modelar erros de transição entre os estados.

Exemplo 3.2.1. A estrutura $M = \langle \{q_0, q_1, q_2\}, \{a\}, \varphi \rangle$ com:

$$\begin{aligned} \varphi(q_0, a, q_0) &= 0 & \varphi(q_1, a, q_0) &= 0 & \varphi(q_2, a, q_0) &= 0 \\ \varphi(q_0, a, q_1) &= 0.5 & \varphi(q_1, a, q_1) &= 1 & \varphi(q_2, a, q_1) &= 0 \\ \varphi(q_0, a, q_2) &= 0 & \varphi(q_1, a, q_2) &= 0.9 & \varphi(q_2, a, q_2) &= 0 \end{aligned}$$

é uma máquina de estados *fuzzy*.

Definição 3.2.2. Seja $M = \langle Q, \Sigma, \varphi \rangle$ uma máquina de estados *fuzzy*, o conjunto de transições *fuzzy* φ pode ser estendido para um conjunto *fuzzy* com a assinatura $\hat{\varphi} : Q \times \Sigma^* \times Q \rightarrow [0, 1]$ usando recursão como se segue:

$$\hat{\varphi}(q, \lambda, q') = \begin{cases} 0, & \text{se } q \neq q' \\ 1, & \text{senão} \end{cases} \quad (3.7)$$

$$\hat{\varphi}(q, wa, q') = \bigvee_{q'' \in Q} \left\{ \hat{\varphi}(q, w, q'') \wedge \varphi(q'', a, q') \right\} \quad (3.8)$$

para $a \in \Sigma, w \in \Sigma^*$ e $q, q' \in Q$.

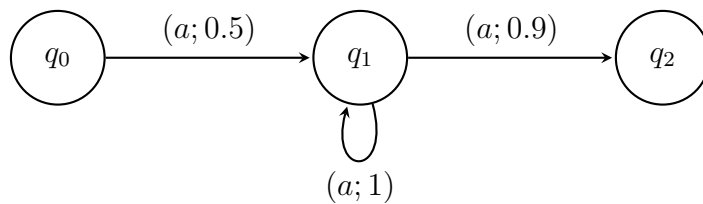


Figura 4 – Representação da máquina de estados *fuzzy* do Exemplo 3.2.1.

É possível, assim como no caso clássico com os AFD's e os AFN's, usar grafos de transição para representar as máquinas de estado *fuzzy*, em que os estados da máquina serão os vértices do grafo, como no caso clássico as aresta entre dois vértices q_i e q_j serão rotuladas, no caso das máquinas de estado *fuzzy* este rótulo corresponde ao par ordenado $(a; x)$ onde $a \in \Sigma$ e $x \in [0, 1]$ representado a equação $\varphi(q_i, a, q_j) = x$, é comum como dito por (COSTA; BEDREGAL, 2020), omitir dos grafos as arestas onde $x = 0$, a Figura 4 representa uma máquina de estados *fuzzy*.

³ As máquinas de estados *fuzzy* têm a mesma arquitetura da descrição informal dos autômatos finitos apresentados na Seção 2.1.

Observação 3.2.2. Note que diferente de um AFD (ou AFN) uma máquina de estado fuzzy não possui um estado destacado para ser o estado inicial da máquina e nem um conjunto de estados finais. Toda computação em uma máquina de estados fuzzy é uma tripla (q, w, q') , em que q será enxergado como estado inicial, q' o estado final e $w \in \Sigma^*$ é a palavra de entrada.

O valor de pertinência de uma computação (q_i, w, q_j) acontecer em um máquina de estados fuzzy, é igual ao valor de pertinência da tripla (q_i, w, q_j) como relação ao conjunto fuzzy $\hat{\varphi}$.

Exemplo 3.2.2. Considerando a máquina de estados fuzzy da Figura 4, uma computação do estado q_0 para o estado q_1 dada a palavra de entrada aa , tem o valor de pertinência dado por:

$$\begin{aligned}\hat{\varphi}(q_0, aa, q_1) &= \bigvee \left\{ \hat{\varphi}(q_0, a, q_0) \wedge \varphi(q_0, a, q_1), \right. \\ &\quad \hat{\varphi}(q_0, a, q_1) \wedge \varphi(q_1, a, q_1), \\ &\quad \left. \hat{\varphi}(q_0, a, q_2) \wedge \varphi(q_2, a, q_1) \right\} \\ &= \bigvee \left\{ \hat{\varphi}(q_0, a, q_0) \wedge 0.5, \hat{\varphi}(q_0, a, q_1) \wedge 1, \hat{\varphi}(q_0, a, q_2) \wedge 0 \right\}\end{aligned}$$

agora calculando $\hat{\varphi}(q_0, a, q_0)$, $\hat{\varphi}(q_0, a, q_1)$ e $\hat{\varphi}(q_0, a, q_2)$ tem-se que,

$$\begin{aligned}\hat{\varphi}(q_0, a, q_0) &= \bigvee \left\{ \hat{\varphi}(q_0, \lambda, q_0) \wedge \varphi(q_0, a, q_0), \right. \\ &\quad \hat{\varphi}(q_0, \lambda, q_1) \wedge \varphi(q_1, a, q_0), \\ &\quad \left. \hat{\varphi}(q_0, \lambda, q_2) \wedge \varphi(q_2, a, q_0) \right\} \\ &= \bigvee \left\{ 1 \wedge 0, 0 \wedge 0, 0 \wedge 0 \right\} \\ &= \bigvee \left\{ 0 \right\} = 0\end{aligned}$$

$$\begin{aligned}\hat{\varphi}(q_0, a, q_1) &= \bigvee \left\{ \hat{\varphi}(q_0, \lambda, q_0) \wedge \varphi(q_0, a, q_1), \right. \\ &\quad \hat{\varphi}(q_0, \lambda, q_1) \wedge \varphi(q_1, a, q_1), \\ &\quad \left. \hat{\varphi}(q_0, \lambda, q_2) \wedge \varphi(q_2, a, q_1) \right\} \\ &= \bigvee \left\{ 1 \wedge 0.5, 0 \wedge 1, 0 \wedge 0 \right\} \\ &= \bigvee \left\{ 0.5, 0 \right\} = 0.5\end{aligned}$$

$$\begin{aligned}\hat{\varphi}(q_0, a, q_2) &= \bigvee \left\{ \hat{\varphi}(q_0, \lambda, q_0) \wedge \varphi(q_0, a, q_2), \right. \\ &\quad \hat{\varphi}(q_0, \lambda, q_1) \wedge \varphi(q_1, a, q_2), \\ &\quad \left. \hat{\varphi}(q_0, \lambda, q_2) \wedge \varphi(q_2, a, q_2) \right\} \\ &= \bigvee \left\{ 1 \wedge 0, 0 \wedge 0.9, 0 \wedge 0 \right\} \\ &= \bigvee \left\{ 0 \right\} = 0\end{aligned}$$

substituindo esse valores na equação de $\hat{\varphi}(q_0, aa, q_1)$ tem-se que,

$$\begin{aligned}\hat{\varphi}(q_0, aa, q_1) &= \bigvee \{0 \wedge 0.5, 0.5 \wedge 1, 0 \wedge 0\} \\ &= \bigvee \{0, 0.5\} = 0.5\end{aligned}$$

assim o valor de pertinência da computação da palavra *aa* iniciada no estado q_0 e finalizada q_1 é igual 0.5.

Observação 3.2.3. *O interesse sobre as máquinas de estados fuzzy é puramente algébrico (MORDESON; MALIK, 2002), apesar de que obviamente tais máquinas computem conjuntos fuzzy, não existe formalmente uma noção de linguagem para máquinas de estado fuzzy.*

Tendo apresentado a noção de máquina de estado fuzzy, agora é possível generalizar esse conceito para a ideia de autômato fuzzy como se segue.

Definição 3.2.3. (KOMEJWAR, 2012) *Um autômato fuzzy é uma quintupla na forma $M = \langle Q, \Sigma, \varphi, \mathbb{I}, \mathbb{F} \rangle$ em que:*

- $\langle Q, \Sigma, \varphi \rangle$ é uma máquina de estados fuzzy;
- $\mathbb{I} : Q \rightarrow [0, 1]$ é um conjunto fuzzy de estados iniciais e
- $\mathbb{F} : Q \rightarrow [0, 1]$ é um conjunto fuzzy de estados finais.

Como dito em (KOMEJWAR, 2012), esta definição de autômato fuzzy coincide como a definição apresentada por Santos em (SANTOS, 1968a), a única diferença entre elas diz respeito ao universo semântico (ou interpretativo), pois para Santos as mudanças entre estados são eventos probabilísticos, já para Komejwar eles são eventos fuzzy.

De forma sucinta, nos autômatos de Santos pelas transições serem eventos probabilísticos existe a chance das **transições não ocorrerem**, já nos autômatos de Komejwar pelas transições serem eventos fuzzy tem-se que **as transições sempre acontecem**, entretanto, com certos valores de pertinência, indicando que o próximo estado assumido pelo autômato, após a transição, pode ser assumido de forma parcial. Para mais detalhes sobre as diferenças entre eventos probabilísticos e fuzzy ver (COSTA, 2012).

Observação 3.2.4. *Como já mencionado as máquinas de estados fuzzy modelam erros de transição, já os autômatos fuzzy por possuírem conjuntos fuzzy de estados iniciais e finais, modelam os erros já mencionados anteriormente de inicialização e finalização. Além disso, uma vez que, os autômatos fuzzy são casos especiais de máquina de estados fuzzy, eles também são capazes de modelar os erros de transição.*

É fácil notar pela definição de autômatos *fuzzy*, que eles são mais próximos de serem extensões dos AFN's que dos AFD's, de fato, para um autômato *fuzzy* mais próximo de estender os AFD's ver as referências (BĚLOHLÁVEK, 2002; MORDESON; MALIK, 2002). Note também que a Definição 3.2.1 proíbe a existência de λ -transições não nulas entre dois estados distintos, para autômatos *fuzzy* como λ -transições entre estados distintos ver (ASTRAIN; MENDÍVIL; GARITAGOITIA, 2006; MACIEL, 2005).

A única diferença entre a representação das máquinas de estados *fuzzy* e os autômatos *fuzzy* usando grafos de transição é que, os vértices da representação dos autômatos *fuzzy* são rotulados por triplas ordenadas da forma $(q; x; y)$, onde q é o rótulo do estado do autômato *fuzzy* e $x, y \in [0, 1]$ tal que $\mathbb{I}(q) = x$ e $\mathbb{F}(q) = y$, um exemplo de autômato *fuzzy* usando a representação por grafos de transição pode ser observado na Figura 5.

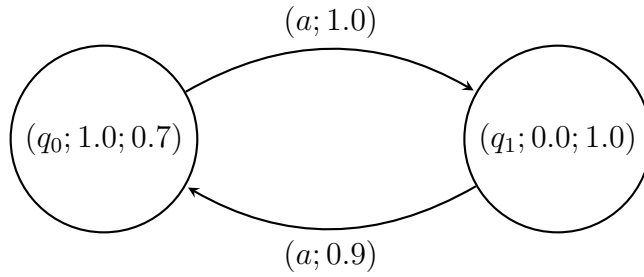


Figura 5 – Exemplo de um autômato *fuzzy*.

Como explicado em (MALIK; MORDESON, 1996), os autômatos *fuzzy* são capazes de computar conjuntos *fuzzy* sobre um dado alfabeto, isto é, autômatos *fuzzy* computam linguagens *fuzzy* (LEE; ZADEH, 1969).

Definição 3.2.4. (KOMEJWAR, 2012) *Seja $M = \langle Q, \Sigma, \varphi, \mathbb{I}, \mathbb{F} \rangle$ um autômato fuzzy. A linguagem aceita (ou computada) por M é o conjunto fuzzy $f_M : \Sigma^* \rightarrow [0, 1]$ definido por:*

$$f_M(x) = \bigvee_{q, p \in Q} \left\{ \mathbb{I}(q) \wedge \widehat{\varphi}(q, x, p) \wedge \mathbb{F}(p) \right\} \quad (3.9)$$

Observação 3.2.5. *Sempre que o contexto for claro é possível remover o M da indexação, e assim escrever apenas f em vez de f_M .*

Exemplo 3.2.3. *Não é difícil mostrar por indução sobre o tamanho das palavras que a linguagem computada pelo autômato fuzzy da Figura 5, corresponde ao seguinte conjunto fuzzy:*

$$f_M(w) = \begin{cases} 0.7, & \text{se } |w| \text{ é par} \\ 0.9, & \text{se } |w| > 1 \text{ e } |w| \text{ é ímpar} \\ 1.0, & \text{qualquer outro caso} \end{cases}$$

Komejwar provou em sua tese de doutorado (KOMEJWAR, 2012), os resultados a seguir, estabelecendo com eles alguns operadores de fecho algébrico para as linguagens dos autômatos *fuzzy*.

Teorema 3.2.1. (*KOMEJWAR, 2012*) Se $\mathbb{L}_1$ e $\mathbb{L}_2$ são linguagens fuzzy sobre um alfabeto Σ aceitas por algum autômatos fuzzy, então existe um autômato fuzzy M tal que $f_M = \mathbb{L}_1 + \mathbb{L}_2$.

Teorema 3.2.2. (*KOMEJWAR, 2012*) Se $\mathbb{L}_1$ e $\mathbb{L}_2$ são linguagens fuzzy sobre um alfabeto Σ aceitas por algum autômatos fuzzy, então existe um autômato fuzzy M tal que $f_M = \mathbb{L}_1 \cdot \mathbb{L}_2$.

Teorema 3.2.3. (*KOMEJWAR, 2012*) Se $\mathbb{L}_1$ e $\mathbb{L}_2$ são linguagens fuzzy sobre um alfabeto Σ aceitas por algum autômatos fuzzy, então existe um autômato fuzzy M tal que $f_M = \mathbb{L}_1 \times \mathbb{L}_2$.

Teorema 3.2.4. (*KOMEJWAR, 2012*) Se $\mathbb{L}_1$ e $\mathbb{L}_2$ são linguagens fuzzy sobre um alfabeto Σ aceitas por algum autômatos fuzzy, então existe um autômato fuzzy M tal que $f_M = \mathbb{L}_1 \mathbb{L}_2^{-1}$.

Teorema 3.2.5. (*KOMEJWAR, 2012*) Se $\mathbb{L}_1$ e $\mathbb{L}_2$ são linguagens fuzzy sobre um alfabeto Σ aceitas por algum autômatos fuzzy, então existe um autômato fuzzy M tal que $f_M = \mathbb{L}_2^{-1} \mathbb{L}_1$.

Teorema 3.2.6. (*KOMEJWAR, 2012*) Seja $\widehat{H}$ um homomorfismo estendido. Se $\mathbb{L}$ é uma linguagem fuzzy sobre um alfabeto Σ aceita por algum autômato fuzzy, então existe um autômato fuzzy M tal que $f_M = \mathbb{L}^{\widehat{H}}$.

Como explicado em (*MORDESON; MALIK, 2002*), a Definição 3.2.3 é um tanto quanto artificial, no sentido que, no momento inicial de processamento o autômato se encontra em um estado de sobreposição de estados iniciais, cada estado com seu próprio valor de pertinência de iniciar o processamento, sendo que não necessariamente a composição desses valores de pertinência será igual a 1 (isto é, a certeza total), assim fica-se a possibilidade de nunca ser possível ter certeza total (valor 1) de que um determinado autômato fuzzy de fato seja capaz de iniciar o processamento de uma determinada palavra de entrada.

Esta natureza incerta do início do processamento nos autômatos fuzzy é contraditória com a realidade do funcionamento dos computadores, pois ao processar algo, pelo menos no momento inicial tem-se total convicção que os computadores iniciam o processamento dos dados de entrada (*MORDESON; MALIK, 2002*). É natural exigir que os modelos de computação não convencionais (fuzzy, probabilísticos ou quânticos) também possuam essa característica de sempre ter certeza do início do processamento, uma vez que os modelos convencionais de computação são casos particulares dos modelos fuzzy, probabilísticos e quânticos.

A partir deste momento será considerado uma definição simplificada da formulação de autômato fuzzy, esta simplificação possui a característica de ter um único estado inicial, e este estado especial como na definição clássica de AFD e AFN, garante que o processamento do autômato fuzzy sempre seja iniciado. Aqui este tipo de autômato será chamado apenas de Autômato fuzzy simplificado, porém, na literatura ele poderá receber outros nomes, exemplo disto é o trabalho (*CAO; EZAWA, 2012*).

Definição 3.2.5. (*COSTA; BEDREGAL, 2020*) Um autômato fuzzy simplificado é uma quintupla na forma $M = \langle Q, \Sigma, \varphi, q_0, \mathbb{F} \rangle$ onde $Q, \Sigma, \varphi, \mathbb{F}$ são da mesma forma que na Definição 3.2.3 e $q_0 \in Q$ é o estado inicial.

Observação 3.2.6. (*COSTA; BEDREGAL, 2020*) A Definição 3.2.5 é um caso particular da Definição 3.2.3 com $\mathbb{I}(q_0) = 1$ e para todo $q \neq q_0$ tem-se $\mathbb{I}(q) = 0$.

A representação visual de um autômato fuzzy simplificado usando grafos de transição é similar a representação das máquinas de estados fuzzy, exceto que, na representação dos autômatos fuzzy simplificados os vértices são rotulados por $(q; y)$, onde $q \in Q$ e $y \in [0, 1]$ com $\mathbb{F}(q) = y$, além disso, como no caso clássico de AFD's e AFN's, o estado inicial é marcado no grafo com sendo o início dos processos de computação, um exemplo da representação visual de um autômato fuzzy simplificado pode ser visto na Figura 6.

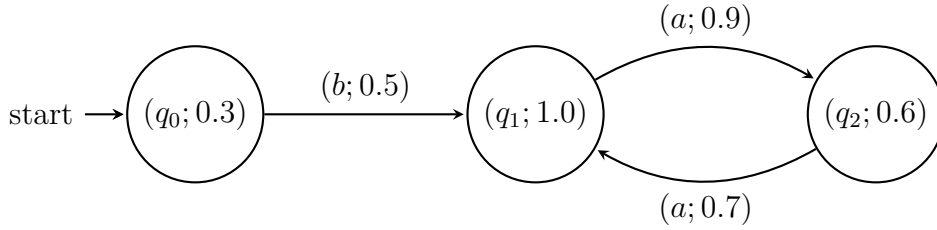


Figura 6 – Representação de um autômato fuzzy simplificado.

O grau de pertinência com que uma palavra $w \in \Sigma^*$ tem sua computação finalizada em um estado $q \in Q$ em um autômato fuzzy simplificado, corresponde ao valor do ínfimo entre os valores pertinência de $\hat{\varphi}$ aplicado à tripla (q_0, w, q) e do valor de $\mathbb{F}(q)$.

Exemplo 3.2.4. Seja M o autômato fuzzy simplificado da Figura 6, a pertinência da computação da palavra ba terminar no estado q_2 é igual a:

$$\hat{\varphi}(q_0, ba, q_2) \wedge \mathbb{F}(q_2) = 0.5 \wedge 0.6 = 0.5$$

Definição 3.2.6. (*COSTA; BEDREGAL, 2020*) Seja $M = \langle Q, \Sigma, \varphi, q_0, \mathbb{F} \rangle$ um autômato fuzzy simplificado a linguagem aceita (ou computada) pela autômato M é um conjunto fuzzy $f_M : \Sigma^* \rightarrow [0, 1]$ definido como se segue:

$$f_M(x) = \bigvee_{q \in Q} \left\{ \hat{\varphi}(q_0, x, q) \wedge \mathbb{F}(q) \right\} \quad (3.10)$$

Exemplo 3.2.5. Seja M o autômato fuzzy simplificado da Figura 6, usando indução sobre w é fácil concluir que a linguagem aceita por M corresponde ao seguinte conjunto fuzzy:

$$f_M(w) = \begin{cases} 0.3, & \text{se } w = \lambda \\ 0.5, & \text{se } w = ba^n \text{ com } n \geq 0 \\ 0, & \text{qualquer outro caso} \end{cases}$$

Observação 3.2.7. Como dito em (COSTA; BEDREGAL, 2020), os autômatos fuzzy e os autômatos fuzzy simplificados são equivalentes, no sentido que ambos reconhecem a mesma família de linguagens fuzzy, que aqui será denotada por $Ling_F$.

Desde que autômatos fuzzy e autômatos fuzzy simplificados são equivalentes, neste texto os autômatos fuzzy simplificados serão adotados como modelo padrão, assim deste ponto em diante, sempre que for mencionado o termo autômatos fuzzy estará sendo considerado o modelo simplificado.

3.3 Extensões da Teoria de Autômatos Fuzzy e uma Hierarquia de Linguagens

Como o passar das décadas diversas extensões dos conjuntos fuzzy têm sido apresentadas (BUSTINCE et al., 2016), em paralelo a isto, extensões para a teoria dos autômatos fuzzy também foram introduzidas, nesta seção, serão apresentadas algumas extensões da teoria dos autômatos fuzzy, e será tecido comentários sucintos sobre cada extensão.

Usando o conceito de conjunto fuzzy intervalar (BUSTINCE et al., 2016; RAVI; ALKA, 2010; PEKALA, 2018), Ravi e Alka em (RAVI; ALKA, 2010), introduzem a noção autômato fuzzy intervalar. Eles provam em seu trabalho que a família de linguagens aceitas por autômatos fuzzy intervalares coincide com a família de conjuntos fuzzy intervalares “escritos” por gramáticas regulares fuzzy intervalares. Formalmente um autômato fuzzy intervalar é definido como se segue.

Definição 3.3.1. (RAVI; ALKA, 2010) Um autômato fuzzy intervalar (ou simplesmente AFI) é uma quintupla $m = \langle Q, \Sigma, \vartheta, S, F \rangle$ onde Q, Σ, F são da mesma forma que na Definição 2.1.3, $S \subseteq Q$ é um conjunto não vazio de estados iniciais e $\vartheta : Q \times \Sigma \times Q \rightarrow I[0, 1]$ é um conjunto fuzzy intervalar de transições entre estados, onde $I[0, 1] = \{[a, b] \mid 0 \leq a \leq b \leq 1\}$.

Pela definição apresentada por Ravi e Alka, é possível notar que seu autômato foi projetado para ser capaz de modelar erros nas transições entre os estados do autômato, uma vez que a função de transição é um conjunto fuzzy intervalar, aqui não será discutido os méritos da representação de erros e incertezas pelas teoria fuzzy e a teoria fuzzy intervalar, para isso é recomendado a leitura dos trabalhos (SANTIAGO; BEDREGAL; ACIÓLY, 2006; BEDREGAL; TAKAHASHI, 2006).

Note também que assim como os autômatos fuzzy (Definição 3.2.3), os autômatos fuzzy intervalares podem iniciar seu processamento em uma sobreposição de estados, entretanto, ao contrario dos autômatos fuzzy onde a sobreposição inicial é feita para modelar erros de inicialização, nos autômatos fuzzy intervalares apresentados por Ravi e

Alka não fica claro o porquê desta possível sobreposição inicial, visto que no caso deles o conjunto de estados iniciais é clássico, assim não existe incerteza nesse momento inicial e nem modelagem de erro, no sentido descrito em (MORDESON; MALIK, 2002). Entretanto, vale ressaltar que um conjunto clássico de estados iniciais não é algo novo na teoria dos autômatos para detalhes ver (BEDREGAL, 2011; BEDREGAL, 2015).

Ainda em (RAVI; ALKA, 2010), é também introduzido um outro tipo de autômato fuzzy intervalar, esse por sua vez recebe o nome de Autômato Não-determinístico com Estados Fuzzy Intervalares (ou AFN-EFI).

Definição 3.3.2. (RAVI; ALKA, 2010) *Um AFN-EFI é uma quintupla com a forma $m = \langle Q, \Sigma, \delta_N, S, \tilde{F} \rangle$ onde: Q, Σ, δ_N são iguais a Definição 2.1.6, $S \subseteq Q$ é um conjunto não vazio de estados iniciais e $\tilde{F} : Q \rightarrow I[0, 1]$ é um conjunto fuzzy intervalar de estados finais.*

Este segundo autômato fuzzy intervalar apresentado por Ravi e Alka, substitui a modelagem de erros durante as transições, pela modelagem de erros de finalização, uma vez que, a diferença entre os dois tipos é apenas a questão de que o primeiro tipo tinha um conjunto fuzzy intervalar de transições, enquanto que neste segundo tipo o conjunto fuzzy intervalar diz respeito aos estados finais. Para este segundo tipo de autômato Ravi e Choubey em (RAVI; CHOUBEY, 2010), provaram uma extensão do teorema de Myhill-Nerode (MARTIN, 1991), estabelecendo assim a existência de um AFN-EFI mínimo.

Observação 3.3.1. *Vale a ressalva que os autômatos fuzzy intervalares e os autômatos não-determinísticos com estados fuzzy intervalares reconhecem exatamente a mesma família de linguagens, como mostrado pelo Teorema 4.1 em (RAVI; ALKA, 2010), tal família será aqui representada por $Ling_{FI}$.*

Wang, Yin e Gu em (WANG; YIN; GU, 2013), motivados pela generalização da teoria dos conjuntos apresentada pela teoria dos multiconjuntos (CARMO; GOUVEIA; DIONÍSIO, 2013), introduziram os autômatos fuzzy baseados em multiconjuntos. Uma vez que, multiconjuntos não são o foco desse capítulo, eles não serão tratados aqui, para um leitor que deseje uma introdução sobre o tema pode consultar o Apêndice A, antes de prosseguir para a Definição 3.3.3.

Definição 3.3.3. (WANG; YIN; GU, 2013) *Um autômato fuzzy baseado em multiconjunto (ou simplesmente AFMC) é uma quintupla na forma $\mathbb{M} = \langle Q, \overline{\Sigma}^\oplus, \beta, Q_I, Q_F \rangle$ onde: Q é da mesma forma que na Definição 2.1.3 e $\overline{\Sigma}^\oplus$ é a família de todos os multiconjuntos⁴ unitários sobre Σ , além disso, tem-se que:*

⁴ Para detalhes sobre multiconjuntos ver o Apêndice A ou consultar (CARMO; GOUVEIA; DIONÍSIO, 2013).

- $\beta : Q \times (\overline{\Sigma^\oplus} \cup \{\lambda\}) \times Q \rightarrow [0, 1]$ é um conjunto fuzzy de transição entre estados;
- $Q_I : Q \rightarrow [0, 1]$ é um conjunto fuzzy de estados iniciais e
- $Q_F : Q \rightarrow [0, 1]$ é um conjunto fuzzy de estados finais.

A arquitetura do autômato apresentado em (WANG; YIN; GU, 2013), possui a característica de ser capaz de modelar os três tipos de erros, isto é, erros de inicialização, finalização e transição, isto acontece porquê tal tipo de autômato possui conjuntos fuzzy de estados iniciais, finais e de transição entre estados, além disso, as instruções fuzzy em um AFMC podem ser λ -instruções entre estados distintos.

Definição 3.3.4. (WANG; YIN; GU, 2013) Um AFMC $\mathbb{M}$ computa um conjunto fuzzy com a assinatura $f_{\mathbb{M}} : \Sigma^\oplus \rightarrow [0, 1]$, com $\Sigma^\oplus$ sendo a família de todos os multiconjuntos sobre Σ .

Observação 3.3.2. A família de linguagens aceitas por autômatos fuzzy baseados em multiconjuntos aqui será denotada por, $Ling_{MF}$.

Quanto aos resultados obtidos para esse tipo de autômatos pode-se destacar, a prova da equivalência entre os autômatos fuzzy baseados em multiconjuntos e as gramáticas baseadas em multiconjuntos fuzzy (WANG; YIN; GU, 2013). Depois em (TIWARI; GAUTAM; DUBEY, 2016), são apresentados os autômatos fuzzy baseados em multiconjuntos determinísticos e a prova que eles são equivalentes aos autômatos fuzzy baseados em multiconjuntos padrão (Definição 3.3.3).

Depois Sharma *et al.* em (SHARMA et al., 2017), apresentaram as condições suficientes e necessárias para que os conjuntos fuzzy computados por autômatos fuzzy baseados em multiconjuntos não sejam funções constantes. Além disso, em (SHARMA; TIWARI; SHARAN, 2016), foram investigadas as propriedades básicas de fechos algébricos para as linguagens computadas pelos autômatos fuzzy baseados em multiconjuntos.

Observação 3.3.3. Como mostrado no Apêndice A todo $f : \Sigma^\oplus \rightarrow [0, 1]$ é isomórfico a um $f' : \Sigma^* \rightarrow [0, 1]$, além disso, como todo multiconjunto unitário de Σ consiste de um conjunto com apenas um elemento de Σ , pode-se transformar qualquer AFMC em um autômato fuzzy com λ -instruções (ASTRAIN; MENDÍVIL; GARITAGOITIA, 2006), apenas alterando na definição $\overline{\Sigma^\oplus}$ por Σ e atribuindo as λ -instruções ao conjunto fuzzy μ_ε da definição do autômato fuzzy apresentada em (ASTRAIN; MENDÍVIL; GARITAGOITIA, 2006). E como as λ -instruções não aumentam o poder computacional dos autômatos fuzzy (ASTRAIN; MENDÍVIL; GARITAGOITIA, 2006), as famílias de linguagens aceitas pelos dois tipos de autômatos fuzzy baseados em multiconjuntos e os autômatos fuzzy padrão são isomórficas ($\cong$), isto é, $Ling_{MF} \cong Ling_F$.

Outra extensão de autômatos *fuzzy* é a aquela baseada em conjuntos *fuzzy* intuicionistas (BUSTINCE et al., 2016). Um autômato finito não-determinístico como transições *fuzzy* intuicionistas (ou simplesmente AFN-TFI) é um autômato com a característica de que cada uma de suas transições é valorada por um par ordenado $(x, y) \in [0, 1] \times [0, 1]$, onde x denota o valor de pertinência da transição acontecer e y denota o valor de pertinência da transição não ocorrer (CHOUBEY; RAVI, 2009), assim é evidente que a arquitetura de um AFN-TFI é pensada para modelar erros de transição entre os estados, formalmente um AFN-TFI é definido como se segue.

Definição 3.3.5. (CHOUBEY; RAVI, 2009) *Um AFN-TFI é uma quintupla na forma $m = \langle Q, \Sigma, \mathcal{T}, T, q_0, F \rangle$ onde Q, q_0, Σ, F são da mesma forma que na Definição 2.1.3, e o par $(\mathcal{T}, T)$ são conjuntos fuzzy sobre $Q \times \Sigma \times Q$ que juntos formam o conjunto fuzzy intuicionista de transições.*

Observação 3.3.4. *Como dito em (CHOUBEY; RAVI, 2009), a linguagem de qualquer AFN-TFI corresponde a um conjunto fuzzy intuicionista sobre Σ^* , aqui será denotado a família de todas as linguagens aceitas por autômatos fuzzy intuicionistas por $Ling_{FI_n}$.*

Quanto aos resultados usando AFN-TFI, Choubey e Ravi em (CHOUBEY; RAVI, 2009), apresentaram uma série de operadores de fecho sobre as linguagens *fuzzy* intuicionista. Depois em (SRIVASTAVA; TIWARI, 2010), é realizado um estudo investigativo sobre o relacionamento entre topologias *fuzzy* intuicionistas e os AFN-TFI.

Observação 3.3.5. *Antes de prosseguir vale ressaltar que, como dito em (TIZHOOSH, 2008), os conjuntos fuzzy intervalares e os conjuntos fuzzy intuicionistas são isomórficos ($\cong$), ou seja, as linguagens fuzzy intervalares ($Ling_{FI}$) e as linguagens fuzzy intuicionistas ($Ling_{FI_n}$) são também isomórficas.*

Por fim, a generalização apresentada por Qiu em seu seminal trabalho (QIU, 2001), e depois em (WU; QIU; XING, 2002), foi inspirado pelo trabalho de Gougen (GOGUEN, 1967), que generalizou a ideia de conjunto *fuzzy* apresentado os conjuntos $\mathcal{L}$ -fuzzy, Qiu apresenta a ideia de autômato $\mathcal{L}$ -valorado, onde $\mathcal{L}$ é um reticulado completo residuado (FARIAS et al., 2016b), e por consequência, também apresenta a ideia de linguagem $\mathcal{L}$ -valorada.

Definição 3.3.6. (QIU, 2001) *Seja $\langle \mathcal{L}, \vee, \wedge, \odot, \star \rangle$ um reticulado completo residuado. Um autômato $\mathcal{L}$ -valorado é uma estrutura $M = \langle Q, \Sigma, \delta^* \rangle$ onde: Q e Σ são da mesma forma que na definição 2.1.3 e $\delta^* : Q \times \Sigma \times Q \rightarrow \mathcal{L}$ é um conjunto $\mathcal{L}$ -fuzzy.*

Como dito em (QIU, 2007), esta generalização é extremamente forte porque generaliza praticamente todos os tipos autômatos, seja de forma direta ou via homomorfismo. Entretanto, algumas generalização da teoria dos autômatos e das linguagens formais não

precisam de toda a força do arcabouço matemático da teoria de reticulados residuados como apontado em (COSTA; BEDREGAL; SANTIAGO, 2020), assim mesmo que a teoria dos autômatos $\mathcal{L}$ -valorados generalize boa parte das demais teorias, o estudo específico das outras teorias ainda é válido, no sentido de que, essas teorias apresentam as exigências mínimas estruturais (em termos de álgebra), para a realização dos processos de computação.

Observação 3.3.6. *A família das linguagens aceitas por autômatos $\mathcal{L}$ -valorados aqui será denotada por $Ling_{\mathcal{L}}$.*

Entre os resultados já demonstrados para os autômatos $\mathcal{L}$ -valorados pode-se mencionar, o conceito de equivalência entre autômatos $\mathcal{L}$ -valorados provado em (XING et al., 2007), o lema do bombeamento $\mathcal{L}$ -valorado (QIU, 2006) e as condições para a minimização de autômatos $\mathcal{L}$ -valorados demonstradas em (WU; QIU, 2010).

A teoria dos autômatos $\mathcal{L}$ -valorados obteve tanto sucesso que tem se tornado o modelo padrão para os estudos de computabilidade *fuzzy* (WIEDERMANN, 2004; LI, 2006; BEDREGAL; FIGUEIRA, 2006; BEDREGAL; FIGUEIRA, 2008; FARIAS et al., 2016c). Diversos modelos de máquinas de computação vem ganhando versões baseadas nessa teoria, por exemplos, as máquinas de Mealy e Moore (LI; PEDRYCZ, 2006; WU; QIU, 2010), autômatos de pilha (QIU, 2007) e principalmente máquinas de Turing (LI, 2009; WU; QIU; XING, 2012).

Baseado nos comentários feitos acima sobre as diferentes extensões da teoria dos autômatos *fuzzy* e de suas respectivas linguagens, e inspirado no relacionamento de inclusão apresentado por Lima em (LIMA, 2019), entre as diferentes teorias que generalizam os conjuntos *fuzzy*, é possível apresentar uma hierarquia de linguagens entre as linguagens aceitas pelas diferentes extensões dos autômatos *fuzzy*, a Figura 7 a seguir ilustra essa hierarquia.

3.4 Aplicações de Autômatos Fuzzy

O interesse pelos autômatos *fuzzy*, e também o motivo que marcou o início da teoria, se dá pela possibilidade do uso dos autômatos *fuzzy* para aplicações práticas como revela a referência inicial da área, isto é, a tese de Wee (WEE, 1967). Nesta seção, serão listados alguns trabalhos que mostram a viabilidade do uso de autômatos *fuzzy* em aplicações práticas, e como poderá ser observado aqui, a área de reconhecimento de padrões e o aprendizado de máquina são os principais campos de aplicação dos autômatos *fuzzy*.

Para iniciar, Girijamma em sua tese de doutorado (GIRIJAMMA, A., 2010), detalha duas aplicações possíveis, a primeira aplicação consiste em comparar duas palavras de entrada, isto é, verificar a semelhanças entre elas. Portanto, autômatos *fuzzy* são máquinas

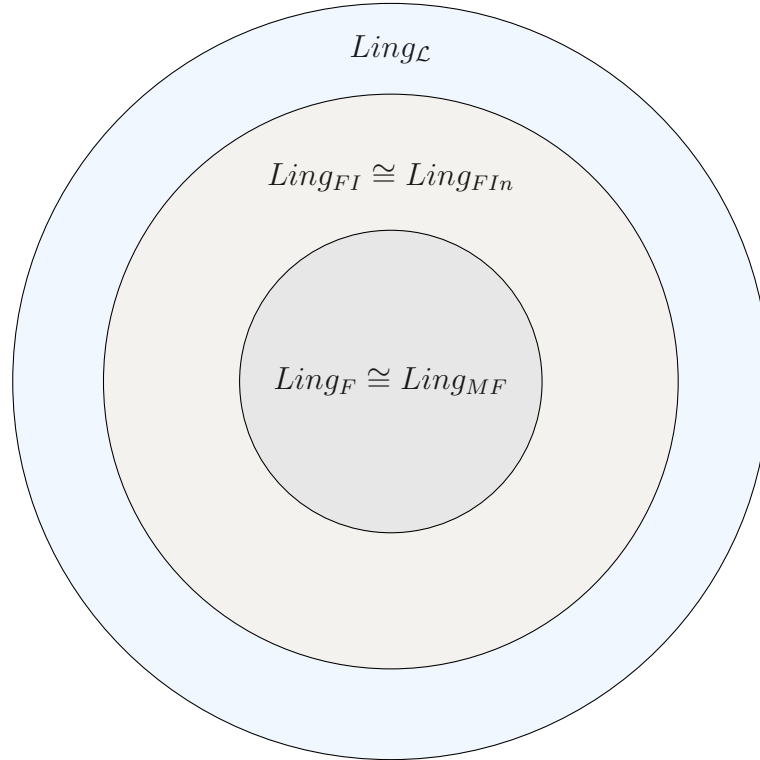


Figura 7 – Hierarquia entre as linguagens aceitas por autômatos *fuzzy* e suas extensões.

capazes de reconhecer, e de certa forma, medir a distância (ou similaridades) sobre objetos que podem ser descritos na forma de palavras, outro trabalho que se dedicou a esse estudo foi a Dissertação de mestrado apresentada por Maciel ([MACIEL, 2005](#)).

Além disso, Girijamma ainda em ([GIRIJAMMA, A., 2010](#)), descreve uma variação do algoritmo de busca A^* , que foi implementado para realizar buscas em um dicionário de palavras. Nesta aplicação foi construído um autômato *fuzzy* para realizar a busca no dicionário, a fim de, encontrar uma palavra procurada, por essa aplicação tem-se que os autômatos *fuzzy* podem ser usados para solucionar problemas de busca.

Na área das telecomunicações, mais especificamente no campo de desenvolvimento e monitoramento de sistemas tolerante a falhas, Rigatos demonstrou no seu trabalho ([RIGATOS, 2009](#)), como construir um modelo de sistema capaz de detectar e isolar falhas usando autômatos *fuzzy*.

Já em ([STEIMANN; ADLASSNIG, 1994](#)), usando autômatos *fuzzy*, foi possível modelar um *framework* para um monitor inteligente de cabeceira. Este monitor foi então capaz de derivar uma abstração do estado atual do paciente, e também foi mostrado que esse monitor é capaz de antecipar possíveis pioras no quadro de saúde do paciente monitorado, dessa forma autômatos *fuzzy* são capazes de detectar padrões de mudança.

Mais recentemente, Ko *et. al.* em seu trabalho ([KO; HAM; NAM, 2011](#)), propõe um novo método para detecção de chamadas, seu método usa autômatos *fuzzy* para reconhecer padrões de chamadas em vídeos, esta abordagem mostra a capacidade de autômatos *fuzzy*

para processar imagens e reconhecer padrões visuais.

Em todas as aplicações que foram descritas aqui, os autômatos *fuzzy* sempre são usados para detectar (ou reconhecer) algum tipo de padrão, outros trabalhos que seguem esta linha de aplicação são (THOMASON, 1973; BAILADOR; TRIVIÑO, 2010). Para um leitor que deseja se aprofundar nos campos de aplicação dos autômatos *fuzzy*, é fortemente recomendado a leitura do livro escrito por Malik e Mordeson (MORDESON; MALIK, 2002).

3.5 Considerações Finais

Este capítulo foi iniciado retornando as definições básicas das linguagens *fuzzy* como definidas por Lee e Zadeh em seu seminal trabalho (LEE; ZADEH, 1969), em seguida foram apresentados algumas operações já conhecidas sobre as linguagens *fuzzy* (KOMEJWAR, 2012). Após esses tópicos, foi realizada uma breve descrição conceitual da noção de autômato *fuzzy*.

Após a descrição conceitual, foi apresentada formalmente a noção de máquina de estado *fuzzy* e de seu processo de computação, nos mesmos critérios descritos em (MORDESON; MALIK, 2002). A partir da definição de máquina de estados *fuzzy* foram então definidos os autômatos *fuzzy*, e logo em seguida, de forma similar a (MALIK; MORDESON, 1996; MORDESON; MALIK, 2002), foi definida a noção de linguagens *fuzzy* aceitas (ou computadas) por autômatos *fuzzy*. Além disso, foi apresentado alguns operadores de fechos algébrico para as linguagens *fuzzy* aceitas por autômatos *fuzzy*, seguindo os resultados obtidos em (KOMEJWAR, 2012), depois da apresentação destes resultados, foi introduzido o modelo simplificado dos autômatos *fuzzy*, como descrito em (COSTA; BEDREGAL, 2020).

Continuando o capítulo, foram apresentadas os modelos de autômatos *fuzzy* intervalares (RAVI; ALKA, 2010; RAVI; CHOUBEY, 2010), autômatos *fuzzy* baseados em multiconjuntos (WANG; YIN; GU, 2013; TIWARI; GAUTAM; DUBEY, 2016), autômatos *fuzzy* intuicionista (CHOUBEY; RAVI, 2009), e por fim, o modelo de autômatos $\mathcal{L}$ -valorados (QIU, 2001). Então inspirado pelo trabalho de Lima (LIMA, 2019), foi esboçada uma hierarquia entre as extensões da teoria dos autômatos *fuzzy* e suas respectivas linguagens.

E como último ato para finalizar o capítulo, foram apresentados uma série de trabalhos onde os autômatos *fuzzy* são aplicados na prática, isto é, fora do campo teórico. Em especial foi chamado a atenção para o poder dos autômatos *fuzzy* em reconhecer padrões.

Parte II

Contribuições

4 Uma Teoria de Autômatos *Fuzzy* Hesitantes Típicos

Este capítulo apresenta a primeira contribuição desta tese, inicialmente será apresentada a estrutura matemática necessária para a definição das diferentes variações dos autômatos *fuzzy* hesitantes típicos, em seguida, são introduzidos os autômatos em si e serão traçados os relacionamentos entre os diferentes tipos de autômatos *fuzzy* hesitantes típicos, além disso, será apresentada uma caracterização para a família de linguagens *fuzzy* hesitantes típicas $\mathbb{H}$ -regulares.

4.1 Combinações e Ordem sobre EFHT's

Como explicado na Seção 2.4, diversos operadores já foram definidos sobre os conjuntos *fuzzy* hesitantes típicos, esses operadores são então definidos sobre funções, pois como explicado em (RODRÍGUEZ et al., 2014), os conjuntos *fuzzy* hesitantes típicos são funções, aqui entretanto, serão apresentados dois novos operadores sobre os EFHT's, isto é, serão definidos dois operadores sobre o conjunto $\mathbb{H}$. Mais adiante, será dada uma explicação em termos operacionais do comportamento dos operadores que serão aqui definidos.

Definição 4.1.1. (*SANTOS et al., 2015*) *Seja $X \in \mathbb{H}$, uma ordenação crescente de X é uma função $\sigma_X : \mathbb{Z}_X^* \rightarrow X$ onde $\mathbb{Z}_X^* = \{1, \dots, \#X\}$ tal que $\sigma_X(i) < \sigma_X(i+1)$ para todo $1 \leq i \leq \#X - 1$.*

Exemplo 4.1.1. *Seja $X = \{0.4, 0.8, 0.2, 0.1, 0.99, 0.23\}$ tem-se então que:*

$$\begin{array}{ll} \sigma_X(1) = 0.1 & \sigma_X(4) = 0.4 \\ \sigma_X(2) = 0.2 & \sigma_X(5) = 0.8 \\ \sigma_X(3) = 0.23 & \sigma_X(6) = 0.99 \end{array}$$

Agora será formalmente definida a inf-combinação para, em seguida, apresentar alguns resultados acerca das propriedades deste operador.

Definição 4.1.2. *O operador $\otimes : \mathbb{H} \times \mathbb{H} \rightarrow \mathbb{H}$ definido para todo $X, Y \in \mathbb{H}$ como sendo:*

$$X \otimes Y = \{x \wedge y \mid x \in X, y \in Y\} \quad (4.1)$$

é chamado de inf-combinação.

Exemplo 4.1.2. *Dado os elementos fuzzy hesitantes típicos $X = \{0.4, 1, 0.9, 0.99\}$ e $Y = \{0.12, 0.7\}$ tem-se que, $X \otimes Y = \{0.12, 0.4, 0.7\}$.*

A interpretação operacional da inf-combinação é que tal operador tem o comportamento de um combinador do tipo filtro passa baixa (DINIZ; SILVA; NETTO, 2014; GONZALEZ; WOODS, 2009) com respeito aos elementos de $\mathbb{H}$, isto é, ele pega dois EFHT's X e Y e realiza a combinação dos dois em um novo EFHT através da passagem do componente de menor intensidade (ou significância) para cada par $(x, y) \in X \times Y$, a Figura 8 ilustra essa ideia da passagem dos pontos com menor intensidade.

