

**ANALÚCIA SCHIAFFINO MORALES DE FRANCESCHI**

**DESENVOLVIMENTO DE AGENTES AUTÔNOMOS PARA  
GERÊNCIA DE REDES DE COMPUTADORES**

**FLORIANÓPOLIS**  
1999

**UNIVERSIDADE FEDERAL DE SANTA CATARINA**  
**PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA**

**DESENVOLVIMENTO DE AGENTES AUTÔNOMOS**  
**PARA GERÊNCIA DE REDES DE**  
**COMPUTADORES**

Exame de Qualificação de Doutorado submetido à  
Universidade Federal de Santa Catarina

**ANALÚCIA SCHIAFFINO MORALES DE FRANCESCHI**

Florianópolis, agosto de 01.

# DESENVOLVIMENTO DE AGENTES AUTÔNOMOS PARA GERÊNCIA DE REDES DE COMPUTADORES

**Analúcia Schiaffino Morales De Franceschi**

Dezembro/1999

Orientador: Jorge Muniz Barreto.

Área de Concentração: Sistemas de Informação.

Palavras-chave: sistemas dinâmicos, agentes autônomos, redes recorrente, sistema de raciocínio distribuído, complexidade de problemas distribuídos.

Número de Páginas: XXX.

Avançam o uso de novas técnicas para resolução de problemas complexos, tais como agentes autônomos, redes neurais artificiais e computação evolucionária. No entanto, existem poucos trabalhos voltados ao estudo da complexidade de problemas solucionados por tais sistemas, que permitam uma escolha adequada da técnica de resolução de problema. O primeiro grande trabalho no sentido de criar uma teoria da complexidade para redes neurais artificiais foi o de Minsky e Papert, que classificaram os problemas em linearmente separáveis e, não linearmente separáveis. Sem dúvida uma grande contribuição, a partir dela foi possível saber se a rede neural a ser utilizada deverá possuir ou não camada de neurônios intermediária, além da camada de entrada e de saída. Outra questão importante, pertencente a teoria da complexidade conexionista e um dos problemas mais frequentes encontrados em trabalhos sobre redes neurais, têm sido o uso de redes neurais diretas (também conhecidas como *feedforward*) com neurônios estáticos para solucionar problemas dinâmicos.

A essência deste trabalho é explorar soluções para problemas distribuídos de gerência de redes aplicando técnicas de IA (AAs e RNAs recorrentes). Estas técnicas serão implementadas na forma de agentes autônomos para auxiliar no processo de automação da gerência de redes de computadores. A gerência de redes é composta por agentes e gerentes passivos. Ou seja, não existe autonomia nenhuma. A tomada de decisões deve ser realizada por um administrador humano.

Analisando a gerência de redes como um todo tem-se um problema extremamente complexo. No entanto, é possível subdividi-la em cinco áreas funcionais utilizando o modelo funcional proposto pela OSI. Os problemas foram então, investigados conforme suas características estáticas ou dinâmicas. Além disso, foi também considerado o tipo de comportamento do gerenciamento, reativo ou pró-ativo.

O presente trabalho defende então, duas maneiras para solucionar os problemas de gerência de redes. Se o problema possui características estáticas devem ser utilizadas soluções com estas características, chamam-se heurísticas para a solução do problema. As duas ferramentas mais conhecidas para solucionar este tipo de problema são as regras de produção (paradigma simbólico da IA) ou através de redes neurais diretas (paradigma conexionista da IA). Ao contrário, os problemas com características dinâmicas podem ser “bem” solucionados apenas por ferramentas dinâmicas. Conhece-se até o presente momento, três formas de associar dinamismo a uma rede neural artificial: aplicando uma seqüência de retardos à rede, utilizando redes de retroação (com ciclos) ou através do uso de neurônios dinâmicos. Propõe-se então, o uso de exemplos para alcançar às soluções destes tipos de problemas. Quando os valores de entrada e saída são conhecidos, utiliza-se uma rede neural recorrente para estimar as trocas de estado do sistema através do treinamento da rede.

Através desta metodologia pretende-se desenvolver soluções adequadas que respeitem as qualidades de cada área funcional da gerência de redes. Estas soluções tomarão a forma de agentes autônomos para auxiliar o processo de automação da área de gerência de redes de computadores.

Abstract of Thesis presented to UFSC as a partial fulfillment of the requirements for the degree of Doctor in Electrical Engineering.

## **DEVELOPING AUTONOMOUS AGENTS TO COMPUTER NETWORK MANAGEMENT**

**Analúcia Schiaffino Morales De Franceschi**

December/1999

Advisor: Jorge Muniz Barreto.

Area of Concentration: Information Systems.

Keywords: dynamic systems, autonomous agents, recurrent networks, distributed reasoning system, distributed problem complexity.

Number of Pages: XXX.

Application of new techniques to complex problem solving such as autonomous agents, artificial neural networks (ANN) and evolutionary computation have been growing up. With these new tools new question arrives and, the most important is: "can new problems be solved? With how much effort?" This is particularly important with neural networks where a new computer paradigm is involved and the construction of a connectionist computability and complexity theory must be accomplished. Neural computability was treated initially by McCulloch & Pitts using logic. They proved the equivalence of a neural network with input devices and a Turing Machine. After, Arbib proposed an intuitive demonstration of this equivalence. However, in the complexity field, the two approaches are different because they require different resources. Minsky and Papert provided the first contribution to such theory when they proved that a feed forward ANN must have a hidden layer to solve a non-linearly separable problem. Another result is that to solve a dynamical problem a recurrent dynamical ANN is simpler than a feed forward one. In the scientific literature, it is usual to find dynamic problems solved by static feed forward neural networks. Using this approach, explicitly or implicitly the state of the dynamical system must be supplied, leading to a very big neural network and a corresponding longer training time even if the famous back propagation is used.

