

RUBEN CARLO BENANTE

PROCESSO: 366/99

PROGRAMA DE FILOSOFIA

DEFESA: 13/12/2001

Alcance e limites das
Redes Neurais Artificiais na
Ciência Cognitiva

Marília

2001



unesp



UNIVERSIDADE ESTADUAL PAULISTA
CAMPUS DE MARÍLIA
Faculdade de Filosofia e Ciências

RUBEN CARLO BENANTE

**Alcance e limites das
Redes Neurais Artificiais na
Ciência Cognitiva**

1410001288



RUBEN CARLO BENANTE

**Alcance e limites das
Redes Neurais Artificiais na
Ciência Cognitiva**

**Dissertação apresentada à Faculdade de Filosofia
e Ciências da Universidade Estadual Paulista
“Júlio de Mesquita Filho”, Campus de Marília,
para a obtenção do título de Mestre em Filosofia,
Área de Concentração: Ciência Cognitiva e
Filosofia da Mente.**

Orientadora: Prof. Dra. Maria Eunice Quilici Gonzalez

Marília

2001



Nº cham 153.4
B 456a

Nº Tombo 1288
Aquisição D
Preço 5,00
Data 10/14/02
Proced. Docentes

Benante, Ruben Carlo

Alcance e limites das redes neurais artificiais na ciência
cognitiva / Ruben Carlo Benante. -- Marília, 2001
246f.; 30 cm

Dissertação (Mestrado em Filosofia) - Faculdade de Filosofia e
Ciências – Universidade Estadual Paulista, 2001.

Bibliografia: f.230-243

1. Ciência cognitiva – redes neurais artificiais 2. Ciência
cognitiva – inteligência artificial. I. Autor. II. Título

CDD 153.4

RUBEN CARLO BENANTE

**Alcance e limites das
Redes Neurais Artificiais na
Ciência Cognitiva**

COMISSÃO JULGADORA

DISSERTAÇÃO PARA OBTENÇÃO
DO TÍTULO DE MESTRE

Presidente e Orientadora:	Prof. Dra. Maria Eunice Quilici Gonzalez Departamento de Filosofia - FFC - UNESP
2º Examinador:	Prof. Dr. Aluizio Fausto Ribeiro Araújo Departamento de Engenharia Elétrica - EESC - USP
3º Examinador:	Prof. Dr. Hércules de Araújo Feitosa Departamento de Matemática - FC - UNESP
4º Examinador:	Prof. Dr. Willem F. Haselager NICI - University of Nijmegen - Holanda

Marília, 13 de dezembro de 2001.

DADOS CURRICULARES

RUBEN CARLO BENANTE

NASCIMENTO	24 Dez 1974 – Araçatuba/SP
FILIAÇÃO	Luiz Benante Leda Maria Oliveira Vieira Benante
E-MAIL	benante@terra.com.br
1993/1998	Curso de Graduação em Ciências da Computação no Instituto de Biociências, Letras e Ciências Exatas de S. J. do Rio Preto, SP –UNESP.
1999/2001	Curso de Pós-Graduação em Filosofia, Área de Concentração em Ciência Cognitiva e Filosofia da Mente, Mestrado, na Faculdade de Filosofia e Ciências de Marília, SP - UNESP.
2001	Professor Titular do Departamento de Ciências da Computação, da Faculdade de Tecnologia, da Universidade Vale do Rio Doce de Governador Valadares, MG - UNIVALE.

Dedico este trabalho a todas as linhas causais
que engendraram sua realização.

Agradecimentos

São incontáveis as pessoas que me apoiaram e merecem mais do que meu simples agradecimento. Este trabalho dobraria seu tamanho se me fosse possível citar aqui um a um, reverenciando seu apoio e externando minha gratidão. Fica expresso o meu carinho a todos que, de uma forma ou de outra, colaboraram com a sua finalização. Em especial, minha gratidão:

aos amigos e à família, cujo apoio é antes um conforto à alma que uma solução de problemas específicos quaisquer;

à Prof. Dra. Maria Eunice Quilici Gonzalez, orientadora e incentivadora, de olhar crítico aguçado, mas sempre justo. Sua perseverança é fonte capaz de matar a sede de Tântalo e encher Sísifo de esperança;

ao Prof. Dr. Aluizio Fausto Ribeiro Araújo, pela tenacidade de embriagar-nos de suas enriquecedoras sugestões, num campo novo que está muito longe do consenso absoluto. A união de sua inteligência com sua afabilidade tornam sua presença deleitável;

ao Prof. Dr. Hércules de Araújo Feitosa, pela exemplar habilidade de tecer críticas com a estrutura de um diálogo e a delicadeza de um elogio. Pudera ele ter o poder de seu homônimo da mitologia grega e colaborar com a correção de doze trabalhos por vez, que a Ciência Cognitiva no Brasil em muito cresceria;

ao Prof. Dr. Willem F. Haselager, pelo apoio, incentivo, e pelas suas explicações esclarecedoras, tomadas na calada deste singelo departamento de filosofia. Pela sua capacidade de análise e síntese, que nos permitiu engrandecer este trabalho. E finalmente, por seu ótimo senso de humor, que o torna uma pessoa acessível e cheia de humildade, dando brilho à sua sabedoria.

“ O cálculo na imaginação é, em certo sentido, menos real do que o cálculo no papel? É o *verdadeiro* — cálculo de cabeça. — É semelhante ao cálculo no papel? — Não sei se devo dizer que é semelhante. Um pedaço de papel com alguns traços pretos é semelhante a um corpo humano?”

“ Uma máquina poderia pensar? — Poderia ter dores? — Ora, pode-se dizer que o corpo humano seja uma tal máquina? Ele chegará brevemente a ser tal máquina.

Mas uma máquina não pode pensar! [...]”

“ Nosso erro é procurar uma explicação lá onde deveríamos ver os fatos como ‘fenômenos primitivos’. Isto é, onde deveríamos dizer: *joga-se esse jogo de linguagem.*”

Wittgenstein (1889-1951)

Sumário

LISTA DE FIGURAS	11
LISTA DE TABELAS	13
LISTA DE ABREVIATURAS	14
INTRODUÇÃO GERAL.....	16
I.1 - HISTÓRICO DA CIÊNCIA COGNITIVA	16
I.2 - METAS E COLABORAÇÕES DESTE TRABALHO.....	23
I.3 - NOTAS.....	26
CAPÍTULO 1 - INTELIGÊNCIA ARTIFICIAL <i>VERSUS</i>	
REDES NEURAIIS ARTIFICIAIS	28
1.1 - APRESENTAÇÃO.....	28
1.2 - COMO SURTIU A ABORDAGEM DA INTELIGÊNCIA ARTIFICIAL.....	28
1.2.1 - <i>Modelos da Inteligência Artificial</i>	37
1.3 - COMO SURTIU A ABORDAGEM DAS REDES NEURAIIS ARTIFICIAIS.....	47
1.4 - A CONSTRUÇÃO DOS MODELOS COGNITIVOS.....	52
1.5 - NOTAS	58
CAPÍTULO 2 - MÁQUINAS DE TURING	60
2.1 - APRESENTAÇÃO.....	60

2.2 - MÁQUINAS DE TURING	60
2.3 - A CALCULABILIDADE EFETIVA	71
2.4 - MÁQUINAS DE TURING COM MÚLTIPLAS FITAS	75
2.5 - COMPUTABILIDADE	78
2.5.1 - Máquinas de Turing entendidas como aceitadores.....	82
2.5.2 - Máquinas de Turing entendidas como geradores	84
2.5.3 - Máquinas de Turing entendidas como algoritmos	86
2.6 - NOTAS	89
CAPÍTULO 3 - OS MODELOS DE REDES NEURAIIS ARTIFICIAIS	91
3.1 - APRESENTAÇÃO.....	91
3.2 - A INSPIRAÇÃO BIOLÓGICA DAS REDES NEURAIIS ARTIFICIAIS.....	91
3.3 - MODELOS DE REDES NEURAIIS ARTIFICIAIS.....	98
3.3.1 - A rede de McCulloch e Pitts.....	105
3.3.2 - O perceptron de uma camada e de multicamadas	115
3.3.3 - A rede de Hopfield.....	124
3.3.4 - Os mapas auto-organizáveis de Kohonen	130
3.4 - NOTAS	137
CAPÍTULO 4 - SIMULANDO O NEOCONEXIONISMO.....	139
4.1 - APRESENTAÇÃO.....	139
4.2 - PLANOS DE ANÁLISE	140
4.3 - PRINCIPAIS CARACTERÍSTICAS DO CONEXIONISMO	152
4.3.1 - Processamento numérico versus processamento simbólico.....	153



4.3.2 - <i>Processamento analógico versus processamento digital</i>	158
4.3.3 - <i>Processamento paralelo versus processamento seqüencial</i>	162
4.3.4 - <i>Processamento e armazenamento distribuído versus processamento centralizado</i>	167
4.3.5 - <i>Redundância e tolerância à falhas</i>	170
4.3.6 - <i>Redes Neurais Artificiais consideradas como funções</i>	173
4.4 - NOTAS	183
CAPÍTULO 5 - SISTEMAS AUTO-ORGANIZÁVEIS	187
5.1 - APRESENTAÇÃO	187
5.2 - SISTÊMICA E AUTO-ORGANIZAÇÃO	187
5.3 - CONDIÇÕES PARA A OCORRÊNCIA DA AUTO-ORGANIZAÇÃO	193
5.4 - AUTO-ORGANIZAÇÃO NAS REDES NEURAIIS ARTIFICIAIS	203
5.5 - AUTO-ORGANIZAÇÃO NAS MÁQUINAS DE TURING	212
5.6 - NOTAS	220
NOTAS FINAIS	222
N.1 - CONCLUSÕES	222
N.2 - PESQUISAS FUTURAS	227
REFERÊNCIAS BIBLIOGRÁFICAS	230
RESUMO	244
ABSTRACT	246

Lista de Figuras

FIGURA 1: ESQUEMA DE UMA MÁQUINA DE TURING	64
FIGURA 2: DIAGRAMA DA MÁQUINA DE TURING REPRESENTADA NA FIGURA 1	65
FIGURA 3: MÁQUINA DE TURING EM UM ETERNO CICLO	67
FIGURA 4: EQUIVALÊNCIA ENTRE MÁQUINAS DE TURING: (A) MÚLTIPLAS FITAS, (B) ÚNICA FITA.....	77
FIGURA 5: UMA RELAÇÃO ENTRE OS TIPOS DE LINGUAGENS FORMAIS	84
FIGURA 6: NEURÔNIO PRÉ-SINÁPTICO E PÓS-SINÁPTICO.....	93
FIGURA 7: NEURÔNIO EM REPOUSO	94
FIGURA 8: NEURODO GERAL, ESQUEMA INTERNO DE UM NEURODO K.....	100
FIGURA 9: UMA REDE LÓGICA DE McCULLOCH E PITTS QUE IMPLEMENTA O CONECTIVO LÓGICO OU-EXCLUSIVO	106
FIGURA 10: ESQUEMA DE NEURODO DE McCULLOCH E PITTS	108
FIGURA 11: MÁQUINA DE TURING QUE COMPUTA UM NEURODO DE McCULLOCH E PITTS.....	109
FIGURA 12: FITA GERAL PARA NEURODO DE McCULLOCH E PITTS.....	111
FIGURA 13: MÁQUINA DE TURING DE MÚLTIPLAS FITAS SIMULANDO UMA REDE DE McCULLOCH E PITTS PARA RESOLVER O PROBLEMA DO OU-EXCLUSIVO.....	112
FIGURA 14: DISTRIBUIÇÃO DAS ENTRADAS A E B NOS NEURODOS M E N DA REDE DE McCULLOCH E PITTS, SIMULADA NUMA MÁQUINA DE TURING	113

FIGURA 15: VÁRIAS MÁQUINAS DE TURING DE MÚLTIPLAS FITAS INTERLIGADAS EM REDE	114
FIGURA 16: (A) PERCEPTRON; (B) DUAS CLASSES, A E B; (C) CONVERGÊNCIA NO APRENDIZADO; (D) O PROBLEMA DO OU-EXCLUSIVO	117
FIGURA 17: REGIÕES DE DECISÃO QUE PODEM SER FORMADAS COM PERCEPTRONS	120
FIGURA 18: O MODELO DE REDE DE HOPFIELD.....	126
FIGURA 19: UMA SOLUÇÃO DO CAIXEIRO VIAJANTE USANDO REDE DE HOPFIELD	129
FIGURA 20: MAPAS AUTO-ORGANIZÁVEIS DE KOHONEN. (A) ESQUEMA DE UMA ARQUITETURA TÍPICA. (B) VETOR PESO DO NEURODO I NA MATRIZ. (C) MAPAS AUTO-ORGANIZADOS.....	131
FIGURA 21: PLANOS DE HASELAGER	146
FIGURA 22: A GRAMÁTICA REGULAR COM PESOS NA HIERARQUIA DE CHOMSKY .	174

Lista de Tabelas

TABELA 1: PASSOS SEGUIDOS PELO CONTROLADOR PARA EXECUTAR UMA INSTRUÇÃO NA MÁQUINA DE TURING.....	64
TABELA 2: EXECUÇÃO DA PRIMEIRA INSTRUÇÃO DA FIGURA 1	65
TABELA 3: TABELA DE VERDADE DOS CONECTIVOS LÓGICOS OU, E, E DO OU-EXCLUSIVO.....	107
TABELA 4: COMPARAÇÕES ENTRE IA E RNA	148

Lista de Abreviaturas

AO = Auto-Organização

AF = Autômato Finito

AP = Autômato com uma Pilha

APROB = Autômato PROBabilístico

BOVEIA = BOa e VELha Inteligência Artificial (*GOFAI = Good Old-Fashioned Artificial Intelligence*)

CCD = Ciência Cognitiva Dinâmica

FTC = Função Turing-Computável (função recursiva)

FÑTC = Função Não Turing-Computável (função não-recursiva)

GPS = *General Problem Solver* (Solucionador geral de problemas)

IA = Inteligência Artificial (Clássica)

LISP = LIST Processing (processamento de listas)

LLC = Linguagem livre de contexto

LREC = Linguagem RECursiva

LREG = Linguagem REGular

LRP = Linguagem Regular com Pesos

LSC = Linguagem sensível ao contexto

LT = *Logic Theorist* (Teórico lógico)

MAO = Mapas Auto-Organizáveis (*SOM = Self-organizing Maps*)

MC = Máquina Conexionista

MLP = *Multi Layer Perceptron* (perceptron multicamadas)

MP = McCulloch e Pitts

MRC = Modelos Realistas do Cérebro

MSC = Modelos Simplistas do Cérebro

MT = Máquina de Turing

MTU = Máquina de Turing Universal

MVN = Máquina de Von Neumann

NG = Neurodo Geral

PC('s) = *Personal Computer(s)* (Computador(es) Pessoal(ais). Computador digital de propósito geral, baseado nas MVN)

PEPS = Potencial Excitatório Pós-Sináptico

PIPS = Potencial Inibitório Pós-Sináptico

RH = Rede de Hopfield

RK = Rede de Kohonen

RNA = Rede(s) Neural(ais) Artificial(ais)

TAD = Tipo Abstrato de Dados

TAO = Teoria da Auto-Organização

TC = Tese de Church

TE = Teorema de Equivalência

TT = Teorema de Turing

TTC = Tese de Turing-Church

VA = Vida Artificial



Introdução Geral

I.1 - Histórico da Ciência Cognitiva

A ciência cognitiva surgiu em um momento de grande entusiasmo científico, com a intenção de tentar resolver alguns dos problemas filosóficos que vêm incomodando a humanidade há aproximados três milênios. Desde o seu início, a ciência cognitiva foi formada por duas visões, mas um só objetivo em comum. O objetivo era fundar uma ciência da mente, empírica, baseada em modelos computacionais. As duas visões surgiram na revolução lógica da cibernética nos anos 30 e se firmaram cada vez mais como duas vertentes distintas: a Inteligência Artificial (IA ou cognitivismo clássico) e as Redes Neurais Artificiais (RNA ou connexionismo).

Uma das motivações para o desenvolvimento da área foi um problema de lógica que ficou conhecido como o *problema da decisão*, proposto por Hilbert, que fazia a seguinte pergunta: Dada uma fórmula qualquer do cálculo de predicados clássico, existe um procedimento sistemático finito que permita determinar se essa fórmula é demonstrável ou não? (DUPUY, 1995, p.28). Um matemático britânico se tornou famoso por resolver esse problema de uma forma muito criativa, mas mais do que isso, por propor as bases matemáticas do mecanicismo. Alan M. Turing, no seu artigo de 1936, "*On Computable Numbers, with an Application to the Entscheidungsproblem*", definiu o conceito formal de

uma máquina simples capaz de realizar qualquer cálculo descritível por um procedimento sistemático, geral, finito, sem ambigüidades (que formaria a noção de algoritmo), e então reformulou a pergunta de Hilbert para sua equivalente: “Existe uma máquina de Turing capaz de computar se determinado procedimento sistemático finito levará a uma demonstração de um teorema ou não?”. A conclusão a que chega – de que não existe tal máquina – é no mínimo impressionante, pois a máquina pode provar teoremas (como um ser humano, ou talvez até de maneira mais estruturada), mas não pode prever se conseguirá ou não provar tais e quais teoremas sem tentar efetivamente a prova (também como um ser humano). Isso levou os cientistas da época a crerem que a máquina era capaz de realizar tarefas que pareciam estar no domínio apenas do ser humano.

Nesse mesmo ano, Church propôs uma classe de funções, independentemente de Turing, que ao ser confrontada com a classe de funções que a máquina de Turing era capaz de computar, revelou serem equivalentes (chamadas hoje de *funções Turing-computáveis* ou *funções recursivas*), o que corroborou a tese de Turing de que sua máquina modelava todos os cálculos mecânicos. Junto veio sua noção de Máquina de Turing Universal (MTU), a qual seria a máquina de Turing geral, que abarcaria todas as máquinas de Turing específicas. Esta máquina constituiria a base do modelo mecânico da mente, na concepção dos pesquisadores da época, que depois bifurcaram-se entre os adeptos da Inteligência Artificial e os chamados Conexionistas.

Assim se desenvolve o movimento cibernético, que se utilizará da noção de máquinas para dar uma resposta materialista ao problema do controle

das ações inteligentes, eliminando em princípio a mente e anos mais tarde absorvendo uma versão funcionalista originada do artigo de Turing. Mais precisamente, a cibernética teve início a partir do ano de 1943, com a publicação de dois artigos, cujos autores estavam imersos no turbilhão gerado por esses novos paradigmas mecanicistas.

O primeiro artigo foi escrito pelo fundador da cibernética, Norbert Wiener, com a colaboração do fisiologista Rosenblueth e de Bigelow, intitulado “*Behavior, Purpose, and Teleology*”, no qual se expôs pela primeira vez a idéia de uma ciência baseada no controle e na comunicação. Wiener notou que os problemas com que se deparavam a engenharia de controle e a engenharia de comunicação não eram oriundos das técnicas utilizadas, eram antes de âmbitos gerais, provindos das noções ainda mal formuladas de comunicação, fosse ela realizada em meios elétricos, mecânicos ou nervosos. Com isso foi gerada uma nova taxionomia que conseguia classificar em um patamar bem estruturado tanto espécies vivas como máquinas dependendo do grau de comunicabilidade (DUPUY, 1995, p.45).

Outra característica interessante desse artigo é a introdução do conceito até então vago de *processos circulares* que usam o mecanismo de *realimentação* (ou *feedback*). A importância da realimentação, porém, só foi devidamente ponderada anos mais tarde na segunda cibernética (ou neocibernética) com von Foerster, que qualificou como “máquinas triviais” aquelas que possuíam uma estrutura fixa de resposta a estímulos dados, e

“máquinas não-triviais” aquelas que, através do mecanismo de retroalimentação, conseguiam se autocorrigir em busca de um objetivo (GARDNER, 1996).

As máquinas não-triviais a que se referiu von Foerster seriam os autômatos de estados finitos (que são tipos limitados de máquinas de Turing). Esses autômatos têm a interessante propriedade de exibir um comportamento altamente complexo devido à explosão combinatória que os diferentes tipos de estados internos podem assumir para cada estímulo externo dado. Essa explosão combinatória torna uma análise de suas respostas tarefa humanamente impossível, só realizável por uma outra máquina (de capacidades de ordem exponencialmente superior) que *simularia* o comportamento (simulação da simulação) da primeira, não tendo necessariamente que chegar a resposta alguma (devido ao problema de parada, previsto por Turing), e levando muito mais tempo do que o necessário para simplesmente simular o algoritmo. Dependendo da complexidade do algoritmo, sua simulação poderia levar anos ou até milênios!

São nessas máquinas que os cientistas da inteligência artificial irão se concentrar, inspirados nesse novo paradigma que a cibernética de Wiener e Rosenblueth apresentaram, utilizando os conceitos de *resolução de problemas* da inteligência nas máquinas proposto por Turing em 1950 no artigo “*Computing Machinery and Intelligence*”, no qual expõe o teste chave da ciência cognitiva: o Teste de Turing¹.

O artigo de Wiener, Bigelow e Rosenblueth deixou marcas em tantas áreas quantas se pode imaginar, da psicologia familiar à produção de armas de guerra (influenciando inclusive o surgimento do connexionismo), apesar de

estarmos aqui enfatizando apenas uma delas, qual seja, a tentativa materialista de solução do problema mente/corpo, através do estudo do controle das ações inteligentes.

A vertente connexionista se identificou com o outro artigo fundador da cibernética – e tão importante quanto o primeiro – do neuropsiquiatra Warren McCulloch e do matemático e lógico Walter Pitts. Esse artigo, intitulado “*A Logical Calculus of the Ideas Immanent in Nervous Activity*”, mostrou ser possível entender a atividade neural em termos lógicos identificando o pulso elétrico de um neurônio com o valor lógico *verdadeiro* e seu repouso (ou não-pulso) com o valor lógico *falso*, de maneira que a rede neural toda corresponderia a um conjunto de proposições lógicas interconectadas que se inferem mutuamente (MCCULLOCH & PITTS, 1943).

A rede que McCulloch e Pitts propuseram era muito limitada, mas suas incapacidades não foram vistas no momento, muito pelo contrário, a rede causou grande alvoroço entre os pensadores presentes. McCulloch e Pitts chegaram a argumentar que existia uma rede apropriada que podia realizar qualquer computação que fosse exaustivamente descrita de maneira inequívoca. Surgiram comparações entre as máquinas de Turing e a rede de McCulloch-Pitts, que só mais tarde, com as correções de Kleene (1956), foram rechaçadas, mostrando que as funções que a rede de McCulloch-Pitts podia calcular estavam estritamente dentro das funções recursivas (ou as funções Turing-computáveis) (DUPUY, 1995, p.65). Mas, apesar de todos os contratempos que foram impostos

a essa rede, a idéia filosófica de modelar o cérebro humano através de neurônios artificiais ideais estava lançada (GARDNER, 1996).

Essas duas ramificações da ciência cognitiva possuem alguns pressupostos em comum, que começam pelo método de análise dos resultados. Por se tratar de uma *ciência* faz-se necessário, antes de tudo, que os modelos levantados como hipóteses explicativas sejam *objetivos* e *testáveis*, como os modelos das diversas ciências naturais. Ambas as vertentes da ciência cognitiva compartilham das hipóteses: (1) *mecanicista da mente*, que implica em admitir que os estados mentais são regidos por leis, e que uma vez conhecidas essas leis, pode-se explicar a mente através de um modelo computacional implementado em uma máquina (que pode ser conexionista ou do tipo von Neumann²); (2) da *hipótese representacionista*, segundo a qual nosso conhecimento sobre o mundo é mediado por representações mentais (considerando-se a diferença conceitual entre as representações do conexionismo e as representações da IA); e (3) da *hipótese funcionalista*, segundo a qual a mente pode ser modelada baseando-se no funcionamento dos processos estruturais que a geram. Também aqui há uma diferença entre o funcionalismo da IA (ou funcionalismo lógico-computacional) e o do Conexionismo – denominado funcionalismo neurocomputacional (GONZALEZ, 1989).

Apesar dessas duas abordagens serem dominantes, existem outras entre as quais podemos destacar a abordagem através da construção de sistemas dinâmicos (BROOKS, 1997; VAN GELDER, 1997; AGRE & CHAPMAN, 1987) que preconiza a inteligência sem representação mental,

através da ação direta do sistema com o meio, conhecida como *Ciência Cognitiva Dinâmica* (CCD). Essa terceira vertente vem ganhando força, tomando o lugar do Conexionismo que, como a IA, começou a esbarrar em problemas após uma euforia inicial. Além disso, a CCD é a única vertente que, aparentemente, consegue o intento que era inicialmente dos conexionistas, de eliminar o representacionismo e o funcionalismo, permanecendo apenas a hipótese mecanicista. Há ainda na literatura muitas outras formas de abordar o problema de entender a mente inteligente além da IA, da RNA, e da CCD, como, por exemplo, as vertentes denominadas: Vida Artificial; Neuroetologia Computacional; Representação Dinâmica; Mecanismo de Esquemas; Teoria da Seleção de Grupos Neurais; Algoritmo de Reação e Planejamento; Teoria do Pandemônio; Sociedade da Mente e outras conforme nos indica Franklin (1995, p.11-17). Não trataremos aqui de tais abordagens, mas nos prenderemos à IA e à RNA, consideradas atualmente as principais vertentes da Ciência Cognitiva.

Há também uma tentativa entre essas abordagens (da IA e da RNA) de deixarem de lado os aspectos sociais e emocionais do sujeito cognoscente numa tentativa de simplificar o estudo da inteligência, concentrando-se nos aspectos racionais, lógicos, perceptivos e na capacidade de abstração que um sujeito inteligente pode demonstrar. Alguns cientistas, entretanto, afirmam que a inteligência racional e a inteligência emocional estão estreitamente ligadas e não podem, por essa razão, serem estudadas separadamente (MATURANA & VARELA, 1980; DAMÁSIO, 1996).

O estudo interdisciplinar é outra característica da ciência cognitiva, ressaltando-se o método de troca de pesquisas em áreas como a psicologia, a ciência da computação, a filosofia, a lingüística e a neurociência. Essas cinco disciplinas são os pilares da ciência cognitiva³. Gardner (1996), na sua classificação da ciência cognitiva, substitui a ciência da computação por apenas um de seus ramos, a saber, a inteligência artificial, o que em nossa opinião não supre as necessidades da ciência cognitiva, pois é a ciência da computação como um todo (através da teoria de computação, teoria dos autômatos, análise de sistemas, complexidade de algoritmos, etc.) que apóia tanto a inteligência artificial quanto as redes neurais artificiais⁴.

Finalmente, entre as cinco disciplinas citadas, temos o importante papel desempenhado pela filosofia, como um crítico avaliador dos avanços que a área tem feito, e como um proponente de questões a serem respondidas. É com o auxílio das reflexões filosóficas e computacionais, principalmente, que a presente dissertação se desenvolve.

I.2 - Metas e colaborações deste trabalho

Este trabalho propõe-se a investigar dois problemas centrais, que denominaremos *problema da emergência* e *problema da Turing-computabilidade*. O problema da emergência consiste em verificar se não há perda de propriedades relevantes que emergem das Redes Neurais Artificiais (em especial as Redes de Kohonen) quando estas são postas para funcionar numa *simulação* feita em uma máquina de Turing. Nesse caso, se não houver perda dessas propriedades, seu

surgimento seria devido a processos de auto-organização, como alguns pretendem (e poderíamos falar em auto-organização nas máquinas de Turing), ou simplesmente propriedades emergentes intrínsecas aos algoritmos computacionais.

Há muita polêmica envolvida na classificação das redes neurais artificiais, entendidas como algoritmos que podem ser implementados nas máquinas de Turing, uma vez que os modelos conexionistas são altamente paralelos e distribuídos e, a rigor, só poderiam ser implementados em máquinas conexionistas reais. Questionaremos se tais máquinas computam alguma classe de funções mais abrangente do que aquelas passíveis de serem computadas pelas máquinas de Turing, ou se são equivalentes a estas.

O problema da Turing-computabilidade consiste em responder a seguinte questão: seriam os modelos de redes neurais artificiais funções Turing-computáveis? A razão da existência do problema da Turing-computabilidade diz respeito à uma divergência entre as vertentes da IA e RNA que requerem para si o mérito de apresentarem o método apropriado para o estudo da mente. Procuraremos mostrar que tal divergência pode decorrer de um problema conceitual relativo à classificação dos modelos da RNA e IA. Dependendo das características que se focalizam no estudo da mente, parecem haver sutis diferenças de abordagem que tornam um ou outro modelo mais apropriado, contudo se as simulações das RNA nas máquinas de Turing apresentarem as mesmas propriedades emergentes apresentadas por suas contrapartes em máquinas conexionistas reais, então poderemos concluir que as duas vertentes (IA Clássica e

Conexionismo) são fundamentalmente equivalentes. Isso significa dizer que qualquer uma dessas vertentes conseguindo seu objetivo terá corroborado a proposta de Turing⁵.

Os problemas da emergência e da Turing-computabilidade acima descritos serão investigados em cinco capítulos. No capítulo 1, introduziremos as duas grandes vertentes da ciência cognitiva: a IA e a RNA, ressaltando as principais distinções metodológicas dessas áreas na investigação do comportamento inteligente. No capítulo 2, apresentaremos os conceitos que serão utilizados na elaboração de uma possível solução aos dois problemas aqui apresentados, incluindo a definição de máquinas de Turing, do conceito de computabilidade, e de funções efetivamente calculáveis. No capítulo 3, faremos uma introdução às neurociências, em seguida avaliaremos cinco dos mais conhecidos modelos de RNA: a rede de McCulloch e Pitts, o perceptron, o perceptron multicamadas, a rede de Hopfield e as redes auto-organizadas de Kohonen. No capítulo 4, apresentaremos uma discussão sobre planos de análise, e então traçaremos a estratégia de análise do problema da emergência, comparando as propriedades das máquinas connexionistas, assim como as propriedades dos modelos abstratos de RNA. No capítulo 5, levaremos as reflexões sobre máquinas de Turing e redes neurais artificiais, até então desenvolvidas, para serem reavaliadas sob um novo ponto de vista: o da auto-organização.

I.3 - Notas

¹ *The Imitation Game* (O jogo da Imitação), consiste em colocar isoladamente, em três salas, uma máquina de Turing executando um programa que simula a inteligência humana, um ser humano, e um interrogador humano. O interrogador tentará através de questionário escrito descobrir a identidade dos outros dois participantes do jogo. Neste jogo, a máquina de Turing deverá enganar o interrogador e fazer passar-se por um ser humano. O papel do ser humano é convencer o interrogador da verdadeira identidade dos participantes (TURING, 1950, p.40).

² As máquinas de von Neumann, como são conhecidas na ciência da computação, são os computadores digitais sequenciais dotados de uma memória que armazena tanto programas quanto dados. Von Neumann se baseou nas máquinas de Turing acrescentando apenas a inovação que lhe é atribuída de ter colocado na *fita* da máquina de Turing o programa codificado em notação binária além, é claro, dos dados.

³ Gardner (1996, p.20) chama de Ciências Cognitivas o conjunto interdisciplinar formado por essas cinco áreas e ainda uma sexta, a antropologia. Entendemos que as Ciências Cognitivas diferem da Ciência Cognitiva por vários outros aspectos além deste, como por exemplo, a centralidade do computador como modelo (GONZALEZ, 1989). É também discutível se as Ciências Cognitivas teriam estas disciplinas colaborando de forma interdisciplinar ou se seria antes um esforço multidisciplinar. Notamos ainda a importância da matemática, que originou a ciência da computação e a suporta.

⁴ Alguns autores (BITTENCOURT, 1998, p.301; RICH & KNIGHT, 1993, p.560) da ciência da computação colocam as redes neurais artificiais dentro da inteligência artificial como apenas uma outra técnica que visa a produção artificial de inteligência. Esses autores esquecem, em nossa opinião, que as redes neurais artificiais têm um pé nas neurociências, que só a ciência cognitiva com seu compromisso com a interdisciplinaridade é capaz de suportar. Isto contribui para a mixagem dos conceitos entre estas duas áreas, e do alcance de cada uma.



Capítulo 1 - Inteligência Artificial versus

⁵ Turing (1950) acreditava que, devidamente programadas, as máquinas de Turing apresentarão inteligência suficiente para confundir-se com seres humanos no *Jogo da Imitação* ou *Teste de Turing*. (TURING, 1950, p.10)



Capítulo 1 - Inteligência Artificial *versus*

Redes Neurais Artificiais

1.1 - Apresentação

Neste capítulo analisaremos as principais diferenças e semelhanças entre as duas vertentes da Ciência Cognitiva – Inteligência Artificial e Redes Neurais Artificiais –, no contexto histórico de seu surgimento. Na seção 1.2, faremos um breve relato do surgimento da IA e consideraremos algumas críticas direcionadas ao seu método de investigação. Analisaremos também, na subseção 1.2.1, algumas técnicas utilizadas pela IA no desenvolvimento de seus modelos. Na seção 1.3, introduziremos um histórico das RNA, levantando algumas das críticas mais comuns a essa vertente da ciência cognitiva. Finalmente, na seção 1.4, investigaremos a técnica de construção de modelos para a simulação do comportamento inteligente, entendida como um método científico apropriado para o estudo da mente.

1.2 - Como surgiu a abordagem da Inteligência Artificial

As idéias dos precursores da cibernética, principalmente aquelas contidas no artigo de 1950 de Turing, “*Computing Machinery and Intelligence*” constituem o ponto de apoio para uma nova geração de cientistas que estavam



prontos para colocar em prática a teoria confabulada numa época em que os computadores foram chamados de cérebros eletrônicos.

O termo *Inteligência Artificial* (IA) foi primeiramente utilizado por McCarthy (GARDNER, 1996), que juntamente com Minsky, Simon, Newell e outros cientistas com formação em lógica e matemática resolveram se reunir, em 1956, em Hanover com o propósito (que mais ou menos define o que seria a IA nos próximos anos) de realizar um estudo sobre a natureza da mente inteligente fundamentado na hipótese que os aspectos da inteligência podem ser precisamente descritos, a ponto de se criar uma máquina que os simulem (GARDNER, 1996, p.153). Nesse contexto, Marr (1990) define a inteligência artificial como:

Artificial Intelligence is the study of complex information-processing problems that often have their roots in some aspect of biological information-processing. The goal of the subject is to identify interesting and solvable information-processing problems, and solve them. (MARR, 1990, p.133)¹

Um ponto polêmico, que colaborou com as divergências entre os adeptos da IA e os conexionistas, foi a suposição daqueles de que não importava a arquitetura da máquina computacional que estaria executando o algoritmo de simulação de inteligência. Esse desprendimento da inteligência em relação ao elemento corpóreo é característica das investigações da Inteligência Artificial, segundo as quais a mente é um programa cujo suporte físico não é importante para o entendimento da sua estrutura funcional. Putnam (1967), por exemplo, argumentava que é possível a simulação de algoritmos inteligentes em computadores em nada parecidos com o cérebro humano.

Essa despreocupação com a estrutura física do sistema inteligente tem suas raízes, principalmente, no artigo de Turing (1950, p.62). Considerado o pai do mecanicismo lógico, ele deixa claro nesse artigo a sua opinião de que uma máquina de Turing devidamente programada poderá apresentar inteligência o suficiente para confundir um interrogador cético, pelo menos setenta por cento das vezes em que for submetido ao teste de Turing, em no máximo 50 anos – portanto no ano 2000 (TURING, 1950, p.49).

Assim se forma um outro pressuposto importante dessa vertente da Ciência Cognitiva, o *funcionalismo*.

Os funcionalistas, dentre os quais encontramos Putnam (1967), Dennett (1978) e Pylyshyn (1984), diminuem um pouco a relevância do cérebro em relação aos estados mentais. Ele é apenas o meio pelo qual estados mentais do ser humano são realizados, mas não é necessário para a sua realização. O Substrato material de uma entidade qualquer deve apenas ter um certo nível de complexidade para poder ‘computar’ os processos que caracterizam os estados mentais. (ALVES, 1999, p.31)

O pressuposto funcionalista indica como saída aos problemas das teorias da mente que lhe precederam que os estados mentais não devem ser vistos como vinculados a algum tipo de estrutura física, mas sim como sinônimos das funções que esses estados executam (PUTNAM, 1967).

Desconsiderando-se o funcionamento do cérebro para resolver problemas, por um lado, e considerando-se o teste de Turing como apropriado para avaliar a capacidade da máquina de responder coerentemente a perguntas, por outro lado, deu-se início a uma corrida “programamentista” para se

desenvolver computadores que, dado um *input* (entrada), responderiam com um *output* (saída) correto. O intermediário desconhecido (o funcionamento do cérebro) foi substituído por técnicas de programação que foram desenvolvidas especificamente para esse fim.

Como as técnicas eram também desconhecidas até então, podemos perceber quão excitante era criar tais técnicas computacionais (que tiveram seus muitos méritos e avanços). Notemos que o que inspirava esse novo campo era a possibilidade de serem essas técnicas² computacionais as próprias técnicas que estavam inscritas na mente humana. Surge portanto a idéia de se compreender os *estados mentais* como abstrações de regras e o raciocínio como um processo de manipulação de símbolos através de algoritmos.

A visão da mente como um conjunto de algoritmos, devidamente implementados, não era compartilhada por todos os pesquisadores da época (início da década de sessenta), cujas controvérsias na área apontaram para uma divisão da inteligência artificial: a IA Forte, segundo a qual a mente é um programa instanciado no cérebro e por isso computadores devidamente programados teriam a capacidade de pensar; e a IA Fraca, segundo a qual é possível tornar os computadores inteligentes (até mais inteligentes que os seres humanos) programando-os utilizando-se do paradigma das regras abstratas e dos algoritmos, mas tais programas computacionais não refletiriam necessariamente o modo como a mente trabalha. Searle (1997, p.288) chama de IA Forte a teoria que considera que tudo o que existe na mente pode ser programado e de IA Fraca a concepção de que os processos cerebrais e os processos mentais são

computacionalmente simuláveis³. Essas duas vertentes da IA caracterizam-se pela técnica de análise *top-down* do seu objeto de investigação

Por *top-down* entendemos a direção de investigação adotada pelo pesquisador ao desenvolver uma solução para um determinado problema, buscando construir suas macroestruturas e, eventualmente, corrigindo pequenos detalhes no plano microscópico de análise. Esta abordagem contrasta-se com aquela conhecida como *bottom-up*, em que o pesquisador identifica e desenvolve as microestruturas concernentes ao problema e aposta na interação entre esses elementos para a emergência espontânea da solução procurada. Assim, Franklin (1995, p.11-17) coloca: (a) A psicologia cognitiva como uma abordagem *top-down* e analítica. *Top-down* porque estuda os conceitos psicológicos de alto nível, e analítica porque analisa mentes existentes e tenta entendê-las. (b) A Inteligência Artificial Clássica como uma abordagem *top-down* e sintética. *Top-down* porque o projetista cria estruturas simbólicas representativas de funções cognitivas de alto nível, e sintética porque, pelo menos na IA Forte, é uma tentativa de construir modelos de mentes.

Quanto à Inteligência Artificial, entendemos que sua classificação está bem colocada: a abordagem *top-down* da IA, característica típica dessa vertente, se deve ao modo como o projetista (de cima) impõe através de regras (o que inclui as meta-regras que possuem a capacidade de alterar suas regras subordinadas) um comportamento à máquina (para baixo).

No início da era da computação, ocorrida entre os meados da década de cinquenta e início da década de setenta, os progressos foram evidentes,

mas mais voltados às técnicas de programação. Foram desenvolvidas linguagens de alto nível para substituir a linguagem de máquina na tarefa de codificação de programas. Uma dessas linguagens, a linguagem LISP, desenvolvida por McCarthy no final da década de sessenta, foi a linguagem mais utilizada pelos cientistas da IA (GARDNER, 1996, p.169).

Os sistemas operacionais também tiveram seus avanços, principalmente no gerenciamento de entrada (que passou das conexões de fios, pelos cartões perfurados, fitas perfuradas, teclados, canetas óticas, filmadoras, etc.), de saída (que passou dos códigos com lâmpadas pelos perfuradores de cartão, impressoras, monitores, mídia magnética e ótica, som, etc.) e de memória (nos primeiros computadores, limitada e fixa, depois com relocação estática, relocação dinâmica, memórias não voláteis – o caso das fitas magnéticas, disquetes, discos-rígidos, etc.).

A cada avanço os programadores tinham mais possibilidades e maior grau de liberdade para produzir sistemas que se assemelhavam cada vez mais aos sistemas ditos inteligentes. Parecia que a hipótese de Turing de que a maioria dos problemas abstratos seria resolvida com a adição de memória estava sendo confirmada (TURING, 1950, p.48, p.54-55, p.61).

Newell e Simon criaram o programa *Logic Theorist* (LT) que realizaria em agosto de 1956 a primeira prova de teoremas, feita por computador (NEWELL & SIMON, 1961). O algoritmo do LT, que foi desenvolvido por esses estudiosos para conseguir que o programa provasse um teorema, foi, sem dúvida, um formidável avanço de *software*.

O LT funcionava baseado em um banco de dados primitivo com um conjunto de operadores lógicos, os quais podiam ser aplicados a qualquer momento da produção da prova. O programa seguia tentando aplicar os operadores à cláusula a ser provada, exaustivamente, gerando uma árvore de possibilidades, até que ele conseguia encontrar nessa árvore a resposta desejada, ou até que se acabassem todas as possibilidades, quando, então, o programa diria que não conseguiu provar o teorema. Resumidamente, temos: 1- O estado inicial (a proposição a ser provada), 2- O estado final (a prova) e 3- Os operadores possíveis (os operadores lógicos).

Esse mesmo algoritmo foi usado das mais diversas maneiras e produziu resultados surpreendentes. Imagine que esse algoritmo pode jogar xadrez! Substitua: 1- O estado inicial: a posição inicial das peças, 2- O estado final: o xeque-mate, 3- Os operadores possíveis: os movimentos das peças. Pronto, temos agora um novo programa, utilizando-se do mesmo algoritmo elaborado para a demonstração de teoremas, mas que faz algo completamente diferente! O programa aplicará cegamente os operadores no estado inicial até conseguir achar o estado final ou até dizer: “Não consegui dar xeque-mate!”.

Essa técnica de busca em árvore é chamada⁴ de *heurística* para a resolução de um problema. Com o LT pronto, Newell, Simon e alguns colegas se propuseram a criar um solucionador geral de problemas (*General Problem Solver* ou simplesmente GPS) que se utilizava de várias heurísticas diferentes além daquela descrita acima. Uma vez que se conseguia descrever em termos lógicos o estado inicial do problema, seus operadores e o seu estado final, o GPS resolvia o



problema (NEWELL & SIMON, 1963). A euforia era tamanha que Simon, em 1957, se permite fazer previsões grandiloquas, entre elas a de que em no máximo dez anos (ou seja, antes da década de setenta) o computador digital seria campeão mundial de xadrez (DREYFUS, 1975). Dreyfus⁵ aponta o GPS como um marco que parece ser a inauguração da Inteligência Artificial. Ele também relembra que no final de quatorze anos essas previsões não foram alcançadas pela IA.

Apesar de alguns críticos acreditarem que o computador digital nunca seria campeão mundial de xadrez, em 1996 a IBM desafiou pela segunda vez o então campeão mundial Garry Kasparov num campeonato composto de seis jogos contra o Deep Blue, uma máquina devotada a jogar xadrez. Deep Blue perdeu o campeonato, mas deixou como marco na história a *primeira vitória* de um computador contra o campeão mundial de xadrez⁶, na partida inaugural do campeonato (perdeu as outras três e empatou duas). Entretanto, em maio de 1997, na sua terceira revanche, Deep Blue, jogando o que foi depois definido por seu adversário como um estilo completamente humano, venceu o campeonato⁷ ganhando duas partidas, perdendo uma e empatando três (SCHAEFFER & PLAAT, 1997; BITTENCOURT, 1998).

Hoje, numa análise *post mortem*, se descobriu que muitos dos problemas que os seres humanos resolvem não podem ser descritos de tal forma, como a busca exaustiva num espaço de soluções usada pela IBM, cujas bases já estavam propostas por Newell e Simon em seu GPS. Esse foi o outro avanço dado pela IA que agora tenta entender como o cérebro humano *modela* os problemas que resolve, e não focalizar somente na solução de problemas abstratos. Há

também a dificuldade de que nem sempre o ser humano tem certeza dos objetivos que almeja, e mesmo assim vive o seu dia a dia com muita coerência (ao invés de, por exemplo, entrar e sair de casa várias vezes relutando em ir para o trabalho, pois gostaria de assistir a partida de futebol beneficente do seu time preferido).

Outra crítica feita a Newell e Simon foi que o GPS não tinha a capacidade de aprender novas heurísticas, mas só trabalhava seguindo as regras que já estavam programadas. É fato notório que o ser humano quando defrontado com um problema desconhecido experimenta técnicas até então desconhecidas para buscar a sua solução. A própria solução é desconhecida (ou não se precisaria buscá-la) e, portanto, deve haver ainda algum critério que diga que a solução foi alcançada quando alguma situação surgir da bricolagem feita pelas diversas tentativas de achá-la. Reconhecer uma solução em tempo real, utilizando critérios de relevância, é uma capacidade impressionante, e admitir isso é um avanço.