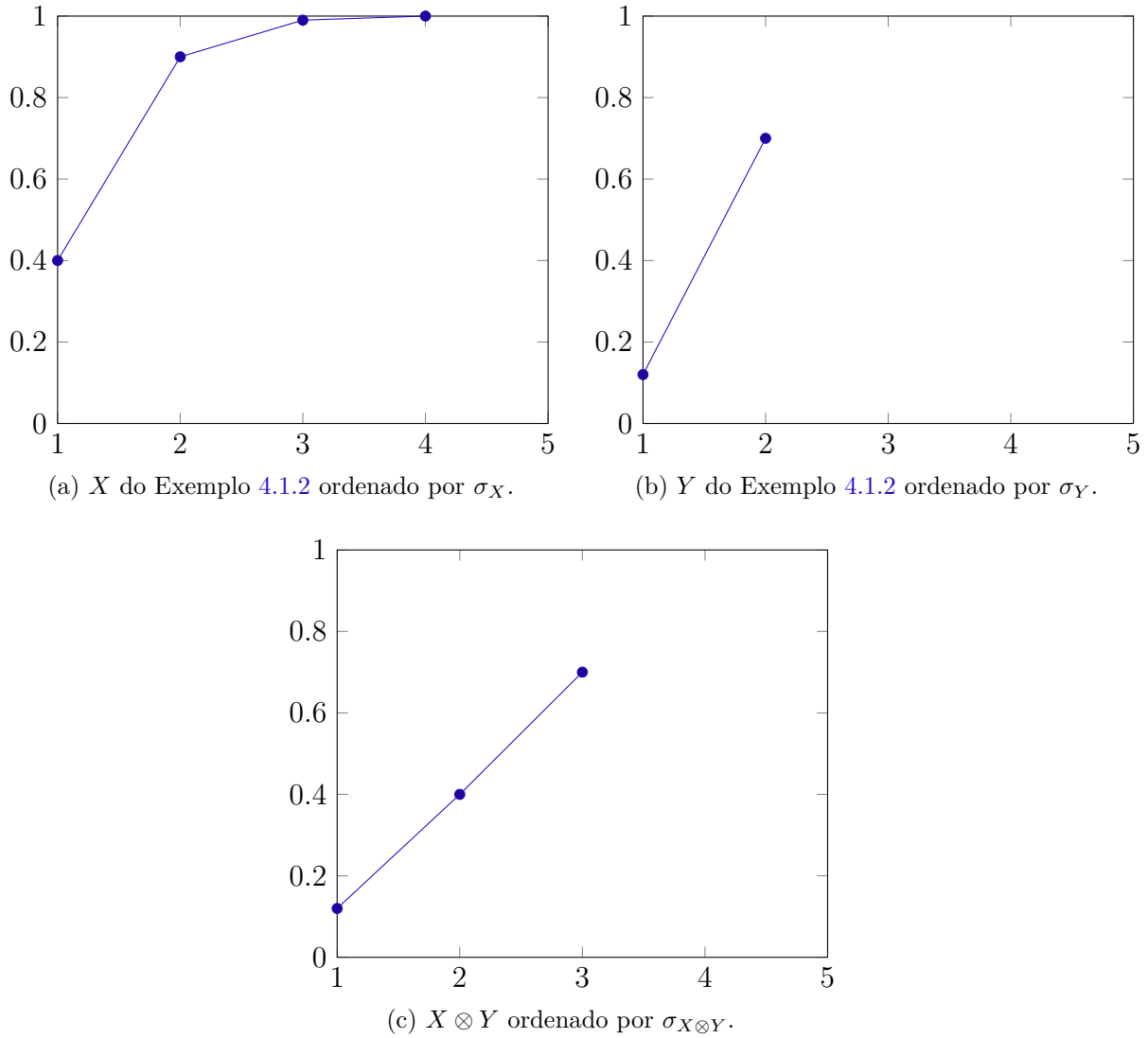


Figura 8 – Gráfico para a interpretação operacional da inf-combinação como um filtro passa baixa.

Agora note que, pelo fato do ínfimo ser comutativo, associativo e idempotente (enquanto operador algébrico), essas propriedades são herdadas por extensão para a inf-combinação, logo tem-se a seguinte proposição.

Proposição 4.1.1. *O operador $\otimes$ é comutativo, associativo e idempotente.*

Demonstração. Direto da definição de $\otimes$. □

Além disso, o operador de inf-combinação apresenta a seguinte característica.

Proposição 4.1.2. *O elemento degenerado $\{1\}$ é o elemento neutro de $\otimes$.*

Demonstração. Pela Observação 2.3.1 é claro que 1 é o máximo do reticulado $[0,1]$, agora por (L2) segue para todo $x \in [0, 1]$ que, $x \wedge 1 = 1 \wedge x = x$, portanto, para todo $X \in \mathbb{H}$, pela definição da inf-combinação, é claro que,

$$X \otimes \{1\} = \{1\} \otimes X = X \quad (4.2)$$

e portanto, $\{1\}$ é o elemento neutro de $\otimes$. $\square$

Como uma consequência dos resultados das Proposições 4.1.1 e 4.1.2 tem-se que, o conjunto dos elementos *fuzzy* hesitantes típicos pode ser visto como um monoide.

Corolário 4.1.1. *O conjunto $\mathbb{H}$ munido com o operador $\otimes$ é um monoide¹ comutativo idempotente.*

Além disso tem-se que, o elemento $\{0\}$ também apresenta uma propriedade especial para a inf-combinação.

Proposição 4.1.3. *O elemento degenerado $\{0\}$ é o elemento aniquilador (ou de absorção) do operador $\otimes$.*

Demonstração. Pela Observação 2.3.1 é claro que 0 é o mínimo do reticulado $[0,1]$, agora por (L4) é óbvio para todo $x \in [0, 1]$ que, $x \wedge 0 = 0 \wedge x = 0$, portanto, para todo $X \in \mathbb{H}$ pela definição da inf-combinação tem-se que,

$$X \otimes \{0\} = \{0\} \otimes X = \{0\} \quad (4.3)$$

e portanto, $\{0\}$ é o elemento aniquilador de $\otimes$. $\square$

Lema 4.1.1. *Seja $X, Y, Z \in \mathbb{H}$. Se $X \subseteq Y$, então $(X \otimes Z) \subseteq (Y \otimes Z)$.*

Demonstração. Suponha que $X \subseteq Y$, assim por definição de subconjunto para todo $x \in X$ tem-se que $x \in Y$, assim para qualquer $x \wedge z \in (X \otimes Z)$ tem-se que, $x \wedge z \in (Y \otimes Z)$, e portanto, $(X \otimes Z) \subseteq (Y \otimes Z)$. $\square$

O resultado do Lema 4.1.1 mostra que $\otimes$ preserva a ordem da inclusão na primeira componente, e pela comutatividade de $\otimes$ também vale para a segunda componente, o Lema 4.1.1 é generalizado pela Proposição 4.1.4 a seguir.

¹ Um monoide é uma estrutura algébrica $\langle A, \star, e \rangle$ onde A é um conjunto não vazio, $\star$ é uma operação binária associativa sobre A e e é o elemento neutro da operação $\star$, no caso de $\star$ ser comutativo então o monoide é chamado de monoide comutativo.

Proposição 4.1.4. *Sejam $X_1, X_2, Y_1, Y_2 \in \mathbb{H}$. Se $X_1 \subseteq Y_1$ e $X_2 \subseteq Y_2$, então $(X_1 \otimes X_2) \subseteq (Y_1 \otimes Y_2)$.*

Demonstração. Direto do Lema 4.1.1. □

Como consequência da Propriedade 4.1.4 tem-se o seguinte resultado.

Corolário 4.1.2. *Seja $X \subseteq Y$, então $X \subseteq (X \otimes Y)$.*

Demonstração. Suponha que $X \subseteq Y$, assim pelo Teorema 4.1.4 é claro que $(X \otimes X) \subseteq (X \otimes Y)$, mas pela Proposição 4.1.1 vale que $(X \otimes X) = X$, portanto, $X \subseteq (X \otimes Y)$. □

Agora será apresentado o operador dual da inf-combinação, tal operador irá receber o nome de sup-combinação, e é definido formalmente como segue.

Definição 4.1.3. *O operador $\sqcup : \mathbb{H} \times \mathbb{H} \rightarrow \mathbb{H}$ definido para todo $X, Y \in \mathbb{H}$ como:*

$$X \sqcup Y = \{x \vee y \mid x \in X, y \in Y\} \quad (4.4)$$

é chamado de sup-combinação.

Exemplo 4.1.3. *Dado $X = \{0.4, 1, 0.9, 0.99\}$ e $Y = \{0.12, 0.8, 0.7, 0.33\}$ tem-se que, $X \sqcup Y = \{0.4, 0.7, 0.8, 1, 0.9, 0.99\}$.*

A interpretação operacional para o operador de sup-combinação é que tal operador se comporta para elementos de $\mathbb{H}$ como um combinador do tipo filtro passa alta (DINIZ; SILVA; NETTO, 2014; GONZALEZ; WOODS, 2009), isto é, ele pega dois EFHT's X e Y , e combina os dois em um novo EFHT através da passagem do componente de maior intensidade de cada par $(x, y) \in X \times Y$, a Figura 9 ilustra essa visão operacional de passagem dos componentes com maior intensidade realizada pelo operador de sup-combinação.

Proposição 4.1.5. *$\sqcup$ é comutativo, associativo e idempotente.*

Demonstração. Similar à demonstração da Proposição 4.1.1. □

Proposição 4.1.6. *O elemento degenerado $\{0\}$ é o elemento neutro de $\sqcup$.*

Demonstração. Similar à demonstração da Proposição 4.1.2. □

Corolário 4.1.3. *O conjunto $\mathbb{H}$ munido com o operador $\sqcup$ é um monoide comutativo idempotente.*

Proposição 4.1.7. *O elemento degenerado $\{1\}$ é o elemento aniquilador do operador $\sqcup$.*

Demonstração. De forma Similar à prova da Proposição 4.1.3. □

Lema 4.1.2. *Seja $X, Y, Z \in \mathbb{H}$. Se $X \subseteq Y$, então $(X \sqcup Z) \subseteq (Y \sqcup Z)$.*

Demonstração. Similar à prova do Lema 4.1.1. □

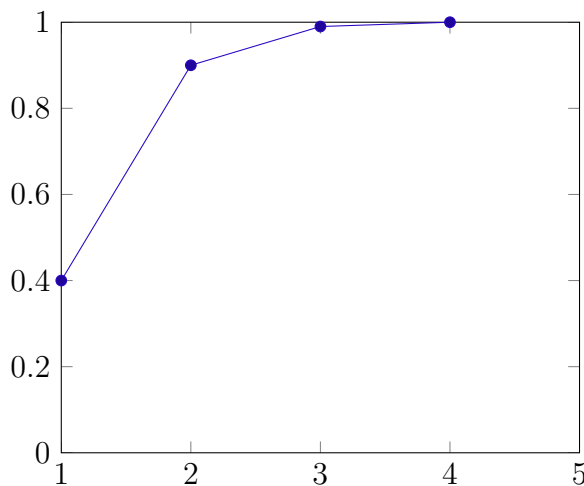
Imediatamente a este resultado tem-se sua generalização.

Proposição 4.1.8. *Sejam $X_1, X_2, Y_1, Y_2 \in \mathbb{H}$. Se $X_1 \subseteq Y_1$ e $X_2 \subseteq Y_2$, então $(X_1 \sqcup Y_1) \subseteq (X_2 \sqcup Y_2)$.*

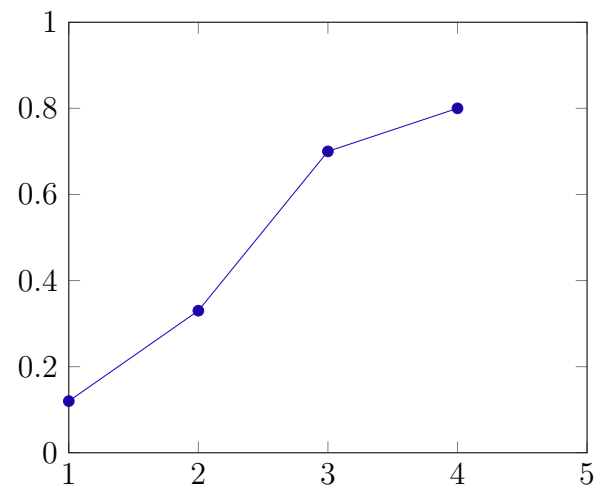
Demonstração. Direto do Lema 4.1.2. □

Corolário 4.1.4. *Se $X \subseteq Y$, então $X \subseteq (X \sqcup Y)$.*

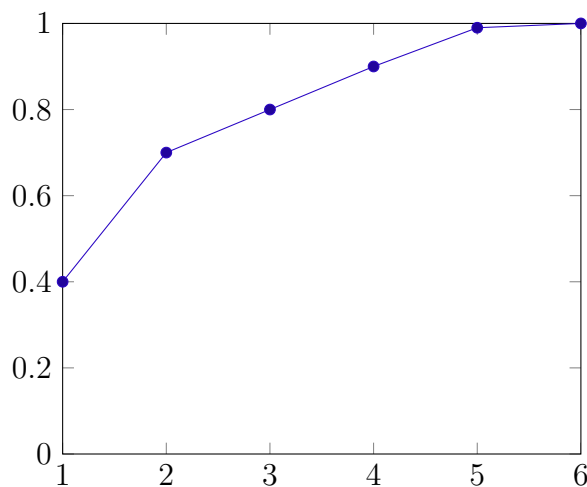
Demonstração. Semelhante a prova do Corolário 4.1.2. □



(a) X do Exemplo 4.1.3 ordenado por σ_X .



(b) Y do Exemplo 4.1.3 ordenado por σ_Y .



(c) $X \sqcup Y$ ordenado por $\sigma_{X \sqcup Y}$.

Figura 9 – Gráfico para a interpretação operacional da sup-combinação como um filtro passa alta.

Agora será mostrado as propriedades distributivas entre a sup-combinação e a inf-combinação.

Teorema 4.1.1. *O operador $\sqcup$ se distribui sobre $\otimes$.*

Demonstração. Para todo $X, Y, Z \in \mathbb{H}$ tem-se que,

$$\begin{aligned}
 X \sqcup (Y \otimes Z) &= X \sqcup \{y \wedge z \mid y \in Y, z \in Z\} \\
 &= \{x \vee (y \wedge z) \mid x \in X, y \in Y, z \in Z\} \\
 &\stackrel{(\mathcal{L}7)}{=} \{(x \vee y) \wedge (x \vee z) \mid x \in X, y \in Y, z \in Z\} \\
 &= \{(x \vee y) \mid x \in X, y \in Y\} \otimes \\
 &\quad \{(x \vee z) \mid x \in X, z \in Z\} \\
 &= (X \sqcup Y) \otimes (X \sqcup Z)
 \end{aligned}$$

e isto completa a prova. $\square$

Teorema 4.1.2. *O operador $\otimes$ se distribui sobre $\sqcup$.*

Demonstração. Similar à prova do Teorema 4.1.1. $\square$

Observação 4.1.1. *Para cada elemento degenerado $\{x\}, \{y\} \in \mathbb{H}$, é claro que $\{x\} \otimes \{y\} = \{x \wedge y\}$ e $\{x\} \sqcup \{y\} = \{x \vee y\}$.*

Proposição 4.1.9. *Seja $\cup$ a união padrão de conjuntos, para todo $X, Y, Z \in \mathbb{H}$ as seguintes igualdades são verdadeiras.*

$$i. (X \cup Y) \otimes Z = (X \otimes Z) \cup (Y \otimes Z).$$

$$ii. (X \cup Y) \sqcup Z = (X \sqcup Z) \cup (Y \sqcup Z).$$

Demonstração. (i) Por definição segue que:

$$\begin{aligned}
 (X \cup Y) \otimes Z &= \{\alpha \wedge z \mid \alpha \in (X \cup Y), z \in Z\} \\
 &= \{\alpha \wedge z \mid \alpha \in X, z \in Z\} \cup \{\alpha \wedge z \mid \alpha \in Y, z \in Z\} \\
 &= (X \otimes Z) \cup (Y \otimes Z)
 \end{aligned}$$

a prova para (ii) é similar. $\square$

O resultado da Proposição 4.1.9 é generalizado pelo teorema a seguir.

Teorema 4.1.3. *Se $\{X_i\}_{i \in I}$ e $\{Y_j\}_{j \in J}$ são duas famílias finitas de elementos de $\mathbb{H}$ e $\circ \in \{\otimes, \sqcup\}$, então $\bigcup_{i \in I} X_i \circ \bigcup_{j \in J} Y_j = \bigcup_{i \in I} \bigcup_{j \in J} (X_i \circ Y_j)$.*

Demonstração. Direto pela Proposição 4.1.9 e do fato de $\sqcup, \otimes$ e $\cup$ serem associativas e comutativas. $\square$

Além disso, pela sup-combinação ser associativa, é possível estender tal operador para aridade n , dado $X_1, X_2, \dots, X_n \in \mathbb{H}$ pode-se calcular $\sqcup$ como se segue.

$$\bigsqcup_{i=1}^n X_i = \left(\bigsqcup_{i=1}^{n-1} X_i \right) \sqcup X_n = X_1 \sqcup \dots \sqcup X_n \quad (4.5)$$

Observação 4.1.2. Em particular se $n = 1$ aqui será considerado que, $\sqcup X_1 = X_1$.

Proposição 4.1.10. Se $X_1, \dots, X_n, Y \in \mathbb{H}$, então $\left(\bigsqcup_{i=1}^n X_i \right) \otimes Y = \bigsqcup_{i=1}^n (X_i \otimes Y)$.

Demonstração. Note que,

$$\begin{aligned} \left(\bigsqcup_{i=1}^n X_i \right) \otimes Y &\stackrel{(Eq. 4.5)}{=} \left(X_1 \sqcup \dots \sqcup X_n \right) \otimes Y \\ &\stackrel{Teo. 4.1.2}{=} (X_1 \otimes Y) \sqcup \dots \sqcup (X_n \otimes Y) \\ &\stackrel{(Eq. 4.5)}{=} \bigsqcup_{i=1}^n (X_i \otimes Y) \end{aligned}$$

o que conclui a demonstração. $\square$

Observação 4.1.3. Leitores familiarizados com álgebra abstrata podem notar que os resultados das Proposições 4.1.1 – 4.1.3, 4.1.5 – 4.1.7 e dos Teoremas 4.1.1 e 4.1.2 permitem dizer que o conjunto $\mathbb{H}$ munido dos operadores $\otimes$ e $\sqcup$ é um semi-anel² comutativo. Este semi-anel será usado mais adiante neste trabalho como espaço de valoração para a construção dos diferentes tipos de autômatos fuzzy hesitantes típicos.

Uma parte importante no estudo dos EFHT's, é encontrar relações para ordenar tais elementos, isto é, encontrar relações de ordem sobre o conjunto $\mathbb{H}$. Visto que, os EFHT's são subconjunto finitos e não vazios do intervalo $[0, 1]$, uma ordem parcial óbvia sobre $\mathbb{H}$ como destacado em (MENDIVIL; GARITAGOITIA, 2015; SANTOS et al., 2015), é a relação de inclusão, isto é, definir a ordem parcial $\preceq_{\mathbb{H}}$ sobre $\mathbb{H}$ como:

$$X \preceq_{\mathbb{H}} Y \iff X \subseteq Y \quad (4.6)$$

para todo $X, Y \in \mathbb{H}$.

Porém, como dito em (SANTOS et al., 2015), a relação $\preceq_{\mathbb{H}}$ sobre o conjunto $\mathbb{H}$ apesar de ser uma ordem parcial, não tem a característica de generalizar a ordem usual $\leq$

² Um semi-anel é uma álgebra $\langle C, +, \star, 0, 1 \rangle$ onde: $\langle C, +, 0 \rangle$ é um monoide comutativo, $\langle C, \star, 1 \rangle$ é um monoide, $\star$ se distribui sobre $+$ e 0 é o elemento aniquilador de $\star$. E no caso de $\star$ ser comutativo o semi-anel é dito ser comutativo

dos números reais, note por exemplo, que os EFHT's $\{0\}$ e $\{1\}$ são incomparáveis pela ordem $\preceq_{\mathbb{H}}$, entretanto, suas imagens diretas³ 0 e 1 são comparáveis pela ordem usual $\leq$ no conjunto $\mathbb{R}$. Agora será introduzida, usando o operador de sup-combinação, uma nova relação de ordem parcial sobre $\mathbb{H}$ que generaliza a ordem usual $\leq$, esta será nomeada como ordem CBS.

Definição 4.1.4. *Seja $X, Y \in \mathbb{H}$ dizemos que X é inferior ou igual a Y pela relação CBS, denotado por $X \sqsubseteq Y$, sempre que $X \sqcup Y = Y$.*

Proposição 4.1.11. *A relação $\sqsubseteq$ é uma relação de ordem parcial sobre $\mathbb{H}$.*

Demonstração. Note que:

(i) Para qualquer $X \in \mathbb{H}$ tem-se que $X \sqcup X \stackrel{Prop. 4.1.5}{=} X$, logo $\sqsubseteq$ é reflexiva.

(ii) Suponha que $X \sqsubseteq Y$ e $Y \sqsubseteq X$, assim tem-se que:

$$X \stackrel{Hip.}{=} Y \sqcup X \stackrel{Prop. 4.1.5}{=} X \sqcup Y \stackrel{Hip.}{=} Y$$

portanto, $\sqsubseteq$ é anti-simétrica.

(iii) Suponha que $X \sqsubseteq Y$ e $Y \sqsubseteq Z$ logo:

$$X \sqcup Z \stackrel{Hip.}{=} X \sqcup (Y \sqcup Z) \stackrel{Prop. 4.1.5}{=} (X \sqcup Y) \sqcup Z \stackrel{Hip.}{=} Y \sqcup Z \stackrel{Hip.}{=} Z$$

consequentemente, $X \sqsubseteq Z$, e portanto, $\sqsubseteq$ é transitiva.

Uma vez que $\sqsubseteq$ é reflexiva, anti-simétrica e transitiva tem-se a relação CBS é uma ordem parcial sobre $\mathbb{H}$. $\square$

Observação 4.1.4. *O fato de $\sqsubseteq$ ser uma ordem parcial baseada na sup-combinação, implica que se $X \sqsubseteq Y$, então os componentes de Y (os valores $y \in Y$) são mais significativos que os componentes de X (os valores $x \in X$), do ponto de vista operacional (no sentido de interpretar os EFHT's como sinais), assim pode-se considerar que a relação $X \sqsubseteq Y$ diz que, o sinal representado por Y é mais significativo que o sinal de X .*

Proposição 4.1.12. *A ordem parcial $\sqsubseteq$ generaliza a ordem usual $\leq$.*

Demonstração. Basta notar que para qualquer $\{x\}, \{y\} \in \mathbb{H}_1$ tem-se que,

$$\{x\} \sqsubseteq \{y\} \iff \{x\} \sqcup \{y\} = \{y\} \iff x \vee y = y \iff x \leq y$$

o que conclui a prova. $\square$

³ Imagem direta aqui diz respeito a imagem da função $Ima : \mathbb{H}_1 \rightarrow [0, 1]$, definida como $Ima(\{x\}) = x$.

A Proposição 4.1.12 mostra que $\leq$ é um caso particular de $\sqsubseteq$, quando está última está restrita aos elementos degenerados.

Proposição 4.1.13. *Se $X \sqsubseteq Y$, então para todo $Z \in \mathbb{H}$:*

$$(R1) \quad (X \sqcup Z) \sqsubseteq (Y \sqcup Z).$$

$$(R2) \quad (X \otimes Z) \sqsubseteq (Y \otimes Z).$$

Demonstração. Assuma que $X \sqsubseteq Y$, assim pela definição de $\sqsubseteq$ tem-se que, $X \sqcup Y = Y$, logo para qualquer $Z \in \mathbb{H}$ tem-se:

$$(X \sqcup Z) \sqcup (Y \sqcup Z) \stackrel{Prop. 4.1.5}{=} (X \sqcup Y) \sqcup Z \stackrel{Hip.}{=} Y \sqcup Z$$

portanto, $(X \sqcup Z) \sqsubseteq (Y \sqcup Z)$, logo (R1) acontece. Além disso,

$$(X \otimes Z) \sqcup (Y \otimes Z) \stackrel{Teo. 4.1.1}{=} (X \sqcup Y) \otimes Z \stackrel{Hip.}{=} Y \sqcup Z$$

portanto, $(X \otimes Z) \sqsubseteq (Y \otimes Z)$, logo (R2) acontece e isso conclui a prova. $\square$

O resultado anterior mostra que tanto a inf-combinação quanto a sup-combinação são monotônicas na primeira componente com respeito a ordem $\sqsubseteq$, além disso, uma vez que ambas as combinações são comutativas o resultado também vale para a segunda componente.

Proposição 4.1.14. *Para $\circ \in \{\otimes, \sqcup\}$ se $X_1 \sqsubseteq Y_1$ e $X_2 \sqsubseteq Y_2$, então $(X_1 \circ X_2) \sqsubseteq (Y_1 \circ Y_2)$.*

Demonstração. Direto pelo fato de $\otimes$ e $\sqcup$ serem monotônicas em ambas as componentes. $\square$

Proposição 4.1.15. *Para $\circ \in \{\otimes, \sqcup\}$ se $X \sqsubseteq Y$, então $X \sqsubseteq (X \circ Y)$.*

Demonstração. Suponha que $\circ \in \{\otimes, \sqcup\}$ e que $X \sqsubseteq Y$, agora pela Proposição 4.1.14 tem-se que $(X \circ X) \sqsubseteq (X \circ Y)$, mas $(X \circ X) = X$, logo $X \sqsubseteq (X \circ Y)$. $\square$

Observação 4.1.5. *Note que para qualquer que seja $X, Y \in \mathbb{H}$ tem-se que, $X \sqcup (X \sqcup Y) = X \sqcup Y$, portanto, $X \sqsubseteq (X \sqcup Y)$.*

Teorema 4.1.4. *Seja $\{X_i\}_{i \in I}$ uma família finita e não vazia de elementos de $\mathbb{H}$, então para qualquer que seja X_j com $j \in I$ segue que $X_j \sqsubseteq \bigsqcup_{i \in I} X_i$.*

Demonstração. Direto pela Observação 4.1.5 e pelo fato da função $\sqcup$ ser comutativa e associativa (Proposição 4.1.5). $\square$

4.2 Autômatos Fuzzy Hesitantes Típicos

Como explicado na Seção 3.2, os autômatos *fuzzy* podem ser vistos como sendo algoritmos *fuzzy* (ZADEH, 1996), em que, as transições entre estados e a noção de estado final correspondem às instruções (ou comandos) na forma de predicados valorados no intervalo $[0,1]$. A grande dificuldade de se trabalhar com os autômatos *fuzzy* como dito em (COSTA; BEDREGAL, 2020), é a necessidade de se estabelecer *a priori* um único valor exato de pertinência para cada uma das transições entre estados, além disso, é necessário definir, também *a priori*, o valor de pertinência exato de cada estado ser um estado final.

Qiu em seu trabalho (QIU, 2001), generalizou o conceito de autômato *fuzzy*, isso foi feito considerando o espaço de valoração como sendo uma estrutura de reticulado residuado completo, entretanto, como explicado em (COSTA; BEDREGAL; SANTIAGO, 2020), alguns modelos de computação podem utilizar de uma álgebra mais simplista (em termos de estrutura) para ser o espaço de valoração. Nesta e nas próximas seções serão introduzidas novas generalizações dos autômatos *fuzzy* que usam o semi-anel $\mathbb{H}$ como espaço de valoração.

Definição 4.2.1. *Um Autômato Fuzzy Hesitante Típico, ou simplesmente AFHT, é uma tupla na forma $\mathcal{M} = \langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$ onde:*

- Q, Σ e q_0 são da mesma forma que na Definição 2.1.3;
- $\psi : Q \times \Sigma \times Q \rightarrow \mathbb{H}$ é um CFHT, chamado programa hesitante e
- $\mathcal{F} : Q \rightarrow \mathbb{H}$ é um CFHT de estados finais (ou de aceitação).

Observação 4.2.1. *Notavelmente os autômatos fuzzy são casos particulares dos AFHT's, em que, a imagem dos CFHT's ψ e $\mathcal{F}$ são restritas ao conjunto de EFHT's degenerados, isto é, possuem as seguintes assinaturas $\psi : Q \times \Sigma \times Q \rightarrow \mathbb{H}_1$ e $\mathcal{F} : Q \rightarrow \mathbb{H}_1$, neste caso, sempre que o contexto for claro, serão removidos os símbolos $\{\}$ do elemento, assim será escrito por exemplo, apenas $\mathcal{F}(q) = x$, em vez de, $\mathcal{F}(q) = \{x\}$, e o mesmo vale para o CFHT ψ .*

O uso de $\mathbb{H}$ na valoração dos AFHT's, além de, modelar a incerteza ligada às instruções de transição e finalização, também consegue modelar a dificuldade de se atribuir o valor exato de pertinência de tais instruções, pois usando as ideias de (RODRÍGUEZ et al., 2014), tem-se por exemplo que, $\psi(q, a, q') = \{x_1, \dots, x_n\}$ significa que, existe uma dúvida entre n valores, sobre o qual melhor modela a incerteza da instrução (q, a, q') ser executada, essa dúvida recebe o nome de hesitação, como dito em (RODRÍGUEZ et al., 2014). Obviamente pela observação 4.2.1 fica claro que autômatos *fuzzy* não conseguem modelar hesitação, pois toda valoração é um conjunto unitário.

Definição 4.2.2. Seja $\mathcal{M}$ um AFHT, o CFHT $\hat{\psi} : Q \times \Sigma^* \times Q \rightarrow \mathbb{H}$ é definido para todo $q, q' \in Q, a \in \Sigma$ e $w \in \Sigma^*$ como:

$$\hat{\psi}(q, \lambda, q') = \begin{cases} \{0\}, & \text{se } q \neq q' \\ \{1\}, & \text{senão} \end{cases} \quad (4.7)$$

$$\hat{\psi}(q, wa, q') = \bigsqcup_{q'' \in Q} \left(\hat{\psi}(q, w, q'') \otimes \psi(q'', a, q') \right) \quad (4.8)$$

A representação visual dos AFHT's é similar à representação dos autômatos *fuzzy* simplificados (ver Seção 3.2), isto é, são usados grafos para representar visualmente os AFHT's, onde os vértices são rotulados por $(q; X)$, com $q \in Q$ e $X \in \mathbb{H}$ tal que $\mathcal{F}(q) = X$, e além disso, para quaisquer dois vértices (q_i, X_i) e (q_j, X_j) tem-se uma aresta ligando os vértices que é rotulada por (a, Y) , que representa a equação $\psi(q_i, a, q_j) = Y$. Um exemplo da representação de um AFHT é vista na Figura 10, ressaltando que são omitidas as arestas para as equações da forma $\psi(q_i, a, q_j) = \{0\}$ para algum $a \in \Sigma$. Além disso, existe uma seta indicando o estado inicial.

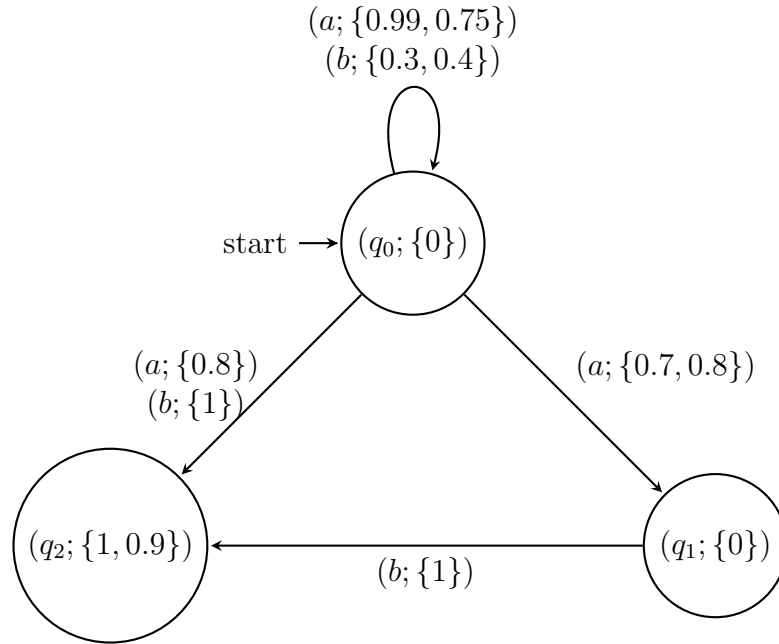


Figura 10 – Representação de um AFHT.

A valoração de uma computação é um AFHT, isto é, o EFHT que representa o conjunto de pertinências com que a computação de uma palavra de entrada $w \in \Sigma^*$ acontece em um AFHT $\mathcal{M}$, é obtido pela aplicação do CFHT $\hat{\psi}$ sobre a tripla (q_0, w, q) , onde $q \in Q$.

Exemplo 4.2.1. Considere o AFHT da Figura 10, agora dado a palavra $w = aa$, tem-se

que a computação iniciada em q_0 e finalizada em q_2 tem a seguinte valoração,

$$\begin{aligned}\widehat{\psi}(q_0, aa, q_2) &= \bigsqcup_{q \in \{q_0, q_1, q_2\}} \left(\widehat{\psi}(q_0, a, q) \otimes \psi(q, a, q_2) \right) \\ &= \left(\widehat{\psi}(q_0, a, q_0) \otimes \psi(q_0, a, q_2) \right) \sqcup \\ &\quad \left(\widehat{\psi}(q_0, a, q_1) \otimes \psi(q_1, a, q_2) \right) \sqcup \\ &\quad \left(\widehat{\psi}(q_0, a, q_2) \otimes \psi(q_2, a, q_2) \right)\end{aligned}$$

desenvolvendo $\psi(q_0, a, q_2)$, $\psi(q_1, a, q_2)$ e $\psi(q_2, a, q_2)$, tem-se respectivamente que,

$$\begin{aligned}\psi(q_0, a, q_2) &= \{0.8\} \\ \psi(q_1, a, q_2) &= \{0\} \\ \psi(q_2, a, q_2) &= \{0\}\end{aligned}$$

uma vez que, $\{0\}$ é o elemento neutro para $\sqcup$ e é o elemento de absorção para $\otimes$, tem-se que, $\widehat{\psi}(q_0, aa, q_2)$ se resume a,

$$\widehat{\psi}(q_0, aa, q_2) = \widehat{\psi}(q_0, a, q_0) \otimes \psi(q_0, a, q_2)$$

desenvolvendo $\widehat{\psi}(q_0, a, q_0)$ tem-se que,

$$\widehat{\psi}(q_0, a, q_0) = \{0.99, 0.75\}$$

e portanto, o conjunto de pertinência que corresponde a valoração da computação de aa iniciar em q_0 e finalizar em q_2 é igual a,

$$\begin{aligned}\widehat{\psi}(q_0, aa, q_2) &= \{0.99, 0.75\} \otimes \{0.8\} \\ &= \{0.8, 0.75\}\end{aligned}$$

Note que, como para o caso dos AFD's e AFN's a existência de um estado inicial q_0 garante que sempre existe um momento inicial de computação, dessa forma, os AFHT's são máquinas que não modelam os erros de inicialização, por outro lado, são máquinas capazes de modelar erros de finalização e de transição (para detalhes consulte a Seção 3.2).

Observação 4.2.2. Não é difícil notar que os AFHT's generalizam diretamente os autômatos fuzzy simplificados, entretanto, uma vez que, os autômatos fuzzy simplificados são equivalentes ao modelo mais genérico de autômatos fuzzy (ver Seção 3.2), não há problema em enfatizar que os AFHT's generalizam os autômatos fuzzy em sua forma mais geral.

Proposição 4.2.1. Se $\mathcal{M}$ é um AFHT, então para todo $x, y \in \Sigma^*$ tem-se que:

$$\widehat{\psi}(q, xy, q') = \bigsqcup_{q'' \in Q} \left(\widehat{\psi}(q, x, q'') \otimes \widehat{\psi}(q'', y, q') \right) \quad (4.9)$$

Demonstração. Seja $\mathcal{M}$ um AFHT e seja $q, q' \in Q$ e $x, y \in \Sigma^*$. No caso de $|y| = 0$ ou $|y| = 1$, a prova é direta pela definição de $\hat{\psi}$. Para o caso geral de $|y| > 1$ a prova é feita por indução sobre o tamanho da palavra de entrada, assim assuma que o resultado é verdadeiro para todo $u \in \Sigma^*$ onde $|u| = n - 1$ com $n > 0$, agora considere que $y = ua$, assim tem-se que,

$$\begin{aligned}
\hat{\psi}(q, xy, q') &= \hat{\psi}(q, xua, q') \\
&= \bigsqcup_{q^* \in Q} \left(\hat{\psi}(q, xu, q^*) \otimes \psi(q^*, a, q') \right) \\
&\stackrel{Hip.}{=} \bigsqcup_{q^* \in Q} \left(\left(\bigsqcup_{q'' \in Q} \left(\hat{\psi}(q, x, q'') \otimes \hat{\psi}(q'', u, q^*) \right) \right) \otimes \psi(q^*, a, q') \right) \\
&\stackrel{Prop. 4.1.10}{=} \bigsqcup_{q^* \in Q} \left(\bigsqcup_{q'' \in Q} \left(\left(\hat{\psi}(q, x, q'') \otimes \hat{\psi}(q'', u, q^*) \right) \otimes \psi(q^*, a, q') \right) \right) \\
&\stackrel{Prop. 4.1.1}{=} \bigsqcup_{q^* \in Q} \left(\bigsqcup_{q'' \in Q} \left(\hat{\psi}(q, x, q'') \otimes \left(\hat{\psi}(q'', u, q^*) \otimes \psi(q^*, a, q') \right) \right) \right) \\
&\stackrel{Prop. 4.1.10}{=} \bigsqcup_{q'' \in Q} \left(\hat{\psi}(q, x, q'') \otimes \bigsqcup_{q^* \in Q} \left(\hat{\psi}(q'', u, q^*) \otimes \psi(q^*, a, q') \right) \right) \\
&= \bigsqcup_{q'' \in Q} \left(\hat{\psi}(q, x, q'') \otimes \hat{\psi}(q'', ua, q') \right) \\
&= \bigsqcup_{q'' \in Q} \left(\hat{\psi}(q, x, q'') \otimes \hat{\psi}(q'', y, q') \right)
\end{aligned}$$

e isto completa a prova. $\square$

Observação 4.2.3. Agora note que pela Definição de $\hat{\psi}$ pode-se concluir que para todo $w \in \Sigma^+$ com $w = a_1 \cdots a_{n-1}a_n$, tem-se que,

$$\hat{\psi}(q_0, a_1 \cdots a_{n-1}a_n, q_n) = \bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\psi(q_0, a_1, q_1) \otimes \cdots \otimes \psi(q_{n-1}, a_n, q_n) \right)$$

onde $a_i \in \Sigma$ e $1 \leq i \leq n$.

Definição 4.2.3. Para qualquer $w \in \Sigma^+$ a sequência de estados $q_1, \dots, q_{n-1}$ vista na Observação 4.2.3 é chamada de caminho ou rota de computação.

Uma vez que, os AFHT's possuem um estado inicial, pode-se então definir a linguagem de um AFHT como sendo o CFHT sobre todas as possíveis computações iniciadas no estado inicial e finalizadas em qualquer outro estado, formalmente isto é definido como se segue.

Definição 4.2.4. Seja $\mathcal{M}$ um AFHT, a linguagem aceita (ou computada) por $\mathcal{M}$ é um CFHT $\hat{f}_{\mathcal{M}} : \Sigma^* \rightarrow \mathbb{H}$ definido para todo $w \in \Sigma^*$ como sendo:

$$\hat{f}_{\mathcal{M}}(w) = \bigsqcup_{q \in Q} \left(\hat{\psi}(q_0, w, q) \otimes \mathcal{F}(q) \right) \quad (4.10)$$

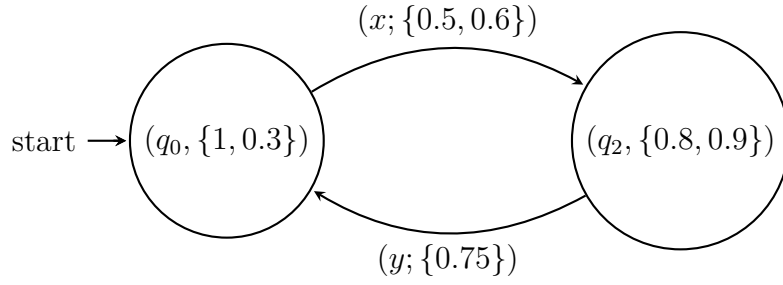


Figura 11 – Um AFHT sobre o alfabeto $\{x, y\}$.

Observação 4.2.4. Pela Definição 4.2.4 para qualquer $\mathcal{M}$ a seguinte igualdade é válida $\hat{f}_{\mathcal{M}}(\lambda) = \mathcal{F}(q_0)$.

Exemplo 4.2.2. Seja $\mathcal{M}$ o AFHT apresentado pela Figura 11, usando indução sobre w é fácil verificar que a linguagem⁴ de $\mathcal{M}$ corresponde ao CFHT definido por,

$$\hat{f}_{\mathcal{M}}(w) = \begin{cases} \{1, 0.3\}, & \text{se } w = \lambda \\ \{0.5, 0.6\}, & \text{se } w = (xy)^n x \text{ com } n \geq 0 \\ \{0.3, 0.5, 0.6\}, & \text{se } w = (xy)^n \text{ com } n \geq 1 \\ \{0\}, & \text{qualquer outro caso.} \end{cases}$$

Agora dado dois AFHT's $\mathcal{M}$ e $\mathcal{M}'$ é possível definir um relacionamento hierárquico entre eles baseado em suas linguagens como se segue.

Definição 4.2.5. Seja $\mathcal{M}$ e $\mathcal{M}'$ dois AFHT's sobre o mesmo alfabeto Σ , é dito que $\mathcal{M}$ é menos ou tão hesitante quanto $\mathcal{M}'$, denotado por $\mathcal{M} \leq \mathcal{M}'$, sempre que $\hat{f}_{\mathcal{M}}(w) \subseteq \hat{f}_{\mathcal{M}'}(w)$ para todo $w \in \Sigma^*$.

Observação 4.2.5. Não é difícil verificar que a relação $\leq$ é uma ordem parcial sobre o conjunto de todos os AFHT's.

Como o programa hesitante ψ é um CFHT, ele é uma função total, e portanto, para qualquer par $(q, a) \in Q \times \Sigma$ tem-se que o número de computações possíveis na forma (q, a, q') será igual a $\#Q$, neste sentido, os AFHT's são não-determinísticos. Agora usando a noção de sucessor é possível apresentar a ideia de determinismo para os AFHT's.

Definição 4.2.6. Seja $\mathcal{M}$ um AFHT, para todo $q \in Q$ e $a \in \Sigma$ o conjunto dos sucessores de q com respeito ao símbolo a é o conjunto dado por,

$$SUC_{\mathcal{M}}(q, a) = \{q' \in Q \mid \psi(q, a, q') \neq \{0\}\} \quad (4.11)$$

Definição 4.2.7. Um Autômato Fuzzy Hesitante Típico Determinístico (ou simplesmente AFHTD) $\mathcal{M}$ é um AFHT tal que para todo $q \in Q$ e $a \in \Sigma$ tem-se que $\#SUC_{\mathcal{M}}(q, a) \leq 1$.

⁴ A notação $(xy)^n$ diz respeito a concatenação da palavra xy com ela mesmo n vezes, no caso particular de $n = 0$ tem-se que $(xy)^0 = \lambda$.

O teorema que segue mostra que para qualquer AFHT sempre é possível obter um AFHTD que computa sobre o mesmo alfabeto, porém, com um nível de hesitação maior.