The essence of this work is investigating the right distributed problem solving of the computer network management applying AI techniques (Autonomous agents, feedforward and recurrent neural networks). The AI techniques will be implemented as autonomous agents format which will be able to automating the network management process. Passive managers and agents normally compose the network management. They have no autonomy and network administrator must reach the decision make.

Analyzing the network management process as a whole we have a big complex problem. Moreover, using the functionality division of the OSI reference model it turns five smaller problems. These problems were investigated if they have static or dynamic features. In addition, the network management behavior also was considered, if it was reactive or proactive.

Following methodology is defended in the present work. If is a static problem then it must have a heuristic solution. There are two forms to implement this case, through production rules (Symbolic paradigm) or using feedforward neural networks (Connectionist paradigm). On the contrary, the dynamic problems must be “well” solved by dynamic tools. There are following forms presently to include dynamism in a neural network solution. To apply a sequential line of time delays between two inputs of a feedforward neural network, or using a network with cycles and dynamical neurons (ex: Hopfield network and recurrent neural networks). So, the example usage is proposed. If the input and output of neural network were known we may use a recurrent neural network which may be trained estimating the unknown state changes.

Therefore, applying this methodology the autonomous agents for network management must be implemented. These solutions will compose a new automation process for the network management.

# Publicações

- [1] A.S.M. De Franceschi, J.M. Barreto. *Autonomous Agents based on Recurrent Neural Networks Applied to Computer Network Management*, ISAS'99 – 5<sup>th</sup> International Conference on Information Systems, Analysis and Synthesis, Orlando, Florida, 31/07-04/08/1999.
  
- [2] A.S.M. De Franceschi, J.M. Barreto. *Distributed Problem Solving Based on Recurrent Neural Networks Applied to Computer Network Management*, ICT'99 IEEE/International Conference on Telecommunications, Cheju, Korea, 15/06-18/06/1999.

# Sumário

Publicações.....	vii
Sumário.....	viii
Lista de Figuras.....	xi
Lista de Tabelas.....	xii
Lista de Siglas.....	xiii
<b>Capítulo 1 Introdução.....</b>	<b>15</b>
1.1 Motivação.....	15
1.2 Objetivos.....	19
1.2.1 <i>Objetivo Principal</i> .....	19
1.2.2 <i>Objetivos Secundários</i> .....	20
1.3 Organização do Trabalho.....	21
<b>Capítulo 2 Gerência de Redes.....</b>	<b>23</b>
2.1 Gerência OSI.....	25
2.2 Áreas Funcionais.....	25
2.2.1 <i>Gerência de Falhas</i> .....	25
2.2.2 <i>Gerência de Configuração</i> .....	26
2.2.3 <i>Gerência de Desempenho</i> .....	26
2.2.4 <i>Gerência de Contabilização</i> .....	27
2.2.5 <i>Gerência de Segurança</i> .....	27
2.3 Gerência Internet.....	27
2.3.1 <i>Protocolo SNMP</i> .....	28
2.3.2 <i>SNMPv2</i> .....	28
2.3.3 <i>Elementos da arquitetura SNMP</i> .....	29
2.4 Gerência de Telecomunicações.....	30
<b>Capítulo 3 Redes Neurais Artificiais (RNAs).....</b>	<b>31</b>
3.1 Modelos de neurônio.....	32
3.2 Topologias das RNAs.....	34
3.3 Algoritmos de treinamento.....	35
3.3.1 <i>Supervisionado</i> .....	35
3.3.2 <i>Não Supervisionado</i> .....	35
3.3.3 <i>Por Reforço</i> .....	36
3.4 Redes Diretas X Redes Recorrentes.....	36
3.5 Redes Recorrentes.....	37



3.5.1	<i>Aprendizado em RNAs Recorrentes</i> .....	38
3.6	Teoria de Problemas.....	39
3.6.1	<i>Problemas estáticos linearmente separáveis</i> .....	40
3.6.2	<i>Problemas dinâmicos com dinâmica finita</i> .....	40
3.6.3	<i>Problemas dinâmicos com dinâmica infinita</i> .....	40
3.7	O que são Problemas Distribuídos.....	41
3.7.1	<i>Problemas comuns em redes de computadores</i> .....	42
3.8	Computabilidade e Complexidade de RNAs.....	43
3.8.1	<i>Confluência</i> .....	44
<b>Capítulo 4</b>	<b>Teoria de Sistemas</b> .....	<b>46</b>
4.1	Sistema Geral.....	47
4.1.1	<i>Exemplos de Sistema Geral</i> .....	47
4.2	Sistema Orientado.....	48
4.2.1	<i>Exemplos de Sistema Orientado</i> .....	49
4.3	Sistema Temporal.....	50
4.3.1	<i>Exemplos de Sistema Temporal</i> .....	50
4.4	Sistema Funcional.....	51
4.4.1	<i>Exemplos de Sistema Funcional</i> .....	52
4.5	Sistema Dinâmico.....	53
4.5.1	<i>Exemplos de Sistema Dinâmico</i> .....	54
4.5.2	<i>Modelo Geral de Neurônio como Exemplo de Sistema Dinâmico</i> .....	55
4.6	Sistema Complexo.....	55
4.6.1	<i>Exemplos de Sistema Complexo</i> .....	56
<b>Capítulo 5</b>	<b>Inteligência Artificial Distribuída (IAD)</b> .....	<b>58</b>
5.1	Agentes Autônomos.....	59
5.2	Modelo de Agente.....	60
5.2.1	<i>Modelo Multi-agentes</i> .....	60
5.3	Por que utilizar IAD.....	61
5.4	Sistema de Raciocínio Distribuído (SRD).....	62
5.4.1	<i>Vantagens de um SRD</i> .....	63
5.5	Mecanismos de Coordenação e Controle.....	64
5.5.1	<i>Planejamento de execução multi-agentes</i> .....	64
5.5.2	<i>Planejamento e negociação: redes de contrato</i> .....	64
5.5.3	<i>Planejamento distribuído</i> .....	64
5.6	Mecanismos de Comunicação.....	64
5.6.1	<i>Modelo Quadro Negro</i> .....	65
5.6.2	<i>Sistema de Transmissão de Mensagens</i> .....	65
5.6.3	<i>Linguagem KQML</i> .....	65
5.7	Aplicação da IAD em Gerência de Redes.....	66
<b>Capítulo 6</b>	<b>Agentes Autônomos para Gerência de Redes</b> .....	<b>67</b>
6.1	Metodologia.....	69
6.2	Gerência de Falhas.....	71
6.2.1	<i>Exemplo de Diagnóstico de Falhas</i> .....	72