A IA hoje está mais madura, já desenvolveu muitas técnicas para o processamento simbólico de informação que simula o comportamento inteligente. Bittencourt (1998, p.52) divide a Inteligência Artificial em três grandes fases: (1) A IA Clássica, de 1956 a 1970, cujo objetivo era simplesmente simular a inteligência humana, e deve seu fracasso à subestimação da complexidade computacional dos problemas envolvidos. (2) A IA Romântica, de 1970 a 1980, cujo objetivo era simular a inteligência humana em situações específicas. Seu fracasso se deve à subestimação da quantidade de conhecimento necessária para tratar os problemas mais simples. (3) A IA Moderna, de 1980 a 1990, cujo objetivo era a criação de sistemas especialistas. Os sistemas

especialistas tiveram seus méritos, e só não avançaram mais devido à dificuldade de resolver problemas que envolvem a aquisição de conhecimento. Assim, resta saber, quando falamos de um programa de IA, sobre o que exatamente estamos falando⁸. Veremos alguns exemplos de técnicas da IA no próximo tópico.

1.2.1 - Modelos da Inteligência Artificial

Os modelos da IA aproveitam o máximo da capacidade de processamento algorítmico e lógico das máquinas do tipo von Neumann (MVN). Os adeptos da IA utilizam um método de abordar o problema que consiste em quatro passos (RICH & KNIGHT, 1993, p.35):

- (1) Definir o problema precisamente. A definição precisa do problema não implica que o problema não possa ter cunho estatístico ou probabilístico. É necessário, entretanto, definir qual é o ponto de partida, quais são os pontos considerados soluções, e que tipo ou classe de operadores podem ser utilizados.
- (2) Analisar o problema. Levantar características importantes do problema que podem ter grande impacto sobre a capacidade de resolução do algoritmo.
- (3) Isolar e representar o conhecimento necessário à solução do problema. A representação do conhecimento é uma das tarefas mais difíceis da IA. A maneira como o conhecimento é representado na máquina influencia diretamente a possibilidade de solução do problema, e também a velocidade da resposta. Internamente à MVN, sabemos que as informações são representadas na forma de dígitos binários, mas é a representação que o algoritmo da IA utiliza (TAD) que deve ser compatível com o objeto real representado, relevante ao problema.

(4) Escolher a melhor técnica ou as melhores técnicas de solução de problemas e aplicá-la(s) àquele problema em particular.

Podemos fazer um paralelo entre esses quatro itens e aqueles apontados por Polya (1978) para a procura da solução de um problema qualquer ou mais especificamente, de um problema matemático. Polya aponta quatro fases nesta busca, respectivamente: (i) Compreensão do problema. Para bem se compreender um problema é necessário “[...] identificar as partes principais do problema, a incógnita, os dados, a condicionante” (POLYA, 1978, p.4). (ii) Estabelecimento de um plano. O estabelecimento de um plano só é conseguido após a compreensão do problema e consiste em buscar idéias e situações semelhantes já conhecidas, ou reformular o problema através da generalização, da particularização ou da analogia. (iii) Execução de um plano. O necessário nesta fase é paciência e tenacidade para realizar o plano concebido, realizando, a cada passo, uma averiguação dos resultados parciais obtidos. (iv) Retrospecto. Reavaliar todo o problema, sua compreensão, os planos concebidos que, por ventura, falharam, e por que falharam, e o resultado final.

Ao nosso ver, o item (1) da abordagem da IA se enquadra no item (i) de Polya, como também o item (2) no (ii). Os itens (3) e (4), entretanto, parecem-nos refinamentos do item (ii), ficando (iii), a execução, e (iv), a verificação, sem correspondentes explicitados – o que não significa que a abordagem da IA deixe de executar e verificar seus algoritmos, mas, sim, que ela enfatiza a construção de um bom plano para a resolução de problemas difíceis como a simulação da inteligência humana. Nos parece clara a falta de ênfase nos

itens (iii) e (iv), na abordagem da IA, quando compreendemos que, na seqüência de resolução que Polya nos apresenta, só é possível a realização da execução de um plano quando o temos. Segundo Polya⁹ temos “um plano quando conhecemos, pelo menos de um modo geral, quais as contas, os cálculos ou os desenhos que precisamos executar para obter a incógnita”. Assim, a IA se encontra ainda nas duas primeiras fases: compreender o problema e conceber um plano.

Esses quatro itens da abordagem da IA definem o seu paradigma dominante (seja na IA clássica ou moderna), que é o da *busca de soluções em um espaço de estados* (HONAVAR, 1995, p.10). Em problemas não-triviais, uma busca cega, exaustiva, se torna inviável. Nesse caso utilizam-se heurísticas que tendem a dar um bom resultado, mas nem sempre o ótimo.

Um exemplo bastante conhecido da Ciência Cognitiva é o problema clássico do caixeiro viajante. Esse problema consiste em determinar o itinerário de um caixeiro para que sua viagem seja a menor possível em quilômetros e inclua uma e somente uma visita a todas as cidades que irá percorrer. Dependendo do número de cidades esse problema se torna extremamente complexo e uma busca por todas as possibilidades seria impensável. Uma heurística típica para se conseguir uma boa solução entre as soluções possíveis seria dada por meio de uma regra (RICH & KNIGHT, 1993, p.49):

REGRA 1- “Escolha como próxima cidade aquela de menor distância em relação à cidade atual”.

Podemos notar que a utilização de regras tão rígidas torna possível prever com certa facilidade qual será a resposta dada para um conjunto específico de cidades. É possível melhorar a resposta desse algoritmo se modificarmos a regra acima e incluirmos uma nova:

REGRA 1- “Escolha como próxima cidade uma entre as três cidades mais próximas da origem”.

REGRA 2- “Combine os vários resultados da REGRA 1 e separe aquele que apresentar menor valor total em quilômetros”.

O item (1) descrito acima – da abordagem da IA – definiria, no problema do caixeiro viajante, como as distâncias entre as cidades seriam armazenadas (seja como matrizes, regras, relatos, descrições físicas) e qual é a solução requerida (a soma da distância total percorrida pelo viajante). O item (2), quando analisado, nos revelaria a presença de heurísticas que poderiam ser implementadas na forma das regras escritas acima, para facilitar a busca pela solução. O item (4) indicaria se a melhor solução seria tentar uma busca em profundidade (que significa sair do estado inicial e caminhar até o estado final, avaliando o resultado), uma busca em amplitude (que significa explorar todas as possibilidades para o próximo passo, ao sair do estado inicial, escolhendo vagarosamente o caminho até o estado final) ou uma combinação das duas. Um outro algoritmo muito utilizado na IA, chamado de “SUBIDA DE ENCOSTA”, pode ser assim descrito (RICH & KNIGHT, 1993, p.81):

1. Avalie o estado inicial. Se for uma solução pare.

2. Repita este passo até encontrar uma solução ou até que não haja operador disponível:

(A) Escolha um operador que não tenha sido usado e aplique-o ao estado corrente para produzir um novo estado.

(B) Avalie o novo estado.

(I) Se for uma solução, pare.

(II) Se estiver mais próximo da solução que o estado corrente, torne-o o estado corrente.

(II) Se não for melhor que o estado corrente ignore-o.

(III) Repita o passo 2.

A combinação desse algoritmo com as regras heurísticas vistas acima tornam o problema de busca de solução em espaço de estados mais simples. Mas existem problemas que, mesmo com a inclusão de várias técnicas heurísticas, são muito complexos para serem resolvidos através dessa abordagem. Isso se deve à dificuldade de realizar o item (1) descrito acima, a definição do problema, e o item (3), a representação do conhecimento necessário à resolução do problema.

Existem várias formas de representar informações na IA (área conhecida pelo jargão de *representação do conhecimento*), das quais citaremos três (BITTENCOURT, 1998, p.64; RICH & KNIGHT, 1993):

1- Representação do conhecimento usando a lógica de predicados clássica:

Os fatos são representados como proposições da lógicas de predicados clássica.

Para representar, por exemplo, o fato que “Todo homem é mortal”, usamos:

$\forall x: \text{HOMEM}(x) \rightarrow \text{MORTAL}(X)$

Para representar o fato que “César é homem”, incluímos, na base de conhecimento, a proposição:

$\text{HOMEM}(\text{César})$

O próprio mecanismo de inferência, já pré-estabelecido, se encarrega de deduzir o fato:

$\text{MORTAL}(\text{César})$, que significa que César é mortal.

O mecanismo de inferência é um tipo de ferramenta de busca em espaço de estados, semelhante ao algoritmo de “SUBIDA DE ENCOSTA” mostrado acima, com a diferença de que, ao invés de procurar por estados mais próximos da solução, procura por fatos possíveis de serem deduzidos da base de conhecimento representada através da lógica de predicados clássica.

A representação do conhecimento de forma estática, como uma base de dados, é chamada de representação *declarativa*. Se os próprios fatos representados na base de conhecimento informarem, além de seu conteúdo, o modo como devem ser utilizados, teremos a representação *procedimental* (RICH & KNIGHT, 1993).

2- Representação do conhecimento por meio de regras gerais:

O que diferencia uma representação realizada por meio de lógica de predicados clássica de uma representação por meio de regras é a característica *procedimental* que esta possui. As regras gerais não apenas informam fatos sobre o mundo, mas quando e como utilizar esses fatos. Um programa clássico da IA que utiliza regras é o ELIZA de Weizenbaum, que simula uma entrevista de um

psicólogo rogeriano (RICH & KNIGHT, 1993, p.216). Vejamos algumas regras que o ELIZA segue no decorrer de seu processamento:

(X eu Y) → (X você Y?)

(Eu lembro X) → (Por que você lembra X exatamente agora?)

(Meu {membro-família} é Y) → (Quem mais em sua família é Y?)

(X {membro-família} Y) → (Fale mais sobre sua família.)

(Você é X) → (Vamos falar sobre você, não sobre mim.)

Apesar do exemplo do programa ELIZA parecer simplório à primeira vista, foi utilizando os mesmos conceitos que se criaram sistemas médicos especialistas capazes de ajudar no diagnóstico de vários tipos de patologias de difícil percepção. Muitas vezes é necessário indicar uma probabilidade de que uma regra se aplica em certos contextos, para os sistemas da IA não passem a falsa informação de que têm certeza do que estão propondo. Veremos o terceiro tipo de representação do conhecimento.

3- Raciocínio estatístico:

Entre os muitos modelos de raciocínio estatístico, o mais simples (RICH & KNIGHT, 1993, p.271) liga duas medidas estatísticas a cada regra: a medida de certeza (MC) e a medida de dúvida (MD).

$MC[h, e_1]$ - é um valor entre $[0,1]$ que indica até que ponto a evidência e_1 sustenta a hipótese h . Se $MC = 0$, a evidência não sustenta a hipótese.

$MD[h, e_2]$ - é um valor entre $[0,1]$ que indica até que ponto a evidência e_2 sustenta a negação da hipótese h . Se $MD = 0$, a evidência sustenta a hipótese.

O fator de certeza $FC[h,e]$ é dado então pela diferença entre MC e MD:

$$FC[h,e] = MC[h,e] - MD[h,e]$$

Com essa ferramenta, as regras são aplicadas à medida que um limiar pré-estabelecido seja atingido (que pode ser quando a certeza for maior que 50% ou 95%, por exemplo). O resultado de uma consulta a um sistema especialista que utiliza o raciocínio estatístico é mais confiável que os dos outros dois tipos de representação apresentados, pois vem associado a um valor que indica a possibilidade de erro do sistema. Com algum ajuste e uma realimentação dos resultados indicados pelo sistema com o fenômeno real ocorrido é possível alterar essas probabilidades associadas às regras produzindo um tipo de aprendizagem em pequeno grau.

Criar um programa que aprende é o sonho do cientista da IA, uma vez que a atividade bem sucedida de resolução de problemas está diretamente relacionada à quantidade de conhecimento de que se dispõe acerca do problema. Em vista da dificuldade de inserir, manualmente, todo o conhecimento necessário à solução das diversas tarefas que um sistema inteligente teria que resolver, torna-se imprescindível a criação de um algoritmo que aprenda este conhecimento por seus próprios meios. Existem várias técnicas de aprendizagem simbólica na IA, cada uma delas com suas vantagens e desvantagens. Vimos como se implementa um mecanismo simples de aprendizado no raciocínio estatístico acima, modificando as probabilidades associadas a cada regra. No raciocínio por meio de regras isso pode ser feito adicionando-se novas regras à base de conhecimento, conforme a iteração do programa com o meio for revelando fatos importantes,

como também eliminando regras desnecessárias ou de pouca utilidade. No raciocínio que usa representação por meio de lógica de predicados clássica, pode-se incluir, na base de conhecimento, regras que foram deduzidas das anteriores ou que podem ser inferidas após um número repetitivo de aplicação de uma sequência de regras quaisquer (indução).

Um tipo especial de aprendizagem, chamado de treinamento por exemplos, possui uma analogia muito forte com o conceito de aprendizagem em RNA. O treinamento por exemplos consiste em classificar corretamente os vários objetos que são apresentados ao sistema, diferenciando nestes objetos detalhes em sua forma ou estrutura que definiriam a que classe eles pertencem. As classes podem ser definidas por uma função do tipo:

$$C_A = p_1t_1 + p_2t_2 + p_3t_3 + \dots + p_nt_n, \text{ onde:}$$

C_A é a classe A;

t_i é um parâmetro que corresponde a uma característica relevante da classe A;

p_j é um peso que indica quão relevante é o parâmetro t_j para a classificação do objeto apresentado como pertencentes à classe A. Pesos negativos indicam características cuja presença é evidência contrária à classificação de determinado objeto como pertencentes à classe A (RICH & KNIGHT, 1993, p.527).

A modificação desses parâmetros ocorre durante a apresentação dos objetos que serão usados como exemplos das suas correspondentes classes. Um dos grandes empecilhos deste tipo de aprendizagem está em definir as classes com estas funções, pois normalmente existem grandes quantidades de

características necessárias à sua descrição e os pesos de cada característica dificilmente são conhecidos.

Uma técnica para contornar a dificuldade de criar classes é defini-las como estruturas simbólicas. O programa de aprendizado do mundo dos blocos de Winston (WINSTON, 1984, p.13 et seq.), por exemplo, consegue associar conceitos que diferenciam as montagens feitas com blocos que simbolizam “arcos”, “tendas”, “torres”, etc. Tais montagens ficam descritas na forma de estruturas simbólicas que podem ser alteradas caso novos exemplos invalidem a descrição atual. Por exemplo, a definição de “casa” poderia ser:

CASA: tem(pirâmide) e tem(cubo) e sustentado-por(pirâmide, cubo).

Caso se apresentasse um novo exemplo de casa em que o seu telhado não fosse mais uma pirâmide, porém um trapézio, teríamos o aprendizado refletido na modificação automática da estrutura simbólica para:

CASA: tem(X) e é-um(X, pirâmide) ou é-um(X, trapézio) e tem(cubo) e suportado-por(X, cubo).

Esperamos com essa explanação termos dado uma visão geral do que são os modelos da IA. A cada dia surgem novas técnicas, formas de representação, algoritmos refinados de busca em espaço de estados e novas heurísticas. A IA não se prende a nenhuma delas, mas busca uma combinação ideal, um modelo geral que finalmente simule (e por isso explique) a inteligência.

Em contraposição ao modo como a IA se coloca diante do problema de entender os processos cognitivos existem outras vertentes, entre as quais destacamos duas: as Redes Neurais Artificiais e a Ciência Cognitiva

Dinâmica. A Ciência Cognitiva Dinâmica trata de sistemas físicos, não necessariamente representacionistas ou simbólicos. Apesar de discutirmos no quarto capítulo alguns pontos que fazem parte do conjunto de conceitos dessa vertente, tais conceitos somente nos interessarão aqui por serem utilizados pelos conexionistas para apoiar as RNA. Van Gelder (1997, p.439-40), por exemplo, classifica as Redes Neurais Artificiais como um sistema instável que fica entre a abordagem computacional e a abordagem dinâmica no estudo da mente, e por isso mantém pontos em comum com ambas. Veremos agora uma introdução histórica das RNA, alguns dos avanços mais marcantes e uma análise de seus pressupostos.

1.3 - Como surgiu a abordagem das Redes Neurais Artificiais

Com o artigo de McCulloch e Pitts (1943), sobre a possibilidade de uma rede de neurônios formais calcular proposições lógicas, se abre uma enorme perspectiva para o entendimento do cérebro através de modelos computacionais conexionistas, entre eles os de redes neurais artificiais. Diferentemente da IA, que estaria mais ligada à psicologia teórica, a rede neural artificial está em contato direto com os avanços da neurologia. Seus pesquisadores têm como ambição entender o funcionamento do cérebro e a partir daí criar um modelo matemático para tentar implementá-lo tanto em uma máquina conexionista real (física), como em simulações realizadas nas máquinas de von Neumann.



Também no connexionismo temos a hipótese mecanicista, explanada anteriormente. Já a hipótese funcionalista e a hipótese representacionista merecem nova abordagem no paradigma connexionista.

Apesar de, no início, os connexionistas resistirem a admitir o funcionalismo, com a intenção de tentar propor uma abordagem completamente nova do problema da mente inteligente mecânica, julgamos que essa resistência foi apenas parcial, fruto de desentendimento do alcance das próprias propostas que os connexionistas apresentavam. Argumentaremos neste trabalho que as implicações filosóficas, tanto dos adeptos da IA quanto dos adeptos das RNA, sempre foram relegadas a um segundo plano, talvez por ingenuidade ou desconhecimento (DREYFUS, 1975). Não basta negarmos, demagogicamente, o fim de uma ideologia considerada infrutífera, se continuarmos a utilizar seus pressupostos. Entender o cérebro, com base no estudo de simplificações matemáticas de neurônios, cujas funções deveriam ser modeladas em uma rede para que se pudesse apreciar o surgimento de inteligência, é um tipo de funcionalismo, que foi chamado de funcionalismo neurocomputacional (GONZALEZ, 1989).

Já a representação mental, numa rede de neurônios formais, é agora vista como padrões estáveis de informação formados num dado momento na rede. Isso torna a representação de um fenômeno mental algo diferente do que podemos conceber através da noção de representação da IA. Em nenhuma das duas vertentes, é bom saber, existe uma cópia do objeto real no objeto representado. Enquanto a IA usa Tipos Abstratos de Dados (TAD) para

representar o objeto de estudos da melhor maneira para que o computador possa trabalhar (levando em conta a economia de memória, a velocidade de cálculo, a possibilidade de recombinação entre diversos objetos representados), a representação das RNA é feita por padrões de conexões distintos entre seus neurônios símiles (também chamados de unidades de processamento, nós, nodos, ou neurodos¹⁰) para representar objetos distintos.

Também as noções de *percepção, pensamento e aprendizagem* estão vinculadas ao desenvolvimento de alterações dinâmicas dos pesos (e.g. aritméticos) das conexões entre neurodos (o equivalente às sinapses dos neurônios biológicos).

A neurociência hoje está revendo a posição de enfatizar as sinapses em detrimento do metabolismo interno de cada neurônio, assim como está revendo o papel do axônio como único transmissor do pulso elétrico, uma vez que já há estudos indicando a possibilidade dos dendritos retropropagarem estes pulsos, o que trará conseqüências para os novos modelos de redes neurais artificiais.^{11, 12}

Uma das críticas que se tem feito ao connexionismo é a sua tendência de considerar os neurônios como elementos do tipo tudo-ou-nada, pois a informação está estritamente contida na comunicação (ou não) dos pulsos elétricos entre eles. Wiener previu esse erro de modelagem quando disse:

É, portanto, necessário que, ao considerar o problema da aprendizagem, sejamos deveras precavidos no que concerne à adoção de uma teoria do tipo tudo-ou-nada para o sistema nervoso: cumpre, antes, fazer uma crítica intelectual dessa noção e

dispor de comprovação experimental específica que fundamente nossa adoção.

(WIENER, 1954, p.65-66)

A tarefa da rede diz-se concluída quando, dada uma *entrada* para a rede, a rede estabiliza (ou relaxa), após uma série de iterações, num padrão qualquer, sendo esse padrão reconhecido como o padrão de resposta ou *saída* da rede. No caso de redes que são projetadas de modo a não se estabilizarem, como as redes recorrentes, é durante o próprio processo de iteração que as respostas da rede são colhidas.

A abordagem connexionista do problema de explicar a natureza da inteligência é sobremaneira diferente da IA. Não se tenta aqui determinar os comportamentos inteligentes através de regras predeterminadas que devem ser seguidas numa seqüência lógica, mas tenciona-se que eles emergirão das interações de elementos simples num processo guiado por leis não determinantes. Essa abordagem, do simples para o complexo, parece mais condizente com a teoria evolucionista do que a abordagem da IA. Ela é chamada de abordagem *bottom-up*, pois se espera que construindo a base (a rede neural devidamente modelada) da máquina, algum comportamento inteligente complexo emergirá sem imposição externa (MCCLELLAND & RUMELHART, 1987).

Os avanços das redes neurais artificiais nos seus primórdios estavam diretamente relacionados com os esforços de cientistas interessados no problema da percepção visual. Um dos primeiros trabalhos que se seguiram com a idéia de neurônios formais dada por McCulloch foi o *Perceptron* de Rosenblatt (1962), no fim da década de cinquenta.



O Perceptron é uma rede particular de McCulloch-Pitts, estruturada de tal maneira que a primeira camada recebe sinais de sensores fotoelétricos que repassam para as camadas intermediárias, que por sua vez realizam alguns cálculos e enviam os resultados para a camada de saída, onde se pode ver o resultado da rede. No Perceptron, somente as conexões entre a camada intermediária e a camada de saída têm capacidade para o aprendizado. Por esse motivo ele ficou conhecido como perceptron de uma camada.

Minsky e Papert (1988) acabaram por desacreditar esse tipo de rede por alguns anos (de 1968 ao início dos anos 70, quando voltaram a aparecer alguns trabalhos interessantes) com a publicação de um livro intitulado *Perceptrons*, em que demonstravam as limitações que tais redes possuíam.

Os trabalhos foram retomados com uma nova perspectiva a partir da década de 70, com a tentativa de conciliar os estudos das redes neurais artificiais com a teoria da auto-organização dos sistemas biológicos, que iniciou o neoconexionismo. Na França esses trabalhos foram guiados por Atlan, nos Estados Unidos por Kauffman e no Chile a escola de autopoiese fez avanços com Maturana e Varela.

Os Perceptrons ganharam uma nova linha de pesquisa mais complexa com a corrente de estudos de Processos Paralelos Distribuídos (do inglês, PDP). Uma outra linha, dirigida por físicos, que estuda as propriedades emergentes das redes de autômatos, foi fundada por Hopfield, denominada *Attractor Neural Network*¹³ (DUPUY, 1995, p.132-3).

Todos esses pequenos – mas significativos – avanços mostram que o estudo das redes neurais artificiais está ainda engatinhando, embora algumas críticas a este campo já estejam maduras. Gardner (1996, p.313) comenta que mesmo quando uma cena é perfeitamente analisada por um programa de redes neurais artificiais, surge um sentimento de que não existe ali naquele momento nada mais do que um truque inteligente de engenharia, e não mecanismos psicológicos. O problema de explicar o plano psicológico da mente utilizando redes neurais artificiais, tema de muito debate (RUMELHART & MCCLELLAND, 1989; BROADBENT, 1985; FODOR & PYLYSHYN, 1997), será retomado no capítulo 4.

Vimos nesta seção e na seção anterior, como modelos da IA e da RNA, criados para entender os sistemas cognitivos, lidam com os pressupostos mecanicistas, funcionalistas e representacionistas da mente. Acompanhamos também alguns dos avanços históricos mais significativos no desenvolvimento de ambas as áreas. Tivemos ainda a oportunidade de apreciar as técnicas utilizadas pela IA para a modelagem da cognição. Iremos agora averiguar o papel dos modelos na ciência, e em especial chamaremos a atenção para *como* comparar as semelhanças e distinções dos modelos da IA e da RNA. A descrição de alguns modelos específicos de RNA foi deixada para o terceiro capítulo.

1.4 - A construção dos modelos cognitivos

O entendimento dos fenômenos físicos a partir de uma perspectiva científica inclui a criação de modelos que possam, pela sua análise,



demonstrar capacidade de previsibilidade. Uma das características importantes da ciência é a capacidade de fabricação de objetos na forma de modelos. O paradigma científico de análise e explicação dos fenômenos através de modelos é tão forte que chega a ser verdade incontestável (DUPUY, 1995, p.22). Mas o que é um modelo? Dupuy¹⁴ aponta uma discrepância entre o termo utilizado no senso comum e seu significado na ciência. De acordo com o senso comum, um modelo é aquilo que imita ou simula um fenômeno, com o intuito de simplificar sua posterior realização, como uma maquete prepara o empreendedor para a construção de um prédio. Uma inversão se dá na ciência, posto que o modelo é exatamente o que a ciência faz. O fenômeno em si interessa apenas para falsificar um modelo ou dar-lhe melhores características, mas é o modelo que explica um fenômeno. No modelo se encontram as partes discriminadas de um sistema de relações que exibem, mesmo que insuficientemente, devido às simplificações necessárias, as propriedades que se quer explicar dos fenômenos. Um fenômeno, sem um modelo, é um mistério.

Não parece ser mistério a existência do modelo sem o fenômeno. Existem áreas científicas que lidam exclusivamente com modelos: modelos puros, desvinculados da realidade fenomenal, cujo estudo de suas características se mostra extremamente enriquecedor, como também o estudo da própria capacidade mental da concepção dos modelos. Van Gelder (1991, p.432) referenciando-se aos sistemas dinâmicos define duas áreas de atuação: *modelagem dinâmica e teoria dos sistemas dinâmicos*. Para a área de modelagem dinâmica, interessa entender os fenômenos naturais como comportamento de sistemas



dinâmicos naturais, utilizando modelos de sistemas dinâmicos como ferramentas aproximadas. Na teoria dos sistemas dinâmicos, estudam-se os sistemas abstratos, os modelos, sua estrutura e dinâmica.

A explicação da capacidade humana de modelar um objeto que se percebe para poder compreendê-lo é o que se busca na ciência cognitiva: um sistema cognitivo capaz de modelar o processo de modelagem. Turing (1950) já propunha essa empreitada em outros termos (a simulação da simulação) no teste de Turing, em que uma máquina não deveria somente simular um comportamento humano padronizado, mas ser ela mesma capaz de dissimular, de enganar o interrogador.

Se os modelos são fabricados para explicar os objetos, diferentes objetos são modelados de diferentes modos (e mesmo um único objeto pode ser modelado de diversas formas). Nesse sentido, a IA cria seus modelos analisando eventos psicológicos, enquanto as RNA modelam eventos fisiológicos. O objetivo, no entanto, que cada uma quer alcançar continua o mesmo: como os modelos criados explicam a natureza da mente inteligente?

Neste ponto já podemos perceber como a abordagem de um mesmo problema através de diferentes pontos de vista, criando modelos completamente diferentes, pode tornar-se de difícil escrutínio se não possuímos uma forma adequada de comparar esses modelos. A análise dos fenômenos naturais em planos distintos, com a conseguinte criação de modelos explicativos que, forçosamente, estão em planos distintos de análise, obriga o cientista a situar as suas explicações de cada modelo em seus respectivos planos de análise.



Um avanço, digno de nota, realizado nesse contexto investigativo, veio de um neurofisiologista, Marr (1982), que compreendeu que um estudo direcionado apenas ao cérebro (mais especificamente aos neurônios) não iria nos levar a entender o seu funcionamento global. Ele propôs para isso o que hoje é chamado de *Os Três Níveis de Marr* (*Marr's three levels*): em primeiro lugar, a investigação de um objeto de estudos necessita de uma teoria computacional que explique o funcionamento desse objeto e o objetivo da computação nela envolvida; em segundo lugar viria um algoritmo (que pode ser entendido como um algoritmo seqüencial ou paralelo, desde que se adapte à teoria computacional escolhida) e o tipo apropriado de representação; e por último a implementação, na qual se escolhe qual a arquitetura de computador que melhor se adapta aos dois planos anteriores (esses planos serão mais bem explanados no capítulo 4).

A abordagem feita pelos connexionistas vista do prisma desses planos, abarca-os como segue: o objetivo sempre existiu, é entender a mente inteligente, fundamentado na hipótese mecanicista; o algoritmo, aqui chamado de modelo, está ainda em evolução, começando com Rosenblatt (1962) apresentando o *Perceptron*, e ainda antes com Hebb (1949), com sua regra para modificações sinápticas. Para o último nível, o implementacional, pouca importância foi dada, como se os modelos abstratos pudessem sobreviver por si mesmos. Isso se percebe na hora de corroborar os modelos, uma vez que muitos dos adeptos das RNA não vêem problemas em utilizar as máquinas de Turing para suas simulações.

Julgamos que a RNA e a IA são duas abordagens distintas quando entendemos que a IA primeiro se apegou ao modelo de máquina – a máquina de Turing materializada na arquitetura de von Neumann – e então se pôs a produzir os diversos tipos de algoritmos inteligentes que tal máquina suportava com a intenção de, uma vez funcionando o algoritmo (ou seja, simulando processos inteligentes), conseguir tirar conclusões sobre a teoria computacional que rege o cérebro. Já a RNA primeiro pretende modelar as características físicas relevantes do cérebro em um modelo matemático de rede neural artificial para, então, implementar esse modelo em uma máquina conexionista, e só depois observar a emergência espontânea de inteligência nessa máquina, o que comprovaria que o modelo de máquina construído é adequado. Franklin (1995, p.163, nota 5), por exemplo, que está envolvido em um projeto chamado AMNIAC para a criação de um neurocomputador de propósito geral, coloca:

Also, keep in mind that until now, all connectionist models have been simulated on digital computers. Neural network chips are now on the market, with neural network boards and even neurocomputers not far behind. (FRANKLIN, 1995, p.143)¹⁵

Neste momento se coloca a questão sobre o que acontece com a teoria das RNA quando os seus adeptos, ao invés de criarem máquinas para testar as hipóteses de seus modelos matemáticos, implementam diretamente esses modelos nos computadores digitais, de forma a se utilizar da vantagem das simulações como meio de diminuir despesas, de aumentar a velocidade da construção de modelos hipotéticos (já que agora não se trata mais de manusear

peças e chips, mas apenas símbolos), sem ter a necessidade de laboratórios eletrônicos especializados, aproveitando a enorme facilidade de acesso a computadores digitais que se tem nos grandes laboratórios. Mas, mais do que isso, dado que algumas máquinas conexionistas são fabricadas para testar os modelos, elas produzem respostas diferentes dos modelos simulados? Existem dois itens que precisamos analisar separadamente: os modelos e as máquinas. Indagamos, então, se os modelos matemáticos das RNA são Turing-computáveis e, também, se as máquinas conexionistas são casos particulares das máquinas de Turing.

Responder a essas perguntas, e a outras que nos serão colocadas durante este trabalho, não é tarefa trivial e devemos nos acautelar para não nos precipitarmos a uma resposta que pode ser fruto de interpretação errônea do problema. Tentaremos agora abordar essas questões em duas partes: primeiro colocando as RNA, conceituando-as; e em seguida analisando como (e se) as máquinas conexionistas diferem das máquinas de Turing.



1.5 - Notas

¹ Tradução nossa: A inteligência artificial é o estudo de problemas de processamento de informação complexos, geralmente com raízes em aspectos do processamento de informação biológico. O objetivo desta área é identificar problemas de processamento de informação interessantes e solúveis, e solve-los.

² Como por exemplo, para citar algumas, a pesquisa em profundidade, a pesquisa em amplitude, as possibilidades amplas das regras *se-então*, etc.

³ Searle chama, também, de cognitivismo a visão de que o cérebro é um tipo de computador digital.

⁴ É um tipo de heurística. Existem outras que não a busca em árvore, como por exemplo o relaxamento paralelo de diversas restrições, utilizado pelo conexionismo.

⁵ Ibidem.

⁶ Este computador tinha 256 co-processadores operando em paralelo, com uma capacidade de avaliar 200 milhões de lances por segundo. Kasparov, ao final do torneio profetizou: "Sou o último ser humano campeão de xadrez" (BITTENCOURT, 1998, P.45).

⁷ Apesar do resultado brilhante, o Deep Blue ainda não pode ser considerado campeão mundial de xadrez, pois não participou das fases eliminatórias para ter o direito de desafiar Garry Kasparov.

⁸ A Inteligência Artificial (assim como as outras vertentes da Ciência Cognitiva) continua em sua rápida evolução. Existem novos conceitos utilizados pela IA na década de 90 em diante, como por exemplo a utilização conjunta de agentes inteligentes em interação com o meio e entre si, que não foram analisados neste trabalho.

⁹ Ibidem, p.5.

¹⁰ O termo neurodo (*neurode*) é utilizado por Caudill & Butler (1992, p.3). Preferimos esse termo entre outros, para referenciar as unidades de processamento das RNA e diferenciá-las dos neurônios, indicando sua similaridade.

Capítulo 2 - Máquinas de Turing

¹¹ PEREIRA JÚNIOR, A. (Marília, UNESP). Comunicação pessoal, out. 1999.

¹² VERSHURE, P. (Campinas, UNICAMP). Comunicação pessoal, 28 mar. 2001.

¹³ Tradução nossa: atratores em redes neurais.

¹⁴ Ibidem.

¹⁵ Tradução nossa: Também, tenha em mente que até agora todos os modelos conexionistas têm sido simulados em computadores digitais. Chips de redes neurais estão agora no mercado, e placas de redes neurais e mesmo neurocomputadores estarão em breve.

2.1 - Máquinas de Turing

Capítulo 2 - Máquinas de Turing

2.1 - Apresentação

Introduziremos neste capítulo os conceitos que serão necessários para a avaliação da possível existência de auto-organização nas máquinas de Turing, quando simulando redes neurais artificiais. Em primeiro lugar discutiremos na seção 2.2 a formalização dessas máquinas. A discussão sobre os limites da máquina de Turing obriga, inevitavelmente, uma revisão da Tese de Turing-Church, que faremos na seção 2.3. Na seção 2.4, mostraremos a equivalência entre tipos diferentes de máquinas de Turing. Finalmente, na seção 2.5, examinaremos o conceito de computabilidade como usualmente encontrado na literatura, através das linguagens formais e autômatos.

2.2 - Máquinas de Turing

O que é um *cálculo efetivo*? O que significa dizer de um procedimento que ele é um *procedimento sistemático*? O que é, afinal, um *algoritmo*? Foram essas as perguntas que levaram Turing a formular o que mais tarde ficou conhecido como *Máquina de Turing*. Ele buscava entender, e, se possível, solucionar um problema lógico proposto por Hilbert, sobre a possibilidade de existir um algoritmo (procedimento sistemático finito ou cálculo

efetivo), capaz de determinar se uma fórmula do cálculo de predicados clássico seria demonstrável ou não, conforme ressaltamos na introdução.

O primeiro passo para resolver o problema da decisão de Hilbert foi torná-lo um problema bem definido¹, caracterizando a noção de procedimento sistemático. Assim, um *procedimento sistemático* para a resolução de uma dada tarefa é um conjunto finito de instruções, executadas por um agente ou controlador. A tarefa é dada como *insumo* ao procedimento sistemático via sua formalização em uma memória de entrada, e sua resolução é explicitada em uma memória de saída. A operação de entrada e saída é feita por uma cabeça de leitura e gravação.

Podemos dividir uma máquina de Turing em três partes (TURING, 1936, p.231):

(a) a memória de entrada e saída que é representada por uma fita, potencialmente infinita para ambos os lados, quadriculada, onde se pode, em cada quadriculado, ler e escrever um símbolo;

(b) uma cabeça de leitura e gravação sobre essa fita, capaz de alterar e recuperar símbolos, além de se mover para a direita e para a esquerda um quadriculado por vez (ou permanecer em sua posição);

(c) um controlador, que se encontra, a cada passo discreto, em um estado (ou situação) interno distinto, atuando conforme instruções, que são determinadas para todas as combinações de estado e símbolo. Essas instruções indicam: (i) o símbolo que deve ser gravado na fita, (ii) a direção para onde a

cabeça de leitura/gravação deve se mover e (iii) o novo estado que o controlador deve assumir (figura 1).

A máquina de Turing contém, além do mecanismo descrito acima, um suporte formal que pode ser definido por (TURING, 1936; SHANNON, 1956):

1- O conjunto finito de estados internos S_m , sendo $Q_f \in S_m$ um estado final e $Q_i \in S_m$ um estado inicial.

2- O alfabeto finito A_n (conjunto dos símbolos possíveis).

3- O conjunto finito de instruções I_{mn}

Cada instrução I_{mn} é uma quintupla da forma:

$I_{ij} = (S_i, A_j, A_k, M_h, S_f)$, onde:

I_{ij} é a instrução executada caso a máquina se encontre no estado S_i e tenha sobre sua cabeça de leitura/gravação o símbolo A_j . Uma vez determinada a instrução por essas duas variáveis, sua execução causa a gravação do símbolo A_k na posição atual da cabeça, o movimento M_h da cabeça na fita (que pode ser para esquerda, direita ou não se mover), e a passagem do controle para o novo estado S_f (TURING, 1936), conforme pode ser inferido da análise da figura 1.

A lista de instruções distintas nunca tem mais elementos que o produto do número de símbolos do alfabeto pelo número de estados internos. Sendo ambos finitos, também o é o número de instruções. A importância da existência tanto de um alfabeto quanto de um conjunto de estados finito decorre, como mostrou Turing com um argumento topológico, de que, se assim não fosse,

existiriam símbolos, ou estados, tão semelhantes entre si que se tornariam indistinguíveis. Além disso, há a necessidade da máquina ser capaz de encontrar a próxima instrução – numa lista finita de instruções – a ser executada para configurar um procedimento sistemático finito, ou não se teria conseguido o intuito inicial (qual seja, definir procedimento sistemático).

Temos, então, que um procedimento sistemático, efetivo, para a resolução de uma tarefa, nada mais é que uma máquina de Turing com o devido conjunto de instruções, projetada para resolver tal tarefa – essa é, em epítome, a Tese de Turing-Church, que veremos mais adiante. A noção de algoritmo que daí surge não está somente no conjunto de instruções, mas num conjunto *finito* de instruções coerentes com a resolução de determinada tarefa, e no fluxo de mudança de uma instrução para a próxima.

Poderíamos até, neste momento, realizar uma inversão da explanação de máquinas de Turing dada acima e explicar o seu funcionamento por meio de um procedimento efetivo, utilizando assim o explicado pelo explicador. Para que a máquina de Turing possa executar as instruções que lhe estão inseridas e assim operar sobre a fita, o controlador² deve seguir os passos dados pela tabela 1:

dois estados internos (S_1 e S_2). Logo podemos contar com seis instruções distintas (o produto de três por dois), que são as linhas da figura 1. Suponhamos o estado inicial S_1 e iniciemos a execução:

- 1- Estado atual $S_i = S_1$
- 2- Ler o símbolo $A_j = A_1$ (posição da cabeça de leitura/gravação).
- 3- Próxima instrução $I_{ij} = (S_i, A_j, A_k, M_h, S_f) = (S_1, A_1, A_2, D, S_1)$.
- 4- Escrever o símbolo $A_k = A_2$ na fita.
- 5- Mover a cabeça de leitura/gravação para a $M_h = D$, direita.
- 6- Fazer do estado da instrução, $S_f = S_1$, o estado atual S_i .
- 7- Repetir todos os passos anteriores.

Tabela 2: Execução da primeira instrução da figura 1

Depois desse primeiro ciclo (de sete passos, da tabela 2) realizado pelo controlador, que representa a execução de uma única instrução, a máquina de Turing gravou A_2 na posição inicial e moveu a cabeça para a direita, permanecendo no mesmo estado interno S_1 . Continuando analogamente, teremos, após 31 ciclos, todos os símbolos do intervalo da fita da figura 1 modificados para o símbolo A_3 . Fica mais simples entender o funcionamento dessa máquina seguindo o diagrama da figura 2:

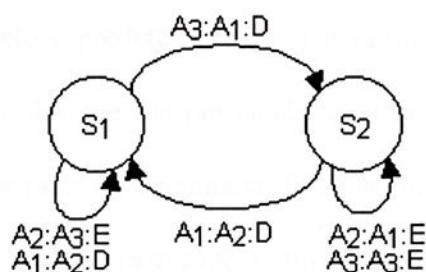


Figura 2: Diagrama da máquina de Turing representada na figura 1

O diagrama da figura 2 representa uma máquina de Turing. Os círculos representam os estados internos da máquina, sendo o de número 1 o estado inicial. As setas representam as transições de estados que são realizadas de acordo com as instruções inscritas sobre elas. Essas instruções possuem três termos, separados por (:). O primeiro é o símbolo lido na fita da máquina (A_j), o segundo é o símbolo a ser gravado na mesma posição (A_k), e o terceiro é o movimento que a cabeça de leitura e gravação deve fazer ($M_h = 'D'$ para direita e $'E'$ para esquerda). A quintupla que descreve a instrução da máquina de Turing $I_{ij}=(S_i, A_j, A_k, M_h, S_f)$, fica completa ao se integrar o estado S_i atual (o número dentro do círculo atual) e a passagem do controle para o novo estado S_f (o número dentro do próximo círculo, apontado pela seta). Devemos também indicar (veja figura 1) os dados iniciais na fita, a posição inicial da cabeça de leitura e gravação e o estado inicial, para termos uma descrição precisa.

Nesse nosso exemplo a máquina não irá interromper sua execução por motivo algum, uma vez que ela só pararia se recebesse uma combinação estado-símbolo que não estivesse pré-programada. Contudo, podemos verificar na figura 1, que todas as seis instruções possíveis foram dadas de antemão. Como o alfabeto é predeterminado, a máquina nunca irá encontrar, na fita infinita, um símbolo, durante sua permanência num estado qualquer, que não possua instrução prevista para ser executada. Pode acontecer o surgimento de um ciclo em que a cabeça de leitura/gravação fique presa em um trecho da fita, andando da esquerda para a direita e vice-versa. Podemos exemplificar esse comportamento imaginando uma máquina (veja figura 3) cuja fita inicial seja

composta pelos símbolos AB, com a cabeça de leitura/gravação posicionada inicialmente sobre A, no único estado 1, e com duas instruções, $I_{1A}=(1, A, A, D, 1)$ e $I_{1B}=(1, B, B, E, 1)$.

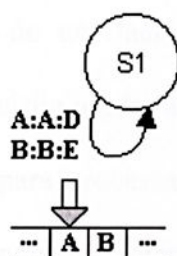


Figura 3: Máquina de Turing em um eterno ciclo

Agora, a máquina irá parar caso o controlador não consiga achar a próxima instrução, conforme o passo 3 de seu algoritmo (tabela 1). Por exemplo, suponha que na figura 1, ao invés das reticências que se encontram na fita, se encontrasse o símbolo A_4 . A máquina pararia assim que lesse o símbolo A_4 e tentasse executar qualquer de ambas as instruções iniciadas por $(S_1, A_4, _, _, _)$ ou $(S_2, A_4, _, _, _)$. Analogamente, o mesmo aconteceria se alguma instrução válida executada anteriormente remetesse a máquina para um estado inexistente (S_3 , por exemplo).

Todos esses desfechos - a falta de instrução para seguir, ou o ciclo repetitivo da cabeça de leitura/gravação - são passíveis de serem previstos e, teoricamente, evitados por via de uma programação mais cuidadosa, uma vez que a fita é dada de antemão.

A máquina abstrata de Turing sofreu, nas mãos de von Neumann, uma importante modificação que não a descaracterizou como máquina de Turing, e ficou conhecida como arquitetura de von Neumann. Von Neumann

mostrou que a máquina de Turing não precisava de duas memórias distintas para ser fabricada, uma para as instruções e outra para os dados da fita (DUPUY, 1995). Com isso, dividiu a fita infinita em duas partes, colocando as instruções codificadas em suas quintuplas de um lado da fita e os dados do outro. A arquitetura de von Neumann consistia ainda, além da memória para programas e dados, de uma unidade lógica (para processar a informação), uma unidade de controle (para organizar o funcionamento interno), e os dispositivos de entrada e saída (fios, cabos, comutadores, teletipos, teclados, impressoras, telas, etc.).

Os computadores digitais de hoje têm seu funcionamento baseado na arquitetura de von Neumann. Um computador, atualmente, utilizado nas tarefas do dia-a-dia, costuma ter, em média, uma memória volátil (RAM) de 64 Megabytes, acompanhada de um disco-rígido de aproximadamente 10 Gigabytes, o que equivaleria a uma fita com quase 100 bilhões de quadriculados ($8,6 * 10^{10}$). O alfabeto, entretanto, está limitado a ter apenas dois símbolos, que caracterizam o sistema binário. Essa limitação se deve ao funcionamento das células de memória (aqui vistos como quadriculados), que acusam a modificação de seu conteúdo pela presença ou ausência de energia, que é então interpretado pelo engenheiro como zero ou um.

Essa limitação do alfabeto a apenas dois símbolos não limita a máquina de Turing, conforme demonstrado por Shannon:

It is also possible, as we will now show, to construct a machine, C, which will act like any given Turing machine A and use only two symbols 1 and 0 on its tape, one of which, 0 say, is the symbol for a blank square. (SHANNON, 1956, p.163)³

A noção de máquina de Turing universal deriva da possibilidade de – dada uma máquina de Turing específica para realizar determinada computação – conceber-se uma máquina universal que simule a máquina específica, simplesmente codificando a fita e a descrição da máquina específica na fita da máquina universal (SHANNON, 1956, p.158). Nas próprias palavras de Turing:

It is possible to invent a single machine, which can be used to compute any computable sequence. If this machine U is supplied with a tape on the beginning of which is written the standard description (SD) of some computing machine M , then U will compute the same sequence as M . (TURING, 1936, p.241-2)⁴

Este é o Teorema de Turing (TT), que pode ser colocado da seguinte forma (FLORIDI, 2001): *Dada uma função f computável por uma máquina de Turing M , existe uma máquina de Turing universal U que simula M e computa f .*

Ou seja, dada uma máquina de Turing qualquer, existe outra mais complexa que a simula. Esse teorema afirma que não existem duas máquinas de Turing distintas que não podem ser realizadas por uma única máquina universal, isto é, são de alguma forma equivalentes: *“It will be seen that as a consequence of this all digital computers are in a sense equivalent”* (TURING, 1950, p.48)⁵.