Teorema 4.2.1. *Para todo AFHT $\mathcal{M}$ existe um AFHTD $\mathcal{M}'$ tal que $\mathcal{M} \trianglelefteq \mathcal{M}'$.*

Demonstração. Dado um AFHT $\mathcal{M} = \langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$ é possível construir um novo AFHT $\mathcal{M}' = \langle \wp^+(Q), \Sigma, \psi', \{q_0\}, \mathcal{F}' \rangle$ onde, $\wp^+(Q) = \wp(Q) - \{\emptyset\}$ e para todo $X, Y \in \wp^+(Q)$ e $a \in \Sigma$ tem-se que:

$$\psi'(X, a, Y) = \begin{cases} \bigcup_{q \in X, q' \in Y} \psi(q, a, q'), & \text{se } Y = \bigcup_{q \in X} SUC_{\mathcal{M}}(q, a) \\ \{0\}, & \text{qualquer outro caso} \end{cases}$$

e

$$\mathcal{F}'(X) = \bigcup_{q \in X} \mathcal{F}(q).$$

Agora será mostrado que $\mathcal{M}'$ é um AFHTD, note que para qualquer $a \in \Sigma, X \in \wp^+(Q)$ se,

$$\bigcup_{q \in X} SUC_{\mathcal{M}}(q, a) = Y$$

então existe $q' \in Y$, tal que para algum $q \in X$ é claro que, $\psi(q, a, q') \neq \{0\}$ com $q' \in Q$, logo,

$$\bigcup_{q \in X, q' \in Y} \psi(q, a, q') \neq \{0\}$$

assim, $\psi'(X, a, Y) \neq \{0\}$, consequentemente, $Y \in \#SUC_{\mathcal{M}'}(X, a)$. Por outro lado, se

$$Y \neq \bigcup_{q \in X} SUC_{\mathcal{M}}(q, a)$$

pela definição de ψ' teremos que, $\psi'(X, a, Y) = \{0\}$, assim $Y \notin \#SUC_{\mathcal{M}'}(X, a)$. Portanto, para todo $a \in \Sigma, X \in \wp^+(Q)$ tem-se que $\#SUC_{\mathcal{M}'}(X, a) \leq 1$, logo $\mathcal{M}'$ é AFHTD. Agora será mostrado que para todo $w \in \Sigma^*$ é válido que $\hat{f}_{\mathcal{M}}(w) \subseteq \hat{f}_{\mathcal{M}'}(w)$, inicialmente note que no caso de $w = \lambda$ pela Observação 4.2.4 e a definição de $\mathcal{F}'$ segue que,

$$\hat{f}_{\mathcal{M}}(\lambda) = \mathcal{F}(q_0) = \mathcal{F}'(\{q_0\}) = \hat{f}_{\mathcal{M}'}(\lambda) \quad (4.12)$$

e para qualquer $w \in \Sigma^+$ com $w = a_1 \cdots a_n$ por definição tem-se que,

$$\hat{f}_{\mathcal{M}}(a_1 \cdots a_n) = \bigsqcup_{q_n \in Q} \left(\hat{\psi}(q_0, a_1 \cdots a_n, q_n) \otimes \mathcal{F}(q_n) \right)$$

entretanto, pela definição de ψ' é fácil ver que para todo $\psi(q_i, a, q_j)$, segue que,

$$\psi(q_i, a, q_j) \subseteq \psi'(X, a, Y)$$

para cada $X, Y \in \wp^+(Q)$ tal que $q_i \in X, q_j \in Y$. Logo pela definição de $\widehat{\psi}$ e pela Proposição 4.1.4 é claro que,

$$\widehat{\psi}(q_0, a_1 \cdots a_n, q_n) \subseteq \widehat{\psi}'(\{q_0\}, a_1 \cdots a_n, Z)$$

para algum $q_n \in Q, Z \in \wp^+(Q)$ tal que $q_n \in Z$ (onde $\#Z \geq 1$), consequentemente, $\mathcal{F}(q_n) \subseteq \mathcal{F}'(Z)$. Desde que,

$$\widehat{f}_{\mathcal{M}'}(a_1 \cdots a_n) = \bigsqcup_{Z \in \wp^+(Q)} \left(\widehat{\psi}'(\{q_0\}, a_1 \cdots a_n, Z) \otimes \mathcal{F}'(Z) \right)$$

pelas Proposições 4.1.4 e 4.1.8 pode-se concluir que,

$$\widehat{f}_{\mathcal{M}}(a_1 \cdots a_n) \subseteq \widehat{f}_{\mathcal{M}'}(a_1 \cdots a_n) \quad (4.13)$$

Assim, pelas Equações (4.12) e (4.13) tem-se $\widehat{f}_{\mathcal{M}}(w) \subseteq \widehat{f}_{\mathcal{M}'}(w)$ para todo $w \in \Sigma^*$. E portanto, $\mathcal{M} \sqsubseteq \mathcal{M}'$. $\square$

Esse aumento de hesitação apresentado no Teorema 4.2.1, se dá pela fato de como os CFHT's ψ' e $\mathcal{F}'$ são construídos, observe que, a operação que é responsável pela construção de tais CFHT's é a união, e uma vez que, a união não é uma das operações do semi-anel $\mathbb{H}$, não existe “controle computacional⁵” para que o AFHTD conserve a linguagem, este método de remoção pode ser considerado como sendo um método ingênuo.

Definição 4.2.8. *Seja $\mathcal{M}$ um AFHT, um estado $q' \in Q - \{q_0\}$ tal que para todo $q \in Q$ e $a \in \Sigma$ satisfaz a condição $\psi(q, a, q') = \{0\}$ é chamado de estado nulo. O conjunto de todos os estados nulos é denotado por π , ou seja, $\pi = \{q' \in Q \mid \forall a \in \Sigma, \forall q \in Q \psi(q, a, q') = \{0\}\}$.*

A existência de estados nulos em um AFHT não aumentam o poder computacional deste AFHT como pode ser visto pelo teorema a seguir.

Teorema 4.2.2. *Para qualquer AFHT $\mathcal{M}$, existe um AFHT $\mathcal{M}'$ sem estados nulos tal que $\widehat{f}_{\mathcal{M}} = \widehat{f}_{\mathcal{M}'}$.*

Demonstração. Seja $\mathcal{M} = \langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$ um AFHT, agora defina um novo AFHT $\mathcal{M}' = \langle Q', \Sigma, \psi', q_0, \mathcal{F}' \rangle$ tal que $Q' = Q - \pi$ onde $\psi' = \psi|_{Q'}$ e $\mathcal{F}' = \mathcal{F}|_{Q'}$. Logo é claro que, o AFHT $\mathcal{M}'$ não possui estados nulos. Agora será mostrado que $\widehat{f}_{\mathcal{M}} = \widehat{f}_{\mathcal{M}'}$. Se $w = \lambda$ então pela Observação 4.2.4 e a definição de $\mathcal{F}'$ segue que,

$$\widehat{f}_{\mathcal{M}}(\lambda) = \mathcal{F}(q_0) = \mathcal{F}'(q_0) = \widehat{f}_{\mathcal{M}'}(\lambda). \quad (4.14)$$

⁵ Controle computacional aqui diz respeito a ação de que os caminhos de computação não-determinísticos para um mesmo w são mapeados em um único caminho determinístico, a valoração deste caminho determinístico, por sua vez, é interpretada semanticamente como sendo a disjunção das valorações dos caminhos não-determinísticos

Por outro lado, se $w = a_1 \cdots a_n$ com $a_i \in \Sigma$ onde $1 \leq i \leq n$, então existem dois casos possíveis. (1) Considere que $\hat{f}_{\mathcal{M}}(w) \neq \{0\}$, assim,

$$\begin{aligned} \hat{f}_{\mathcal{M}}(w) \neq \{0\} &\iff \bigsqcup_{q_n \in Q} \left(\hat{\psi}(q_0, w, q_n) \otimes \mathcal{F}(q_n) \right) \neq \{0\} \\ &\stackrel{\text{Prop. 4.1.6}}{\iff} (\exists q_n) (\hat{\psi}(q_0, w, q_n) \otimes \mathcal{F}(q_n)) \neq \{0\} \\ &\stackrel{\text{Prop. 4.1.3}}{\iff} (\exists q_n) \hat{\psi}(q_0, w, q_n) \neq \{0\} \text{ e } \mathcal{F}(q_n) \neq \{0\} \end{aligned}$$

agora pela Observação 4.2.3 segue que,

$$\begin{aligned} \hat{\psi}(q_0, w, q_n) \neq \{0\} &\iff \bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\psi(q_0, a_1, q_1) \otimes \cdots \otimes \psi(q_{n-1}, a_n, q_n) \right) \neq \{0\} \\ &\stackrel{\text{Prop. 4.1.6}}{\iff} (\exists q_1, \dots, q_{n-1}) (\psi(q_0, a_1, q_1) \otimes \cdots \otimes \psi(q_{n-1}, a_n, q_n)) \neq \{0\} \\ &\stackrel{\text{Prop. 4.1.3}}{\iff} (\exists q_1, \dots, q_{n-1}) \psi(q_0, a_1, q_1) \neq \{0\}, \dots, \psi(q_{n-1}, a_n, q_n) \neq \{0\} \end{aligned}$$

consequentemente, $q_1, \dots, q_{n-1}, q_n \in Q'$ e para todo $1 \leq i \leq n$ tem-se,

$$\psi'(q_{i-1}, a_i, q_i) = \psi(q_{i-1}, a_i, q_i)$$

além disso, $\mathcal{F}'(q_n) = \mathcal{F}(q_n)$ e também é claro que,

$$\hat{\psi}'(q_0, a_1 \cdots a_n, q_n) = \hat{\psi}(q_0, a_1 \cdots a_n, q_n)$$

portanto,

$$\hat{f}_{\mathcal{M}}(w) = \hat{f}_{\mathcal{M}'}(w) \tag{4.15}$$

(2) Considere agora que $\hat{f}_{\mathcal{M}}(w) = \{0\}$ assim,

$$\begin{aligned} \hat{f}_{\mathcal{M}}(w) = \{0\} &\iff \bigsqcup_{q_n \in Q} \left(\hat{\psi}(q_0, w, q_n) \otimes \mathcal{F}(q_n) \right) = \{0\} \\ &\stackrel{\text{Prop. 4.1.6}}{\iff} (\forall q_n) (\hat{\psi}(q_0, w, q_n) \otimes \mathcal{F}(q_n)) = \{0\} \\ &\stackrel{\text{Prop. 4.1.3}}{\iff} (\forall q_n) \hat{\psi}(q_0, w, q_n) = \{0\} \text{ ou } \mathcal{F}(q_n) = \{0\} \end{aligned}$$

assim pela Observação 4.2.3 segue que,

$$\begin{aligned} \hat{\psi}(q_0, w, q_n) = \{0\} &\iff \bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\psi(q_0, a_1, q_1) \otimes \cdots \otimes \psi(q_{n-1}, a_n, q_n) \right) = \{0\} \\ &\stackrel{\text{Prop. 4.1.6}}{\iff} (\forall q_1, \dots, q_{n-1}) (\psi(q_0, a_1, q_1) \otimes \cdots \otimes \psi(q_{n-1}, a_n, q_n)) = \{0\} \end{aligned}$$

mas pela definição de Q' para todo q_i no caminho de computação $q_1, \dots, q_{n-1}$ se $q_i \notin \pi$ então $q_i \in Q'$, logo em $\mathcal{M}'$ existe o um caminho de computação em $\mathcal{M}'$ com os mesmo estados $q_1, \dots, q_{n-1}$, assim, para todo $1 \leq i \leq n$ tem-se que,

$$\psi'(q_{i-1}, a_i, q_i) = \psi(q_{i-1}, a_i, q_i)$$

agora para cada caminho de computação $q'_1, \dots, q'_{n-1}$ que possuir algum $q'_j \in \pi$ tem-se que em $\mathcal{M}'$ não existe tal caminho pois $q'_j \notin Q'$, assim,

$$\begin{aligned} \widehat{\psi}'(q_0, a_1 \cdots a_n, q_n) &= \bigsqcup_{q_1, \dots, q_{n-1} \in Q'} \left(\psi'(q_0, a_1, q_1) \otimes \cdots \otimes \psi'(q_{n-1}, a_n, q_n) \right) \\ &= \bigsqcup_{q_1, \dots, q_{n-1} \in Q - \pi} \left(\psi(q_0, a_1, q_1) \otimes \cdots \otimes \psi(q_{n-1}, a_n, q_n) \right) \\ &= \{0\} \end{aligned}$$

consequentemente,

$$\begin{aligned} \widehat{f}_{\mathcal{M}'}(w) &= \bigsqcup_{q_n \in Q - \pi} \left(\widehat{\psi}'(q_0, w, q_n) \otimes \mathcal{F}'(q_n) \right) \\ &= \bigsqcup_{q_n \in Q - \pi} \left(\{0\} \otimes \mathcal{F}'(q_n) \right) \\ &\stackrel{\text{Prop. 4.1.3}}{\stackrel{\text{Prop. 4.1.6}}{=}} \{0\} \end{aligned}$$

logo,

$$\widehat{f}_{\mathcal{M}'}(w) = \widehat{f}_{\mathcal{M}}(w) \quad (4.16)$$

portanto, pelas conclusões expostas nas Equações (4.14), (4.15) e (4.16) tem-se que $\widehat{f}_{\mathcal{M}} = \widehat{f}_{\mathcal{M}'}$. $\square$

Como consequência dos dois últimos resultados segue o corolário abaixo.

Corolário 4.2.1. *Se $\mathcal{M}$ é um AFHT, então existe um AFHTD $\mathcal{M}'$ sem estados nulos tal que $\mathcal{M} \trianglelefteq \mathcal{M}'$.*

Demonstração. Direto dos Teoremas 4.2.1 e 4.2.2. $\square$

Exemplo 4.2.3. *Considere o AFHT $\mathcal{M}$ da Figura 10 aplicando os Teoremas 4.2.1 e 4.2.2 é obtido o AFHTD $\mathcal{M}' = \langle Q', \{a, b\}, \psi', \{q_0\}, \mathcal{F}' \rangle$ onde $Q' = \{\{q_0\}, \{q_0, q_2\}, \{q_0, q_1, q_2\}\}$:*

$$\begin{aligned} \psi'(\{q_0\}, a, \{q_0, q_1, q_2\}) &= \{0.7, 0.8, 0.99, 0.75\} \\ \psi'(\{q_0\}, b, \{q_0, q_2\}) &= \{0.3, 0.4, 1\} \\ \psi'(\{q_0, q_1, q_2\}, a, \{q_0, q_1, q_2\}) &= \{0.7, 0.8, 0.99, 0.75\} \\ \psi'(\{q_0, q_1, q_2\}, b, \{q_0, q_1, q_2\}) &= \{0.3, 0.4, 1\} \end{aligned}$$

para as demais triplas $(S, x, S') \in Q' \times \Sigma \times Q'$ tem-se que $\psi'(S, x, S') = \{0\}$, além disso, tem-se:

$$\begin{aligned} \mathcal{F}'(\{q_0\}) &= \{0\} \\ \mathcal{F}'(\{q_0, q_1, q_2\}) &= \{0, 1, 0.9\} \\ \mathcal{F}'(\{q_0, q_2\}) &= \{0, 1, 0.9\} \end{aligned}$$

fazendo $n_1 = \{0.7, 0.8, 0.99, 0.75\}$, $n_2 = \{0.3, 0.4, 1\}$, $n_3 = \{0\}$ e $n_4 = \{0, 1, 0.9\}$, pode-se representar o AFHTD $\mathcal{M}'$ pela Figura 12 a seguir, ressaltando que as transições da forma $\varphi(X_i, a, X_j) = \{0\}$ são omitidas.

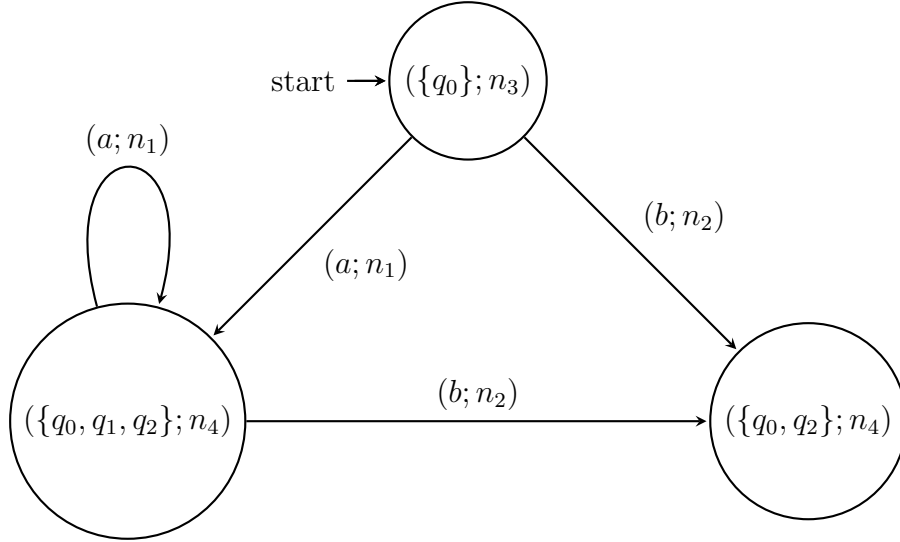


Figura 12 – AFHTD criado a partir do AFHT da Figura 10, pela aplicação dos Teoremas 4.2.1 e 4.2.2.

Agora será analisado o relacionamento de poder computacional existente entre os AFHT's e os autômatos *fuzzy*.

Teorema 4.2.3. *Se $\mathcal{M}$ é um AFHT, então existe uma família finita de autômatos fuzzy $\{M_k\}_{k \in K}$ tal que para todo $w \in \Sigma^*$ tem-se que $\hat{f}_{\mathcal{M}}(w) = \bigcup_{k \in K} f_{M_k}(w)$.*

Demonstração. Seja $\mathcal{M} = \langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$ um AFHT, como Q e Σ são conjuntos finitos tem-se que, $m = \#(Q \times \Sigma \times Q)$ com $m \in \mathbb{N}$ e $m > 0$. Agora considerando uma bijeção fixas entre todas as possíveis,

$$\rho : \{1, \dots, m\} \rightarrow Q \times \Sigma \times Q$$

dado que,

$$t = \prod_{i=1}^m \# \psi(\rho(i))$$

e

$$s = \prod_{q \in Q} \# \mathcal{F}(q)$$

é possível definir um conjunto finito $K = \{1, \dots, t\} \times \{1, \dots, s\}$. Agora observe que existem exatamente t diferentes funções $h_{k'} : \{1, \dots, m\} \rightarrow [0, 1]$ tal que $h_{k'}(i) \in \psi(\rho(i))$ para cada $i \in \{1, \dots, m\}$. Analogamente, existem exatamente s diferentes funções $h_{k''} : Q \rightarrow [0, 1]$

tal que $h_{k''}(q) \in \mathcal{F}(q)$ para cada $q \in Q$. Agora para qualquer $k = (k', k'') \in K$ existe um autômato fuzzy $M_k = \langle Q, \Sigma, \varphi_{k'}, q_0, h_{k''} \rangle$ onde,

$$\varphi_{k'}(q, a, q') = h_{k''}(\rho^{-1}(q, a, q'))$$

para cada $(q, a, q') \in (Q \times \Sigma \times Q)$. Observe que para todo AFHT $\mathcal{M}$ as condições a seguir são satisfeitas:

$$\mathcal{F}(q) = \bigcup_{k''=1}^s \{h_{k''}(q)\}$$

e

$$\psi(q, a, q') = \bigcup_{k'=1}^t \{h_{k'}(\rho^{-1}(q, a, q'))\} = \bigcup_{k'=1}^t \{\varphi_{k'}(q, a, q')\}$$

agora por definição para todo $w \in \Sigma^+$ com $w = a_1 \cdots a_n$ segue que,

$$\hat{f}_{\mathcal{M}}(w) = \bigsqcup_{q_n \in Q} \left(\hat{\psi}(q_0, w, q_n) \otimes \mathcal{F}(q_n) \right)$$

mas pela Observação 4.2.3 tem-se que,

$$\hat{\psi}(q_0, w, q_n) = \bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\psi(q_0, a_1, q_1) \otimes \cdots \otimes \psi(q_{n-1}, a_n, q_n) \right)$$

consequentemente,

$$\begin{aligned} \hat{\psi}(q_0, a_1 \cdots a_n, q_n) &= \bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\bigcup_{k'=1}^t \{\varphi_{k'}(q_0, a_1, q_1)\} \otimes \cdots \otimes \bigcup_{k'=1}^t \{\varphi_{k'}(q_{n-1}, a_n, q_n)\} \right) \\ &\stackrel{\text{Obs. 4.2.1}}{=} \bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\bigcup_{k'=1}^t \varphi_{k'}(q_0, a_1, q_1) \otimes \cdots \otimes \bigcup_{k'=1}^t \varphi_{k'}(q_{n-1}, a_n, q_n) \right) \\ &\stackrel{\text{Teo. 4.1.3}}{=} \bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\bigcup_{k'=1}^t \left(\varphi_{k'}(q_0, a_1, q_1) \otimes \cdots \otimes \varphi_{k'}(q_{n-1}, a_n, q_n) \right) \right) \\ &\stackrel{\text{Teo. 4.1.3}}{=} \bigcup_{k'=1}^t \left(\bigsqcup_{q_1, \dots, q_{n-1} \in Q} \left(\varphi_{k'}(q_0, a_1, q_1) \otimes \cdots \otimes \varphi_{k'}(q_{n-1}, a_n, q_n) \right) \right) \\ &\stackrel{\text{Obs. 4.1.1}}{=} \bigcup_{k'=1}^t \left(\bigvee_{q_1, \dots, q_{n-1} \in Q} \left(\varphi_{k'}(q_0, a_1, q_1) \wedge \cdots \wedge \varphi_{k'}(q_{n-1}, a_n, q_n) \right) \right) \\ &= \bigcup_{k'=1}^t \hat{\varphi}_{k'}(q_0, a_1 \cdots a_n, q_n) \end{aligned}$$

substituindo na equação de $\hat{f}_{\mathcal{M}}$ tem-se,

$$\begin{aligned}
 \hat{f}_{\mathcal{M}}(a_1 \cdots a_n) &= \bigsqcup_{q_n \in Q} \left(\bigcup_{k'=1}^t \hat{\varphi}_{k'}(q_0, a_1 \cdots a_n, q_n) \otimes \bigcup_{k''=1}^s \{h_{k''}(q_n)\} \right) \\
 &\stackrel{\text{Obs. 4.2.1}}{=} \bigsqcup_{q_n \in Q} \left(\bigcup_{k'=1}^t \hat{\varphi}_{k'}(q_0, a_1 \cdots a_n, q_n) \otimes \bigcup_{k''=1}^s h_{k''}(q_n) \right) \\
 &\stackrel{\text{Teo. 4.1.3}}{=} \bigsqcup_{q_n \in Q} \left(\bigcup_{k'=1}^t \bigcup_{k''=1}^s \left(\hat{\varphi}_{k'}(q_0, a_1 \cdots a_n, q_n) \otimes h_{k''}(q_n) \right) \right) \\
 &\stackrel{\text{Teo. 4.1.3}}{=} \bigcup_{k'=1}^t \bigcup_{k''=1}^s \left(\bigsqcup_{q_n \in Q} \left(\hat{\varphi}_{k'}(q_0, a_1 \cdots a_n, q_n) \otimes h_{k''}(q_n) \right) \right) \\
 &\stackrel{\text{Obs. 4.1.1}}{=} \bigcup_{k'=1}^t \bigcup_{k''=1}^s \left(\bigvee_{q_n \in Q} \left(\hat{\varphi}_{k'}(q_0, a_1 \cdots a_n, q_n) \wedge h_{k''}(q_n) \right) \right) \\
 &= \bigcup_{k \in K} f_{M_k}(a_1 \cdots a_n)
 \end{aligned}$$

a prova da igualdade para o caso em que $w = \lambda$ é trivial. Portanto, para todo $w \in \Sigma^*$ tem-se que, $\hat{f}_{\mathcal{M}}(w) = \bigcup_{k \in K} f_{M_k}(w)$. $\square$

Observação 4.2.6. *É claro que os autômatos fuzzy gerados no Teorema 4.2.3 são autômatos fuzzy simplificados, porém, pela Observação 3.2.7 sempre é possível construir uma família de autômatos fuzzy com a definição mais generalista (Definição 3.2.3).*

O Teorema 4.2.3 pode ser visto como uma forma de separar (dividir ou decompor) um AFHT em todos os seus autômatos fuzzy constituintes. Neste sentido, o Teorema 4.2.3 pode ser chamado de **Teorema da separação**. Além disso, o Teorema 4.2.3 esboça que o poder computacional de um único AFHT sobre o alfabeto Σ é corresponde ao poder computacional de uma família finita de autômatos fuzzy sobre o mesmo alfabeto Σ .

Exemplo 4.2.4. *Seja $\mathcal{M}$ um AFHT representado pela Figura 13, aplicando o Teorema 4.2.3 em $\mathcal{M}$ é obtida a família de autômatos fuzzy composta por $M_{(1,1)}$, $M_{(1,2)}$, $M_{(2,1)}$, $M_{(2,2)}$, $M_{(3,1)}$, $M_{(3,2)}$, $M_{(4,1)}$ e $M_{(4,2)}$ que estão representados nas Figuras 14a-14h.*

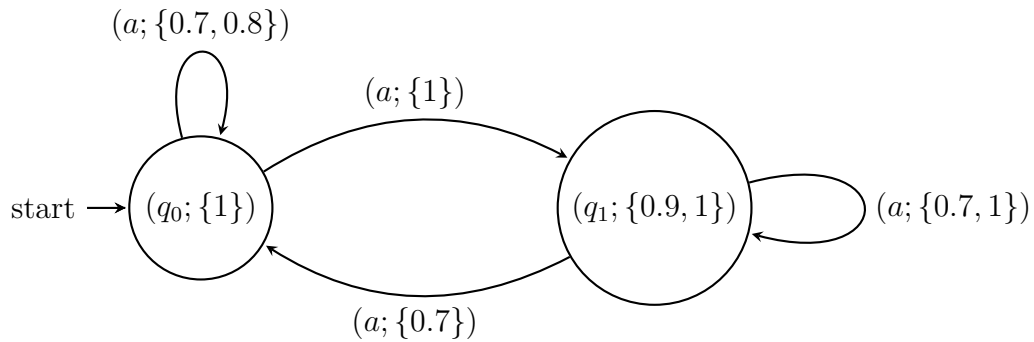


Figura 13 – Um AFHT sobre o alfabeto unitário $\{a\}$.

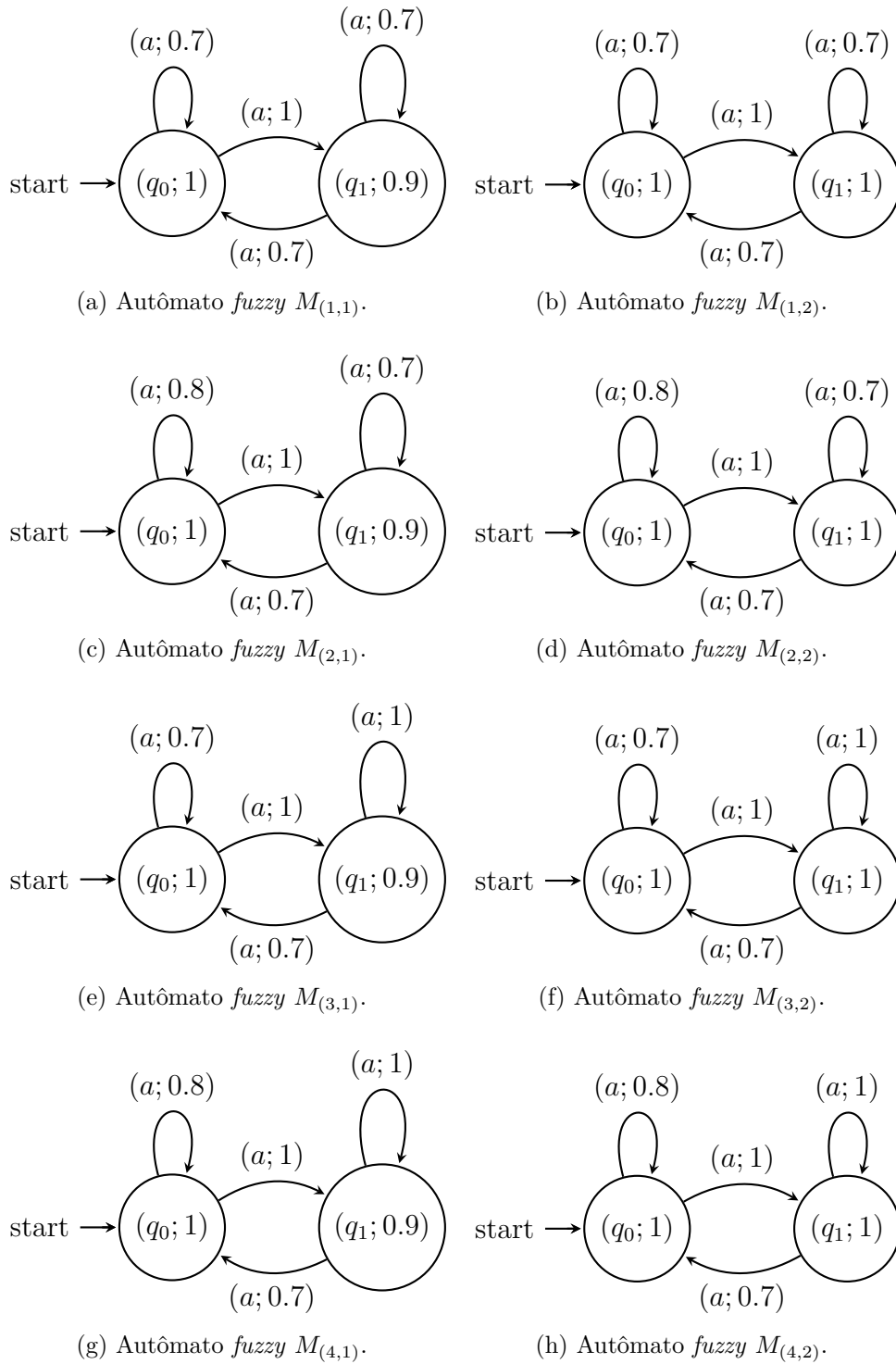


Figura 14 – Família de autômatos fuzzy gerada pela aplicação do Teorema 4.2.3 sobre o AFHT da Figura 13.

Outro ponto sobre o Teorema 4.2.3, é que ele possibilita apresentar uma interpretação sobre o que representa um AFHT. Lembre que como descrito na seção 3.2 os autômatos fuzzy são algoritmos fuzzy, isto é, são programas com incerteza, agora como a computação de um AFHT corresponde (pelo Teorema 4.2.3) à união finita da computação de vários autômatos fuzzy, isso diz que, um AFHT pode ser visto como sendo um sistema

computacional que executa diversos programas *fuzzy* em paralelo, ou seja, um AFHT pode ser visto como um grid computacional que executa programas *fuzzy*. Além disso, o Teorema 4.2.3 permite concluir o seguinte resultado.

Corolário 4.2.2. *A linguagem de qualquer AFHT é equivalente à união finita de linguagens fuzzy aceitas por autômatos fuzzy.*

Demonstração. Direto do Teorema 4.2.3. □

Vale observar que a união mencionada no Corolário 4.2.2 não é a união de linguagens *fuzzy* apresentada por Lee e Zadeh em (LEE; ZADEH, 1969), mas a operação padrão de união entre conjuntos presente na teoria ingênua dos conjuntos (ABE; PAPAVERO, 1991).

4.3 Caracterização das Linguagens $\mathbb{H}$ -Regulares.

Generalizando o conceito de linguagens *fuzzy* apresentado por Lee e Zadeh em (LEE; ZADEH, 1969), pode-se introduzir o conceito de linguagem *fuzzy* hesitante típica como se segue.

Definição 4.3.1. *Seja Σ um alfabeto, um CFHT $\hat{f} : \Sigma^* \rightarrow \mathbb{H}$ chama-se Linguagem Fuzzy Hesitante Típica (ou simplesmente LFHT). Uma LFHT é $\mathbb{H}$ -regular se existe um AFHT $\mathcal{M}$ tal que $\hat{f} = \hat{f}_{\mathcal{M}}$. O conjunto de todas as LFHT's é denotado por $\mathbb{T}$, e o conjunto de todas as LFHT's que são $\mathbb{H}$ -regulares é denotado por $\mathbb{T}_R$.*

Pela Definição 4.3.1 tem-se que, os AFHT's são modelos de computação que computam LFHT's que são $\mathbb{H}$ -regulares. Por conta dessa definição é natural fazer o seguinte questionamento:

Toda LFHT é $\mathbb{H}$ -regular?

Obviamente este questionamento é equivalente a pergunta se $\mathbb{T} = \mathbb{T}_R$. Como fruto da busca pela resposta deste questionamento, nesta seção será apresentada uma caracterização para as linguagens dos AFHT's, isto é, será apresentado uma caracterização dos elementos que compõem o conjunto $\mathbb{T}_R$, para isso considere a definição a seguir.

Definição 4.3.2. *Dado duas LFHT's que são $\mathbb{H}$ -regulares $\hat{f}_1 : \Sigma^* \rightarrow \mathbb{H}$ e $\hat{f}_2 : \Sigma^* \rightarrow \mathbb{H}$ a $\mathbb{H}$ -união de $\hat{f}_1$ e $\hat{f}_2$, denotada por $\hat{f}_1 \uplus \hat{f}_2$, é a LFHT definida por,*

$$\hat{f}_1 \uplus \hat{f}_2(w) = \hat{f}_1(w) \sqcup \hat{f}_2(w) \quad (4.17)$$

Agora será mostrado que a operação de $\mathbb{H}$ -união é um operador de fecho sobre o conjunto $\mathbb{T}_R$, isto é, a $\mathbb{H}$ -união de quaisquer duas LFHT's $\mathbb{H}$ -regulares é também uma LFHT que é $\mathbb{H}$ -regular.

Teorema 4.3.1. *Se $\hat{f}_1, \hat{f}_2 \in \mathbb{T}_R$, então existe um AFHT $\mathcal{M}$ tal que $\hat{f}_{\mathcal{M}} = \hat{f}_1 \uplus \hat{f}_2$.*

Demonstração. Assume que $\hat{f}_1 : \Sigma^* \rightarrow \mathbb{H}$ e $\hat{f}_2 : \Sigma^* \rightarrow \mathbb{H}$ são $\mathbb{H}$ -regulares, assim existem $\mathcal{M}_1 = \langle S, \Sigma, \psi_1, s_0, \mathcal{F}_1 \rangle$ e $\mathcal{M}_2 = \langle P, \Sigma, \psi_2, p_0, \mathcal{F}_2 \rangle$ tal que $\hat{f}_1 = \hat{f}_{\mathcal{M}_1}$ e $\hat{f}_2 = \hat{f}_{\mathcal{M}_2}$. Agora sem perda de generalidade assume que $S \cap P = \emptyset$, assim é possível definir um novo AFHT $\mathcal{M} = \langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$ onde,

(a) $Q = S \cup P \cup \{q_0\}$ com $q_0 \notin (S \cup P)$.

(b) Para todo $q \in Q$ tem-se que,

$$\mathcal{F}(q) = \begin{cases} \mathcal{F}_1(q), & \text{se } q \in S \\ \mathcal{F}_2(q), & \text{se } q \in P \\ \mathcal{F}_1(s_0) \sqcup \mathcal{F}_2(p_0), & \text{se } q = q_0 \end{cases} \quad (4.18)$$

(c) Para todo $q, q' \in Q$ e $a \in \Sigma$ tem-se que,

$$\psi(q, a, q') = \begin{cases} \psi_1(q, a, q'), & \text{se } q \in S, q' \in S \\ \psi_1(s_0, a, q'), & \text{se } q = q_0, q' \in S \\ \psi_2(q, a, q'), & \text{se } q, q' \in P \\ \psi_2(p_0, a, q'), & \text{se } q = q_0, q' \in P \\ \{0\}, & \text{qualquer outro caso} \end{cases} \quad (4.19)$$

agora é claro que,

$$\begin{aligned} \hat{f}_{\mathcal{M}}(\lambda) &\stackrel{\text{Rem. 4.2.4}}{=} \mathcal{F}(q_0) \\ &\stackrel{\text{Eq. (4.18)}}{=} \mathcal{F}_1(s_0) \sqcup \mathcal{F}_2(p_0) \\ &\stackrel{\text{Rem. 4.2.4}}{=} \hat{f}_{\mathcal{M}_1}(\lambda) \sqcup \hat{f}_{\mathcal{M}_2}(\lambda) \\ &= \hat{f}_1 \uplus \hat{f}_2(\lambda) \end{aligned} \quad (4.20)$$

além disso, para todo $w \in \Sigma^+$ com $w = a_1 a_2 \cdots a_n$ segue que,

$$\begin{aligned} \hat{f}_{\mathcal{M}}(a_1 a_2 \cdots a_n) &= \bigsqcup_{q_f \in Q} \left(\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_f) \otimes \mathcal{F}(q_f) \right) \\ &= \bigsqcup_{q_f \in Q - \{q_0\}} \left(\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_f) \otimes \mathcal{F}(q_f) \right), \\ &\quad \left(\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_0) \otimes \mathcal{F}(q_0) \right) \end{aligned} \quad (4.21)$$

mas pela Equação (4.19) e pela Proposição 4.1.3 é fácil ver que,

$$\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_0) = \{0\}$$

portanto,

$$\begin{aligned}
 \hat{f}_{\mathcal{M}}(a_1 a_2 \cdots a_n) &= \bigsqcup \left(\bigsqcup_{q_f \in Q - \{q_0\}} \left(\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_f) \otimes \mathcal{F}(q_f) \right), \right. \\
 &\quad \left. \left(\{0\} \otimes \mathcal{F}(q_0) \right) \right) \\
 &\stackrel{\text{Prop. 4.1.3}}{=} \bigsqcup_{q_f \in Q - \{q_0\}} \left(\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_f) \otimes \mathcal{F}(q_f) \right) \quad (4.22) \\
 &= \bigsqcup \left(\bigsqcup_{q_f \in S} \left(\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_f) \otimes \mathcal{F}(q_f) \right), \right. \\
 &\quad \left. \bigsqcup_{q_f \in P} \left(\hat{\psi}(q_0, a_1 a_2 \cdots a_n, q_f) \otimes \mathcal{F}(q_f) \right) \right) \\
 &= \hat{f}_{\mathcal{M}_1}(w) \sqcup \hat{f}_{\mathcal{M}_2}(w) \\
 &= \hat{f}_1 \uplus \hat{f}_2(w)
 \end{aligned}$$

consequentemente, pelas Equações (4.20) e (4.22) tem-se que, $\hat{f}_{\mathcal{M}} = \hat{f}_1 \uplus \hat{f}_2$, complementando assim a prova. $\square$

Exemplo 4.3.1. Seja $\mathcal{M}_1$ e $\mathcal{M}_2$ os AFHT's representados respectivamente pelas Figuras 15a e 15b a seguir.

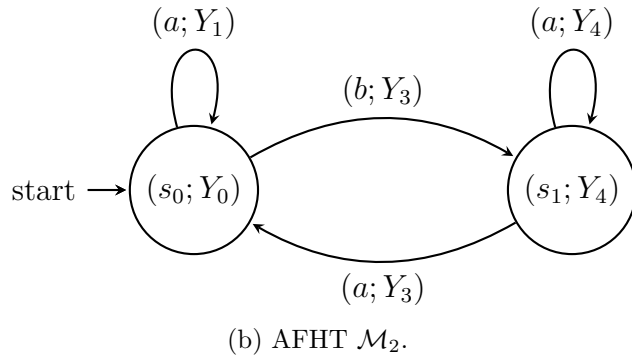
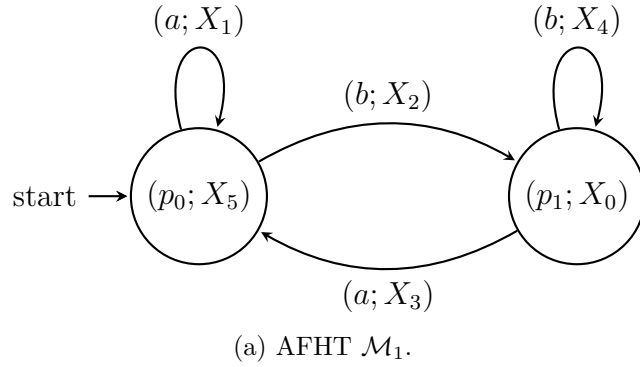


Figura 15 – Dois AFHT's sobre o alfabeto $\{a, b\}$.

O AFHT $\mathcal{M}$ representado pela Figura 16 é construído aplicando o método descrito na prova do Teorema 4.3.1 e o mesmo computa exatamente a $\mathbb{H}$ -união de $\hat{f}_{\mathcal{M}_1}$ e $\hat{f}_{\mathcal{M}_2}$.

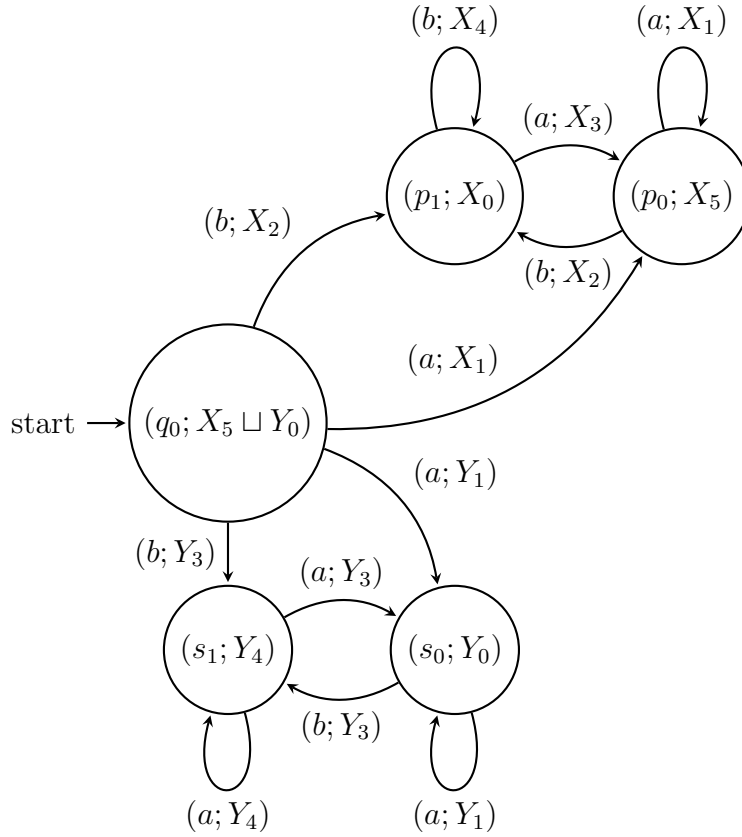


Figura 16 – AFHT construído a partir dos AFHT's das Figuras 15a e 15b, seguindo os passos descritos na prova do Teorema 4.3.1.

Vale ressaltar, que como dito anteriormente a *sup-combinação* é um combinador do tipo *passa alta*, assim pela definição de $\mathbb{H}$ -união as computações realizada por $\mathcal{M}$ na verdade correspondem a “filtragens” realizadas sobre as computações de $\mathcal{M}_1$ e $\mathcal{M}_2$, no sentido de que, $\mathcal{M}$ sempre irá preservar os maiores valores de pertinências apresentados por $\mathcal{M}_1$ e $\mathcal{M}_2$.

Como consequência direta do Teorema 4.3.1 segue o seguinte corolário.

Corolário 4.3.1. Se $\{\hat{f}_i\}_{i \in I}$ é uma família finita de LFHT's que são $\mathbb{H}$ -regulares, então existe um AFHT $\mathcal{M}$ tal que $\hat{f}_{\mathcal{M}} = \bigsqcup_{i \in I} \hat{f}_i$.

Demonstração. Por indução sobre o conjunto de índices I e usando o Teorema 4.3.1. $\square$

Definição 4.3.3. Dado uma LFHT $\hat{f} : \Sigma^* \rightarrow \mathbb{H}$ e $k \in \mathbb{H}$, são definidos os conjuntos $R_{\hat{f}}$ e $S_{\hat{f}}^k$ como se segue,

$$(a) \ R_{\hat{f}} = \{X \in \mathbb{H} \mid \hat{f}(w) = X, w \in \Sigma^*\}.$$

$$(b) \ S_{\hat{f}}^k = \{w \in \Sigma^* \mid k \sqsubseteq \hat{f}(w)\}.$$

O próximo resultado estabelece as condições suficientes e necessárias para que uma LFHT seja $\mathbb{H}$ -Regular.

Teorema 4.3.2. *Se $\hat{f} : \Sigma^* \rightarrow \mathbb{H}$ é uma LFHT. Então as seguintes afirmações são equivalentes.*

(i) $\hat{f}$ é $\mathbb{H}$ -regular.

(ii) $R_{\hat{f}}$ é finito e para cada $k \in R_{\hat{f}}$ o conjunto $S_{\hat{f}}^k$ é uma linguagem regular.

Demonstração. (i) $\Rightarrow$ (ii) Suponha que $\hat{f} : \Sigma^* \rightarrow \mathbb{H}$ é $\mathbb{H}$ -regular, assim existe um AFHT $\mathcal{M} = \langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$ tal que $\hat{f} = \hat{f}_{\mathcal{M}}$, isto é, para todo $w \in \Sigma^*$ tem-se que,

$$\hat{f}(w) = \bigsqcup_{q_n \in Q} \left(\hat{\psi}(q_0, w, q_n) \otimes \mathcal{F}(q_n) \right)$$

mas pela definição de $\mathcal{M}$ tem-se que $(Q \times \Sigma \times Q)$ é finito, portanto, é claro que R_{ψ} e $R_{\mathcal{F}}$ são conjuntos finitos. Além disso, pelas Definições de $\sqcup$ e $\otimes$ é obvio que $R_{\hat{\psi}}$ também é um conjunto finito, logo o conjunto $R_{\hat{f}}$ é finito. Agora para todo $k \in R_{\hat{f}}$ definimos um AFD $A_k = \langle Q, \Sigma, q_0, \delta_k, F_k \rangle$ onde

$$\delta_k(q, a) = p \iff k \sqsubseteq \psi(q, a, p) \quad (4.23)$$

e

$$q \in F_k \iff k \sqsubseteq \mathcal{F}(q) \quad (4.24)$$

agora para todo $w \in \Sigma^*$ segue que,

$$\begin{aligned} w \in L(A_k) &\iff \hat{\delta}_k(q_0, w) \in F_k \\ &\iff \exists q_1, \dots, q_{n-1}, q_n \in Q \text{ tal que } q_n \in F_k \text{ e} \\ &\quad \delta_k(q_0, a_1) = q_1, \dots, \delta_k(q_{n-1}, a_n) = q_n \\ &\stackrel{\text{Eq. (4.23)}}{\iff} \exists q_1, \dots, q_{n-1}, q_n \in Q \text{ tal que } k \sqsubseteq \mathcal{F}(q_n) \text{ e} \\ &\quad k \sqsubseteq \mu(q_0, a_1, q_1), \dots, k \sqsubseteq \mu(q_{n-1}, a_n, q_n) \\ &\iff \exists q_n \in Q \text{ tal que } k \sqsubseteq \mathcal{F}(q_n) \text{ e } k \sqsubseteq \hat{\mu}(q_0, w, q_n) \\ &\stackrel{\text{Prop. 4.1.14}}{\iff} \exists q_n \in Q \text{ tal que } k \sqsubseteq \left(\mathcal{F}(q_n) \otimes \hat{\mu}(q_0, w, q_n) \right) \\ &\stackrel{\text{Teq. 4.1.4}}{\iff} k \sqsubseteq \bigsqcup_{q_n \in Q} \left(\hat{\mu}(q_0, w, q_n) \otimes \mathcal{F}(q_n) \right) \\ &\iff k \sqsubseteq \hat{f}(w) \\ &\iff w \in S_{\hat{f}}^k \end{aligned}$$

consequentemente, $L(A_k) = S_{\hat{f}}^k$. E desde que A_k é um AFD, o conjunto $S_{\hat{f}}^k$ é uma linguagem regular.