6.2.2	<i>Exemplos prático</i> .....	74
6.3	Gerência de Desempenho.....	75
6.3.1	<i>Exemplo de Controle de Rotas de Pacotes</i> .....	76
6.4	Gerência de Configuração.....	77
6.4.1	<i>Exemplo de Gerência de Configuração</i> .....	78
6.5	Gerência de Contabilização.....	78
6.6	Gerência de Segurança.....	79
6.7	Classes de Agentes.....	80
6.8	Aspectos de Implementação.....	81
6.8.1	<i>Vantagens de Utilizar JAVA em ambientes distribuídos</i> .....	81
6.9	Alguns Experimentos.....	82
6.9.1	<i>Agente Paridade</i> .....	82
6.9.2	<i>Agente de Falhas</i> .....	84
6.10	Outros AAs para Gerência de Redes.....	85
<b>Capítulo 7 Considerações finais</b> .....		<b>87</b>
7.1	Cronograma.....	90
<b>Referências Bibliográficas</b> .....		<b>91</b>
<b>Glossário</b> .....		<b>97</b>

# Lista de Figuras

Figura 1 – Relacionamento gerente-agentes utilizando o protocolo SNMP.....	30
Figura 2 - (a) Neurônios Biológicos e (b) Neurônios Artificiais.....	32
Figura 3 - O neurônio artificial.....	33
Figura 4 - Classificação de Redes Neurais Artificiais.....	34
Figura 5 - Exemplo de algoritmo competitivo.....	36
Figura 6 - Implementação neural de autômato finito.....	38
Figura 7 - Problemas Distribuídos.....	42
Figura 8 – Problemas linear e não linearmente separáveis.....	44
Figura 9- Exemplo sobre confluência com parábola.....	45
Figura 10 - Teoria de Sistemas.....	46
Figura 11 - O sistema geral visto como uma caixa preta.....	47
Figura 12 - Um processador é um bom exemplo de sistema geral.....	47
Figura 13 - Sistema orientado.....	48
Figura 14 - Computador como exemplo de sistema orientado.....	49
Figura 15 - Sistema temporal.....	50
Figura 16 – Um Sistema Temporal.....	51
Figura 17 - Sistema funcional.....	52
Figura 18 - Exemplo de Sistema Funcional.....	53
Figura 19 - Sistema dinâmico.....	53
Figura 20 - Sistema Complexo.....	56
Figura 21 - Agente Autônomo em seu ambiente.....	60
Figura 22 - Sistema multi-agentes.....	61
Figura 23 - Comportamento da Gerência de Redes.....	68
Figura 24 - Metodologia Híbrida para solução de problemas distribuídos.....	69
Figura 25 - Gerência de falhas.....	71
Figura 26 - Sistema de Diagnóstico de Falhas.....	72
Figura 27 - Exemplo de diagnóstico de falhas.....	75
Figura 28- Gerência de Desempenho.....	76
Figura 29 - Gerência dinâmica da rede.....	77
Figura 30- Gerência de configuração.....	77
Figura 31- Gerência de Contabilização.....	79
Figura 32- Gerência de segurança.....	79
Figura 33 – Duas abordagens diferentes para solucionar o problema da paridade.....	83
Figura 34 – Execução da rede neural com o conceito de paridade.....	84
Figura 35 – Agente para detecção de falhas em uma rede.....	84

# Lista de Tabelas

Tabela I - Principais diferenças entre RNAs Diretas e Recorrentes.....	37
Tabela II - Diferenças entre sistemas distribuídos e monolíticos.....	61
Tabela III – Cronograma para as atividades restantes do trabalho.....	90

## Lista de Siglas

CCITT	<i>Consultative Committee on International Telephony and Telegraphy</i>
CMIP	<i>Common Management Information Protocol</i>
CMISE	<i>Common Management Information Services</i>
CNPq	<i>Conselho Nacional de Pesquisa e Desenvolvimento Tecnológico</i>
ComSoc	<i>Communication Society</i>
CTC	<i>Centro Tecnológico</i>
DCE	<i>Data Communications Equipment</i>
DME	<i>Distributed Management Environment</i>
DTE	<i>Data Terminal Equipment</i>
IEC	<i>International Electrotechnical Committee</i>
IEE	<i>The Institution of Electrical Engineers</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
IETF	<i>International Engineering Task Force</i>
IP	<i>Internet Protocol</i>
ISO	<i>International Organization for Standardization</i>
ITU-T	<i>International Telecommunications Union</i>
EEL	<i>Departamento de Engenharia Elétrica</i>
EMA	<i>Enterprise Management Architecture</i>
LAN	<i>Local Area Networks</i>
MIB	<i>Management Information Base</i>
NIST	<i>National Institute of Standards and Technology</i>
NPD	<i>Núcleo de Processamento de Dados</i>
OMNIPoints	<i>Open Management Interoperability Points</i>
OSF	<i>Open Software Foundation</i>
OSI	<i>Open Systems Interconnection</i>
PDU	<i>Protocol Data Unit</i>
ProTeM-CC	<i>Programa Temático Multi-institucional – Ciência da Computação</i>
RNAs	<i>Redes Neurais Artificiais</i>
SNMP	<i>Simple Network Management Protocol</i>
TCP	<i>Transport Control Protocol</i>
TMN	<i>Telecommunications Management Network</i>
UDP	<i>User Datagram Protocol</i>
UFMG	<i>Universidade Federal de Minas Gerais</i>