Desde o seu início, a máquina de Turing materializada na arquitetura de von Neumann permitia, pelos dispositivos de entrada, inserir os dados na memória (ou fita de Turing) de dois modos, sendo que a evolução de um

para o outro quase não foi relatada: os primeiros computadores deviam ter seu programa e seus dados inseridos de uma só vez, no que foi chamado de *programação em lote*, para então iniciarem o processamento. A programação em lote não era uma limitação teórica do modelo, mas tecnológica.

Mesmo a máquina abstrata de Turing previa uma intervenção externa para a inserção de dados na fita, que não foi enfatizada pelos adeptos da IA Clássica, chamada por Dreyfus (1975) de “a BOa e VElha Inteligência Artificial” (BOVEIA, do inglês: *Good Old-Fashioned Artificial Intelligence – GOFAI*). Os adeptos da BOVEIA minimizaram esta intervenção feita na fita da máquina de Turing, pois acreditavam que, uma (única) vez introduzido na máquina o programa correto, ela exibiria o tão esperado comportamento inteligente. Com a evolução dos dispositivos de entrada e saída, como por exemplo, o teclado, o *mouse*, e o MODEM (que permitiu o advento da INTERNET), o computador passou a operar com dados escritos em uma fita constantemente em mudança. Essas mudanças de símbolos na fita não são feitas apenas pela cabeça de leitura/gravação como propôs Turing, mas são feitas também pelo meio-ambiente no qual o computador está inserido, ocorrendo mesmo durante a execução de um algoritmo. Esse dado novo não altera a previsibilidade da máquina em si, nem os tipos de funções que ela é capaz de computar, que continuam a ser funções Turing-computáveis, mas torna a própria fita, a memória onde opera a máquina, um local imprevisível, que pode ser alterado por diversas causas externas, além, é claro, das alterações feitas pela própria máquina.

2.3 - A calculabilidade efetiva

Ainda hoje não sabemos exatamente o alcance prático da máquina de Turing. Os computadores digitais (os PC's comuns com os quais temos maior contato) fabricados segundo a idéia da máquina abstrata de Turing têm gerado polêmica sobre seus alcances e limites até os dias atuais. Podemos, entretanto, apontar alguns resultados matemáticos.

O *problema da decisão* proposto por Hilbert foi apenas um catalisador de esforços para a tentativa de definir formalmente a noção intuitiva de calculabilidade efetiva:

[...] calculabilidade efetiva é de caráter intuitivo e refere-se a um procedimento que, dados n números naturais como entrada, termina depois de um número finito de passos proporcionando um número natural como saída. (BIRABEN, 1996, p.43)

Gödel, no começo da década de 30, preocupado em provar o que veio a ser conhecido como *Teorema da Incompletude*⁶, desenvolveu uma definição de funções recursivas gerais para apoiar seu teorema. Uma vez provado, tal teorema indicava uma solução negativa para o problema da decisão de Hilbert (BIRABEN, 1996, p.31). Mas ainda faltava uma definição formal do que seria *calculabilidade efetiva*. A definição de função recursiva geral ficou intuitivamente ligada à noção de calculabilidade efetiva justamente por causa do método utilizado por Gödel – de diagonalizar as sentenças e as seqüências de fórmulas, mostrando ser possível aritmetizar uma proposição que afirma se dada sentença é demonstrável ou não (DUPUY, 1996, p.29).

A recursão, no raciocínio matemático, tinha, com Gödel, recebido uma definição geral, e como o próprio raciocínio recursivo parecia intuitivamente um cálculo efetivo, ou seja, um procedimento finito utilizado para computar, Church arriscou propor sua tese.

Church, que trabalhava independente de Gödel, atacando um problema de fundamentação matemática, em meados da década de 30 demonstrou que uma classe de funções⁷, por ele descoberta, era equivalente às funções recursivas gerais de Gödel. Sua tese pode ser assim formulada (BIRABEN, 1996, p.42):

Tese de Church (TC): *Uma função é efetivamente calculável se e somente se é recursiva geral.*

A tese de Church é uma hipótese, pois tenta ligar um conceito formal, de recursividade geral, a uma intuição não definida de calculabilidade efetiva. Gödel, justamente por não aceitar essa ligação intuitiva, não apoiou a tese de Church, e propunha antes uma axiomatização do conceito intuitivo. Ele não duvidava da calculabilidade efetiva de todas as funções recursivas, mas sim da recíproca (BIRABEN, 1996).

Nesse contexto entra Turing e é aclamado tanto por Gödel como por Church, como quem finalmente formalizou o conceito intuitivo de calculabilidade efetiva⁸. A máquina de Turing apela para uma noção, que além de formal, é tão intuitiva quanto o objeto que quer explicar - a calculabilidade efetiva. Assim, mesmo sem axiomatizar a noção de cálculo efetivo, como queria

Gödel, a máquina Turing utiliza todas as propriedades que seriam importantes para definir a efetividade, quais sejam, um computador humano:

- (A) Operando sobre uma fita infinita.
- (B) Observando um intervalo finito desta fita por vez.
- (C) Utilizando um conjunto de símbolos finitos.
- (D) Estando em cada espaço discreto de tempo em um estado de um conjunto finito de estados possíveis.
- (E) Dando ordens discretas a cada tempo, para trocar símbolos, para trocar de estado interno e para trocar de quadriculado observado na fita.

Por ser intuitiva, por um lado, e formal por outro, a máquina de Turing pode ser entendida como uma ponte entre a noção de calculabilidade efetiva e sua definição matemática, expressa numa modificação da tese de Church, que pode ser feita uma vez que as funções recursivas gerais são equivalentes às funções Turing-computáveis:

Tese de Turing-Church (TTC): *Uma função é efetivamente calculável se e somente se é Turing-computável.*

Conforme ressalta Dupuy (1996, p.31), essa tese, como a de Church, também não pode ser provada, pois a calculabilidade mecânica (ou efetiva) é ainda um conceito vago.

Ao avaliar a capacidade de sua máquina, Turing (1936, p.254) definiu as funções que eram passíveis de serem executadas por ela, chamando-as de funções computáveis, ou funções Turing-computáveis. Gödel temia que as funções recursivas estivessem estritamente incluídas dentro do espaço de funções

de cálculo efetivo (BIRABEN, 1996, p.46), e seria uma prova interessante desse ponto de vista se as funções Turing-computáveis incluíssem, além das funções recursivas, outras não-recursivas. Mas não foi isso o que aconteceu, e Turing provou a equivalência das funções Turing-computáveis com as funções recursivas⁹.

A reivindicação de Turing, de que ele formalizou o cálculo efetivo, apesar de não poder ser provada, é de grande força intuitiva. Basta notar que para invalidar tal tese é necessário conceber um procedimento sistemático, efetivamente calculável, mecânico, que não possa ser executado pela máquina de Turing. Biraben (1996) coloca isso com outras palavras:

O fato de que todas as funções efetivamente calculáveis submetidas a teste resultassem serem lambda definíveis, não é evidência suficiente para pensar que toda função efetivamente calculável é lambda definível. (BIRABEN, 1996, p.50)

Ao considerarmos que a reivindicação de Turing foi feita antes mesmo da fabricação dos primeiros computadores, podemos perceber os motivos para tanta relutância em aceitar uma definição de calculabilidade efetiva. Hoje, porém, se admitirmos *definições por indução*, fica mais fácil de admitir que a máquina de Turing seja realmente a formalização dos procedimentos mecânicos, como queria seu autor. Claro, em toda indução, sempre há a possibilidade de surgir um contra-exemplo, mas não deixaremos de acreditar no nascimento do sol amanhã por desconfiança do raciocínio indutivo.

2.4 - Máquinas de Turing com múltiplas fitas

É possível conceber um sistema similar à máquina de Turing original, composto de uma unidade de controle que atuará em múltiplas cabeças de leitura e gravação, dispostas por sua vez sobre múltiplas fitas (figura 4A). Essa nova máquina, com capacidade de operar em paralelo escrevendo independentemente nas várias fitas, tem na sua unidade de controle poucas alterações em relação à máquina de Turing original.

As instruções, que antes eram dadas na forma de quintuplas, dependerão agora dos símbolos lidos em cada fita, e terão disponíveis os símbolos a serem gravados por cada cabeça independente. Cada cabeça pode se mover para a esquerda ou para a direita, uma única posição por vez, não importando o sentido do movimento das outras.

Teorema de Equivalência (TE): “*Dada uma máquina de Turing M , com múltiplas fitas e múltiplas cabeças, existe uma máquina de Turing M' , com uma única fita e uma única cabeça, tal que M' simula M* ” (LUCCHESI, 1979, p.111; SIMON, 1979, p.36 e p.55).

A demonstração matemática desse teorema fica fora do tema desta dissertação, mas apresentaremos a idéia principal que é bem intuitiva. Suponha a máquina M da figura 4A, com quatro fitas denominadas t_1 , t_2 , t_3 e t_4 , e alfabeto (A_1, A_2, A_3, A_4) . Então é possível criar uma máquina de Turing M' de uma única fita, com alfabeto $(A_1, A'_1, A_2, A'_2, A_3, A'_3, A_4, A'_4)$, tal que cada símbolo na fita de M' seja uma quádrupla (m -tupla para m fitas) que representa o conteúdo de todas as fitas da máquina M (figura 4B). Para identificar o símbolo

lido por uma cabeça na máquina M , denotamos na fita da máquina M' o mesmo símbolo acrescido do sinal ($'$).

Para cada passo dado pela máquina M , a máquina M' o simula seguindo o algoritmo abaixo (LUCCHESI, 1979, p.112):

Passo 1:

M' avança na fita até encontrar todos os símbolos marcados com ($'$), armazenando-os no controle.

Passo 2:

M' retorna na fita, num processo inverso, escrevendo todos os símbolos correspondentes.

Passo 3:

M' avança na fita, movendo o sinal ($'$) para a direita, de acordo com o que seria a movimentação das múltiplas cabeças da máquina M que estariam caminhando para a direita.

Passo 4:

M' retorna na fita, movendo o sinal ($'$) para a esquerda, de acordo com o que seria a movimentação das múltiplas cabeças de M que estariam caminhando para a esquerda.

Uma propriedade interessante é que, se a máquina M parar depois de t passos, a máquina M' não precisará de mais do que $4t^2$ passos para parar, dando a mesma resposta. Isso acontece porque em t passos na máquina M , cada cabeça, por se mover apenas um quadriculado por vez, pode se afastar no máximo t posições do seu ponto inicial. Logo, cada passo de M requer no máximo

$4t$ passos de M' , e a simulação toda requer no máximo $4t^2$ passos (LUCCHESI, 1979, p.113).

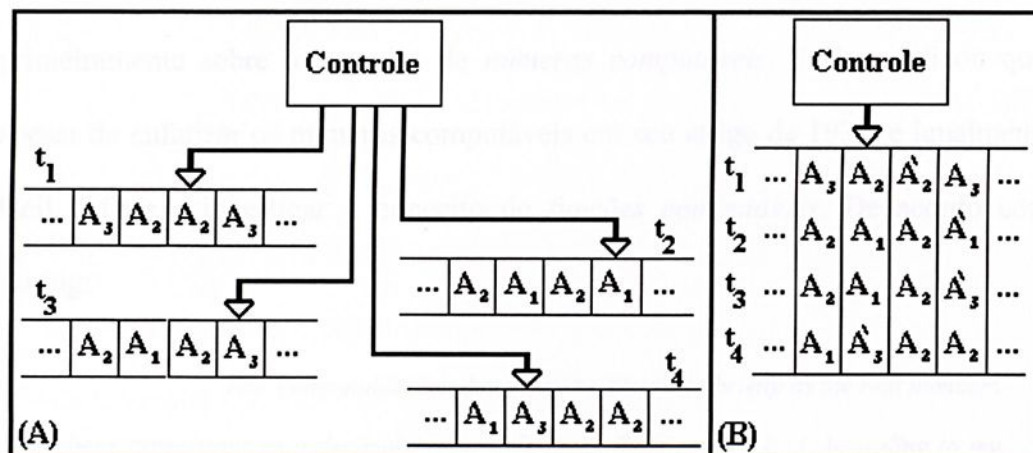


Figura 4: Equivalência entre máquinas de Turing: (A) múltiplas fitas, (B) única fita¹⁰

A equivalência entre esses dois sistemas computacionais nos será de grande importância para analisarmos os modelos de RNA. Além dessa equivalência, há outra propriedade interessante que ressaltamos: não se ganha nem se perde poder computacional tornando a fita potencialmente infinita para apenas um dos lados ao invés de ambos (SIMON, 1979, p.34).

Por enquanto é suficiente apontar que as máquinas de Turing parecem não apresentar qualquer tipo de auto-organização (AO), uma vez que seu comportamento é completamente determinado pelas instruções internas e pelos dados contidos em sua fita. Esse determinismo é proveniente das características do sistema controlador que rege a máquina seguindo passos predeterminados e não possui abertura para uma mudança em seu comportamento. Voltaremos a refletir sobre as propriedades necessárias para a AO no capítulo 5.

2.5 - Computabilidade

O termo *computabilidade* foi definido por Turing (1936), primeiramente sobre o conceito de *números computáveis*. Turing indicou que, apesar de enfatizar os números computáveis em seu artigo de 1936, é igualmente fácil definir e investigar o conceito de *funções computáveis*. De acordo com Turing:

The 'computable' numbers may be described briefly as the real numbers whose expressions as a decimal are calculable by finite means. [...] According to my definition, a number is computable if its decimal can be written down by a (Turing) machine. (TURING, 1936, p.1)¹¹

Ludermir (1992) define o conceito de computabilidade de uma *máquina* introduzindo a noção de linguagens formais: “*The computability of a machine is its ability to recognize a class of languages*” (LUDERMIR, 1992, p.262-263)¹². Introduziremos o conceito de linguagens formais, fundamentando-nos em Manna (1974), para clarificar a definição de Turing-computabilidade.

Seja A um conjunto finito de símbolos quaisquer, chamado doravante de *alfabeto*. Os elementos de A são chamados *letras* do alfabeto, ou *símbolos terminais*. Uma *palavra* do alfabeto A é uma seqüência finita de letras do alfabeto. A palavra *vazia* é denotada por Λ e consiste de uma palavra sem letras. O conjunto vazio é denotado por $\emptyset = \{\}$. A^* indica o conjunto de *todas* as palavras possíveis de se formar a partir do alfabeto A , incluindo a palavra vazia, e A^+ , idem, não incluindo a palavra vazia (MANNA, 1974, p.2).

Por exemplo, seja $A = \{0,1\}$, um alfabeto composto por dois símbolos, o zero e o um, temos:

$$A^* = \{\Lambda, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 110, 111, 0000, \dots\}$$

Uma gramática G_0 consiste de quatro objetos (MANNA, 1974, p.41):

(1) Um conjunto A finito e não vazio de símbolos, chamado *alfabeto* ou *símbolos terminais*, denotados por letras minúsculas ou numerais.

(2) Um conjunto N finito e não vazio de símbolos, chamado *símbolos não-terminais*, denotados por letras maiúsculas.

(3) Um elemento distinto I , contido no conjunto N , chamado *símbolo inicial*.

Normalmente utiliza-se a letra S .

(4) Um conjunto P de *regras de produção*, da forma $\alpha \rightarrow \beta$, onde $\alpha \in (N \cup A)^+$ e $\beta \in (N \cup A)^*$, ou seja, β é qualquer seqüência de símbolos, e α é qualquer seqüência de símbolos exceto a vazia.

A definição acima pode ser exemplificada através da seguinte situação: temos, $G'_0 = \{A, N, I, P\}$, tal que $A = \{0, 1\}$, $N = \{S, D, E\}$, $I = S$, e P , o conjunto de cinco regras de produção:

$$1- S \rightarrow D1$$

$$2- D \rightarrow D1$$

$$3- D \rightarrow E0$$

$$4- E \rightarrow E1$$

$$5- E \rightarrow \Lambda$$

O que significa dizer que:

$$G'_0 = \{\{0, 1\}, \{S, D, E\}, S, \{S \rightarrow D0, D \rightarrow D0, D \rightarrow E0, E \rightarrow E1, E \rightarrow \Lambda\}\}$$

Uma palavra w é dita pertencer a G_0 se existe uma seqüência de regras de produção, começando no símbolo inicial, que gera w . No nosso exemplo, começando com o símbolo inicial S , e substituindo-o, aplicando a regra 1, seguida das regras 2, 2, 3, 4, 4 e 5 teremos:

$$S \xrightarrow{1} D1 \xrightarrow{2} D11 \xrightarrow{2} D111 \xrightarrow{3} E0111 \xrightarrow{4} E10111 \xrightarrow{4} E110111 \xrightarrow{5} 110111$$

Alternativamente, aplicando as regras 1, 3 e 5, somente, teremos a menor palavra possível em G'_0 :

$$S \xrightarrow{1} D1 \xrightarrow{3} E01 \xrightarrow{5} 01$$

Logo, as palavras '110111' e '01' pertencem a G'_0 . Notaremos que as palavras que pertencem a G'_0 são todas da forma:

$L(G'_0) = \{1^n 01^m, \text{ tal que } n \geq 0 \text{ e } m \geq 1\}$, onde $L(G'_0)$ significa a *linguagem* gerada por G'_0 ; ou seja, o conjunto de todas as palavras pertencentes a G'_0 .

A gramática G_0 explicitada acima é classificada por Chomsky como *gramática tipo-0*, ou *gramática de frase estruturada* ou ainda *gramática irrestrita*, e a linguagem $L(G_0)$ gerada por G_0 é chamada de *linguagem recursiva* ou *linguagem tipo-0* (MANNA, 1974, p.42).

Podemos restringir as regras de produção, item (4) da definição de gramática dada, e criarmos uma nova gramática, mantendo os itens (1), (2) e (3) da gramática anterior. Essa nova gramática é chamada de *gramática tipo-1*, ou *gramática sensível ao contexto*¹³.

(4) Um conjunto P de *regras de produção*, da forma $\alpha \rightarrow \beta$, tal que $|\alpha| \leq |\beta|$, onde $\alpha \in (N \cup A)^+$ e $\beta \in (N \cup A)^*$, ou seja, β é qualquer seqüência de símbolos, α é qualquer seqüência de símbolos exceto a vazia, e a quantidade de símbolos da seqüência α é menor que a quantidade de símbolos da seqüência β .

Com essa restrição, podemos concluir que a linguagem gerada por G_1 está contida naquelas geradas por G_0 (MANNA, 1974, p.43). Restringiremos ainda mais as regras de produção, e criaremos agora uma *gramática tipo-2*, ou *gramática livre de contexto*¹⁴:

(4) Um conjunto P de *regras de produção*, da forma $\alpha \rightarrow \beta$, onde $\alpha \in N$ e $\beta \in (N \cup A)^+$, ou seja, β é qualquer seqüência de símbolos não vazia e α é um único símbolo não-terminal.

Finalmente, o último tipo de gramática, chamada de *gramática tipo-3* ou *gramática regular* é conseguido restringindo ainda mais as regras de produção, tornando¹⁵:

(4) Um conjunto P de *regras de produção*, da forma:

$\alpha \rightarrow a\beta$ ou $\alpha \rightarrow a$, onde:

$\alpha, \beta \in N$ e $a \in A$, ou seja, α e β são quaisquer símbolos não-terminais, e a é um símbolo terminal (inclui o símbolo Λ).

Uma vez explanadas as *gramáticas tipo-i*, com $0 \leq i \leq 3$, Manna (1974) define o conjunto de palavras sobre A , geradas por tais gramáticas, como a *linguagem tipo-i*, respectivamente. Podemos ainda resumir as relações de inclusão entre as classes de conjuntos sobre A (MANNA, 1974, p.43):

Linguagens Tipo-0 $\supset$ Linguagens Tipo-1 $\supset$ Linguagens Tipo-2 $\supset$ Linguagens Tipo-3.

Utilizando as linguagens formais, Manna (1974) ressalta que há três formas de escrutinar uma máquina de Turing: (1) como um *aceitador*, (2) como um *gerador* e (3) como um *algoritmo*. Como um aceitador é possível mostrar que a máquina de Turing é o dispositivo de computação *mais geral possível*, uma vez que é capaz de aceitar as *linguagens tipo-0*, geradas pelas *gramáticas irrestritas*. Como um gerador, as MT são capazes de computar as funções totalmente, ou parcialmente recursivas, e somente elas. Como um algoritmo, as máquinas de Turing resolvem (decidem), ou parcialmente resolvem uma classe de problemas do tipo SIM ou NÃO.

2.5.1 - Máquinas de Turing entendidas como aceitadores

Para entendermos o conceito de *computabilidade* parece apropriado investigar a questão do que significa dizer que uma máquina reconhece uma linguagem. Uma máquina de Turing M , definida com um alfabeto A , recebendo como entrada uma palavra w de A , pode se comportar de três maneiras: (a) *aceitar* a palavra w , e parar em um estado final denominado *aceito*; (b) *rejeitar* a palavra w , e parar em um estado final denominado *rejeito*; (c) ou não parar, caso em que se dirá que a máquina entrou em um *ciclo* eterno. O conjunto de todas as palavras aceitas pela máquina M é definido como $Aceito(M)$; o conjunto de todas as palavras rejeitadas pela máquina M é definido como $Rejeito(M)$ e o conjunto de

todas as palavras em que a máquina M entra em um ciclo eterno é definido como $Ciclo(M)$ (MANNA, 1974, p.20).

Uma máquina M reconhece uma linguagem L gerada por uma gramática G sobre um alfabeto A se, e somente se, $Aceito(M) = L(G)$, e $Rejeito(M) \cup Ciclo(M) = A^* - L(G)$. Em outras palavras, a máquina M reconhece a linguagem L se pára em um estado de *aceito* recebendo qualquer palavra da linguagem L , e no caso de todas as outras palavras possíveis do alfabeto A , a máquina pára em um estado *rejeito* ou entra em um *ciclo* eterno¹⁶.

Duas máquinas M_1 e M_2 quaisquer são ditas equivalentes se $Aceito(M_1) = Aceito(M_2)$, e claramente, $Rejeito(M_1) \cup Ciclo(M_1) = Rejeito(M_2) \cup Ciclo(M_2)$. Manna (1974) ressalta que uma classe de máquinas Ω_1 tem o mesmo *poder computacional* que uma classe de máquinas Ω_2 se para cada máquina de Ω_1 existe uma máquina equivalente em Ω_2 e vice-versa. Uma classe de máquinas Ω_1 é dita *mais poderosa* que a classe de máquinas Ω_2 se para cada máquina de Ω_2 existe uma equivalente em Ω_1 , mas não vice-versa.

É possível comprovar que a classe de *máquinas de Post* tem o mesmo poder computacional que a classe das máquinas de Turing, e também a classe de *máquinas de estados finitos com duas pilhas*¹⁷.

Para efeito de comparações, os *autômatos finitos*, que são os dispositivos de computação mais simples possíveis, são capazes de reconhecer as *linguagens tipo-3*, mas não as de outros tipos. São equivalentes às *máquinas de estados finitos sem pilha* e às *máquinas não-determinísticas de estados finitos sem pilha*. As *linguagens tipo-2* são reconhecidas pelos *autômatos com uma pilha* (que

reconhecem também as *linguagens tipo-3*). As *linguagens tipo-1* (como também as *tipo-2* e *tipo-3*) são reconhecidas pelos *autômatos linearmente limitados* (LUDERMIR, 1992, p.263; MANNA, 1974, p.37).

Como é conhecido na literatura e ressaltado por Ludermir (1992) as máquinas de Turing são *computacionalmente mais poderosas* que todos esses exemplos de dispositivos de computação, pois reconhecem, além dessas linguagens, as *linguagens tipo-0*. Na figura 5 podemos ver as relações de inclusão entre as linguagens formais:

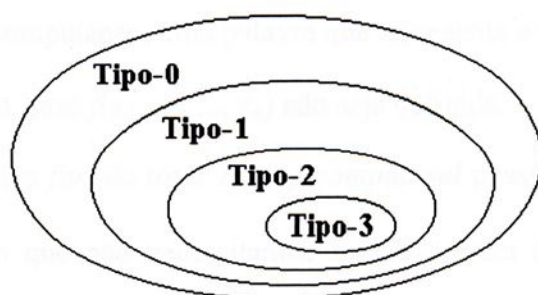


Figura 5: Uma relação entre os tipos de linguagens formais (inspirado em LUDERMIR, 1992, p.268)

Em resumo, de acordo com a definição de *computabilidade de máquinas* dada por Ludermir (1992), podemos concluir que uma máquina qualquer é equivalente à máquina de Turing e, portanto, computa as mesmas funções que uma máquina de Turing se, e somente se for capaz de reconhecer uma linguagem tipo-0.

2.5.2 - Máquinas de Turing entendidas como geradores

Uma *função parcial* n -ária é definida por Manna (1974) como uma função que mapeia n -tuplas de palavras de A em A . Esta é chamada de função

parcial pois pode ser indefinida para alguma n -tupla de palavras possíveis de A .

Uma *função total* n -ária é definida para todas as n -tuplas (MANNA, 1974).

Uma função parcial $f(x_1, x_2, \dots, x_n): A \times A \times \dots \times A \rightarrow A$ é dita uma *função Turing-computável* se existe uma máquina de Turing M , definida com um alfabeto $A + \{*\}$ que se comporte tal que (MANNA, 1974, p.44):

M recebe como entrada em sua fita uma única palavra w composta pela concatenação: $x_1 * x_2 * \dots * x_n$ e:

- (1) M pára (em *aceito* ou em *rejeito*), caso $f(x_1, x_2, \dots, x_n)$ seja definida, contendo na fita, ao final da computação, uma palavra que representa o valor de f .
- (2) M entra em *ciclo*, caso $f(x_1, x_2, \dots, x_n)$ não seja definida.

Uma *função total Turing-computável* possui a mesma definição dada acima, exceto que não necessitamos incluir o item (2), pois a função é definida para todas as n -tuplas. Nesse caso, a máquina de Turing deve parar sempre.

Entendemos a máquina de Turing como um gerador quando visualizamos que programar tal máquina nada mais é que criar um gerador de palavras, ou seja, uma gramática. Podemos criar máquinas de Turing para gerar palavras de qualquer das *linguagens tipo- i* definidas anteriormente¹⁸. Sabemos, entretanto, que nenhuma máquina de Turing específica consegue gerar uma linguagem mais abrangente que as *linguagens tipo-0*, pois:

Teorema: Um conjunto de palavras sobre A é *recursivamente enumerável* se, e somente se, é uma *linguagem tipo-0* (MANNA, 1974, p.42).

Ou seja, a gramática tipo-0 gera o mesmo conjunto recursivo que é passível de ser gerado pelas máquinas de Turing.

2.5.3 - Máquinas de Turing entendidas como algoritmos

Resta-nos, por último, dar uma visão do que é compreender uma máquina de Turing como um algoritmo. Como vimos, um *algoritmo* é um procedimento sistemático *finito*, ou seja, uma máquina executando um algoritmo *deve* necessariamente parar em um estado de *aceito* ou *rejeito* e, portanto não pode entrar em *ciclo* (MANNA, 1974, p.53). Podemos concluir que só existem algoritmos para calcular *funções totais Turing-computáveis*. Concluimos ainda que toda função total Turing-computável é um algoritmo, pela própria definição de ambas.

Os algoritmos definem uma classe interessante de problemas matemáticos, a saber, aqueles problemas que tem respostas do tipo SIM ou NÃO. Dizemos, pois, que um problema dessa classe é *decidível* se existe um algoritmo que o compute, ou seja, o algoritmo pára em *aceito* para SIM ou pára em *rejeito* para NÃO. Um problema é dito *parcialmente decidível* se existe uma máquina de Turing que pare em *aceito* para SIM, e pare em *rejeito* ou entre em *ciclo* para NÃO. O décimo problema de Hilbert é um exemplo de problema *parcialmente decidível*, pois existe uma máquina de Turing M_1 que pára em *aceito* caso lhe seja dada como entrada em sua fita f_1 a descrição d_2 e a fita f_2 de uma outra máquina M_2 que, por sua vez, pára em *aceito* quando recebe como entrada a fita f_2 . Mas, caso a máquina M_2 entre em *ciclo*, também entra em *ciclo* a máquina M_1 . Para se

enquadrar em um problema do tipo *decidível*, M_1 deveria parar em *rejeito* caso a máquina M_2 entrasse em *ciclo*. Portanto, não existe um algoritmo que resolva o décimo problema de Hilbert (MANNA, 1974, p.53).

Em resumo, introduzimos, neste capítulo, conceitos que julgamos necessários para se compreender a noção de computabilidade, equivalência e poder computacional de máquinas. Vimos também o funcionamento interno de uma máquina de Turing, inserido no contexto histórico de sua criação, o que inclui a noção de calculabilidade efetiva e a proposta de solução do problema da decisão de Hilbert. Mostramos seu funcionamento e explicitamos que todos os problemas modelados por essas máquinas se incluem nas funções Turing-computáveis que são equivalentes às funções recursivas. Dessa discussão, salientamos a importância da definição, mesmo que intuitiva, de algoritmo e procedimento sistemático. Vimos que a máquina de Turing não pode computar qualquer função que não seja passível de descrição numa sequência finita de instruções determinadas. Mostramos a equivalência entre as máquinas de Turing como originalmente propostas e sua inovação, com múltiplas cabeças e fitas. Finalmente, apresentamos o conceito de computabilidade fundamentado nas linguagens formais e autômatos, e discutimos os três modos que se pode compreender a máquina de Turing: como aceitadores de linguagens, como geradores de linguagem e como algoritmos.

Num primeiro instante, parece-nos que a auto-organização está impedida de acontecer nos moldes da máquina de Turing, uma vez que esta possui um controlador único, regido por leis que determinam de modo inequívoco o

comportamento da máquina, não abrindo qualquer espaço para o acaso, exceto aquele proveniente do meio-ambiente, que age diretamente na fita da máquina, mas não nas suas instruções. Discutiremos essa possibilidade novamente, após apresentarmos a teoria da auto-organização.

Veremos no próximo capítulo os avanços da neurociência no estudo dos neurônios biológicos, para entendermos como esses neurônios, com toda sua complexidade, foram usados como inspiração para a criação dos modelos de RNA. A seguir veremos alguns modelos de RNA, seu funcionamento, sua estrutura interna e seus algoritmos, comparando-os, sempre que possível, com o funcionamento, a estrutura e os algoritmos das máquinas de Turing.



2.6 - Notas

¹ *A problem proposed to an information processing system is well defined if a test exists, performable by the system, that will determine whether an object proposed as a solution is in fact a solution* (NEWELL & SIMON, 1972, p. 73).

Tradução nossa: Um problema proposto a um sistema de processamento de informação é dito bem definido se existe um teste, executável pelo sistema, que determinará se um objeto apresentado como solução é de fato uma solução.

² Devemos enfatizar que o “controlador” não é um agente dotado de qualquer intencionalidade, mas um sistema mecânico, reativo às instruções que lhe chegam.

³ Tradução nossa: Também é possível, como mostraremos agora, construir uma máquina, C, que atuará como qualquer máquina de Turing, A, dada, e usará somente dois símbolos, 1 e 0, em sua fita, um dos quais, o 0 digamos, sendo o símbolo para o quadriculado vazio.

A citação começa com “Também...” pois, nesse artigo, Shannon tinha acabado de mostrar que não se perde a universalidade da máquina de Turing utilizando apenas dois estados internos.

⁴ Tradução nossa: É possível inventar uma máquina única que pode ser usada para computar qualquer seqüência computável. Se esta máquina U for suprida com uma fita em que no seu início esteja escrita a descrição padrão de alguma máquina M, então U irá computar a mesma seqüência que M.

⁵ Tradução nossa: Será visto que, como consequência disso [da existência da MTU], todos os computadores digitais são em um sentido equivalentes.

⁶ Para todo sistema formal no sentido da lógica (o que implica ser dada uma linguagem formal, fórmulas dessa linguagem tidas como axiomas e regras de inferência) suficientemente rico para conter a aritmética, temos a seguinte propriedade: ou o sistema é inconsistente (leva a teoremas contraditórios), ou existe pelo menos uma proposição verdadeira que não é demonstrável nesse sistema. (DUPUY, 1996, p.28-9)



⁷ As funções lambda definíveis.

⁸ Ibidem, p.50.

⁹ Via sua equivalência com as funções lambda definíveis. (TURING, 1936, p.263)

¹⁰ Figura exemplo nosso.

¹¹ Tradução nossa: Um número computável pode ser brevemente descrito como um número real cuja expressão como um decimal é calculável por meios finitos. [...] De acordo com minha definição, um número é computável se seu decimal pode ser escrito por uma máquina (de Turing).

¹² Tradução nossa: A computabilidade de uma máquina é sua habilidade de reconhecer uma classe de linguagens.

¹³ Ibidem.

¹⁴ Ibidem, p.42.

¹⁵ Ibidem, p.43.

¹⁶ Ibidem.

¹⁷ Ibidem, p.27, p.33.

¹⁸ Veja p.81.

Capítulo 3 - Os modelos de

Redes Neurais Artificiais

3.1 - Apresentação

Neste capítulo introduziremos os conceitos fundamentais da abordagem conexionista para o estudo da mente. Faremos isso de modo prático, explanando alguns modelos de RNA ao mesmo tempo em que chamamos a atenção para as discussões mais teóricas. Sempre que possível, faremos comparações com as máquinas de Turing, para situarmos o poder computacional dos modelos conexionistas. A seção 3.2 inaugura a discussão sobre RNA dando uma visão geral dos avanços da neurociência no estudo dos neurônios e seu papel no cérebro. A seção 3.3, dividida em subseções, apresenta cinco dos mais conhecidos modelos de RNA, a saber, a rede de McCulloch e Pitts (MP), o perceptron de uma camada, o perceptron multicamadas (MLP), a rede de Hopfield e os mapas auto-organizáveis de Kohonen.

3.2 - A inspiração biológica das Redes Neurais Artificiais

O argumento da plausibilidade biológica nas RNA é invocado tanto para combater as críticas feitas pelos adeptos da IA como para, de fato, mostrar que o objetivo dos adeptos das RNA é entender o funcionamento do

cérebro pelo estudo de processos mecânicos e funcionais dos neurônios. Faz-se mister então averiguarmos o que já sabemos sobre o funcionamento dos neurônios (biológicos), a fim de podermos concluir sobre a plausibilidade biológica levantada.

Os neurônios são compostos de três partes distintas: 1- o corpo celular, que contém o núcleo que, por sua vez, contém os cromossomos que, ainda, contém os genes responsáveis (em grande parte) pela atuação do neurônio em relação ao conjunto de que faz parte; 2- a árvore dendrítica, uma ramificação de dendritos que fazem contato com outros neurônios e com os sensores externos do corpo, para coletar informação do meio; 3- o axônio, um prolongamento que varia de alguns centímetros a até um metro e pouco, que faz contato com (os dendritos de¹) outros neurônios para lhes passar alguma informação (ROCHA, 1999).

Os contatos entre dendritos e axônios são chamados de *sinapses*, ou fendas sinápticas (já que não há um contato real, mas um pequeno espaço). O neurônio que transmite a informação para a sinapse através de seu axônio é chamado de pré-sináptico, e o que recebe tais informações das sinapses para seus dendritos é chamado de pós-sináptico (figura 6).

Acreditava-se até a bem pouco tempo atrás que os neurônios em indivíduos adultos não mais se reproduziam. Existem hoje fortes indícios em experiências feitas em ratos que provam o contrário, afirmando a possibilidade da neurogênese (LEITE, 1999).

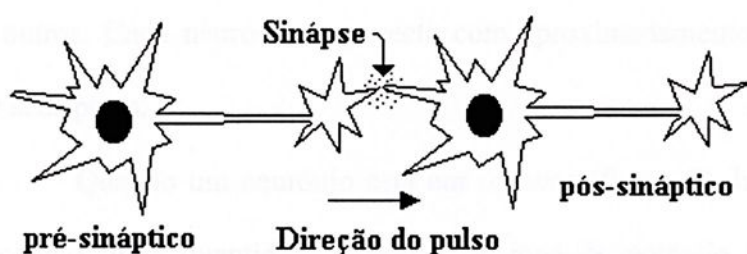


Figura 6: Neurônio pré-sináptico e pós-sináptico²

O cérebro de um ser humano adulto possui aproximadamente 10^{10} neurônios, talvez até mais. Acredita-se, nos estudos de redes neurais artificiais, que os neurônios (biológicos) são as unidades computadoras do cérebro. Assim, pouca ou quase nenhuma importância se dá a outras estruturas cerebrais, como por exemplos as células denominadas *Glias* que também fazem parte da massa encefálica, atribuindo-lhas o ínfimo papel de dar suporte ao correto funcionamento dos neurônios.

Os neurônios são muito ativos metabolicamente. Pesando cerca de 1% do total do corpo humano, o cérebro utiliza aproximadamente 25% da energia total produzida pelo organismo. Aparentemente é a eletroquímica de seu funcionamento que demanda essa alta taxa de energia. Pode-se também averiguar o grande número de mitocôndrias presentes nas sinapses, indicador de alta taxa de metabolismo (ANDERSON, 1995).

Cada neurônio humano possui na ordem de centenas ou milhares de conexões com outros neurônios. Pelo total de neurônios estimados nota-se que é uma modesta quantidade de conexões cujo total chega a ser da ordem de 10^{14} , muito distante de 10^{20} , que teríamos no caso de todos os neurônios se conectarem

a todos os outros. Cada neurodo se conecta com aproximadamente menos que 0,0001% de seus pares.

Quando um neurônio está em repouso (figura 7), há, dentro de seu corpo celular, uma quantidade razoável de íons de potássio (K^+) e uma pequena quantidade de íons de sódio (Na^+), que estão em grande concentração no meio externo. Nessas condições o neurônio é mais negativo, em relação ao meio extracelular.

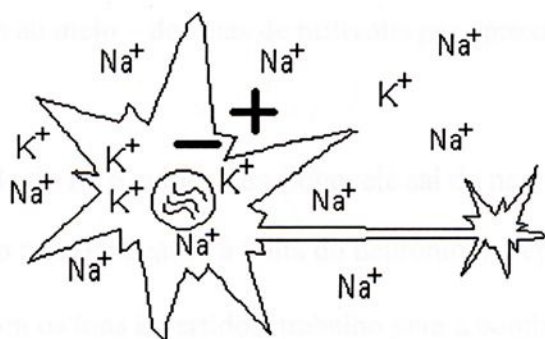


Figura 7: Neurônio em repouso³

Esse estado de repouso é mantido às custas de energia devido ao funcionamento da "bomba de sódio". Tscamente, existem duas forças atuando nos íons (tanto de sódio como de potássio) para mantê-los em suas situações preferenciais: a diferença de concentração e a atração elétrica (ROCHA, 1999).

No caso do potássio essas forças estão em equilíbrio, pois a grande concentração de potássio dentro do neurônio é compensada pela atração que o meio interno (mais negativo) exerce sobre esse íon.

Já o sódio está altamente desbalanceado, pois as duas forças têm tendência a impeli-lo neurônio à dentro: a concentração interna é pequena e a atratividade elétrica é grande. Por esse motivo, e sendo a membrana algo

permeável, (assim como pelos disparos dos neurônios causarem a entrada de sódio e a saída do potássio) a bomba de sódio deve estar constantemente em operação, transferindo íons de sódio para fora do neurônio.

Quando o neurônio é ativado inicia-se um ciclo que é chamado de *potencial de ação* (ROCHA, 1999, p.31):

- 1- Aumenta a permeabilidade do Na^+ (logo, ele entra no neurônio).
- 2- Tal aumento causa uma inversão no potencial elétrico (o neurônio agora fica positivo em relação ao meio – dezenas de milivolts por aproximadamente meio milisegundo)
- 3- A permeabilidade do K^+ é aumentada (logo, ele sai do neurônio).
- 4- A saída do K^+ do neurônio causa a volta do neurônio ao repouso elétrico, nesta parte do axônio (com os íons invertidos, trabalho para a bomba de sódio).

Esse ciclo começa em geral na base do axônio e propaga-se até as suas terminações, onde vai causar aumento da permeabilidade do Ca^+ , que entra no neurônio. O Ca^+ provoca a abertura das vesículas que contém os neurotransmissores, que são liberados para as sinapses.

A velocidade de propagação do potencial de ação pelo axônio pode ser medida, e nota-se que a falta de isolamento do axônio com o meio externo é um empecilho para maiores velocidades. A evolução dos vertebrados descobriu um truque para torná-la mais adequada: uma capa isolante composta de mielina pode aumentar a velocidade de propagação do sinal em até 30 vezes. O diâmetro do axônio também influencia a velocidade de transmissão dos sinais, de

maneira que axônios mais grossos transmitem sinais a maiores velocidades, mas é um tipo de adaptação cujo custo/benefício não é dos melhores.

Os neurotransmissores liberados vão se acoplar aos receptores da membrana pós-sináptica, e vão ativá-los. Ao ser ativado, um receptor pode:

A - Ativar um *canal iônico* e:

- 1- Despolarizar o neurônio, permitindo a passagem de Na^+ para dentro dos dendritos. O neurônio fica mais positivo, e tende a se ativar (PEPS - Potencial Excitatório Pós-sináptico).
- 2- Hiperpolarizar o neurônio, permitindo a passagem de K^+ para fora dos dendritos. O neurônio fica mais negativo, e tende a ficar em repouso (PIPS - Potencial Inibitório Pós-sináptico).

B - Ativar *enzimas* que podem:

- 1- Modificar o comportamento da membrana
- 2- Modificar processos intracelulares
- 3- Modificar a leitura dos genes

O grau de influência do neurônio pré-sináptico sobre o pós-sináptico depende de três fatores básicos:

- 1- A quantidade de neurotransmissores liberados pelo neurônio pré-sináptico.
- 2- A afinidade dos receptores do neurônio pós-sináptico para esses neurotransmissores.
- 3- A quantidade de receptores do neurônio pós-sináptico que tenham alguma afinidade.

Esses três fatores (que incluem a qualidade e a quantidade dos neurotransmissores) são diretamente controlados pela leitura que o neurônio pós-sináptico está habilitado a fazer de seus próprios genes.

Pelo item *B-3* acima, sabemos que existem neurotransmissores que vão atuar exatamente nesta leitura dos genes, e isso disponibiliza uma ferramenta indispensável para o mecanismo de aprendizagem. Essa alteração pode modificar a afinidade química entre os neurotransmissores e os receptores e, também, controlar a quantidade produzida de receptores e de neurotransmissores.

Esse é um pequeno resumo das características menos polêmicas do que se sabe sobre o funcionamento do cérebro. Alguns pontos já estão mais esclarecidos do que o aqui colocado, como, por exemplo, a química dos neurotransmissores, neuromoduladores, hormônios, etc., que já são fabricados em forma de drogas para causarem efeitos específicos em determinados pacientes com determinadas anomalias cerebrais. Outros pontos continuam um mistério até hoje, como por exemplo, o que leva os neurônios a avaliarem positivamente uma solução correta encontrada meses após a tomada de decisão gerada, e como atribuir créditos aos neurônios corretos que geraram essa tomada de decisão? Veremos na próxima seção alguns dos modelos mais conhecidos de redes neurais artificiais e as características que lhes são atribuídas, tornando-os precursores de diversos modelos que surgiram nos últimos anos. Abordaremos no início da próxima seção uma descrição geral de neurodo, dada por Rumelhart e McClelland (1989), que pode ser modificada (especializada) para atuar como a maioria dos neurodos de outras redes, uma espécie de neurodo universal.



3.3 - Modelos de Redes Neurais Artificiais

Os primeiros modelos de RNA foram muito simplificados não por acaso, mas devido à pouca capacidade computacional, que não permitia simulações que levassem em conta grande quantidade de variáveis, além do pouco conhecimento da neurobiologia na época. Esses modelos, baseados na rede de McCulloch e Pitts (1943) e no *Perceptron* de Rosenblatt, constituem a primeira geração de RNA. As RNA estão hoje na sua segunda geração, que segundo Gardner (1993), continuam tão implausíveis biologicamente quanto a primeira geração, tendo se modificado apenas na sua arquitetura, na sua tecnologia e na sua funcionalidade, sem, entretanto, terem alterado substancialmente sua neuromorficalidade. A terceira geração, que Gardner (1993) sugere constituir a união entre os modelos de redes neurais artificiais existentes com a teoria neurológica do cérebro, seria o que Sejnowski, Koch e Churchland (1990) chamam de *Modelos Realistas do Cérebro* (MRC). Em contraste com os MRC estariam os *Modelos Simplistas do Cérebro* (MSC), que apesar de englobarem a primeira e segunda geração, não estagnaram seu desenvolvimento, como quer Gardner justificar para abandoná-los, mas mantêm ativa participação no desenvolvimento de teorias sobre como o cérebro entende o meio à sua volta.

Existe um caráter dubio no estudo das RNA, notado por Crick (1989), uma vez que alguns pesquisadores estão interessados em entender o cérebro, em especial o cérebro humano, enquanto outros estão apenas interessados em gerar algoritmos computacionalmente úteis. Esses últimos normalmente se prendem aos MSC, mas é possível utilizar MSC também para entender o cérebro.