(ii) $\Rightarrow$ (i) Assume que $R_{\hat{f}} = \{k_1, \dots, k_n\}$ para algum $n \in \mathbb{Z}_+^*$ e que para todo $k \in R_{\hat{f}}$ o conjunto S_f^k é uma linguagem regular, consequentemente, existe uma família finita de AFD's $\{A_k\}_{k \in R_{\hat{f}}}$ onde $A_k = \langle Q_k, \Sigma, \delta_k, q_0^k, F_k \rangle$ tal que $L(A_k) = S_f^k$, agora para cada $k \in R_{\hat{f}}$ defina um AFHT $\mathcal{M}_k = \langle Q_k, \Sigma, \psi_k, q_0^k, \mathcal{F}_k \rangle$ onde para cada $q, q' \in Q_k$ e $a \in \Sigma$ tem-se que:

$$\psi_k(q, a, q') = \begin{cases} \{1\}, & \text{se } \delta_k(q, a) = q' \\ \{0\}, & \text{senão} \end{cases} \quad (4.25)$$

e

$$\mathcal{F}_k(q) = \begin{cases} k, & \text{se } q \in F_k \\ \{0\}, & \text{senão} \end{cases} \quad (4.26)$$

agora é óbvio pela construção de $\mathcal{M}_k$ que para todo $w \in \Sigma^*$,

$$\hat{f}_{\mathcal{M}_k}(w) = \begin{cases} k, & \text{se } w \in L(A_k) \\ \{0\}, & \text{senão} \end{cases}$$

além disso, desde que $\mathcal{M}_k$ é um AFHT tem-se que cada $\hat{f}_{\mathcal{M}_k}$ é $\mathbb{H}$ -Regular. Como $R_{\mathcal{L}}$ é finito pelo Corolário 4.3.1 existe um AFHT $\mathcal{M}$ tal que,

$$\hat{f}_{\mathcal{M}} = \biguplus_{k \in R_{\hat{f}}} \hat{f}_{\mathcal{M}_k}$$

e é claro que $\hat{f}_{\mathcal{M}} = \hat{f}$, portanto, $\hat{f}$ é $\mathbb{H}$ -regular. $\square$

Observação 4.3.1. Note que a Equação (4.25) garante que todos os AFHT's gerados na segunda parte do Teorema 4.3.2, possuem a característica de que para toda tripla (q, a, q') tem-se que $\psi(q, a, q') = \{1\}$ ou $\psi(q, a, q') = \{0\}$.

O resultado apontado pelo Teorema 4.3.2 implica na seguinte conclusão.

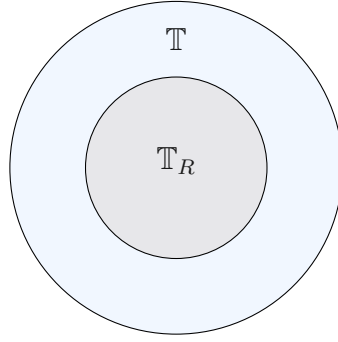
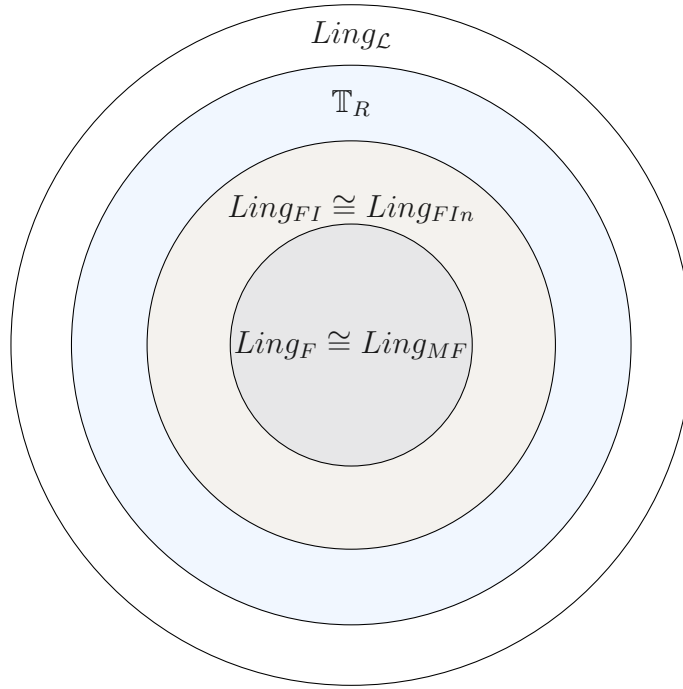
Corolário 4.3.2. Existem LFHT's que não são $\mathbb{H}$ -regulares.

Demonstração. É suficiente observar que a LFHT $\hat{f} : \Sigma^* \rightarrow \mathbb{H}$ definida por:

$$\hat{f}(w) = \bigcup_{i=0}^{|w|} \left\{ \frac{1}{(2^i + 1)} \right\}$$

é tal que o conjunto $R_{\hat{f}}$ é infinito. Consequentemente, pelo Teorema 4.3.2, tem-se que $\hat{f}$ não é $\mathbb{H}$ -regular. $\square$

Observe que, o Corolário 4.3.2 apresenta uma resposta negativa para o questionamento feito anteriormente, se toda LFHT era $\mathbb{H}$ -regular. Na Figura 17 a seguir, é possível visualizar o relacionamento de inclusão comprovado pelo Corolário 4.3.2 para os conjuntos $\mathbb{T}$ e $\mathbb{T}_R$.

Figura 17 – Hierarquia entre os conjuntos $\mathbb{T}$ e $\mathbb{T}_R$.Figura 18 – Extensão da hierarquia de linguagens da Figura 7, considerando as LFHT $\mathbb{H}$ -regulares.

Agora de forma similar ao que foi feita para o conjunto de todas as linguagens probabilísticas por Rabin em (RABIN, 1963), será mostrado que o conjunto $\mathbb{T}_R$ não é enumerável.

Teorema 4.3.3. *O conjunto $\mathbb{T}_R$ não é enumerável.*

Demonstração. Para cada $x \in [0, 1]$ defina um AFHT $\mathcal{M}_x = \langle \{q_0^x, q_1^x\}, \Sigma, \psi, q_0^x, \mathcal{F}_x \rangle$ onde para todo $q, q' \in \{q_0^x, q_1^x\}$ e $a \in \Sigma$ tem-se que:

$$\psi(q, a, q') = \{x\}$$

e

$$\mathcal{F}_x(q) = \{x\}$$

obviamente para todo $w \in \Sigma^*$ tem-se que $\widehat{f}_{\mathcal{M}_x}(w) = \{x\}$. Agora construa o conjunto $\Theta = \{\widehat{f}_{\mathcal{M}_x} \mid x \in [0, 1]\}$, é claro que $\Theta \subset \mathbb{T}_R$ e que existe uma bijeção entre Θ e o intervalo $[0, 1]$, logo Θ não é enumerável, e portanto, $\mathbb{T}_R$ também não é enumerável. $\square$

Agora levando em consideração as LFHT's $\mathbb{H}$ -regulares, pode-se estender a hierarquia de linguagens apresentada anteriormente na Figura 7 para uma nova hierarquia de linguagens descrita pela Figura 18, aqui o relacionamento de inclusão entre as teorias das LFHT's $\mathbb{H}$ -regulares e as linguagens *fuzzy* intervalares é similar à relação de inclusão entre as teorias dos CFHT e dos conjuntos *fuzzy* intervalares que foi descrita nos trabalhos (LIMA, 2019; BUSTINCE et al., 2016).

4.4 Pseudo Autômatos Fuzzy Hesitante Típicos

Nesta seção será apresentada uma segunda formulação de autômatos cujo o espaço de valoração corresponde ao semi-anel $\mathbb{H}$, um autômato desta nova formulação é chamado de Pseudo Autômato Fuzzy Hesitante Típico (ou simplesmente PAFHT), e é formalmente definido como se segue.

Definição 4.4.1. *Um PAFHT é uma tupla $\mathcal{N} = \langle Q, \Sigma, \delta_{\mathcal{N}}, q_0, \mathcal{F} \rangle$ onde Q, Σ, q_0 e $\mathcal{F}$ são da mesma forma que na Definição 4.2.1 e $\delta_{\mathcal{N}}$ é uma função de transição igual aquela apresentada pela Definição 2.1.6.*

Os Pseudo Autômatos Fuzzy Hesitante Típicos (PAFHT's) apresentam em sua definição uma maior simplicidade que os AFHT's, uma vez que, o único conjunto valorado que tais autômatos possuem é exatamente o CFHT $\mathcal{F}$, o que contrasta como os AFHT's que possuem em sua definição dois conjuntos valorados, sendo eles, ψ e $\mathcal{F}$.

Definição 4.4.2. *Seja $\mathcal{N} = \langle Q, \Sigma, \delta_{\mathcal{N}}, q_0, \mathcal{F} \rangle$ um PAFHT a LHFT aceita (ou computada) por $\mathcal{N}$ é o CFHT definido por:*

$$\widehat{f}_{\mathcal{N}}(w) = \bigsqcup_{q \in \widehat{\delta}_{\mathcal{N}}(q_0, w)} \mathcal{F}(q) \quad (4.27)$$

para todo $w \in \Sigma^*$.

Agora será mostrado que qualquer LFHT aceita (ou computada) por um PAFHT é $\mathbb{H}$ -regular.

Teorema 4.4.1. *Se $\mathcal{N}$ é um PAFHT, então existe um AFHT $\mathcal{M}$ tal que $\widehat{f}_{\mathcal{N}} = \widehat{f}_{\mathcal{M}}$.*

Demonstração. É suficiente notar que dado um PAFHT $\mathcal{N} = \langle Q, \Sigma, \delta_{\mathcal{N}}, q_0, \mathcal{F} \rangle$, é possível definir um AFHT $\mathcal{M} = \langle Q, \Sigma, \mu, q_0, \mathcal{F} \rangle$ onde:

$$\psi(q, a, q') = \begin{cases} \{1\}, & \text{se } q' \in \delta_{\mathcal{N}}(q, a) \\ \{0\}, & \text{senão} \end{cases} \quad (4.28)$$

para todo $q, q' \in Q$ e $a \in \Sigma$. Agora é obvio que para todo $w \in \Sigma^*$ tem-se que, $\hat{f}_{\mathcal{N}}(w) = \hat{f}_{\mathcal{M}}(w)$, consequentemente, $\hat{f}_{\mathcal{N}} = \hat{f}_{\mathcal{M}}$. $\square$

O próximo resultado estabelece o mecanismo inverso ao apresentado pelo Teorema 4.4.1, isto é, será mostrado que para qualquer $\hat{f}$ que é $\mathbb{H}$ -regular existe um PAFHT que computa $\hat{f}$.

Teorema 4.4.2. *Se $\hat{f}$ é $\mathbb{H}$ -regular, então existe um PAFHT $\mathcal{N}$ tal que $\hat{f} = \hat{f}_{\mathcal{N}}$.*

Demonstração. Assuma que $\hat{f}$ é uma LFHT $\mathbb{H}$ -regular, logo usando a segunda parte da demonstração do Teorema 4.3.2 é possível obter um AFHT $\mathcal{M} = \langle Q, \Sigma, \psi, q_0, \mathcal{F} \rangle$, e como dito na Observação 4.3.1 tal AFHT que possui a restrição que: $\psi(q, a, q') = \{1\}$ ou $\psi(q, a, q') = \{0\}$ para todo $q, q' \in Q$ e $a \in \Sigma$. Agora defina o seguinte PAFHT $\mathcal{N} = \langle Q \cup \{q_{\mathbb{N}}\}, \Sigma, \delta_{\mathcal{N}}, q_0, \mathcal{F}_{\mathcal{N}} \rangle$ com $q_{\mathbb{N}} \notin Q$ e $\delta_{\mathcal{N}}$ sendo definida usando as regras a seguir.

(r1) Se $\psi(q, a, q') = \{1\}$, então $q' \in \delta_{\mathcal{N}}(q, a)$ para qualquer $q, q' \in Q$ e $a \in \Sigma$.

(r2) Se $\psi(q, a, q') = \{0\}$, então $q_{\mathbb{N}} \in \delta_{\mathcal{N}}(q, a)$ para qualquer $q, q' \in Q$ e $a \in \Sigma$.

(r3) $\delta_{\mathcal{N}}(q_{\mathbb{N}}, a) = \{q_{\mathbb{N}}\}$ para todo $a \in \Sigma$.

além disso, defina $\mathcal{F}_{\mathcal{N}}$ como sendo

$$\mathcal{F}_{\mathcal{N}}(q) = \begin{cases} \mathcal{F}(q), & \text{se } q \in Q \\ \{0\}, & \text{senão} \end{cases} \quad (4.29)$$

agora por esta construção não é difícil verificar que, $\hat{f}_{\mathcal{M}}(w) = \hat{f}_{\mathcal{N}}(w)$, para qualquer que seja o $w \in \Sigma^*$, completando assim a prova. $\square$

Através dos resultados anteriores obtém-se a seguinte conclusão.

Corolário 4.4.1. *Uma linguagem $\hat{f} : \Sigma^* \rightarrow \mathbb{H}$ é $\mathbb{H}$ -regular se, e somente se, existe um PAFHT $\mathcal{N}$ tal que $\hat{f} = \hat{f}_{\mathcal{N}}$.*

Demonstração. Direto dos Teoremas 4.4.1 e 4.4.2. $\square$

Observação 4.4.1. *Observe que embora os PAFHT's e os AFHT's sejam equivalentes em termos de linguagens, os PAFHT's necessitam de pelo menos mais um estado (ver a demonstração do Teorema 4.4.2) que os AFHT's para realizar sua computação.*

Definição 4.4.3. *Um PAFHT $\mathcal{N}$ é dito ser determinístico, ou simplesmente PAFHTD, se sua função de transição $\delta_{\mathcal{N}}$ é determinística no sentido clássico, isto é, $\#\delta_{\mathcal{N}}(q, a) \leq 1$ para todo $(q, a) \in (Q \times \Sigma)$, ou seja, $\delta_{\mathcal{N}}$ é igual à Definição 2.1.3.*

Observação 4.4.2. Note que para qualquer que seja o PAFHTD $\mathcal{D}$, para todo $w \in \Sigma^*$ tem-se que $\widehat{f}_{\mathcal{D}}(w) = \mathcal{F}(\widehat{\delta_{\mathcal{D}}}(q_0, w))$.

Lema 4.4.1. Para cada PAFHT $\mathcal{N} = \langle Q, \Sigma, \delta_{\mathcal{N}}, q_0, \mathcal{F} \rangle$, existe um PAFHTD $\mathcal{D}$ com $2^{\#Q}$ estados tal que $\widehat{f}_{\mathcal{N}} = \widehat{f}_{\mathcal{D}}$.

Demonstração. É suficiente notar que dado um PAFHT $\mathcal{N} = \langle Q, \Sigma, \delta_{\mathcal{N}}, q_0, \mathcal{F} \rangle$, é possível definir um PAFHTD $\mathcal{D} = \langle \wp(Q), \Sigma, \delta_{\mathcal{D}}, \{q_0\}, \mathcal{F}_{\mathcal{D}} \rangle$ onde para todo $X \in \wp(Q)$, $a \in \Sigma$ tem-se que:

$$\delta_{\mathcal{D}}(X, a) = \bigcup_{q \in X} \delta_{\mathcal{N}}(q, a)$$

e

$$\mathcal{F}_{\mathcal{D}}(X) = \begin{cases} \{0\}, & \text{se } X = \emptyset \\ \bigsqcup_{q \in X} \mathcal{F}(q), & \text{senão} \end{cases}$$

agora para qualquer $q \in \widehat{\delta_{\mathcal{N}}}(q_0, w)$ tem-se que existe um $X \in \wp(Q)$ tal que $q \in X$ e $\widehat{\delta_{\mathcal{D}}}(\{q_0\}, w) = X$, conseqüentemente, é claro que $\mathcal{L}_{\mathcal{N}}(w) = \mathcal{L}_{\mathcal{D}}(w)$. Além disso, pela construção de $\mathcal{D}$ é claro que $\mathcal{D}$ possui $2^{\#Q}$ estados, completando assim a prova. $\square$

Na Seção 4.2, foi apresentada uma maneira de remover o não determinismo dos AFHT e assim converte-lós para AFHTD, entretanto, esse método não preservava a linguagem aceita pelo autômato original, em vez disso o método aumentava o grau de hesitação da máquina resultante em relação a hesitação da máquina original. Agora será mostrado um método alternativo usando de PAFHT e PAFHTD que é capaz de converter um AFHT em um AFHTD preservando a linguagem.

Teorema 4.4.3. Para todo AFHT $\mathcal{M}$ existe um AFHTD $\mathcal{M}'$ tal que $\widehat{f}_{\mathcal{M}} = \widehat{f}_{\mathcal{M}'}$.

Demonstração. Seja $\mathcal{M}$ um AFHT pelo Teorema 4.4.2 existe um PAFHT $\mathcal{N}$ tal que $\widehat{f}_{\mathcal{M}} = \widehat{f}_{\mathcal{N}}$, mas pelo Lema 4.4.1, existe um PAFHTD $\mathcal{D}$ tal que $\widehat{f}_{\mathcal{N}} = \widehat{f}_{\mathcal{D}}$. Agora pelo Teorema 4.4.1, é possível a partir de $\mathcal{D}$ obter um AFHT $\mathcal{M}'$ tal que $\widehat{f}_{\mathcal{D}} = \widehat{f}_{\mathcal{M}'}$, desde que $\mathcal{D}$ era um PAFHTD é claro que $\mathcal{M}'$ é um AFHTD, além disso,

$$\widehat{f}_{\mathcal{M}} = \widehat{f}_{\mathcal{N}} = \widehat{f}_{\mathcal{D}} = \widehat{f}_{\mathcal{M}'}$$

o que completa a prova. $\square$

Um problema comum encontrado na literatura da teoria dos autômatos é a minimização de autômatos. Agora será apresentado as noções de redução de estados para os PAFHTD's, para isso será preciso considerar a definição a seguir.

Definição 4.4.4. *Seja $\mathcal{D} = \langle Q, \Sigma, \delta_{\mathcal{D}}, q_0, \mathcal{F} \rangle$ um PAFHTD, dois estados $q, q' \in Q$ são ditos equivalentes, denotado por $q \approx q'$, se e somente se, $\mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q, w)) = \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q', w))$ para todo $w \in \Sigma^*$.*

Proposição 4.4.1. *Seja $\mathcal{D} = \langle Q, \Sigma, \delta_{\mathcal{D}}, q_0, \mathcal{F} \rangle$ um PAFHTD. Se $q \approx q'$, então $\mathcal{F}(q) = \mathcal{F}(q')$.*

Demonstração. É suficiente notar que pelo fato de $\delta_{\mathcal{D}}$ ser uma função de transição determinística não existem λ -transições, portanto, para todo par de estados q e q' segue que,

$$q \approx q' \iff (\forall w \in \Sigma^*) \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q, w)) = \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q', w)) \Rightarrow \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q, \lambda)) = \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q', \lambda))$$

mas pela definição de função de transição estendida tem-se que,

$$\mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q, \lambda)) = \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q', \lambda)) \iff \mathcal{F}(q) = \mathcal{F}(q')$$

o que completa a prova. $\square$

Teorema 4.4.4. *Se $\mathcal{D} = \langle Q, \Sigma, \delta_{\mathcal{D}}, q_0, \mathcal{F} \rangle$ é um PAFHTD e existem dois estados q e q' tal que $q \approx q'$, então existe um PAFHTD $\mathcal{D}' = \langle Q', \Sigma, \delta_{\mathcal{D}'}, q'_0, \mathcal{F}' \rangle$ tal que $\widehat{f}_{\mathcal{D}} = \widehat{f}_{\mathcal{D}'}$ e $\#Q' = \#Q - 1$.*

Demonstração. Suponha que $\mathcal{D} = \langle Q, \Sigma, \delta_{\mathcal{D}}, q_0, \mathcal{F} \rangle$ é um PAFHTD e que existem dois estados q_i e q_j que são equivalentes, assim defina $\mathcal{D}' = \langle Q', \Sigma, \delta_{\mathcal{D}'}, q'_0, \mathcal{F}' \rangle$ tal que $Q' = Q - \{q_j\}$ com $\mathcal{F}' = \mathcal{F}|_{Q'}$ e,

$$\delta'_{\mathcal{D}'}(q, a) = \begin{cases} \delta'_{\mathcal{D}}(q, a) & \text{se } \delta'_{\mathcal{D}}(q, a) \neq q_j \\ q_i & \text{senão} \end{cases} \quad (4.30)$$

agora note que a Equação (4.30) garante que sempre que $\widehat{\delta}_{\mathcal{D}}(q_0, xa) = q_j$, tem-se que $\widehat{\delta}_{\mathcal{D}'}(q_0, xa) = q_i$ para todo $x \in \Sigma^*, a \in \Sigma$. Por outro lado, por hipótese q_i e q_j são equivalentes, logo para qualquer que seja $y \in \Sigma^*$ tem-se que $\mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q_j, y)) = \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q_i, y))$, agora pela definição de $\mathcal{F}'$ é claro que $\mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q_i, y)) = \mathcal{F}'(\widehat{\delta}_{\mathcal{D}'}(q_i, y))$, e portanto, para todo $xay \in \Sigma^+$ tem-se que $\widehat{f}_{\mathcal{D}}(w) = \widehat{f}_{\mathcal{D}'}(w)$. E para o caso da palavra vazia a igualdade $\widehat{f}_{\mathcal{D}}(\lambda) = \widehat{f}_{\mathcal{D}'}(\lambda)$ é trivial, portanto, $\widehat{f}_{\mathcal{D}} = \widehat{f}_{\mathcal{D}'}$, por fim, é claro que $\#Q' = \#Q - 1$. $\square$

Observação 4.4.3. *Um PAFHTD $\mathcal{D}$ é irreduzível quando para todo $q \in Q$ tem-se $[q] = \{q\}$.*

Corolário 4.4.2. *Para toda LFHT $\mathbb{H}$ -regular $\widehat{f}$ existe um PAFHTD $\mathcal{D}$ irreduzível tal que $\widehat{f} = \widehat{f}_{\mathcal{D}}$.*

Demonstração. Suponha que $\widehat{f}$ é uma LFHT $\mathbb{H}$ -regular, assim existe um PAFHTD $\mathcal{D} = \langle Q, \Sigma, \delta_{\mathcal{D}}, q_0, \mathcal{F} \rangle$ tal que $\widehat{f} = \widehat{f}_{\mathcal{D}}$, agora defina um novo PAFHTD $\mathcal{D}' = \langle Q_{/\approx}, \Sigma, \delta_{\mathcal{D}'}, q_0, \mathcal{F}' \rangle$ onde:

$$\delta_{\mathcal{D}'}([q], a) = [\delta_{\mathcal{D}}(q, a)] \quad (4.31)$$

e

$$\mathcal{F}'([q]) = \mathcal{F}(q) \quad (4.32)$$

agora pela Equação (4.31) e a definição de $\widehat{\delta}_{\mathcal{D}}$ tem-se que,

$$\widehat{\delta}_{\mathcal{D}'}([q_0], w) = [\widehat{\delta}_{\mathcal{D}}(q_0, w)] \quad (4.33)$$

além disso, é claro que para todo $w \in \Sigma^*$ tem-se:

$$\begin{aligned} \widehat{f}_{\mathcal{D}}(w) &\stackrel{Obs. 4.4.2}{=} \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q_0, w)) \\ &\stackrel{Eq. (4.32)}{=} \mathcal{F}'([\widehat{\delta}_{\mathcal{D}}(q_0, w)]) \\ &\stackrel{Eq. (4.33)}{=} \mathcal{F}'(\widehat{\delta}_{\mathcal{D}'}([q_0], w)) \\ &\stackrel{Obs. 4.4.2}{=} \widehat{f}_{\mathcal{D}'}(w) \end{aligned}$$

logo $\widehat{f} = \widehat{f}_{\mathcal{D}'}$. Agora para mostrar que $\mathcal{D}'$ é irreduzível, suponha que existem dois estados $[q], [q'] \in Q_{/\approx}$ tal que $[q] \approx [q']$ e $[q] \neq [q']$, como $[q]$ e $[q']$ são distintos tem-se pelo resultado (ABE; PAPAVERO, 1991, Corolário 12.4.2.2) que $[q] \cap [q'] = \emptyset$, mas pela definição de $\approx$ tem-se que,

$$\begin{aligned} [q] \approx [q'] &\iff [q'] \in [[q]] \\ &\iff (\forall w \in \Sigma^*) \mathcal{F}'(\widehat{\delta}_{\mathcal{D}'}([q], w)) = \mathcal{F}'(\widehat{\delta}_{\mathcal{D}'}([q'], w)) \\ &\stackrel{Eq. (4.33)}{\iff} (\forall w \in \Sigma^*) \mathcal{F}'([\widehat{\delta}_{\mathcal{D}}(q, w)]) = \mathcal{F}'([\widehat{\delta}_{\mathcal{D}}(q', w)]) \\ &\stackrel{Eq. (4.32)}{\iff} (\forall w \in \Sigma^*) \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q, w)) = \mathcal{F}(\widehat{\delta}_{\mathcal{D}}(q', w)) \\ &\iff q \approx q' \\ &\iff q' \in [q] \\ &\iff q \in [q'] \end{aligned}$$

assim $[q] \cap [q'] \neq \emptyset$, mais isso contradiz a hipótese inicial, e portanto, não existem dois estados distintos $[q], [q'] \in Q_{/\approx}$ tais que $[q] \approx [q']$, logo para todo $q \in Q_{/\approx}$ tem-se $[[q]] = \{[q]\}$, consequentemente, $\mathcal{D}'$ é irreduzível. $\square$

Assim como a operação de $\mathbb{H}$ -união agora será definida a operação $\mathbb{H}$ -interseção, em seguida será mostrado que tal operação é fechada sobre o conjunto $\mathbb{T}_R$.

Definição 4.4.5. Dado duas LFHT's que são $\mathbb{H}$ -regulares $\widehat{f}_1 : \Sigma^* \rightarrow \mathbb{H}$ e $\widehat{f}_2 : \Sigma^* \rightarrow \mathbb{H}$ a $\mathbb{H}$ -interseção de $\widehat{f}_1$ e $\widehat{f}_2$, denotada por $\widehat{f}_1 \nabla \widehat{f}_2$, é a LFHT definida por,

$$\widehat{f}_1 \nabla \widehat{f}_2(w) = \widehat{f}_1(w) \otimes \widehat{f}_2(w) \quad (4.34)$$

Teorema 4.4.5. *Se $\hat{f}_1$ e $\hat{f}_2$ são $\mathbb{H}$ -regulares. Então existe um PAFHTD $\mathcal{M}$ tal que $\hat{f}_{\mathcal{M}} = \hat{f}_1 \nabla \hat{f}_2$.*

Demonstração. Suponha que $\hat{f}_1$ e $\hat{f}_2$ são $\mathbb{H}$ -regulares, assim pelo Corolário 4.4.1 e o Lema 4.4.1 existem dois PAFHTD's $\mathcal{D}_1 = \langle Q, \Sigma, \delta_{\mathcal{D}_1}, q_0, \mathcal{F}_1 \rangle$ e $\mathcal{D}_2 = \langle P, \Sigma, \delta_{\mathcal{D}_2}, p_0, \mathcal{F}_2 \rangle$ tal que $\hat{f}_1 = \hat{f}_{\mathcal{D}_1}$ e $\hat{f}_2 = \hat{f}_{\mathcal{D}_2}$, sem perda de generalidade assuma que $Q \cap P = \emptyset$, agora defina um novo PAFHTD $\mathcal{D} = \langle Q \times P, \Sigma, \delta_{\mathcal{D}}, (q_0, p_0), \mathcal{F} \rangle$ onde,

$$\delta_{\mathcal{D}}((q, p), a) = (\delta_{\mathcal{D}_1}(q, a), \delta_{\mathcal{D}_2}(p, a)) \quad (4.35)$$

e

$$\mathcal{F}((q, p)) = \mathcal{F}_1(q) \otimes \mathcal{F}_2(p) \quad (4.36)$$

agora é claro pela Equação (4.35) e pela Definição de $\widehat{\delta_{\mathcal{D}}}$ que para $w \in \Sigma^*$ tem-se que,

$$\widehat{\delta_{\mathcal{D}}}((q_0, p_0), w) = (\widehat{\delta_{\mathcal{D}_1}}(q_0, w), \widehat{\delta_{\mathcal{D}_2}}(p_0, w)) \quad (4.37)$$

portanto, para qualquer $w \in \Sigma^*$ que,

$$\begin{aligned} \hat{f}_{\mathcal{M}}(w) &= \mathcal{F}(\widehat{\delta_{\mathcal{M}}}((q_0, p_0), w)) \\ &\stackrel{\text{Eq. (4.37)}}{=} \mathcal{F}((\widehat{\delta_{\mathcal{D}_1}}(q_0, w), \widehat{\delta_{\mathcal{D}_2}}(p_0, w))) \\ &\stackrel{\text{Eq. (4.36)}}{=} \mathcal{F}_1(\widehat{\delta_{\mathcal{D}_1}}(q_0, w)) \otimes \mathcal{F}_2(\widehat{\delta_{\mathcal{D}_2}}(p_0, w)) \\ &= \hat{f}_{\mathcal{D}_1}(w) \otimes \hat{f}_{\mathcal{D}_2}(w) \\ &= \hat{f}_1 \nabla \hat{f}_2 \end{aligned}$$

o que completa a prova. $\square$

Como mencionado na Seção 4.1, a inf-combinação é um combinador do tipo passa baixa, assim a $\mathbb{H}$ -interseção corresponde a uma filtragem realizada entre os valores de pertinência de duas LFHT's, isto é, o autômato que computa a $\mathbb{H}$ -interseção se comporta como um mecanismo de decisão, que seleciona os valores mais baixos de pertinência. Além disso, como consequência do resultado apresentado no Teorema 4.4.5 são obtidas as seguintes conclusões.

Corolário 4.4.3. *Se $\hat{f}_1, \hat{f}_2 \in \mathbb{T}_R$, então $\hat{f}_1 \nabla \hat{f}_2 \in \mathbb{T}_R$.*

Demonstração. Direto do Teorema 4.4.5 e do Corolário 4.4.1. $\square$

Corolário 4.4.4. *Se $\{\hat{f}_i\}_{i \in I}$ é uma família finita de LFHT's que são $\mathbb{H}$ -regulares, então existe um PAFHTD $\mathcal{M}$ tal que $\hat{f}_{\mathcal{M}} = \nabla_{i \in I} \hat{f}_i$*

Demonstração. Por indução sobre o conjunto de índices I e aplicando o Teorema 4.4.5. $\square$

4.5 Considerações finais

Inicialmente neste capítulo foram construídos dois novos operadores sobre o conjunto $\mathbb{H}$, tais operadores foram nomeados como inf-combinação e sup-combinação, em seguida, mostrou-se que tais operadores “fecham” o conjunto $\mathbb{H}$ numa estrutura algébrica de semi-anel. Além disso, a partir da sup-combinação foi introduzida uma nova relação de ordem parcial sobre o conjunto $\mathbb{H}$, denotada por $\sqsubseteq$, que generaliza a ordem usual $\leq$ dos números reais. Vale ressaltar que a ordem $\sqsubseteq$ pode ser facilmente estendida para CFHT’s, assim este capítulo apresenta uma contribuição sólida para a teoria dos CFHT’s ao introduzir uma ordem parcial nos parâmetros descritos por (GARMENDIA; CAMPO; RECASENS, 2017; SANTOS et al., 2015), ou seja, uma ordem sobre CFHT’s que estenda a ordem usual dos reais.

Outro ponto importante deste capítulo foi a formalização dos autômatos *fuzzy* hesitantes típicos e dos pseudo autômatos *fuzzy* hesitantes típicos, além das suas respectivas versões determinísticas. A definição de tais modelos de autômatos segue os conceitos de autômatos *fuzzy* como apresentados inicialmente por Wee (WEE, 1967) e depois melhor formalizados por Qiu (QIU, 2001), ou seja, os autômatos aqui são máquinas valoradas sobre uma estrutura algébrica, ou como na visão de Zadeh (ZADEH, 1996), são algoritmos valorados, aqui tal estrutura de valoração corresponde ao semi-anel $\mathbb{H}$.

A teoria dos (pseudo) autômatos *fuzzy* hesitantes típicos apresentada aqui, é a formalização de um modelo computacional que, além de, possibilitar a representação das incertezas ligadas ao processo da computação (ZADEH, 1996; WU; QIU; XING, 2002; SANTOS, 1976), também é capaz de modelar a hesitação de se definir os valores de pertinência das transições entre estados e dos estados serem estados finais. Um importante resultado demonstrado aqui é o teorema da separação, isto é, o Teorema 4.2.3, que além de mostrar que, um único autômato *fuzzy* hesitante típico é equivalente a uma família finita de autômatos *fuzzy*, também oferecer um significado semântico/operacional para os autômatos *fuzzy* hesitantes típicos, como sendo computadores que executam uma quantidade contável de algoritmos *fuzzy* em paralelo. Além disso, foi mostrado que a versão determinística dos (pseudo) autômatos *fuzzy* hesitantes típicos tem o mesmo poder computacional da versão padrão.

Ainda neste capítulo foi caracterizada a família de linguagens aceitas (ou computadas) pelos (pseudo) autômatos *fuzzy* hesitantes típicos, que foram chamadas de linguagens *fuzzy* hesitantes típicas $\mathbb{H}$ -regulares, ligada a isso foi mostrado pelo Corolário 4.3.2, que existem linguagens *fuzzy* hesitantes típicas, isto é, conjuntos *fuzzy* hesitantes típicos sobre alfabetos, que não são computáveis por (pseudo) autômatos *fuzzy* hesitantes típicos.

Para fortalecer a teoria das linguagens *fuzzy* hesitantes típicas $\mathbb{H}$ -regulares, neste capítulo, foram definidas as operações de $\mathbb{H}$ -união e $\mathbb{H}$ -interseção e mostradas que elas

são operadores de fecho sobre a família de linguagens *fuzzy* hesitantes típicas $\mathbb{H}$ -regulares. Por fim, mas não menos importante, foi mostrado pelo Corolário 4.4.2, que para qualquer linguagem *fuzzy* hesitante típica que é $\mathbb{H}$ -regular existe um pseudo autômato *fuzzy* hesitante típico determinístico irredutível (ou mínimo) que reconhece a linguagem.

5 Ruído tipo sal e Pseudo Autômatos *Fuzzy* Hesitantes Típicos

Inspirado nas aplicações dos autômatos *fuzzy* descritas na Seção 3.4 do Capítulo 3, neste capítulo será apresentada uma aplicação para os pseudo autômatos *fuzzy* hesitantes típicos (PAFHT's) na área de processamento digital de imagens. De fato, o uso de autômatos valorados em processamento digital de imagens não é algo recente, Culik e seus colaboradores em (CULIK; KARI, 1993; CULIK; FRIŠ, 1995; CULIK; VALENTA, 1997), realizaram estudos sobre a aplicação de autômatos valorados para a compressão de imagens, e mais recentemente, novos estudos tem sido feitos nesta área (HAFNER, 1996; JIANG; LITOW; VEL, 2000; KATRITZKE; MERZENICH; THOMAS, 2001).

Aqui será exposto o uso dos pseudo autômatos *fuzzy* hesitantes típicos determinísticos (PAFHTD's) na tarefa de tratamento de ruídos de tipo sal em imagens. Será apresentado um método para o tratamento de ruídos que utiliza as definições dos autômatos apresentados no capítulo anterior, antes de apresentar o método e seus algoritmos em si é preciso chamar atenção para alguns pontos.

Observação 5.0.1. *Aqui o alfabeto Σ sempre irá representar o conjunto finito $\{a, b, c, d\}$ para todos os autômatos neste capítulo, Σ^n corresponderá ao conjuntos de todas as palavras sobre Σ com tamanho exatamente igual a n . Se $w = x_1 \cdots x_{n-1}x_n$ é uma palavra, então para todo $1 \leq i \leq n$ a palavra $w_i = x_1 \cdots x_i$ é chamada prefixo de w , o conjunto de todos os prefixos de uma palavra w será denotado por Pre_w , diferente da teoria clássica de autômatos aqui a palavra vazia não será prefixo de nenhuma palavra, ou seja, $\lambda \notin Pre_w$.*

Definição 5.0.1. *Para cada $n > 0$ a função $\mathcal{E}_n : \Sigma^n \rightarrow \mathbb{Z}_+^*$ é uma enumeração, ou seja, uma bijeção do conjunto Σ^n para $\mathbb{Z}_+^*$, respeitando a ordem lexicográfica, por exemplo para $n = 4$ tem-se:*

$$\begin{aligned} \mathcal{E}_4(aaaa) &= 1 \\ \mathcal{E}_4(aaab) &= 2 \\ \mathcal{E}_4(aaac) &= 3 \\ &\vdots \quad \quad \quad \vdots \quad \quad \quad \vdots \\ \mathcal{E}_4(dddd) &= 256 \end{aligned}$$

Todas as demais notações, nomenclaturas e formalismos referente ao processamento digital de imagens que serão usadas neste capítulo, podem ser encontradas no Apêndice B deste documento, para leitores não familiarizados com processamento digital de imagens sempre será indicado onde buscar no Apêndice B o conceito necessário.

5.1 Imagens e PAFHTD's

Esta seção descreve o algoritmo responsável por construir PAFHTD's a partir de imagens. Vale destacar que, as imagens que serão usadas são todas imagens quadradas (para detalhes ver a Observação B.0.1), além disso, as dimensões de todas as imagens usadas neste trabalho sempre serão potências de 2, isto é, as imagens aqui consideradas serão sempre elementos do conjunto $Mat([0, 1], 2^n, 2^n)$ para algum $n \in \mathbb{N}$ tal que $n > 0$, para detalhes sobre a definição deste conjunto ver a Definição B.0.2.

Observação 5.1.1. *Como exposto pelo Teorema B.0.1 toda imagem quadrada $\mathcal{I}$ de dimensão 2^n pode ser dividida em quatro quadrantes disjuntos de dimensão 2^{n-1} , isto é, cada quadrante tem que sua dimensão corresponde a metade do valor da dimensão¹ da imagem a qual o quadrante pertence, além disso, os quadrantes não compartilham pixels em comum. Aqui os quadrantes serão indexados pelas letras do alfabeto Σ no sentido da esquerda para à direita e de cima para baixo, a Figura 19 a seguir ilustra este conceito.*

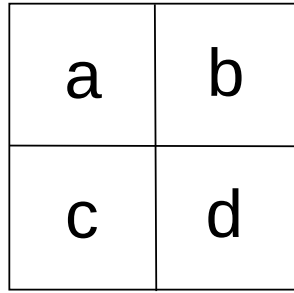


Figura 19 – Imagem dividida em quatro quadrantes indexados pelos elementos de Σ .

Dado uma imagem $\mathcal{I}$ neste texto será considerado que o conjunto $\theta_{\mathcal{I}}$ de todos os quadrantes de $\mathcal{I}$ será visto como sendo um SPO, como respeito à ordem $\lll$ descrita na Definição B.0.6. A estratégia introduzida neste trabalho para a construção de um PAFHTD $\mathcal{M}$ a partir de uma imagem de entrada $\mathcal{I}$, é descrita de forma informal pelo pseudocódigo do Algoritmo 1 a seguir.

Algoritmo 1 Algoritmo para construção de um PAFHTD.

Entrada Uma imagem quadrada $\mathcal{I}$ de dimensão 2^n .

Saída Um PAFHTD $\mathcal{M}$.

- 1: Execute o Algoritmo 2 e obtenha um AFD $A = \langle Q, \Sigma, \delta, q_0, F \rangle$ e $\xi : Q \rightarrow \theta_{\mathcal{I}}$.
 - 2: Execute o Algoritmo 3 e obtenha o CFHT $\mathcal{F} : Q \rightarrow \mathbb{H}$.
 - 3: Defina $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, \mathcal{F} \rangle$.
 - 4: **retorne** $\mathcal{M}$.
-

O que o Algoritmo 1 faz é obter um AFD a partir da imagem de entrada, depois ele cria um CFHT $\mathcal{F}$ de estados finais, então o conjunto de estados finais F do AFD é

¹ A dimensão de um quadrante $\mathcal{Q}$ é denotado por $\dim(\mathcal{Q})$ para detalhes ver o Apêndice B.

substituído pelo CFHT $\mathcal{F}$, o que transforma o AFD em um PAFHTD. Para continuar, agora serão analisados alguns resultados e também os detalhes do funcionamento do Algoritmo 2, responsável por obter o AFD.

Algoritmo 2 Algoritmo de construção do AFD.

Entrada Uma imagem quadrada $\mathcal{I}$ de dimensão 2^n .

Saída Um AFD $A = \langle Q, X, \delta, q_0, \emptyset \rangle$ e uma função $\xi : Q \rightarrow \theta_{\mathcal{I}}$.

```

1:  $Q \leftarrow \{q_0\}$ 
2:  $\xi(q_0) \leftarrow \mathcal{I}$ 
3:  $s \leftarrow 0$ 
4: enquanto verdadeiro faça
5:    $T \leftarrow \xi(q_s)$ 
6:    $d \leftarrow \dim(T)$ 
7:    $r \leftarrow \frac{d}{2}$ 
8:   se  $r < 1$  então
9:     Interrompa o laço do enquanto
10:  fim se
11:  Obtenha os 4 quadrantes disjuntos  $\mathcal{Q}_T^a, \mathcal{Q}_T^b, \mathcal{Q}_T^c$  e  $\mathcal{Q}_T^d$  de  $T$ .
12:   $s_1 \leftarrow \#Q + 1, s_2 \leftarrow \#Q + 2, s_3 \leftarrow \#Q + 3, s_4 \leftarrow \#Q + 4$ 
13:   $Q \leftarrow Q \cup \{q_{s_1}, q_{s_2}, q_{s_3}, q_{s_4}\}$ 
14:  Defina  $\xi(q_{s_1}) = \mathcal{Q}_T^a, \xi(q_{s_2}) = \mathcal{Q}_T^b, \xi(q_{s_3}) = \mathcal{Q}_T^c$  e  $\xi(q_{s_4}) = \mathcal{Q}_T^d$ 
15:  Defina  $\delta(q_s, a) = q_{s_1}, \delta(q_s, b) = q_{s_2}, \delta(q_s, c) = q_{s_3}$  e  $\delta(q_s, d) = q_{s_4}$ 
16:   $s \leftarrow s + 1$ 
17: fim enquanto
18: Defina  $A = \langle Q, \{a, b, c, d\}, \delta, q_0, \emptyset \rangle$ 
19: retorne  $A$  e  $\xi$ 
```

Observação 5.1.2. *Pela Observação 5.1.1, o passo 11 do Algoritmo 2 garante que a cada volta do laço (instruções 4 até 17) a dimensão dos quadrantes gerados sempre decai em razão de 2, isto é, a cada interação tem-se que, $\dim(\mathcal{Q}_T^x) = \frac{\dim(T)}{2}$ para todo T e $x \in \Sigma$, ressaltando que $\mathcal{Q}_T^x \lll T$, além disso, é claro que ξ é uma bijeção.*

A ideia do Algoritmo 2 pode ser descrita da seguinte forma, dado uma imagem de entrada $\mathcal{I}$ no primeiro passo é criado o estado q_0 , em seguida, é definida a função ξ para o estado q_0 de forma que, q_0 mapeia a imagem de entrada $\mathcal{I}$. Após esses dois primeiros comandos o algoritmo entra em um laço que só iria se encerrar após todos os quadrantes disjuntos com dimensões maiores ou iguais que 1 estiverem sendo a imagem da função ξ para algum estado do AFD que está sendo gerado.

Os passos do laço (instruções 4 até 17) do Algoritmo 2 podem ser descritos iterativamente da seguinte forma, para cada quadrante T da imagem de entrada, T é quebrado em 4 quadrantes disjuntos, em seguida, para cada um desses novos quadrantes $\mathcal{Q}'$ é criado um novo estado q' tal que $\xi(q') = \mathcal{Q}'$, depois dependendo da indexação do quadrante descrita na Observação 5.1.1 é criada uma transição da forma, $\delta(q, x) = q'$, tal que $\xi(q) = T$ e $\xi(q') \lll \xi(q)$.

Observação 5.1.3. Para toda imagem de entrada $\mathcal{I}$ tem-se que $\xi(q_0) = \mathcal{I}$.

O Exemplo 5.1.1 a seguir ilustra o funcionamento do Algoritmo 2.

Exemplo 5.1.1. Considere a imagem representada pela seguinte matriz,

$$\mathcal{I} = \begin{bmatrix} 0.1 & 0.0 & 0.0 & 0.2 \\ 0.3 & 0.0 & 0.3 & 0.0 \\ 0.2 & 0.1 & 0.0 & 0.0 \\ 0.3 & 0.2 & 0.1 & 0.1 \end{bmatrix}$$

inicialmente o Algoritmo 2 cria um estado q_0 e faz,

$$\xi(q_0) = \begin{bmatrix} 0.1 & 0.0 & 0.0 & 0.2 \\ 0.3 & 0.0 & 0.3 & 0.0 \\ 0.2 & 0.1 & 0.0 & 0.0 \\ 0.3 & 0.2 & 0.1 & 0.1 \end{bmatrix}$$

em seguida, o laço (instruções 4 até 17) do Algoritmo 2 cria os estados q_1, q_2, q_3, q_4 e define,

$$\xi(q_1) = \begin{bmatrix} 0.1 & 0.0 \\ 0.3 & 0.0 \end{bmatrix}, \quad \xi(q_2) = \begin{bmatrix} 0.0 & 0.2 \\ 0.3 & 0.0 \end{bmatrix}, \quad \xi(q_3) = \begin{bmatrix} 0.2 & 0.1 \\ 0.3 & 0.2 \end{bmatrix}, \quad \xi(q_4) = \begin{bmatrix} 0.0 & 0.0 \\ 0.1 & 0.1 \end{bmatrix}$$

além disso, são definidas as transições,

$$\delta(q_0, a) = q_1, \quad \delta(q_0, b) = q_2, \quad \delta(q_0, c) = q_3, \quad \delta(q_0, d) = q_4$$

ao continuar com o laço tem-se que,

$$\begin{aligned} \xi(q_5) &= [0.1], & \xi(q_6) &= [0.0], \\ \xi(q_7) &= [0.3], & \xi(q_8) &= [0.0], \\ \xi(q_9) &= [0.0], & \xi(q_{10}) &= [0.2], \\ \xi(q_{11}) &= [0.3], & \xi(q_{12}) &= [0.0], \\ \xi(q_{13}) &= [0.2], & \xi(q_{14}) &= [0.1], \\ \xi(q_{15}) &= [0.3], & \xi(q_{16}) &= [0.2], \\ \xi(q_{17}) &= [0.0], & \xi(q_{18}) &= [0.0], \\ \xi(q_{19}) &= [0.1], & \xi(q_{20}) &= [0.1] \end{aligned}$$

não é difícil verificar que o passo 15 do Algoritmo 2 garante a existência das seguintes transições,

$$\begin{aligned} \delta(q_1, a) &= q_5, & \delta(q_1, b) &= q_6, & \delta(q_1, c) &= q_7, & \delta(q_1, d) &= q_8 \\ \delta(q_2, a) &= q_9, & \delta(q_2, b) &= q_{10}, & \delta(q_2, c) &= q_{11}, & \delta(q_2, d) &= q_{12} \\ \delta(q_3, a) &= q_{13}, & \delta(q_3, b) &= q_{14}, & \delta(q_3, c) &= q_{15}, & \delta(q_3, d) &= q_{16} \\ \delta(q_4, a) &= q_{17}, & \delta(q_4, b) &= q_{18}, & \delta(q_4, c) &= q_{19}, & \delta(q_4, d) &= q_{20} \end{aligned}$$

visualmente o AFD gerado pode ser visto na Figura 20 a seguir.