UFSC            *Universidade Federal de Santa Catarina*  
UNMA           *Universal Network Management Architecture*  
WAN            *Wide Area Networks*

# Capítulo 1

## Introdução

### 1.1 Motivação

“A primeira visão de Cris pela manhã foram pelo menos uns cem recados, no seu terminal uma mensagem piscava fora de controle e haviam várias pessoas rondando o laboratório. Em coro, gritaram: “A rede está fora do ar!”. Estava começando mais um dia na vida de um administrador de rede. Rapidamente, começou a pressionar as teclas de seu terminal. Chicago, Singapura, Nova Iorque, São Francisco e Paris, não havia acesso nenhum. Os principais acessos da *MegaNet Company* pareciam ter desaparecido. Cris suspirou e por um instante gostaria de apertar um botão sobre a parede dizendo: “Não entrem em pânico!”. No setor de atendimento a clientes, ouvia-se uma voz ao fundo: “Desculpe pode ligar mais tarde, a rede está fora do ar...”. À direita, o setor de cobrança aguardava para entrar os dados da empresa. O pessoal do departamento de pesquisa e desenvolvimento foram dispensados e escalonados para trabalhar após às 17 horas. Apenas Cris e os gerentes ocupavam o escritório. Cris sabia exatamente como seu dia iria proceder: isolar o problema, solucionar o problema e imprimir relatórios e gráficos de gerência mostrando o que aconteceu e por quê, para não se repetir.”

Esta estória foi retirada de [41] ilustrando um cenário familiar aqueles que trabalham ou utilizam as redes de computadores. As redes vêm se tornando muito comuns no nosso dia-a-dia. É o caso dos serviços dos bancos que podem ser acessados de casa ou de um terminal 24 horas. Compras com cartões de crédito podem ser aprovadas

computador pessoal e viajar através da Internet para qualquer lugar no mundo, acessando de forma rápida e eficiente uma vasta quantidade de informação. No entanto, estamos tão dependentes destas redes que no caso de uma queda de acesso, podem ser perdidos importantes negócios, pode haver atrasos no recebimento de dados importantes e até mesmo, inacessibilidade a sua própria conta corrente.

As redes mais simples são formadas por quatro tipos de hardware, basicamente: terminais ou DTEs, dispositivos de transmissão também conhecidos como DCEs – como por exemplo, modems; linhas de telefone e uma unidade de processamento ou CPU. Além disso, sem o software necessário para cada equipamento não teremos uma rede. Existem ainda centenas de palavras que fazem parte do jargão relacionado às redes maiores, tais como, *gateways*, *bridges*, *hosts*, barramento *Ethernet*, *Token Ring*, *X.25*, etc. Além de protocolos de gerência, como SNMP, CMIP e TMN; protocolos de transmissão, como TCP, UDP e IP e milhares de aplicações.

As redes podem ser subdivididas em LANs e WANs. As LANs são as redes locais que podem ser interligadas umas com as outras formando uma grande rede local como é feito na universidade. A rede da EEL, a rede do CTC, a do NPD, etc., interligadas formam a rede local da UFSC. E as WANs são as redes de longa distância, por exemplo a UFSC tem linhas dedicadas que conectam a rede local da UFSC com outras redes fora deste domínio, uma com o Rio Grande do Sul, outra com o Rio de Janeiro, etc. Existem ainda as redes de Telecomunicações, que compreendem “fax” e telefones, transmitindo voz e dados, e telefones celulares analógicos e digitais.

Provavelmente, qualquer um dos leitores deste trabalho já tenha tido contato com algum tipo de falha durante a operação de alguma rede. As redes portanto, são equipamentos que operam de forma distribuída e estão sujeitos a diversos tipos de falhas: - Falhas nos equipamentos devido a ação do tempo, como umidade e calor excessivo; - Falhas operacionais ou de uso indevido dos equipamentos; - Falha de sobrecarga; etc. Além de falhas existem outras preocupações para aqueles que utilizam e administram seus serviços. A configuração, o desempenho, a contabilização e a segurança são outros aspectos funcionais sobre redes que constituem a Gerência de Redes. Gerência de Redes é o processo



para controlar uma rede de dados complexa (pode ser de computadores ou de telecomunicações) para aumentar a sua eficiência e garantir a sua produtividade [57].

A Gerência de Redes de Computadores por um longo período foi caracterizada como proprietária, desenvolvida por cada fabricante. São exemplos de sistemas de gerência o *SunNet Manager* (da *SunConnect*), o *NetView 6000* (da *IBM*) e o *OpenView* (da *HP Hewlett-Packard*). A *AT&T* possui seu próprio pacote de produtos e protocolos chamado UNMA, a *Digital Equipment Corporation* possui uma arquitetura proprietária chamada EMA e seu sistema de gerência chama-se *DECmcc Director* [71].