Buscamos esta breve caracterização das RNA para termos uma visão geral do terreno onde assentaremos os estudos desta seção: “*A neural network is an information processing system that is nonalgorithmic, nondigital, and intensely parallel*” (CAUDILL & BUTLER, 1992, p.3)⁴.

Rumelhart e McClelland (1989, p.46) propuseram uma estrutura geral para o entendimento dos diferentes tipos de modelos de redes neurais artificiais, que ainda hoje permanece válida, e talvez continue adequada para a terceira geração de RNA, ou os MRC. Foram apontados oito itens que englobam as características presentes na grande maioria das RNA:

- i.1- Um conjunto de unidades de processamento (neurodos)
- i.2- O estado de ativação
- i.3- Uma função de saída para cada neurodo
- i.4- Um esquema de conectividade entre os neurodos
- i.5- Uma regra de propagação
- i.6- Uma regra de ativação
- i.7- Uma regra de aprendizado
- i.8- O ambiente em que a RNA deve operar

A figura 8, abaixo, ilustra a estrutura interna de um único neurodo geral (NG), em que podemos avaliar cada um dos oito itens expostos acima (RUMELHART & MCCLELLAND, 1989, p.46-54).

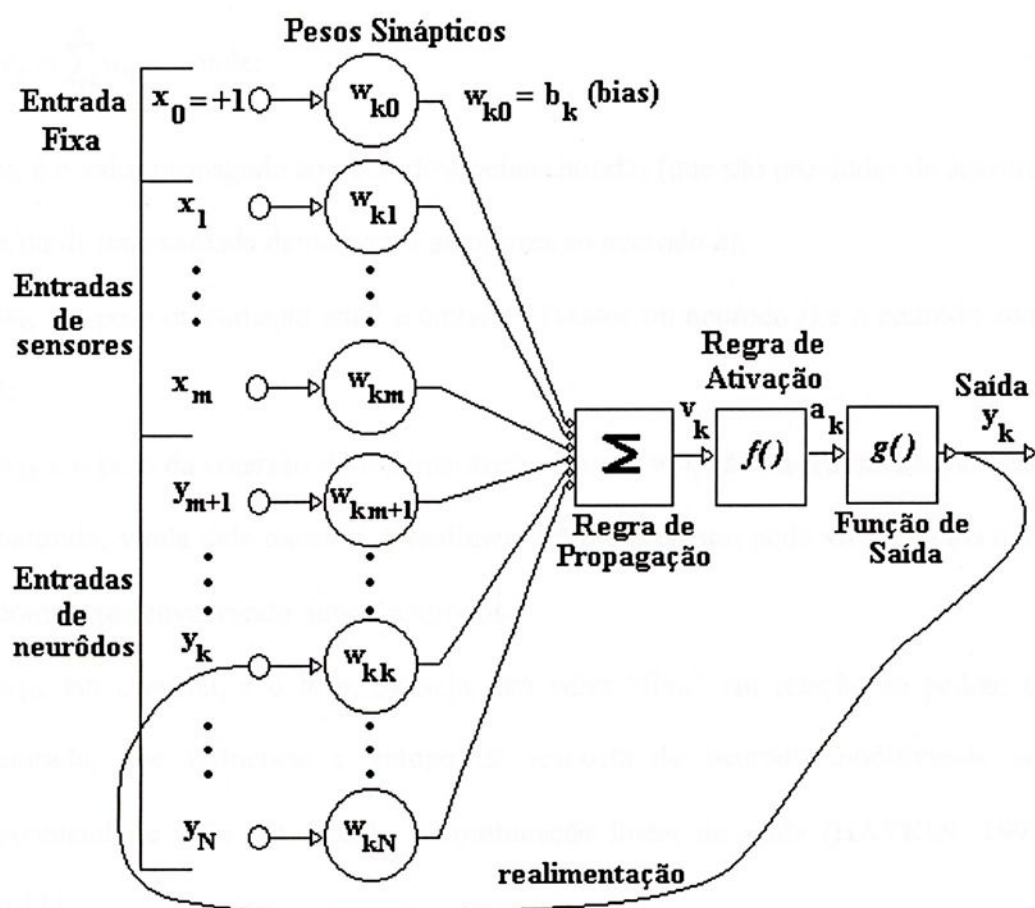


Figura 8: Neurodo geral, esquema interno de um neurodo k^5

Nessa figura podemos observar um esquema abrangente adequado a representar os neurodos (item i.1) da maioria dos modelos de redes neurais propostos. O padrão é apresentado ao neurodo k na forma de um vetor numérico $X = [x_1, \dots, x_m]$ que é passado à regra de propagação (item i.5), juntamente com as entradas providas tanto dos outros neurodos da rede como dele mesmo, $Y = [y_{m+1}, \dots, y_k, \dots, y_N]$. O conjunto total de valores de entrada pode ser descrito por uma matriz de entrada E , da forma:

$$E = [x_0 \quad x_1 \quad \dots \quad x_m \quad y_{m+1} \quad y_{m+2} \quad \dots \quad y_k \quad \dots \quad y_N]$$

O valor propagado ao neurodo k pode ser calculado pela equação:

$$v_k = \sum_{j=0}^N w_{kj} e_j, \text{ onde:}$$

v_k é o valor propagado ao neurodo k pelas entradas (que são providas de sensores e/ou de uma camada de neurodos anteriores ao neurodo k);

w_{kj} é o peso da conexão entre a entrada j (sensor ou neurodo j) e o neurodo atual k ;

w_{kk} é o peso da conexão de realimentação. Se $w_{kk} = 0$, não há realimentação neste neurodo, vinda dele mesmo. A realimentação, entretanto, pode vir de ciclos mais complexos envolvendo outros neurodos.

w_{k0} , em especial, é o *bias*, ou seja, um valor “fixo” em relação ao padrão de entrada, que influencia o campo de resposta do neurodo modificando seu potencial de ação em relação à combinação linear de saída (HAYKIN, 1999, p.11);

e_j é o elemento j da matriz de entrada E , representando o valor recebido pelo neurodo k provindo da entrada j (que pode ser um sensor ou o valor da função de saída do neurodo j - **item i.3**). Caso $j=k$, este valor é a saída do próprio neurodo k que é realimentada para sua entrada.

Uma vez calculado o valor v_k propagado ao neurodo k , esse é passado à regra de ativação $a_k = f(v_k, \dots)$ (**item i.6**), que determinará se esse neurodo ficará ativado ou desativado, dependendo de tantos fatores quantos se queira implementar, como por exemplo, os estados anteriores, um limiar a ser alcançado, um determinado tempo de carência ou latência a ser cumprido, etc, calculando então o valor de ativação a_k . Uma função de ativação muito utilizada é a função degrau, de saída bipolar, com limiar de ativação igual a zero:

$$a_k = f(v_k) = \begin{cases} +1 & \text{se } v_k \geq 0 \\ -1 & \text{c.c.} \end{cases}$$

Finalmente, estando agora determinado se o neurodo está ativado ou não, a última computação feita é calcular o valor que este neurodo deve passar para os demais que lhe seguem. Isso é feito pela função de saída $y_k = g(a_k, \dots)$ (**item i.3**), que recebe como parâmetro o valor calculado pela regra de ativação. Normalmente utiliza-se a função identidade⁶, ignorando assim a função de saída, permitindo que a própria regra de ativação determine a saída do neurodo, fazendo $y_k = a_k$.

A saída y_k fornecida pelo neurodo pode agora ser comparada a uma saída desejada (no caso de redes supervisionadas), ou pode ser agrupada em mapas de padrões (no caso das não-supervisionadas). Caso a saída desejada, ou os mapas formados não sejam satisfatórios, aplica-se uma regra de aprendizado (**item i.7**) que alterará os pesos das conexões w_{kj} levando em conta diversos fatores, como gradientes de derivadas, funções de vizinhanças, etc. Para exemplificar uma regra de aprendizado comumente usada, no caso de redes supervisionadas, temos, conforme resume Haykin (1999, p.52-3):

$W(t+1) = W(t) + \eta \delta Y$, onde:

$W(t+1)$ é a matriz de pesos no tempo $t+1$;

$W(t)$ é a matriz de pesos no tempo t ;

$0 < \eta < 1$ é a taxa de aprendizagem;

δ é o erro, dado pela diferença entre o padrão obtido e o desejado;

Y é o vetor que representa a saída da rede.

Esta equação de ajuste dos pesos é explanada por Haykin⁷ com as seguintes palavras:

The adjustment made to a synaptic weight of a neuron is proportional to the product of the error signal and the input signal of the synapse in question.

(HAYKIN, 1999, p.53)⁸

Foram deixados de lado os itens i.2, i.4 e i.8 que serão agora retomados. O **item i.2** refere-se à definição do que deverá apresentar um neurodo quando estiver ativado ou não, que deve ser pré-definida. Além disso, o estado de ativação da rede como um todo é o que de fato captura o que a rede está representando no momento. Deve haver, portanto, um vetor $A(t)$ com N elementos, que contenha a cada instante de tempo t o padrão de ativação da rede composta de N neurodos, e $a_i(t)$ é o estado de ativação no tempo t do neurodo i (RUMELHART & MCCLELLAND, 1989, p.48).

O **item i.4** refere-se a como cada neurodo está conectado com os outros e com a entrada e saída da rede. Caso não haja realimentação, ou seja, os neurodos das camadas finais não retornem suas saídas para os neurodos das camadas anteriores, dizemos que a rede é de *alimentação direta (feedforward)*. Temos ainda que, a partir da entrada do padrão na rede, cada conjunto de neurodos que recebem esses valores simultaneamente é chamado de camada. A primeira camada é composta de neurodos ligados diretamente aos sensores de entrada da rede, e a camada de saída é a camada de neurodos que exprimem através de sua saída o padrão apreendido. No caso de redes realimentadas a noção de camadas se perde conforme aumenta a complexidade da rede. A camada de

saída é desconsiderada caso o padrão expresso pela rede dependa de todos os neurodos que a compõe.

O item i.8 refere-se à representação do meio-ambiente que a rede deve ser capaz de oferecer:

It is crucial in the development of any model to have a clear model of the environment in which this model is to exist. In PDP models, we represent the environment as a time-varying stochastic function over the space of input patterns.

(RUMELHART & MCCLELLAND, 1989, p.53)⁹

Como cada padrão de entrada tem uma probabilidade de ser apresentado à rede ao longo de sua execução, deve-se levar em conta, ao planejar uma RNA supervisionada, que a apresentação de determinado padrão possa ser corretamente interpretado. Por outro lado, numa RNA não-supervisionada, é a falta desse planejamento¹⁰ *explícito* que leva a rede a classificar os padrões apresentados segundo relações internas próprias (que são resultados da aplicação das suas regras internas, que acabam por perfazer um planejamento *implícito*).

Franklin (1995, p.137) classifica as RNA em cinco classes, segundo o seu estilo de treinamento: (1) Altamente supervisionada. Os pesos são pré-programados, como o caso da rede de MP e da rede de Hopfield. (2) Correção de erro. As respostas são comparadas a um padrão correto e os pesos são ajustados em função dessa comparação. É o caso do perceptron e das redes MLP. (3) Aprendizado por reforço. Uma função de custo avalia as tentativas da rede, e essa avaliação é usada como guia para reajustar os pesos (HAYKIN, 1999, p.603). (4) Aprendizado estocástico. Mudança Hebbiana dos pesos, ou mesmo uma mudança

randômica, por tentativa e erro, como o algoritmo aleatório de treinamento de Solis e Wets (AATSW) proposto por Zamarreño e Vega (1998). (5) Auto-organização. Os pesos são modificados em resposta à entrada, como por exemplo, no caso da rede de Kohonen. Com esta introdução dos modelos gerais de redes neurais artificiais poderemos nos aprofundar em alguns modelos específicos.

3.3.1 - A rede de McCulloch e Pitts

A rede de McCulloch e Pitts (1943) está fundamentada em alguns argumentos considerados biologicamente plausíveis para seus criadores e que hoje se mostram equivocados, como por exemplo, o caráter tudo-ou-nada da comunicação entre neurônios. Seus neurodos também não mostram funcionamento similar ao metabolismo interno dos neurônios. Esse metabolismo foi suprimido, e o que restou foi a mais simples descrição de uma RNA. Seus autores, entretanto, sabiam da simplificação que estavam fazendo ao modelarem estes neurônios símiles do cérebro. As hipóteses levantadas para a concepção dessa rede são:

1 - The activity of the neurone is an 'all-or-none' process; 2- A certain fixed number of synapses must be excited within the period of latent addition in order to excite a neurone at any time, and this number is independent of previous activity and position on the neurone; 3- The only significant delay within the nervous system is synaptic delay; 4- The activity of any inhibitory synapse absolutely prevents excitation of the neurone at that time; 5- The structure of the net does not change with time.
(MCCULLOCH & PITTS, 1943, p.25)¹¹

Podemos exemplificar o funcionamento de uma rede de McCulloch e Pitts (MP) analisando a figura 9. A rede representada por essa figura possui duas entradas, A e B, que podem estar a qualquer momento ativadas ou não. Essas entradas seguem os dados da tabela 3, a *Tabela de Verdade* do conectivo lógico OU-EXCLUSIVO. Elas não são neurodos e servem apenas para distribuir os valores de entrada. O conectivo OU-EXCLUSIVO assume o valor 'verdadeiro' quando uma, e apenas uma, de suas entradas (A e B) é verdadeira. Diferentemente, o conectivo OU, assume 'verdadeiro' no caso de qualquer das duas entradas serem 'verdadeiras', e é utilizado como cálculo parcial pelo neurodo N, que o computa. O neurodo M, por sua vez, computa o conectivo E, que resulta em verdadeiro somente se ambas entradas forem 'verdadeiras'. O conectivo OU-EXCLUSIVO pode ser destrinchado na expressão: " $(A \text{ OU } B) \text{ E NÃO } (A \text{ E } B)$ ", que intuitivamente significa a ocorrência de qualquer das entradas, mas não a ocorrência das duas simultaneamente.

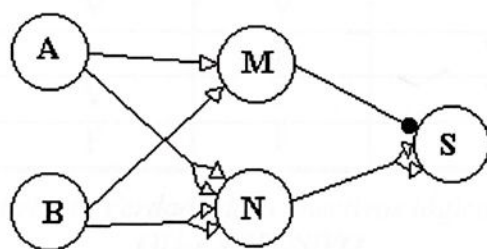


Figura 9: Uma rede lógica de McCulloch e Pitts que implementa o conectivo lógico OU-EXCLUSIVO¹²

Assumimos, na figura 9, que para disparar, um neurodo deve receber pelo menos dois impulsos. O número de impulsos e sua proveniência podem ser inferidos pelo número de setas na figura. Desse modo, o neurodo M

recebe uma única seta de A e uma única seta de B, implicando que só disparará caso ambos enviem seu impulso no mesmo tempo de latência da adição, configurando o conectivo E. Já o neurodo N recebe duas setas de cada neurodo da camada de entrada, o que indica que disparando apenas um ou ambos da camada de entrada, esse também disparará, configurando o conectivo OU. Já o neurodo da camada de saída, S, recebe duas setas de N, o que indica que seu disparo pode ser causado apenas pelo disparo de N, mas recebe uma conexão inibitória (representada graficamente por uma bolinha preta) vinda do neurodo M. A conexão inibitória é semelhante à negação, ou ao conectivo lógico NÃO, que indica que o neurodo S não disparará caso o neurodo M dispare, não importando as outras sinapses. É semelhante, mas não idêntica, pois numa rede representativa da fórmula $S = \text{NÃO}(B)$, o disparo de B inibirá S, mas o não disparo de B *não* excitará o neurodo S, diferindo, pois.

A	B	$N = A \text{ OU } B$	$M = A \text{ E } B$	$S = N \text{ E } \text{NÃO } M$
V	V	V	V	F
V	F	V	F	V
F	V	V	F	V
F	F	F	F	F

Tabela 3: Tabela de Verdade dos conectivos lógicos OU, E, e do OU-EXCLUSIVO

Internamente, o neurodo da rede MP possui uma *função degrau*, que pode ser pré-ajustada para disparar quando os impulsos de entrada atingirem um dado limiar qualquer. Na figura 10 podemos observar um neurodo cujo limiar de disparo foi ajustado para 2. Esse limiar, como vimos no esquema anterior,

atuará de modo a barrar seu disparo até que pelo menos duas de suas entradas excitatórias estimulem o neurodo. As entradas do neurodo da figura 10 são:

- 1) X_1 e X_2 : inibitórias. Seu disparo bloqueia a saída Y .
- 2) X_3 , X_4 e X_5 : excitatórias. O disparo de pelo menos duas destas entradas produz uma saída em Y .

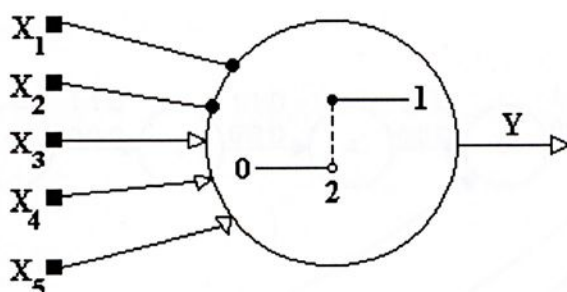


Figura 10: Esquema de neurodo de McCulloch e Pitts¹³

Assim, Y pode ser descrito em termos lógicos, considerando 1 como verdadeiro e 0 como falso:

$$(\text{NÃO } X_1) \text{ E } (\text{NÃO } X_2) \text{ E } [(X_3 \text{ E } X_4) \text{ OU } (X_3 \text{ E } X_5) \text{ OU } (X_4 \text{ E } X_5) \text{ OU } (X_3 \text{ E } X_4 \text{ E } X_5)]$$

O que significa que Y irá apresentar saída 1 (verdadeira) se X_1 e X_2 não dispararem e, simultaneamente, pelo menos duas das entradas X_3 , X_4 ou X_5 dispararem, e saída 0 (falsa) caso contrário.

É perfeitamente possível realizar essa mesma computação por meio de uma máquina de Turing especialmente projetada para isso. O diagrama da figura 11 representa uma máquina de Turing que simula o comportamento do neurodo da figura 10. No diagrama, o círculo de número 1 é o estado inicial, o

conteúdo inicial da fita e a posição da cabeça de leitura e gravação estão indicados na parte inferior da figura.

Cada posição da fita representa, neste modelo específico, um valor binário que indica a presença ou não do estímulo na sinapse de entrada X_i , reservando a última posição para o valor computado de saída Y .

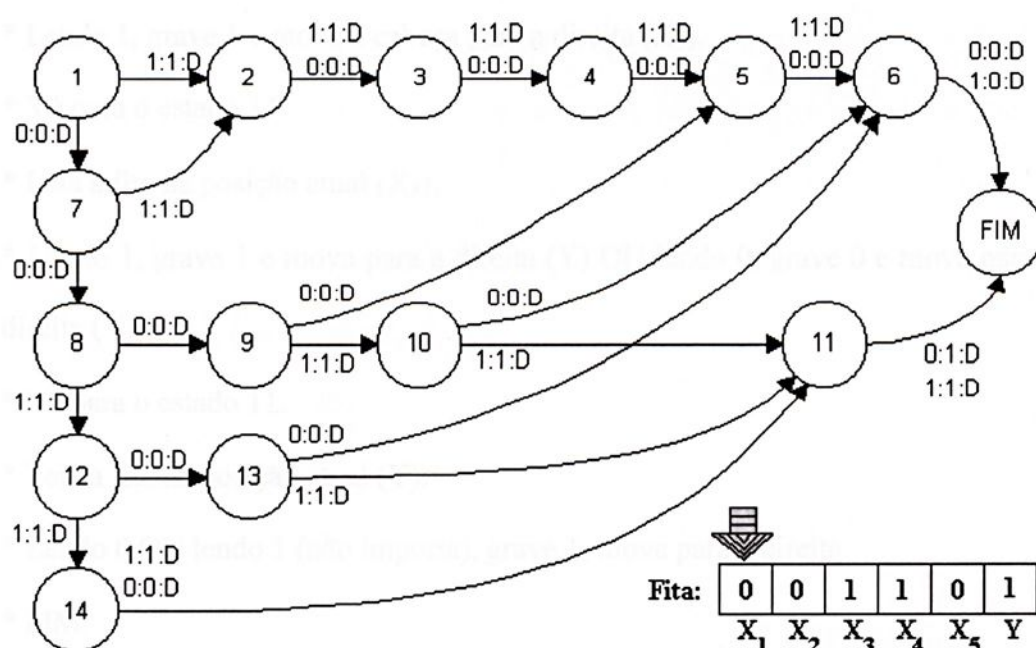


Figura 11: Máquina de Turing que computa um neuródo de McCulloch e Pitts¹⁴

Um exemplo de uma execução dessa máquina, na configuração dada acima, seguiria desta forma:

- * Iniciando no estado 1 (círculo 1), leia a fita na posição atual (X_1).
- * Lendo o valor 0, grave 0 na mesma posição (X_1) e mova a cabeça para a direita (X_2).
- * Vá para o estado 7.
- * Leia a fita na posição atual (X_2).

- * Lendo 0, grave 0 e mova a cabeça para a direita (X_3).
- * Vá para o estado 8.
- * Leia a fita na posição atual (X_3).
- * Lendo 1, grave 1 e mova a cabeça para a direita (X_4).
- * Vá para o estado 12.
- * Leia a fita na posição atual (X_4).
- * Lendo 1, grave 1 e mova a cabeça para a direita (X_5).
- * Vá para o estado 14.
- * Leia a fita na posição atual (X_5).
- * Lendo 1, grave 1 e mova para a direita (Y) OU lendo 0, grave 0 e mova para a direita (Y).
- * Vá para o estado 11.
- * Leia a fita na posição atual (Y).
- * Lendo 0 OU lendo 1 (não importa), grave 1, mova para a direita.
- * FIM.

McCulloch e Pitts (1943, p.27) generalizam essa regra para o disparo de um neurodo c_i qualquer. Suponha que $X_1, X_2, \dots, X_p$ sejam entradas inibitórias no neurodo c_i e $X_{p+1}, X_{p+2}, \dots, X_q$ sejam entradas excitatórias. Suponha ainda que k seja o conjunto de subclasses de $X_{p+1}, X_{p+2}, \dots, X_q$, com θ ou mais elementos, tal que a soma das entradas excitatórias ultrapasse o valor do limiar θ . Com essas suposições, c_i irá disparar de acordo com:

$$Y(c_i) = \left(\bigcap_{m=1}^p \neg X_m \right) \cap \left(\bigcup_{\alpha \in K} \bigcap_{\delta \in \alpha} X_\delta \right), \text{ onde:}$$

$Y(c_i)$ é o disparo do neurodo i ;

$\cup$ e $\cap$ são respectivamente os símbolos sintáticos para a disjunção (OU, $\vee$) e conjunção (E, $\wedge$);

$\neg$ é o símbolo para o NÃO lógico.

Se desmembrarmos a equação acima, teremos resultado semelhante ao dado na equação (Ψ), que descreve o comportamento da saída do neurodo S da figura 9. Generalizar uma máquina de Turing para resolver o disparo de um neurodo para qualquer número de entradas inibitórias e excitatórias é tarefa trivial se nos basearmos no exemplo dado pela figura 11. Bastaria-nos modificar os estados do diagrama da figura 11 para comportar as novas entradas e modificar sua fita para a representada pela figura 12:

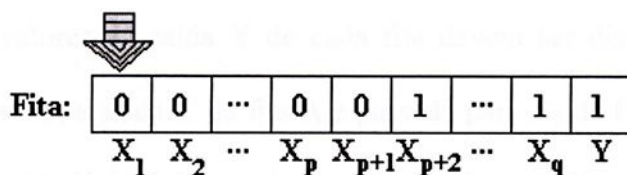


Figura 12: Fita geral para neurodo de McCulloch e Pitts

Uma vez determinado como se comporta o neurodo de McCulloch e Pitts em uma modelização nas máquinas de Turing, podemos exemplificar o funcionamento da rede voltando ao problema do OU-EXCLUSIVO da figura 9. Cada neurodo M, N e S terá sua própria descrição em uma fita separada na máquina de Turing, enquanto as entradas A e B apenas distribuem os valores a serem computados para os neurodos correspondentes (figura 13).

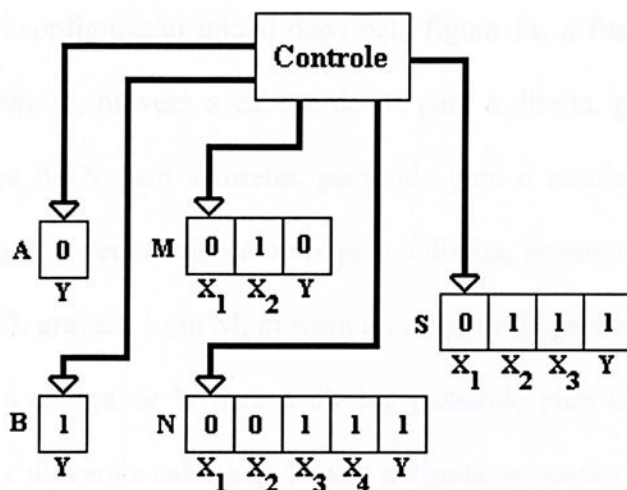


Figura 13: Máquina de Turing de múltiplas fitas simulando uma rede de McCulloch e Pitts para resolver o problema do OU-EXCLUSIVO¹⁵

O funcionamento dessa máquina de Turing segue o algoritmo explicado para um único neurodo, que é aplicado independentemente a cada fita M, N e S. Os valores de saída Y de cada fita devem ser distribuídos para as consequentes. Assim, a saída Y da fita A é passada para X₁ da fita M e para X₁ e X₂ da fita N. A saída Y de B é passada para X₂ de M e para X₃ e X₄ de N. A saída Y de M é passada para a única conexão inibitória da rede, o quadriculado X₁ da fita S. Finalmente, a saída Y de N é passada para X₂ e X₃ de S, completando a última fita, o que permite sua computação. A saída Y de S é o valor da expressão (A OU-EXCLUSIVO B).

Para exemplificar como a saída Y de A e de B podem ser distribuídas para as entradas nos neurodos M e N apresentamos um algoritmo, que não pretende ser único, no diagrama da figura 14. Nesse diagrama, G significa 'grave', L, 'leia', D, 'mova para direita', A, B, M e N são as fitas na qual a operação deve ser executada, 0 e 1 são os símbolos do alfabeto. Partindo do

estado 1, com a configuração inicial dada pela figura 11, a fita A contém '0', o que gravará 0 em M, moverá a cabeça de M para a direita, gravará 0 em N e moverá a cabeça de N para a direita, passando para o estado 3. No estado 3, gravará 0 em N, e moverá a cabeça de N para a direita, passando para o estado 4. Em 4, lerá 1 de B, gravará 1 em M, moverá a cabeça de M para a direita, gravará 1 em N, moverá a cabeça de N para a direita, passando para o estado 5. Em 5, gravará 1 em N e moverá a cabeça de N para a direita, passando finalmente para o estado 7 com as saídas de A e B distribuídas para as respectivas fitas.

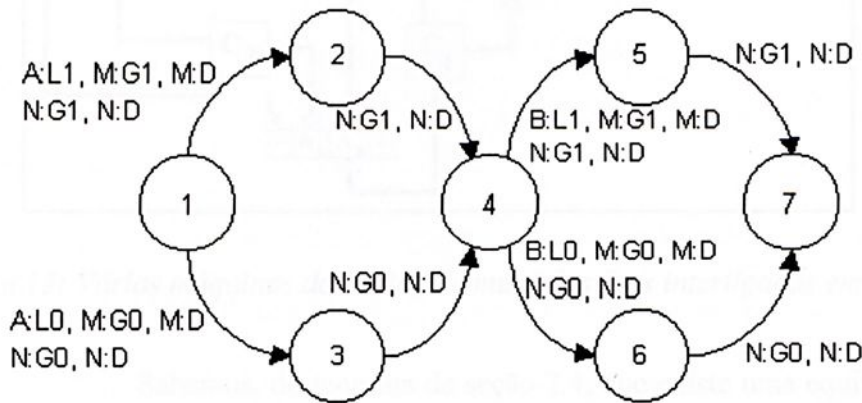


Figura 14: Distribuição das entradas A e B nos neurodos M e N da rede de McCulloch e Pitts, simulada numa máquina de Turing¹⁶

Podemos generalizar esse procedimento de criação de máquinas de Turing com múltiplas fitas e múltiplos controles para assumir qualquer configuração da rede de MP desejada, utilizando, para cada neurodo, uma fita para representar sua entrada, outra para sua saída, e um controle para computar seu disparo (figura 15). Na figura 15, o controle C_1 , por exemplo, recebe sinais das entradas t_1 e t_2 e o seu disparo é gravado na fita t_3 . Seguindo sucessivamente, temos o controle C_5 , que recebe suas entradas das fitas t_5 e t_6 que representam as

saídas dos “neurodos” C_3 e C_4 , respectivamente. A saída de C_5 , gravada em t_7 , representa a saída da rede.

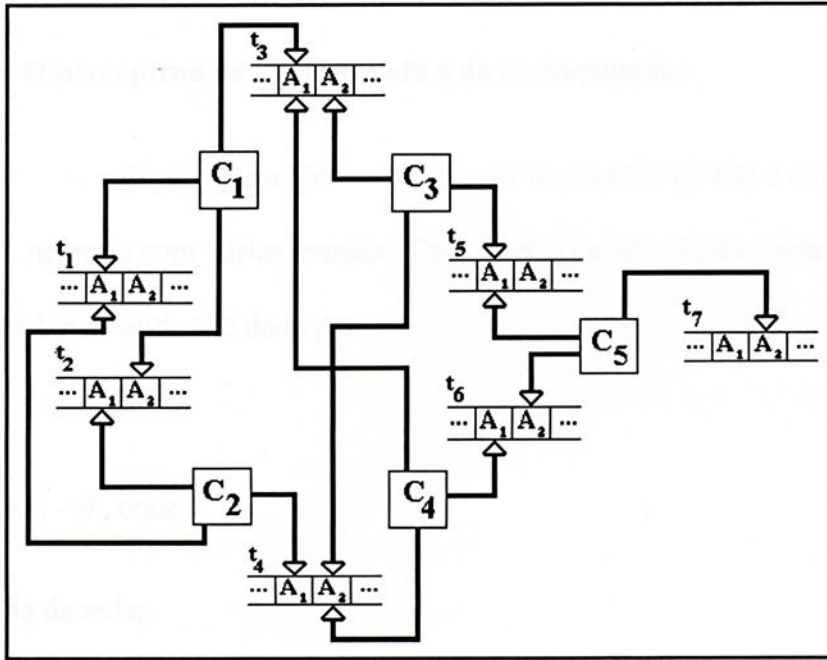


Figura 15: Várias máquinas de Turing de múltiplas fitas interligadas em rede¹⁷

Sabemos, do teorema da seção 2.4, que existe uma equivalência entre as máquinas de múltiplas fitas (do tipo construído para simular uma rede MP) e as máquinas de Turing originais. Admitindo a possibilidade de se computar as somas e a função degrau que cada neurodo de uma rede MP realiza em uma fita, e que múltiplas fitas podem simular múltiplos neurodos, somos levados a concluir que as redes de McCulloch e Pitts são realizáveis em máquinas de Turing sem perda de quaisquer propriedades e, portanto, são Turing-computáveis. Kleene (1956), de fato, mostrou que as redes de MP são equivalentes aos *autômatos finitos*, comprovando que o conjunto de palavras que podem ser geradas por tal

rede está contido no conjunto das *linguagens regulares* ou *linguagens tipo-3*, e que, portanto, são passíveis de serem descritas por uma máquina de Turing.

3.3.2 - O perceptron de uma camada e de multicamadas

O perceptron desenvolvido por Rosenblatt (1962) é composto de um único neurodo com várias entradas ($X_0, \dots, X_{n-1}$) e uma única saída Y (figura 16A). O valor da saída Y é dado por:

$$Y = f_h(v_k)$$

$$v_k = \sum_{i=0}^{n-1} w_i x_i - \theta, \text{ onde:}$$

Y é a saída da rede;

f_h é uma função não-linear de saída da rede - item i.3 da estrutura geral apontado por Rumelhart e McClelland (1989) - que pode ser discreta e binária $\{0, +1\}$, bipolar $\{-1, +1\}$, ternária $\{-1, 0, +1\}$, contínua, com valores reais no intervalo $[0, 1]$, ou outra qualquer;

v_k é o valor propagado ao neurodo perceptron;

w_i é o vetor de pesos das conexões entre a entrada e o perceptron, indexado pelo número da entrada;

x_i é o vetor que representa o padrão de entrada, indexado pelo número da entrada;

θ é o valor do limiar (*threshold*).

Dessa forma, utilizando:

$$f(v_k) = \begin{cases} +1 & \text{se } v_k \geq 0 \\ -1 & \text{c.c.} \end{cases}$$

Y admitirá dois valores: +1 representando a classificação do padrão de entrada como sendo da classe A e -1 representando a inclusão do padrão na classe B (figura 16B).

Na figura 16A, temos o esquema gráfico do perceptron, com suas entradas ($X_0, \dots, X_{n-1}$), seus pesos nas conexões ($W_0, \dots, W_{n-1}$), o corpo do neurodo implementando a função Σ com o limiar θ , a função de saída f_h , e a saída Y. Ao comparar este esquema com aquele dado por Rumelhart e McClelland (1989), apresentado na figura 8, inferimos que este neurodo é um caso particular do NG.

Na figura 16B, temos a representação da linha limite (a linha pontilhada) que separa o espaço do problema em dois semiplanos. No semiplano superior o perceptron classifica corretamente a classe A, enquanto no inferior classifica, corretamente, a classe B.

Na figura 16C, temos a representação do movimento realizado pela linha limite durante a fase de treinamento do perceptron.

Finalmente, na figura 16D, temos a representação do espaço do problema de classificar as saídas resultantes da aplicação do conectivo lógico OU-EXCLUSIVO, em que se pode notar a impossibilidade de separar as duas classes A e B com a linha pontilhada.

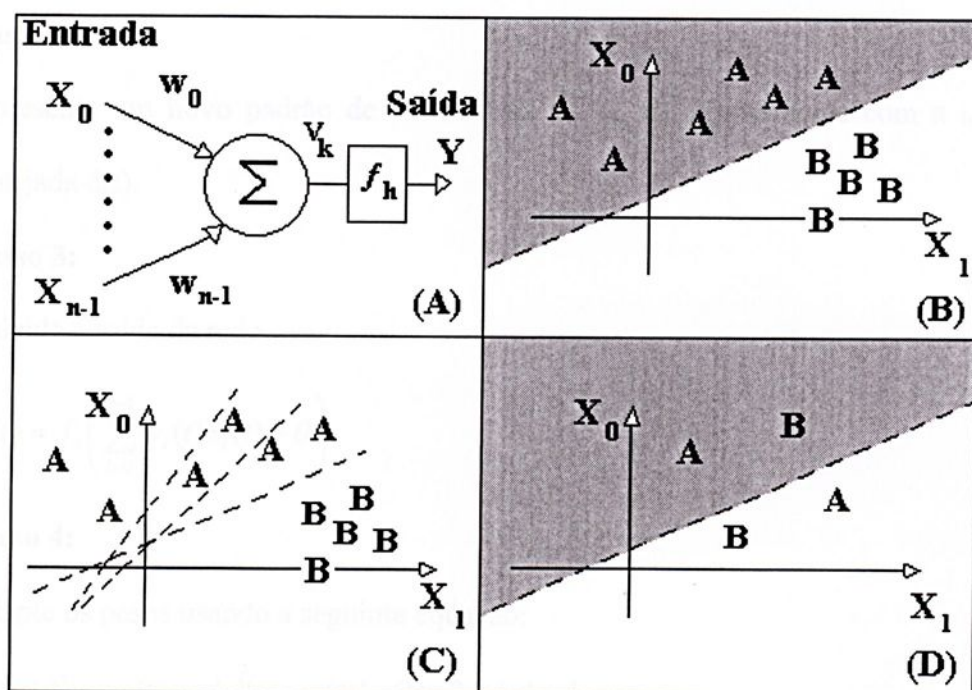


Figura 16: (A) perceptron; (B) duas classes, A e B;
 (C) convergência no aprendizado; (D) o problema do OU-EXCLUSIVO
 (LIPPMANN, 1987)

O processo de treinamento do perceptron de uma camada é realizado de acordo com o algoritmo de convergência abaixo, segundo Lippmann (1987, p.13). Trata-se de um procedimento que, num número finito de passos, possibilitará o ajuste dos pesos W_i , de tal forma que eles convergirão a um valor estável que representa a solução do problema. A solução do problema, entretanto, só será alcançada, ou convergirá, caso o problema esteja incluído na classe de problemas linearmente separáveis.

O algoritmo de convergência do perceptron de uma camada:

Passo 1:

Atribua um valor aleatório próximo de zero ao vetor peso $w_i(t)$, com $(0 \leq i \leq n-1)$, e ajuste o limiar θ .

Passo 2:

Apresente um novo padrão de entrada $x_0, x_1, \dots, x_{n-1}$, juntamente com a saída desejada $d(t)$.

Passo 3:

Calcule a saída da rede.

$$Y(t) = f_h \left(\sum_{i=0}^{n-1} w_i(t) x_i(t) - \theta \right)$$

Passo 4:

Adapte os pesos usando a seguinte equação:

$$w_i(t+1) = w_i(t) + \eta[d(t) - y(t)]x_i(t), 0 \leq i \leq n-1$$

$$d(t) = \begin{cases} +1 & \text{Classe A} \\ -1 & \text{Classe B} \end{cases}$$

Passo 5:

Repita todo o algoritmo partindo do passo 2.

Nessas equações, η (a taxa de aprendizado) pode ser expressa através de uma fração de ganho positiva e menor que 1 e $d(t)$ é o valor correto (desejado) da saída corrente. Note que os pesos não se alteram quando a rede classifica corretamente uma entrada, pois a subtração $d(t)-y(t)$ se anula (LIPPMANN, 1987).

O perceptron divide o espaço do problema em duas regiões por uma linha ou hiperplano (a linha pontilhada na figura 16B). Esse algoritmo exposto acima treina o perceptron movimentando esta linha (figura 16C), que é dada por:

$$x_1 = \frac{-w_0}{w_1} x_0 + \frac{\theta}{w_1}$$

Uma das limitações no modelo do perceptron, que Minsky e Papert (1988) demonstraram, é a restrição desses modelos a problemas linearmente separáveis. Fica excluída a possibilidade, entre outras, de modelar, por exemplo, o OU-EXCLUSIVO, cujo espaço do problema é mostrado na figura 16D. Tais limitações foram desmotivadoras para os adeptos das RNA, o que ocasionou grande atraso no desenvolvimento das pesquisas nessa área. Entretanto, essa limitação está restrita ao perceptron de uma camada, não permanecendo para o caso de várias camadas de neurodos. Para redes de perceptrons multicamadas (também chamadas de redes MLP – *Multi Layer Perceptron*) é possível delimitar qualquer região do espaço do problema (figura 17). Conforme explicitaremos ao final desta seção, as críticas de Minsky e Papert (1988) foram respondidas através da elaboração do perceptron multicamadas.

Na figura 17A temos o perceptron de uma camada. Conforme observaram Minsky e Papert (1988), o problema do OU-EXCLUSIVO não pode ser solucionado por essa estrutura, que está restrita a solucionar problemas que são separáveis em apenas duas regiões.

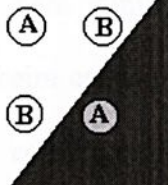
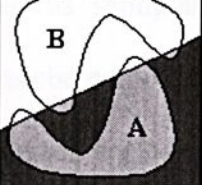
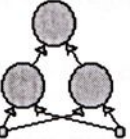
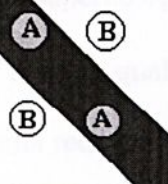
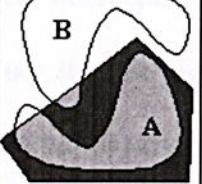
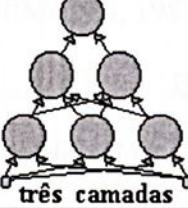
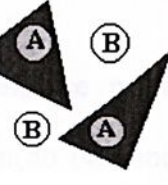
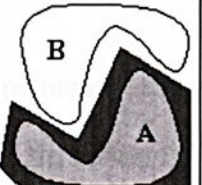
	Estrutura	Tipo de Região	Problema do OU-EXCLUSIVO	Classes entrelaçadas	Formas de regiões gerais
(A)	 uma camada	Plano separado por reta			
(B)	 duas camadas	Regiões convexas abertas ou fechadas			
(C)	 três camadas	Arbitrária (Complexidade limitada pelo número de neurôdos)			

Figura 17: Regiões de decisão que podem ser formadas com perceptrons (LIPPMANN, 1987, p.14)

A figura 17B apresenta uma rede de perceptrons multicamadas, com duas camadas. Nesse caso é possível solucionar o problema do OU-EXCLUSIVO, bem como separar com maior grau de precisão classes com regiões entrelaçadas pela formação de regiões convexas abertas ou fechadas. O número de neurodos na primeira camada determina quantos lados terá a figura delimitadora do espaço do problema. Cada neurodo da primeira camada criará um semiplano que sofrerá uma intersecção feita pelos neurodos da segunda camada. Essa intersecção corresponde a uma operação lógica E.

A figura 17C mostra uma rede de perceptron multicamadas, com três camadas. A rede de três camadas pode ajustar regiões de decisão de formas

arbitrárias, convexas ou não, e mesmo desconexas. A primeira camada continua determinando os semiplanos que limitam a região, e a segunda camada continua atuando com operações lógicas E para combinar os semiplanos em figuras convexas abertas ou fechadas. A terceira camada recebe como entrada as regiões convexas da segunda camada, e a combinação dessas regiões convexas pode resultar em regiões não convexas por superposições, interseções, uniões, etc. Com esse procedimento é possível montar qualquer tipo de figura, não sendo necessárias mais que três camadas numa rede de perceptrons de alimentação direta (LIPPMANN, 1987, p. 16).

Essa discussão centra-se principalmente no perceptron cuja função de saída é dada por uma função não-linear descontínua. Caso se utilize como função de saída uma função contínua do tipo sigmóide o comportamento será aproximadamente o mesmo, apesar de um pouco mais complexo. A complexidade deriva de que teríamos curvas suaves ao invés de segmentos de retas (LIPPMANN, 1987, p.16).

As redes de perceptron multicamadas, com funções de saídas contínuas do tipo sigmóide (ou outra não-linear, contínua e diferenciável) podem ser treinadas pelo algoritmo de treinamento de retropropagação (*backpropagation training algorithm*), que é o principal algoritmo empregado em grande parte das redes neurais. Tal rede envolve um algoritmo diferente do anterior uma vez que agora terá que lidar com pesos em conexões interneuronas, que não existiam no perceptron de uma camada.



Trata-se de um algoritmo de treinamento iterativo, criado para minimizar o erro mínimo quadrado entre a saída real de uma rede de perceptrons multicamadas, com alimentação direta (*feedforward*) e a saída desejada, utilizando para o cálculo do erro a função gradiente δ . Tal algoritmo pode ser descrito nos seguintes passos (LIPPMANN, 1987, p.17):

O algoritmo de treinamento por retropropagação:

Passo 1:

Inicialize os pesos com valores aleatórios pequenos, próximos de zero.

Passo 2:

Apresente um vetor $(X_1, X_2, \dots, X_{n-1})$ contendo um padrão, e especifique as saídas desejadas $(d_0, d_1, \dots, d_{m-1})$ para cada neurodo de saída.

Passo 3:

Fase de propagação para frente: calcule a saída de cada camada e use-a como entrada para a camada seguinte, até calcular a saída da rede $y_0, y_1, \dots, y_{m-1}$. Use como função de saída para cada neurodo a função:

$$f(v_k) = \frac{1}{1 + e^{-(v_k - \theta)}}, \text{ onde:}$$

v_k é o valor propagado ao neurodo k , e θ é o limiar.

Passo 4:

Fase de retropropagação: ajuste os pesos das conexões, partindo da última camada e calculando sua variação retroativamente até a primeira camada, utilizando a equação:

$$w_{ij}(t+1) = w_{ij}(t) + \eta \delta_j x_i, \text{ onde:}$$

$w_{ij}(t)$ é o peso da conexão entre o neurodo i ou a entrada i e o neurodo atual j , t é um tempo dado e $t+1$ representa o próximo instante de tempo;

η é a taxa de aprendizado, também chamado de termo de ganho;

$\dot{x}_i$ é a saída do neurodo i , ou é a entrada i na camada de entrada.

δ_j é o erro apresentado pela saída do neurodo j . δ_j é calculado por:

(I) Caso o neurodo j seja um neurodo da camada de saída:

$$\delta_j = f'(v_k)(d_j - y_j), \text{ onde,}$$

$f'(v_k)$ é a derivada primeira da função de saída (neste caso a sigmóide dada acima);

y_j é a saída atual do neurodo j , e d_j é a saída desejada.

(II) Caso o neurodo j seja um neurodo de alguma camada intermediária:

$$\delta_j = f'(v_k) \sum_{k=0}^{N-1} \delta_k w_{jk}$$

Passo 5:

Repita a partir do passo 2.