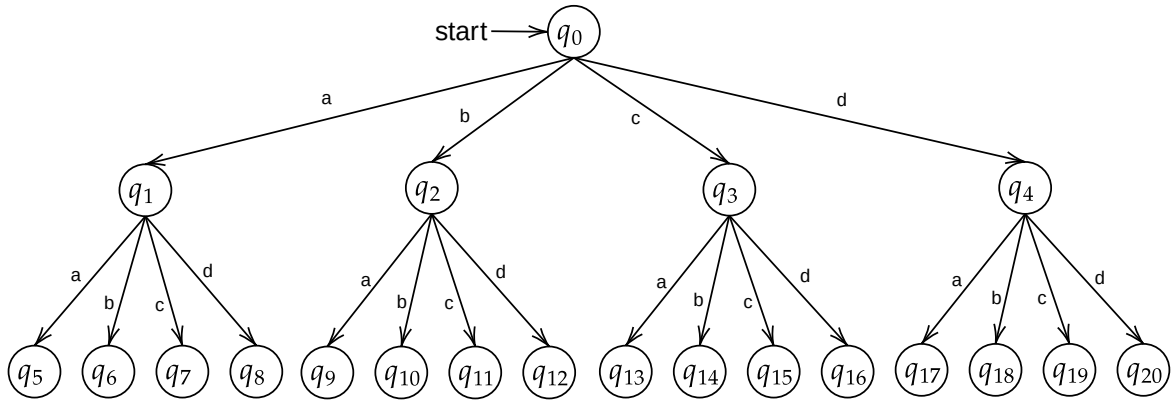


Figura 20 – AFD gerado aplicando o Algoritmo 2 sobre a imagem $\mathcal{I}$ do Exemplo 5.1.1.

O resultado a seguir mostra que dada uma imagem quadrada $\mathcal{I}$, cuja dimensão da mesma é uma potência de 2, todo quadrante desta imagem está associado com uma palavra sobre o alfabeto Σ .

Teorema 5.1.1. *Se $\mathcal{I}$ é uma imagem quadrada com $\dim(\mathcal{I}) = 2^n$, então para todo quadrante $\mathcal{Q} \in \theta_{\mathcal{I}}$ existe uma palavra w tal que $\xi(\hat{\delta}(q_0, w)) = \mathcal{Q}$.*

Demonstração. Dado uma imagem $\mathcal{I}$ quadrada com dimensão 2^n , pelo Algoritmo 2 é criado um AFD $A = \langle Q, \{a, b, c, d\}, \delta, q_0, \emptyset \rangle$ e uma função ξ , agora usando indução sobre a dimensão dos quadrantes (lembrando que a dimensão dos quadrantes decai em razão de 2), o caso base da indução corresponde a:

$$\begin{aligned} \xi(\hat{\delta}(q_0, \lambda)) &= \xi(q_0) \\ &\stackrel{\text{Obs. 5.1.3}}{=} \mathcal{I} \end{aligned}$$

agora para o caso geral da indução assuma que o resultado é verdadeiro para todos os quadrantes de dimensão r , pela Observação 5.1.2 para qualquer quadrante $\mathcal{Q}'$ com $\dim(\mathcal{Q}') = \frac{r}{2}$ existe um quadrante $\mathcal{Q}$ com $\dim(\mathcal{Q}) = r$, estados $q, q' \in Q$ e um símbolo $x \in \Sigma$ tal que $\xi(q) = \mathcal{Q}$, $\xi(q') = \mathcal{Q}'$ e $\delta(q, x) = q'$. Mas pela hipótese indutiva existe w tal que $\xi(\hat{\delta}(q_0, w)) = \mathcal{Q}$, logo $\hat{\delta}(q_0, w) = q$, e portanto, existe wx tal que,

$$\xi(\hat{\delta}(q_0, wx)) = \xi(\delta(\hat{\delta}(q_0, w), x)) \stackrel{\text{Hip.}}{=} \xi(\delta(q, x)) \stackrel{\text{Hip.}}{=} \xi(q') = \mathcal{Q}'$$

o que completa a prova. □

O exemplo a seguir ilustra o mapeamento descrito pelo Teorema 5.1.1.

Exemplo 5.1.2. Dado os AFD A e a função ξ gerados no Exemplo 5.1.1, tem-se que “aa” e “ab” são respectivamente as palavras que acessam os quadrantes $\xi(q_5)$ e $\xi(q_6)$, pois:

$$\begin{aligned}\xi(\widehat{\delta}(q_0, aa)) &= \xi(\delta(\widehat{\delta}(q_0, a), a)) \\ &= \xi(\delta(\delta(\widehat{\delta}(q_0, \lambda), a), a)) \\ &= \xi(\delta(\delta(q_0, a), a)) \\ &= \xi(\delta(q_1, a)) \\ &= \xi(q_5)\end{aligned}$$

e

$$\begin{aligned}\xi(\widehat{\delta}(q_0, ab)) &= \xi(\delta(\widehat{\delta}(q_0, a), b)) \\ &= \xi(\delta(\delta(\widehat{\delta}(q_0, \lambda), a), b)) \\ &= \xi(\delta(\delta(q_0, a), b)) \\ &= \xi(\delta(q_1, b)) \\ &= \xi(q_6)\end{aligned}$$

Como consequência do resultado apresentado pelo Teorema 5.1.1 é obtida a seguinte conclusão.

Corolário 5.1.1. Se $\mathcal{Q}' \lll \mathcal{Q}$ e $\dim(\mathcal{Q}') = \frac{\dim(\mathcal{Q})}{2}$, então $|w'| = |w| + 1$ tal que $\xi(\widehat{\delta}(q_0, w')) = \mathcal{Q}'$ e $\xi(\widehat{\delta}(q_0, w)) = \mathcal{Q}$ com $w, w' \in \Sigma^*$.

Demonstração. Suponha que $\mathcal{Q}' \lll \mathcal{Q}$ e $\dim(\mathcal{Q}') = \frac{\dim(\mathcal{Q})}{2}$, pelo Teorema 5.1.1 tem-se que existe um w tal que $\xi(\widehat{\delta}(q_0, w)) = \mathcal{Q}$, e pelo Algoritmo 2 existe um q' e um símbolo $x \in \Sigma$ tal que $\delta(\widehat{\delta}(q_0, w), x) = q'$ e $\xi(q') = \mathcal{Q}'$, portanto, $\xi(\widehat{\delta}(q_0, wx)) = \mathcal{Q}'$, consequentemente, $w' = wx$ e é claro que $|wx| = |w| + 1$. $\square$

Os resultados apresentados pelo Lema 5.1.1 e o Teorema 5.1.2 a seguir mostram que a dimensão de um quadrante e o tamanho da palavra que o representa (ou acessa) estão relacionados.

Lema 5.1.1. Seja $\mathcal{I}$ uma imagem quadrada de dimensão 2^n . Se $\dim(\mathcal{Q}) = r$, então $r = 2^{n-|w|}$ e $\xi(\widehat{\delta}(q_0, w)) = \mathcal{Q}$ tal que $w \in \Sigma^*$.

Demonstração. A prova é feita por indução sobre a dimesão dos quadrantes, dado uma imagem $\mathcal{I}$ quadrada com dimensão 2^n , aplicando o Algoritmo 2 é obtido um AFD $A = \langle Q, \{a, b, c, d\}, \delta, q_0, \emptyset \rangle$ e uma função ξ , agora o caso base é trivial pois,

$$\dim(\mathcal{I}) = 2^n$$

mas $2^n = 2^{n-|\lambda|}$ e $\xi(\widehat{\delta}(q_0, \lambda)) = \mathcal{I}$. Agora para o caso geral assuma o passo indutivo de que para $\mathcal{Q}$ com $\dim(\mathcal{Q}) = r$ as igualdades $r = 2^{n-|w|}$ e $\xi(\widehat{\delta}(q_0, w)) = \mathcal{Q}$ são válidas. Logo para qualquer $\mathcal{Q}'$ tal que $\mathcal{Q}' \lll \mathcal{Q}$ e $\dim(\mathcal{Q}') = \frac{r}{2}$ teremos que,

$$\begin{aligned} \dim(\mathcal{Q}') &= \frac{r}{2} \\ &= \frac{\dim(\mathcal{Q})}{2} \\ &\stackrel{Hip.}{=} \frac{2^{n-|w|}}{2} \\ &= 2^{n-(|w|+1)} \end{aligned}$$

mas pelo Corolário 5.1.1, $|w| + 1 = |w'|$ tal que $\xi(\widehat{\delta}(q_0, w')) = \mathcal{Q}'$, terminando a prova. $\square$

Teorema 5.1.2. *Seja $\mathcal{I}$ uma imagem quadrada de dimensão 2^n . Se $\xi(\widehat{\delta}(q_0, w)) = \mathcal{Q}$, então $w \in \Sigma^{-(\log_2(\dim(\mathcal{Q}))) + n}$.*

Demonstração. Pelo Lema 5.1.1 para qualquer $\mathcal{Q}$ tem-se que $\dim(\mathcal{Q}) = 2^{n-|w|}$ tal que $\xi(\widehat{\delta}(q_0, w)) = \mathcal{Q}$, mas,

$$\begin{aligned} \dim(\mathcal{Q}) = 2^{n-|w|} &\iff n - |w| = \log_2(\dim(\mathcal{Q})) \\ &\iff -|w| = (\log_2(\dim(\mathcal{Q}))) - n \\ &\iff |w| = -(\log_2(\dim(\mathcal{Q}))) + n \end{aligned}$$

portanto, $w \in \Sigma^{-(\log_2(\dim(\mathcal{Q}))) + n}$. $\square$

Corolário 5.1.2. *Se $\mathcal{I}$ é uma imagem quadrada de dimensão 2^n e A é o AFD gerado pelo Algoritmo 2, então o tamanho da maior palavra w para a qual A têm uma computação é igual n .*

Demonstração. Suponha que $\mathcal{I}$ é uma imagem quadrada de dimensão 2^n e A o AFD gerado pelo Algoritmo 2. Note que, pelo Algoritmo 2, A não possui ciclos, além disso, os últimos estados gerados, atendem a restrição que $\dim(\xi(q)) = 1$. Agora pela construção de A , existe w tal que $\widehat{\delta}(q_0, w) = q$ mas,

$$|w| \stackrel{Teo. 5.1.2}{=} -(\log_2(\dim(\xi(q)))) + n = -(\log_2(1)) + n = -0 + n = n$$

o que conclui a prova. $\square$

Agora será descrito o modelo adotado neste trabalho para a valoração dos estados de um AFD usando o espaço de valoração $\mathbb{H}$, ou seja, um mecanismo para transformar qualquer AFD em um PAFHTD. Note que essa tarefa nada mais é do que a construção do CFHT $\mathcal{F}$ da definição 4.4.3. Primeiro será apresentado o pseudocódigo que descreve o método usado neste trabalho, depois será feito uma análise descritiva do mesmo, afim de, fornecer um melhor entendimento do mesmo.

Algoritmo 3 Algoritmo de valoração dos estados finais *fuzzy* hesitantes típicos.

Entrada Um AFD $A = \langle Q, X, \delta, q_0, \emptyset \rangle$ e uma função $\xi : Q \rightarrow \theta_{\mathcal{I}}$ e uma família finita $\{g_j\}_{j \in J}$ de funções de agregação.

Saída Um CFHT $\mathcal{F}$.

```

1: Defina  $\mathcal{F}(q_0) = \{0, 1\}$ 
2:  $s \leftarrow 0$ 
3:  $M_s \leftarrow \xi(q_s)$ 
4: enquanto  $\dim(\xi(q_s)) > 1$  faça
5:   para cada  $x \in \Sigma$  faça
6:      $M_x \leftarrow \xi(\delta(q_s, x))$ 
7:     Defina  $\mathcal{F}(\delta(q_s, x)) = \bigcup_{j \in J} \{1 - |g_j(M_s) - g_j(M_x)|\}$ 
8:   fim para
9:    $s \leftarrow s + 1$ 
10:   $M_s \leftarrow \xi(q_s)$ 
11: fim enquanto
12: retorne  $\mathcal{F}$ 

```

Observação 5.1.4. Para o estudo experimental, que será detalhado a seguir, foi considerado que a família finita $\{g_j\}_{j \in J}$ de funções de agregação usadas no Algoritmo 3 é formada pelas agregações: mínimo, máximo, média aritmética e mediana. A escolha de mínimo e máximo se deu por estas serem respectivamente filtros passa baixa e passa alta, já muito usadas na área de processamento de imagens, já media e mediana foram escolhidas por serem filtros de tipo médio (GONZALEZ; WOODS, 2009), isto é, que tentam preservar a média do sinal de entrada.

Observação 5.1.5. Funções de agregação são funções que mapeiam n elementos de $[0, 1]$ em um único elemento $x \in [0, 1]$ para detalhes ver (FARIAS et al., 2016a).

A ideia do Algoritmo 3 é a seguinte, o processo de valorar qualquer $q \in Q$ como um estado final é visto como a valoração do predicado, “O quadrante $\xi(q)$ não apresenta ruído”. Obviamente não é possível determinar se o processo de captura que gerou a imagem de entrada $\mathcal{I}$ sofreu ou não algum tipo de interferência que gerou ruído, o que é simbolizado com a atribuição de $\mathcal{F}(q_0) = \{0, 1\}$. Depois de forma iterativa para cada estado q a valoração de seus sucessores, isto é, dos estados q' tal que $\delta(q, x) = q'$ para todo $x \in \Sigma$, é dada com sendo o resultado da união das similaridades² das agregações de $\xi(q)$ e $\xi(\delta(q, x))$, ou seja,

$$\mathcal{F}(q') = \bigcup_{j \in J} \{1 - |g_j(\xi(q)) - g_j(\xi(\delta(q, x)))|\}$$

tal que $x \in \Sigma$ e $\delta(q, x) = q'$. Assim, os valores que separam as similaridades entre as agregações de $\xi(q)$ e as agregações de $\xi(\delta(q, x))$ do valor 1, são compreendidos como sendo os níveis de ruído no quadrante $\xi(\delta(q, x))$, o Exemplo 5.1.3 a seguir, ilustra estes conceitos.

² A similaridade considerada aqui corresponde ao pseudo complemento do intervalo $[0, 1]$, muito utilizada em estudo sobre semânticas *fuzzy* (RUSPINI, 1991).

Exemplo 5.1.3. Considerando o AFD e a função ξ descritos no Exemplo 5.1.1, adotando as agregações mencionadas na Observação 5.1.4 tem-se que,

Quadrantes	Média aritmética	Mínimo	Máximo	Mediana
$\xi(q_0)$	0.11875	0.0	0.3	0.1
$\xi(q_1)$	0.1	0.0	0.3	0.0
$\xi(q_2)$	0.125	0.0	0.3	0.1
$\xi(q_3)$	0.2	0.1	0.3	0.2
$\xi(q_4)$	0.05	0.0	0.1	0.05

Tabela 1 – Agregações de cada quadrante $\xi(q)$ tal que $q \in \{q_0, q_1, q_2, q_3, q_4\}$.

Agora como para todo $i > 0$ tem-se que $\xi(q_i) \lll \xi(q_0)$, assim são obtidos através do passo 7 do Algoritmo 3 os seguintes valores de similaridades entre as agregações dos pares de quadrantes $(\xi(q_0), \xi(q_i))$,

Pares de quadrantes	Média aritmética	Mínimo	Máximo	Mediana
$(\xi(q_0), \xi(q_1))$	0.98125	1.0	1.0	0.9
$(\xi(q_0), \xi(q_2))$	0.99375	1.0	1.0	1.0
$(\xi(q_0), \xi(q_3))$	0.91875	0.9	1.0	0.9
$(\xi(q_0), \xi(q_4))$	0.93125	1.0	0.8	0.95

Tabela 2 – Resultados das similaridades entre as agregações de cada quadrante $\xi(q)$ tal que $q \in \{q_1, q_2, q_3, q_4\}$ e o quadrante $\xi(q_0)$.

E continuando o passo 7 do Algoritmo 3 tem-se que, as valorações dos estados q_1, q_2, q_3 e q_4 com respeito ao CFHT $\mathcal{F}$ são respectivamente,

$$\mathcal{F}(q_1) = \{0.98125, 1.0, 0.9\}$$

$$\mathcal{F}(q_2) = \{0.99375, 1.0\}$$

$$\mathcal{F}(q_3) = \{0.91875, 0.9, 1.0\}$$

$$\mathcal{F}(q_4) = \{0.93125, 1.0, 0.8, 0.95\}$$

Observação 5.1.6. Note que para qualquer $\hat{\delta}(q_0, w)$ tal que $\dim(\xi(\hat{\delta}(q_0, w))) = 1$ tem-se que, $\mathcal{F}(\hat{\delta}(q_0, w))$ pode-ser interpretado com sendo o conjunto dos valores de similaridade entre o pixel acessado por $\hat{\delta}(q_0, w)$ e o quadrante que contém tal pixel.

5.2 O Algoritmo ReSPA

Nesta seção será descrito o algoritmo de remoção (ou tratamento) de ruído³ baseado em PAFHTD's, que será nomeado como ReSPA. Neste trabalho será considerado um

³ Um ruído em uma imagem digital corresponde a uma variação aleatória (que não condiz com a realidade) de brilho, contraste e cor (em imagens coloridas) nas imagens digitais produzidas pelo dispositivo de captura (uma câmera digital por exemplo).

dos tipos clássicos de ruídos estudados no processamento de imagens em tons de cinza, chamado de ruído de **sal** (GONZALEZ; WOODS, 2009). O ruído do tipo sal é caracterizado como dito em (SOLOMON; BRECKON, 2011), por uma amplitude de valor máximo para um determinado pulso do sinal que não deveria ter tal valor, isto é, um *pixel* que apresenta ruído de sal é aquele que deveria possuir o valor $n < 1$, mas por algum erro de captura acaba recebendo o valor 1.

A construção do algoritmo ReSPA introduzido neste trabalho usa os demais algoritmos anteriormente apresentados, e algumas outras ideias descritas a seguir.

Observação 5.2.1. *Note que considerando uma imagem quadrada $\mathcal{I}$ de dimensão 2^n , a função ξ definida pelo Algoritmo 2 e o PAFHTD $\mathcal{M}$ definido pelo Algoritmo 1 com respeito à imagem $\mathcal{I}$ tem-se que,*

- (a) *Pelo Corolário 5.1.2 o tamanho da maior palavra computada pelo AFD A gerado pelo Algoritmo 2 dada a imagem $\mathcal{I}$ é exatamente igual a n .*
- (b) *Pelo Lema 5.1.1, para todo $|w| = n$ tem-se que $\dim(\xi(\widehat{\delta}(q_0, w))) = 1$, ou seja, para todos os w tal que $|w| = n$ tem-se que $\xi(\widehat{\delta}(q_0, w))$ são exatamente os endereços dos pixels da imagem.*
- (c) *Como a quantidade de pixels na imagem quadrada $\mathcal{I}$ é obviamente finita, então sempre pode-se considerar que existe uma enumeração $\mathcal{W} : \theta_{\mathcal{I}}^1 \rightarrow \{1, 2, \dots, 2^{2n}\}$, onde $\theta_{\mathcal{I}}^1 = \{Q \in \theta_{\mathcal{I}} \mid \dim(Q) = 1\}$.*
- (d) *Considerando a observação acima, sempre que for escrito $\mathcal{I}(k)$, estará sendo referido o k -ésimo pixel de $\mathcal{I}$ em relação à enumeração $\mathcal{W}$.*

Para prosseguir é necessário considerar as seguintes definições.

Definição 5.2.1. *Seja $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, \mathcal{F} \rangle$ um PAFHTD, a computação dos prefixos por $\mathcal{M}$ é uma função $CP_{\mathcal{M}} : \Sigma^+ \times \mathbb{N} \rightarrow \mathbb{H}$, definida como sendo,*

$$CP_{\mathcal{M}}(w, \alpha) = \begin{cases} \bigsqcup_{w' \in Pre_w^\alpha} \mathcal{F}(\widehat{\delta}(q_0, w')), & \text{se } \alpha \leq |w| \\ \{0\}, & \text{senão} \end{cases} \quad (5.1)$$

onde $Pre_w^\alpha = \{w' \in Pre_w \mid |w'| \geq |w| - \alpha\}$.

Definição 5.2.2. *Seja $X \in \mathbb{H}$ e $y \in [0, 1]$, a multiplicação por escalar de y por X , denotado por $y \star X$, é definido como,*

$$y \star X = y \cdot \text{Mean}(X) \quad (5.2)$$

onde $\text{Mean}(X)$ é a média aritmética de X .

Algoritmo 4 ReSPA.

Entrada Uma imagem quadrada $\mathcal{I}$ com dimensão 2^n para algum $n > 0$ e um natural α .

Saída Uma imagem quadrada de $\mathcal{I}'$.

- 1: Execute o Algoritmo 2 e obtenha um AFD $A = \langle Q, \Sigma, \delta, q_0, \emptyset \rangle$ e uma função $\xi : Q \rightarrow \theta_{\mathcal{I}}$
- 2: Execute o Algoritmo 3 e obtenha um CFHT $\mathcal{F}$
- 3: Defina o PAFHTD $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, \mathcal{F} \rangle$
- 4: Defina n como o tamanho da maior palavra computada por A (ver Corolário 5.1.2)
- 5: Defina a enumeração $\mathcal{E}_n$ (ver Definição 5.0.1)
- 6: $s \leftarrow 1$
- 7: Defina uma nova imagem $\mathcal{I}'$ quadrada de dimensão 2^n
- 8: **enquanto** $s \leq 2^{2^n}$ **faça**
- 9: $k \leftarrow \mathcal{W}(\xi(\delta(q_0, \mathcal{E}_n^{-1}(s))))$
- 10: $\mathcal{I}'(k) \leftarrow \mathcal{I}(k) \star CP_{\mathcal{M}}(\mathcal{E}_n^{-1}(s), \alpha)$
- 11: $s \leftarrow s + 1$
- 12: **fim enquanto**
- 13: **retorne** $\mathcal{I}'$

O pseudocódigo do Algoritmo 4 descreve como implementar o ReSPA. As ideias do algoritmo ReSPA consistem em usar um PAFHTD $\mathcal{M}$ (construído nas instruções de 1 a 3) como um mecanismo para eliminar ruídos do tipo sal nos *pixels*.

O procedimento de tratamento do ruído acontece exatamente no laço que compreende as instruções 8 até 12, a instrução 9 calcula qual é o índice do *pixel* que será tratado usando para isso a palavra que acessa o *pixel* aplicando a enumeração das palavras (ver Definição 5.0.1) e a função de transição do autômato, e a instrução 10 trata o ruído.

A ideia da instrução 10 é a seguinte, primeiro é calculada a computação dos prefixos (em relação a um certo α) da palavra que acessa o *pixel* k que esta sendo tratado, este α representa quantos quadrantes que contém o *pixel* k devem ser analisados para determinar se o *pixel* k possui ou não ruído, isto é, quantos estados anteriores ao estado que é mapeado no quadrante do *pixel* k , devem ser levados em consideração para afirmar se o *pixel* k possui ou não ruído.

Agora como foi descrito na seção anterior, sempre que não existir incerteza nem hesitação em afirmar que os n quadrantes imediatamente maiores pela ordem $\lll$ que contém o *pixel*, ou seja, $\xi(\mathcal{E}_n^{-1}(s)) \lll \mathcal{Q}_1 \lll \dots \lll \mathcal{Q}_n$, não possuírem ruído tem-se que $CP_{\mathcal{M}}(\mathcal{E}_n^{-1}(s), n) = \{1\}$, e assim é claro que a multiplicação por escalar (Definição 5.2.2) da instrução 10 corresponderá ao próprio valor do *pixel*. Isto é razoável, pois se $\mathcal{Q}_i$ contém o *pixel* em questão, e $\mathcal{Q}_i$ não possui ruído, obviamente o *pixel* não pode apresentar ruído.

Por outro lado, quando existir incerteza e possivelmente hesitação, isso implicará na possibilidade da existência de ruído, e assim o valor do *pixel* deverá ser tratado e possivelmente modificado, através da multiplicação por escalar, note que neste caso esse produto nunca será igual a 1, e assim os ruídos do tipo sal serão removidos.

Observação 5.2.2. *É claro que se $\alpha = 0$ tem-se que $CP_{\mathcal{M}}(\mathcal{E}_n^{-1}(s), \alpha) = \mathcal{F}(\hat{\delta}(q_0, \mathcal{E}_n^{-1}(s)))$, ou seja, nesse caso o Algoritmo ReSPA só irá considerar o próprio pixel no momento de determinar a existência ou não de ruído do tipo sal.*

5.3 Resultados Experimentais

Para avaliar o desempenho do algoritmo ReSPA foram executados cinco cenários de experimentos, cada cenário diz respeito à proporção de ruído introduzido de forma aleatória nas imagens utilizadas neste experimento. As proporções (porcentagem) de *pixels* com ruído do tipo sal adotadas são 0.05, 0.10, 0.15, 0.20 e 0.25.

Os experimentos têm por finalidade analisar o ReSPA contra os tradicionais (e bem conhecidos) métodos:

- Bilateral filter (TOMASI; MANDUCHI, 1998).
- Denoise Wavelet (CHANG; YU; VETTERLI, 2000; DONOHO; JOHNSTONE, 1994).
- Richardson-Lucy deconvolution (FISH et al., 1995; RICHARDSON, 1972; LUCY, 1974).

Para a realização dos experimentos foram usadas as implementações presentes no *Framework Scikit-image* (WALT et al., 2014) dos métodos mencionados, os algoritmos foram todos executados com seu parâmetros padrão, como descrito na documentação do *Scikit-image*. Por sua vez o método ReSPA apresentado aqui, foi executado com o parâmetro α variando de 0 até 2. Para os experimentos foram usadas seis imagens⁴ em formato BMP como dimensão de 512×512 *pixels*. Para realizar a análise, os parâmetros de comparação adotados foram as bem conhecidas métricas PSNR e RNMSE (ver Apêndice B).

Observação 5.3.1. *Os valores expressos nas Tabelas 3–7 foram todos truncados para duas casa decimais, como forma de melhorar a visualização e interpretação dos resultados, além disso, a melhor média em cada cenário experimental é destacada em negrito para auxiliar na visualização.*

Como se pode observar pelas Tabelas 3–7, o método ReSPA com $\alpha = 0$ é nitidamente o método mais eficiente pois, em ambas as métricas adotadas e em todas as imagens e em todos os cenários, tal método apresentou um desempenho significativamente superior aos demais métodos. Os leitores interessados em uma comparação visual do desempenho dos métodos utilizados, basta verificar as imagens produzidas pelos experimentos que estão exposta no Apêndice C.

Imagem	ReSPA $\alpha = 0$		ReSPA $\alpha = 1$		ReSPA $\alpha = 2$		Richardson-Lucy		Denoise Wavelet		Bilateral filter	
	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE
Img1	26.60	0.06	22.82	0.10	22.53	0.10	19.59	0.14	22.24	0.11	22.86	0.10
Img2	24.04	0.13	18.34	0.25	17.88	0.26	17.90	0.26	17.73	0.26	18.04	0.25
Img3	25.70	0.08	21.52	0.13	21.17	0.13	19.34	0.16	20.88	0.14	21.70	0.12
Img4	24.16	0.16	16.84	0.37	16.34	0.40	18.64	0.30	16.10	0.41	17.06	0.36
Img5	25.58	0.10	19.46	0.19	19.03	0.20	19.57	0.19	18.76	0.21	19.22	0.20
Img6	25.06	0.12	18.40	0.25	17.93	0.26	18.92	0.24	17.68	0.27	18.68	0.24
Média	25.19	0.11	19.56	0.21	19.14	0.22	18.99	0.21	18.89	0.23	19.59	0.21

Tabela 3 – Resultados do cenário com proporção de 0.05 de ruído do tipo sal nas imagens.

Imagem	ReSPA $\alpha = 0$		ReSPA $\alpha = 1$		ReSPA $\alpha = 2$		Richardson-Lucy		Denoise Wavelet		Bilateral filter	
	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE
Img1	24.31	0.08	20.16	0.14	19.69	0.14	18.90	0.16	19.30	0.15	21.01	0.12
Img2	21.25	0.18	15.60	0.34	14.99	0.36	16.40	0.31	14.86	0.37	16.02	0.32
Img3	23.20	0.10	18.71	0.18	18.18	0.19	18.36	0.18	17.91	0.19	19.71	0.16
Img4	20.71	0.24	13.97	0.52	13.31	0.56	16.25	0.40	13.05	0.58	14.65	0.48
Img5	22.58	0.14	16.75	0.27	16.17	0.28	17.94	0.23	15.87	0.29	17.36	0.25
Img6	21.67	0.17	15.51	0.35	14.89	0.38	17.00	0.29	14.61	0.39	16.29	0.32
Média	22.28	0.15	16.78	0.30	16.20	0.31	17.47	0.26	15.93	0.32	17.50	0.27

Tabela 4 – Resultados do cenário com proporção de 0.10 de ruído do tipo sal nas imagens.

Imagem	ReSPA $\alpha = 0$		ReSPA $\alpha = 1$		ReSPA $\alpha = 2$		Richardson-Lucy		Denoise Wavelet		Bilateral filter	
	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE
Img1	22.50	0.10	18.46	0.16	17.86	0.18	18.09	0.17	17.60	0.18	19.29	0.15
Img2	19.31	0.22	14.01	0.40	13.26	0.44	15.15	0.36	13.30	0.44	14.56	0.38
Img3	21.49	0.13	17.19	0.21	16.53	0.23	17.43	0.20	16.71	0.22	18.24	0.19
Img4	18.53	0.31	12.37	0.63	11.59	0.68	14.53	0.49	11.36	0.70	13.14	0.57
Img5	20.50	0.17	15.13	0.32	14.41	0.35	16.55	0.27	14.15	0.36	15.93	0.29
Img6	19.58	0.22	13.93	0.42	13.18	0.46	15.50	0.35	12.96	0.47	14.80	0.38
Média	18.80	0.19	15.18	0.35	14.47	0.39	16.20	0.30	14.34	0.39	15.99	0.32

Tabela 5 – Resultados do cenário com proporção de 0.15 de ruído do tipo sal nas imagens.

Imagem	ReSPA $\alpha = 0$		ReSPA $\alpha = 1$		ReSPA $\alpha = 2$		Richardson-Lucy		Denoise Wavelet		Bilateral filter	
	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE
Img1	21.34	0.12	17.45	0.19	16.78	0.20	17.48	0.18	18.08	0.17	18.12	0.17
Img2	17.81	0.26	12.89	0.46	12.06	0.51	14.04	0.40	12.53	0.48	13.36	0.44
Img3	20.14	0.15	16.09	0.24	15.35	0.26	16.66	0.22	16.60	0.22	16.95	0.22
Img4	16.80	0.38	11.32	0.71	10.40	0.79	13.20	0.57	10.90	0.74	12.01	0.65
Img5	18.81	0.21	14.00	0.37	13.14	0.40	15.33	0.31	13.51	0.39	14.69	0.34
Img6	17.93	0.26	12.84	0.48	11.94	0.53	14.26	0.40	11.94	0.53	13.54	0.44
Média	18.80	0.23	14.09	0.40	13.27	0.44	15.16	0.34	13.92	0.42	14.77	0.37

Tabela 6 – Resultados do cenário com proporção de 0.20 de ruído do tipo sal nas imagens.

Imagem	ReSPA $\alpha = 0$		ReSPA $\alpha = 1$		ReSPA $\alpha = 2$		Richardson-Lucy		Denoise Wavelet		Bilateral filter	
	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE	PSNR	RNMSE
Img1	20.29	0.13	16.66	0.20	15.93	0.22	16.92	0.20	17.49	0.18	17.16	0.19
Img2	16.46	0.31	12.00	0.51	11.12	0.56	13.07	0.45	12.99	0.46	12.38	0.49
Img3	18.93	0.17	15.22	0.26	14.41	0.29	15.82	0.25	16.67	0.22	15.80	0.25
Img4	15.41	0.44	10.43	0.78	9.44	0.88	12.01	0.65	14.29	0.50	10.97	0.73
Img5	17.55	0.24	13.12	0.40	12.20	0.45	14.31	0.35	15.65	0.30	13.64	0.38
Img6	16.59	0.31	11.98	0.53	11.00	0.59	13.20	0.46	13.00	0.47	12.51	0.49
Média	25.12	0.26	13.23	0.44	12.35	0.49	14.22	0.39	15.18	0.35	13.74	0.42

Tabela 7 – Resultados do cenário com proporção de 0.25 de ruído do tipo sal nas imagens.

⁴ Todas as imagens utilizadas encontra-se disponíveis no repositório <<https://github.com/grupo-alicia/ALiCIA-Resources>>.

5.4 Considerações Finais

Neste capítulo foi apresentado um algoritmo para tratamento (ou remoção) de ruído do tipo sal para imagens em escala de tons de cinza, esse algoritmo usa os pseudo autômatos *fuzzy* hesitantes típicos determinísticos (introduzidos no Capítulo 4) como um mecanismo de tratamento (ou remoção) de ruído. Para a construção deste algoritmo de tratamento são usados outros três algoritmos, que também foram propostos neste capítulo.

O Algoritmo 2 é o que apresenta as características mais desenvolvidas, ele é responsável por construir um AFD a partir de uma imagem de entrada e também de definir uma função ξ que mapeia os quadrantes da imagem de entrada. Este algoritmo foi o mais extensamente estudado, foram expostos diversos resultados entre o AFD por ele gerado e a imagem de entrada.

Por sua vez, o Algoritmo 3 é responsável pela definição do conjunto de estados finais *fuzzy* hesitantes típicos, ele recebe como entrada o AFD gerado pelo Algoritmo 2 e devolve como saída um CFHT $\mathcal{F} : Q \rightarrow \mathbb{H}$. Por fim, o Algoritmo 1 combina os Algoritmos 2 e 3 e constrói um PAFHTD.

O capítulo prossegue com a definição do algoritmo para tratamento (ou remoção) de ruído do tipo sal anteriormente mencionado, este algoritmo recebeu o nome de ReSPA, o mesmo recebe com entrada uma imagem a ser tratada e um número natural α . Em relação ao desempenho, tal algoritmo se mostra superior a todos os outros métodos de tratamento de ruído contra os quais foi testado, chamando atenção que em alguns dos cenários experimentais, a vantagem do ReSPA com $\alpha = 0$ sobre os demais métodos é significativamente grande chegando as diferenças de mais de 10 dB (sobre a métrica PSNR) e a mais de 16% (sobre a métrica RNMSE), como apontado pelos resultados expressos nas Tabelas 3–7.

6 Classificação de Dados e Pseudo Autômatos *Fuzzy* Hesitantes Típicos

Como mencionado no Capítulo 3, a própria origem dos autômatos *fuzzy* está ligada com a tarefa de classificação de dados (ou reconhecimento de padrões), e uma vez que, os PAFHTD são extensões dos autômatos *fuzzy* é natural que também seja possível aplicar tais autômatos em tarefas de classificação de dados.

Neste capítulo será apresentado um novo algoritmo de classificação de dados, que será chamado de HeCA, tal algoritmo é baseado em PAFHTD e nas ideias de inferência gramatical (HIGUERA, 2010). Antes de apresentar o HeCA é necessário mostrar como converter uma base de exemplos em um PAFHTD.

6.1 Transformando a Base de Exemplos em um PAFHTD

Aqui será considerada a definição formal de base (ou conjunto) de dados da mesma forma que na área de aprendizado de máquina (HAYKIN, 2007).

Definição 6.1.1. (FACELI et al., 2011) Uma base de dados X_d^n é uma matriz de dimensão $n \times d$ em que, n é o número de elementos na base de dados e d é o número de atributos da base. x_j^i denotada o valor do j -ésimo atributo para o i -ésimo elemento na base. Assim o valor x_d^i expressa o rótulo da classe que o i -ésimo elemento pertence, e os demais atributos x_j^i com $1 \leq j < d$ são as características que descrevem o elemento.

Exemplo 6.1.1. A matriz a seguir é um exemplo de uma base de dados da condição de saúde de um grupo de pessoas, a primeira linha nomeia cada atributo, as demais são os elementos da base, isto é, formam a base em si.

idade do paciente	sexo	febril	manchas no corpo	diagnóstico
10	m	n	s	doente
25	f	s	n	doente
10	m	n	s	curado
41	m	s	s	doente

Tabela 8 – Base de dados baseada da Tabela 2.1 de (FACELI et al., 2011).

Obviamente cada x_j^i em uma base de dados pode ser visto como uma palavra sobre algum alfabeto. Para prosseguir, e apresentar o algoritmo de transformação de uma base de dados para um PAFHTD, antes é necessário considerar as seguintes definições.

Definição 6.1.2. *Seja X_d^n uma base de dados e seja $[x_1^i, \dots, x_{d-1}^i, x_d^i]$ um elemento em X_d^n , ou seja, uma linha de X_d^n com $1 \leq i \leq n$, então $x_1^i \dots x_{d-1}^i$ é a palavra que descreve o i -ésimo elemento, uma y -palavra é uma palavra $x_1^i \dots x_{d-1}^i$ em que $x_d^i = y$.*

Exemplo 6.1.2. *Considere a base expressa no Exemplo 6.1.1 tem-se que 41mss é a palavra que descreve o quarto elemento da base.*

Observação 6.1.1. *Note que a palavra 10mns descreve tanto o primeiro quanto o terceiro elemento da base, assim uma palavra pode descrever diversos elementos em uma base de dados.*

Definição 6.1.3. *Seja X_d^n uma base de dados $Lis_{X_d^n}$ é a lista de todas as palavras que descrevem os elementos de X_d^n e $Lis_{X_d^n}^y$ é a lista de todas as y -palavras sobre X_d^n .*

Exemplo 6.1.3. *Considerando a base X_d^n expressa no Exemplo 6.1.1 tem-se que:*

- $Lis_{X_d^n} = (10mns, 25fsn, 10mns, 41mss).$
- $Lis_{X_d^n}^{doente} = (10mns, 25fsn, 41mss).$
- $Lis_{X_d^n}^{curado} = (10mns).$

Definição 6.1.4. *Seja X_d^n uma base de dados, o conjunto $\mathcal{W}_{X_d^n}$ é nomeado como, conjunto das palavras da base de dados X_d^n . Além disso, o conjunto $\mathcal{W}_{X_d^n}^y$ é nomeado como conjunto das y -palavras da base de dados X_d^n .*

Exemplo 6.1.4. *Considerando a base X_d^n expressa no Exemplo 6.1.1 tem-se que:*

- $\mathcal{W}_{X_d^n} = \{10mns, 25fsn, 41mss\}.$
- $\mathcal{W}_{X_d^n}^{doente} = \{10mns, 25fsn, 41mss\}.$
- $\mathcal{W}_{X_d^n}^{curado} = \{10mns\}.$

Observação 6.1.2. *Obviamente é claro que as seguintes desigualdades $\#\mathcal{W}_{X_d^n} \leq \#Lis_{X_d^n}$ e $\#\mathcal{W}_{X_d^n}^y \leq \#Lis_{X_d^n}^y$ são sempre satisfeitas. Além disso, para todo y tem-se que $\mathcal{W}_{X_d^n}^y \subseteq \mathcal{W}_{X_d^n}$.*

Agora para converter uma base de dados em um AFD, a ideia é bem simples, basta criar um AFD que seja capaz de computar todas as palavras do conjunto $\mathcal{W}_{X_d^n}$, isto é feito seguindo os passos descritos pelo Algoritmo 5 a seguir. Este algoritmo tem um funcionamento similar ao Algoritmo 12.1 do livro (HIGUERA, 2010), a diferença básica entre os dois algoritmos é a indexação dos estados, no Algoritmo 5 a indexação é feita usando números inteiros e para o algoritmo de (HIGUERA, 2010) a indexação usa strings. Outra diferença é que o Algoritmo 5 não necessita de uma função externa que gera todas os prefixos das palavras de $\mathcal{W}_{X_d^n}$ como acontece no algoritmo de (HIGUERA, 2010).

Algoritmo 5 Algoritmo para inferir um AFD a partir de um conjunto de palavras.

Entrada Um conjunto $\mathcal{W}_{X_d^n}$.

Saída Um AFD $A = \langle Q, \Sigma, \delta, q_0, \emptyset \rangle$.

```

1:  $Q \leftarrow \{q_0\}$ 
2:  $\Sigma \leftarrow \emptyset$ 
3: para cada  $w = a_1 \cdots a_{d-1} \in \mathcal{W}_{X_d^n}$  faça
4:    $q \leftarrow q_0$ 
5:    $i \leftarrow 1$ 
6:   enquanto  $i \leq d - 1$  faça
7:     se  $\delta(q, a_i)$  não está definido então
8:        $k \leftarrow \#Q$ 
9:       Adicione o estado  $q_k$  em  $Q$  e  $a_i$  no alfabeto  $\Sigma$ 
10:      Defina  $\delta(q, a_i) = q_k$ 
11:    fim se
12:     $q \leftarrow \delta(q, a_i)$ 
13:     $i \leftarrow i + 1$ 
14:  fim enquanto
15: fim para
16: Defina  $A = \langle Q, \Sigma, \delta, q_0, \emptyset \rangle$ 
17: retorne  $A$ 

```

Exemplo 6.1.5. Considere o conjunto $\mathcal{W}_{X_d^n}$ do Exemplo 6.1.4, não é difícil verificar que o AFD gerado pelo Algoritmo 5 para o conjunto $\mathcal{W}_{X_d^n}$ corresponde à Figura 21 a seguir.

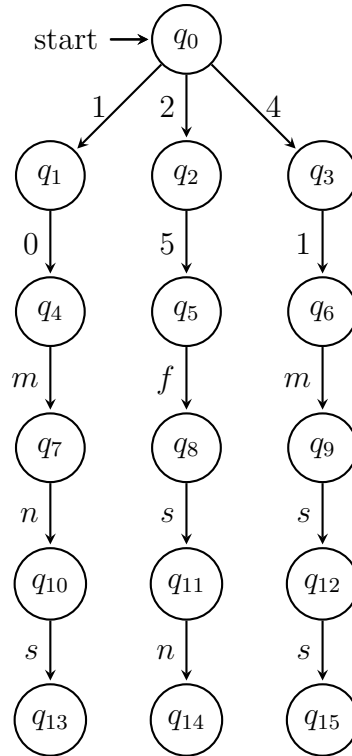


Figura 21 – Representação visual do AFD gerado para o conjunto $\mathcal{W}_{X_d^n}$ do Exemplo 6.1.4.

O próximo passo para a obtenção de um PAFHTD é valorar os estados finais, aqui a ideia é que, valorar um estado q_i é valorar o predicado, “a palavra w tal que $\hat{\delta}(q_0, w) = q_i$, não possui nenhum erro”, note que dizer que este predicado é verdadeiro, é dizer que todos os atributos¹ que descrevem um determinado elemento de X_d^n estão todos corretos.

Para realizar a valoração dos estados finais, aqui será adotada uma ferramenta matemática bastante conhecida nas áreas de processamento de imagens (GONZALEZ; WOODS, 2009) e teoria da informação (SHANNON, 1948), tal ferramenta se chama transformada *wavelet* discreta de Haar, ou simplesmente, transformada de Haar (STANKOVIĆ; FALKOWSKI, 2003), tal ferramenta é descrita brevemente no Apêndice D. Além disso, serão consideradas as duas definições que se segue.

Definição 6.1.5. *Seja $w = a_1 \cdots a_n \in Lis_{X_d^n}$. Um estado $q \in Q - \{q_0\}$ é dito usado por w , se e somente se, $\delta(q_0, a_1) = q_1, \dots, \delta(q_{n-1}, a_n) = q_n$ e $q = q_i$ para algum $1 \leq i \leq n$.*

Exemplo 6.1.6. *Considere o autômato da Figura 21, tem-se que o estado q_7 é usado pela palavra 10mns.*

Definição 6.1.6. *Seja $Lis_{X_d^n} = (w_1, w_2, \dots, w_n)$, para todo $q \in Q$ o sinal de uso do estado q com respeito à lista $Lis_{X_d^n}$ é um vetor de n valores definidos da seguinte forma,*

$$Sig(q) = \begin{cases} (\underbrace{0, \dots, 0}_{n\text{-vezes}}), & \text{se } q = q_0 \\ (v_1, \dots, v_n), & \text{senão} \end{cases} \quad (6.1)$$

tal que para todo $1 \leq i \leq n$,

$$v_i = \begin{cases} 1, & \text{se } q \text{ é usado por } w_i \\ 0, & \text{senão} \end{cases} \quad (6.2)$$

Exemplo 6.1.7. *Considerando a lista $Lis_{X_d^n}$ do Exemplo 6.1.3 para os estados q_0, q_8, q_{10} e q_{12} tem-se que,*

$$\begin{aligned} Sig(q_0) &= (0, 0, 0, 0) \\ Sig(q_8) &= (0, 1, 0, 0) \\ Sig(q_{10}) &= (1, 0, 1, 0) \\ Sig(q_{12}) &= (0, 0, 0, 1) \end{aligned}$$

O algoritmo 6 a seguir é responsável por valorar os estados finais do AFD, afim de, transformar o AFD em um PAFHTD, ou seja, ele é o algoritmo que converte a base de dados em um PAFHTD.