Na verdade os protocolos de gerência disponíveis não satisfaziam todos os fabricantes. O SNMP é um protocolo de pergunta/resposta (*request/reply*) muito simples. Foi introduzido no final dos anos 80 para controlar e monitorar redes TCP/IP. Devido a sua simplicidade para implementação e baixos custos, agentes baseados neste protocolo foram implementados por diversos fabricantes de redes. As aplicações que não exigiam notificações de falhas em tempo-real foram eficientemente implementadas no SNMP. No sentido de melhorar a segurança dos endereçamentos e aperfeiçoar o protocolo SNMP foi lançada a versão 2. O SNMPv2, como é chamado, suporta entre outras novidades, comunicação de gerente-gerente e recuperação de um bloco de dados (na outra versão, era feito linha a linha) [41][57][64].

No sentido de padronizar o processo de gerência a ISO lançou o Modelo de Referência OSI. Este modelo incorpora o modelo estrutural, o modelo informacional e o modelo funcional. O protocolo de gerência CMIP/CMISE que faz parte do modelo informacional, determina como o processo gerente deve invocar as operações de gerência e como o sistema gerenciado envia as notificações ao gerente.

Entre as principais diferenças entre o CMIP e o SNMP destacam-se [64][71].

- CMIP define um conjunto de mensagens muito grandes, ao contrário do SNMP que é simplificado;
- o CMIP distribui a carga de tráfego gerada por gerentes e agentes;
- as mensagens do CMIP contém informações sobre os parâmetros e identificação dos objetos gerenciados.

Ou seja, é um protocolo com mais recursos de gerência, mais caro e mais difícil de ser implementado. O SNMP, por outro lado, é muito mais simples e é o mais utilizado.

Em meados dos anos 90 iniciou-se um processo para distribuir a gerência em ambientes heterogêneos. O objetivo de integrar estas redes é criar a imagem de uma rede virtual única, conhecida como arquitetura aberta, formada por componentes de diversos fabricantes, para facilitar o processo de gerência. Vários órgãos de regulamentação reuniram-se com o intuito de criar uma regulamentação para as redes interoperáveis, tais como: IETF, OSF, NIST, OMNI*Points*, etc. O DME é um conjunto de especificações para produtos de gerência de redes distribuída. O DME está baseado no CMIP e no SNMP, e é orientado-a-objetos [57][71].

No Brasil, segundo artigo do professor da UFMG, José Marcos Nogueira [54] a pesquisa e desenvolvimento na área de Gerência de Redes de Computadores e Telecomunicações vêm sendo realizada, na sua grande maioria, por departamentos e instituições universitárias (na maioria públicas). Alguns grupos de instituições associam-se a empresas particulares ou estatais buscando solução de problemas práticos. Existem ainda grupos multi-institucionais financiados por agências públicas que trabalham em projetos temáticos. Um destes projetos, citado em [54] pelo professor Nogueira foi o Projeto PlaGeRe – Plataformas para Gerência de Redes – financiado pelo CNPq no Programa ProTeM-CC.

Um dos principais motivos para abordar esse assunto vem da experiência obtida durante o mestrado [22] e durante os doze meses de participação no projeto mencionado. Durante este período, foi possível desenvolver um trabalho junto aos alunos de graduação como co-orientadora de projetos de conclusão de curso [2][30][66]. Além disso, a oportunidade de publicar resultados intermediários e finais em congressos internacionais [25][26][27][28]. Dois destes trabalhos foram apresentados durante a participação no Projeto PLAGERE, um em Toronto, no Canadá; e outro em Dallas, nos Estados Unidos.

Outro incentivo a realização deste trabalho são as duas publicações aceitas em eventos internacionais relevantes. Um trabalho publicado na *World Multiconference on Information Systems, Analysis and Synthesis and, Systemics, Cybernetics and Informatics - ISAS/SCT'99* realizada em Julho/99 em Orlando, Flórida. E outro publicado nos

*Proceedings of International Conference on Telecommunications - ICT '99*, financiado pela IEEE, ComSoc, IEE e *King's College London*, realizado em Junho/99 em Cheju, Korea.

## 1.2 Objetivos

No entanto, o processo de gerência de redes não é automatizado por completo, isto é, necessita ainda da intervenção humana na tomada de decisões. Os sistemas de gerência disponíveis no mercado, citados no item anterior, são ferramentas que auxiliam o processo de gerência, fornecendo relatórios e emitindo alarmes após alguma falha ser detectada. São sistemas passivos, não executam nenhum tipo de tomada de decisões aguardando que o administrador da rede solucione o problema.

Existem duas formas de aplicar as técnicas de Inteligência Artificial aos sistemas de gerência. É possível, baseado em heurísticas de administradores de redes, criar regras de produção ou utilizar RNAs diretas para desenvolver sistemas estáticos [25][26]. Ou utilizando exemplos, em que as entradas e saídas são fornecidas a uma RNA recorrente responsável por estimar as trocas de estados do sistema (abordagem dinâmica) [23][24].

### *1.2.1 Objetivo Principal*

O principal objetivo deste trabalho é, portanto, explorar as aplicações adequadas das técnicas de inteligência artificial para automatizar o processo de gerência de redes. Desta forma propõe-se uma metodologia para o desenvolvimento de agentes autônomos respeitando as características dos problemas e o comportamento do gerenciamento. Os problemas de gerência foram classificados como estáticos ou dinâmicos. E, o comportamento da gerência de redes considerado como reativo (quando a tomada de decisões é feita após o problema ter acontecido) ou pró-ativo (comportamento preventivo).

Soluções para problemas estáticos chamam-se heurísticas para a solução do problema. As duas ferramentas mais conhecidas para solucionar este tipo de problema são as regras de produção (paradigma simbólico da IA) ou através de redes neurais diretas (paradigma conexionista da IA). Ao contrário, quando os problemas possuem caráter

dinâmico devem ser solucionados por ferramentas dinâmicas. Conhece-se até o presente momento, três formas de associar dinamismo a uma rede neural artificial: aplicando uma seqüência de retardos à rede, utilizando redes de retroação (com ciclos) ou através do uso de neurônios dinâmicos. Propõe-se então, o uso de exemplos para alcançar às soluções destes tipos de problemas. Quando os valores de entrada e saída são conhecidos, utiliza-se uma rede neural recorrente para estimar as trocas de estado do sistema através do treinamento da rede.