Uma vez estabelecido o algoritmo de funcionamento da rede MLP, ao analisarmos-lo percebemos que em nada difere daqueles algoritmos utilizados para programar as máquinas de Turing. Esse algoritmo de retropropagação, bem como o algoritmo do perceptron de única camada, possui uma sequência de passos finitos (neste caso, cinco passos), cada qual realizando uma computação passível de ser feita por uma máquina de Turing. Existe ainda uma ordem no tempo, que impede que o passo 4, por exemplo, que inicia a fase de retropropagação do erro seja realizado antes do passo 3, a fase de propagação para

frente. Poderia-se argumentar que o passo 3, na verdade, envolve uma quantidade imensa de cálculos que são realizados simultaneamente para todos os neurodos da rede. Nada impede, devemos lembrar, que uma máquina de Turing realize essas mesmas operações em paralelo. Existiriam, entretanto, outras características que impediriam esse algoritmo de ser realizado com sucesso numa máquina de Turing? Outra questão é: supondo que o algoritmo possa ser simulado em uma MT, o resultado da simulação desse algoritmo, ou seja, o que esse algoritmo calcula é uma função Turing-computável? Teremos oportunidade de rever essas questões no capítulo 4. Por enquanto nos contentaremos em analisar mais dois modelos de RNA, a rede de Hopfield e a rede de Kohonen.

3.3.3 - A rede de Hopfield

A rede criada por Hopfield (1982), no início dos anos 80, é normalmente utilizada como uma memória associativa, ou para resolver problemas de otimização. Como memória associativa, a rede de Hopfield é capaz de recuperar um padrão corrompido ou parcialmente apresentado a ela, devolvendo como resposta o padrão completo. Na solução de problemas de otimização, essa rede fornece uma solução viável, não necessariamente ótima, após a convergência do padrão de saída.

A rede de Hopfield difere das RNA em geral, principalmente no primeiro passo, o da inicialização. É comum ver os modelos de RNA inicializarem seus pesos das conexões com valores aleatórios próximos de zero e então, durante o processo de aprendizado, esses pesos serem modificados até se

Tese 1288

adequarem ao problema. Já a rede de Hopfield tem em seus pesos iniciais o essencial para o seu bom funcionamento. Uma vez inicializados os pesos corretos, estes ficam fixos durante todo o restante do procedimento de convergência, até que a rede classifique ou otimize o padrão apresentado corretamente. Lippmann (1987, p.7), em sua taxonomia das RNA, classifica a rede de Hopfield como supervisionada, uma vez que seus pesos devem ser pré-definidos, seja programando-os diretamente, ou inserindo-os (uma única vez) via alguma função.

O processo de convergência na rede de Hopfield não é a busca pelos pesos estáveis, como ocorre com outros modelos. É o padrão de entrada que é modificado nas iterações do procedimento e converge em outro padrão qualquer para ser o padrão de saída da rede.

Lippmann indica que as redes de Hopfield são usualmente modeladas para usarem entradas binárias ou bipolares, o que torna sua convergência mais fácil. Na figura 18 apresentamos um modelo de rede de Hopfield, o qual passaremos a explicar (LIPPMANN, 1987).

O padrão de entrada é apresentado na forma de um vetor $X=(x_0, \dots, x_{n-1})$, o que causa imediatamente a saída $(x'_0, \dots, x'_{n-1})$, bem como a realimentação dessa saída nos neurôdos que a geraram $(\mu_0, \dots, \mu_{n-1})$, que passam a repetir o processo, gerando novas saídas e novas realimentações até que não haja mais mudança significativa entre uma iteração e outra no padrão realimentado, caso em que o algoritmo descrito abaixo deverá parar. As saídas intermediárias de $(x'_0, \dots, x'_{n-1})$ fornecidas por esse procedimento devem ser ignoradas, resguardando-se apenas a última, que é a resposta da rede.

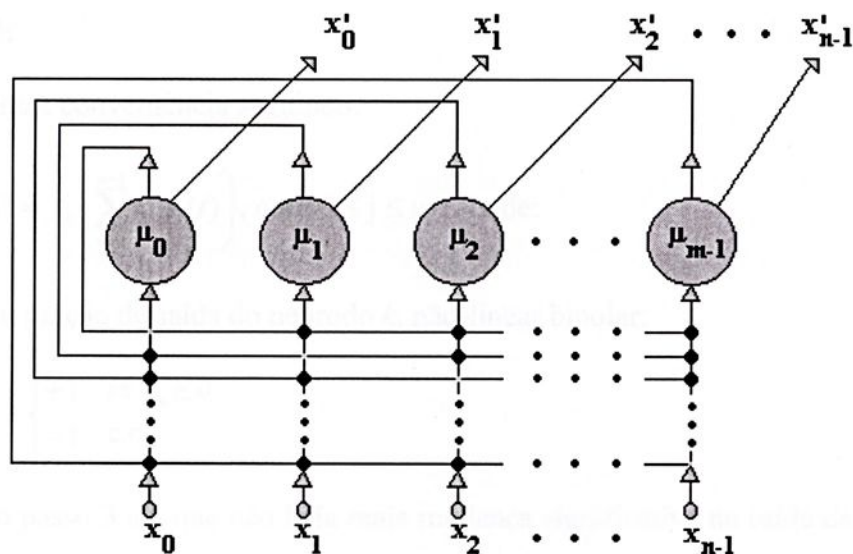


Figura 18: O modelo de Rede de Hopfield¹⁸

O algoritmo da rede de Hopfield

Passo 1:

Atribua os pesos das conexões segundo a equação:

$$t_{ij} = \begin{cases} \sum_{s=0}^{m-1} x_i^s x_j^s, & i \neq j \\ 0, & i = j \end{cases}, \text{ com } 0 \leq i, j \leq m-1, \text{ onde:}$$

t_{ij} é o peso da conexão do neuródo i com o neuródo j ;

x_i^s (que pode assumir os valores +1 e -1) é o elemento i do vetor X exemplar do padrão da classe S .

Passo 2:

Apresente um padrão desconhecido $X=(x_0, \dots, x_{n-1})$ para $\mu(t)$, no tempo $t=0$.

Assim:

$$\mu_i(0)=x_i, \text{ com } 0 \leq i \leq n-1$$

Passo 3:

Iteraja até a convergência seguindo:

$$\mu_i(t+1) = f_h \left(\sum_{j=0}^{n-1} t_{ij} \mu_j(t) \right), \text{ com } 0 \leq j \leq n-1, \text{ onde:}$$

$f_h(v_k)$ é a função de saída do neurodo k , não-linear bipolar:

$$f_h(v_k) = \begin{cases} +1 & \text{se } v_k \geq 0 \\ -1 & \text{c.c.} \end{cases}$$

Repita o passo 3 até que não haja mais mudança significativa na saída da rede ou no erro apresentado.

Passo 4:

Repita o procedimento para um novo padrão, indo para o passo 2.

A rede de Hopfield, conforme aqui descrita, não possui a capacidade de aprendizado¹⁹, entretanto, é uma propriedade emergente desse tipo de rede a capacidade de generalização fraca. Fraca porque ela não é capaz de criar categorias novas, mas é capaz de classificar corretamente, com uma probabilidade de sucesso razoável, padrões que nunca antes foram apresentados à rede. Os perceptrons, tanto o de uma camada como o multicamadas (com seu algoritmo de retropropagação), apresentam como propriedade emergente a capacidade de aprendizagem supervisionada, que o torna apto a classificar corretamente os padrões utilizados no seu treinamento, demonstrando ainda a capacidade de generalização fraca.

Utilizar a rede de Hopfield para otimizar funções é relativamente simples. Os pesos são arranjados de modo a refletirem as

delimitações, restrições ou os custos que envolvem a passagem de uma situação para outra, representadas pelos neurodos.

Exemplificaremos essa qualidade da rede de Hopfield mostrando como se soluciona o problema do caixeiro viajante que foi abordado no primeiro capítulo. A rede terá n^2 neurodos, onde n é o número de cidades que devem ser visitadas. Suponha, por exemplo, que nosso caixeiro precise visitar as cinco cidades apontadas na figura 19. O passo 1 do algoritmo descrito acima seria eliminado, tornando a inicialização dos pesos tarefa realizada manualmente. Os 25 neurodos da figura 19 teriam seus pesos pré-fixados de modo que:

- 1- Os neurodos de uma mesma linha (ou cidade) se inibam, para que cada cidade só seja visitada uma vez.
- 2- Os neurodos de uma mesma coluna se inibam, pois o caixeiro só pode visitar uma cidade de cada vez.
- 3- Os neurodos das colunas adjacentes se inibam ou se excitam na proporção da distância entre as cidades que representam.

Com a rede preparada, damos como entrada um padrão que representa nosso itinerário preferido. A rede irá operar nesse itinerário inicial e otimizá-lo, estabilizando-se em outro próximo do original, mas com menor distância total. Na figura 19, o itinerário que foi dado como resposta pela rede foi: Marília, São Carlos, Rio Claro, Bauru, Araçatuba e novamente Marília (RICH & KNIGHT, 1993, p.593).

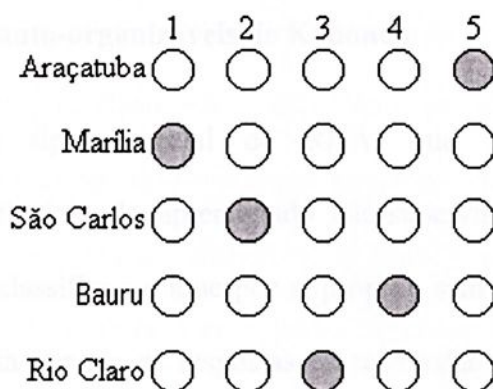


Figura 19: Uma solução do caixeiro viajante usando rede de Hopfield²⁰

Por não alterar seus pesos, essa rede apresenta uma estrutura rígida, pronta para resolver um problema específico e nada mais. As respostas, entretanto, surpreendem não pela exatidão, mas pela boa aproximação alcançada, uma característica humana na solução de problemas. As redes de Hopfield simuladas em MVN convergem para soluções ótimas, seguindo rigorosamente o algoritmo estabelecido. Também seguindo esse algoritmo, criam uma camada sobre as máquinas de Turing, uma camada algorítmica e representacional, que transforma a memória endereçada dessas máquinas em memória associativa, quando usadas para tal fim. Diferentemente da rede lógica de McCulloch e Pitts, na qual a sua equivalência com os autômatos finitos foi comprovada (KLEENE, 1956; LUDERMIR, 1992, p.269) pelo simples motivo de que as proposições lógicas codificadas naquela podem ser codificadas neste, as redes de Hopfield parecem mostrar um desafio maior. Chama-nos a atenção a obviedade de que essas redes possam ser simuladas em máquinas de Turing e ainda assim escaparem das funções Turing-computáveis. Veremos finalmente o nosso último exemplo de modelo de RNA, um modelo auto-organizável.

3.3.4 - Os mapas auto-organizáveis de Kohonen

Um tipo especial de RNA, que agora descreveremos, fundamenta-se no princípio do aprendizado não-supervisionado. Com isso, se pretende que a rede classifique e atue por si própria, sem a intervenção forte de um professor que dita quando as respostas da rede estão certas e quando estão erradas. Tais redes, denominadas de redes de Kohonen, produzem durante sua execução os chamados Mapas Auto-Organizáveis de Kohonen ou, simplesmente, Mapas Auto-Organizáveis (MAO).

Lippmann (1987, p.19) aponta uma semelhança entre os MAO e a estrutura biológica do córtex auditivo, em que ambas, de uma forma ou de outra, mantêm uma ordem entre suas unidades (neurodos e neurônios) que reflete características físicas dos estímulos externos recebidos.

Na figura 20A podemos observar uma típica rede de Kohonen, com o padrão de entrada codificado pelo vetor $X(t)=(x_1, \dots, x_n)$ sendo passado para camada de neurodos de Kohonen. Em seu modelo original (KOHONEN, 1987), o vetor de entrada é normalizado, calculando depois a menor distância euclidiana entre esse vetor e os pesos das conexões de cada neurodo. Considera-se então o neurodo de menor distância como o vencedor e, assim, o representante desta classe de padrões. Esse neurodo terá suas conexões fortificadas, bem como os seus vizinhos. A quantidade de vizinhos que serão afetados varia com o tempo, começando bem abrangente e diminuindo até utilizar somente os contíguos ou mesmo nenhum.

É a vizinhança, nesse procedimento de ajuste dos pesos, que dá ao mapa gerado a característica de neurodos, topologicamente próximos, terem pesos semelhantes, conseqüentemente respondendo a estímulos semelhantes.

A figura 20B apresenta o modo como o neurodo i da camada de neurodos de Kohonen está interligado, via conexões com seus respectivos pesos ($w_{i1}, w_{i2}, \dots, w_{in}$), com todos os elementos do vetor de entrada de padrões.

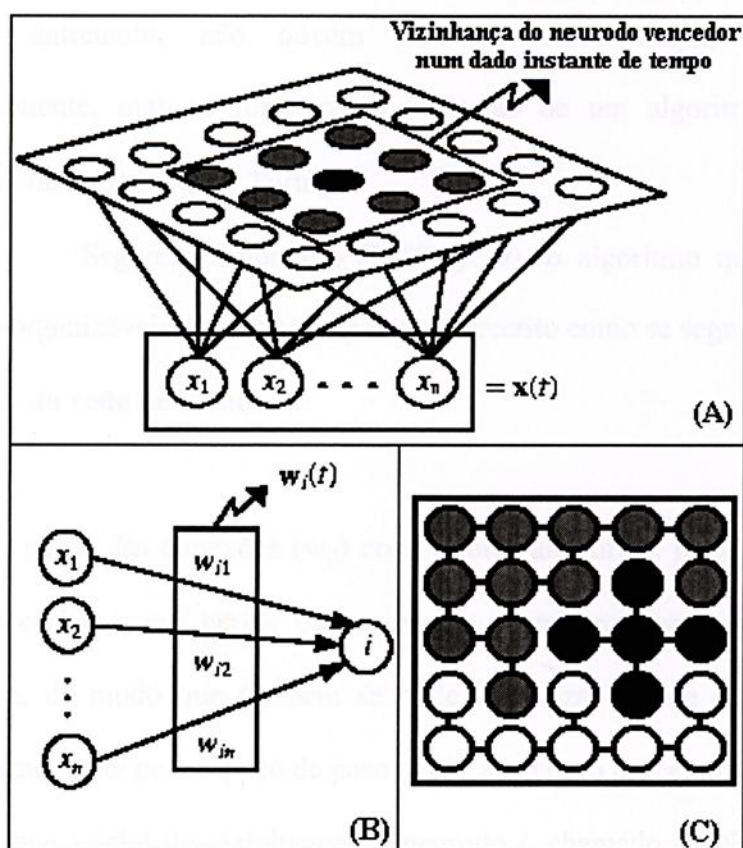


Figura 20: Mapas auto-organizáveis de Kohonen. (A) Esquema de uma arquitetura típica. (B) Vetor peso do neurodo i na matriz. (C) Mapas auto-organizados²¹.

Após a apresentação de repetidos padrões, e de todo o processo de iteração, os neurodos da rede tendem a se especializar, respondendo a apenas uma classe de padrões. Na figura 20C vemos três tipos de neurodos,

caracterizados pelas cores branco, cinza e preto, que se especializaram em responder a três classes de padrões diferentes. As linhas pretas interligando-os representam conexões excitatórias, sua ausência representa uma conexão inibitória. Notamos, com certa admiração, a característica desses mapas se espalharem pela camada de Kohonen de uma forma mais ou menos organizada, mantendo juntos os neurodos que respondem a classes semelhantes. Nossa admiração, entretanto, não advém de que esses mapas se formam inexplicavelmente, mas se formam da execução de um algoritmo detalhado simulado em uma máquina de Turing.

Segundo Lippmann (1987, p.19), o algoritmo que produz os mapas auto-organizáveis de Kohonen pode ser descrito como se segue:

O algoritmo da rede de Kohonen:

Passo 1:

Inicialize os pesos das conexões (w_{ij}) com valores aleatórios, próximos de zero. Os pesos aleatórios, em certos casos, levam a um processo muito lento de convergência, de modo que também se pode inicializar a rede distribuindo os pesos uniformemente pelo espaço de pesos, para acelerar o processo.

Inicialize o raio inicial da vizinhança do neurodo j , chamado de $N_j(0)$, para um valor relativamente grande.

Passo 2:

Apresente um padrão de entrada $x_1(t)$, $x_2(t)$, ..., $x_n(t)$, onde $x_i(t)$ é a entrada no neurodo i , no tempo t .

Passo 3:

Calcule a distância d_j entre o padrão entrado e cada neurodo j da rede, usando:

$$d_j = \sum_{i=1}^n (x_i(t) - w_{ij}(t))^2$$

Passo 4:

Selecione o neurodo j^* que teve a saída de menor distância d_j e o designe como sendo o neurodo vencedor.

Passo 5:

Atualize os pesos do neurodo vencedor j^* e dos seus vizinhos que estejam dentro de $N_{j^*}(t)$. Os novos pesos são dados por:

$$w_{ij}(t+1) = w_{ij}(t) + \eta(t)(x_i(t) - w_{ij}(t)), \text{ para todo } j \in N_{j^*}(t), \text{ e } 0 \leq i \leq n, \text{ onde:}$$

$\eta(t)$ é a taxa de aprendizagem (ou termo de ganho), tal que $0 < \eta(t) < 1$, que decresce com o tempo, diminuindo assim a variação dos pesos (causando uma sintonia fina);

$N_{j^*}(t)$ decresce em tamanho com o tempo, diminuindo a vizinhança do neurodo vencedor (focalizando a área do MAO de maior atividade).

Passo 6:

Repita o processo a partir do passo 2.

Ao executar esse algoritmo, seja em uma máquina de Turing ou em uma máquina conexionista, o número de classes que se formará e quais neurodos participarão delas depende do sorteio inicial dos pesos, da ordem de apresentação dos padrões, e do decaimento da função vizinhança. Mantendo-se essas variáveis sob controle, os MAO se organizarão sempre da mesma forma,

enquanto se repetir o experimento. É possível manter as variáveis da rede de Kohonen sob controle numa simulação em MVN, e podemos observar o mesmo mapa sendo reconstruído repetidas vezes. Seria possível realizar esse experimento também numa MC? Para conseguir esse intento, tanto em uma máquina como em outra, temos que eliminar o acaso trazido pela interação com o meio-ambiente durante a apresentação dos padrões. Manter o sorteio inicial dos pesos e a função de vizinhança é, aparentemente, tarefa mais simples. Ainda, mesmo que conseguíssemos manter as variáveis da rede sob controle, o algoritmo descrito acima abriria brechas para computar algo além do programado? Tentaremos, no próximo capítulo, esclarecer essas e outras questões que surgem no momento.

Neste capítulo apresentamos uma visão geral das redes neurais artificiais, começando com uma introdução à neurociência, apresentando um estudo dos neurônios biológicos, destacando o papel dos neurotransmissores e do metabolismo interno que são desconsiderados nos modelos simplificados de neurodos que constituem a maioria das redes neurais artificiais, os modelos simplificados do cérebro (MSC). Essa introdução à neurociência serviu como base para compreender como o neurônio biológico foi simplificado e modelado para a criação das redes neurais artificiais.

Colocamos em seguida cinco modelos clássicos de RNA, a rede de McCulloch e Pitts, o perceptron de uma camada, o perceptron multicamadas, a rede de Hopfield e a rede de Kohonen, explicitando o funcionamento, os algoritmos e algumas das propriedades que emergem dessas redes. A rede MP não apresenta capacidade de aprendizado, nem de generalização, e sua capacidade

computacional se encontra estritamente dentro da que é apresentada por uma máquina de Turing, sendo no máximo equivalente a um autômato finito. O perceptron de uma camada tem sérias limitações, que foram mostradas por Minsky e Papert (1988), mas serviram de inspiração para outras redes. A rede MLP utiliza o algoritmo de treinamento de retropropagação, que corrige seus pesos de acordo com uma resposta desejada por um supervisor. Ao fim do treinamento a rede está apta a classificar uma gama de padrões, incluindo padrões que nunca foram apresentados. A RH, por outro lado, mantém os pesos fixos e altera em suas iterações o próprio padrão, até que ele se encaixe em alguma classe de padrões pré-definidos. É capaz de generalizar com razoável destreza, classificando padrões nunca antes apresentados. A RK altera seus pesos de modo a manter próximos neurodos que respondem a padrões próximos, criando com isso mapas especializados que respondem a classes diferenciadas. As classes são criadas pela própria rede, não tendo um supervisor que as definam de antemão. Essas propriedades que as redes aqui expostas apresentam são ditas emergentes, pois resultam da interação dos elementos constituintes das RNA e não são frutos de um planejamento direto (*top-down*) feito pelo projetista. O surgimento dessas propriedades pode ser observado em simulações dessas redes em MVN. Teremos oportunidade de voltarmos a esse ponto.

No próximo capítulo, tentaremos colocar alguma luz na confusão que existe entre os planos de análise propostos pelos adeptos de cada vertente. Um dos planos de análise que deveremos nos prender relaciona-se com a implementação física das máquinas que simulam os modelos propostos. Neste

contexto, traçaremos as distinções entre as máquinas connexionistas e as máquinas de Turing. Discutiremos também a capacidade da máquina de Turing implementar tais modelos sem perda das características que lhes são essenciais.

3.4 - Notas

¹ Também se tem conhecimento da existência de conexões entre dois axônios.

² Figura exemplo nosso.

³ Figura exemplo nosso.

⁴ Tradução nossa: Uma rede neural (artificial) é um *sistema de processamento de informação* que é não algorítmico, não digital, e intensivamente paralelo. Grifo nosso. Devemos notar que este livro vem acompanhado de um disquete que contém simuladores para computadores PC's e Macintosh's, que simulam, entre outras, as redes de Kohonen e o algoritmo de treinamento por retropropagação (Ibidem, p.15).

⁵ Inspirado em Haykin, 1999, p.13.

⁶ ARAÚJO, A. F. R. (São Carlos, USP). Comunicação pessoal, abr. 2000.

⁷ Ibidem.

⁸ Tradução nossa: O ajuste feito a um peso sináptico de um neurônio (neurodo) é proporcional ao produto do sinal de erro pelo sinal de entrada da sinapse em questão.

⁹ Tradução nossa: É crucial para o desenvolvimento de qualquer modelo (PDP) ter um claro modelo do meio-ambiente no qual este modelo (PDP) irá existir. Nos modelos PDP nós representamos o meio-ambiente como uma função estocástica variável no tempo sobre um espaço de padrões de entrada.

¹⁰ Por planejamento entendemos a criação ou a modelagem de uma RNA, por parte de um projetista, com o objetivo específico de resolver determinada tarefa ou problema.

¹¹ Tradução nossa: 1- A atividade de um neurônio é um processo tipo 'tudo-ou-nada'; 2- Um certo número fixo de sinapses deve ser excitado durante um intervalo de tempo de latência da adição, para que excite um neurônio naquele tempo, e este número é independente de atividade anterior e da posição no neurônio; 3- O único atraso significativo no sistema nervoso é o atraso sináptico; 4-

A ativação de qualquer sinapse inibitória bloqueia absolutamente a excitação do neurônio naquele tempo; 5- A estrutura da rede não muda com o tempo.

¹² Figura exemplo nosso.

¹³ Figura exemplo nosso.

¹⁴ Figura e modelagem, exemplo nosso. Representa uma possível solução para a modelagem do processamento interno de um neurodo de McCulloch e Pitts, usando grafos.

¹⁵ Figura e modelagem, exemplo nosso. Essa figura clarifica a idéia de utilizarmos uma máquina de Turing de múltiplas fitas para simular uma rede com vários neurodos. Grosso modo, esse exemplo também clarifica o funcionamento de uma MVN simulando uma RNA.

¹⁶ Figura e modelagem, exemplo nosso.

¹⁷ Figura e modelagem, exemplo nosso. Nessa figura podemos constatar uma possível formação conjunta de máquinas de Turing, configuradas para a realização de uma rede neural artificial paralela e distribuída. Veremos mais sobre as propriedades de processamento paralelo e distribuído no próximo capítulo.

¹⁸ ARAÚJO, A. F. R. (São Carlos, USP). Comunicação pessoal, 2000.

¹⁹ Note que o passo 1 do algoritmo da rede de Hopfield é executado uma única vez, na inicialização dos pesos. A partir daí os pesos permanecem os mesmos durante todo o processo de convergência.

²⁰ Inspirado em Rich e Knight, 1993, p.594.

²¹ Figura (A) e (B), BARRETO & ARAÚJO, 2001. Figura (C), exemplo nosso.

Capítulo 4 - Simulando o neconexionismo

4.1 - Apresentação

Neste capítulo colocaremos juntos dois tópicos que parecem, na literatura disponível, incompatíveis, mas ao mesmo tempo utilizados continuamente para darem suporte um ao outro, de forma dissimulada e aparentemente ingênua. Por um lado, temos a máquina de Turing, utilizada pelos adeptos da IA para processar seus programas simbólicos (os modelos da IA), algorítmicos, baseados em tipos abstratos de dados, com memória localizada e demais propriedades que compartilham as características da máquina de Turing. Por outro lado temos os modelos de redes neurais artificiais que, teoricamente, possuem memória distribuída, processamento em paralelo e não-simbólico de informação (ou pelo menos sub-simbólico¹), não algorítmico², e até, talvez, auto-organizadas. O que vemos, entretanto, nos textos que acessamos, são esses mesmos modelos sendo simulados em máquinas de Turing e apresentando as propriedades que deveriam emergir, em princípio, somente em máquinas conexionistas. Neste contexto coloca-se uma pergunta instigante: as simulações deixam de lado alguma propriedade importante e, nesse sentido, não seriam simulações completas das RNA? Em caso de uma resposta positiva, isto é, essas

simulações deixam de lado importantes propriedades, isso seria devido a um problema ontológico ou apenas uma decorrência de limitações tecnológicas? Em caso de resposta negativa, teríamos uma equivalência no plano formal (veja figura 21). A equivalência no plano formal poderia ser levada para um plano superior, onde se trata dos modelos? Tais aspectos constituem o fio condutor deste capítulo.

Começaremos, na seção 4.2, traçando uma discussão sobre os diferentes planos de investigação possíveis num mesmo objeto de estudo, para não incorrerem em erro quando comparações se fizerem necessárias. Na seção 4.3 discutimos as principais características das máquinas e dos modelos connexionistas, ponderando sobre o que pode haver de novo nelas que extrapole o mecanismo da máquina de Turing, impedindo sua completa simulação, e quais as diferenças ou vantagens de se utilizar o modelo da IA ou da RNA.

4.2 - Planos de análise

Um movimento começou a surgir em 1980, caracterizando o neoconexionismo (BROADBENT, 1985; RUMELHART & McCLELLAND, 1989), que discute justamente o que é ou não apropriado aos modelos cognitivos da mente. Esse movimento foi uma resposta natural, dada à conferência ministrada por Ashby em 1952 (DUPUY, 1995, p.210), que introduziu as noções de complexidade e ofereceu uma explicação da inteligência fundamentada em redes de autômatos que incorporassem em seus mecanismos o papel do *acaso*³. A importância de Ashby (1962) na modificação do paradigma *paleoconexionista* para o neoconexionista também se fez notar em seu artigo "*Principles of the Self-*

organizing System”, no qual discute a contradição de haver funções que se auto-organizam, isto é, que modifiquem a si próprias de acordo com a eficiência de sua atuação no meio. É característica do neoconexionismo uma maior aproximação com teorias de sistemas dinâmicos e com a auto-organização, bem como discussões antes inexistentes sobre a representação mental mapeada em redes neurais artificiais.

McCulloch e Pitts, como pudemos ver no capítulo anterior, acreditavam que suas redes eram mais uma prova da universalidade da máquina de Turing. A modelagem do sistema cognitivo, na visão desses cientistas, é algo que se dilui numa lacuna intermediária entre o neoconexionismo e a BOVEIA, uma criação que envolvia RNA como ferramenta e a pré-definição das características do cérebro como algoritmos lógicos. Uma modelagem que podia, pela equivalência que McCulloch e Pitts *acreditavam*⁴ ter demonstrado entre suas redes e máquina de Turing, ser transposta de uma para outra (DUPUY, 1995, p.119).

A novidade do neoconexionismo é que veio justamente para tentar quebrar, de forma mais explícita, a abordagem top-down (da criação de sistemas mecânicos cognitivos através de projetos feitos de antemão). Na abordagem bottom-up, o sistema cognitivo (no modelo da RNA) é projetado para lidar apenas com as possibilidades, sem, entretanto, se prender a particularidades. O acaso aplicado às conexões das unidades do sistema é continuamente absorvido, auto-adaptado pelo sistema próprio, no confronto com os padrões entrados e as respostas esperadas ou criadas, em busca de estabilidade (DUPUY, 1995, p.199).

Discutir RNA nesses termos é tratar dos modelos de RNA, e não de máquinas físicas que implementam esses modelos. Como funções matemáticas compostas, os modelos são descritos na forma de algoritmos, como vimos no capítulo 3.

Apontamos no capítulo 1 que um grande passo foi dado para a elucidação da discussão sobre os modelos que melhor se aplicam à explicação da natureza da mente inteligente. David Marr, em 1982, dividiu a abordagem explanatória dos modelos cognitivos em três grandes níveis que se tornaram referência obrigatória quando se pretende comparar os modelos simbólicos da IA com os modelos numéricos das RNA (RUMELHART & MCCLELLAND, 1985):

(1) O nível computacional: neste nível se descreve o objetivo da computação, por que a computação é apropriada e qual é a lógica da estratégia pela qual a computação pode ser realizada nos modelos explicativos, por exemplo, da visão.

(2) O nível representacional e algorítmico: neste nível se descreve como a computação estruturada em (1) pode ser modelada, qual é a representação apropriada para a entrada e saída, e qual é o algoritmo que realiza a transformação de uma em outra.

(3) O nível implementacional: aqui se descreve como a representação e o algoritmo caracterizados em (2) podem ser realizados fisicamente.

Há divergências entre os neoconexionistas (RNA) e os cognitivistas (IA) quanto aos dois primeiros planos, e principalmente em relação

ao segundo, mas encontramos pouca discussão sobre o terceiro plano, o implementacional.

Em se tratando de uma explicação psicológica, os modelos cognitivos propostos por qualquer teoria devem, na visão de Broadbent (1985), por exemplo, focalizar o plano computacional. Esse nível explicita o plano mais geral da maquinaria cognitiva, e deveria, por isso mesmo, ser o único de interesse dos cognitivistas (em especial dos psicólogos), que pretendem entender os mecanismos lógicos que levam a uma tomada de decisão inteligente. Não interessa, no plano psicológico, saber do funcionamento neural do indivíduo, que estaria vinculado ao nível representacional e algorítmico, e teria explicações a dar somente ao plano fisiológico do cérebro.

Broadbent admite que os modelos das RNA, por serem modelos de processamento distribuído, estão em maior conformidade com as neurociências que os modelos clássicos da IA. Entretanto, ele ressalta que esse detalhe (e outros que veremos adiante) está enraizado no nível implementacional, e não no computacional, conforme ilustramos na seção 2.4. Ainda que o processamento distribuído pudesse oferecer qualquer vantagem para uma explicação psicológica da mente, existe uma equivalência computacional (de computabilidade⁵) entre o processamento distribuído e o centralizado, que transforma qualquer vantagem de um sobre o outro em fato irrelevante, mera questão pragmática.

It is known that a simple Turing machine, with a single tape storage on which specific entries are made by a single processing head, is equivalent in

computational power to alternative configurations with multiple heads, multiple tapes, or even multidimensional systems. (BROADBENT, 1985, p.190)⁶

Com essa argumentação, Broadbent não mina, como desejaria fazê-lo, o atrativo aos modelos das RNA. De fato, o processamento distribuído pode ser, em princípio, simulado em máquinas de Turing, conforme mostramos quando expomos a equivalência entre máquinas de Turing de uma e de múltiplas fitas. Simular uma máquina de Turing de múltiplas fitas em uma de fita única requer apenas um procedimento que compatibilize as operações realizadas em uma na outra. A simulação dessa característica da RNA mostra como é sutil a passagem do nível implementacional (da máquina física) para o nível representacional e algorítmico. Broadbent erra o alvo, pois confunde esses dois níveis, levando Rumelhart e McClelland (1985) a rebater suas críticas:

Whereas Broadbent acknowledges only two levels of theory, the computational and the implementational, Marr actually proposes three, the computational, the algorithmic, and the implementational levels. (RUMELHART & MCCLELLAND, 1985, p.193)⁷

Concluem dizendo: *As psychologists, we are committed to an elucidation of the algorithmic level (RUMELHART & MCCLELLAND, 1985, p.194)⁸*. Assumindo a defesa do nível representacional e algorítmico, deixando claro que acreditam na interdependência e importância dos outros dois níveis, Rumelhart e McClelland argumentam que não se trata aqui de diferentes níveis, mas sim de modelos (matemáticos) diferentes⁹.

Esquematizamos na figura 21 os Planos de Haselager, que mostram o papel universal da máquina de Turing, na opinião de seu autor (HASELAGER, 2001)¹⁰. No plano superior temos a execução dos modelos das RNA e da IA, realizada em cima da máquina de Turing, que, conforme argumenta Haselager, são todos modelos Turing-computáveis. Os adeptos da IA incorrem em erro se atribuírem a seus modelos algorítmicos capacidades explanatórias do nível computacional, bem como erram ao tentar minimizar os modelos das RNA para o terceiro nível, o implementacional. Como teremos oportunidade de ver na próxima seção, ambos modelos são representacionais e algorítmicos, e, portanto, se encontram no mesmo nível explanatório de Marr, a saber, o segundo. As explicações que ambos modelos oferecem sobre a natureza da mente são hipóteses criadas com o estudo do funcionamento de seus respectivos algoritmos e de sua dinâmica, e nem são empiricamente comprováveis, uma vez que não se pode contrastá-las com a *verdadeira* natureza da mente. O máximo que podem indicar, o que já seria um ótimo começo, é que seus algoritmos *simulam* uma mente (SEARLE, 1997, p.311).

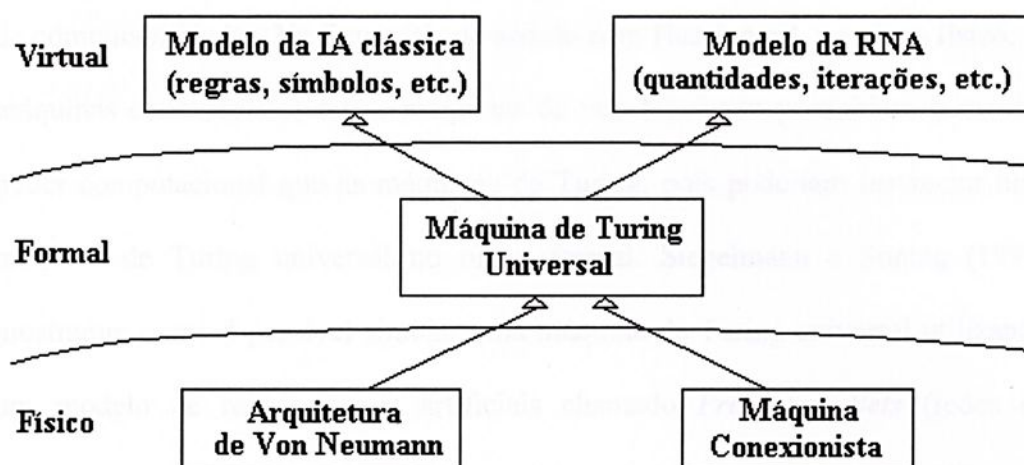


Figura 21: Planos de Haselager

Rumelhart e McClelland (1985), entretanto, confundem a equivalência levantada por Broadbent, quando este aponta o fato de que a computação distribuída não gera qualquer aumento no escopo das funções computáveis que tais máquinas distribuídas poderiam realizar. Essa equivalência está no plano formal (figura 21), que fica intimamente relacionado com o nível implementacional de Marr:

Casual appeals to Turing's thesis may be sufficient to establish equivalence at the computational level, but they are generally of little use to us in psychology, precisely because we are concerned with stronger forms of equivalence of our models to psychological processes. (RUMELHART & MCCLELLAND, 1985, p.195)¹¹

Não é uma equivalência no nível computacional proposto por Marr que a tese de Turing-Church estabelece, mas sim no nível implementacional, no qual, não importa o tipo de máquina que se fabrica, se ela consegue realizar cálculos efetivos é um tipo de máquina de Turing. É, portanto, uma equivalência

de computabilidade⁵. Na figura 21, de acordo com Haselager¹², no plano físico, as máquinas conexionistas ou as máquinas de von Neumann possuiriam o mesmo poder computacional que as máquinas de Turing, pois poderiam instanciar uma máquina de Turing universal no plano formal. Siegelmann e Sontag (1991) mostraram como é possível simular uma máquina de Turing universal utilizando um modelo de redes neurais artificiais chamado *Processor Nets* (redes de processadores), e citam Pollack que também resolveu esse problema utilizando redes recorrentes:

*In particular, and ignoring issues of training and practicality of implementation, one would like to know if every problem that can be solved by a digital computer is also solvable – in principle – using a net. [...] Indeed, Jordan Pollack (1987) showed that a certain recurrent net model [...] is universal. (SIEGELMANN & SONTAG, 1991, p.1)*¹³

Com o modelo de rede equivalente às máquinas de Turing proposto por Siegelmann e Sontag, concluímos que existe pelo menos uma rede neural artificial que computa uma dada função recursiva qualquer, ou melhor, uma dada função Turing-computável. Logo, as redes neurais artificiais são, no mínimo, tão poderosas quanto (equivalentes às) as máquinas de Turing. Um pouco mais adiante nesse mesmo artigo, Siegelmann e Sontag (1991) averiguam de maneira, infelizmente, não tão óbvia para nós:

It is obvious that one can simulate a processor net with a Turing machine, as we took all values to be rational. Our main remark is that, conversely, any

function computable by a Turing machine can be computed by a processor net.
(SIEGELMANN & SONTAG, 1991, p.2)¹⁴

Ao admitir que vale a recíproca, Siegelmann e Sontag nos indicam que as redes de processadores não são mais poderosas que as máquinas de Turing, mas não eliminam nosso questionamento. Uma vez que temos um modelo de RNA que é equivalente a uma MT, podemos ter um outro, mais poderoso que a MT?

IA	RNA
Simbólico.	Não-Simbólico.
Modelos: Ênfase no nível computacional.	Modelos: Ênfase no nível algorítmico e representacional.
Nível Implementacional: Máquinas von Neumann.	Nível Implementacional: Máquinas Conexionistas.
Representações sintaticamente estruturadas em TAD.	Representações como padrões ativos de conexões entre neurodos.
Cognição realizada através de regras.	Problemas resolvidos pela estabilização da rede em estados que ponderam as restrições e custos.
Múltiplas restrições programadas seqüencialmente.	Múltiplas restrições e custos codificados na rede de uma só vez.
Representações armazenadas.	Representações ativas presentes, e inativas como estados disposicionais.
Plano Formal: Máquinas de Turing	

Tabela 4: Comparações entre IA e RNA¹⁵

A tabela 4 apresenta uma outra forma de visualizar o problema de discutir e comparar os conceitos conexionistas dos conceitos cognitivistas. O simbolismo da IA, como visão filosófica, argumenta no plano mais geral de análise que é importante explicar o funcionamento do sistema cognitivo utilizando-se de símbolos e estruturas algorítmicas, pois somente assim poderíamos satisfazer a ânsia de conhecer como os processos psicológicos ocorrem no cérebro. Na outra coluna, o fisicalismo não-simbólico da RNA propõe que o estudo das neurociências e dos sistemas dinâmicos compostos por neurodos similares a neurônios que – em constante interação – produzirão os próprios símbolos, que por isso não podem ser pré-programados, com a penalidade de se inserir *a priori* justamente o que se quer explicar *a posteriori*, ou seja, como o cérebro humano lida e constrói seus próprios símbolos. O posicionamento do cientista como simbolista ou não-simbolista é determinante na produção dos modelos que pretende implementar, apesar de ambas visões compartilharem hipóteses fundamentais da ciência cognitiva, como tivemos oportunidade de ver no capítulo 1.

Uma dessas hipóteses compartilhadas, devemos lembrar, é acreditar na possibilidade de se construir uma *máquina* capaz de apresentar inteligência. As duas vertentes, com visões filosóficas distintas (e com pontos em comum), criaram modelos para solucionar problemas específicos em cada uma. Os adeptos da IA propõem algoritmos de buscas, estratégias de lidar com conhecimento representado em tipos abstratos de dados, estruturas simbólicas agrupadas em regras que podem ser executadas como procedimentos efetivos. A

máquina de von Neumann, implementação da máquina de Turing, é perfeita para seus propósitos. A corrente da IA não está presa a processos sequenciais, centralizados, digitais, ou a qualquer outro limitador de como esses algoritmos serão implementados. Os adeptos da IA utilizam computadores paralelos ou não, com memória distribuída ou não, analógicos ou não, importando apenas que seus modelos possam ser efetivamente computados. Os adeptos das RNA, por sua vez, construíram seus modelos matemáticos com a intenção de explicar o sistema cognitivo pela emergência de propriedades de suas redes altamente conectadas, paralelas, que tratam de quantidades analógicas ao invés de símbolos em um alfabeto finito. Mas esses modelos não fogem de uma das hipóteses compartilhadas lembrada acima, a saber, que são modelos que podem ser realizados mecanicamente, em uma implementação física, e não experiências de pensamento sem nenhuma aplicabilidade prática. A questão sobre qual *modelo* é mais adequado à explicação psicológica dos sistemas cognitivos que é encontrada na literatura sobre o pano de fundo da discussão de planos de análise. Essa questão é pertinente, e pretendemos abordá-la analisando tanto o nível algorítmico quanto o implementacional, enfatizando, entretanto este último, onde esses modelos estão implementados.

Turing (1950) afirma que sua máquina, devidamente programada e com memória suficiente, é capaz de passar no teste de Turing, o que poderíamos dizer, em outras palavras, que a inteligência humana (que é desafiada pelo teste de Turing) é uma *função* Turing-computável. A criação dessa função é o objetivo da IA; a análise dela, uma vez criada, é o objetivo dos psicólogos



cognitivistas. Questionamos aqui se os adeptos das RNA modificaram o paradigma da criação de modelos mecânicos a tal ponto, que conseguiram fugir das funções Turing-computáveis. Se os modelos das RNA se mostrarem Turing-computáveis, e as máquinas connexionistas casos particulares das máquinas de Turing, o que os connexionistas estão modelando não passam, afinal, de funções cujo processamento efetivo elevaria finalmente a máquina de Turing ao posto que seu criador lhe reservou: o modelo universal e mecânico da mente. Uma questão imediatamente se coloca: os modelos das RNA são capazes de resolver funções não Turing-computáveis? Ora, criar um modelo de RNA para responder essa pergunta e implementá-lo fisicamente em uma máquina de von Neumann para simulá-lo é uma contradição, uma vez que estas são instâncias de máquinas de Turing. Já implementá-lo em uma máquina connexionista nos leva a outra questão: as máquinas connexionistas são casos particulares de máquinas de Turing? Se assim forem, ambas as questões estarão resolvidas, senão, teremos uma nova máquina que realiza procedimentos efetivamente calculáveis que estão fora do escopo computacional das máquinas de Turing, o que demonstraria, através de um contra-exemplo que a tese de Turing-Church é equivocada. Sabemos que até hoje, todas as funções efetivamente computáveis se revelaram Turing-computáveis, e por isso mesmo a TTC de que não existe procedimento efetivo que não seja Turing-computável ganha força a cada dia.

Na discussão sobre planos de análise, pretendemos colaborar com um plano que não tem recebido a devida atenção: o nível implementacional. É sobre esse nível que se desenvolvem os outros dois. Só poderemos pensar em

uma abordagem realmente distinta entre ambas as vertentes se pudermos conceber que os modelos das RNA não estão incluídos nas funções Turing-computáveis. Caso contrário, teremos que admitir um domínio comum entre ambas no baixo nível implementacional, e considerar a discussão de suas distinções apenas no plano da abordagem explicativa dos modelos, levando em conta o fato de que qualquer uma de ambas as vertentes que conseguirem finalmente o objetivo maior de modelar a natureza da mente terá colaborado positivamente para a conclamação de Turing (1950). Em outras palavras, se não for possível criar máquinas de cálculo efetivo que extrapolem as funções recursivas, optar pela RNA ou pela IA seria apenas optar por um melhor algoritmo, e não por um paradigma diferente, uma vez que ambas estariam limitadas ao formalismo das máquinas de Turing. Considerando ainda esse caso, se questionarmos a afirmação de que a mente humana é redutível às funções Turing-computáveis, devemos abandonar ambas as abordagens e abandonar o computador digital como modelo apropriado para explicar a atividade mental na ciência cognitiva?

4.3 - Principais características do connexionismo

Passaremos a analisar nesta seção as características relevantes às máquinas e aos modelos connexionistas que poderiam torná-los capazes de realizar computações que estariam além daquelas classificadas como Turing-computáveis. Em cada subseção que se segue, enfocaremos primeiramente a máquina e em seguida faremos considerações sobre os modelos, terminando com uma breve conclusão ou intuição. Focalizaremos o processamento numérico de informação

versus o processamento simbólico; processamento analógico (ou contínuo) *versus* processamento digital (ou discreto); processamento paralelo *versus* sequencial; processamento e armazenamento distribuído *versus* centralizado; redundância; tolerância à falhas e representação (HONAVAR, 1995). Também trataremos da plausibilidade biológica e da visão de redes neurais artificiais como funções.

4.3.1 - Processamento numérico *versus* processamento simbólico

Desde sua invenção, o computador digital tem sido utilizado tanto para processar estruturas simbólicas como cálculos numéricos, como aponta Honavar (1995), em citação transcrita adiante. A máquina de Turing, pela sua simplicidade, torna fácil compreender que internamente a um computador digital tudo o que teremos em sua memória está reduzido a símbolos. O que torna então esses computadores capazes de realizar operações numéricas (com a precisão que se queira)?