Observação 6.1.3. *A função normalize usado no Algoritmo 6 diz respeito à normalização de valores de uma lista para valores no intervalo $[0,1]$, e a função set diz respeito à função que converte uma lista em um conjunto.*

¹ No caso de w ser um prefixo de algum $w' \in W_{X_d^n}$, mas que $w \neq w'$, a valoração do predicado como verdadeiro significa que alguns dos atributos que descrevem um elemento de X_d^n estão corretos.

Algoritmo 6 Algoritmo para valorar os estados finais de um AFD a partir de uma lista de palavras.

Entrada Uma lista $Lis_{X_d^n}$ e um AFD $A = \langle Q, \Sigma, \delta, q_0, \emptyset \rangle$.

Saída Um PAFHTD $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, \mathcal{F} \rangle$.

- 1: **para** cada $w \in Lis_{X_d^n}$ **faça**
 - 2: **para** cada $q \in Q$ **faça**
 - 3: Compute $Sig(q)$
 - 4: Compute o Algoritmo 9 com entrada $Sig(q)$ e $k = \log_2(\#Lis_{X_d^n})$, e salve o resultado em T_L
 - 5: Defina $\mathcal{F}(q) = \bigcup_{T \in T_L} set(normalize(T))$
 - 6: **fim para**
 - 7: **fim para**
 - 8: Defina $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, \mathcal{F} \rangle$
 - 9: **retorne** A
-

6.2 Generalizando os PAFHTD

Não é difícil verificar que os PAFHTD's gerados pelo Algoritmo 6 possuem a característica de que, a valoração das computações dos mesmos são diferente de $\{0\}$ apenas para as palavras que foram usadas em suas construções, isto é, apenas para $w \in Lis_{X_d^n}$. Do ponto de visto de inteligência artificial isso não é uma propriedade interessante, visto que, isso caracterizaria o super ajustamento à base de treinamento (FACELI et al., 2011).

Uma forma de contornar esse problema, é descrita para autômatos probabilísticos em (HIGUERA, 2010), tal método é baseado na minimização aproximada, isto é, um método que minimiza o autômato mas não se preocupa em preservar a linguagem reconhecida, neste trabalho será adotado a mesma estratégia e para isso é necessário considerar as seguintes definições.

Definição 6.2.1. *Seja $X, Y \in \mathbb{H}$ a $\mathbb{H}$ -distância entre X e Y é uma função $d : \mathbb{H} \times \mathbb{H} \rightarrow [0, 1]$ que satisfaz as seguintes condições:*

$$(d_1) \quad 0 \leq d(X, Y) \leq 1.$$

$$(d_2) \quad d(X, Y) = 0 \text{ se, e somente se, } X = Y.$$

$$(d_3) \quad d(X, Y) = d(Y, X).$$

Definição 6.2.2. *As $\mathbb{H}$ -distância D_p, D_m e D_j são definidas² como:*

$$D_p(X, Y) = \max_{x \in X, y \in Y} |x - y| \quad (6.3)$$

$$D_m(X, Y) = \frac{1}{\#X \cdot \#Y} \left(\bigoplus_{x \in X, y \in Y} |x - y| \right) \quad (6.4)$$

² Vale ressaltar que estas distâncias são métricas já bem conhecida na área de topologia geral (PITOMBEIRA, 1971).

$$D_j(X, Y) = 1 - \frac{\#(X \cap Y)}{\#(X \cup Y)} \quad (6.5)$$

onde $\bigoplus_{x \in X, y \in Y}$ é o somatório indexado.

Definição 6.2.3. *Seja $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, \mathcal{F} \rangle$ um PAFHTD e $\alpha \in [0, 1]$. Dois estados $q, q' \in Q$ são ditos α -próximos com respeito a uma $\mathbb{H}$ -distância D sempre que,*

$$D(\mathcal{F}(q), \mathcal{F}(q')) \leq \alpha \quad (6.6)$$

Baseado no algoritmo Alergia ([HIGUERA, 2010](#)), que realiza a generalização de autômatos probabilísticos, agora será apresentado uma versão do Alergia para PAFHTD.

Algoritmo 7 Alergia para PAFHTD, baseado no Alergia definido em ([HIGUERA, 2010](#)).

Entrada Um PAFHTD $\mathcal{M} = \langle Q, \Sigma, \delta, q_0, \mathcal{F} \rangle$, $\alpha \in [0, 1]$ e uma $\mathbb{H}$ -distância D .

Saída Um PAFHTD $\mathcal{M}'$ tal que $|\mathcal{M}'| \leq |\mathcal{M}|$.

```

1:  $reds \leftarrow \{q_0\}$ 
2:  $blues \leftarrow reds$ 
3: enquanto  $\#blues \geq 0$  faça
4:   selecione aleatoriamente um  $q_b \in blues$ 
5:    $promover \leftarrow Verdadeiro$ 
6:   para  $q_r \in reds$  faça
7:     se  $D(\mathcal{F}(q_b), \mathcal{F}(q_r)) \leq \alpha$  então
8:       Execute o algoritmo de Merge para  $q_b$  e  $q_r$ 
9:        $promover \leftarrow Falso$ 
10:    Interrompa o laço para
11:  fim se
12: fim para
13: se  $promover = Verdadeiro$  então
14:    $reds \leftarrow reds \cup \{q_b\}$ 
15: fim se
16:  $Temp \leftarrow \emptyset$ 
17: para  $q_r \in reds$  faça
18:   para  $a \in \Sigma$  faça
19:     se  $\delta(q_r, a) \notin reds$  então
20:        $Temp \leftarrow Temp \cup \{\delta(q_r, a)\}$ 
21:   fim se
22: fim para
23: fim para
24:  $blues \leftarrow Temp$ 
25: fim enquanto
26: Defina  $\mathcal{M} = \langle reds, \Sigma, \delta|_{reds}, q_0, \mathcal{F}|_{reds} \rangle$ 
27: retorne  $\mathcal{M}$ 

```

Observação 6.2.1. *O algoritmo Merge usado no Algoritmo 7 está definido em ([HIGUERA, 2010](#)).*

A demonstração das propriedades de generalização, minimização (ou redução) e de não preservação de linguagem do Algoritmo 7 é similar a do Alergia original em (HIGUERA, 2010), além disso, o Alergia apresentado em (HIGUERA, 2010) funciona sobre autômatos probabilísticos e o Algoritmo 7 sobre PAFHTD, assim eles diferem na natureza de suas máquinas básicas. Outro ponto é que o Algoritmo 7 considera $\mathbb{H}$ -distâncias arbitrárias, enquanto que o Alergia original considera apenas a distância padrão entre números reais.

6.3 O Algoritmo HeCA

Agora que foram introduzidos as ferramentas básicas para o algoritmo *Hesitant classification automaton*, ou simplesmente HeCA, é possível apresentar o algoritmo em si, e discutir seu funcionamento, para prosseguir, antes é necessário considerar um resultado básico.

Teorema 6.3.1. *Seja $\{\mathcal{M}_1, \dots, \mathcal{M}_k\}$ uma família finita de PAFHTD's sobre o mesmo alfabeto Σ , então existe um PAFHTD $\mathcal{M}'$ tal que, $\hat{f}_{\mathcal{M}'} = \bigsqcup_{i=1}^k \mathcal{F}_{\mathcal{M}_k}$.*

Demonstração. Dado uma família finita $\{\mathcal{M}_1, \dots, \mathcal{M}_k\}$ como para todo $i \leq k$ tem-se que $\mathcal{M}_i = \langle Q_i, \Sigma, \delta_i, q_0^i, \mathcal{F}_i \rangle$, agora defina $Q = Q_1 \times \dots \times Q_k$ e para todo $a \in \Sigma$:

$$\delta\left((q^1, \dots, q^k), a\right) = \left(\delta_1(q^1, a), \dots, \delta_k(q^k, a)\right)$$

onde $q^i \in Q_i$ para todo $i \leq k$, além disso, defina:

$$\mathcal{F}\left((q^1, \dots, q^k)\right) = \bigsqcup_{i=1}^k \mathcal{F}_i(q^i)$$

Agora fazendo $\mathcal{M}' = \langle Q, \Sigma, \delta, (q_0^1, \dots, q_0^k), \mathcal{F} \rangle$, por indução sobre o tamanho das palavras não é difícil verificar que $\hat{f}_{\mathcal{M}'} = \bigsqcup_{i=1}^k \hat{f}_{\mathcal{M}_k}$. □

Observação 6.3.1. *O resultado do teorema acima não é uma supresa, na verdade este resultado é similar ao já demonstrado pelo Corolário 4.3.1.*

A ideia por trás do algoritmo HeCA pode ser descrita da seguinte forma, para toda classe y na base de dados construir um PAFHTD $\mathcal{M}_y$ que reconheça os exemplos que são da classe y , em seguida generalizar cada $\mathcal{M}_y$ usando o Algoritmo 7, depois construir um PAFHTD $\mathcal{M}$ usando o Teorema 6.3.1. Então para todo exemplo, isto é, toda palavra w , que se quer classificar, compute $\hat{f}_{\mathcal{M}_k}(w)$ para todo $1 \leq i \leq k$ e também compute $\hat{f}_{\mathcal{M}}(w)$, depois usando alguma das distâncias da Definição 6.2.1 encontrar o j tal que,

$$\min_{i=1}^k D(\hat{f}_{\mathcal{M}_k}(w), \hat{f}_{\mathcal{M}}(w)) = D(\hat{f}_{\mathcal{M}_j}(w), \hat{f}_{\mathcal{M}}(w)) \quad (6.7)$$

ou seja, encontrar o PAFHTD que mais influência na $\mathbb{H}$ -união. Depois disso classificar o exemplo como pertencente à classe usada para construir $\mathcal{M}_j$, caso existam mais de um j que satisfazem a igualdade da Equação (6.7), então a classe deve ser decidida aleatoriamente entre as classe usadas na construção de todos os PAFHTD's que satisfazem a igualdade da Equação (6.7). A seguir é exposto o pseudo código do HeCA e em seguida um estudo experimental de tal algoritmo.

Observação 6.3.2. *Agora semanticamente falando, quando é aplicado uma palavra w a um PAFHTD $\mathcal{M}_y$, isto é o mesmo que valorar o predicado, “O exemplo w pertence à classe y ”, por outro lado, ao aplicar a palavra ao PAFHTD $\mathcal{M}$, é o mesmo que valorar o predicado, “O exemplo w pertence a alguma das classes”. Isso significa que quando $\hat{f}_{\mathcal{M}}(w) = \{0\}$ é o mesmo que dizer que, w não pertence a nenhuma classe do problema.*

Algoritmo 8 Algoritmo de Classificação HeCA.

Entrada Uma Base de Dados X_d^n e uma lista de exemplos Ex a ser classificados, um $\alpha \in [0, 1]$ e uma $\mathbb{H}$ -distância D .

Saída Uma lista que contém a classe dos exemplos em Ex .

```

1: para cada classe  $y$  em  $X_d^n$  faça
2:   Execute o Algoritmo 5 com entrada  $\mathcal{W}_{X_d^n}$  e gere o AFD  $A$ .
3:   Execute o Algoritmo 6 com entrada  $A$  e  $Lis_{X_d^n}$  e gere  $\mathcal{M}_k$ 
4:   Execute o Algoritmo 7 com entrada  $\mathcal{M}_k$ , o valor  $\alpha$  e a distância  $D$ .
5: fim para
6: Construa usando o Teorema 6.3.1 o PAFHTD  $\mathcal{M}$ .
7: Defina a lista vazia  $clc_{Ex}$ 
8: para cada  $w \in Ex$  faça
9:   para cada classe  $y$  em  $X_d^n$  faça
10:    Compute  $\hat{f}_{\mathcal{M}_k}(w)$ 
11:   fim para
12:   Compute  $\hat{f}_{\mathcal{M}}(w)$ 
13:   se  $\hat{f}_{\mathcal{M}}(w) \neq \{0\}$  então
14:     Encontre o conjunto  $\mathbb{J}$  de todos os  $j$  tais que  $\min_{i=1}^k D(\hat{f}_{\mathcal{M}_i}(w), \hat{f}_{\mathcal{M}}(w)) =$ 
        $D(\hat{f}_{\mathcal{M}_j}(w), \hat{f}_{\mathcal{M}}(w))$ 
15:     se  $\#\mathbb{J} > 1$  então
16:       Escolha aleatoriamente  $j \in \mathbb{J}$ 
17:       adicione o  $j$  no final da lista  $clc_{Ex}$ 
18:     senão
19:       adicione o  $j$  contido em  $\mathbb{J}$  no final da lista  $clc_{Ex}$ 
20:     fim se
21:   senão
22:     adicione a informação “exemplo não classifique  $w$ ” ao fim da lista  $clc_{Ex}$ 
23:   fim se
24: fim para
25: retorne  $clc_{Ex}$ 

```

6.4 Análise experimental

Para realizar a análise experimental do algoritmo HeCA foram realizados diversos experimentos, com diferentes configurações para o algoritmo HeCA, a tabela a seguir detalha as configurações do HeCA adotadas.

id da configuração	valor de α	distância usada
H-1	0.1	D_j
H-2	0.3	D_j
H-3	0.5	D_j
H-4	0.1	D_p
H-5	0.3	D_p
H-6	0.5	D_p
H-7	0.1	D_m
H-8	0.3	D_m
H-9	0.5	D_m

Tabela 9 – Configurações usada do Algoritmo HeCA.

Com relação aos experimentos foram adotadas 5 base de dados já bem conhecidas da área de aprendizado de máquina descritas na Tabela 10 a seguir.

id da base	número de exemplos	número de atributos	número de classes
kr-vs-kp	3196	37	2
tae	151	6	3
tic-tac-toe	958	10	2
vote	435	17	2
zoo	101	18	7

Tabela 10 – Base de dados usadas nos experimentos.

A estratégia experimental adotada neste trabalho foi o *cross-validation* com 10-folds (COSTA et al., 2018). Neste trabalho como métricas de comparação para avaliar o desempenho experimental, foram adotadas duas medidas padrão (entre muitas possíveis) da área de aprendizado de máquina (FACELI et al., 2011), a acurácia e a *f1-measure*.

As Tabelas 11 e 12 apresentam respectivamente os resultados médios normalizados (com 3 casas decimais), das métricas acurácia e *f1-measure* para todas as configurações do HeCA em cada base, e de forma geral do experimento.

Observação 6.4.1. *Em ambas as métricas quanto mais próximo de 1 melhor é o resultado, além disso, nas Tabelas 11 e 12, o melhor resultado médio total dos experimentos encontra-se destacado em negrito.*

id da base	H-1	H-2	H-3	H-4	H-5	H-6	H-7	H-8	H-9
kr-vs-kp	0.586	0.581	0.673	0.942	0.561	0.568	0.945	0.934	0.938
tae	0.508	0.475	0.474	0.668	0.653	0.373	0.620	0.648	0.661
tic-tac-toe	0.960	0.992	0.992	0.504	0.951	0.540	0.509	0.476	0.480
vote	0.517	0.573	0.517	0.709	0.592	0.549	0.717	0.723	0.746
zoo	0.600	0.560	0.660	0.630	0.740	0.500	0.700	0.710	0.690
resultado médio total	0.634	0.636	0.663	0.690	0.699	0.506	0.698	0.698	0.703

Tabela 11 – Acurácia média para as configurações do Algoritmo HeCA.

id da base	H-1	H-2	H-3	H-4	H-5	H-6	H-7	H-8	H-9
kr-vs-kp	0.570	0.556	0.651	0.939	0.457	0.441	0.941	0.932	0.936
tae	0.476	0.449	0.417	0.658	0.630	0.204	0.599	0.635	0.650
tic-tac-toe	0.955	0.990	0.990	0.495	0.946	0.394	0.501	0.465	0.469
vote	0.444	0.527	0.443	0.700	0.424	0.349	0.709	0.713	0.732
zoo	0.395	0.337	0.382	0.463	0.603	0.236	0.546	0.565	0.565
resultado médio total	0.568	0.571	0.576	0.651	0.612	0.324	0.659	0.662	0.670

Tabela 12 – média da métrica F1-measure para as configurações do Algoritmo HeCA.

Como pode ser visto nas Tabelas 11 e 12 a configuração H-9 do algoritmo HeCA obteve o melhor desempenho numérico em ambas as métricas de comparação, assim essa configuração foi determinada com sendo a melhor numericamente, e foi escolhida para ser testada estatisticamente com o teste- t (COSTA et al., 2018), contra os tradicionais métodos de classificação KNN e árvore de decisão (FACELI et al., 2011), as médias de desempenho de ambos os algoritmos estão destacadas na Tabela 13.

nome do método	Média total da acurácia	Média total da $f1$ -measure
KNN	0.91	0,9
árvore de decisão	0.6	0.58

Tabela 13 – Média dos resultados para os algoritmo KNN e árvore de decisão.

Pelo teste estatístico levando em conta um valor de confiança igual a 95% e considerada a hipótese nula de que:

$$H_0: \text{O desempenho do HeCA é superior ou igual.}$$

foram obtidos os valores de p e t esboçados na Tabela 14 para a acurácia contra os métodos do KNN e da árvore de decisão.

nome do método	valor de p	valor de t
KNN	0,031716921669594	-1,89919752736753
árvore de decisão	0,483965843319756	-0,040408458029094

Tabela 14 – Valores de p e t do teste estatístico para a acurácia.

Agora analisando com respeito ao KNN tem-se que $p < 0.05$ e $t < 0$, assim pela definição da aceitação ou rejeição da hipótese nula no teste- t (COSTA et al., 2018), a

hipótese nula deve ser rejeitada, e portanto, com respeito à acurácia o KNN é estatisticamente superior ao HeCA. Já para a árvore de decisão tem-se que $p > 0.05$ e $t < 0$, então a hipótese nula não pode ser rejeitada, e portanto, a hipótese nula de que o desempenho do HeCA é maior ou igual estatisticamente (em relação a acurácia) ao desempenho do algoritmo de árvore de decisão, é válida para as bases testadas.

Em relação à métrica *f1-measure* foram obtidos valores de p e t descritos na Tabela 15 a seguir.

nome do método	valor de p	valor de t
KNN	0,005735217050277	-2,62684841343468
árvore de decisão	0,268180173806588	-0,622713384111219

Tabela 15 – Valores de p e t do teste estatístico para a *f1-measure*.

Refazendo a análise para o KNN, novamente tem-se um cenário em que $p < 0.05$ e $t < 0$, portanto, em relação a *f1-measure* o KNN é estatisticamente superior ao HeCA, pois a hipótese nula deve ser rejeitada. Já para a árvore de decisão, o cenário novamente se repete de ter $p > 0.05$ e $t < 0$, então novamente a hipótese nula não pode ser rejeitada, e assim a hipótese de que desempenho do HeCA é maior ou igual estatisticamente ao desempenho do algoritmo de árvore de decisão é válida para as bases testadas, sob a ótica da métrica *f1-measure*.

6.5 Considerações Finais

Neste capítulo foi apresentado um algoritmo para a classificação de dados, que foi nomeado com HeCA, esse algoritmo usa os pseudo autômatos *fuzzy* hesitantes típicos determinísticos (introduzidos no Capítulo 4) como parte de um mecanismo de decisão, para classificar exemplos não rotulados.

A construção do HeCA se baseia substancialmente, no Algoritmo Alergia apresentado em (HIGUERA, 2010) e também é dependente da ferramenta matemática chamada transformada de Haar (STANKOVIĆ; FALKOWSKI, 2003). Com respeito ao Algoritmo 7, diferentemente do algoritmo Alergia de (HIGUERA, 2010), ele se mostra como mais generalista, um vez que, o mesmo funciona usando distâncias arbitrárias, e não apenas uma distância fixa.

Em relação ao desempenho, primeiro foi numericamente determinada que a configuração do algoritmo HeCA que possui melhor desempenho, em relação às métricas acurácia e *f1-measure*, é aquela usando a distância média e o valor de *alpha* igual a 0.5. Após essa conclusão inicial, esta configuração do HeCA foi testada estatisticamente, usando o teste- t , contra os tradicionais métodos de classificação de dados KNN e árvore de decisão.

O desempenho do HeCA com respeito ao algoritmo KNN, é inferior estatisticamente em ambas as métricas, entretanto, em relação ao algoritmo da árvore de decisão, o HeCA se mostrou igual ou superior ao método tradicional, isso mostra que o método de classificação baseado em PAFHTD, isto é, o algoritmo HeCA, é um método viável e com desempenho estatisticamente similar a um dos métodos mais clássicos da literatura de aprendizado de máquina.

7 Conclusão

Neste capítulo são apresentadas as considerações finais deste trabalho. Portanto, as contribuições produzidas no desenvolvimento desta tese são revisitadas de forma sucinta. Por fim, serão apresentados possíveis trabalhos futuros como continuação deste trabalho.

7.1 Resultados Teóricos

Na literatura, como descrito no Capítulo 3 existem diversas extensões de autômatos *fuzzy*, cada uma a sua maneira ataca o problema de representar as incertezas nos processos de computação *fuzzy*. As contribuições deste trabalho começam no Capítulo 4, em que foi apresentada uma nova extensão para a teoria dos autômatos *fuzzy*.

Entretanto, as extensões de autômatos *fuzzy* são como destacado em (QIU, 2001), máquinas de computação valoradas, dessa forma, a primeira contribuição marcante desta tese se caracteriza como sendo a construção de uma estrutura algébrica de semi-anel para o conjunto dos elementos *fuzzy* hesitantes típicos ($\mathbb{H}$), esta estrutura baseada nas novas operações de sup-combinação e inf-combinação (também introduzidas neste trabalho), fornecem o espaço de valoração sobre o qual os autômatos *fuzzy* hesitantes típicos (AFHT's) são construídos, outra contribuição relacionada a esta estrutura, foi fornecer uma relação de ordem parcial que generaliza a ordem usual $\leq$ no conjunto $\mathbb{H}$, sendo esse um resultado que vinha sendo buscando nos últimos anos na literatura (SANTOS et al., 2015; GARMENDIA; CAMPO; RECASENS, 2017).

O conceito de AFHT's, apresentado neste trabalho, é que tais modelos de autômatos são capazes de lidar com a incertezas apresentadas naturalmente pela tarefa de computar instruções *fuzzy*, e também que agrega em sua própria definição a hesitação de se definir os valores de pertinência com que as instruções são computadas pelos autômatos.

No decorrer do Capítulo 4, sobre a teoria dos AFHT's foi apresentado uma generalização da ideia de determinismo, e assim foi introduzida a subclasse dos autômatos *fuzzy* hesitantes típicos determinísticos, ou simplesmente AFHTD's, foi então mostrado duas formas de converter AFHT's em AFHTD's, uma conservando a linguagem reconhecida e outra sem a conservação da linguagem.

Ainda sobre as linguagens reconhecidas por AFHT's, foi apresentada uma caracterização para as mesmas, mostrando as condições suficientes e necessárias para que um conjunto *fuzzy* hesitante típico seja aceito (ou computado) por um AFHT, isto é, as condições suficientes e necessárias para que um conjunto *fuzzy* hesitante típico seja $\mathbb{H}$ -regular. Esse resultado permitiu concluir que a família de conjuntos *fuzzy* hesitantes típicos é

expressivamente maior que a família de linguagens *fuzzy* hesitantes típicas $\mathbb{H}$ -regulares.

Além disso, foi demonstrado que os AFHT's são significativamente mais poderosos que os autômatos *fuzzy*, de fato, foi mostrado que o poder de um único AFHT é equivalente ao poder de uma família finita de autômatos *fuzzy*. Isso em um certo sentido, permitiu dar um significado semântico para a ideia de AFHT como sendo, um computador que executa múltiplos programas *fuzzy* em paralelo.

7.2 Resultados Práticos

Como mencionado no Capítulo 3, a própria origem dos autômatos *fuzzy* está ligada com a tarefa de classificação de dados (WEE, 1967), além disso, diversos trabalhos foram realizados sobre a aplicação de autômatos valorados no processamento de imagens (CULIK; KARI, 1993; CULIK; FRIŠ, 1995; CULIK; VALENTA, 1997), inspirado nesses trabalhos, o Capítulo 5 apresentada a contribuição experimental desta tese para a área de processamento de imagens (GONZALEZ; WOODS, 2009), em tal capítulo foi apresentado um algoritmo para tratamento de ruídos do tipo sal em imagens em escala de cinza. Esse algoritmo, nomeado como ReSPA, é baseado em pseudo autômatos *fuzzy* hesitantes típicos determinísticos (PAFHTD), que foram introduzidos no Capítulo 4.

Em relação ao desempenho foram usadas duas métricas comparativas fundamentais em processamento de imagens (GONZALEZ; WOODS, 2009), o PSNR e o RNMSE. Para ambas as métricas o algoritmo ReSPA com parâmetro $\alpha = 0$ se mostrou superior a alguns tradicionais métodos de remoção de ruído, em todos os cenários experimentais a que foi submetido, isso mostra o poder da aplicação prática dos PAFHTD's.

Como efeito colateral do algoritmo ReSPA, no Capítulo 5 também podem ser encontrados algoritmos capazes de gerar representações baseadas em autômatos finitos para imagens digitais em escala de cinza.

Já no Capítulo 6 é apresentada uma contribuição experimental na área de aprendizado de máquina, em tal capítulo foi apresentado um algoritmo para classificação de dados. Essa algoritmo recebeu o nome de HeCA, e assim como o ReSPA também é baseado em PAFHTD's.

Para medir o desempenho do HeCA, foram adotadas duas métricas da área de aprendizado de máquina, a acurácia e a *f1-measure*. Inicialmente foi determinado numericamente qual era a melhor configuração do algoritmo HeCA. Em seguida, esta configuração do HeCA foi testada estatisticamente, usando o teste-*t*, contra os tradicionais métodos de classificação de dados KNN e árvore de decisão.

Em relação às bases usadas o teste revelou que a performance do Heca é inferior estatisticamente ao KNN, porém, é similar estatisticamente às árvores de decisão, o que

corroborar o HeCA como uma opção válida na tarefa de classificar dados.

7.3 Trabalhos Futuros

Como trabalhos futuros, que darão diretamente continuidade à pesquisa realizada por essa tese pretende-se:

- Aprofundar os estudos sobre os AFHT e os PAFHT nas perspectivas: topológica e algébrica.
- Estudar outros operadores de fecho para as linguagens $\mathbb{H}$ -regulares, além da inf-combinação e da sup-combinação.
- Modificar o algoritmo ReSPA para tratar ruídos dos tipos: pimenta e gaussiano.
- Modificar o algoritmo HeCA para se aplicado nas tarefas de classificação hierárquica e multi-rótulo.
- Estudar outros modelos de máquinas de computação valoradas no espaço $\mathbb{H}$.

Além desses temas, atualmente encontra-se em desenvolvimento como projeto de iniciação científica na Universidade Federal do Vale do São Francisco, a construção de um simulador de autômatos valorados, tal simulador irá conter um módulo de simulação para AFHT e PAFHT.

Apêndices

APÊNDICE A – O Isomorfismo Entre os Conjuntos *Fuzzy* Sobre $\Sigma^\oplus$ E Sobre Σ^*

Aqui será apresentada a existência de um isomorfismo entre a família dos conjuntos *fuzzy* sobre $\Sigma^\oplus$ e a família dos conjuntos *fuzzy* sobre Σ^* , para isso, considere as definições básicas e os resultados apresentados nas seções a seguir.

A.1 Um pouco de Teoria dos Multiconjuntos

Definição A.1.1. *Seja Σ um alfabeto, um multiconjunto T sobre Σ é um par ordenado $\langle \Sigma, \text{map}_T \rangle$, onde $\text{map}_T : \Sigma \rightarrow \mathbb{N}$ é uma função que conta quantas vezes cada elemento de Σ aparece no multiconjunto T . O multiconjunto vazio $\langle \Sigma, \text{map}_\emptyset \rangle$ é tal que $\text{map}_\emptyset(x) = 0$ para todo $x \in \Sigma$. A família de todos os multiconjuntos sobre Σ , é denotado por $\Sigma^\oplus$.*

Exemplo A.1.1. *Seja $\Sigma = \{a, b, c, d\}$ o par $\langle \Sigma, \text{map}_T \rangle$ com $\text{map}_T(a) = \text{map}_T(b) = 3$ e $\text{map}_T(c) = \text{map}_T(d) = 1$ é um multiconjunto.*

Observação A.1.1. *Um multiconjunto T é dito unitário se para todo $x \in \Sigma$ tem-se que $\text{map}_T(x) = 1$.*

Observação A.1.2. *Aqui todo multiconjunto será escrito apenas por seu rótulo ficando a estrutura de par ordenado subentendida.*

Definição A.1.2. *Seja $T \in \Sigma^\oplus$ a cardinalidade de T é dada por:*

$$\#T = \bigoplus_{x \in \Sigma} \text{map}_T(x) \quad (\text{A.1})$$

onde $\bigoplus_{x \in \Sigma}$ é o somatório indexado pelos elementos do alfabeto Σ .

Como dito em (CARMO; GOUVEIA; DIONÍSIO, 2013), é possível usar a representação padrão da teoria dos conjuntos para representar os multiconjuntos finitos, a única diferença é que no caso dos multiconjuntos os elementos podem ser repetidos. Nessa representação o multiconjunto apresentado no Exemplo A.1.1 corresponde a $T = \{a, b, a, d, c, b, b, a\}$, lembrando que nesta representação a ordem dos elementos não é importante.

Lema A.1.1. *Seja Σ um alfabeto, então todo $T \in \Sigma^\oplus$ é finito.*

Demonstração. Dado um alfabeto Σ pela Definição 2.1.1 temos que existe um $n \in \mathbb{N}$ tal que $\Sigma = \{x_1, \dots, x_n\}$, agora para $T \in \Sigma^\oplus$, fazendo $\text{map}_T(x_i) = y_i$ para todo $1 \leq i \leq n$

tem-se que:

$$\#T = \bigoplus_{i=1}^n \text{map}_T(x_i) = y_1 + \cdots + y_n$$

como a soma é fechada nos naturais, existe um $y \in \mathbb{N}$ tal que $y = y_1 + \cdots + y_n$, consequentemente, $\#T = y$, e portanto, T é finito. $\square$

A.2 A Bijeção entre $\Sigma^\oplus$ e Σ^*

Como explicado em (BOOLOS; BURGESS; JEFFREY, 2012), sobre qualquer conjunto finito A é possível definir uma relação de ordem total¹ sobre A , denotada por $\preceq_A$, assim qualquer alfabeto Σ pode ser visto como um sistema totalmente ordenado por uma ordem total $\preceq_\Sigma$.

Observação A.2.1. *No caso de alfabetos essa ordem total costuma ser chamada de ordem alfabética.*

Como apontado pelo Lema A.1.1 todo elemento de $\Sigma^\oplus$ é finito, e portanto, assim como dito em (BOOLOS; BURGESS; JEFFREY, 2012), o processo de listagem de seus elementos é computável, isto é, para qualquer $T \in \Sigma^\oplus$ sempre existe um algoritmo para criar uma lista com os elementos de T seguindo algum critério, a seguir é formalizado um tipo (entre muitos possíveis) de lista sobre os elementos de $\Sigma^\oplus$.

Definição A.2.1. *Dado $T \in \Sigma^\oplus$, $|T\rangle$ corresponde à lista ordenada de forma crescente dos elementos de T com respeito a uma ordem total $\preceq_\Sigma$, isto é,*

$$|T\rangle = [x_1, \dots, x_{\#T}]$$

tal que $x_1 \preceq_\Sigma \cdots \preceq_\Sigma x_n$ com $x_i \in T$ e $1 \leq i \leq \#T$. No caso particular de $\emptyset$ temos que $|\emptyset\rangle = []$, onde $[]$ denota a lista vazia. O conjunto de todas as lista ordenadas de forma crescente sobre $\Sigma^\oplus$, será denotado por $\langle\langle \Sigma^\oplus \rangle\rangle$.

Teorema A.2.1. *A função $g : \Sigma^\oplus \rightarrow \langle\langle \Sigma^\oplus \rangle\rangle$ definida por,*

$$g(T) = \begin{cases} [], & \text{se } \#T = 0 \\ |T\rangle, & \text{senão} \end{cases}$$

é uma bijeção.

Demonstração. Trivial pela própria definição de g . $\square$

Agora será mostrado que o conjunto $\langle\langle \Sigma^\oplus \rangle\rangle$ é enumerável, ou seja, como explicado em (RIBENBOIM, 2012), será mostrado que existe uma bijeção entre $\langle\langle \Sigma^\oplus \rangle\rangle$ e o conjunto dos números naturais.

¹ Uma ordem total sobre um conjunto A é uma ordem parcial $\leq$ que satisfaz a seguinte condição: Para todo $x, y \in A$ se $x \neq y$, então $x < y$ ou $y < x$.

Definição A.2.2. Uma codificação do alfabeto Σ é uma bijeção $Cod : \Sigma \rightarrow \{1, \dots, n\}$.

Definição A.2.3. Seja $|T\rangle \in \langle\langle \Sigma^\oplus \rangle\rangle - \{[]\}$, um número de Gödel $n_{|T\rangle}$ associado a $|T\rangle$ é um número natural da forma,

$$n_{|T\rangle} = \prod_{i=1}^{\#T} p_i^{Cod(x_i)} \quad (\text{A.2})$$

onde x_i é o i -ésimo elemento da lista $|T\rangle$, p_i é o i -ésimo número primo. Aqui o conjunto de todos os números de Gödel é denotado por $\mathbb{N}_G$.

Proposição A.2.1. $0 \notin \mathbb{N}_G$.

Demonstração. Direto da definição de $n_{|T\rangle}$. □

Proposição A.2.2. Se $|T_1\rangle \neq |T_2\rangle$ tais que $|T_1\rangle, |T_2\rangle \neq []$, então $n_{|T_1\rangle} \neq n_{|T_2\rangle}$.

Demonstração. Direto pelo teorema fundamental da aritmética (SANTOS, 1998). □

Proposição A.2.3. (SERNADAS; SERNADAS, 2012) O conjunto $\mathbb{N}_G$ é enumerável.

Lema A.2.1. O conjunto $\mathbb{N}_G \cup \{0\}$ é enumerável.

Demonstração. É suficiente notar que a função $\mathcal{H} : \mathbb{N}_G \cup \{0\} \rightarrow \mathbb{N}$ definida por,

$$\mathcal{H}(x) = \begin{cases} h(x) + 1, & \text{se } x \in \mathbb{N}_G \\ 0, & \text{senão} \end{cases}$$

é uma bijeção, onde $h : \mathbb{N}_G \rightarrow \mathbb{N}$ é também uma bijeção cuja existência é garantida pela Proposição A.2.3. □

Teorema A.2.2. O conjunto $\langle\langle \Sigma^\oplus \rangle\rangle$ é enumerável.

Demonstração. Defina a função $COD : \langle\langle \Sigma^\oplus \rangle\rangle \rightarrow \mathbb{N}_G \cup \{0\}$ como sendo,

$$COD(|T\rangle) = \begin{cases} 0, & \text{se } |T\rangle = [] \\ n_{|T\rangle}, & \text{senão} \end{cases}$$

agora pela Proposição A.2.2 cada $n_{|T\rangle}$ é associado a um única lista $|T\rangle$, além disso, a função COD também associa o número 0 a uma única lista, além disso, pela própria definição é fácil ver que a função COD é uma bijeção. Por outro lado, usando a função $\mathcal{H}$ do Lema A.2.1 defina-se uma nova função $\mathcal{R} : \langle\langle \Sigma^\oplus \rangle\rangle \rightarrow \mathbb{N}$ como sendo $\mathcal{R} = \mathcal{H} \circ COD$, como COD e $\mathcal{H}$ são ambas bijeções, a função $\mathcal{R}$ é também uma bijeção, e portanto, $\langle\langle \Sigma^\oplus \rangle\rangle$ é enumerável. □

Corolário A.2.1. O conjunto $\Sigma^\oplus$ é enumerável.

Demonstração. Pela Teorema A.2.1 existe uma bijeção $g : \Sigma^\oplus \rightarrow \langle\langle \Sigma^\oplus \rangle\rangle$ e pelo Teorema A.2.2 existe uma bijeção $\mathcal{R} : \langle\langle \Sigma^\oplus \rangle\rangle \rightarrow \mathbb{N}$, como g e $\mathcal{R}$ são bijeções a função $fun : \Sigma^\oplus \rightarrow \mathbb{N}$, definida como $fun = \mathcal{R} \circ g$, é também uma bijeção, e portanto, $\Sigma^\oplus$ é enumerável. $\square$

Corolário A.2.2. *Existe uma bijeção $\mathcal{Z} : \Sigma^\oplus \rightarrow \Sigma^*$.*

Demonstração. Como apresentado em (HOPCROFT; MOTWANI; ULLMAN, 2008), o conjunto Σ^* é enumerável, logo existe uma bijeção $Enu : \Sigma^* \rightarrow \mathbb{N}$, além disso, pelo Corolário A.2.1 existe uma bijeção $fun : \Sigma^\oplus \rightarrow \mathbb{N}$, e assim a função $\mathcal{Z} : \Sigma^\oplus \rightarrow \Sigma^*$ definida como sendo,

$$\mathcal{Z} = Enu^{-1} \circ fun$$

é uma bijeção, completando a prova. $\square$

A.3 O Isomorfismo $Map_{\mathcal{Z}}$

Para provar a existência do isomorfismo entre as famílias de conjuntos fuzzy sobre $\Sigma^\oplus$ e Σ^* , deve-se considerar a proposição e a definição a seguir.

Proposição A.3.1. (CARMO; GOUVEIA; DIONÍSIO, 2013) *Sejam $\mathcal{E} : A \rightarrow B$, $\mathcal{O} : B \rightarrow C$ e $\mathcal{U} : C \rightarrow D$ funções e A, B, C e D conjuntos quaisquer, as seguintes igualdades são satisfeitas:*

$$P1. \mathcal{U} \circ (\mathcal{O} \circ \mathcal{E}) = (\mathcal{U} \circ \mathcal{O}) \circ \mathcal{E}.$$

$$P2. \text{ Se } \mathcal{E} \text{ é bijetora, então } \mathcal{E} \circ \mathcal{E}^{-1} = Id_B \text{ e } \mathcal{E}^{-1} \circ \mathcal{E} = Id_A.$$

$$P3. Id_B \circ \mathcal{E} = \mathcal{E} \text{ e } \mathcal{E} \circ Id_A = \mathcal{E}.$$

Definição A.3.1. *Seja Σ um alfabeto, $FS(\Sigma^\oplus)$ corresponde a família de todos os conjuntos fuzzy sobre $\Sigma^\oplus$ e $FS(\Sigma^*)$ corresponde à família de todos os conjuntos fuzzy sobre Σ^* . As relações:*

- $f \leq_{\Sigma^\oplus} g \iff f(T) \leq g(T)$ para todo $T \in \Sigma^\oplus$.
- $f' \leq_{\Sigma^*} g' \iff f'(w) \leq g'(w)$ para todo $w \in \Sigma^*$.

são respectivamente ordens parciais sobre $FS(\Sigma^\oplus)$ e $FS(\Sigma^*)$, com $\leq$ sendo a ordem usual nos números reais.

Teorema A.3.1. *$FS(\Sigma^\oplus)$ e $FS(\Sigma^*)$ são isomórficos.*

Demonstração. Dado a bijeção Z apresentada no Corolário A.2.2, defina uma função Map_Z de $FS(\Sigma^\oplus)$ em $FS(\Sigma^*)$ como sendo,

$$Map_Z(f) = f \circ Z^{-1}$$

pela definição de Map_Z é fácil ver que para qualquer conjunto *fuzzy* $f : \Sigma^\oplus \rightarrow [0, 1]$ tem-se que $Map_Z(f) : \Sigma^* \rightarrow [0, 1]$, logo Map_Z é bem definida. Agora suponha que $f, g \in \Sigma^\oplus$ tal que $f \leq_{\Sigma^\oplus} g$ logo,

$$\begin{aligned} (\forall T \in \Sigma^\oplus) f(T) \leq g(T) &\Rightarrow (\forall w \in \Sigma^*) f(Z^{-1}(w)) \leq g(Z^{-1}(w)) \\ &\Rightarrow Map_Z(f) \leq Map_Z(g) \end{aligned}$$

Portanto, Map_Z preserva a estrutura de sistema parcialmente ordenado, logo tal função é um homomorfismo de $FS(\Sigma^\oplus)$ em $FS(\Sigma^*)$. Além disso, a função Map_Z é inversível, isto é, a mesma possui uma inversa, pois dada a bijeção Z para todo conjunto *fuzzy* $g \in FS(\Sigma^*)$ a função Map_Z^{-1} é definida como sendo,

$$Map_Z^{-1}(g) = g \circ Z$$

É claro que Map_Z^{-1} é uma função de $FS(\Sigma^*)$ em $FS(\Sigma^\oplus)$ bem definida, também não é difícil verificar que Map_Z^{-1} é um homomorfismo. Por fim, note que para todo $f \in FS(\Sigma^\oplus)$ tem-se que,

$$\begin{aligned} Map_Z^{-1}(Map_Z(f)) &= Map_Z^{-1}(f \circ Z^{-1}) \\ &= (f \circ Z^{-1}) \circ Z \\ &\stackrel{P1}{=} f \circ (Z^{-1} \circ Z) \\ &\stackrel{P2}{=} f \circ Id_{\Sigma^\oplus} \\ &\stackrel{P3}{=} f. \end{aligned}$$

Dualmente, para todo $g \in FS(\Sigma^*)$ tem-se que,

$$\begin{aligned} Map_Z(Map_Z^{-1}(g)) &= Map_Z(g \circ Z) \\ &= (g \circ Z) \circ Z^{-1} \\ &\stackrel{P1}{=} g \circ (Z \circ Z^{-1}) \\ &\stackrel{P2}{=} g \circ Id_{\Sigma^*} \\ &\stackrel{P3}{=} g. \end{aligned}$$

Assim, de fato, Map_Z^{-1} é a função inversa de Map_Z , e portanto, Map_Z é uma função bijetora. Consequentemente, Map_Z é um isomorfismo, e assim $FS(\Sigma^\oplus)$ e $FS(\Sigma^*)$ são isomórficos, o que completa a prova. $\square$

APÊNDICE B – Formalizações para o processamento digital de imagens.

Aqui são apresentados os conceitos e o vocabulário usado pelos algoritmos no capítulo 5, neste apêndice será estabelecido um certo nível de formalismo para o processamento digital de imagens, e isto é feito com as definições e observações a seguir.

Definição B.0.1. *Seja A um conjunto não vazio, o conjunto de todas as matrizes de elementos de A com dimensão m por n é denotado por, $Mat(A, m, n)$, onde $m, n \in \mathbb{Z}_+$. A dimensão de uma matriz $\mathcal{I}$ é referenciada como $dim(\mathcal{I})$, isto é, se $\mathcal{I} \in Mat(A, m, n)$, então $dim(\mathcal{I}) = m \times n$.*

Definição B.0.2. *A representação digital de uma imagem em escala de cinza, é um elemento $\mathcal{I} \in Mat([0, 1], m, n)$.*

Observação B.0.1. *Uma imagem é dita ser quadrada quando sua matriz é quadrada, isto é, a imagem $\mathcal{I}$ pertence ao conjunto $Mat([0, 1], n, n)$ para algum $n \in \mathbb{Z}_+$. Neste caso particular, será escrito apenas que, $dim(\mathcal{I}) = n$, em vez de escrever $dim(\mathcal{I}) = n \times n$.*

Definição B.0.3. *Seja $\mathcal{I} \in Mat([0, 1], m, n)$ uma imagem, um pixel é qualquer par (i, j) tais que $i \leq m$ e $j \leq n$. A intensidade de um pixel é o valor $\mathcal{I}_{(i,j)}$ da matriz $\mathcal{I}$.*

Definição B.0.4. *Um quadrante $\mathcal{Q}$ em uma imagem $\mathcal{I} \in Mat([0, 1], m, n)$ é uma matriz dentro da imagem $\mathcal{I}$, ou seja, $dim(\mathcal{Q}) = m' \times n'$ com $m' \leq m$. Tal que para todo pixel (i, j) do quadrante $\mathcal{Q}$ existe um único par (k_1, k_2) com $k_1 \leq m, k_2 \leq n$ tal que $\mathcal{Q}_{(i,j)} = \mathcal{I}_{(i+k_1, j+k_2)}$*

Exemplo B.0.1. *Seja $\mathcal{I}$ a imagem representada pela matriz,*

$$\begin{bmatrix} \mathcal{I}_{(1,1)} & \mathcal{I}_{(1,2)} & \mathcal{I}_{(1,3)} & \dots & \mathcal{I}_{(1,n)} \\ \mathcal{I}_{(2,1)} & \mathcal{I}_{(2,2)} & \mathcal{I}_{(2,3)} & \dots & \mathcal{I}_{(2,n)} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \mathcal{I}_{(n,1)} & \mathcal{I}_{(n,2)} & \mathcal{I}_{(n,3)} & \dots & \mathcal{I}_{(n,n)} \end{bmatrix}$$

as matrizes $\begin{bmatrix} \mathcal{I}_{(1,1)} & \mathcal{I}_{(1,2)} \\ \mathcal{I}_{(2,1)} & \mathcal{I}_{(2,2)} \end{bmatrix}$ e $\begin{bmatrix} \mathcal{I}_{(1,2)} & \mathcal{I}_{(1,3)} \\ \mathcal{I}_{(2,2)} & \mathcal{I}_{(2,3)} \end{bmatrix}$ são quadrantes de $\mathcal{I}$.