### *1.2.2 Objetivos Secundários*

Normalmente, os profissionais que trabalham na área de gerência de redes não dispõem de tempo para pesquisar as técnicas de IA para automatizar o processo. Em muitos casos acabam por aplicar a ferramenta incorreta na solução de um determinado problema. Neste sentido, este trabalho tem como um segundo objetivo fornecer uma espécie de manual para o desenvolvimento de agentes com características inteligentes para gerência de redes.

Para alcançar o objetivo principal do trabalho várias etapas serão necessárias, entre elas destacam-se:

- Estudo de problemas em redes locais, o ambiente para coleta dessas informações deverá ser a rede local do GPEB;
- Coletar exemplos de problemas/soluções mais freqüentes no ambiente de testes;
- Desenvolver uma maneira de automatizar o processo para a criação de agentes;
- Definir os agentes que deverão ser implementados;
- Desenvolver os agentes através de redes neurais recorrentes;
- Implementá-los em JAVA e C e testá-los;
- Observar e analisar o comportamento dos agentes;
- Definir conceitos para a solução dos problemas distribuídos.

## 1.3 Organização do Trabalho

O presente trabalho está organizado em sete capítulos. Seguem a introdução disposta no presente capítulo, o Capítulo 2 com aspectos importantes da área de gerência de redes necessários à compreensão do escopo do trabalho capítulo. Entre eles, a gerência OSI com as cinco áreas funcionais utilizadas para a divisão da gerência em partes menores, a gerência Internet com o protocolo SNMP e um breve comentário sobre a gerência na área de telecomunicações.

No terceiro capítulo apresenta-se um estudo sobre as redes neurais artificiais (RNAs), em que são apresentados modelos de neurônio, as topologias mais conhecidas de RNAs e alguns algoritmos de aprendizado. Com o objetivo de diferenciar redes diretas e redes recorrentes é apresentado uma quadro comparativo entre essas duas topologias. No final do capítulo são abordados a teoria de problemas e os problemas distribuídos, bem como aspectos sobre estudos de complexidade e computabilidade das RNAs.

Definições e exemplos sobre sistema geral, sistema orientado, sistema funcional, sistema dinâmico e sistema complexo são apresentados no Capítulo 4.

A inteligência artificial distribuída (IAD) é discutida no quinto capítulo. O conceito de agentes autônomos, bem como o que são sistemas multi-agentes, sistemas de raciocínio distribuído e alguns dos mecanismos mais conhecidos para desse tipo de sistema são apresentados.

A essência e a originalidade do trabalho estão delineadas através dos itens do sexto capítulo. É neste capítulo, que as soluções para os problemas distribuídos são analisadas conforme as cinco áreas funcionais da OSI: gerência de falhas, de desempenho, de configuração, de contabilização e de segurança. A metodologia de desenvolvimento de agentes autônomos considerando problemas estáticos e dinâmicos e o comportamento da gerência é apresentada considerando cada uma das cinco áreas mencionadas. Neste capítulo, também são apresentados aspectos da implementação e alguns experimentos realizados durante o desenvolvimento do trabalho.

Seguem as considerações finais, as referências bibliográficas e o glossário.



## Capítulo 2

# Gerência de Redes

Um dos objetivos deste trabalho é estudar soluções para problemas distribuídos. No escopo de problemas distribuídos poderão ser encontradas tanto problemas de redes de computadores quanto problemas em ambientes distribuídos em geral. Na literatura existe uma certa confusão entre redes de computadores e sistemas distribuídos. Para Tanenbaum [69], a diferença chave é que para usuários de sistemas distribuídos, a existência de vários processadores é transparente, e para os usuários de redes não. Ou seja, o usuário de um sistema distribuído não tem consciência de que os processadores realizam as tarefas compartilhando recursos (alocação de tarefas para os processadores, acessos a disco, etc.), ao contrário, para o usuário parece um processador único. Em uma rede o processo é bem diferente. Explicitamente o usuário precisa “logar” em uma máquina, enviar arquivos, executar programas remotos, etc. No entanto, o autor classifica um sistema distribuído como um caso especial de uma rede, com distinção dos softwares que executam em cada um. Para ele uma rede de computadores significa uma coleção de computadores autônomos interligados, e dois computadores estão interligados se eles trocam informações.

Nogueira em [29], declara que aumenta o número de sistemas computacionais que se agrupam como redes de elementos que se comunicam através de uma estrutura de transporte de dados. E que sistemas de telecomunicações, distribuídos por natureza, estão se assemelhando aos sistemas computacionais. No entanto, são necessárias abordagens de gerência que facilitem e simplifiquem a operação desses sistemas. Entre as diferenças em gerenciar redes de computadores e redes de telecomunicações, destacam-se: a escala, a

abrangência geográfica, os serviços, a contabilidade de uso, a heterogeneidade de recursos, a tecnologia e a longevidade de equipamentos.

Enquanto as redes de telecomunicações prestam serviços de telefonia e canais de dados, as redes de computadores prestam serviços de informação e transferência de dados. Existem várias definições para gerência de redes na literatura (boas referências são citadas em [57]), a maioria delas foram produzidas por organizações de padronização e utilizam termos muito específicos. Sistemas de computação distribuída ou redes, são ambientes difíceis de operar, diagnosticar e detectar problemas. Gerência de Redes é o processo para controlar uma rede de dados complexa (pode ser de computadores ou de telecomunicações) para aumentar a sua eficiência e garantir a sua produtividade [1][41][57][71].