A resposta está na sutil diferença entre o valor de um número e o seu símbolo representativo (numeral). Trabalhando apenas com processamento simbólico, uma máquina de Turing é capaz de computar o resultado de uma soma, de uma multiplicação, e até mesmo de uma comparação entre dois números, que são representados com seus símbolos na fita da máquina e manipulados segundo as regras do algoritmo implementado, sem compreender o significado do valor numérico de cada símbolo manipulado.

But numbers are in fact symbols for quantities; and any computable function over numbers can be computed by symbolic process. In fact, general purpose

digital computers have been performing both symbolic as well as numeric processing ever since they were invented. (HONAVAR, 1995, p.15)¹⁶

O processamento simbólico significa apenas que uma estrutura ou conjunto de símbolos é manipulado de acordo com certas regras que produzem uma outra estrutura ou conjunto diferente de símbolos como saída (HONAVAR, 1995). Ao trabalhar com quantidades, as MC recebem como parâmetros de entrada um padrão, composto de várias medidas de vários sensores de entrada, que podem seguir para seus neurodos como números codificados por símbolos (provindos de sensores digitais), ou como quantidades analógicas (provindas de sensores analógicos). Veremos a diferença entre analógico e digital na próxima seção, entretanto, nesta seção, basta compreendermos que os modelos concebidos de RNA possuem neurodos que não avaliam ou compreendem o significado do valor passado a eles. Eles apenas aplicam esses valores, sejam um conjunto de símbolos ou uma quantidade física (como a voltagem, por exemplo) a uma função que realizará operações sobre esses valores e gerará o resultado modelado pelo pesquisador. A função computada deve ser realizada fisicamente em uma máquina e dar a resposta computada independente do tipo de máquina. Sendo uma máquina de Turing, a função será computada de acordo com manipulações nos símbolos de entrada que representam a quantidade, e terá como saída um valor numérico também representado por uma cadeia de símbolos na fita.

Quando os adeptos da RNA argumentam que não há manipulação simbólica nos *modelos* de RNA, estão se referindo ao plano algorítmico e representacional, e não ao plano implementacional, (no qual, na

verdade, não importa como o valor numérico é interpretado) uma vez que estão preocupados em entender a dinâmica da rede como um todo, e não uma particularidade em nível tão baixo como o implementacional. Nesse sentido, basta que o neurodo responda adequadamente ao estímulo apresentado, seja ele um valor representado por uma cadeia simbólica ou uma quantidade física. Zipser e Rabin (1989), ao proporem um simulador de redes neurais artificiais, exemplificado com uma rede de aprendizado por competição que, segundo os autores, foi muito bem simulada, ressaltam:

Before the start of a simulation, values must be given to all parameters and to all outputs. Each value is always a single computer word in length. The interpretation of this word depends on how the methods use it. (ZIPSER & RABIN, 1989, p.491)¹⁷

O termo *palavra* (*word*) no computador significa um conjunto de símbolos, com número fixo de elementos, que varia de máquina para máquina e tem aumentado de tamanho acompanhando a evolução e o barateamento das memórias. Algumas máquinas mais antigas consideravam as palavras (*words*) como conjuntos de dois *bytes*, assim: *bits*, representam dois símbolos (o zero e o um) em uma única célula de memória; *bytes*, o conjunto de oito *bits*, capaz, portanto, de representar 256 objetos (ou representar os números naturais de 0 a 255, por exemplo); *palavra* (*word*), composto de dois *bytes*, que o torna capaz de codificar 65.536 objetos distintos (ou representar os números naturais de 0 a 65.535, ou então números de -32.767 a +32.768 incluindo o zero, ou mesmo todos os números com duas casas de precisão depois da vírgula de 0,00 a 499,99). A

interpretação de tal *palavra* (*word*), como aponta Zipser e Rabin, depende dos métodos (que são algoritmos escritos em LISP, neste simulador) utilizados por cada neurodo em particular, para computar sua saída. A manipulação de palavras (*words*) pelos algoritmos, (por exemplo, de duas palavras (*words*) em conjunto, formando através de regras uma terceira palavra (*word*) que quando vista por um observador permite a inferência de que a terceira representa um valor numérico que é a soma dos valores numéricos representados pelas outras duas) não significa, em absoluto, que este sistema compreende estes símbolos ou a própria operação do algoritmo em questão (no exemplo, a adição).

Não é na compreensão dos símbolos que reside a força explicativa das máquinas de Turing, mas na sua capacidade de manipulação. Assim como não é na interpretação dos valores entrados em cada neurodo em particular que reside a força explicativa das máquinas conexionistas, mas na sua correta computação e comunicação de sinais (sejam símbolos ou quantidades). Quando se trata da investigação de tarefas realizadas em níveis tão próximos do físico, dificilmente atribuímos qualquer tipo de compreensão a tais mecanismos simplórios, seja na computação de um limiar de disparo em um neurodo (da RNA) ou na computação de uma comparação de uma regra *se-então* (da IA).

A análise da compreensão de símbolos na RNA e na IA deve se prender aos modelos propostos, bem como na dinâmica de realização de seus algoritmos completos. Da mesma forma que os símbolos que os modelos de RNA supostamente explicam não são os que representam os valores de entrada nos neurodos, mas o *conjunto* composto pela ativação e estabilização de toda uma



rede em um processo de convergência, assim também os símbolos que a IA atribui ao sistema cognitivo não são os do alfabeto da máquina de Turing, mas *conjuntos* complexos de dados que podem representar sistemas lógicos completos e suas regras de inferência. No nível algorítmico, os símbolos devem ser entendidos como conceitos, e não mais como letras de um alfabeto. Especialmente nas simulações das redes de Kohonen em MVN podemos notar a criação de categorias que classificam os padrões de entrada de acordo com algum critério, como por exemplo a cor, a forma ou a similaridade, e portanto criam conceitos. Alguns programas mais complexos da IA, como tivemos a oportunidade de ver no capítulo 1, também possuem a capacidade de criar conceitos utilizando técnicas diferentes (WINSTON, 1984). Notamos que, apesar de utilizarem diferentes métodos, ambas as vertentes têm respostas satisfatórias para o problema de como a mente cria símbolos, ou conceitos. A diferença no nível representacional e algorítmico fica por conta de uma maior plausibilidade psicológica na IA contra uma maior plausibilidade neurológica na RNA.

No plano implementacional, para o caso das RNA, se é possível utilizar símbolos para representar números, não teremos distinção entre os valores numéricos (quantidades) e sua representação simbólica. Há, entretanto, números que não podem ser representados por um conjunto finito de símbolos, como os irracionais, por exemplo, e devem ser aproximados a uma precisão adequada, mas esse é um caso a ser abordado no próximo tópico.

4.3.2 - Processamento analógico *versus* processamento digital

O processamento analógico é realizado por sistemas dinâmicos que lidam com valores contínuos e podem ser descritos por equações diferenciais contínuas. Tais sistemas não possuem transições de estados discretos como a máquina de Turing, mas operam em tempo contínuo. Também não têm limitado em precisão os valores com os quais têm que lidar, que são quantidades físicas que sofrem as computações por processos contínuos resultando em valores também contínuos como saída. Normalmente essas quantidades, em computadores analógicos, são tensões de energia elétrica, mas podem ser de qualquer tipo, como fluxo de água, de ar, quantidade de atrito, calor, afinidade, permeabilidade, etc (HONAVAR, 1995, p.15).

Na forma como a computabilidade foi definida por Turing, é algo confuso definir processamento analógico como computação. Por não haverem passos discretos na mudança de estados, nem haver um conjunto finito de estados possíveis, o sistema analógico se descaracteriza, de certa forma, como sendo equivalente às máquinas de Turing, o que significaria dizer que tais máquinas não possuem um *procedimento efetivo* de cálculo. Não devemos deixar de notar, mas, que a máquina de von Neumann é um sistema analógico, montado com circuitos (como o transistor, por exemplo) que são analógicos, que, com certas restrições, podem responder de forma discreta. Um circuito CMOS, por exemplo, entende como entrada 0 valores de 0 V a 2,7 V; como 1, valores entre 6,3 V e 9 V; e responde de forma indefinida a valores intermediários entre 2,7 V e 6,3 V (GOÑI, 2000). Uma máquina conexionista analógica não perderá, por esse

mesmo motivo, a característica de que um neurodo fisicamente implementado só disparará quando certo limiar for atingido, não disparará quando esse limiar se encontrar abaixo de determinado valor, e poderá responder de forma indefinida na faixa intermediária.

A característica de um sistema possuir precisão infinita, ou finita e extremamente grande, não influencia na utilização de máquinas discretas para simulá-lo se a precisão for desprezível a partir de algum patamar e puder ser superada pelas técnicas disponíveis atualmente. O mesmo não aconteceria se pretendêssemos simular o universo como um todo, no que necessitaríamos de um sistema computacional de precisão superior à do próprio universo.

A questão torna-se saber se os limites de precisão necessários para simular um cérebro podem ser atingidos por uma máquina de Turing com memória suficiente, e tecnicamente viável de ser implementada fisicamente: *"Analog systems can be, and often are simulated on digital computers at the desired level of precision"* (HOVANAR, 1995, P.15)¹⁸. Para modelos realistas do cérebro, onde toda a química, o metabolismo e a comunicação do meio, além da elétrica, devem ser simulados, esse é um ponto importante. Todavia, nos modelos em que os neurônios foram simplificados para se comunicarem pelos pulsos de seus axônios e dendritos, esse limite foi superado. A comunicação elétrica interneurônios é lenta o suficiente, como pudemos ver no capítulo anterior, para que os computadores atuais dêem conta de simulá-las no tempo correto. Há vários outros exemplos em que vemos os limites do sistema analógico humano serem superados por sistemas digitais artificiais, como por exemplo, a impressão quadro

a quadro das telas de cinema, a televisão digital de alta definição, os telefones digitais, sistemas de comunicação por satélite, etc. Mesmo os MRC sempre deixam de lado algumas características, seja por ignorância do real funcionamento do cérebro, seja visando a sua simplificação, permitindo que tais modelos sejam simuláveis, sem perda de propriedades relativas à precisão da resposta. Qual é a precisão que podemos simular, onde está o limite entre a precisão do sistema analógico humano (se houver) e a precisão dos modernos computadores digitais, cada vez maiores (por terem cada vez mais memória), permanecem como questões em aberto. O que não se discute é que a qualidade dos valores que podem ser representados nos computadores atuais é tamanha que o que se despreza, em termos matemáticos, o erro, a diferença entre o real e o representado, é tão ínfimo que podemos analisar experimentos nunca antes concebidos, teorias sobre o universo microscópico, o controle e posicionamento de satélites em órbitas, simulações de bombas atômicas, simulações de movimentos planetários, etc., tornando o erro valor que na prática se mostra realmente desprezível. Sem contarmos, é claro, com a existência de técnicas de correção de erro, pela realimentação e reavaliação dos valores originais, o que contribui para minimizar ainda mais sua influência.

Além da precisão dos valores numéricos temos, outro ponto característico do processamento analógico: o *tempo contínuo*. Nas máquinas de Turing devemos discretizar o tempo em intervalos que podem ser tão curtos quanto idealmente se queira. Nos neurônios biológicos, os pulsos são recebidos de forma contínua por seus dendritos, e são processados assincronicamente. Uma



máquina de Turing que simula um neurodo tem pelo menos dois ciclos discretos: o recebimento do padrão e seu processamento. Estando em um ciclo, não realiza o outro. Uma vez lido o padrão de entrada, o processamento é feito em passos discretos (que consomem tempo), e até que sua saída não seja resolvida não será lido o próximo padrão de entrada. Se no intervalo de tempo em que a máquina de Turing estiver processando o padrão em sua memória, ocorrer uma modificação rápida e brusca no padrão de entrada, que logo volta ao valor anterior, a máquina não perceberá essa modificação. Tal variação no padrão deve ser tão rápida quanto o ciclo de processamento do neurodo para não ser percebida.

Podemos nos remeter, para bem esclarecer esse ponto, às mensagens sub-liminares que são apresentadas em imagens rápidas o suficiente para que o indivíduo não as torne conscientes, mas lentas o suficiente para influenciar os hábitos do referido indivíduo. Se a mensagem for apresentada muito rapidamente não surtirá nenhum efeito, indicando que o padrão apresentado não teve tempo suficiente para poder ser percebido pela retina, quem dirá tornar-se consciente. Também a retina tem o seu tempo mínimo de latência para ser excitada. No plano implementacional, a máquina de von Neumann deve ter ciclos suficientemente rápidos para conseguir processar tanto mensagens apresentadas em tempo razoável quanto apresentadas em tempos ínfimos e sub-liminares. O efeito psicológico das mensagens sub-liminares não deve ser explicado no plano implementacional, mas no plano computacional, algorítmico e representacional, nos quais os modelos cognitivos podem utilizar a informação do tempo de duração da mensagem apresentada para gerar suas explicações corretamente. A



máquina, connexionista ou de von Neumann, apenas deve permitir e disponibilizar tal informação.

De modo geral, as fatias de tempo da máquina de estados discretos, que pretende simular uma máquina analógica, devem ser finas o suficiente para tornar seus intervalos de ciclos irrelevantes no processo de simulação, desprezíveis na mesma precisão ou mais em que o são pelas máquinas que serão simuladas. Se houver na MT simuladora um único controlador para processar todos os neurodos da RNA, ela deve realizar o ciclo de processamento em todos os neurodos numa única fatia de tempo. Se houverem vários controladores, a restrição do tempo poderá ser um pouco relaxada, pois teríamos a opção de processar alguns, ou todos os neurodos em paralelo.

4.3.3 - Processamento paralelo *versus* processamento seqüencial

Processar em paralelo significa realizar duas ou mais computações ao mesmo tempo, o que implica necessariamente em se possuir dois ou mais processadores independentes para realizá-las. Existem problemas que podem ser resolvidos utilizando-se algoritmos paralelos, pelo menos parcialmente. Considere o problema de calcular a média de todas as provas realizadas em um ano, de todos os alunos de uma universidade. A média de cada aluno é independente das outras, e pode ser computada em paralelo, ou seja, simultaneamente, em relação à dos outros alunos. Se cada aluno assumir o papel de processador de suas próprias notas, o resultado de todas as médias será alcançado tão rápido quanto o aluno mais lento acabar seu cálculo. Se tivermos

apenas dois processadores, teremos a cada ciclo duas médias calculadas, e o tempo de se processar todos os resultados por um único processador cai aproximadamente pela metade. Por outro lado, a seqüência de operações que se deve realizar para calcular uma única média pode ser parcialmente, mas não totalmente, paralelizada. Suponha que o aluno em questão recebeu, entre provas e trabalhos, 15 notas. Trabalhando sozinho, ele deve somar as 15 notas e dividir o resultado por 15. Se um colega se dispuser a ajudá-lo, o aluno poderá somar apenas as 8 primeiras notas e pedir ao colega que some, em paralelo, as outras 7. Recebendo do colega o resultado parcial da soma das 7 notas e adicionando-o ao resultado parcial de sua soma das 8 notas, obterá finalmente o resultado da soma das 15 notas que agora poderá ser dividido, calculando-se finalmente a média. Não é possível, entretanto, seguir para o cálculo da média sem receber todos os resultados parciais das somas: a operação de divisão não pode ser realizada em paralelo às operações de soma, nesse caso. Em outras palavras, não existe nenhuma tarefa *única* que possa ser processada *totalmente* em paralelo, ou ela teria que ser composta de partes *completamente* desconexas e independentes.

A característica de trabalhar simultaneamente não altera o fato de que as funções que estão sendo computadas por cada processador em si são funções efetivas, computáveis por meio de várias máquinas de Turing independentes. Os neurodos, como vimos no caso das redes de McCulloch e Pitts, e no modelo geral proposto por McClelland e Rumelhart (1987) apresentado no capítulo anterior, figura 8, computam funções de soma, funções degraus, funções



sigmóides, e outras que se queiram, que são todas passíveis de serem realizadas em máquinas de Turing.

Alguns adeptos das RNA (MCCLELLAND & RUMELHART, 1987) argumentam que é na dinâmica de interação entre as unidades de neurodos trabalhando em paralelo que surgem as propriedades que se pretendem estudar e que são dadas de antemão nos modelos da IA. Não vemos essa arguição como um problema no nível implementacional. Ela só se torna relevante no nível representacional e algorítmico e, principalmente, no nível computacional, no qual deveria explicar se o comportamento inteligente do algoritmo escolhido, com a representação escolhida, serve de modelo para entender a natureza da mente. O fato é que essa questão, apesar de relevante no nível representacional e algorítmico – pois pode mostrar que várias unidades processadoras trabalhando em paralelo, cooperando ou competindo umas com as outras proporcionam um ambiente adequado para o surgimento de propriedades emergentes –, não nos obriga a deixar o espaço abarcado pelas máquinas de Turing. Se tais propriedades emergentes existem e são essenciais para o entendimento da mente, temos um divisor que mostra que os modelos de RNA são mais adequados que os modelos da IA. Isso não significa, sobremaneira, que as máquinas de Turing devem ser deixadas de lado. Ao contrário, modelos de RNA simulados em máquinas de Turing (por exemplo, a rede de Hopfield), apresentam tais propriedades – como memória associativa ou a convergência em soluções de problemas.

Máquinas paralelas podem ser simuladas por uma máquina de Turing, uma vez que Turing (1936) mostrou ser possível criar uma máquina

universal de Turing capaz de simular quaisquer outras máquinas de Turing dadas, e as máquinas paralelas nada mais são que conjuntos de máquinas de Turing. A única penalidade é a perda de velocidade, que conforme vimos na subseção anterior não pode ser maior que o nível de precisão requerido para realizar a computação de forma adequada para simular o sistema original. Dupuy (1995) apreende a inversão histórica de simular neurodos em máquinas de von Neumann quando a expõe:

Os neurônios são extremamente numerosos no cérebro e funcionam lentamente: a organização em paralelo é que otimiza o processamento da informação. Em contrapartida, os componentes de uma calculadora artificial podem alcançar velocidades muito grandes e são relativamente pouco numerosos: é preciso montá-los em série. Ainda hoje, quase todos os computadores - às vezes chamados de 'máquinas de von Neumann' - são máquinas sequenciais. Essa dominação é tão forte que até as redes neuronais do conexionismo, essas máquinas altamente paralelas, são *simuladas* sobre as máquinas de von Neumann. Esta é apenas uma dessas inversões de papéis de que esta história não é avara. (DUPUY, 1995, p.78)

Seqüencializar uma operação em paralelo requer velocidade de processamento suficiente para poder deixar momentaneamente uma das operações estagnadas enquanto realiza-se a outra, para rapidamente retomar a primeira, num ciclo constante, em que a troca do foco de processamento é feita em intervalos de tempo que não afetam nenhuma das operações em si, como uma criança brincando de manter no ar duas bolas de tênis quicando-as alternadamente com a mesma raquete.

Zipser e Rabin (1989) contornam o problema do paralelismo incluindo em seu simulador uma técnica de sincronização dos ciclos de cálculo dos neurodos, fazendo com que todos sejam atualizados em um único ciclo. Desta forma, trabalhando seqüencialmente em cada neurodo da rede, o processador calcula cada entrada e saída de cada neurodo que deveria estar em paralelo, num intervalo de tempo “congelado”, e só após o cálculo da saída do último neurodo é que as influências das modificações feitas são passadas e o ciclo é retomado. Algumas vezes é desejado que certos neurodos (ou grande parte deles, ou até todos eles) se atualizem de forma assíncrona. Isso é conseguido incluindo-se mais uma conexão ao neurodo, que criaria um intervalo de tempo especial para ele, entre um ciclo e outro da rede (ZIPSER & RABIN, 1989, p.489-90). Estes intervalos congelados de tempo, enfatizamos, devem ser curtos o suficiente para não interferirem na simulação que pretende se aproximar ao máximo da real condição do neurônio em seu ambiente.

A característica que as máquinas de Turing possuem de poderem trabalhar equivalentemente com múltiplas fitas, múltiplas cabeças, ou múltiplos controles em sistemas multidimensionais (como vimos na seção 2.4) ou com uma única fita, única cabeça e único controle conforme a proposição original, leva-nos a concluir que não é pelo paralelismo que se poderia defender que tais máquinas resolveriam problemas que não estivessem incluídos nas funções Turing-computáveis. A própria rede de McCulloch e Pitts, uma máquina altamente paralela, revelando-se equivalente aos autômatos finitos, demonstram que o mero

paralelismo não é fundamental para aumentar o poder computacional de uma máquina.

Um olhar mais atento aos modelos da IA e da RNA nos revela que o que temos na verdade é uma mistura de processamento paralelo e seqüencial. Muitas MVN modernas executam parte dos algoritmos, produzidos em princípio para serem seqüenciais, de forma paralela, sem o programador tomar conhecimento desse fato. Nas RNA, assim como a operação de divisão não pode ser realizada antes ou simultaneamente às operações de soma no cálculo da média do aluno, os neurodos pós-sinápticos só podem ter seus valores calculados depois do processamento dos neurodos pré-sinápticos. Para o quesito do processamento paralelo, tudo nos indica que não há aqui vantagem qualquer no nível implementacional. Não devemos, porém, generalizar tais conclusões para outros níveis.

4.3.4 - Processamento e armazenamento distribuído *versus* processamento centralizado

Há muita divergência sobre o significado que se dá ao termo distribuído, quando se levanta essa característica no connexionismo. Um dos problemas está relacionado ao que expomos na primeira seção deste capítulo, os níveis a que o termo se refere. No nível implementacional, uma máquina pode ter processamento distribuído, armazenamento distribuído ou ambos.

Uma máquina que apresenta processamento distribuído o possui por ser constituída de vários (mais de um) processadores, o que indica que essa



máquina pode (mas não é obrigada a) realizar processamento paralelo. O poder de computação de cada neurodo, capaz de realizar por si próprio a computação dos valores de entrada em suas conexões e calcular seu estado de ativação mostra claramente que as máquinas conexionistas possuem processamento distribuído. Como vimos na subseção anterior, as máquinas paralelas devem possuir processamento distribuído para realizarem suas operações concorrentemente. Não precisamos nos estender novamente na mesma argumentação dada na subseção anterior para indicar que essas máquinas distribuídas são passíveis de serem simuladas em máquinas de Turing, mais precisamente em máquinas de Turing de múltiplos controles, que são equivalentes às máquinas de Turing de único controle e única fita.

Já quando os conexionistas se referem à qualidade de armazenamento distribuído das máquinas conexionistas, pretendem utilizar o termo de uma forma inovadora que reflete o modo como as informações ficam representadas simultaneamente nos vários neurodos, tendo ainda que cada neurodo pode colaborar para a representação de vários padrões informacionais distintos e independentes (VAN GELDER, 1991). Essa é uma propriedade representacional do modelo de RNA, e não propriamente de uma suposta máquina conexionista.

Uma máquina conexionista teria neurodos implementados fisicamente que, ao assumirem um entre vários estados de ativação, comporiam o sistema de armazenamento da máquina. Como esse sistema seria utilizado, se aproveitando sobreposições da informação, ou se assumindo que um único



neurodo representaria uma informação completa, é característica do modelo executado na máquina, portanto do nível representacional e algorítmico, não da máquina em si. Poderíamos até mesmo conceber um modelo de RNA que utilizasse esses neurodos físicos de forma binária e codificasse o alfabeto seguindo a mesma codificação da tabela¹⁹ utilizada pelas máquinas de von Neumann. No que concerne às máquinas, elas devem ser capazes de oferecer a possibilidade de distribuição da informação na memória, e essa distribuição seguirá as regras do modelo cognitivo utilizado. Os neurodos ou as máquinas de Turing de múltiplas fitas oferecem a possibilidade de armazenamento distribuído, o que não aumenta o poder representacional dessas máquinas, como nota Honavar:

And, as pointed out by Hanson and Burr (1990), distributed coding in and of itself, offers no representational capabilities that are not realizable using non-distributed coding. (HONAVAR, 1995, p.20)²⁰

Não vemos, portanto, distinção crucial no nível implementacional, uma vez que podemos conceber sistemas distribuídos utilizando máquinas connexionistas ou máquinas de Turing. Entretanto, implementar nessas máquinas – connexionistas ou de Turing – modelos que distribuem a informação torna os algoritmos implementados mais capazes de lidar com, por exemplo, memória associativa, generalização, seleção de regras que melhor se adaptam a uma situação, e mesmo propriedades emergentes inesperadas, na visão de Rumelhart & McClelland (1989, p.78). Sabemos, ainda, que o cérebro humano armazena informação de forma distribuída (RUMELHART

& MCCLELLAND, 1989), o que tornam os modelos distribuídos mais condizentes com o funcionamento cerebral.

Não devemos confundir as características dos modelos e, portanto do nível algorítmico e representacional, com as características do nível implementacional. Parece-nos perfeitamente possível implementar modelos distribuídos de RNA em máquinas de Turing, o que indicaria que as máquinas conexionistas, no nível implementacional, seriam equivalentes às máquinas de Turing, para esse caso. Entretanto, apesar da suposta equivalência no plano implementacional, acreditamos que é mais natural modelar o cérebro humano através de algoritmos que utilizam informação distribuída, o que significaria que uma máquina de Turing utilizando tais heurísticas poderia apresentar melhores resultados na solução do problema de modelar a mente inteligente que uma máquina de Turing que utilizasse informação centralizada. Esses melhores resultados são características dos modelos e, portanto, do nível algorítmico e representacional: sabemos que existem vários algoritmos para resolver um mesmo tipo de problema, e que uns são mais eficientes que outros. A vantagem das RNA, para o caso de processamento e armazenamento distribuído, reside na sua eficiência em resolver os problemas levantados por Rumelhart & McClelland (1989) citados no parágrafo anterior.

4.3.5 - Redundância e tolerância à falhas

Redundância de processadores e memória causam naturalmente uma tolerância à falhas na execução de uma determinada tarefa, uma vez que o



processador ou memória que foi danificado não acarreta a parada geral do funcionamento da máquina. Para que isso ocorra é necessário o planejamento das disposições internas dos circuitos que compõe a máquina. De nada adiantaria, por exemplo, utilizar a micro-eletrônica para construir um único chip que integraria milhões de componentes constituintes de milhares de neurodos, uma vez que, queimado o chip, todo o cérebro eletrônico se perderia. Teríamos que pensar em chips contendo um único neurodo, ou pelo menos um pequeno grupo deles, e que ficassem espacialmente separados. Também as conexões entre esses chips não poderiam otimizar a comunicação através de um barramento único de dados, pois uma vez quebrado, interromperia as conexões entre todos os neurodos. Assim, esses chips teriam que se conectar de forma independente com seus pares e, ainda, deveriam possuir fonte de alimentação distinta, pelos mesmos motivos.

Essas técnicas para conseguir redundância de processamento, armazenamento e alimentação já são utilizadas em computadores digitais, principalmente servidores de arquivos ligados à internet, que devem se manter funcionando 24 horas por dia (ARAÚJO, 2001).

Há, porém, diferenças consideráveis no resultado de um mau funcionamento de tais máquinas relevante ao modelo cognitivo implementado. Nos modelos da IA, não há alteração nenhuma da execução algorítmica programada, e os processadores e memória redundantes assumem a posição dos defeituosos e assim permanecem até que estes sejam trocados por peças novas.

Já nos modelos da RNA, o defeito em um neurodo modifica as características de convergência da rede toda, que passa a responder com menos

precisão às informações já apreendidas, dependendo da abrangência do defeito. A substituição dos neurodos defeituosos por novos não re-estabelece o funcionamento algorítmico e representacional que antes estava estabelecido. Nos modelos de RNA supervisionados, há ainda um ponto que não foi devidamente esclarecido. Não são os próprios neurodos de saída que se comparam ao padrão desejado, mas um supervisor externo à rede (LYCAN, 1991, p.268). Há, portanto, um processador dedicado a acompanhar o algoritmo de aprendizado. O algoritmo de aprendizado é externo à rede. O algoritmo de cada neurodo simplesmente computa suas entradas resultando sua saída. Com isso, um defeito no processador que cuida da aprendizagem da rede acarretaria a estagnação dos pesos dos neurodos, não importando o quanto se estimule a entrada com um padrão a ser aprendido. Essas diferenças, porém, são devido aos modelos, e não devido à máquina.

Podemos concluir desta subseção que, no que concerne às máquinas, elas devem possuir redundância de suas partes críticas para permitir que os modelos utilizem essa propriedade para implementar da forma que lhes convier a tolerância à falhas. A tolerância à falhas, em relação aos modelos, uma vez implementada em uma máquina qualquer, pode se mostrar de duas formas: (1) Uma resposta precisa e rápida, porém rígida, para modelos que substituem completamente os módulos defeituosos. (2) Uma resposta flexível e degradê, porém de estabilização lenta, que distribui as tarefas do módulo defeituoso para outros módulos. Uma máquina de Turing executando um algoritmo em que o conteúdo de suas fitas (e/ou controles) é replicado para ser usado em substituição

à fita principal (e/ou controle), no caso desta (e/ou deste) apresentar defeito, apresentaria uma resposta do tipo (1). Já uma máquina de Turing executando um algoritmo em que suas fitas (e/ou controles) sejam independentes e se interconectem umas às outras para comunicação de resultados parciais (ou seja, um modelo de RNA), apresentaria uma resposta do tipo (2) no caso de uma falha em uma de suas fitas (e/ou controles). Parece-nos que os modelos de RNA demonstram mais afinidade com que de fato acontece com a maquinaria cerebral no caso de falhas, má-formação, amputações ou perda de massa encefálica, entretanto essas qualidades que podemos julgar serem mérito das RNA estão no nível algorítmico e representacional, e não, ao nosso ver, no nível implementacional.

4.3.6 - Redes Neurais Artificiais consideradas como funções

Para o estudo da computabilidade das RNA, é comum a referência à teoria dos autômatos para descrever o funcionamento de um modelo de RNA. Ludermir (1992) define uma nova gramática G_p , chamada *gramática regular com pesos* que é reconhecida por um *autômato probabilístico*. Essa nova gramática é usada pela autora para avaliar o poder computacional de um modelo de RNA chamado PLN (*Probabilistic Logic Node*).

A gramática G_p é definida (LUDERMIR, 1992, p.266) como uma quádrupla (A, N, I, P_p) , tal que:

- (1) A é o alfabeto, um conjunto de símbolos não-terminais.
- (2) N é o conjunto de símbolos terminais.

(3) I é o símbolo inicial, tal que $I \in N$.

(4) P_p é um conjunto finito de regras de produção com pesos, to tipo:

$\alpha_i \rightarrow \beta_{ij}(p_{ij})$, onde

$\alpha \in N$

β_{ij} é da forma wS_j , com $S_j \in N$ e w é um *padrão* de saída da rede.

p_{ij} é um peso associado com a (frequência de) aplicação da respectiva regra de produção.

A linguagem $L(G_p)$ gerada por essa gramática, chamada *linguagem regular com pesos* (LRP) pode ser comparada com a hierarquia de Chomsky no seguinte diagrama:

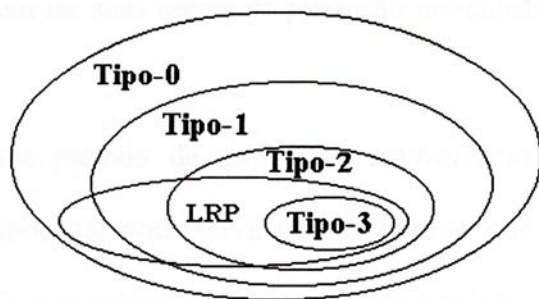


Figura 22: A gramática regular com pesos na hierarquia de Chomsky (LUDERMIR, 1992, p.268)

Notamos que o autômato probabilístico reconhece as linguagens regulares, ou linguagens tipo-3, além de reconhecer algumas linguagens tipo-2, tipo-1 e mesmo tipo-0. Entretanto, esse autômato não reconhece nenhuma linguagem além daquelas reconhecidas pelas máquinas de Turing, ou seja, não vai além das linguagens tipo-0, ou linguagens recursivas.

Ludermir (1992, p.263) provou dois teoremas para mostrar a equivalência entre o modelo de RNA tipo PLN e os autômatos probabilísticos:

Teorema (1): Seja $L(G_p)$ a linguagem gerada por uma gramática regular com pesos, então existe uma RNA tipo PLN que reconhece $L(G_p)$.

Teorema (2): Seja $L(PLN)$ um conjunto de padrões que é reconhecido por uma RNA tipo PLN, então esse conjunto pode ser gerado por uma gramática regular com pesos.

É interessante notar que, mesmo que uma RNA do tipo PLN tenha poder computacional igual ao de um autômato probabilístico, as redes PLN são treinadas para reconhecer um conjunto qualquer, enquanto os autômatos probabilísticos devem ter suas regras de produção pré-construídas (LUDERMIR, 1992, p.285).

Esse método de mostrar a *equivalência* entre uma RNA e alguma gramática tipo-i não pode servir para comprovar que um modelo de RNA é mais poderoso que uma máquina de Turing, pois o máximo que pode conseguir é comprovar a equivalência entre o modelo de RNA e a gramática tipo-0, o que significaria que o modelo é Turing-computável. Por outro lado, não podemos criar uma gramática G_L que seja mais livre que a gramática tipo-0 para depois comprovarmos que um modelo de RNA qualquer é equivalente a G_L . A gramática tipo-0, como o próprio nome diz, *gramática irrestrita*, aceita qualquer tipo de regras de produção. Não há como torná-la mais livre. É por esse motivo que o matemático Manna (1974, p.1) chama as máquinas de Turing de “*the ‘most general possible’ computing device*”²¹. Sabemos, entretanto, que existem muitas

funções não-Turing-computáveis. O que não existem são dispositivos computacionais capazes de calculá-las de modo efetivo.

Como podemos compreender os modelos de RNA como funções? Toscamente falando, uma função é um procedimento que recebe uma entrada e retorna uma saída.²² Essa noção pode se adequar ao caso das redes MP, das redes de Hopfield e das redes de Kohonen, uma vez que é isso que essas redes fazem: recebem um padrão de entrada e computam um padrão de saída. A noção parece se tornar mais complicada para o caso das redes MLP (incluindo o perceptron de uma camada), pois além do padrão de entrada, fornecemos também a saída desejada, que será utilizada no treinamento da rede. Se diferenciarmos a *saída desejada* da *saída obtida*, e considerarmos a saída desejada como apenas mais um parâmetro de entrada na função, talvez possamos conceber tais redes como funções. Intencionamos que a rede MLP treine, entregando-lhe como entrada tanto o padrão quanto a saída desejada, e sua resposta, a saída obtida, seja melhorada até, eventualmente, ser idêntica à saída desejada. Desse novo ponto de vista, as redes MLP também se enquadram em uma função que recebe como entrada uma dupla: o *padrão* e a *saída desejada*, e nos dá como resultado a *saída obtida*. Se pudermos usar esse artifício, fica a questão: qual a relação entre a *saída obtida* e a *saída desejada*? Voltaremos a esse ponto alguns parágrafos adiante.

Esses modelos de rede, MP, perceptron, MLP, RH e RK, têm uma importante propriedade: dada uma entrada na rede (um padrão e se necessário uma saída desejada), essas redes retornam *sempre* uma saída qualquer. Portanto podemos dizer que essas redes são *definidas* para todas as entradas, o

que Rumelhart e McClelland (1989, p.29) chamam de *default assignment*²³. Conforme tivemos a oportunidade de ver na seção 2.5.2, uma função desse tipo é classificada como uma *função total n-ária*. De acordo com Manna (1974, p.53), uma máquina de Turing só pode ser utilizada como um *algoritmo* para funções totais, pois a máquina não pode entrar em ciclo para o caso de um valor indefinido na função. Perguntamos: essas funções totais n-árias são Turing-computáveis? Se forem, então existe um algoritmo que as compute. Mas, sabemos também, da seção 2.5.3, que vale a recíproca: dado um algoritmo para calcular uma função total, essa função é Turing-computável. Não precisamos de nada além dos algoritmos dados no capítulo 3 para afirmarmos que essas redes aqui apresentadas, especificamente, as redes MP, o perceptron, as MLP, a RH e a RK representam funções Turing-computáveis.

Podemos restringir ainda mais o alcance dessas redes caso os seus algoritmos possam ser programados em linguagens como C ou PASCAL. Brandão²⁴ indica que tais linguagens são construídas utilizando *gramáticas livre de contexto*, ou seja, são linguagens tipo-2, logo um algoritmo programado com tais linguagens reconheceria apenas, e no máximo, o conjunto de funções que se inserem nas geradas pelas gramáticas tipo-2.

No caso das redes MLP simuladas em máquinas de Turing, seu algoritmo retorna como resposta a *saída obtida*, que, por inferência do raciocínio desenvolvido acima seria a resposta de uma função Turing-computável (FTC). Entretanto, a *saída desejada*, dada pelo projetista como entrada na rede, junto com o padrão, não precisa, necessariamente, ser a saída de uma FTC. Suponhamos,

portanto, que a *saída desejada* seja a de uma FÑTC, como, por exemplo, uma resposta ao problema de decisão de Hilbert, que vimos não ser passível de ser descrito por um algoritmo. Sabemos, por uma limitação formal da própria máquina de Turing, que, no exemplo acima, a *saída obtida* nunca será igual à *saída desejada*, por mais iterações que se permita o algoritmo fazer para aproximar o resultado (pois se assim for, teremos uma máquina de Turing que computa uma função não Turing-computável, o que é um absurdo). Até que ponto a saída obtida por um processo algorítmico pode se aproximar de uma função não Turing-computável (FÑTC)? Imaginemos, em um exemplo hipotético de uma rede MLP, que a *saída obtida* seja idêntica à *saída desejada* em 95% dos casos. O que isso significaria? Ao nosso ver, significaria que os pesos do modelo de RNA convergiram de tal forma que produziram como resposta ao problema dado uma FTC que coincide, ou interpola bem, 95% dos casos da FÑTC desejada. Ora, esse método não parece estar buscando meios de efetivamente calcular uma FÑTC, mas parece estar buscando uma FTC que possa *aproximar* bem o problema proposto.

Ludermir (1992) indica um novo método para avaliar a capacidade de uma máquina reconhecer uma linguagem, que não é mais do tipo tudo-ou-nada: “*Patterns will be recognized by the network if the probability associated with them is greater than the threshold*” (LUDERMIR, 1992, p.271)²⁵. No nosso exemplo acima, se uma MLP implementada em uma máquina de Turing aproxima uma FÑTC com 95% de acerto poderíamos concluir que uma máquina de Turing executando um algoritmo de RNA tem seu poder computacional

aumentado para além das FTC? Por outro lado, não temos mais a certeza algorítmica de um processo dedutivo que dá respostas para 100% dos casos, com acerto – o algoritmo deixa de ser visto como o procedimento efetivo finito que dá como resposta a *saída obtida* e passa a ser um procedimento não-efetivo que aproxima uma *saída desejada*. Temos ainda um algoritmo, sim, mas que implementa um processo indutivo, que transforma a máquina de Turing em um aproximador de funções, e se ‘ganhamos’ poder computacional, também ganhamos um completo desacoplamento entre algoritmo e procedimento efetivo (FLORIDI, 2001). O desacoplamento acontece quando o algoritmo, que deveria ser um procedimento para efetivamente calcular o resultado de uma função, passa a calcular tal resultado através de aproximações que permitem erros: o algoritmo não mais calcula a *saída desejada*, porém calcula uma *saída obtida* que é uma aproximação daquela.

De qualquer forma, o conjunto das pequeninas funções Turing-computáveis representativas de cada neurodo, e suas relações através de pesos entre suas entradas e saídas não deixam de ser uma única grande função composta, que representaria a RNA como um todo. Manna (1974, p. 51) indica que a composição de funções Turing-computáveis resulta em uma função Turing-computável. Nesse sentido, uma RNA, ao nosso ver, é uma função Turing-computável que aproxima (interpola ou extrapola) outras funções quaisquer, sejam Turing-computáveis ou não, determinadas pelas combinações possíveis de pesos nas conexões. Ludermir (1992, p.262) apresenta essa propriedade dos modelos de RNA:

The performance of a learning algorithm is measured by looking at the structure achieved through such learning processes and comparing the desired function f to the function computed by the network acting as a classical automaton. It is important to characterize the functions which can be computed by the network in this fixed structure, since if there is no configuration which allows the computation of f , then the network cannot learn to compute f . (LUDERMIR, 1992, P.262)²⁶

Nesta citação a autora enfatiza que há uma limitação no aprendizado de um modelo de RNA, caso não haja uma combinação de pesos que bem aproxime a função desejada. Se as FNTC possuem boas aproximações dentre as FTC (no nosso exemplo, com acerto de 95%), e se os modelos de RNA seriam ideais para se implementar essas funções, permanecem questões em aberto.

Nesta seção 4.3, analisamos as características que são invocadas pelos conexionistas em vários momentos, para a defesa do modelo de RNA e de uma suposta máquina conexionista. Nossa análise não se restringiu ao plano algorítmico e representacional, ou dos modelos, mas também analisamos com cuidado o plano implementacional. Resumindo os temas abrangidos, o processamento numérico de informação, o processamento paralelo e distribuído e a tolerância à falhas não tiram as máquinas conexionistas do patamar de máquinas passíveis de serem simuladas por máquinas de Turing. O maior empecilho parece vir do processamento analógico, que foi dividido em duas partes: a precisão das quantidades representadas e a característica de continuidade no tempo. Quanto à precisão das quantidades representadas por símbolos, uma vez que se tenha uma precisão finita é possível conceber uma máquina de Turing com memória suficiente para armazenar tal valor. Sobre a continuidade no tempo, a não ser que

os modelos sejam modificados para tornar tal propriedade imprescindível, do modo como são hoje concebidos, tornam-se discretizáveis, passíveis de serem simulados por máquinas rápidas o suficiente para realizarem os cálculos dentro de fatias de tempo toleráveis. Não vemos, pois, diferença alguma que impeça a realização de uma máquina connexionista em uma máquina de Turing, e neste sentido concordamos com os planos de Haselager (figura 21) de que ambas são instâncias de uma máquina de Turing universal.

Os modelos da IA e da RNA, entretanto, apresentam características muito distintas no nível representacional e algorítmico. Avaliamos que o processamento não-simbólico que seria realizado pelas RNA não está relacionado aos símbolos representativos das quantidades entradas nos neurodos, mas à criação de símbolos complexos na interação da rede como um todo. A IA também cria símbolos mais complexos partindo de símbolos mais simples, entretanto esses símbolos mais simples já vêm carregados com uma gama muito maior de semântica e de significado, impostas pelo projetista. Também avaliamos que o processamento analógico, quando se trata de modelos realistas do cérebro, poderia impedir esses modelos de serem implementados em máquinas de Turing, tanto em relação à precisão infinita de valores como em relação ao tempo contínuo de atuação do neurodo. Por outro lado, as simulações que têm sido feitas nas MVN mostram que a precisão necessária para bons resultados pode ser alcançada através de técnicas de discretização do tempo e truncamento da precisão. A característica de competição e cooperação imposta pelos modelos de RNA quando utilizam processamento paralelo (real ou simulado) parecem



propiciar um ambiente adequado para o surgimento de propriedades emergentes, de generalização, de aprendizagem. O processamento e o armazenamento distribuído, apesar de não extrapolarem o escopo das funções Turing-computáveis, permitem uma maior eficiência na resolução de problemas complexos. Vimos também que a tolerância à falhas nos modelos de RNA apresenta uma característica notável de degradação suave, tipicamente encontrada em seres vivos.

Os indícios de equivalência no nível implementacional, e as vantagens apresentadas pelos modelos de RNA no nível algorítmico e representacional nos levam a concluir que os modelos de RNA não são nada mais que algoritmos de máquinas de Turing, porém mais eficientes e mais condizentes com a solução procurada para o problema de conhecer o funcionamento da mente inteligente, no nível computacional. Nos parece interessante agora questionarmos a possibilidade de auto-organização nesses modelos de RNA, e na própria máquina de Turing.

4.4 - Notas

¹ O termo sub-simbólico foi criado por Smolensky (1988) para caracterizar quantidades (de energia) que ao serem processadas numa rede neural poderiam dar lugar a estruturas simbólicas com as quais o sistema cognitivo trabalha.

² CAUDILL & BUTLER, 1992, p.3.

³ (Grego: αὐτόματον. Latim: *Casus*. Inglês: *Chance*. Francês: *Hasard*. Alemão: *Zufall*. Italiano: *Caso*. Holandês: *Toeval*). O acaso aqui é visto por nós de acordo com a definição de Cournot, na qual o evento imprevisível pode ser entendido como o cruzamento de linhas causais independentes. Esta definição não parece incluir verdadeiramente a imprevisibilidade, uma vez que teríamos apenas a complexificação da previsibilidade que estaria agora vinculada a diversas linhas causais, e não a apenas uma, ou seja, não se nega o funcionamento da própria causalidade. Peirce (talvez influenciado por Hume) compreende o acaso como probabilidade, enfraquecendo a doutrina de que tudo no mundo ocorre segundo leis causais. (ABBAGNANO, 1998, p.11-13)

⁴ A rede de McCulloch e Pitts, como Kleene (1956) demonstrou, é equivalente a um autômato finito e, portanto, mais limitada que a máquina de Turing.

⁵ Não devemos confundir o termo 'computacional' usado por Marr para descrever o primeiro nível do processo de modelagem com a noção de computabilidade ou cálculo efetivo explanada no capítulo 2. O nível computacional refere-se ao entendimento e planejamento do objetivo da computação. A computação, por sua vez, é a realização inequívoca de procedimentos implementados em máquinas de Turing.