Definição B.0.5. *Seja $\mathcal{I}$ uma imagem, dois quadrantes $\mathcal{Q}_1$ e $\mathcal{Q}_2$ de $\mathcal{I}$ são ditos disjuntos entre si, sempre que não existir interseção de pixels entre eles.*

Exemplo B.0.2. *Considere a imagem $\mathcal{I}$ do Exemplo B.0.1, tem-se que os quadrantes,*

$$\mathcal{Q}_1 = \begin{bmatrix} \mathcal{I}_{(1,1)} & \mathcal{I}_{(1,2)} \\ \mathcal{I}_{(2,1)} & \mathcal{I}_{(2,2)} \end{bmatrix} \quad e \quad \mathcal{Q}_2 = \begin{bmatrix} \mathcal{I}_{(n-1,1)} & \mathcal{I}_{(n-1,2)} \\ \mathcal{I}_{(n,1)} & \mathcal{I}_{(n,2)} \end{bmatrix}$$

são disjuntos entre si. Já o quadrante,

$$\mathcal{Q}_3 = \begin{bmatrix} \mathcal{I}_{(2,2)} & \mathcal{I}_{(2,3)} \\ \mathcal{I}_{(3,2)} & \mathcal{I}_{(3,3)} \end{bmatrix}$$

é disjunto de $\mathcal{Q}_2$, entretanto, ele não é disjunto de $\mathcal{Q}_1$, pois ambos possuem o pixel $(2,2)$.

Teorema B.0.1. *Se $\mathcal{I}$ é uma imagem tal que $\dim(\mathcal{I}) = 2^n$ para algum $n \in \mathbb{Z}_+$ com $n > 0$. Então existem no máximo 4 quadrantes disjuntos entre si em $\mathcal{I}$ com dimensão 2^{n-1} .*

Demonstração. Suponha que $\mathcal{I}$ é uma imagem tal que $\dim(\mathcal{I}) = 2^n$ para algum $n \in \mathbb{Z}_+$ com $n > 0$, assim é claro que,

$$\begin{aligned} \mathcal{Q}_1 &= \begin{bmatrix} \mathcal{I}_{(1,1)} & \cdots & \mathcal{I}_{(1,2^{n-1})} \\ \vdots & \ddots & \vdots \\ \mathcal{I}_{(2^{n-1},1)} & \cdots & \mathcal{I}_{(2^{n-1},2^{n-1})} \end{bmatrix} \\ \mathcal{Q}_2 &= \begin{bmatrix} \mathcal{I}_{(1,2^{n-1}+1)} & \cdots & \mathcal{I}_{(1,2^n)} \\ \vdots & \ddots & \vdots \\ \mathcal{I}_{(2^{n-1},2^{n-1}+1)} & \cdots & \mathcal{I}_{(2^{n-1},2^n)} \end{bmatrix} \\ \mathcal{Q}_3 &= \begin{bmatrix} \mathcal{I}_{(2^{n-1}+1,1)} & \cdots & \mathcal{I}_{(2^{n-1}+1,2^{n-1})} \\ \vdots & \ddots & \vdots \\ \mathcal{I}_{(2^n,1)} & \cdots & \mathcal{I}_{(2^n,2^{n-1})} \end{bmatrix} \\ \mathcal{Q}_4 &= \begin{bmatrix} \mathcal{I}_{(2^{n-1}+1,2^{n-1}+1)} & \cdots & \mathcal{I}_{(2^{n-1}+1,2^n)} \\ \vdots & \ddots & \vdots \\ \mathcal{I}_{(2^n,2^{n-1}+1)} & \cdots & \mathcal{I}_{(2^n,2^n)} \end{bmatrix} \end{aligned}$$

são todos quadrantes de $\mathcal{I}$ que são disjuntos entre si, além disso, é claro que $\dim(\mathcal{Q}_i) = 2^{n-1}$ para todo $i \leq 4$. Assuma também que existe outro quadrante de $\mathcal{I}$ da forma,

$$\mathcal{Q}' = \begin{bmatrix} \mathcal{I}_{(a,b)} & \cdots & \mathcal{I}_{(c,d)} \\ \vdots & \ddots & \vdots \\ \mathcal{I}_{(e,b)} & \cdots & \mathcal{I}_{(f,d)} \end{bmatrix}$$

tal que $\mathcal{Q}' \neq \mathcal{Q}_i$ com $\dim(\mathcal{Q}') = 2^{n-1}$ e que seja disjunto a todo $\mathcal{Q}_i$ com $i = 1, \dots, 4$. Agora para qualquer configuração de valores possível para a, b, c, d, e, f não é difícil mostrar que $\mathcal{Q}'$ sempre irá apresentar *pixels* comuns com algum dos quadrantes $\mathcal{Q}_i$, com $i = 1, \dots, 4$, o que contradiz a hipótese, e portanto, $\mathcal{Q}'$ não pode ser disjunto a $\mathcal{Q}_1, \mathcal{Q}_2, \mathcal{Q}_3$ e $\mathcal{Q}_4$ o que completa a prova. $\square$

Definição B.0.6. *Seja $\mathcal{I}$ uma imagem, o conjunto $\theta_{\mathcal{I}}$ é o conjunto de todos os quadrantes da imagem $\mathcal{I}$. Além disso, sobre o conjunto $\theta_{\mathcal{I}}$ é possível definir a seguinte relação de ordem parcial.*

$$\mathcal{Q} \lll \mathcal{Q}' \iff \mathcal{Q} \text{ é um quadrante de } \mathcal{Q}' \quad (\text{B.1})$$

Observação B.0.2. Dado $\mathcal{I} \in \text{Mat}([0, 1], m, n)$ é claro que $\lll$ é uma ordem parcial e que as matrizes unitárias $[\mathcal{I}_{(i,j)}]$ para todo $1 \leq i \leq m$ e $1 \leq j \leq n$ são os elementos mínimos e $\mathcal{I}$ é o maior elemento de $\langle \theta_{\mathcal{I}}, \lll \rangle$ respectivamente.

Entre as métricas para avaliação da qualidade dos métodos de processamento de imagens se destacam duas, o PSNR (FARIAS et al., 2016a; GONZALEZ; WOODS, 2009) e a raiz do erro médio quadrado normalizado (abreviado em inglês para RNMSE) (GUPTA; KLING, 2011). O PSNR entre duas imagens $\mathcal{I}_1, \mathcal{I}_2 \in \text{Mat}([0, 1], m, n)$ é uma medida em decibéis definido pela equação,

$$PSNR(\mathcal{I}, \mathcal{I}') = 20 \cdot \log_{10} \left(\frac{MAX_{\mathcal{I}}}{\sqrt{MSE(\mathcal{I}, \mathcal{I}')}} \right) \quad (\text{B.2})$$

onde $MAX_{\mathcal{I}}$ corresponde ao valor máximo da intensidade que pode ocorrer nos *pixels* de uma imagem $\mathcal{I}$, e MSE é o erro quadrático médio definido por:

$$MSE(\mathcal{I}, \mathcal{I}') = \frac{1}{mn} \sum_{i=1}^m \sum_{j=1}^n \|\mathcal{I}_{(i,j)} - \mathcal{I}'_{(i,j)}\|^2 \quad (\text{B.3})$$

em que $\|\bullet\|$ é a operação de módulo usual nos reais, e $\mathcal{I}_{(i,j)}, \mathcal{I}'_{(i,j)}$ são respectivamente os *pixels* (i, j) nas imagens $\mathcal{I}$ e $\mathcal{I}'$.

Observação B.0.3. Note que quando o MSE se aproxima de zero o $PSNR$ tende ao infinito. Além disso, quanto maior $PSNR$ de um processamento $\mathcal{P}$ sobre uma imagem $\mathcal{I}$ com respeito à própria imagem $\mathcal{I}$, melhor é a qualidade do processamento $\mathcal{P}$, isto é, quando $PSNR(\mathcal{P}'(\mathcal{I}), \mathcal{I}) \leq PSNR(\mathcal{P}(\mathcal{I}), \mathcal{I})$ significa que o processamento $\mathcal{P}$ é melhor que $\mathcal{P}'$.

Por sua vez, a métrica da raiz do erro médio quadrado normalizado de uma imagem $\mathcal{I}_1$ com respeito a uma imagem de referência $\mathcal{I}$ é formalmente definida por:

$$RNMSE(\mathcal{I}_1, \mathcal{I}) = \frac{RMSE(\mathcal{I}_1, \mathcal{I})}{\max(p) - \min(p)} \quad (\text{B.4})$$

onde $\max(p)$ e $\min(p)$ corresponde respectivamente ao valor da maior e menor intensidade que podem ocorrer nos *pixels* da imagem $\mathcal{I}$, e a medida de $RMSE$ entre duas imagens é definido como segue,

$$RMSE(\mathcal{I}_1, \mathcal{I}) = \sqrt{MSE(\mathcal{I}_1, \mathcal{I})} \quad (\text{B.5})$$

Observação B.0.4. Obviamente pela própria noção de erro, quanto menor o $RNMSE$ de um processamento $\mathcal{P}$ sobre uma imagem $\mathcal{I}$ com respeito à própria imagem $\mathcal{I}$, melhor é a qualidade do processamento $\mathcal{P}$, isto é, quando $RMSE(\mathcal{P}'(\mathcal{I}), \mathcal{I}) \leq RMSE(\mathcal{P}(\mathcal{I}), \mathcal{I})$ significa que o processamento $\mathcal{P}'$ é melhor que $\mathcal{P}$.

APÊNDICE C – Imagens produzidas pelos experimentos do Capítulo 5

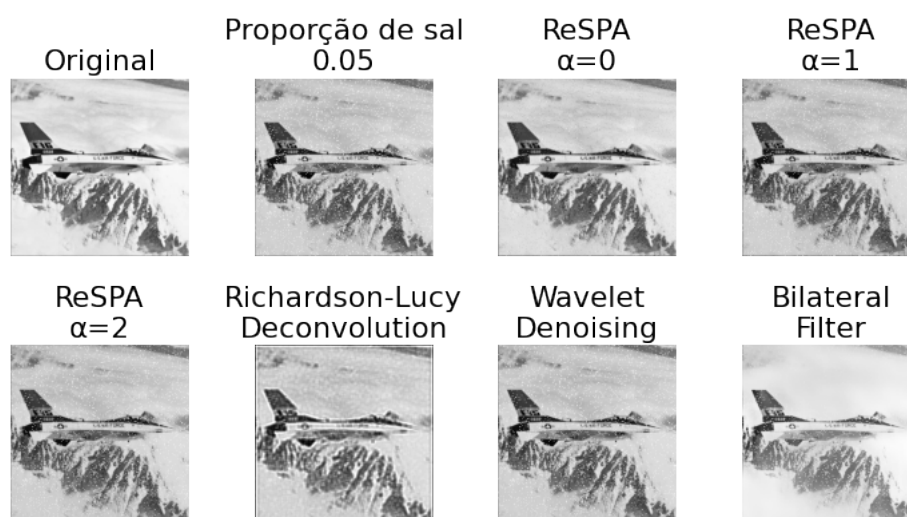


Figura 22 – Processamento da imagem Img1 com proporção de 0.05 de sal.

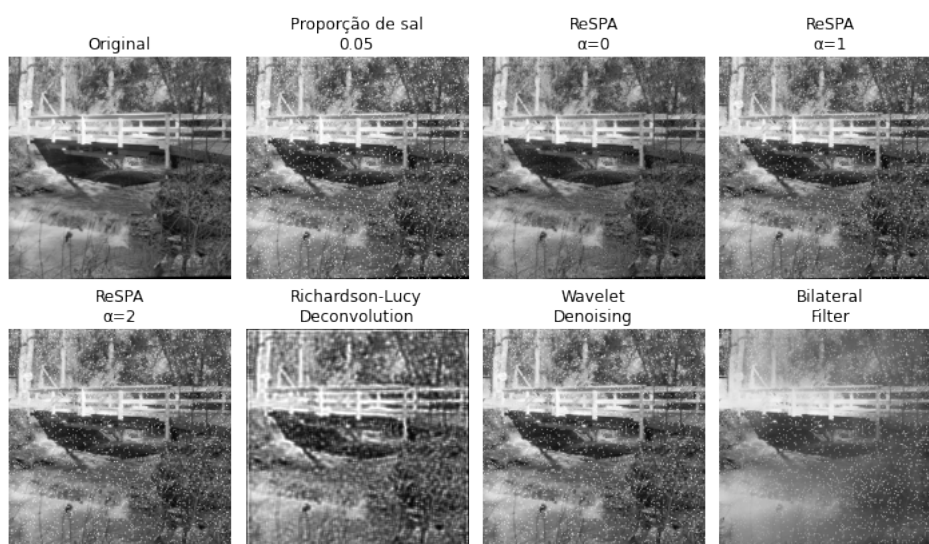


Figura 23 – Processamento da imagem Img2 com proporção de 0.05 de sal.

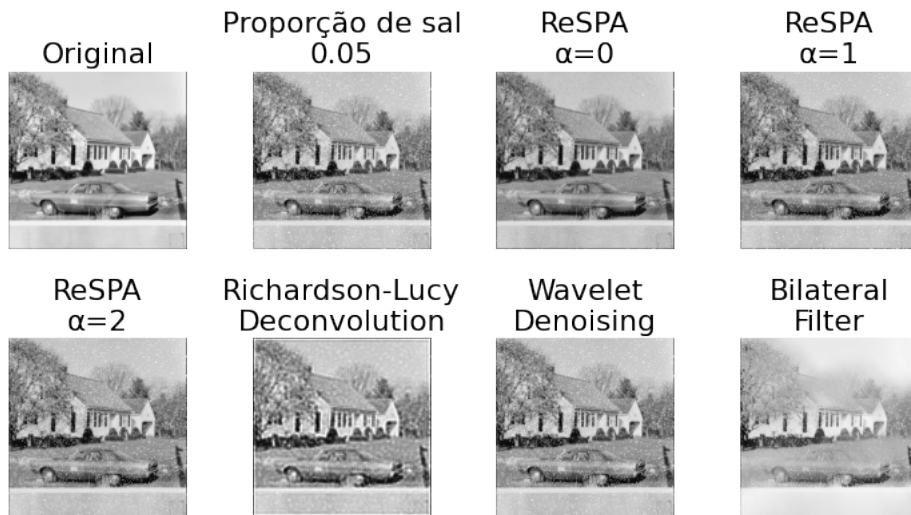


Figura 24 – Processamento da imagem Img3 com proporção de 0.05 de sal.



Figura 25 – Processamento da imagem Img4 com proporção de 0.05 de sal.

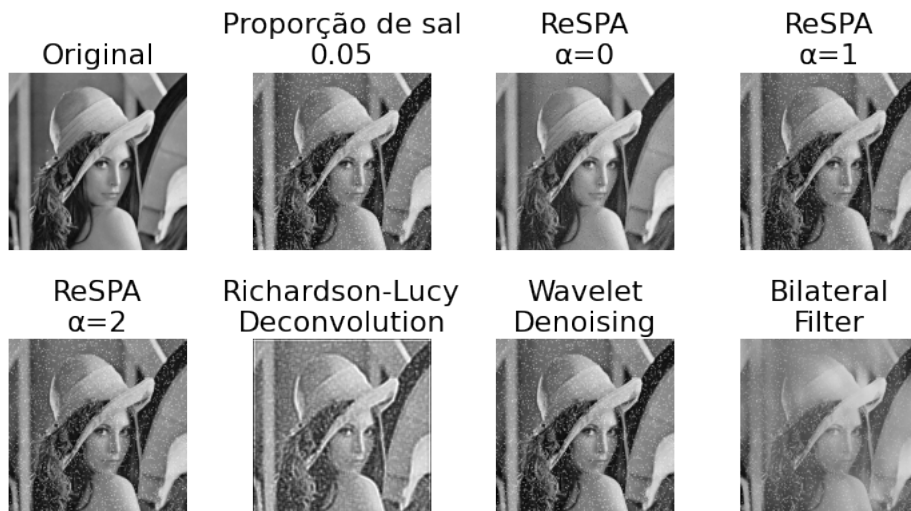


Figura 26 – Processamento da imagem Img5 com proporção de 0.05 de sal.



Figura 27 – Processamento da imagem Img6 com proporção de 0.05 de sal.

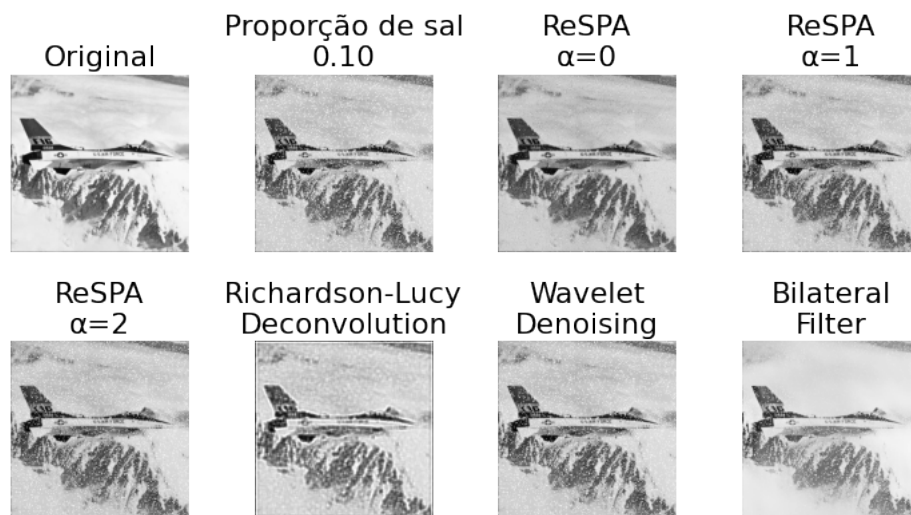


Figura 28 – Processamento da imagem Img1 com proporção de 0.10 de sal.

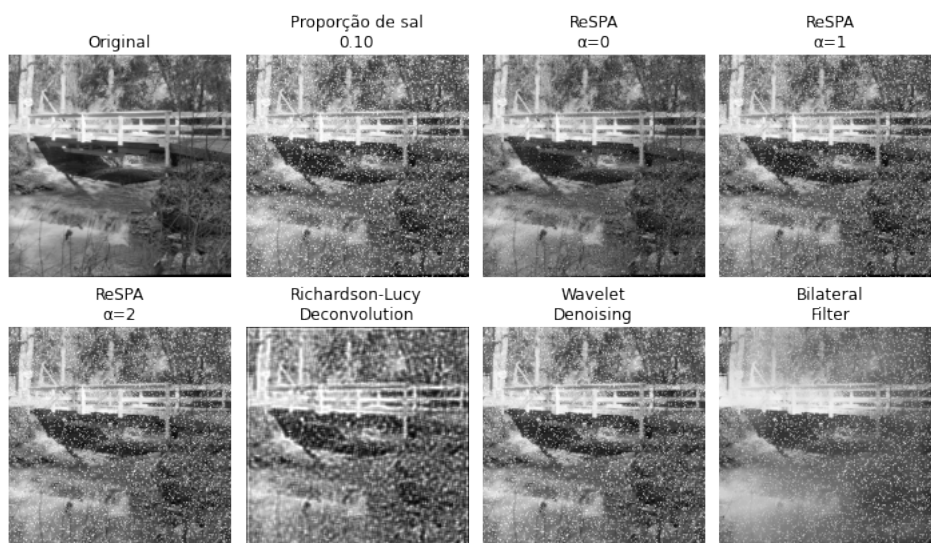


Figura 29 – Processamento da imagem Img2 com proporção de 0.10 de sal.

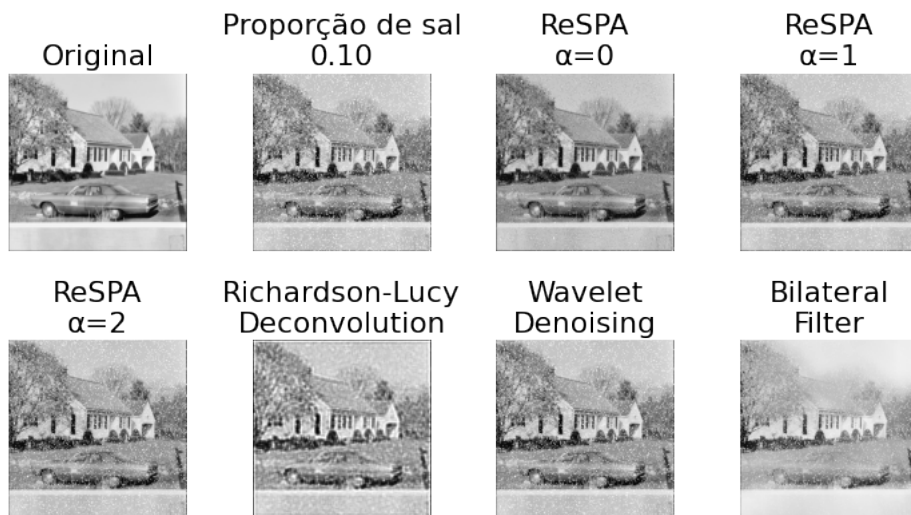


Figura 30 – Processamento da imagem Img3 com proporção de 0.10 de sal.



Figura 31 – Processamento da imagem Img4 com proporção de 0.10 de sal.

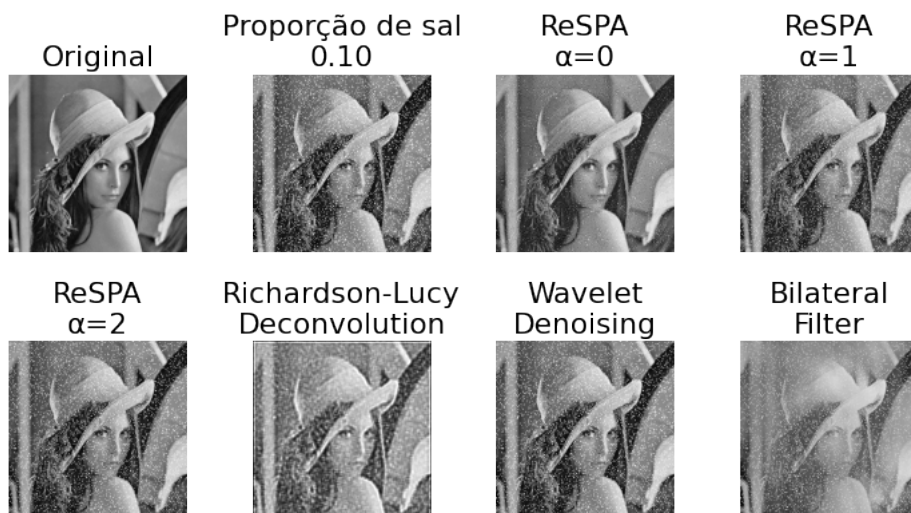


Figura 32 – Processamento da imagem Img5 com proporção de 0.10 de sal.

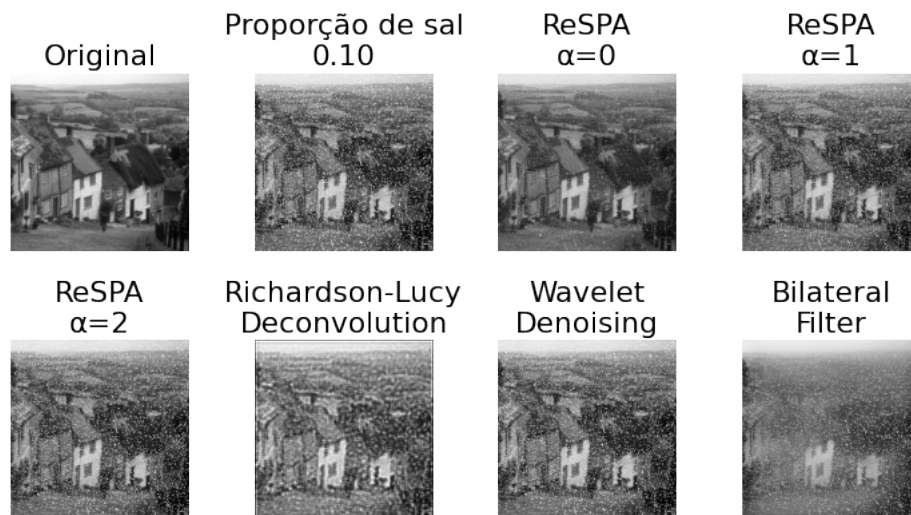


Figura 33 – Processamento da imagem Img6 com proporção de 0.10 de sal.

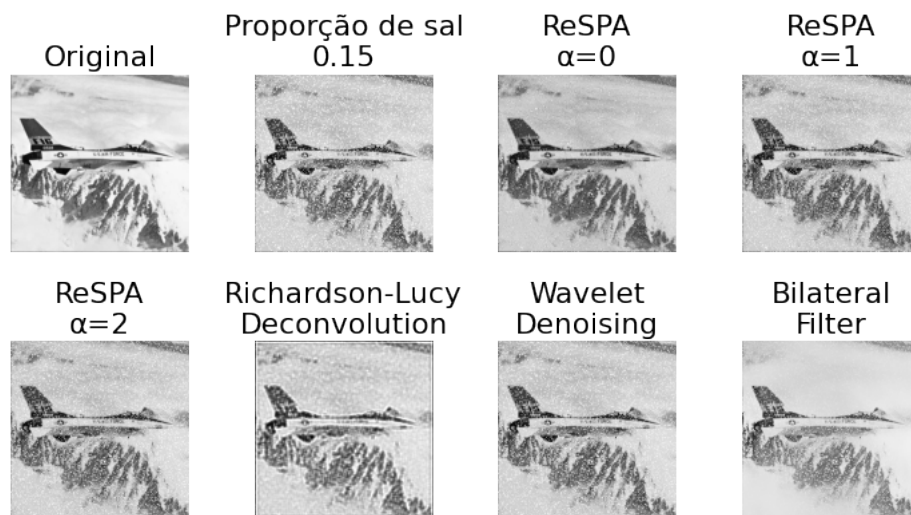


Figura 34 – Processamento da imagem Img1 com proporção de 0.15 de sal.

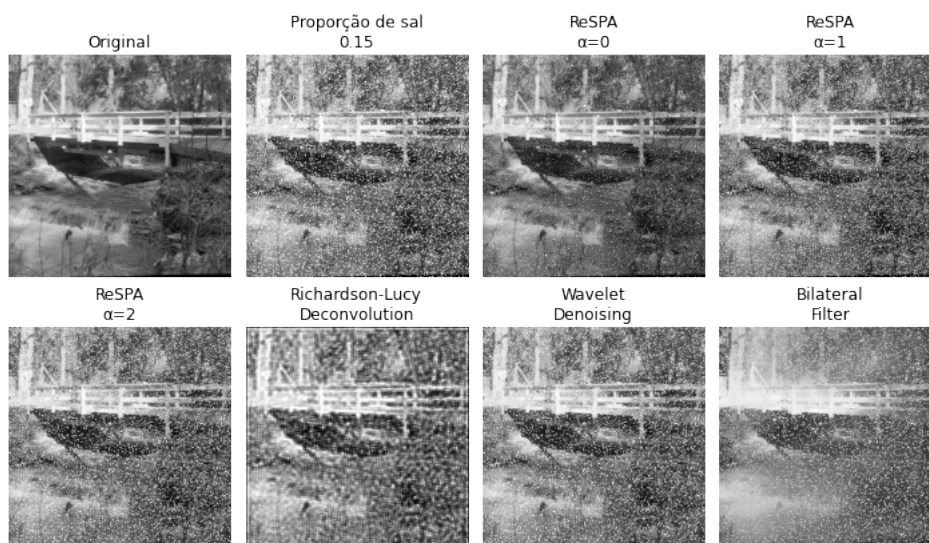


Figura 35 – Processamento da imagem Img2 com proporção de 0.15 de sal.

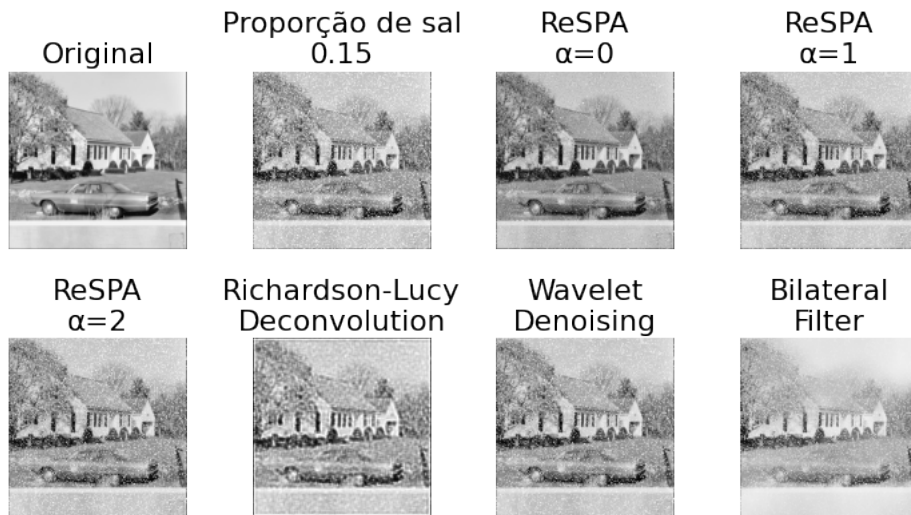


Figura 36 – Processamento da imagem Img3 com proporção de 0.15 de sal.



Figura 37 – Processamento da imagem Img4 com proporção de 0.15 de sal.

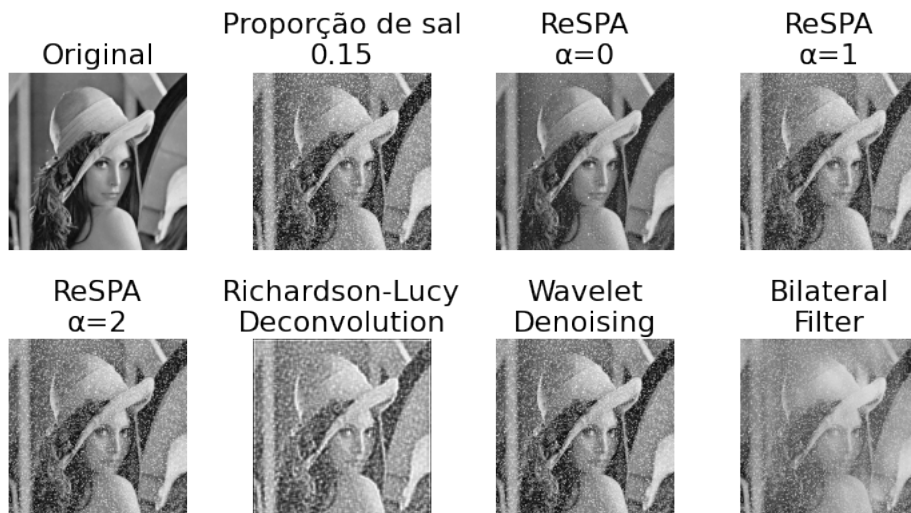


Figura 38 – Processamento da imagem Img5 com proporção de 0.15 de sal.

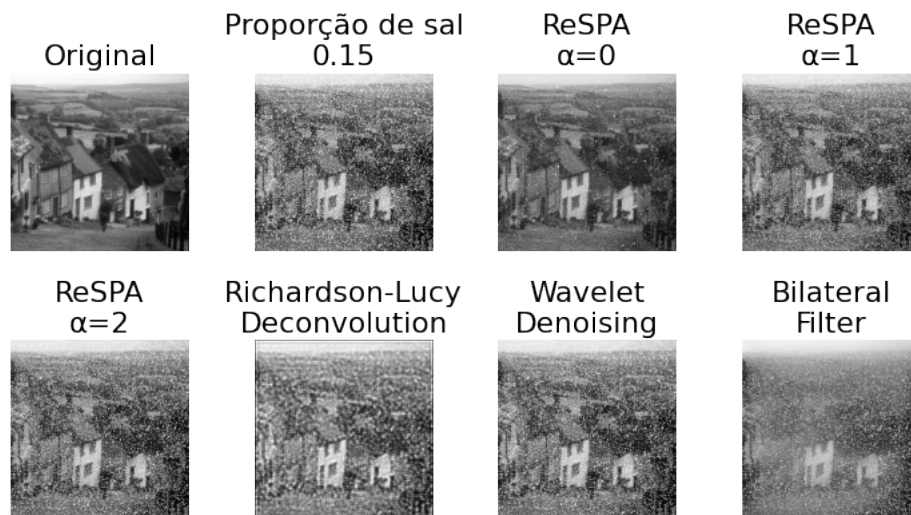


Figura 39 – Processamento da imagem Img6 com proporção de 0.15 de sal.

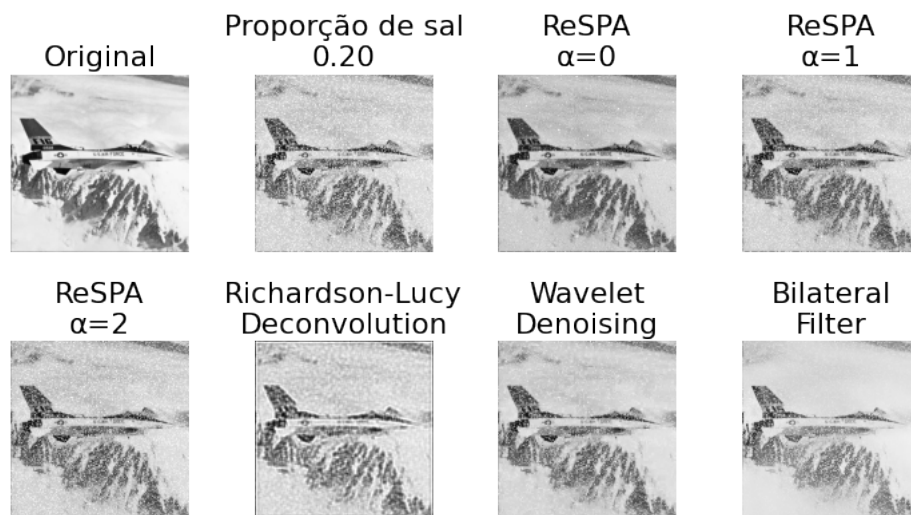


Figura 40 – Processamento da imagem Img1 com proporção de 0.20 de sal.

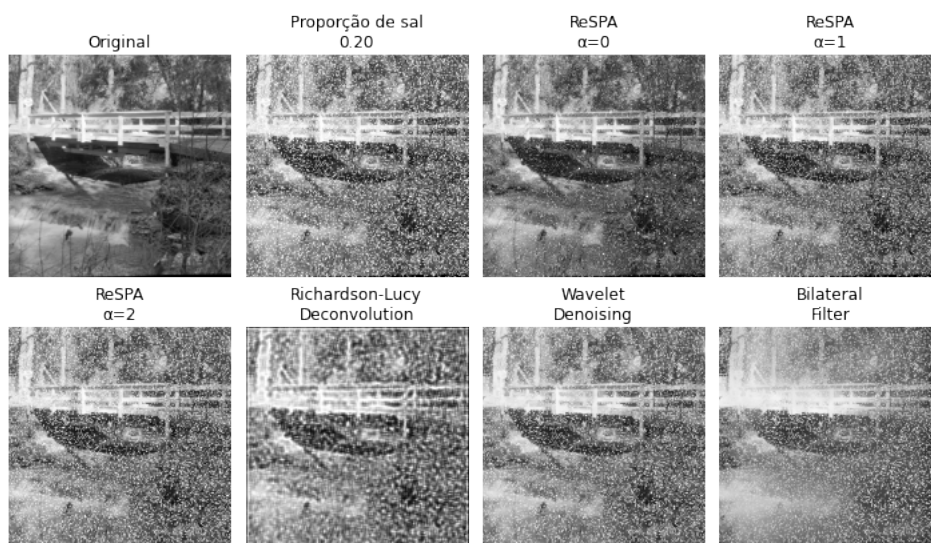


Figura 41 – Processamento da imagem Img2 com proporção de 0.20 de sal.

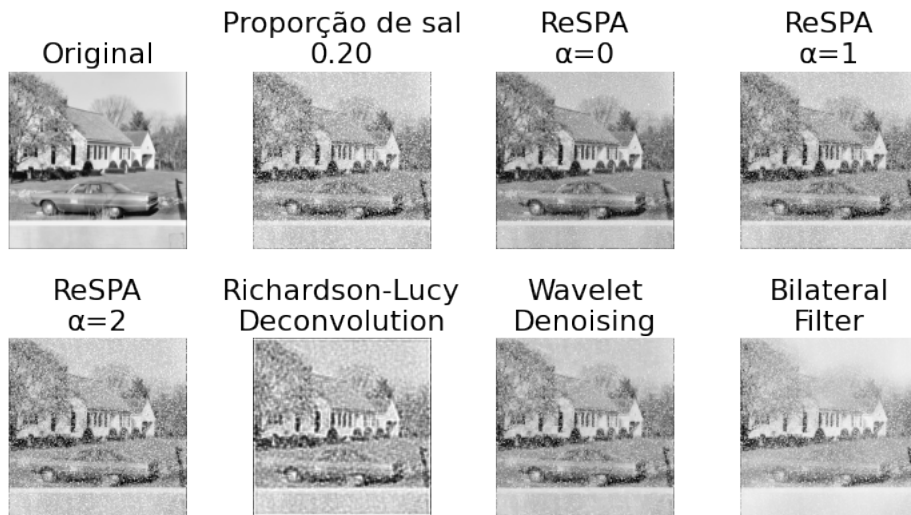


Figura 42 – Processamento da imagem Img3 com proporção de 0.20 de sal.



Figura 43 – Processamento da imagem Img4 com proporção de 0.20 de sal.

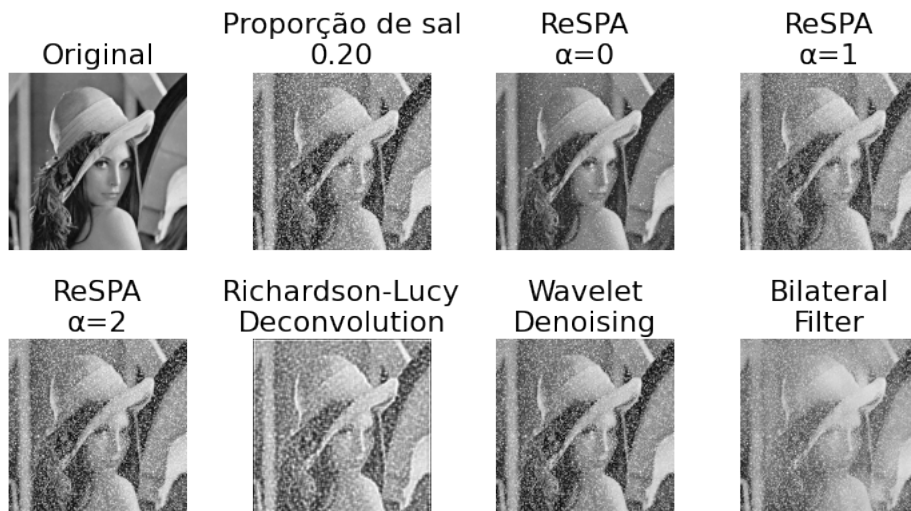


Figura 44 – Processamento da imagem Img5 com proporção de 0.20 de sal.

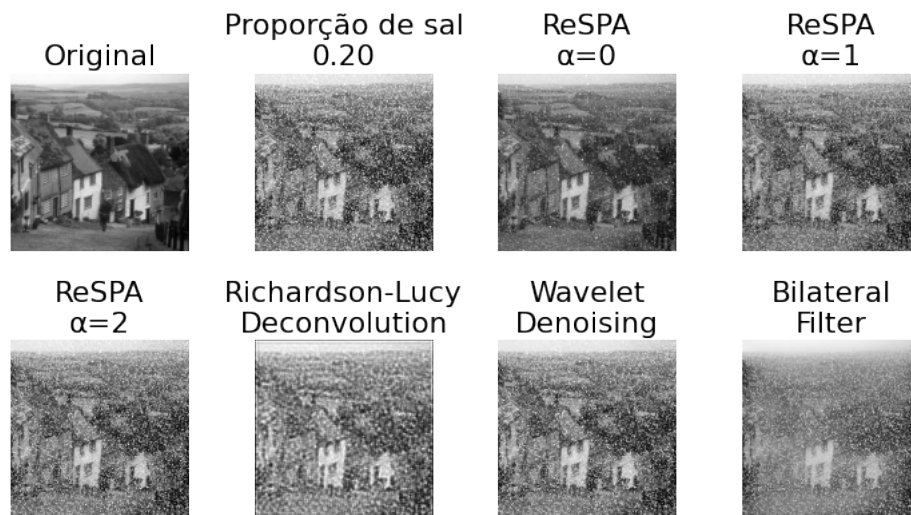


Figura 45 – Processamento da imagem Img6 com proporção de 0.20 de sal.

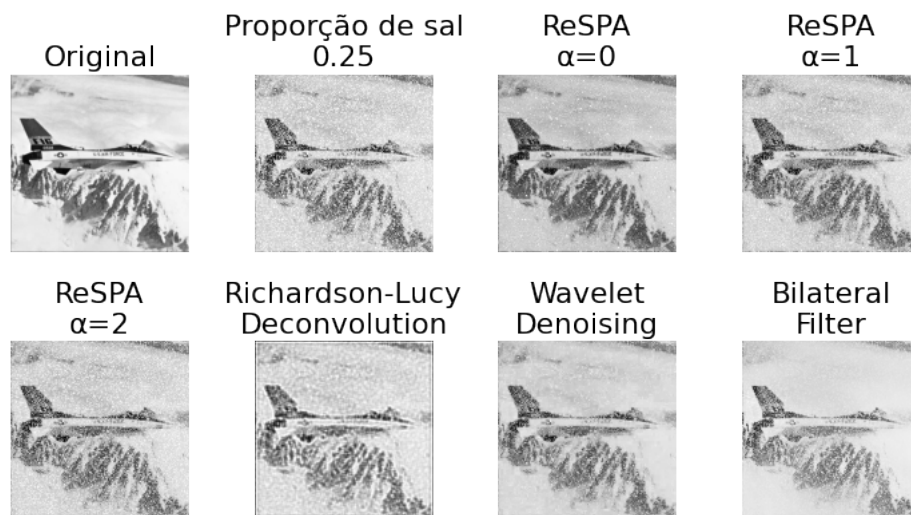


Figura 46 – Processamento da imagem Img1 com proporção de 0.25 de sal.

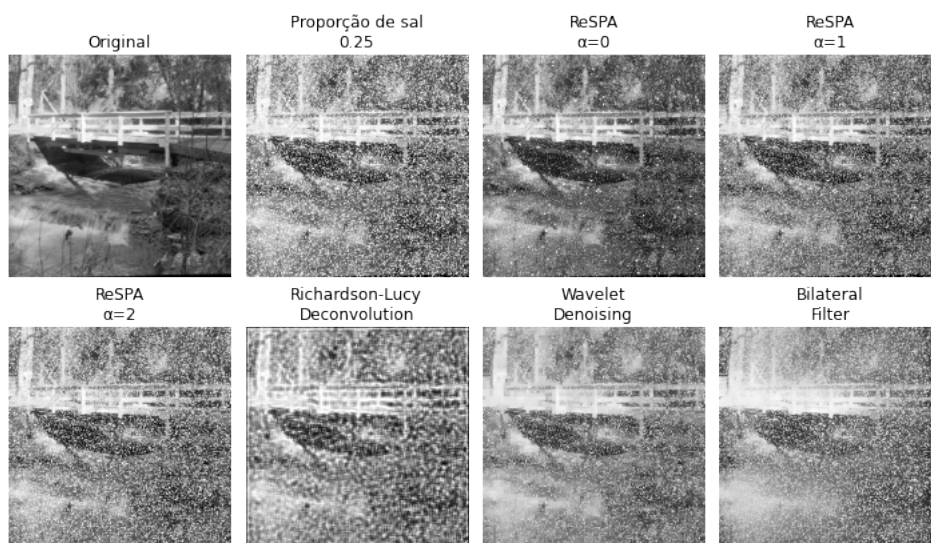


Figura 47 – Processamento da imagem Img2 com proporção de 0.25 de sal.

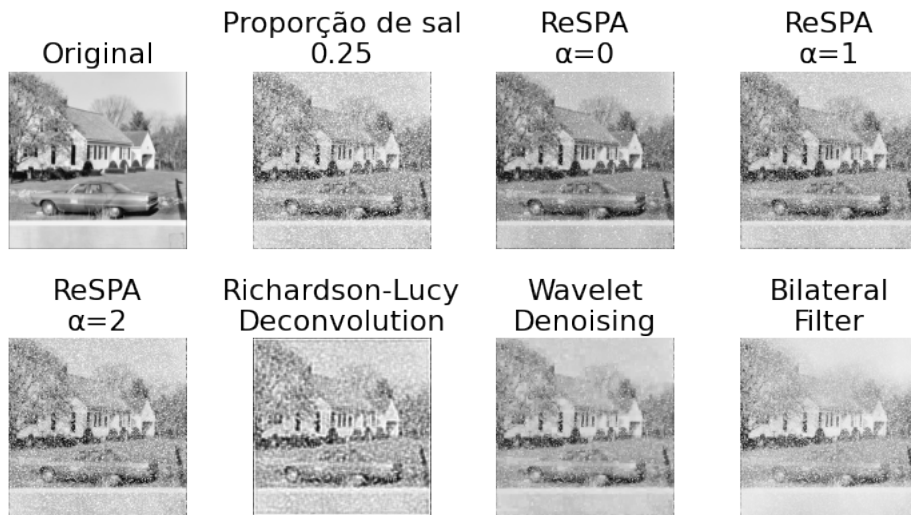


Figura 48 – Processamento da imagem Img3 com proporção de 0.25 de sal.



Figura 49 – Processamento da imagem Img4 com proporção de 0.25 de sal.