Atualmente é possível subdividir a gerência de redes de elementos em três. O Modelo de Referência OSI, a gerência da Internet (estas duas para redes computacionais) e a TMN, esta última destina-se às telecomunicações. Cada uma dessas abordagens possui um protocolo que define quais informações deverão ser coletadas, como estas informações poderão ser obtidas e quais as operações que poderão ser realizadas.

Existem autores de trabalho que dividem a gerência de outra forma. Como Pras [57] que subdivide a gerência de redes em implícita ou explícita. Em que o gerenciamento implícito é quando a tomada de decisões é realizada por um equipamento de hardware ou software. E explícito é aquele realizado pelo operador ou administrador da rede. Outra classificação é quanto à localização das ações de gerência: centralizada ou distribuída. A gerência centralizada é aquela que possui um número limitado de entidades gerentes que controlam outras entidades responsáveis por funções primárias de gerência, chamados agentes, é o caso dos protocolos CMIP e SNMP. Ao contrário, a gerência distribuída não possui sistemas centrais de tomada de decisões. Essas funções são adicionadas ao sistema assim que as funções primárias são realizadas. Exemplos de funções primárias são: a transferência de chamadas telefônicas, que a um tempo atrás eram realizadas por operadores, nos dias de hoje são realizados de forma implícita, automaticamente.



## 2.1 Gerência OSI

O primeiro padrão que descreveu a gerência OSI foi o Modelo de Referência OSI [57][71]. Este modelo identificou a gerência OSI como uma importante área de trabalho e forneceu definições iniciais. Em torno de 1980, um Grupo de Trabalho foi formado dentro da ISO (*Working Group ISO/TC 97/SC 21/WG 4* – nos dias de hoje chama-se ISO-IEC/JTC 1/SC 21/WG 4) para desenvolver a gerência OSI.

O modelo de gerência OSI é composto por:

- Quanto às funções, a gerência de redes foi dividida em cinco áreas funcionais, que serão explicadas mais adiante;
- Quanto a sua estrutura: gerenciamento de sistemas, gerenciamento de camada e operação de camada.
- Quanto a manipulação das informações, com relação a base de informação de gerência MIB, pode-se dizer que: armazenas as informações transferidas ou modificadas pelo uso de protocolos do OSI. Estão definidos gerentes e agentes, capazes de realizar as funções de gerência e interagir com os objetos gerenciáveis, respectivamente. Portanto, as informações podem ser manipuladas por gerentes locais ou remotos e obtidas através de agentes locais ou remotos.

## 2.2 Áreas Funcionais

No sentido de definir um escopo para a gerência de redes o Modelo de Referência OSI definiu cinco áreas funcionais: Falhas, Desempenho, Configuração, Contabilização e Segurança. Esta subdivisão é normalmente usada na área de gerência independente do tipo de protocolo e é importante para o escopo do presente trabalho [19][41][57].

### 2.2.1 Gerência de Falhas

A gerência de falhas é o processo de localizar problemas, ou falhas, em uma rede de dados. Envolve as tarefas de descobrir o problema, isolá-lo e solucioná-lo quando possível. Entre

as causas mais prováveis para falhas em uma rede estão: erros de projeto e implementação da rede, erros de sobrecarga, distúrbios externos, tempo de vida útil de equipamentos expirado e má implementação de softwares (famosos “bugs”). Entre as funções de gerência destacam-se [19][41][57]:

- Manter e examinar registros (“logs”) de erros;
- Aceitar e agir sobre notificações de erros;
- Traçar e identificar falhas;
- Realizar testes de diagnóstico;
- Corrigir falhas.

### 2.2.2 *Gerência de Configuração*

A configuração de alguns dispositivos de uma rede controlam o comportamento dos dados que trafegam pela rede. A gerência de configuração é o processo que determina e configura estes dispositivos críticos. Roteadores, pontes (“bridges”, em inglês), terminais e servidores são exemplos destes dispositivos. A gerência de configuração auxilia a localizar quais os softwares e versões estão dispostos em cada equipamento [19][41][57].

Entre as funções desta área destacam-se:

- Registrar a atual configuração;
- Registrar alterações quando realizadas;
- Identificar todos os componentes da rede endereçando os pontos de acesso à rede;
- Realizar reinicializações quando ocorrer queda nos sistemas;
- Realizar trocas nas tabelas de roteamento.

### 2.2.3 *Gerência de Desempenho*

A gerência de desempenho deve assegurar que a rede tenha capacidade para suportar e acomodar uma certa quantidade de usuários. Ou seja, ela é extremamente necessária para otimizar a Qualidade do Serviço. Este processo mede o desempenho dos equipamentos e softwares disponíveis através de registros de algumas taxas de medidas. Exemplos destas taxas são vazão (“throughput”), taxas de erros, taxas de utilização e tempo de resposta.

Estes registros de desempenho podem ser utilizados em outras áreas tais como [19][41][57] para:

- Auxiliar a detectar falhas na rede;
- Auxiliar a determinar quando serão necessárias alterações na configuração da rede;
- Auxiliar a gerência de contabilização em ajustes de contas.

#### 2.2.4 Gerência de Contabilização

Determinar quais os recursos e a forma que estão sendo utilizados pelos usuários é tarefa da gerência de contabilização. Além disso, este processo auxilia a assegurar que os usuários tenham acesso a quantidade suficiente dos recursos disponíveis. Envolve também, garantir ou remover permissões de acesso à rede [19][41][57].

#### 2.2.5 Gerência de Segurança

A gerência de segurança é o processo que controla o acesso às informações disponíveis na rede. Existem informações armazenadas em computadores ligados à rede que são impróprias a todos os usuários. O grupo de informações mais conhecido em que não fica disponível, para evitar ações impróprias que prejudiquem os usuários, é o conjunto de senhas que permitem o acesso à rede. A gerência de segurança permite que o administrador monitore as tentativas de entrada na rede [19][41][57].