⁶ Tradução nossa: É sabido que uma máquina de Turing simples, com uma única fita de armazenamento, na qual modificações específicas são feitas por uma única cabeça de leitura e gravação, é equivalente em poder computacional a configurações alternativas com múltiplas cabeças, múltiplas fitas ou até mesmo sistemas *multidimensionais*.

Grifo nosso.



⁷ Tradução nossa: Ao passo que Broadbent apreende somente dois níveis da teoria, o computacional e o implementacional, Marr na verdade propôs três níveis, o computacional, o algorítmico e o implementacional.

⁸ Tradução nossa: Como psicólogos, estamos engajados em uma elucidação do nível representacional e algorítmico.

⁹ Ibidem.

¹⁰ HASELAGER, W. (Marília, UNESP). Comunicação pessoal, 30 de maio de 2001.

¹¹ Tradução nossa: Apelos casuais à tese de Turing podem ser suficientes para estabelecer equivalência no nível computacional, mas eles são geralmente de pouco uso para nós na psicologia, precisamente porque nós estamos preocupados com formas mais fortes de equivalência de nossos modelos com os processos psicológicos.

¹² HASELAGER, W. (Marília, UNESP). Comunicação pessoal, 19 de dezembro de 2001.

¹³ Tradução nossa: Em particular, e ignorando questões de treinamento e praticidade de implementação, gostaríamos de saber se todos os problemas que podem ser solucionados por um computador digital são também solúveis – em princípio – por uma rede (neural artificial). [...] De fato, Jordan Pollack (1987) mostrou que um certo modelo de rede recorrente [...] é universal.

O trabalho que Siegelmann e Sontag citam: POLLACK, J. *On connectionist models of natural language processing*. 1987. Tese (Ph. D.), Departamento de Ciência da Computação, Universidade de Illinois, Illinois.

¹⁴ Tradução nossa: É óbvio que se pode simular uma rede de processadores com uma máquina de Turing, já que nós tomamos todos os valores como sendo racionais. Nossa principal observação é que, inversamente, qualquer função computável por uma máquina de Turing pode ser computada por uma rede de processadores.

Grifos nossos, no original e na tradução.

¹⁵ Inspirado em Franklin (1995, p.161), Broadbent (1985, p.189), Rumelhart e McClelland (1985, p.194).

¹⁶ Tradução nossa: Mas números são de fato símbolos para quantidades; e qualquer função computável sobre números pode ser computada por processos simbólicos. De fato, computadores digitais de propósito geral têm realizado tanto processamento simbólico como numérico desde que foram inventados.

¹⁷ Tradução nossa: Antes de iniciarmos uma simulação, valores devem ser dados a todos os parâmetros e a todas as saídas. Cada valor tem sempre o comprimento de uma *única palavra de computador*. A interpretação desta palavra depende de como os *métodos* a utilizam.

Os grifos, no original e na tradução, são nossos.

¹⁸ Tradução nossa: Sistemas digitais podem ser, e frequentemente são, simulados em computadores digitais no nível desejado de precisão.

¹⁹ A tabela ASC-II, *American Standard Code 2*.

²⁰ Tradução nossa: E, como ressaltado por Hanson e Burr (1990), a codificação distribuída nela ou dela mesma, não oferece capacidades representacionais que não são realizáveis usando uma codificação não distribuída.

O trabalho que Honavar cita: HANSON, S. J., BURR, D. What connectionist models learn: learning and representation in connectionist networks. *Behavior and Brain Sciences*, 1990. v.13, p.1-54.

²¹ Tradução nossa: O dispositivo de computação 'mais geral possível'.

²² FEITOSA, H. A. (Marília, UNESP). Comunicação pessoal, 4 jul. 2001.

²³ Tradução nossa: Designação de praxe. Também utilizado na IA, é o tipo de tomada de decisão feita automaticamente no caso de falta de informações mais precisas. Podemos exemplificar a 'designação de praxe' com esta ordem do chefe à sua secretária: – Caso venha o primeiro-ministro Tony Blair, sirva-lhe chá. Caso visite-nos o ministro da fazenda Pedro Malan, sirva-lhe café. Caso seja outra pessoa, dê-lhe um copo d'água.

²⁴ BRANDÃO, M. A. (São José do Rio Preto, UNESP). Comunicação pessoal, 1998.

Capítulo 5 - Sistemas Auto-organizáveis

²⁵ Tradução nossa: Padrões serão reconhecidos por uma rede (neural artificial) se a probabilidade associada a eles forem maiores que o limiar (da RNA).

²⁶ Tradução nossa: A performance de um algoritmo de aprendizado é medida olhando-se a estrutura atingida pelo tal processo de aprendizagem e comparando a função desejada f com a função computada pela rede (neural artificial) atuando como um autômato clássico. É importante caracterizar as funções que podem ser computadas por uma rede (neural artificial) nessa estrutura fixa, uma vez que se não houver configuração que permita a computação de f , então a rede (neural artificial) não pode aprender a computar f .

Por 'configuração' devemos entender uma combinação possível dos pesos das conexões da RNA em questão.

Capítulo 5 - Sistemas Auto-organizáveis

5.1 - Apresentação

Neste capítulo pretendemos elaborar uma discussão sobre a auto-organização nos planos representacional e algorítmico que as RNA parecem apresentar, com o intuito de transpor características da teoria de auto-organização para o plano formal das máquinas de Turing. Na seção 5.2, introduziremos os conceitos da sistêmica, trataremos da teoria da auto-organização, principalmente sobre os conceitos delineados por Ashby (1962), Debrun (1996a, 1996b, 1997) e Atlan (1992, 1998, 2001). Na seção 5.3, discutiremos alguns pontos específicos necessários em um sistema para que este possa apresentar auto-organização. Na seção 5.4, tentaremos verificar até que ponto os conceitos de auto-organização e de propriedade emergente são modelados pela RNA. Na seção 5.5, discutiremos como a auto-organização dos modelos de RNA, se houver, deve ser inserida nos moldes das máquinas de Turing, e como o alcance da própria máquina de Turing poderia ser revisto.

5.2 - Sistêmica e Auto-organização

A sistêmica, nome que se dá à ciência que estuda os sistemas, é o ponto inicial para se entender a noção de auto-organização, pois os processos

auto-organizáveis ocorrem nas interações dos elementos de sistemas. É forçoso então conhecer o que vem a ser um sistema.

Um sistema é um conjunto não vazio de elementos que possuem entre si alguma relação, durante um intervalo de tempo suficiente para lhes garantir uma unidade e uma identidade (BRESCIANI FILHO & D'OTTAVIANO, 2000, p.284-5; PESSOA JÚNIOR, 2000, p.130). A abstração e o recorte desse conjunto de elementos e de suas relações no universo que os contém, devem ser feitas por um sujeito, ainda que disposicional¹, que lhe confere uma identidade, e pode identificar sua funcionalidade. As relações podem já existir, e serem empiricamente comprovadas ou podem ser meras hipóteses a serem testadas. Neste último caso, uma vez comprovada a inexistência das relações levantadas como hipótese, comprova-se também que o recorte feito, de elementos do universo, não se caracterizou como um sistema.

Os elementos, ou partes, do sistema podem ser de dois tipos: os elementos internos e os elementos de fronteira. São considerados, por Bresciani Filho & D'Ottaviano, elementos internos ao sistema aqueles que mantêm alguma relação apenas com outros elementos do sistema, não a possuindo com elementos do meio-ambiente. Os elementos de fronteira, por sua vez, são aqueles que mantêm alguma relação tanto com elementos internos como com elementos do meio-ambiente. Cada elemento, individualmente, por sua vez, pode ser constituído por partes que se relacionam, e por isso são considerados, eles próprios, subsistemas, *ad infinitum*. Dessa definição, na qual um elemento pertence ao sistema se e somente se possui alguma relação com outros elementos do sistema,

podemos concluir que não existem elementos isolados no sistema, caso em que seriam considerados elementos externos ao sistema.

Cada elemento constitutivo do sistema possui suas próprias qualidades, propriedades e afinidades, e cria com os outros elementos relações de diferentes tipos. É justamente esta variedade, segundo Bresciani Filho e D'Ottaviano que permite que o todo não seja a simples soma das partes, mas seja “mais, ou menos, do que a soma das partes” (BRESCIANI FILHO & D'OTTAVIANO, 2000, p.286). Esta posição também é enfatizada por Debrun, que denomina o *motor* da auto-organização a própria junção de elementos realmente distintos, soltos e ativos (DEBRUN, 1996a). Esse é um dos principais pontos para se pensar a questão da *emergência*, que será vista adiante.

O meio-ambiente é o complemento da intersecção entre os elementos do sistema e os do universo, em outras palavras, é tudo aquilo que se convencionou não pertencer ao sistema e ainda assim pertencer ao mesmo universo que o sistema pertence, o “resto”. Um sistema é dito fechado se não interage com seu meio ambiente, ou seja, não possui elementos de fronteira. Caso contrário, teremos um sistema dito aberto.

De acordo com a Teoria da Auto-Organização (TAO), sistemas fechados não apresentam, em princípio, qualquer possibilidade de auto-organização. Debrun (1996a) e Ashby (1962), entre outros, consideram dois tipos de auto-organização: a auto-organização primária e a auto-organização secundária. Resumidamente, utilizando-nos das noções de sistêmica dadas acima, a auto-organização primária ocorre quando elementos distintos (DEBRUN,

1996b), ou disjuntos (ASHBY, 1962), se encontram e passam a constituir uma forma (DEBRUN, 1996b), um sistema, ou passam a se condicionar mutuamente (ASHBY, 1962).

A auto-organização secundária ocorreria em um sistema já com algum grau de organização, e a sua principal característica seria o *aprendizado* pelo sistema, de uma nova estrutura que refletiria um maior grau de organização que o apresentado em seu estado inicial. Utilizaremos a noção de boa e má organização, proposta por Ashby (1962) para definir a auto-organização secundária como sendo o processo pelo qual um sistema consegue, pelos seus próprios meios e interação com o ambiente, sair de uma organização precária para uma organização mais desenvolvida.

Debrun (1996a) apela para o senso comum para explicar o termo “auto-organização” como sendo uma criação de organização (no caso da AO primária) ou o aumento de organização (no caso da AO secundária), num processo “auto”. É possível reconhecer três significados para o termo “auto”, segundo Debrun (1996a, p.6):

- (a) Autonomia: o processo tem autonomia e possibilidade para criar seu próprio desenvolvimento, que ultrapassa as condições de partida e restrições iniciais.
- (b) Auto-referência: o processo consegue se desenvolver num trabalho de si sobre si. Esse é o principal ponto atacado por Ashby (1962), uma vez que ele mostra a impossibilidade lógico-matemática de uma máquina ou função alterar-se a si própria. Atlan (1992, p.39) argumenta que essa impossibilidade só ocorre em sistemas fechados, nos quais não há interação com o meio, incluindo, como já

mencionamos, uma característica essencial à auto-organização, qual seja, que ela ocorre somente em sistemas abertos.

(c) Identidade: finalmente, o termo “auto” pode, segundo Debrun (1996a), ser entendido como a tendência do processo de criar um Todo organizado, e assim constituir uma forma, assumindo uma identidade.

Assim como o termo “auto”, o termo “organização” possui muitos sentidos, que podem diferenciar substancialmente o que se considera auto-organização. Normalmente é aceito como definição que, para um sistema se auto-organizar, este precisa, entre outras coisas, aumentar o seu grau de organização. Pessoa Júnior (1996, p.135) cita as medidas sistêmicas de organização mais comuns na literatura, das quais trazemos as duas que são também consideradas por Ashby (1962): (a) organização enquanto *condicionalidade*, (b) organização enquanto *adaptabilidade*. A primeira medida da organização, enquanto condicionalidade, é uma medida objetiva, definida pela capacidade de cada elemento constitutivo do sistema influenciar os outros, restringindo assim as possibilidades de resposta num espaço de estados (ASHBY, 1962). A segunda medida da organização, enquanto adaptabilidade, é considerada por Ashby (1962) como relativa ao contexto, e não pode ser definida em termos absolutos. Ashby (1962) chama de “boa organização” aquela organização que se adaptou ao seu contexto, ao seu meio, o que traz consigo, como observa Pessoa Júnior (1996), as noções de eficiência e sobrevivência. Bresciani Filho e D’Ottaviano (2000, p.300) consideram a característica de adaptação de um sistema como a transição de um estado de desequilíbrio para outro estado de equilíbrio que configura a solução

encontrada pelo sistema para manter sua existência. Dependendo da medida utilizada para conhecer o grau de organização de um sistema teremos um resultado diferente, de modo que um sistema que aparenta ser auto-organizado com uma medida pode não o ser com outra, como nota Pessoa Júnior (1996).

Nos dois exemplos de medidas sistêmicas dados acima, não é nada óbvio que um sistema que seja altamente condicionado, logo organizado pela primeira medida, tenha uma boa adaptação ao meio em que se encontre, logo seja organizado pela segunda medida. Essa foi uma das dificuldades que Ashby (1962) encontrou na definição de boa e má organização, que tentou superar com a seguinte asserção:

... the organization is judged "good" if and only if it acts so as to keep an assigned set of variables, the "essential" variables, within assigned limits. [...] an organization is "good" if it makes the system stable around an assigned equilibrium. (ASHBY, 1962, p.263)²

Não há uma organização "boa" em sentido absoluto, pois o equilíbrio a ser alcançado pelo sistema depende de sua interação com o meio, o que implica que diferentes meios geram diferentes estados de equilíbrio, e o sistema auto-organizado deve ser capaz de se modificar em busca desse estado de equilíbrio *desconhecido*.

Com esta introdução estamos aptos a nos aprofundar em algumas condições que julgamos necessárias para que um processo em desenvolvimento seja considerado auto-organizado, nas visões de Debrun (1996a, 1996b), Ashby (1962) e Atlan (1992, 1998, 2001). Uma vez citadas essas

condições, poderemos retomá-las analisando os modelos de RNA e a máquina de Turing.

5.3 - Condições para a ocorrência da auto-organização

Existem algumas características que são apontadas na literatura como sendo típicas da auto-organização, e sua ocorrência determinaria se um sistema em estudo poderia ser classificado como auto-organizado. Também se poderia falar em graus de auto-organização, ponto de vista no qual a auto-organização não é uma característica do tipo tudo-ou-nada, mas tem seu grau aumentado ou diminuído por influência de diversos fatores, como por exemplo, a complexidade final da forma auto-organizada comparada com sua estrutura inicial (DEBRUN, 1996a, p.5).

Segundo Debrun (1996b, p.30), podemos classificar os sistemas, de acordo com seu grau de auto-organização, em cinco condições:

- a) *Não há auto-organização*: aqui se encontrariam os mecanismos rígidos, sem nenhuma possibilidade de mudanças estruturais ou dinâmicas, como, por exemplo, os relógios, os brinquedos que imitam o comportamento de animais, como os cãesinhos mecânicos mais simples, etc.
- b) *Grau zero* de auto-organização: seriam os casos em que existem possibilidades além das previsíveis, mas o sistema permanece mais ou menos seguindo uma lei. Não há mudança no comportamento do sistema, apesar de haver flutuações: “uma correção ‘quase mecânica’ desses desvios permite a reprodução ‘quase certa’ da situação anterior” (DEBRUN, 1996b, p.31).

c) *Grau um* de auto-organização: inclui-se aqui o auto-funcionamento cibernético, que é capaz de responder flexivelmente à evolução dos sistemas, mantendo sua busca por um objetivo. O objetivo, entretanto, não pode ser remodelado pelo próprio sistema. Atlan (1998, p.15) chama esta situação de auto-organização fraca, e não considera o grau anterior colocado por Debrun.

d) *Grau dois* de auto-organização: teríamos aqui sistemas complexos, que podem responder flexivelmente às situações, podem mudar seus próprios objetivos, mas não podem remodelar seu próprio ser: existe uma identidade já 'cristalizada'. Acreditamos que Debrun coloca aqui um tipo de auto-organização secundária com menor grau de liberdade. Atlan (1998) nomeia esta situação como auto-organização forte, capaz mesmo de ser programada em sistemas computacionais. Este tipo de grau de auto-organização é reconhecido por Ashby (1962) como uma propriedade natural de todos os sistemas dinâmicos, sendo o grau máximo que, segundo o autor, uma máquina poderia alcançar.

e) *Grau três* de auto-organização: Debrun chama este grau de 'auto-organização propriamente dita'. O sistema com este grau de AO é capaz de "ser a própria gênese de seu ser" (DEBRUN, 1996b, p.31), sem, contudo seguir uma lei ou algoritmo de criação; ele se cria pela sua própria sedimentação. Temos aqui a AO primária. Também podemos ter AO secundária neste grau, caso em que um sistema já existente consegue se redefinir, pelo menos parcialmente, por aprendizagem ou por agregação, aumentando sua complexidade. Este grau máximo de auto-organização é chamado por Atlan (1998, p.24) de 'verdadeira auto-organização' (*truly self-organizing*). Ashby (1962) mostra uma

impossibilidade lógico-matemática na suposição de uma função que conseguisse alterar sua própria estrutura, e fosse, portanto, capaz de assumir este grau de auto-organização, que considera inatingível por qualquer máquina.

Para que um processo atinja o máximo grau de auto-organização, os elementos que o constituem ou o constituirão, sua interação, as condições iniciais e de desenvolvimento ulterior devem seguir alguns princípios, ou pré-requisitos. Extraímos esses princípios principalmente de Debrun (1996a, 1996b), mas também de Atlan (1992, 1998) e Ashby (1962), que são:

i) O processo deve ter um começo ou ponto de amarração, no qual os elementos que constituem o sistema possam se encontrar. Esse início deve representar um corte na realidade, por obra do acaso ou de uma determinação exterior, que traga ao conjunto de elementos em interação uma nova condição de trabalho inserida em um contexto distinto do anterior ao encontro. No começo do desenrolar do processo, as condições iniciais impostas por fatores externos têm um papel muito forte, que será absorvido pelo próprio processo de auto-organização. Se as condições iniciais forem muito rígidas, impedindo o processo de tomar rumos ao sabor da sua dinâmica, teremos pouca ou nenhuma auto-organização (DEBRUN, 1996a, p.8). Os fatores que geraram as condições iniciais, constituem o que Debrun (1996a, p.14) chama de “condições das condições de partida”, as quais devem cessar sua influência e serem esquecidas, permitindo assim a autonomia do processo.

ii) No processo AO devem tomar parte elementos realmente distintos, soltos e equípotentes. A necessidade de termos elementos “realmente”

distintos reside na importância da interação não *viciada* entre esses elementos. Um conjunto de elementos que já possuem uma história em comum tem diminuído o grau de liberdade de suas interações, até o ponto em que, no limite, se avaliarmos um processo no qual tomou parte elementos deterministicamente condicionados, não teremos auto-organização. Tais elementos distintos podem operar, no processo de AO, como elementos ativos, que expõem suas particularidades, ou elementos de união, fazendo o papel de ponte de ligação. A *existência e interação* entre elementos “realmente distintos” são, segundo Debrun (1996a, p.9), o próprio *motor* da auto-organização: “A simples reunião de elementos realmente distintos é uma novidade, quase uma criação” (DEBRUN, 1996a, p.14). Só é possível conceber um *encontro* (ponto de amarração) entre elementos *realmente distintos*, quando não temos elementos idênticos, redundantes ou que já se condicionam de algum modo (DEBRUN, 1996b, p.33). É nesse sentido, permissivo da possibilidade de encontro, que se deve entender os elementos como ‘soltos’ antes do início do processo de auto-organização. Já a equipotência entre os elementos se deve à impossibilidade de um processo se auto-organizar quando tem entre seus elementos algum dominante, que torna tendencioso um desenrolar que deveria ser natural e espontâneo. As características dos elementos, de serem distintos, soltos e equipotentes, são características que podem possuir uma gradação, não sendo, portanto, do tipo tudo-ou-nada, além de que, sua existência influencia diretamente o aumento do grau de auto-organização.

iii) O processo consome tempo. Existe, na visão de Debrun (1996b, p. 53) uma ligação intrínseca da auto-organização com a *temporalidade*.



O processo auto-organizado se desenvolve na dinâmica de interação entre elementos no tempo, o que significa dizer que é o próprio desenrolar do processo que determina como ele irá desembocar, se irá se desfazer ou se irá estabilizar:

Ela [a temporalidade] não poderia ser maior ou menor do que foi. Se fosse maior ou menor tratar-se-ia de *outra* auto-organização; ou de um processo 'não auto-organizado' (DEBRUN, 1996b, p.53).

Na auto-organização, o tempo, direta ou indiretamente, traz novidade: os elementos do sistema estão dispostos tanto espacial quanto temporalmente, cada um tendo seu próprio tempo, seu cansaço, sua vida útil, seu ciclo particular ou relógio biológico. A inclusão do tempo real nas condições necessárias para a ocorrência da auto-organização traz uma outra característica marcante, segundo Debrun (1996b):

iv) A auto-organização não é *reprodutível*:

Com efeito: para ser reprodutível, ela teria que obedecer a uma 'lei de construção' (programa, algoritmo). Mas então essa reprodução, ditada pela lei de construção, não seria mais auto-organizada e sim hetero-organizada. (DEBRUN, 1996b, p.53)

Debrun, chega a sugerir uma *suspensão parcial e provisória do determinismo*, que estaria ligada à ampla possibilidade de interação entre elementos distintos soltos num lapso de tempo caótico, mas que não seria irrestrita, ou seja, "nem tudo pode ocorrer" (DEBRUN, 1996b, p.54).

Em contraste, na visão de Atlan (1998, p.24), não há essa suspensão do determinismo na verdadeira auto-organização, mas tal efeito pode

ser conseguido por esta seguir um programa³ com *sofisticação* infinita. Atlan (1998, p.13) define a sofisticação como a medida da complexidade do *significado* de um programa, para complementar a medida clássica de complexidade de algoritmos, que não leva em conta o significado (IUSEM, 1987). O que Atlan deseja com a sofisticação é, na verdade, não diferenciar programas de complexidades (clássicas) diferentes, mas que resolvem o mesmo problema: esses programas teriam a mesma medida de sofisticação. Um programa muito extenso, mas que só faz imprimir uma cadeia de símbolos aleatória, por exemplo, teria sofisticação próxima de zero (ATLAN, 1998, p.13). Assim, um programa com sofisticação infinita é um programa que codifica infinitos processos e não pode ser simplificado para um outro programa menor. Sua execução seqüencial traz à tona, a cada momento, uma parte nova do programa, que nunca fora antes executada, o que tornaria o processo imprevisível apenas epistemologicamente, mas ontologicamente previsível e determinístico.

v) Não há objetivo iminente no processo de auto-organização. Os elementos constituintes do processo podem apresentar, eventualmente, finalidades locais, desejos ou ambições, mas o desenrolar do processo não é resultado apenas dessas tendências, mas sim de todo o conjunto de interação entre os elementos e o ambiente. Segundo Debrun (1996b), quando, e se, o sistema chega a criar funções, estas são criadas e desenvolvidas em benefício do próprio processo, e não impostas como um comportamento a ser atingido. É interessante apontar que Atlan e Debrun, apesar de concordarem que o processo de AO não permite que se desvende seus objetivos antes da própria execução do processo,

eles discordam em outro ponto: para Atlan (1998, p.21) o processo pode ser totalmente determinístico, o que implica na pré-existência do resultado de sua execução, mas tal objetivo pré-existente não é dado de antemão e só pode ser descoberto após a própria simulação do processo: novamente temos aqui uma distinção de pontos de vista no plano epistemológico e ontológico.

vi) A auto-organização deve ser encontrada, conforme já ressaltamos, na dinâmica de interação dos elementos. Esse é, segundo Debrun (1996a, p.9, 1996b, p.27), o motor da auto-organização: elementos distintos e ativos, quando postos em contato, caso surja uma afinidade unificadora desses elementos em um projeto comum ou sistema, geram na sua interação um novo produto, auto-organizado. Esses elementos, por serem equipotentes, não criam de imediato uma tendência viciosa, nem criam um sujeito único, apenas desenvolvem, no processo, suas atividades e recebem um retorno ou realimentação dessas atividades e das de outros elementos. Esse conjunto de atividades constitui o próprio processo auto-organizando-se. Tal dinâmica, que possibilita a criação e a construção, é invocada por Debrun (1996b, p.11) como um caminho para a solução da impossibilidade lógico-matemática levantada por Ashby (1962) - qual seja, da inviabilidade de funções alterarem sua própria estrutura funcional. Há ainda a necessidade de que os elementos possuam uma autonomia relativa que só vai ser diminuída em benefício do Todo que se organiza.

vii) O processo de auto-organização absorve o acaso. No início do processo o acaso tem, segundo Debrun (1996b, p.35), grande importância para

o desenvolvimento do processo, mas aos poucos, conforme o processo adquire sua identidade, o acaso vai perdendo sua influência, sendo absorvido sem causar grandes transtornos. O processo auto-organizado, diz Debrun, *aprende* a lidar com o acaso, e não se desvirtua mais do caminho que escolheu para si, a não ser quando se depara com um ruído forte o suficiente para lhe desestruturar parcialmente, ou até provocar sua *morte*. Nesse contexto, Atlan (1992, p.51, p.70) indica a possibilidade da criação de ordem à partir do ruído, quando o sistema tem sua complexidade aumentada⁴ pelo acaso. Um sistema que estava já se cristalizando, enrijecendo suas estruturas e tornando-as redundantes, de repente tem novamente sua dinâmica revitalizada quando se depara com um ruído desestabilizador.

viii) Não pode haver um supervisor absoluto. Um elemento dominante que imponha a organização aos outros elementos do sistema transforma um processo que poderia ser auto-organizado em hetero-organizado (DEBRUN, 1996a, p.12). A hetero-organização é a organização de um sistema por meio de um supervisor absoluto, ou quase absoluto. Por outro lado, pode existir na auto-organização elementos temporariamente dominantes, desde que requisitados para tal tarefa, e que não se tornem inflexíveis a ponto de não se deixarem mais substituir. Tal supervisor às vezes aparece de modo dissimulado, quase imperceptível, como elemento que se disfarça externamente ao sistema, quando na realidade é parte dele. Um exemplo oferecido por Debrun é o de um patrocinador que compra um boxeador para que faça corpo mole em uma luta específica, determinando de antemão o resultado do processo. A ausência de um

centro absoluto é, segundo Debrun (1996b, p.36), condição necessária para a endogenização do processo.

Há muito que se discutir sobre auto-organização e muitas são as suas propriedades e condições, que essa lista, longe de ser exaustiva, não menciona, porém acreditamos que os pontos levantados acima sejam suficientes para discutirmos a auto-organização dos processos neurais artificiais, simulados em máquinas de Turing, que se seguirão. Antes, porém, gostaríamos de abordar um último conceito: o conceito de *propriedade emergente*.

Quando um processo auto-organizado está se fazendo, surgem efeitos no sistema, cujas causas não podem, segundo Debrun, ser deterministicamente estabelecidas. Esses efeitos colaterais, que podem ser bons ou não para o sistema, são chamados de *propriedades emergentes por processo de auto-organização*. São emergentes de processos auto-organizados porque se não o fossem, sendo propriedades de sistemas dinâmicos determinísticos, seriam antes *consequências* do próprio desenrolar do sistema, e não emergência (DEBRUN, 1996b). Nesse contexto, Clark (1997, p.110) considera a emergência sob outro aspecto:

Emergent phenomena, on this account, are thus any phenomena whose roots involve uncontrolled variables [...] and are thus products of collective activity rather than of single components or dedicated control systems. (CLARK, 1997, p.110)⁵

Essa definição de emergência proposta por Clark é mais condizente com o modo reducionista de Atlan (1998, p.7) de avaliar o problema da emergência. Segundo Atlan⁶, a nova definição de emergência deve ter traços

tanto da corrente filosófica emergentista, – deixando de lado o caráter quase místico dos fenômenos da vida, mantendo a possibilidade de emergência de propriedades não-triviais, – quanto da corrente reducionista, incluindo espaço para o mecanismo reducionista até mesmo no estudo da intencionalidade:

It is reductionist, since emergent properties are not merely described, even less used to explain the behavior of the system, but have to be explained causally, by nothing more than local interactions between the constituents which define, in mechanistic terms, the dynamics of the system. Indeed, from this viewpoint, such an approach is even more reductionist than traditional reductionism, since the aim of this research or, rather, this research program, is something like a physical theory of intentionality. (ATLAN, 1998, p.9)⁷

Ashby (1962, p.272 et seq.) é ainda mais radical, afirmando que a emergência ocorre, inevitavelmente, em todo sistema dinâmico cujas leis permaneçam inalteradas. Nesse sentido, diz Ashby⁸, “*every machine can be thought of as ‘self-organizing’, for it will develop [...] some functional structure...*” (ASHBY, 1962, p.273)⁹. O autor completa seu raciocínio cercado as possíveis propriedades emergentes em um sistema: “*What emerges depends simply on what are the system’s law and from what state it started*” (ASHBY, 1962, p.273)¹⁰.

O debate sobre a noção de propriedade emergente está longe de encontrar unanimidade entre os estudiosos da TAO. A suspensão parcial do determinismo, por exemplo, que Debrun (1996b) propõe, não é compartilhada por Atlan (1998), cujo conceito de acaso está fadado a ser apenas um problema epistemológico. Quando o acaso deixa de ser algo cuja explicação não pode ser

dada por causalidade, para ser tão somente algo que desconhecemos, não há contradição em conceber um *acaso determinístico*, fundamentando uma emergência a partir de leis mecânicas, inconcebível na visão de Debrun.

No que se segue, tratemos destes tópicos aplicados aos sistemas que nos propusemos a analisar, quais sejam, as redes neurais artificiais e as máquinas de Turing.

5.4 - Auto-organização nas Redes Neurais Artificiais

Após a discussão apresentada até este ponto, estamos inclinados a assertar que a qualidade que distinguiria os modelos conexionistas elaborados a partir do patamar da Turing-computabilidade seria aquela que possibilita as propriedades emergentes através do processo de auto-organização, tal como caracterizado por Debrun.

Por que propriedades emergentes *por processo de auto-organização, tal como caracterizado por Debrun*? Não bastaria a apresentação das próprias propriedades emergentes? Não, não bastaria: de fato, as redes neurais artificiais simuladas em máquinas de Turing apresentam muitas propriedades emergentes, características próprias de seus modelos, como a organização em agrupamentos dos neurodos das redes de Kohonen, ou a otimização de funções das redes de Hopfield. Estas propriedades poderiam ser ditas *emergentes* dos processos que as realizaram somente se tais processos fossem genuínos processos de auto-organização. Na visão de Debrun (1996b), propriedades consideradas emergentes em processos determinísticos são, *a priori*, conseqüências do

desenvolvimento do próprio processo, e por isso não podem ser ditas realmente *emergentes*.

Se nos prendermos aos modelos de RNA aqui apresentados, e os que são deles variantes, notaremos características semelhantes em sua estrutura, como o neurodo geral (MCCLELLAND & RUMELHART, 1986), as formas de conexão, os algoritmos que os integram, as equações matemáticas que os definem, entre outras. A partir da visão sistêmica, apresentada na segunda seção deste capítulo, podemos caracterizar uma RNA como um sistema cujos elementos são: os neurodos e as conexões. Esse sistema segue regras que alteram o estado dos neurodos e regras que alteram os estados das conexões. Os elementos de fronteira, que fazem a comunicação entre o meio-ambiente e o sistema neural artificial são os neurodos da camada de entrada e os neurodos da camada de saída, enquanto os outros neurodos são elementos internos ao sistema. O meio-ambiente pode ser real, caso em que os neurodos de entrada atuam fazendo leituras de sensores do ambiente e os neurodos de saída acionam atuadores que interferem no meio. O meio-ambiente também pode ser ideal, representado apenas matematicamente, ou simulado, caso em que os neurodos receberiam estímulos e atuariam em uma máquina simuladora.

Os próprios neurodos podem ser reais, como, por exemplo, em máquinas connexionistas; ideais, representados matematicamente; ou simulados em implementações em máquinas simuladoras. Apresentamos no capítulo 3 os modelos matemáticos, como foram concebidos por seus autores, e vimos, no capítulo 4, as possíveis diferenças entre uma máquina connexionista e uma

máquina de Turing que simula uma RNA. Tentaremos agora focalizar os modelos matemáticos para avaliarmos a possibilidade de auto-organização.

Um dos principais pontos que Debrun coloca para a realização de auto-organização num sistema é o encontro de elementos não analiticamente, mas realmente distintos (DEBRUN, 1996b, p.27, p.29). Os elementos distintos que podemos inferir em uma primeira análise de um modelo de RNA são os neurodos em relação às conexões. Não distinguimos neurodos entre si, nos modelos apresentados neste trabalho, que são todos instâncias de um mesmo e único objeto, apesar de poderem estar em diferentes estados de ativação durante o funcionamento da rede. A complexidade de resposta de uma RNA não vem da qualidade estrutural de seus neurodos, mas da grande quantidade de elementos *idênticos* interagindo. Sobre as conexões, simples multiplicadores da energia esvaída dos neurodos, se distinguem umas das outras apenas pelo fator multiplicativo que incorporam, e em nada mais, sendo sua operação idêntica em todos os pontos da rede. Em uma segunda análise, podemos fazer um recorte mais sutil do sistema, considerando o neurodo o conjunto do seu “corpo celular” (ou seu “soma”) e suas conexões. Esse novo elemento, mais completo, é o único que constitui o sistema neural artificial. Não parece muito razoável supor que aqui existiriam diferenças entre elementos, uma vez que uns teriam mais ou menos conexões que outros, já que sabemos que a inexistência de uma conexão nada mais é que seu valor multiplicativo colocado em zero. Temos, pois, que uma rede neural artificial seria um sistema constituído por um único elemento, do qual se fazem várias instâncias interagirem.

O fato que essas instâncias de um mesmo elemento ficam diferentes ao longo do tempo, cada qual reagindo a determinados estímulos, não os tornam elementos *realmente distintos* como propõe Debrun¹¹. Mas a existência de elementos realmente distintos é o *motor* da auto-organização: elementos distintos e soltos se encontrando para que sua dinâmica possa gerar, criativamente, uma forma (DEBRUN, 1996a, p.9). Essas instâncias de neurodos, combinadas, geram no máximo um espaço de estados possíveis, do qual alguns pontos do espaço são respostas desejadas a determinado problema. Se as respostas desejadas não possuírem configuração adequada no espaço de estados, não podem ser aprendidas pela rede (LUDERMIR, 1992, p.262). Honavar (1995, p.12) afirma que as RNA trabalham como um algoritmo de busca em espaço de estados. O espaço é dado em função das possíveis combinações entre os estados dos neurodos, que podem inclusive ser contínuos, e a busca é feita por um processo paralelo de estabilização da rede.

Dadas as características dos elementos que constituem uma RNA, outro fator logo aparece discrepante: a reprodutibilidade, que Debrun (1996b, p.53) apresenta como incompatível com a auto-organização, é fato nos modelos de RNA. Uma rede inicializada com os mesmos pesos e com os mesmos estados de ativação, recebendo os mesmos padrões de entrada, na mesma ordem e na mesma frequência de tempo, ou em outras palavras, mantidas as mesmas condições, irá convergir para o mesmo ponto de estabilidade final. Isso é verdade também para redes que envolvem ruídos ou outros fatores probabilísticos externos. A saída da rede só não será a mesma se a entrada não o for. Debrun¹²

chama o sistema reprodutível de hetero-organizado, pois o processo segue um algoritmo (reveja o item *iv* da seção anterior). É interessante notar que até esse ponto da análise, a questão dos elementos distintos e a questão da reprodutibilidade dos experimentos, não há nada contra o ponto de vista de Atlan (1998). Atlan sustenta em sua definição de *sofisticação* a possibilidade de termos auto-organização em algoritmos, e propõe um algoritmo¹³ que, segundo o autor, representa um sistema auto-organizado que possui intencionalidade.

O ponto de amarração, ou ponto de encontro entre os elementos, é determinado externamente por um projetista, e possui leis rígidas o suficiente para impedir qualquer criação completamente nova por parte do sistema. É inconcebível, por exemplo, que um modelo de Hopfield extrapole suas condições iniciais e passe a atuar como um modelo de Kohonen, numa demonstração clara de mudança estrutural: as mudanças que ocorrem não são de caráter estrutural, mas informacional apenas.

A temporalidade, outra característica necessária, segundo Debrun (1996b), não tem respaldo nos modelos aqui apresentados. Tais modelos têm uma independência da dimensão do tempo, como se o tempo fosse algo externo a eles. Aplicar os mesmos padrões de entrada nos mesmos modelos, com tempos proporcionalmente diferentes pode produzir o mesmo resultado, mostrando que a reprodutibilidade também se dá para eventos semelhantes, mas distintos na velocidade de apresentação dos padrões. Alguns modelos (por exemplo, BARRETO & ARAÚJO, 2001) levam a questão do tempo mais a fundo, o que indica uma preocupação crescente com esta característica. Esses modelos,

que incorporam características da temporalidade dos eventos, não são reprodutíveis para eventos semelhantes, mas distintos na velocidade de apresentação, entretanto, continuam produzindo os mesmos resultados para eventos idênticos, quantas vezes se fizer o experimento. Admitimos que um grande passo foi dado, precisamos saber em que direção (DREYFUS, 1975).

Julgamos que a reprodutibilidade do experimento é suficiente para enfraquecer os outros itens das condições que apresentamos na seção anterior: uma vez que o experimento pode ser reproduzido, fica claro que o seu desembocar, sendo o mesmo, é determinado de antemão, logo há um objetivo iminente. Além disso, tal processo não absorve o acaso inicial a que é submetido: os pesos iniciais têm força para direcionar o final do processo. Para Atlan (1998), entretanto, o objetivo pode ser dado de antemão, mas não é previsível sem o próprio desenrolar do processo: não há problema com o fato de uma determinada convergência de uma rede ser reproduzida. Tal convergência foi a resposta encontrada pela própria rede, em conjunto com suas regras e dinâmica própria (ATLAN, 1998).

Apesar de altamente paralela e distribuída, a maioria das redes neurais artificiais possuem um supervisor absoluto. Isso fica claro nas redes supervisionadas, mas também pode ser inferido nas redes não-supervisionadas, ou ditas auto-organizadas. Como já citamos em capítulo anterior, existe um algoritmo de treinamento externo à rede, incluindo aqui as redes de Kohonen. O que esse algoritmo de treinamento faz é corrigir os pesos das conexões entre os neurodos que necessitarem tal correção: essa visão estrutural, que consiste em saber qual

conexão corrigir, só pode ser feita por um supervisor externo à rede. No caso das redes de Kohonen, podemos citar outras tarefas do supervisor externo. Quem determina qual foi o neurodo vencedor, comparando as respostas entre todos eles e indicando a maior? Como é calculada a vizinhança que afetará as próximas correções dos pesos? Essas tarefas precisam ser executadas por um mecanismo externo que tenha à sua disposição toda a rede neural artificial.

Acreditamos que o conceito de auto-organização, tal como proposto por Debrun (1996a, 1996b) é flexível demais para aplicar-se aos modelos de RNA. Na nossa interpretação da TAO proposta por Debrun, as RNA não teriam qualquer grau de auto-organização, seriam antes hetero-organização ou então autopoieticas¹⁴ (DEBRUN, 1996a, p.6, 1996b, p.45). Curiosamente, notamos uma passagem em que Debrun¹⁵ discorre sobre redes neurais, e indica sua potencialidade:

[...] parece plausível considerar os processos que intervêm nas redes neurais como fenômenos de auto-organização. Não se trata apenas dos fenômenos de colocação em órbita ou de modificação (por “espessamento” ou “definhamento” das conexões entre elementos da rede, de acordo com a “regra de Hebb”) da própria rede, mas também das suas operações corriqueiras, à medida que essas operações parecem implicar “antecipações/apostas”, compartilhadas por vários neurônios cerebrais ou “lógicos” em interação [...], logo antes do ajuste perceptivo, motor etc., que vai surgir entre eles. (DEBRUN, 1996b, p.48)

Essa citação, unida com as explicações sobre como algoritmos poderiam caracterizar sistemas hetero-organizados, que Debrun¹⁶ coloca no mesmo texto, nos faz ponderar que ao autor não foi informado do problema da

reprodutibilidade das RNA, fato que não nos surpreenderia devido ao estilo coercivo de tais textos. Por sabermos incompatível para Debrun processos reprodutíveis algoritmicamente com a auto-organização, vemos duas saídas para o dilema da nossa leitura da TAO de Debrun: nesse contexto, acreditamos que Debrun tenderia a ajustar sua teoria da auto-organização de modo a comportar as idéias mecanicistas de Atlan (1998) e Ashby (1962), ou então, o que nos parece mais provável, manteria imaculada sua teoria, fundamentada principalmente no senso comum, e eliminaria a RNA da classe de sistemas auto-organizáveis.

Por outro lado, os pontos aqui analisados são de pouca importância na visão de Atlan (1998, p.5), que acredita na conciliação entre a auto-organização e o determinismo físico. Na prática, diz Atlan (1998, p.7), é impossível diferenciar o comportamento de um programa determinístico *infinito* com o comportamento do acaso. O que significa, na visão do autor, que na prática, é impossível distinguir um sistema mecânico muito complexo, que apresenta características de auto-organização, da *verdadeira* auto-organização (se é que esta existe!).

Basta que o sistema consiga selecionar, entre o conjunto de estados possíveis, aqueles que lhes são caros, para que tenhamos auto-organização, como concebida por Ashby (1962) ou Atlan (1998). Os estados selecionados serão sempre os *melhores para o sistema*, do ponto de vista *do sistema* (ASHBY, 1962, p.273). Para esses autores, os modelos de RNA teriam atingido o grau dois de auto-organização, o grau máximo que uma máquina poderia alcançar. Atlan (1998, p.9, p.14) deixa-nos com a sensação, devido à sua

insistência em um reducionismo da auto-organização e da intencionalidade para uma teoria física, de que afinal, o grau quatro de auto-organização não passe de uma utopia.

Under the influence of the mathematical theory of computability, it is generally assumed that everything in nature is either computable or random. This assumption, known under the name of the physical Church-Turing thesis, implies that self-organization systems could not exist in nature. (ATLAN, 1998, p.24)¹⁷

Não nos parece que o “problema da parada”, por exemplo, um problema bem definido e que por isso pôde ser demonstrado como não Turing-computável (TURING, 1936), seja uma seqüência randômica de caracteres. Ao avaliarmos tal afirmação, de que *é geralmente assumido que* na natureza temos ou funções computáveis ou seqüências aleatórias, explica-se em grande parte o comprometimento de Atlan com sua teoria da sofisticação: um programa infinito não seria nem computável, nem randômico, dando a Atlan o meio termo que falta para que a tese de Turing-Church não fique tão completa quanto a citação acima impõe. Propor a sofisticação é válido, mas ela não deve ser considerada a única forma de se conseguir funções não computáveis.

Classificar os modelos de RNA como sendo do grau dois de auto-organização de Debrun poderia dar a falsa impressão de que o autor concordaria com essa colocação. Ao nosso ver, na concepção de Debrun (1996a) as RNA seriam sistemas hetero-organizados ou mesmo autopoieticos, por isso, se deixarmos de lado as características que necessitam um sistema AO na visão do autor, deveríamos também deixar de lado sua classificação, e considerar, na

denominação de Atlan (1998), que os modelos de RNA apresentam a característica de auto-organização forte.

5.5 - Auto-organização nas máquinas de Turing

As máquinas de Turing não são auto-organizadas.

No capítulo 2 vimos a definição de máquinas de Turing, sua descrição e seu funcionamento. Sabemos, portanto, que não há qualquer possibilidade de tal máquina realizar qualquer operação que não tenha sido expressamente programada por um agente externo, seja ele um ser humano ou outra máquina. A análise da auto-organização nas máquinas de Turing não é tão simples quanto pode parecer à primeira vista.

Mais precisamente, Lady Lovelace objetou que tais máquinas não podiam fazer nada além do que estariam programadas para fazer (TURING, 1950, p.56). Turing¹⁸ rebateu a objeção sugerindo que nada impede que se programe uma máquina para que ela aprenda a solucionar seus próprios problemas.

Estruturalmente, entretanto, as máquinas de Turing não podem se alterar. Sempre seguirão os sete passos apresentados na tabela 1. Falar sobre auto-organização *nas* máquinas de Turing deve ser substituído, para evitar ambigüidade, para auto-organização *dentro das* máquinas de Turing: a própria máquina não se auto-organiza. Mas seria possível criar sistemas nela programados que exibam padrões informacionais em sua(s) fita(s) que representem processos se auto-organizando?

Os algoritmos de redes neurais artificiais simulados em máquinas de Turing são justamente esses processos, segundo Atlan (1998). Na visão de Debrun (1996b), tais processos estariam amarrados a uma estrutura rígida, e sendo determinísticos, reprodutíveis, não poderiam ser considerados auto-organizados.

A diferença que procuramos manter entre a evolução de um sistema dinâmico corrente e um processo auto-organizado reside precisamente na idéia de que o segundo comporta aspectos criativos. (DEBRUN, 1996b, p.46)

Ao nosso ver, tal criatividade intrínseca deve ser relativizada, sob pena de não termos parâmetros para comparar sistemas menos complexos que os seres vivos se não o fizermos. Atlan (1998) propõe que a criatividade do sistema deriva da sua capacidade de caminhar no espaço de estados em busca de soluções que não foram *explicitamente* programadas. Apesar de não terem sido explicitamente programadas, tais soluções são conseqüências indiretas das regras que atuam no sistema.