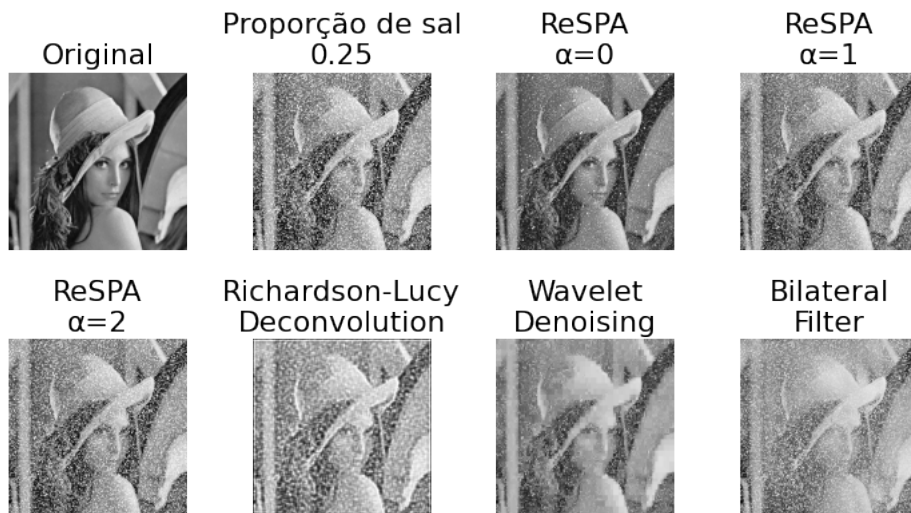


Figura 50 – Processamento da imagem Img5 com proporção de 0.25 de sal.

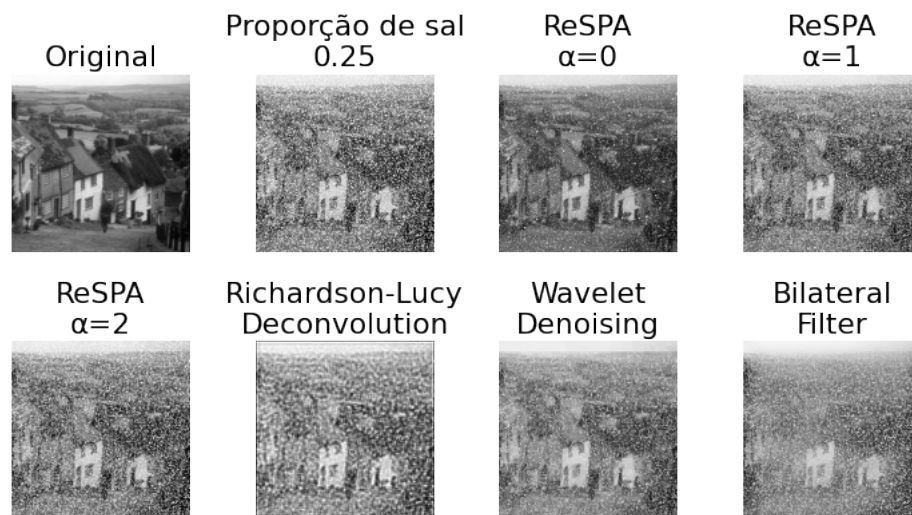


Figura 51 – Processamento da imagem Img6 com proporção de 0.25 de sal.

APÊNDICE D – Uma Formulação Algorítmica da Transformada *Wavelet* Discreta de Haar

A ideia de transformada *wavelet* foi inicialmente proposta pelo matemático húngaro Alfréd Haar no ano de 1909, entretanto, essa ideia só ficou conhecida por *wavelet* em anos recentes (LIMA, 2002). De forma similar a outros textos introdutórios sobre *wavelets* aqui será descrito os conceitos básicos sobre o processo algorítmico para o cálculo multi-resolução da transformada *wavelet* de Haar (VIDAKOVIC; MUELLER, 1994; STOLLNITZ et al., 1996), em sua versão discreta. Vale salientar que embora o foco seja algorítmico, aqui não será abandonado o formalismo e o rigor matemático.

Definição D.0.1. *Um sinal discreto S_n é um vetor de números reais de dimensão n tal que $n = 2^j$ par algum $j \in \mathbb{N}$. Os elementos de S_n , ou seja, os valores dos pulsos do sinal são representados como $S_{n,k}$ tal que $k = 0, \dots, n - 1$.*

Exemplo D.0.1. *O vetor $S_4 = (-2, 3, 5, 6)$ é um sinal e tem-se que $S_{4,0} = -2, S_{4,1} = 3, S_{4,2} = 5$ e $S_{4,3} = 6$.*

Obviamente existem diversas maneiras possíveis de se manipular os elementos de um sinal, isto é, há diversas formas de como os valores dos pulsos do sinal podem ser transformados (modificados) para outros valores, ou seja, gerando um novo sinal. Enquanto função, a assinatura da transformada de Haar pode ser descrita como se segue.

Definição D.0.2. $\mathbb{S}_{2^j}$ denota o conjunto de todos os sinais de dimensão 2^j , assim a transformada de Haar é uma função $\Psi : \mathbb{S}_{2^{j+1}} \rightarrow \mathbb{S}_{2^j} \times \mathbb{S}_{2^j}$, em que para todo sinal S tem-se que a primeira projeção de $\Psi(S)$ é chamada de informações médias ou coeficientes de escala, e a segunda projeção de $\Psi(S)$ é chamada de detalhes ou coeficientes *wavelets*.

Existem diversas formas para a construção da transformada de Haar (VIDAKOVIC; MUELLER, 1994), neste texto será considerada a seguinte forma (entre inúmeras possíveis) da transformada de Haar.

$$\Psi(S_{2^{j+1}}) = \left(E(S_{2^{j+1}}), W(S_{2^{j+1}}) \right) \quad (\text{D.1})$$

onde,

$$E(S_{2^{j+1}}) = (C_0, \dots, C_{2^j-1}) \quad (\text{D.2})$$

e

$$W(S_{2^{j+1}}) = (D_0, \dots, D_{2^j-1}) \quad (\text{D.3})$$

em que para todo $i = 0, \dots, 2^j - 1$,

$$C_i = \frac{S_{2^{j+1}, 2i} + S_{2^{j+1}, 2i+1}}{2} \quad (\text{D.4})$$

e

$$D_i = \frac{S_{2^{j+1}, 2i} - S_{2^{j+1}, 2i+1}}{2} \quad (\text{D.5})$$

Exemplo D.0.2. Considere o sinal $S_8 = (0, 1, 3, 1, 6, 8, 4, 2)$ tem-se que,

$$\Psi(S_8) = \left((0.5, 2, 7, 3), (-0.5, 1, -1, 1) \right)$$

Obviamente o processo de transformação de um sinal usando a transformada *wavelet* de Haar acima apresentado é reversível, isto é, sempre é possível obter o sinal original a partir das projeções de Ψ , pois basta notar que para todo i tem-se que, $C_i + D_i = S_{2^{j+1}, 2i}$ e $C_i - D_i = S_{2^{j+1}, 2i+1}$, ou seja, as coordenadas pares dos pulsos do sinal original são recuperadas pela soma das projeções e as coordenadas ímpares pela diferença das projeções.

Agora como dito em (STOLLNITZ et al., 1996), as *wavelets* podem ser usadas iterativamente para gerar k níveis de resolução, essa ação recebe o nome de multi-resolução e pode ser computada pelo algoritmo a seguir.

Algoritmo 9 Algoritmo para computar a multi-resolução da *wavelet* de Haar.

Entrada Um sinal S_{2^j} e um natural k tal que $j > 0$ e $k \leq j$.

Saída Uma lista de sinais $P_{S_{2^j}}$.

```

1:  $S \leftarrow S_{2^j}$ 
2:  $i \leftarrow 1$ 
3:  $P_S \leftarrow \emptyset$ 
4: enquanto  $i \leq k$  faça
5:    $\left( (C_0, \dots, C_{2^{j-1}}), (D_0, \dots, D_{2^{j-1}}) \right) \leftarrow \Psi(S)$ 
6:    $S \leftarrow (C_0, \dots, C_{2^{j-1}}, D_0, \dots, D_{2^{j-1}})$ 
7:   Adicione  $S$  ao final da lista  $P_{S_{2^j}}$ 
8:    $i \leftarrow i + 1$ 
9: fim enquanto
10: retorne  $P_{S_{2^j}}$ 
```

Exemplo D.0.3. Considere o sinal $S_8 = (1, 8, 2, 4, 0, 3, -2, 4)$ e $k = 2$ aplicando o Algoritmo 9 tem-se que para o primeiro nível de resolução $S = S_8$, o laço de resolução irá iniciar e gerar os seguintes coeficientes de escala e os seguintes coeficientes wavelets,

$$E(S) = (4.5, 3, 1.5, 1)$$

$$W(S) = (-3.5, -2, -1.5, 2)$$

em seguida o sinal S é atualizado de forma que $S = (4.5, 3, 1.5, 1, -3.5, -2, -1.5, 2)$ e é inserido na lista P_{S_8} , então laço continua e para o segundo nível de resolução são gerados os seguintes coeficientes de escala e os seguintes coeficientes wavelets,

$$E(S) = (3.75, 1.25, -2.75, 0.25)$$

$$W(S) = (0.75, 0.25, -0.75, -1.75)$$

novamente S será atualizado e inserido em P_{S_8} , após isso o laço termina e tem-se a seguinte lista como saída do algoritmo:

$$P_{S_8} = [(4.5, 3, 1.5, 1, -3.5, -2, -1.5, 2), (3.75, 1.25, -2.75, 0.25, 0.75, 0.25, -0.75, -1.75)]$$

Referências

- ABE, J. M.; PAPAVERO, N. *Teoria intuitiva dos conjuntos*. Rio de Janeiro: MAKRON Books, 1991. Citado 7 vezes nas páginas 39, 40, 41, 42, 47, 91 e 102.
- ASTRAIN, J. J.; MENDÍVIL, J. R. G. de; GARITAGOITIA, J. R. Fuzzy automata with ε -moves compute fuzzy measures between strings. *Fuzzy Sets and Systems*, v. 157, n. 11, p. 1550–1559, 2006. Citado 2 vezes nas páginas 56 e 61.
- ASVELD, P. R. Fuzzy context-free languages - Part 1: Generalized fuzzy context-free grammars. *Theoretical Computer Science*, v. 347, n. 1-2, p. 167–190, 2005. Citado na página 49.
- ASVELD, P. R. J. Algebraic aspects of families of fuzzy languages. *Theoretical Computer Science*, v. 293, n. 2, p. 417–445, 2003. Citado na página 49.
- ASVELD, P. R. J. Fuzzy context-free languages - Part 2: Recognition and parsing algorithms. *Theoretical Computer Science*, v. 347, n. 1-2, p. 191–213, 2005. Citado na página 49.
- ATANASSOV, K. T. New operations defined over the intuitionistic fuzzy sets. *Fuzzy sets and Systems*, v. 61, n. 2, p. 137–142, 1994. Citado 2 vezes nas páginas 27 e 43.
- BAETS, B. D. Idempotent uninorms. *European Journal of Operational Research*, v. 118, n. 3, p. 631–642, 1999. Citado na página 43.
- BAETS, B. D.; FODOR, J. Residual operators of uninorms. *Soft Computing*, v. 3, n. 2, p. 89–100, 1999. Citado na página 43.
- BAILADOR, G.; TRIVIÑO, G. Pattern recognition using temporal fuzzy automata. *Fuzzy Sets and Systems*, v. 161, n. 1, p. 37–55, 2010. Citado na página 65.
- BARROS, L. C.; BASSANEZI, R. C. *Tópicos de Lógica Fuzzy e Biomatemática, 2ª edição*. São Paulo: IMECC, Universidade Estadual de Campinas, 2015. Citado 2 vezes nas páginas 43 e 44.
- BEDREGAL, B. On interval fuzzy negations. *Fuzzy Sets and Systems*, v. 161, n. 17, p. 2290–2313, 2010. Citado na página 43.
- BEDREGAL, B. λ -ALN: autômatos lineares não-determinísticos com λ -transições. *Tendências em Matemática Aplicada e Computacional*, SciELO Brasil, v. 12, n. 3, p. 171–182, 2011. Citado 2 vezes nas páginas 39 e 60.
- BEDREGAL, B. Nondeterministic Linear Automata and a Class of Deterministic Linear Languages. *Preliminary Proceedings LSFA*, p. 183–196, 2015. Citado 2 vezes nas páginas 39 e 60.
- BEDREGAL, B.; ACIÓLY, B. M.; LYRA, A. *Introdução à Teoria da Computação: Linguagens Formais, Autômatos e Computabilidade*. Natal: Editora UnP, 2010. Citado 6 vezes nas páginas 33, 35, 36, 37, 38 e 39.

- BEDREGAL, B. et al. A class of fuzzy multisets with a fixed number of memberships. *Information Sciences*, v. 189, p. 1–17, 2012. Citado na página 27.
- BEDREGAL, B. et al. New results on overlap and grouping functions. *Information Sciences*, v. 249, p. 148–170, 2013. Citado na página 43.
- BEDREGAL, B. et al. Aggregation functions for typical hesitant fuzzy elements and the action of automorphisms. *Information Sciences*, v. 255, p. 82–99, 2014. Citado 3 vezes nas páginas 27, 46 e 47.
- BEDREGAL, B. et al. Typical hesitant fuzzy negations. *International Journal of Intelligent Systems*, v. 29, n. 6, p. 525–543, 2014. Citado 2 vezes nas páginas 45 e 48.
- BEDREGAL, B. C.; FIGUEIRA, S. On the computing power of fuzzy Turing machines. *Fuzzy Sets and Systems*, v. 159, n. 9, p. 1072–1083, 2008. Citado na página 63.
- BEDREGAL, B. R. C.; FIGUEIRA, S. Classical Computability and Fuzzy Turing Machines. *LATIN 2006: Theoretical Informatics*, v. 3887, p. 154–165, 2006. Disponível em: <http://link.springer.com/10.1007/11682462>. Citado na página 63.
- BEDREGAL, B. R. C.; TAKAHASHI, A. The best interval representations of t-norms and automorphisms. *Fuzzy Sets and Systems*, v. 157, n. 24, p. 3220–3230, 2006. Citado na página 59.
- BĚLOHLÁVEK, R. Determinism and fuzzy automata. *Information Sciences*, v. 143, n. 1-4, p. 205–209, 2002. Citado 3 vezes nas páginas 51, 52 e 56.
- BIRD, M. The equivalence problem for deterministic two-tape automata. *Journal of Computer and System Sciences*, v. 7, n. 2, p. 218–236, 1973. Citado na página 39.
- BOOLOS, G. S.; BURGESS, J. P.; JEFFREY, R. C. *Computabilidade e Lógica*. São Paulo: Fundação editora da UNESP, 2012. Citado na página 140.
- BÜCHI, J. R. Weak second-order arithmetic and finite automata. *Mathematical Logic Quarterly*, v. 6, n. 1-6, p. 66–92, 1960. Citado na página 35.
- BUSTINCE, H. et al. A historical account of types of fuzzy sets and their relationships. *IEEE Transactions on Fuzzy Systems*, vol. 24, no. 1, February 2017, p. 179–194, 2016. Citado 5 vezes nas páginas 27, 43, 59, 62 e 98.
- CAO, Y.; EZAWA, Y. Nondeterministic fuzzy automata. *Information Sciences*, v. 191, p. 86–97, 2012. Citado na página 57.
- CARMO, J.; GOUVEIA, P.; DIONÍSIO, F. M. *Elementos de Matemática Discreta*. London: College Publications, 2013. Citado 3 vezes nas páginas 60, 139 e 142.
- CHANG, S. G.; YU, B.; VETTERLI, M. Adaptive wavelet thresholding for image denoising and compression. *IEEE Transactions on Image Processing*, v. 9, n. 9, p. 1532–1546, 2000. Citado na página 118.
- CHAUDHARI, S. R.; KOMEJWAR, D. D. On Fuzzy Regular Grammars. *Advances in Fuzzy Mathematics*, v. 6, n. 1, p. 89–103, 2011. Citado 4 vezes nas páginas 28, 49, 51 e 52.

- CHOUBEY, A.; RAVI, K. Intuitionistic fuzzy automata and intuitionistic fuzzy regular expressions. *J. Appl. Math. & Informatics*, v. 27, n. 1-2, p. 409–417, 2009. Citado 2 vezes nas páginas 62 e 65.
- CHURCH, A. Logic, arithmetic and automata. In: IMU. *Proceedings of the international congress of mathematicians*. Djursholm, Swede, 1962. p. 23–35. Citado na página 35.
- COSTA, C. G. da. *Probabilidades Imprecisas: Intervalar, fuzzy e fuzzy intuicionista*. Tese (Doutorado) — Programa de Pós-graduação em Engenharia Elétrica e da Computação, Universidade Federal do Rio Grande do Norte, UFRN, Natal, RN, 2012. Citado na página 55.
- COSTA, V. S. *Linguagens Lineares Fuzzy*. Dissertação (Mestrado) — Programa de Pós-graduação em Sistemas e Computação, Universidade Federal do Rio Grande do Norte, UFRN, Natal, RN, 2016. Citado 8 vezes nas páginas 15, 27, 34, 39, 44, 49, 50 e 51.
- COSTA, V. S.; BEDREGAL, B. Fuzzy linear automata and some equivalences. *Tendências em Matemática Aplicada e Computacional*, SciELO Brasil, v. 19, n. 1, p. 127–145, 2018. Citado 4 vezes nas páginas 39, 42, 43 e 51.
- COSTA, V. S.; BEDREGAL, B. On Typical Hesitant Fuzzy Automata. *Soft Computing*, v. 24, n. 12, p. 8725–8736, 2020. Citado 10 vezes nas páginas 27, 28, 34, 51, 52, 53, 58, 59, 65 e 78.
- COSTA, V. S.; BEDREGAL, B.; SANTIAGO, R. H. N. On closure properties of $\mathcal{L}$ -valued linear languages. *Fuzzy Sets and Systems*, 2020. DOI:10.1016/j.fss.2020.09.012. Citado 2 vezes nas páginas 63 e 78.
- COSTA, V. S. et al. Combining multiple algorithms in classifier ensembles using generalized mixture functions. *Neurocomputing*, v. 313, p. 402–414, 2018. Citado 2 vezes nas páginas 129 e 130.
- CULIK, K.; FRIŠ, I. Weighted finite transducers in image processing. *Discrete Applied Mathematics*, v. 58, n. 3, p. 223–237, 1995. Citado 2 vezes nas páginas 107 e 134.
- CULIK, K.; KARI, J. Image compression using weighted finite automata. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, v. 711 LNCS, p. 392–402, 1993. Citado 2 vezes nas páginas 107 e 134.
- CULIK, K.; VALENTA, V. Finite automata based compression of bi-level and simple color images. *Computers and Graphics (Pergamon)*, v. 21, n. 1, p. 61–68, 1997. Citado 2 vezes nas páginas 107 e 134.
- DAVEY, B. A.; PRIESTLEY, H. A. *Introduction to lattices and order*. Cambridge: Cambridge University Press, 2002. Citado na página 39.
- DHURE, S. S. *A Study of Fuzzy Automata Using Techniques in Fuzzy Groups*. Tese (Doutorado) — Shivaji University, Kolhapur, 2014. Citado na página 52.
- DIMURO, G. P. et al. The law of o-conditionality for fuzzy implications constructed from overlap and grouping functions. *International Journal of Approximate Reasoning*, v. 105, p. 27–48, 2019. Citado na página 43.

DINIZ, P. S.; SILVA, E. A. B. da; NETTO, S. L. *Processamento Digital de Sinais: Projeto e Análise de Sistemas*. Porto Alegre: Bookman Editora, 2014. Citado 2 vezes nas páginas 70 e 72.

DONOHO, D. L.; JOHNSTONE, J. M. Ideal spatial adaptation by wavelet shrinkage. *biometrika*, v. 81, n. 3, p. 425–455, 1994. Citado na página 118.

FACELI, K. et al. *Inteligência Artificial-uma abordagem de aprendizado de máquina*. Rio de Janeiro: LTC, 2011. Citado 5 vezes nas páginas 17, 121, 125, 129 e 130.

FARHADINIA, B. A novel method of ranking hesitant fuzzy values for multiple attribute decision-making problems. *International Journal of Intelligent Systems*, v. 28, n. 8, p. 752–767, 2013. Citado na página 46.

FARIAS, A. et al. The image reduction process based on generalized mixture functions. In: IEEE. *Fuzzy Information Processing Society (NAFIPS), 2016 Annual Conference of the North American*. El Paso, TX, USA, 2016. p. 1–6. Citado 2 vezes nas páginas 114 e 147.

FARIAS, A. D. S. et al. A residuated function in a class of Mealy type $\mathcal{L}$ -Valued finite automaton. In: IEEE. *Fuzzy Information Processing Society (NAFIPS), 2016 Annual Conference of the North American*. El Paso, TX, USA, 2016. p. 1–6. Citado 5 vezes nas páginas 33, 39, 41, 52 e 62.

FARIAS, A. D. S. et al. Closure properties for fuzzy recursively enumerable languages and fuzzy recursive languages. *Journal of Intelligence & Fuzzy Systems*, v. 31, n. 3, p. 1795–1806, 2016. Citado 3 vezes nas páginas 27, 52 e 63.

FISH, D. et al. Blind deconvolution by means of the Richardson-Lucy algorithm. *JOSA A*, v. 12, n. 1, p. 58–65, 1995. Citado na página 118.

FODOR, J. C. On fuzzy implication operators. *Fuzzy sets and Systems*, v. 42, n. 3, p. 293–300, 1991. Citado na página 43.

GARMENDIA, L.; CAMPO, R. G. del; RECASENS, J. Partial orderings for hesitant fuzzy sets. *International Journal of Approximate Reasoning*, v. 84, p. 159–167, 2017. Citado 3 vezes nas páginas 46, 104 e 133.

GIRIJAMMA, A. *Studies in Fuzzy Automata*. Tese (Doutorado) — Kuvempu University, Shankaraghatta, 2010. Citado 3 vezes nas páginas 28, 63 e 64.

GOGUEN, J. A. L-fuzzy sets. *Journal of Mathematical Analysis and Applications*, v. 18, n. 1, p. 145–174, 1967. Citado 3 vezes nas páginas 44, 47 e 62.

GONZALEZ, R. C.; WOODS, R. C. *Processamento digital de imagens*. Rio de Janeiro: Pearson, 2009. Citado 7 vezes nas páginas 70, 72, 114, 116, 124, 134 e 147.

GRABISCH, M. et al. *Aggregation functions*. New York: Cambridge University Press, 2009. v. 127. Citado na página 43.

GRZEGORZEWSKI, P. Distances between intuitionistic fuzzy sets and/or interval-valued fuzzy sets based on the Hausdorff metric. *Fuzzy Sets and Systems*, v. 148, n. 2, p. 319–328, 2004. Citado na página 45.

- GUPTA, H. V.; KLING, H. On typical range, sensitivity, and normalization of Mean Squared Error and Nash-Sutcliffe Efficiency type metrics. *Water Resources Research*, v. 47, n. 10, 2011. Citado na página 147.
- HAFNER, U. Refining image compression with weighted finite automata. In: IEEE. *Proceedings of Data Compression Conference-DCC'96*. Snowbird, Utah, USA, 1996. p. 359–368. Citado na página 107.
- HARJU, T.; KARHUMÄKI, J. Decidability of the multiplicity equivalence of multitape finite automata. In: ACM. *Proceedings of the twenty-second annual ACM symposium on Theory of computing*. New York, NY (USA), 1990. p. 477–481. Citado na página 39.
- HARJU, T.; KARHUMÄKI, J. The equivalence problem of multitape finite automata. *Theoretical Computer Science*, v. 78, n. 2, p. 347–355, 1991. Citado na página 39.
- HAYKIN, S. *Redes neurais: princípios e prática*. Porto Alegre: Bookman Editora, 2007. Citado na página 121.
- HIGUERA, C. De la. *Grammatical inference: learning automata and grammars*. London: Cambridge University Press, 2010. Citado 7 vezes nas páginas 34, 121, 122, 125, 126, 127 e 131.
- HOPCROFT, J. E.; MOTWANI, R.; ULLMAN, J. D. *Introduction to Automata Theory, Languages and Computation*. 3ª. ed. USA: Pearson Education India, 2008. Citado 6 vezes nas páginas 33, 34, 35, 38, 47 e 142.
- JIANG, Z.; LITOW, B.; VEL, O. de. Similarity enrichment in image compression through weighted finite automata. In: SPRINGER. *International Computing and Combinatorics Conference*. Sydney, Australia, 2000. p. 447–456. Citado na página 107.
- KATRITZKE, F.; MERZENICH, W.; THOMAS, M. Enhancements of partitioning techniques for image compression using weighted finite automata. In: SPRINGER. *International Conference on Implementation and Application of Automata*. Pretoria, South Africa, 2001. p. 177–189. Citado na página 107.
- KLEENE, S. C. *Representation of events in nerve nets and finite automata*. [S.l.], 1951. Citado na página 33.
- KLEMENT, E. P.; MESIAR, R.; PAP, E. *Triangular norms*. Germany: Springer Science & Business Media, 2013. v. 8. Citado na página 43.
- KO, B. C.; HAM, S. J.; NAM, J. Y. Modeling and formalization of fuzzy finite automata for detection of irregular fire flames. *IEEE Transactions on Circuits and Systems for Video Technology*, v. 21, n. 12, p. 1903–1912, 2011. Citado 2 vezes nas páginas 28 e 64.
- KOBAYASHI, K.; SEKIGUCHI, S. On the class of predicates decidable by two-way multitape finite automata. *Journal of the ACM (JACM)*, v. 13, n. 2, p. 236–261, 1966. Citado na página 39.
- KOMEJWAR, D. D. *A Study of Some Aspects of Fuzzy Automata, Fuzzy Grammars and Languages*. Tese (Doutorado) — Shivaji University, Kolhapur, 2012. Citado 6 vezes nas páginas 49, 50, 55, 56, 57 e 65.

- LEE, E. T.; ZADEH, L. A. Note on fuzzy languages. *Information Sciences*, v. 1, p. 421–434, 1969. Citado 4 vezes nas páginas 49, 56, 65 e 91.
- LI, Y. Some results of fuzzy Turing machines. *Proceedings of the World Congress on Intelligent Control and Automation (WCICA)*, v. 1, p. 3406–3409, 2006. Citado na página 63.
- LI, Y. Lattice-valued fuzzy Turing machines: Computing power, universality and efficiency. *Fuzzy Sets and Systems*, v. 160, n. 23, p. 3453–3474, 2009. Citado na página 63.
- LI, Y.; PEDRYCZ, W. The equivalence between fuzzy Mealy and fuzzy Moore machines. *Soft Computing*, v. 10, n. 10, p. 953–959, 2006. Citado na página 63.
- LIMA, A. A. D.; BEDREGAL, B.; MEZZOMO, I. Ordinal sums of the main classes of fuzzy negations and the natural negations of t-norms, t-conorms and fuzzy implications. *International Journal of Approximate Reasoning*, v. 116, p. 19–32, 2020. Citado na página 43.
- LIMA, A. A. D. et al. Multidimensional fuzzy sets. *IEEE Transactions on Fuzzy Systems*, 2020. DOI:10.1109/TFUZZ.2020.2994997. Citado 2 vezes nas páginas 27 e 43.
- LIMA, A. A. de. *Conjuntos Fuzzy Multidimensionais*. Tese (Doutorado) — Programa de Pós-graduação em Sistemas e Computação, Universidade Federal do Rio Grande do Norte, UFRN, Natal, RN, 2019. Citado 4 vezes nas páginas 43, 63, 65 e 98.
- LIMA, P. C. de. *Wavelets: uma introdução*. *Matemática Universitária*, n. 33, p. 13–44, 2002. Citado na página 161.
- LINZ, P. *An introduction to formal languages and automata*. USA: Jones & Bartlett Learning, 2006. Citado 4 vezes nas páginas 33, 34, 36 e 38.
- LUCY, L. B. An iterative technique for the rectification of observed distributions. *The Astronomical Journal*, v. 79, p. 745, 1974. Citado na página 118.
- MACIEL, A. *Aplicação de autômatos finitos nebulosos no reconhecimento aproximado de cadeias*. Dissertação (Mestrado) — Universidade de São Paulo, São Paulo, SP, 2005. Citado 3 vezes nas páginas 50, 56 e 64.
- MALIK, D. S.; MORDESON, J. N. On fuzzy regular languages. *Journal of Combinatorial Mathematics and Combinatorial Computing*, v. 88, p. 263–273, 1996. Citado 3 vezes nas páginas 51, 56 e 65.
- MARTIN, J. C. *Introduction to Languages and the Theory of Computation*. 4^a. ed. New York: McGraw-Hill NY, 1991. Citado 2 vezes nas páginas 35 e 60.
- MATZENAUER, M. L. et al. Typical hesitant fuzzy sets: Evaluating strategies in gdm applying consensus measures. In: ATLANTIS PRESS. *2019 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (EUSFLAT 2019)*. Prague, Czech Republic, 2019. Citado na página 47.
- MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, v. 5, n. 4, p. 115–133, 1943. Citado na página 33.

- MEALY, G. H. A method for synthesizing sequential circuits. *The Bell System Technical Journal*, v. 34, n. 5, p. 1045–1079, 1955. Citado 2 vezes nas páginas 33 e 39.
- MENDEL, J. M.; JOHN, R. B. Type-2 fuzzy sets made simple. *IEEE Transactions on Fuzzy Systems*, v. 10, n. 2, p. 117–127, 2002. Citado 2 vezes nas páginas 27 e 43.
- MENDÍVIL, J. R. G. D.; GARITAGOITIA, J. R. A comment on “construction of fuzzy automata from fuzzy regular expressions”. *Fuzzy Sets and Systems*, Elsevier B.V., v. 262, p. 102–110, 2015. Citado 2 vezes nas páginas 52 e 75.
- MENEZES, P. B. *Linguagens formais e autômatos*. Porto Alegre: Sagra-Luzzato, 1998. Citado na página 36.
- MEZZOMO, I. *On Fuzzy Ideals and Fuzzy Filters of Fuzzy Lattices*. Tese (Doutorado) — Programa de Pós-graduação em Sistemas e Computação, Universidade Federal do Rio Grande do Norte, UFRN, Natal, RN, 2013. Citado 4 vezes nas páginas 39, 40, 43 e 47.
- MIYAMOTO, S. Multisets and fuzzy multisets. In: *Soft computing and human-centered machines*. New York: Springer, 2000. p. 9–33. Citado 2 vezes nas páginas 27 e 43.
- MOORE, E. F. Gedanken-experiments on sequential machines. *Automata Studies*, v. 34, p. 129–153, 1956. Citado 2 vezes nas páginas 33 e 39.
- MORDESON, J. N.; MALIK, D. S. *Fuzzy automata and languages: theory and applications*. New York: CRC Press, 2002. Citado 8 vezes nas páginas 27, 49, 52, 55, 56, 57, 60 e 65.
- NIELSEN, M. A.; CHUANG, I. *Computação Quântica e Informação Quântica*. Porto Alegre, RS: Bookman, 2005. Citado na página 27.
- PAIVA, R. E. B. *Uma extensão de overlaps e naBL-Álgebras para reticulados*. Tese (Doutorado) — Programa de Pós-graduação em Sistemas e Computação, Universidade Federal do Rio Grande do Norte, UFRN, Natal, RN, 2019. Citado na página 40.
- PAZ, A. *Introduction to probabilistic automata*. New York: Academic Press, 2014. Citado na página 27.
- PEI, Z.; YI, L. A note on operations of hesitant fuzzy sets. *International Journal of Computational Intelligence Systems*, v. 8, n. 2, p. 226–239, 2015. Citado na página 46.
- PEKALA, B. *Uncertainty Data in Interval-Valued Fuzzy Set Theory: Properties, Algorithms and Applications*. London: Springer, 2018. v. 367. Citado 2 vezes nas páginas 45 e 59.
- PELLETIER, M.; SAKAROVITCH, J. On the representation of finite deterministic 2-tape automata. *Theoretical Computer Science*, v. 225, n. 1-2, p. 1–63, 1999. Citado na página 39.
- PINHEIRO, J. et al. A study of (T, N)-implications and its use to construct a new class of fuzzy subethood measure. *International Journal of Approximate Reasoning*, v. 97, p. 1–16, 2018. Citado na página 43.
- PITOMBEIRA, J. B. Lectures on algebraic topology. *Boletim da Sociedade Brasileira de Matemática*, v. 2, n. 1, p. 63–66, 1971. Citado na página 125.

QIU, D. Automata theory based on complete residuated lattice-valued logic. *Science in China (Series F)*, v. 44, n. 6, p. 410–429, 2001. Citado 7 vezes nas páginas [44](#), [52](#), [62](#), [65](#), [78](#), [104](#) e [133](#).

QIU, D. Characterizations of fuzzy finite automata. *Fuzzy Sets and Systems*, v. 141, n. 3, p. 391–414, 2004. Citado na página [52](#).

QIU, D. Pumping lemma in automata theory based on complete residuated lattice-valued logic: A note. *Fuzzy Sets and Systems*, v. 157, n. 15, p. 2128–2138, 2006. Citado na página [63](#).

QIU, D. Automata theory based on quantum logic: Reversibilities and pushdown automata. *Theoretical Computer Science*, v. 386, n. 1-2, p. 38–56, 2007. Citado 2 vezes nas páginas [62](#) e [63](#).

RABIN, M. O. Probabilistic automata. *Information and Control*, v. 6, n. 3, p. 230–245, 1963. Citado 3 vezes nas páginas [33](#), [35](#) e [97](#).

RABIN, M. O.; SCOTT, D. Finite automata and their decision problems. *IBM Journal of Research and Development*, v. 3, n. 2, p. 114–125, 1959. Citado 2 vezes nas páginas [33](#) e [38](#).

RAVI, K.; ALKA, C. Interval-valued fuzzy regular language. *Journal of Applied Mathematics & Informatics*, v. 28, n. 3_4, p. 639–649, 2010. Citado 3 vezes nas páginas [59](#), [60](#) e [65](#).

RAVI, K.; CHOUBEY, A. Myhill-Nerode Theorem for Interval-valued Fuzzy Regular Language. In: AMERICAN INSTITUTE OF PHYSICS. *AIP Conference Proceedings*. Chandigarh, 2010. v. 1324, n. 1, p. 30–33. Citado 2 vezes nas páginas [60](#) e [65](#).

RIBENBOIM, P. *Números primos: velhos mistérios e novos recordes*. Rio de Janeiro: Instituto de Matemática Pura e Aplicada, 2012. Citado na página [140](#).

RICHARDSON, W. H. Bayesian-based iterative method of image restoration. *JoSA*, v. 62, n. 1, p. 55–59, 1972. Citado na página [118](#).

RIGATOS, G. G. Fault detection and isolation based on fuzzy automata. *Information Sciences*, Elsevier Inc., v. 179, n. 12, p. 1893–1902, 2009. Citado 2 vezes nas páginas [28](#) e [64](#).

RODRÍGUEZ, R. M. et al. A position and perspective analysis of hesitant fuzzy sets on information fusion in decision making. towards high quality progress. *Information Fusion*, v. 29, p. 89–97, 2016. Citado 3 vezes nas páginas [27](#), [29](#) e [45](#).

RODRÍGUEZ, R. M. et al. Hesitant fuzzy sets: state of the art and future directions. *International Journal of Intelligent Systems*, v. 29, n. 6, p. 495–524, 2014. Citado 6 vezes nas páginas [29](#), [45](#), [46](#), [47](#), [69](#) e [78](#).

ROMAN, S. *Lattices and ordered sets*. USA: Springer Science & Business Media, 2008. Citado 4 vezes nas páginas [40](#), [41](#), [42](#) e [47](#).

RUSPINI, E. H. On the semantics of fuzzy logic. *International Journal of Approximate Reasoning*, v. 5, n. 1, p. 45–88, 1991. Citado na página [114](#).

- SAMBUC, R. *Function Φ -flows, application a l'aide au diagnostic en pathologie thyroïdienne*. Tese (Doutorado) — University of Marseille, Marseille, France, 1975. Citado na página 27.
- SANTIAGO, R. H. N.; BEDREGAL, B. R. C.; ACIÓLY, B. M. Formal aspects of correctness and optimality of interval computations. *Formal Aspects of Computing*, v. 18, n. 2, p. 231–243, 2006. Citado na página 59.
- SANTOS, E. Fuzzy automata and languages. *Information Sciences*, v. 10, n. 2, p. 193–197, 1976. Citado 2 vezes nas páginas 52 e 104.
- SANTOS, E. S. Maximin Automata. *Information and Control*, v. 13, p. 363–377, 1968. Citado 3 vezes nas páginas 51, 52 e 55.
- SANTOS, E. S. Maximin, minimax, and composite sequential machines. *Journal of Mathematical Analysis and Applications*, v. 24, n. 2, p. 246–259, 1968. Citado na página 52.
- SANTOS, E. S. Max-product machines. *Journal of Mathematical Analysis and Applications*, v. 37, n. 3, p. 677–686, 1972. Citado na página 52.
- SANTOS, E. S. On reductions of maximin machines. *Journal of Mathematical Analysis and Applications*, v. 40, n. 1, p. 60–78, 1972. Citado 2 vezes nas páginas 51 e 52.
- SANTOS, E. S.; WEE, W. G. General formulation of sequential machines. *Information and control*, v. 12, n. 1, p. 5–10, 1968. Citado 2 vezes nas páginas 28 e 52.
- SANTOS, H. et al. Construction of typical hesitant triangular norms regarding Xu-Xia-partial order. In: ATLANTIS PRESS. *2015 Conference of the International Fuzzy Systems Association and the European Society for Fuzzy Logic and Technology (IFSA-EUSFLAT-15)*. Gijón, Asturias (Spain), 2015. Citado 6 vezes nas páginas 46, 47, 69, 75, 104 e 133.
- SANTOS, J. P. de O. *Introdução à teoria dos números*. Rio de Janeiro: Instituto de Matemática Pura e Aplicada, 1998. Citado na página 141.
- SERNADAS, A.; SERNADAS, C. *Fundamentos de Lógica e Teoria da Computação*. London: College Publications, 2012. Citado na página 141.
- SHANG, Y.-g.; YUAN, X.-h.; LEE, E. S. The n-dimensional fuzzy sets and zadeh fuzzy sets based on the finite valued fuzzy sets. *Computers & Mathematics with Applications*, v. 60, n. 3, p. 442–463, 2010. Citado na página 27.
- SHANNON, C. E. A mathematical theory of communication. *The Bell system technical journal*, Nokia Bell Labs, v. 27, n. 3, p. 379–423, 1948. Citado na página 124.
- SHARMA, B. K. et al. On fuzzy multiset regular languages. *Journal of Fuzzy Set Valued Analysis*, v. 2017, n. 1, p. 50–61, 2017. Citado na página 61.
- SHARMA, B. K.; TIWARI, S. P.; SHARAN, S. On Algebraic Study of Fuzzy Multiset Finite Automata. *Fuzzy Information and Engineering*, v. 8, p. 315–327, 2016. Citado na página 61.

SIPSER, M. *Introduction to the Theory of Computation*. New York: Cengage Learning, 2012. Citado na página 37.

SOLOMON, C.; BRECKON, T. *Fundamentals of Digital Image Processing: A practical approach with examples in Matlab*. Oxford: John Wiley & Sons, 2011. Citado na página 116.

SRIVASTAVA, A. K.; TIWARI, S. IF-topologies and IF-automata. *Soft Computing*, v. 14, n. 6, p. 571–578, 2010. Citado na página 62.

STANKOVIĆ, R. S.; FALKOWSKI, B. J. The haar wavelet transform: its status and achievements. *Computers & Electrical Engineering*, v. 29, n. 1, p. 25–44, 2003. Citado 2 vezes nas páginas 124 e 131.

STEIMANN, F.; ADLASSNIG, K.-P. Clinical monitoring with fuzzy automata. *Fuzzy Sets and Systems*, v. 61, n. 1, p. 37–42, 1994. Citado 2 vezes nas páginas 28 e 64.

STOLLNITZ, E. J. et al. *Wavelets for computer graphics: theory and applications*. Burlington, Massachusetts, EUA: Morgan Kaufmann, 1996. Citado 2 vezes nas páginas 161 e 162.

SUDBOROUGH, I. H. On tape-bounded complexity classes and multi-head finite automata. In: IEEE. *14th Annual Symposium on Switching and Automata Theory (swat 1973)*. Washington, DC, 1973. p. 138–144. Citado na página 39.

SYROPOULOS, A. *Theory of fuzzy computation*. Oxford: Springer, 2014. Citado 3 vezes nas páginas 27, 41 e 44.

THOMASON, M. G. Finite fuzzy automata, regular fuzzy languages, and pattern recognition. *Pattern Recognition*, v. 5, n. 4, p. 383–390, 1973. Citado na página 65.

TIWARI, S.; GAUTAM, V.; DUBEY, M. On fuzzy multiset automata. *Journal of Applied Mathematics and Computing*, v. 51, n. 1-2, p. 643–657, 2016. Citado 2 vezes nas páginas 61 e 65.

TIZHOOSH, H. R. Interval-valued versus intuitionistic fuzzy sets: Isomorphism versus semantics. *Pattern Recognition*, v. 41, n. 5, p. 1812–1813, 2008. Citado na página 62.

TOMASI, C.; MANDUCHI, R. Bilateral filtering for gray and color images. In: IEEE. *Sixth international conference on computer vision (IEEE Cat. No. 98CH36271)*. [S.l.], 1998. p. 839–846. Citado na página 118.

TORRA, V. Hesitant Fuzzy Sets. *International Journal of intelligent Systems*, v. 25, p. 529–539, 2010. Citado 5 vezes nas páginas 27, 43, 45, 46 e 48.

TORRA, V.; NARUKAWA, Y. On hesitant fuzzy sets and decision. In: IEEE. *2009 IEEE International Conference on Fuzzy Systems*. Jeju Island, Korea, 2009. p. 1378–1382. Citado 2 vezes nas páginas 27 e 45.

TURING, A. M. On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London mathematical society*, v. 2, n. 1, p. 230–265, 1937. Citado na página 33.

- VIDAKOVIC, B.; MUELLER, P. Wavelets for kids. *Instituto de Estadística, Universidad de Duke*, 1994. Citado na página 161.
- WALT, S. Van der et al. scikit-image: image processing in python. *PeerJ*, v. 2, p. e453, 2014. Citado na página 118.
- WANG, J.; YIN, M.; GU, W. Fuzzy multiset finite automata and their languages. *Soft Computing*, v. 17, n. 3, p. 381–390, 2013. Citado 3 vezes nas páginas 60, 61 e 65.
- WEE, W. G. *On Generalizations of Adaptive Algorithms and Application of the Fuzzy Sets Concept to Pattern Classification*. Tese (Doutorado) — Purdue University, Lafauette, Indiana, 1967. Citado 6 vezes nas páginas 27, 28, 52, 63, 104 e 134.
- WEI, G. Hesitant fuzzy prioritized operators and their application to multiple attribute decision making. *Knowledge-Based Systems*, v. 31, p. 176–182, 2012. Citado na página 46.
- WEN, M.; ZHAO, H.; XU, Z. Hesitant fuzzy lukasiewicz implication operation and its application to alternatives' sorting and clustering analysis. *Soft Computing*, v. 23, n. 2, p. 393–405, 2019. Citado na página 47.
- WIEDERMANN, J. Characterizing the super-Turing computing power and efficiency of classical fuzzy Turing machines. *Theoretical Computer Science*, v. 317, n. 1-3, p. 61–69, 2004. Citado 2 vezes nas páginas 27 e 63.
- WU, L.; QIU, D. Automata theory based on complete residuated lattice-valued logic: Reduction and minimization. *Fuzzy Sets and Systems*, v. 161, n. 12, p. 1635–1656, 2010. Citado na página 63.
- WU, L.; QIU, D.; XING, H. Automata theory based on complete residuated lattice-valued logic (II). *Science in Chine (Series F)*, v. 208, n. 6, p. 43–66, 2002. Citado 2 vezes nas páginas 62 e 104.
- WU, L.; QIU, D.; XING, H. Automata theory based on complete residuated lattice-valued logic: Turing machines. *Fuzzy Sets and Systems*, Elsevier, v. 208, p. 43–66, 2012. Citado na página 63.
- XIA, M.; XU, Z. Hesitant fuzzy information aggregation in decision making. *International journal of approximate reasoning*, v. 52, n. 3, p. 395–407, 2011. Citado na página 46.
- XIA, M.; XU, Z.; CHEN, N. Some hesitant fuzzy aggregation operators with their application in group decision making. *Group Decision and Negotiation*, v. 22, n. 2, p. 259–279, 2013. Citado na página 46.
- XING, H. et al. Equivalence in automata theory based on complete residuated lattice-valued logic. *Fuzzy Sets and Systems*, v. 158, p. 1407–1422, 2007. Citado na página 63.
- XU, Z.; XIA, M. Distance and similarity measures for hesitant fuzzy sets. *Information Sciences*, v. 181, n. 11, p. 2128–2138, 2011. Citado na página 47.
- YAGER, R. R. Uninorms in fuzzy systems modeling. *Fuzzy Sets and Systems*, v. 122, n. 1, p. 167–175, 2001. Citado na página 43.

YU, Z. G. Fuzzy L languages. *Fuzzy Sets and Systems*, v. 117, n. 2, p. 317–321, 2001. Citado na página 49.

ZADEH, L. A. Fuzzy sets. *Information and control*, Elsevier, v. 8, n. 3, p. 338–353, 1965. Citado 4 vezes nas páginas 27, 42, 43 e 47.

ZADEH, L. A. On fuzzy algorithms. p. 127–147, 1996. Citado 5 vezes nas páginas 27, 29, 50, 78 e 104.

ZHANG, F. et al. Hesitant distance set on hesitant fuzzy sets and its application in urban road traffic state identification. *Engineering Applications of Artificial Intelligence*, v. 61, p. 57–64, 2017. Citado na página 47.