Não estamos tratando aqui das regras de “mover para direita” ou “mover para esquerda”, que são seguidas pela cabeça de leitura e gravação da máquina de Turing, mas de regras muito mais complexas que podem ser codificadas na fita da máquina: as regras de interação entre os neurodos, a regra de “aprendizagem” das conexões, etc. Externamente à simulação, temos uma máquina cuja potencialidade de programação é enorme, tão grande quanto a quantidade de funções computáveis existentes. Uma vez programada com determinado modelo de RNA, seu espaço de estados é drasticamente reduzido

àquele do modelo, o que não significa que o espaço do modelo seja pequeno. Pelo contrário, o espaço de estados de um modelo de RNA pode ser tão grande quanto se queira, idealmente, dado pela quantidade e qualidade de neurodos (se binários, ternários etc.), número de conexões etc. Esse enorme espaço de estados é selecionado *automaticamente* pela rede, quando em funcionamento, reduzindo-o ainda mais: essa capacidade da RNA que é chamada de auto-organização (ASHBY, 1962, p.273). Ao projetista é dada uma ferramenta poderosa: ele não precisa cercar detalhadamente o problema em questão, o que levaria a uma programação complexa e minuciosa. Ele apenas cria uma rede cujos limites ele sabe que abarcam o problema em questão, e a rede busca no espaço que tem disponível a solução que não foi pré-programada.

Não há, nos processos simulados em máquinas de Turing, as propriedades emergentes, no sentido de Debrun (1996b), há apenas consequência. Mas essa consequência, segundo Atlan (2001), é de natureza tão complexa que dificilmente pode ser distinguida do verdadeiro processo de auto-organização, uma vez que só saberemos a resposta da simulação depois de feita.

A concepção de criatividade do processo continua a ser aproximadamente aquela mesma que Debrun (1996b) coloca: a dinâmica da interação entre elementos que seguem uma lei. As diferenças, de que os elementos não sejam distintos, e que a lei seja determinística, não eliminam, na visão de Atlan (1998) a característica de auto-organização forte que esses processos possam apresentar. Uma resposta de um algoritmo pode ser criativa, pois pode expressar a intersecção entre as várias leis programadas (linhas causais) cruzando-



se num determinado ponto do espaço de estados. O acaso determinístico de Cournot toma lugar na metafísica de Atlan numa intensidade vista antes em Descartes (1999a), e que parecia ter saído do meio científico.

A aprendizagem tem um significado especial numa máquina de Turing (simulando ou não uma RNA). Acontece uma maior condicionalidade (ASHBY, 1962) entre os elementos descritos em sua fita, condicionalidade esta dada por meio de uma regra também descrita nesta fita. Uma busca realizada com sucesso por um processo “criativo” (no sentido dado acima) é reforçada e tem diminuído o seu tempo de resposta para o caso de uma segunda busca. Esse condicionamento pode ser feito a partir de exemplos (caso do algoritmo de retropropagação) ou pode ser dinamicamente determinado por regras de interação (caso das redes de Kohonen). Esse significado é decapitado das qualidades do aprendizado humano:

Misturar o conceito técnico de “treinar” (como em “treinar a rede”) com suas aplicações não-técnicas, correntes e de uso humano, como em, por exemplo, “treinar um grumete”, “treinar uma criança nos hábitos higiênicos”, ou “treinar uma criança pequena na tabuada de multiplicação” dá origem não a melhores teorias científicas, mas sim a confusão desenfreada. (BUTTON et al., 1998, p.173)

A argumentação de Button et al. (1998) parece coerente quando entendemos que as regras nos dão de antemão o resultado do “aprendizado”, o que descaracterizaria qualquer esforço do sistema em buscar suas próprias soluções. Não há, segundo Button et al.¹⁹, nenhuma autorização para tratar a nova significação do termo “aprender” como tese para uma “teoria psicológica do

aprendizado”. O que Atlan (1998) propõe é reduzir a psicologia a termos mecânicos, pois apesar das limitações do aprendizado das máquinas hoje, não há porque não acreditar que toda a abrangência do conceito de aprendizado humano não seria devido às leis (mais complexas) que o sistema humano está submetido, e que de fato, nossas reações também estão dadas de antemão, determinadas por um programa de sofisticação infinita.

A qualidade da resposta buscada, é sempre boa: “*To itself, its own organization will always, by definition, be good*” (ASHBY, 1962, p.273). Por ser a resposta dinâmica do próprio sistema às leis que o regem, sua organização tende a ser a melhor nas dadas condições. É uma característica, segundo Ashby (1962), de todos os sistemas dinâmicos colocados sob influência de leis imutáveis, que uma organização emerge, e que esta organização é boa. Mas essa é a noção de auto-organização, também segundo Debrun (1996a), com exceção de que tais leis não poderiam determinar completamente o comportamento do sistema. Em Debrun²⁰ há um espaço para o livre-arbítrio, dando mais liberdade para que os elementos *escolham* entre as decisões possíveis de serem tomadas, pelo menos em parte, contrariando sua própria lei de desenvolvimento. Esse espaço não existe em Atlan (2001), de forma que a questão de haver ou não auto-organização nas máquinas de Turing toca um ponto polêmico na visão que temos do mundo: o quanto nossos atos estão sob nossa responsabilidade, e quanto é devido às leis que nos regem no universo?

Atlan (2001) colocou uma questão pertinente quando levantou o fato de que todos os sistemas no universo seguem leis, e que não se sentia

confortável em assumir um posicionamento de ser, ele próprio, um sistema especial, que fugiria a esta regra, possuindo assim livre-arbítrio. Como ficaria aqui a questão das funções sabidamente não-Turing computáveis? Estamos longe de concordar com a visão que Atlan (1998) tem da tese de Turing-Church, de que na natureza ou temos processos computáveis ou são processos aleatórios, e com a sua noção de sofisticação como único modo de driblar esse impasse.

Entretanto, torna-se conveniente permitir que se fale em auto-organização em sistemas determinísticos, como meio de explicar as propriedades que surgem na interação dos elementos do sistema. Não poderíamos discutir aqui a existência da *verdadeira* auto-organização, questão antes metafísica, que deve ser analisada sem nos prendermos a modelos científicos, seja de RNA ou quaisquer outros, mas procurada no significado de termos como *acaso* e *livre-arbítrio*.

O que podemos admitir na nossa análise é a possibilidade de auto-organização forte em máquinas cuja complexidade torna sua resposta algo que permeia o limite entre o determinismo e o previsível. Esse é o limite de nossa compreensão, para além do qual conjecturas, sobre ser o acaso apenas algo que desconhecemos ou que existe independente de leis, tomam lugar.

Nesse capítulo vimos a teoria da auto-organização nas visões de Debrun, Ashby e Atlan, e levantamos as características que tais autores acham relevantes para avaliar as possibilidade de auto-organização em sistemas artificiais, na seção 5.3. Dentre essas características, notamos divergências em relação ao papel do acaso e do determinismo entre Debrun e Atlan. Na seção 5.4,

avaliamos pontualmente essas mesmas características, voltando-nos para as RNA, e pudemos concluir que existe uma impossibilidade de tais modelos serem auto-organizados de acordo com Debrun, devido, entre outros fatores, principalmente à rigidez em que tais modelos se encontram estruturados. Contrário a essa postura, discutimos o conceito de auto-organização de Atlan, que permite uma maior entrada do determinismo. Na seção 5.5, discutimos a possibilidade de auto-organização *dentro das* máquinas de Turing. A princípio, notamos que mudanças estruturais na própria máquina estariam descartadas, por definição. A seguir, a discussão passou a enfocar a possibilidade de simular sistemas completos, com seus elementos e suas leis, codificados em uma máquina de Turing. Esses sistemas, apesar de estarem moldados em uma máquina de Turing, conseguem, no limite de suas possibilidades, buscar soluções consideradas por Atlan (1998) criativas e intencionais. Sistemas determinísticos, segundo Debrun (1996b) não podem ser auto-organizados, no que diverge Atlan (1998), considerando que a complexidade da resposta aproxima tais sistemas do limite da *verdadeira* auto-organização. Vimos também o posicionamento de Ashby (1962) que acredita que todo sistema dinâmico interagindo segundo leis específicas se auto-organiza, em algum sentido restrito.

Essas divergências entre tais autores se devem principalmente às suas visões de mundo, que influenciam significativamente suas definições de auto-organização. Tais definições deveriam chegar a um consenso, para se poder afirmar com clareza sobre a auto-organização em máquinas, uma vez que a definição de máquina de Turing já nos foi legada formal e claramente, e não se

alterará com qualquer metafísica que se adote. Até que a auto-organização seja de fato formalizada, fica a critério de quem avalia definir quais sistemas se auto-organizam. Debrun (1996a) bem coloca o senso comum na definição, pois definir auto-organização parece tarefa mais intuitiva que lógico-matemática. Talvez, o que estamos tentando, ao associarmos a auto-organização às máquinas de Turing seja aproximadamente o mesmo que o fez a associação entre calculabilidade efetiva e Turing-computabilidade: ligar uma noção intuitiva a uma noção formal, a tese de Turing-Church. Mas essa ligação se torna contra-intuitiva se atribuirmos à teoria da auto-organização a capacidade potencial de explicar as funções não Turing-computáveis.

5.6 - Notas

¹ A noção de sujeito disposicional é inserida neste contexto para indicar que a existência ou não de um sujeito que recorta o sistema a ser apreciado não implica na existência ou não de sistemas de modo geral.

² Tradução nossa: ...a organização é dita “boa” se, e somente se, ela atua de modo a manter um designado conjunto de variáveis, as variáveis “essenciais”, dentro de limites designados. [...] uma organização é “boa” se ela faz o sistema permanecer estável em torno de um equilíbrio designado.

³ Ou procedimento sistemático. Não podemos falar em algoritmo com sofisticação infinita, uma vez que a definição de algoritmo implica sua finitude.

⁴ Quando se dá a redução de restrições do sistema.

⁵ Tradução nossa: Os fenômenos da emergência, deste modo, são então quaisquer fenômenos cujas raízes envolvem variáveis não controladas [...] e são então produtos de uma atividade coletiva ao invés de [produtos de] simples componentes ou sistemas de controle dedicados.

⁶ Ibidem.

⁷ Tradução nossa: É *reducionista*, uma vez que as propriedades emergentes não são meramente descritas, ou até mesmo usadas para *explicar* o comportamento do sistema, mas devem *ser explicadas* causalmente, por nada mais que interações locais entre os constituintes que definem, em termos mecânicos, a dinâmica do sistema. De fato, desse ponto de vista, tal abordagem é ainda mais reducionista que o reducionismo tradicional, uma vez que o objetivo desta pesquisa, melhor, deste programa de pesquisa, é algo como uma teoria física da intencionalidade.

⁸ Ibidem.

⁹ Tradução nossa: toda máquina pode ser pensada como sendo auto-organizada, pois ela irá desenvolver [...] alguma estrutura funcional...

¹⁰ Tradução nossa: O que emerge depende simplesmente de quais são as leis do sistema e a partir de qual estado ele iniciou.

¹¹ Ibidem.

¹² Ibidem.

¹³ Atlan (1998, p.29) apresenta uma rede booleana, conectada de forma que cada elemento receba duas entradas binárias de seus vizinhos, e depois de feitas computações internas, responda com sua própria saída para dois outros vizinhos.

¹⁴ O processo é decorrência do seu próprio começo. “Esta frase tem vinte e oito letras”.

¹⁵ Ibidem.

¹⁶ Ibidem.

¹⁷ Tradução nossa: Sob a influência da teoria matemática da computabilidade, é geralmente aceito que tudo na natureza é ou computável ou randômico. Essa asserção, conhecida pelo nome de tese física de Church-Turing, implica que sistemas verdadeiramente auto-organizados não existem na natureza.

¹⁸ Ibidem.

¹⁹ Ibidem.

²⁰ Ibidem.

Notas Finais

N.1 - Conclusões

Este trabalho se propôs a investigar inicialmente dois problemas principais, o problema da emergência, que consiste em saber se há perda de propriedades emergentes nas RNA simuladas em máquinas de Turing, e o problema da Turing-computabilidade, que consiste em avaliar se os modelos de RNA extrapolam as funções Turing-computáveis.

No decorrer do trabalho, avaliamos muitas características das duas grandes vertentes da Ciência Cognitiva, as Redes Neurais Artificiais e a Inteligência Artificial, tocando ainda em algumas outras vertentes, como a Ciência Cognitiva Dinâmica, por exemplo. As características que levantamos foram contextualizadas em uma perspectiva histórico-filosófica, frisando os avanços destas vertentes, por menores que sejam, em direção ao objetivo de modelar uma mente inteligente.

Discutimos as hipóteses em que estão fundamentadas tais vertentes, quais sejam, a hipótese representacionista, a hipótese funcionalista e a hipótese mecanicista computacional, chegando à conclusão de que elas não se diferenciam substancialmente. Tanto a IA quanto as RNA prevêm que o conhecimento do mundo seja intermediado por representações internas, concebem seu estudo através de estruturas funcionais que poderiam substituir o substrato

cerebral, e admitem que os estados mentais são regidos por leis mecânicas, tal que, conhecidas as leis pode-se explicar a mente através de um modelo computacional – com maior ou menor grau de complexidade, dependendo do modelo adotado para a simulação.

Iniciamos a dissertação com o tratamento dos modelos de IA, e de sua abordagem simbolista, apresentando alguns algoritmos para exemplificações, e tivemos a oportunidade de ver quão semelhante eles são aos algoritmos apresentados no capítulo seguinte, das RNA, cuja abordagem, em princípio, seria não simbolista.

Os conceitos abordados, tais como aqueles de aprendizagem e emergência, mostram-nos que não podemos simplificá-los para adequá-los a uma teoria que não possui elementos para explicar a sua complexidade, sob penalidade de perdermos o âmago do conceito. Isso é o que acontece quando tentamos explicar termos cuja raiz está na possibilidade do novo, como o caso da criatividade, dentro de modelos da máquina de Turing, determinística por definição. Se o modelo não explica o fenômeno, devemos modificar o modelo, não cegarmo-nos para o fenômeno.

Ressaltamos que as redes neurais artificiais apareceram como um novo modelo para se tentar lidar com tais dificuldades. Entretanto, procuramos mostrar que suas definições vieram carregadas dos vícios dos modelos da IA, incorporadas nos algoritmos e símbolos, que não conseguiram se desfazer. Argumentamos que, não bastasse o próprio modelo ser ele mesmo apenas uma nova roupagem, de algoritmos seqüenciais para algoritmos paralelos, ainda há o

uso indiscriminado de simulações de RNA em máquinas de Turing, tendo seus bons resultados sido divulgados como prova de que a nova abordagem extrapolava os modelos *algorítmicos* (CAUDILL & BUTTLER, 1992).

Julgamos que não é com a intenção de confundir ambas as vertentes que os connexionistas se utilizam dos computadores digitais, mas pela dificuldade financeira de criar um modelo neural artificial físico para cada tentativa de experimentar uma nova hipótese, sendo que é exatamente este o papel revolucionário que os computadores vieram a ter, qual seja, de simular os experimentos em praticamente todas as áreas do conhecimento, barateando sobremaneira os custos das pesquisas. Entretanto, nenhuma outra área do conhecimento que se vale de simulações, propiciou tanto conflito entre os seus teóricos como o fez o connexionismo, utilizando-se da ferramenta de que queria se afastar, o computador digital, um manipulador de símbolos por excelência. Contudo, não podemos deixar de considerar que os modelos de RNA, ao introduzirem os conceitos de complexidade computacional, interação dinâmica entre elementos simples e de propriedades emergentes, constituem um grande avanço nos métodos atualmente disponíveis de se estudar a mente.

Avaliamos, no capítulo 4, as características que diferenciariam os modelos de RNA das máquinas de Turing, e tivemos a forte indicação de que as máquinas de Turing continuam sendo os modelos que representam categoricamente qualquer máquina, caracterizando ainda a definição do próprio termo *máquina*. Não há, ao nosso ver, um modelo mecânico que extrapole os limites da máquina de Turing. Isso não significa que não existam modelos não-

mecânicos que o façam, mas tais modelos perderiam a qualidade de apresentarem procedimentos sistemáticos. Qualidade essa que, aliás, não necessita ser mantida a todo custo. Não há motivo convincente para crer que o cérebro humano processe sistematicamente todos os seus pensamentos, mesmo que se admita a existência de procedimentos mecânicos para algumas áreas do pensamento racional. Temos até motivos contrários a isso, como o fato de nós conseguirmos compreender o “problema da parada”, por exemplo, um problema que uma máquina de Turing não resolve.

A discussão sobre níveis de análise foi levantada justamente para entendermos quais seriam as diferenças entre as RNA e a IA. Num nível mais baixo, o implementacional, parece-nos que podemos ter uma equivalência entre os modelos de RNA e IA, isso devido à tese de Turing-Church segundo a qual uma máquina de Turing é capaz de executar qualquer procedimento efetivo. Ora, como vimos, existem algoritmos simples e efetivos para computar as saídas das RNA. Em outro plano, entretanto, as RNA parecem mais condizentes com a teoria que tenta explicar como o cérebro humano produz uma mente (figura 21). Mesmo que tal teoria, que focaliza as noções de emergência, complexidade, ruído, entre outras, não consiga atingir seu objetivo de modo completo, as RNA deram um grande passo ao mostrar que existem propriedades simples (como um tipo limitado de aprendizado) que surgem (emergem, na visão de Atlan, 1998), das interações entre elementos simples que compõe o cérebro.

As propriedades que emergem das redes neurais artificiais simuladas em máquinas de Turing são, na visão de Ashby (1962), propriedades de



sistemas dinâmicos com certas características, não sendo, pois, surpresa sua aparição. Essa característica dos sistemas dinâmicos com alguma complexidade só foi possível de ser descoberta com o advento dos computadores digitais, que forneceram, segundo Atlan (1992), objetos nem tão simples a ponto de serem triviais, nem tão complexos a ponto de serem mistério. Se tais propriedades podem ser caracterizadas como resultantes de processos auto-organizados ou se ficam somente relegadas à propriedades de sistemas dinâmicos complexos é devido ao conceito que se faz de auto-organização. O ponto é que sabemos muito mais sobre as máquinas hoje, do que sabemos sobre auto-organização. A classificação de processos mecânicos como auto-organizados só pode ser considerada uma proposta científica se pudermos conceber um universo, por hipótese metafísica, determinista. Atlan (1998) acredita que é possível conceber AO num universo determinista, bastando para isso adequar os conceitos da TAO.

Novamente contrastamos a opinião de Debrun. Não há liberdade de elementos em sistemas determinísticos. Num universo determinístico as interações são, em última instância, pré-definidas, não importando se os resultados das interações possam ser conhecidos de antemão ou não. Não há papel para o acaso, a não ser como sinônimo de desconhecido. Esse universo rígido parece um modelo empobrecido da realidade fluídica que vivemos. A TAO tenta justamente caracterizar o que o determinismo deixou de lado, por isso, na visão de Debrun, relegá-la ao determinismo, ou à sua conceituação lógico-matemática, é, no mínimo, um contra-senso.

Mas então, se fosse o caso, seria preferível renunciar à utilização do termo “auto-organização”, em vez de mantê-lo à força para designar, de acordo com certos modismos (frequentes em algumas áreas científicas), fenômenos que seriam corretamente descritos e explicados sob outras denominações. (DEBRUN, 1996a, p.4)

Nesse sentido, não precisamos da TAO para explicar mecanismos determinísticos. Se o universo for determinístico, não precisamos da TAO para nada mais: o termo auto-organização perde seu significado.

N.2 - Pesquisas futuras

O paradigma das RNA, pela sua inovação na introdução dos conceitos tais como aqueles da emergência, conseguiu resultados significativos no propósito de explicar a inteligência, apesar de serem simples e estarem longe de resolver a complexidade da mente inteligente. Assim, por exemplo, utilizando o conceito de emergência através de simples interações entre elementos de um sistema, os modelos de RNA oferecem uma explicação possível do difícil problema da relação cérebro-mente. Ao possibilitar a simulação de tal fenômeno de emergência, a máquina de Turing teve renovada sua utilidade e seu alcance, indicando um novo caminho que não o dos algoritmos exatos em suas respostas, mas aproximativos, onde o erro é aceito, e trabalha-se com probabilidades.

Qual é o limite explicativo desses modelos? Até que ponto as máquinas de Turing podem atingir utilizando esses modelos de RNA? Seria o caso de se perguntar se as máquinas de Turing podem extrapolar as funções Turing-computáveis, quando damos à ela a liberdade de errar? Qual é o horizonte

de possibilidades que se configura quando programamos não apenas um algoritmo que demonstra uma determinada resposta, do tipo sim e não, mas que busca através de interações entre elementos e suas restrições na forma de regras, respostas num amplo espaço de soluções? Uma vez que o espaço de soluções parece ser restrito, *a priori*, pelo espaço das funções Turing-computáveis, que tipo de novidades podemos esperar de suas simulações? Mesmo que fiquemos apenas com as funções Turing-computáveis, ainda não sabemos até onde podemos simular processos inteligentes usando tais funções. Ainda não conhecemos que grau de inteligência é possível uma máquina simular, dados os seus recursos e pressupostos mecanicistas.

Conforme já frisamos, os modelos de RNA se mostram bons interpoladores de funções. Seria possível conseguir modelos de RNA simulados em máquinas de Turing que interpolem com boa precisão funções sabidamente não Turing-computáveis?

Ressaltamos que para se conseguir modelos de RNA, que realmente extrapolem os limites das funções Turing-computáveis, especial atenção deveria ser dada aos modelos de redes neurais artificiais recorrentes e analógicos. A característica do processamento analógico, dentre todas as que avaliamos, parece-nos a mais promissora para se conseguir máquinas super-Turing (FLORIDI, 2001). A máquina de Turing universal, capaz de simular quaisquer outras máquinas de Turing, só o faz devido à finitude dos estados internos de sua definição. Nesse sentido, uma máquina de Turing universal não é capaz de simular uma máquina analógica, sem o artifício de discretizar os estados

internos (e os tempos de processamento) da máquina analógica. Esse artifício pode ser utilizado, pois não altera o resultado, para máquinas analógicas cujas precisões podem ser suplantadas. Será que todas as máquinas analógicas podem ter suas precisões suplantadas? O cérebro seria uma delas? A mente obedeceria às mesmas leis do cérebro?

Tais questões ainda encontram-se sem resposta no contexto da atual Ciência Cognitiva. Com a presente dissertação esboçamos os diversos e complexos caminhos que poderiam nos conduzir à sua compreensão. Contudo, tais caminhos ainda estão por se construir e como dizia o poeta, o caminho se faz ao caminhar!

Referências bibliográficas

ABBAGNANO, N. *Dicionário de filosofia*. 2.ed. Tradução: A. Bosi. São Paulo: Martins Fontes, 1998. Original italiano.

AGRE, P. E.; CHAPMAN, D. Pengi: an implementation of a theory of activity. *Proceedings of AAAI-87*. Menlo Park, CA: AAAI, 1987, p.268-272 apud FRANKLIN, S. *Artificial Minds*. Cambridge: The MIT Press, 1995. cap.14.

ALVES, M. A. *Mecanicismo e inteligência: um estudo sobre o conceito de inteligência na ciência cognitiva*. 1999. 110f. Dissertação (Mestrado em Ciência Cognitiva e Filosofia da Mente) - Faculdade de Filosofia e Ciências, Universidade Estadual Paulista "Julio de Mesquita Filho", Marília.

ANDERSON, J. A. Properties of single neurons. In: _____. *An introduction to neural networks*. Cambridge: The MIT Press, 1995. cap.1, p.1-35.

ARAÚJO, R. HP netserver LH 6000. *PC Master*, São Paulo, ano 4, n.10, ed.46, p.36-38, mar. 2001.

ASHBY, W.R. Principles of the self-organizing system. In: VON FOERSTER, H.; ZOPF JR., G. W. (Org.). *Principles of self-organization*. Oxford: Pergamon, 1962. p.255-278.

ATLAN, H. *Entre o cristal e a fumaça*. Rio de Janeiro: Zahar, 1992.

_____. Intentional self-organization, emergence and reduction: towards a physical theory of intentionality. *Thesis-Eleven*, n.52, p.5-34, 1998.

_____. Living and knowing. In: COLÓQUIO MICHEL DEBRUN, 4, 2001, UNESP, Marília. Não publicado.

BARRETO, G. A.; ARAÚJO, A. F. R. Time in self-organizing maps: an overview of models. *International Journal of Computer Research*, v.10, n.2, p.235-259, 2001.

BENANTE, R. C. Self-organization on hybrid neural network. In: HERBSTAKADEMIE MEETING, 9, 2000, Ascona, Suíça. *Resumo...* Ascona: Centro Stefano Franscini, 2000. p.20.

_____. Redes neurais artificiais Turing-computáveis. In: SIMPÓSIO EM FILOSOFIA E CIÊNCIA, 4, 2001, Marília. *Resumo...* Marília: Unesp-Marília-Publicações, 2001a. p.109.

_____. Auto-organização nas máquinas de Turing. In: COLÓQUIO MICHEL DEBRUN, 1, 2001b, Campinas, UNICAMP. Não publicado.

BIRABEN, R. C. E. *Tese de Church*: algumas questões histórico-conceituais. Campinas: CLE/UNICAMP, 1996. (Coleção CLE, v.16).

BITTENCOURT, G. *Inteligência artificial: ferramentas e teorias*. Florianópolis: Ed. da UFSC, 1998.

BODEN, M. (Ed.). *The philosophy of artificial intelligence*. Oxford: Oxford University Press, 1990.



_____. (Org.). *Dimensões da criatividade*. Tradução P. Theobald. Porto Alegre: Artmed, 1999. Original inglês.

BRESCIANI FILHO, E.; D'OTTAVIANO, I. M. L. Conceitos básicos de sistêmica. In: D'OTTAVIANO, I. M. L.; GONZALEZ, M. E. Q. (Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 2000. p.283-306. (Coleção CLE, v.30).

BRETON, P. *História da informática*. São Paulo: Ed. UNESP, 1991.

BROADBENT, D. A question of levels: comment on McClelland and Rumelhart. *Journal of Experimental Psychology*, v.114, p.189-192, 1985.

BROOKS, R. A. Intelligence without representation. In: HAUGELAND, J. (Ed.). *Mind design II*. 2.ed. London: The MIT press, 1997.

BUTTOM, G. et al. *Computadores, mentes e conduta*. São Paulo: Ed. UNESP, 1998.

CALINGAERT, P. *Princípios de computação*. Rio de Janeiro: Ao Livro Técnico, 1969.

CAUDILL, M.; BUTLER, C. *Understanding neural networks: computer explorations*. Massachusetts: The MIT Press, 1992, 2 v.

CHURCHLAND, P. M.; CHURCHLAND, P. S. Could a machine think? In: _____. *On the contrary: critical essays, 1987-1997*. Cambridge: The MIT Press, 1998. cap.5. p.47-63.

CLARK, A. Emergence and explanation. In: _____. *Being there: putting brain, body, and world together again*. London: The MIT Press, 1997. cap.6, p.103-128.

CRICK, F. The recent excitement about neural networks. *Nature*, n.337, p.129-132, 1989.

DAMÁSIO, A. R. *O erro de Descartes: emoção, razão e o cérebro humano*. São Paulo: Companhia das Letras, 1996.

DE LEEUW, K et al. Computability by probabilistic machines. In: SHANNON, C. E.; MCCARTHY, J. (Ed.). *Automata studies*. New Jersey: Princeton University Press, 1956. p.183-212.

DEBRUN, M. A idéia de auto-organização. In: DEBRUN, M.; GONZALEZ, M. E. Q.; PESSOA JÚNIOR, O. (Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 1996a. p.3-23. (Coleção CLE, v.18).

_____. A dinâmica de auto-organização primária. In: DEBRUN, M.; GONZALEZ, M. E. Q.; PESSOA JÚNIOR, O. (Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 1996b. p.25-59. (Coleção CLE, v.18).

_____. Auto-organização e ciências cognitivas. In: GONZALEZ, M. E. Q. et al. (Org.). *Encontro com as ciências cognitivas*. Marília: Unesp-Marília-Publicações, 1997. v.1, p.27-34.

DENNETT, D. C. *Tipos de mentes: rumo a uma compreensão da consciência*. Rio de Janeiro: Rocco, 1996.

_____. *Brainstorms: philosophical essays on mind and psychology*. Cambridge: Cambridge Massachusetts Press, 1978.

DESCARTES, R. *Discurso do Método*. São Paulo: Nova Cultural, 1999a. p.33-100. (Coleção os Pensadores).

_____. *As paixões da alma*. São Paulo: Nova Cultural, 1999b. p.101-245. (Coleção os Pensadores).

_____. *Meditações*. São Paulo: Nova Cultural, 1999c. p.247-334. (Coleção os Pensadores).

DREYFUS, H. *O que os computadores não podem fazer: uma crítica da razão artificial*. Rio de Janeiro: A Casa do Livro Eldorado, 1975. Original inglês.

_____. *What computers still can't do: a critique of artificial reason*. 2.ed. rev. e aum. New York: Harper & Row, 1993.

DUPUY, J. P. *Nas origens das ciências cognitivas*. São Paulo: Ed. Unesp, 1995.

FLORIDI, L. Turing vs. super-Turing: a defence of the Church-Turing thesis. *The University of Oxford*, maio 2001. Disponível em:
<<http://www.wolfson.ox.ac.uk/~floridi/ctt.htm>>. Acesso em: 2 maio 2001.

FODOR, J. A.; PYLYSHYN, Z. W. Connectionism and cognitive architecture: a critical analysis. In: HAUGELAND, J. (Ed.). *Mind design II*. 2.ed. London: The MIT Press, 1997.

FRANKLIN, S. *Artificial Minds*. Cambridge: The MIT Press, 1995.

GARDNER, D. Introduction: toward neural neural networks. In: _____. (Ed.). *The neurobiology of neural networks*. Massachusetts: The MIT Press, 1993. cap.1, p.1-12.

GARDNER, H. *A nova ciência da mente*. 2.ed. São Paulo: Edusp, 1996.

GOÑI, J. (Ed.). Digital: funções digitais. *Monte seu Laboratório de Eletrônica*, Madrid, n.2, f.1, 2000.

GONZALEZ, M. E. Q. *A cognitive approach to visual perception*. 1989. 189 f. Tese (Doutorado em Filosofia) - University of Essex, Colchester, UK.

_____. Ação, causalidade e ruído nas redes neurais auto-organizadas. In: DEBRUN, M.; GONZALEZ, M. E. Q.; PESSOA JÚNIOR, O. (Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 1996. p.273-297. (Coleção CLE, v.18).

_____. Auto-organização e perspectivismo: algum acréscimo à ciência cognitiva? In: GONZALEZ, M. E. Q.; BROENS, M. C. (Org.). *Encontro com as ciências cognitivas*. Marília: Unesp-Marília-Publicações, 1998. v.2, p.3-13.

GONZALEZ, M. E. Q.; BROENS, M. C.; SERZEDELLO, J. Auto-organização, autonomia e identidade pessoal. In: D'OTTAVIANO, I. M. L.; GONZALEZ, M. E. Q. (Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 2000. p.69-81. (Coleção CLE, v.30).

GRANGER, R. et al. Unsupervised perceptual learning: a paleocortical model. In: HANSON, S. J.; OLSON, C. R. (Ed.). *Connectionist modeling and brain function: the developing interface*. Massachusetts: The MIT Press, 1990. cap.5, p.105-131.

HANSON, S. J.; BURR, D. What connectionist models learn: learning and representation in connectionist networks. *Behavior and Brain Sciences*, 1990. v.13, p.1-54 apud HONAVAR, V. Symbolic AI and numeric ANN: toward a resolution of the dichotomy. In: SUN, R.; BOOKMAN, L. (Ed.). *Computational*

architectures integrating symbolic and neural process. New York: Kluwer, 1995. p.20.

HANSON, S. J.; OLSON, C. R. Introduction: connectionism and neuroscience. In: _____. (Ed.). *Connectionist modeling and brain function: the developing interface*. Massachusetts: The MIT Press, 1990. cap.1, p.1-4.

HAYKIN, S. *Neural networks: a comprehensive foundation*. 2.ed. New Jersey: Prentice-Hall, 1999.

HEBB, D. *Organization of behavior*. New York: John Wiley, 1949.

HONAVAR, V. Symbolic AI and numeric ANN: toward a resolution of the dichotomy. In: SUN, R.; BOOKMAN, L. (Ed.). *Computational architectures integrating symbolic and neural process*. New York: Kluwer, 1995. p. 351-388.

HOPFIELD, J. J. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, v.79, p.2554-2558, 1982.

IUSEM, A. $P = NP$ ou as sutilezas da complexidade computacional. *Matemática Universitária*, Rio de Janeiro, n.5, p.33-60, jun. 1987.

KLEENE, S. C. Representation of events in nerve nets and finite automata. In: SHANNON, C. E.; MCCARTHY, J. (Ed.). *Automata studies*. New Jersey: Princeton University Press, 1956. p.3-42.

KOHONEN, T. *Self-organizing and associative memory*. Berlim: Springer-Verlag, 1987.

LEITE, M. Aprendizado e exercício produzem novos neurônios. *Folha de São Paulo*, São Paulo, 23 fev. 1999, Folha Mundo, p.12.

LINSKER, R. Self-organization in a perceptual system: how network models and information theory may shed light on neural organization. In: HANSON, S. J.; OLSON, C. R. (Ed.). *Connectionist modeling and brain function: the developing interface*. Massachusetts: The MIT Press, 1990. cap.10, p.351-392.

LIPPMANN, R. An introduction to computing with neural nets. *IEEE ASSP Magazine*, v.4, p.4-22, 1987.

LISTA COGNITIVA. Lista eletrônica de discussão. Site Yahoo Groups. Criador: R. C. Benante. Disponível em: <listacognitiva@yahoogrupos.com.br>. Acesso em: 6 ago. 2001.

LUCCHESI, C. L. et al. *Aspectos teóricos da computação*. Rio de Janeiro: IMPA, 1979.

LUDERMIR, T. B. Computability of logical neural networks. *Journal of Intelligent Systems*, v.2, n.1-4, p.261-289, 1992.

LYCAN, W. G. Homuncular functionalism meets PDP. In: RAMSEY, W. M.; STICH, S. P.; RUMELHART, D. E. (Ed.). *Philosophy and connectionist theory*. New Jersey: Lawrence Erlbaum Associates, 1991. cap.11, p.259-286.

MANNA, Z. *Mathematical theory of computation*. New York: McGraw-Hill, 1974.

MANZOLLI, J.; GONZALEZ, M. E. Q.; VERSHURE, P. Auto-organização, criatividade e cognição. In: D'OTTAVIANO, I. M. L.; GONZALEZ, M. E. Q.

(Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 2000. p.105-125. (Coleção CLE, v.30).

MARR, D. *Vision*. San Francisco: Freeman, 1982.

_____. Artificial intelligence: a personal view. In: BODEN, M. A. (Ed.). *The philosophy of artificial intelligence*. New York: Oxford University Press, 1990. cap.6, p.133-146.

MATURANA, H.; VARELA, F. Autopoiesis and cognition: the realization of the living. *Boston Studies in the Philosophy of Science*, Boston: Reidel, n.42, 1980.

MCCLELLAND, J. L.; RUMELHART, D. E. *Parallel distributed processing: explorations in the microstructure of cognition*. 3.ed. Massachusetts: The MIT Press, 1987. v.2: psychological and biological models.

MCCULLOCH, W.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, v.5, p.115-133, 1943.

MEDLER, D. A. A brief history of connectionism. *Neural Computing Surveys*, v.1, p.61-101, 1998. Disponível em:
<<http://www.icsi.berkeley.edu/~jagota/NCS>>. Acesso em: 20 maio 2000.

MINSKY, M. L.; PAPERT, S. *Perceptrons*. Cambridge: The MIT Press, 1988.

NEWELL, A.; SIMON, H. A. The simulation of human thought. *Current Trends in Psychological Theory*, n.1, p.152-79, 1961.

_____. GPS: a program that simulates human thought. In: FEIGENBAUM, E. A.; FELDMAN, J. (Ed.). *Computers and thought*. New York: McGraw-Hill, 1963. p.279-293.

NIST - NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY.
Dictionary of algorithms, data structures, and problems, jul. 2001. Disponível
em: <<http://hissa.nist.gov/dads/terms.html>>. Acesso em: 20 jul. 2001.

OLIVEIRA, M. B. O que todo cientista cognitivo deve saber sobre lógica. In:
GONZALEZ, M. E. Q. et al. (Org.). *Encontro com as ciências cognitivas*.
Marília: Unesp-Marília-Publicações, 1997. v.1, p.3-12.

PESSOA JÚNIOR, O. Medidas sistêmicas e organização. In: DEBRUN, M.;
GONZALEZ, M. E. Q.; PESSOA JÚNIOR, O. (Org.). *Auto-organização: estudos*
interdisciplinares. Campinas: CLE/UNICAMP, 1996. p.129-161. (Coleção CLE,
v.18).

PEREIRA JÚNIOR, A.; HAASE, V. G. Introdução à filosofia da neurociência
cognitiva. In: GONZALEZ, M. E. Q. et al. (Org.). *Encontro com as ciências*
cognitivas. Marília: Unesp-Marília-Publicações, 1997. v.1. p.143-156.

POLLACK, J. *On connectionist models of natural language processing*. 1987.
Tese (Ph. D.) - Departamento de Ciência da Computação, Universidade de
Illinois, Illinois apud SIEGELMANN, H. T.; SONTAG, E. D. *Turing*
computability with neural nets. Great Britain: Pergamon Press, 1991. p.1.

POLYA, G. *A arte de resolver problemas: um novo aspecto do método*
matemático. Rio de Janeiro: Interciência, 1978.

PUTNAM, H. The mental lives of some machines. In: CANTANEDA, H. (Org.).
Intentionality, minds and perception. Detroit: Wayne State University Press,
1967.



PYLYSHYN, Z. W. *Computation and cognition: toward a foundation for cognitive science*. Cambridge: Cambridge Massachusetts Press, 1984.

RICH, E.; KNIGHT, K. *Inteligência artificial*. 2.ed. rev. e aum. São Paulo: Makron Books, 1993.

ROCHA, A. F. *O cérebro: um breve relato de sua função*. São Paulo: [s.n.], 1999.

ROSENBLATT, F. *Principles of neurodynamics*. New York: Spartan, 1962.

RUMELHART, D. E.; MCCLELLAND, J. L. Levels indeed! A response to Broadbent. *Journal of Experimental Psychology*. v.114, p.189-192, 1985.

_____. *Parallel distributed processing: explorations in the microstructure of cognition*. 9.ed. Massachusetts: The MIT Press, 1989. v.1: foundations.

SCHAEFFER, J.; PLAAT, A. Kasparov versus Deep Blue: the re-mach. *ICCA Journal*, v.20, n.2, p.95-102, 1997.

SEARLE, J. R. *A redescoberta da mente*. São Paulo: Martins Fontes, 1997.

SEJNOWSKI, T. J.; KOCH, C.; CHURCHLAND, P. S. Computational neuroscience. In: HANSON, S. J.; OLSON, C. R. (Ed.). *Connectionist modeling and brain function: the developing interface*. Massachusetts: The MIT Press, 1990. cap.2 , p.5-35.

SHANNON, C. E. A universal Turing machine with two internal states. In: SHANNON, C. E.; MCCARTHY, J. (Ed.). *Automata studies*. New Jersey: Princeton University Press, 1956. p.157-165.

SIEGELMANN, H. T.; SONTAG, E. D. *Turing computability with neural nets*. Great Britain: Pergamon Press, 1991. p.1-15.

SILVEIRA, L. F. B. da. Textos significativos de Peirce para se pensar a questão da máquina lógica. In: GONZALEZ, M. E. Q. et al. (Org.). *Encontro com as ciências cognitivas*. Marília: Unesp-Marília-Publicações, 1997. v.1, p.125-140.

_____. A origem do cosmos e auto-organização na obra de Charles Sanders Peirce. In: D'OTTAVIANO, I. M. L.; GONZALEZ, M. E. Q. (Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 2000. p.325-336. (Coleção CLE, v.30).

SIMON, I. *Introdução à teoria de complexidade de algoritmos*. São Paulo: IME/USP, 1979.

SMOLENSKY, P. On the proper treatment of connectionism. *The Behavioral and Brain Sciences*, v.11, p.1-23, 1988.

SOARES, A. B. Modelos cognitivos de aprendizagem. In: GONZALEZ, M. E. Q. et al. (Org.). *Encontro com as ciências cognitivas*. Marília: Unesp-Marília-Publicações, 1997. v.1, p.47-62.

SOUZA, G. M.; MANZATTO, A. G. Hierarquia auto-organizada em sistemas biológicos. In: D'OTTAVIANO, I. M. L.; GONZALEZ, M. E. Q. (Org.). *Auto-organização: estudos interdisciplinares*. Campinas: CLE/UNICAMP, 2000. p.153-173. (Coleção CLE, v.30).

STEVENS, C. F. Two principles of brain organization: a challenge for artificial neural networks. In: GARDNER, D. (Ed.). *The neurobiology of neural networks*. Massachusetts: The MIT Press, 1993. cap.2, p.13-20.

TENÓRIO, R. *Computadores de papel: máquinas abstratas para um ensino concreto*. 2.ed. São Paulo: Cortez, 2001. (Coleção Questões da Nossa Época, v.80).

THAGARD, P. *Mente: introdução à ciência cognitiva*. Porto Alegre: Artmed, 1998. p.49-62, 109-127.

TURING, A. M. On computable numbers with an application to the entscheidungsproblem. *Proceedings of the London Mathematical Society*, ser.2, v.42, p.230-265, 1936, supl. (1937) v.43, p.544-546.

_____. Computing machinery and intelligence. In: BODEN, M. A. (Ed.). *The philosophy of artificial intelligence*. New York: Oxford University Press, 1990. cap.2, p.40-66.

VAN GELDER, T. What is the "D" in "PDP"? A survey of the concept of distribution. In: RAMSEY, W.; STICH, S. P.; RUMELHART, D. E. (Ed.). *Philosophy and connectionist theory*. New Jersey: Lawrence Erlbaum Associates, 1991. p.33-59.

_____. Dynamics and cognition. In: HAUGELAND, J. (Ed.). *Mind design II*. 2.ed. London: The MIT press, 1997.

WIENER, N. *Cibernética e sociedade: o uso humano de seres humanos*. 4.ed., São Paulo: Cultrix, 1954.

WINSTON, P. H. *Artificial intelligence*. 2.ed. Reading, Massachusetts: Addison-Wesley, 1984.

WITTGENSTEIN, L. *Investigações Filosóficas*. São Paulo: Nova Cultural, 1999. (Coleção os Pensadores).

ZAMARREÑO, J. M.; VEGA, P. State space neural network: properties and application. *Neural Networks*, n.11, p.1099-112, 1998.

ZIPSER, D.; RABIN, D. E. P3: a parallel network simulating system. In:
RUMELHART, D. E.; MCCLELLAND, J. L. *Parallel distributed processing:
explorations in the microstructure of cognition*. 9.ed. Massachusetts: The MIT
Press, 1989. v.1: foundations, p.488-506.

Resumo

BENANTE, R. C. *Reflexões sobre os alcances das Redes Neurais Artificiais*. Marília, 2001. 239f. Dissertação (Mestrado em Filosofia) – Faculdade de Filosofia e Ciências, Campus de Marília, Universidade Estadual Paulista “Júlio de Mesquita Filho”.

Este trabalho tem por finalidade discutir até que ponto as redes neurais artificiais criaram um novo paradigma computacional, traçando um paralelo entre seu funcionamento, tanto no plano físico, implementacional, como no plano algorítmico, isto é, da modelagem, utilizando as máquinas de Turing. Para uma discussão do plano físico, apresentaremos, como método de trabalho, as definições de máquinas de Turing e seu alcance, e colocaremos em contraste as propriedades das máquinas conexionistas que são invocadas para sua defesa. Num segundo momento discutiremos os modelos de RNA, como diferem dos modelos da IA, e o que computam tais modelos. Em especial, estamos interessados em descobrir se esses modelos de RNA computam funções que não são passíveis de serem computadas por uma máquina de Turing. Finalmente, introduziremos a visão da Teoria de Auto-organização para avaliarmos até que ponto os modelos de RNA podem ser ditos auto-organizáveis. Ainda, se esses modelos oferecerem a

possibilidade de serem simulados em máquinas de Turing, discutiremos se a auto-organização se encaixa neste molde.

Palavras-chave: máquinas de Turing; computabilidade; redes neurais artificiais; inteligência artificial; máquinas connexionistas; auto-organização; complexidade de algoritmos.

Keywords: Turing machines; computability; artificial neural networks; artificial intelligence; connectionist machines; self-organization; algorithmic complexity.

Abstract

This dissertation aims to discuss the role of artificial neural networks and to examine whether they create a new computational paradigm. In order to do so, we investigate the functionality of artificial neural networks (ANN) on the physical level, the implementational level and also on the algorithmic level. We present the notion of a universal Turing machine and its computational power, and we relate this to the properties of ANN. Then, we discuss 5 ANN models (McCullouch & Pitts, 1- and multi-layer perceptrons, Hopfield nets and Kohonen nets), analyze how they differ from the models of AI, and investigate their computational abilities. We are especially interested in comparing these ANN models with the computational power of a universal Turing machine. Finally, we introduce the theory of self-organization. We evaluate to which degree ANN models can be said to be self-organized. Since many ANN's are being simulated on Turing Machines (or Von Neumann Architectures), we will discuss whether the property of self-organization is still possessed by these Turing Machine simulations of ANN's.

Keywords: Turing machines; computability; artificial neural networks; artificial intelligence; connectionist machines; self-organization; algorithm complexity.