### 2.3 Gerência Internet

Com o crescimento da Internet em meio à década passada surgiu a necessidade de estruturar e padronizar um gerenciamento apropriado. Em 1987, surgiram três propostas, das quais duas sobreviveram. Uma delas foi o *Simple Network Management Protocol*, ou SNMP como é mais conhecido. E a outra é o *Common Management Over TCP/IP* ou CMOT. O CMOT foi uma tentativa de usar padrões de gerência OSI em ambientes Internet. Problemas como demora de implementações e experiências operacionais com este protocolo

trabalho devido ao ambiente de testes. O trabalho será implementado em um ambiente TCP/IP, caracterizado por estações de trabalho, barramento Ethernet e sistema operacional UNIX. Rose em [64], apresenta um interessante histórico sobre o gerenciamento da Internet.

### 2.3.1 *Protocolo SNMP*

De fato existem poucas diferenças entre o SNMP e as idéias propostas pelo Modelo OSI antes de adotar um gerenciamento orientado-a-objetos, de fato o segundo é muito mais teórico e complexo do que o primeiro. Entre os pontos em comum destacam-se principalmente o uso de gerentes e agentes, operações de GET e SET sobre os objetos gerenciados, uso da linguagem ASN.1 para definição da informação de gerência e utilização de uma ou mais MIBs.

No SNMP, um gerente pode controlar muitos agentes. O protocolo é construído sobre o UDP, que é o protocolo de transporte não orientado a conexão do ambiente Internet. A informação é armazenada em PDUs (*Protocol Data Units*) do SNMP codificadas de acordo com a linguagem ASN.1 diretamente sobre o protocolo de transporte. Os cinco tipos de PDUs são: *GetRequest*, *GetNextRequest*, *SetRequest*, *Response* e *Trap* [15][16][17].

Como o SNMP utiliza um serviço de transporte não confiável, as operações entre gerente e agente são realizadas com confirmação. Como o serviço é não confiável deve-se ter certeza que a mensagem chegou ao destinatário através de um reconhecimento. Se o reconhecimento não chega durante um certo período (“time-out”) a mensagem deve ser retransmitida [41][57].

### 2.3.2 *SNMPv2*

Desde a publicação original do SNMP várias propostas de melhora foram apresentadas. Em 1992, o grupo de pesquisa resolveu acatar algumas dessas propostas e produzir um novo padrão. Entre estas melhorias destacam-se: maior segurança, a possibilidade de construir

---

Marshall T. Rose é um dos importantes nomes em relação à gerência da Internet, é integrante do SNMP Working Group

uma hierarquia de gerentes e uma nova primitiva que permite o resgate de um grupo de informações [41][57].

### 2.3.3 Elementos da arquitetura SNMP

Como o protótipo do trabalho deverá ser realizado dentro do escopo da gerência Internet, é importante o conhecimento de cinco elementos da arquitetura SNMP:

- **Agentes:** são os elementos de rede gerenciáveis, isto é, aqueles computadores, pontes, roteadores, etc., que possuem o protocolo SNMP embutido. Quando um elemento de rede não possui tal protocolo, utiliza-se agentes procuradores (“proxy”, em inglês), estes agentes fornecem funções de conversão de protocolo.
- **Gerentes:** são as estações de gerência, responsáveis por executar as funções de gerenciamento. São responsáveis por relatórios sobre operações e informações dos objetos gerenciáveis. Estas informações são repassadas ao gerente humano através de algum tipo de interface.
- **Base de Dados (MIB):** é uma coleção de objetos gerenciáveis mantida pelos agentes. Os objetos definem as diferentes características dos dispositivos de rede que estão sendo gerenciados.
- **Protocolo:** protocolo do tipo pergunta/responde. É utilizado para trocar informações entre os agentes e gerentes.
- **Autenticação:** é o meio de segurança pelo qual o agente SNMP valida as requisições de uma estação de gerência antes de respondê-las.

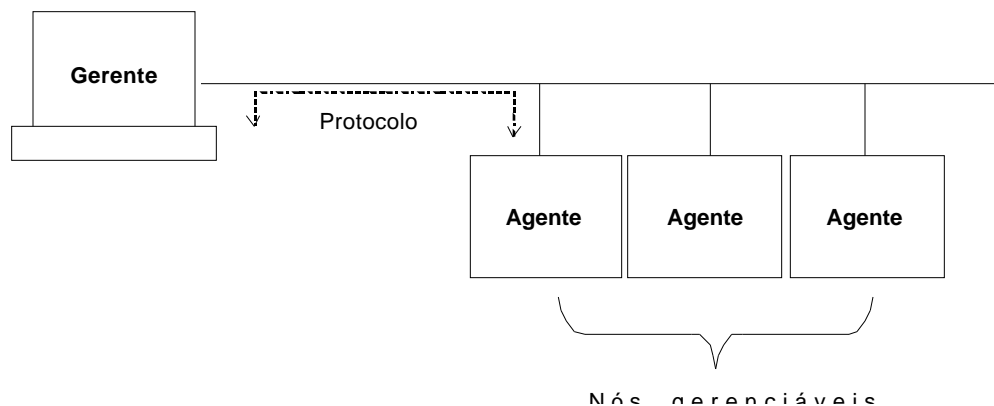


Figura 1 – Relacionamento gerente-agentes utilizando o protocolo SNMP.

## 2.4 Gerência de Telecomunicações

No sentido de padronizar as interfaces e operações necessárias ao gerenciamento das redes de telecomunicações, o termo TMN foi introduzido pelo ITU-T (antigo CCITT) para abreviar *Telecommunications Management Network*. A definição, de acordo com a norma M.3010 (Modelo de Informação Genérica), é que “uma TMN é conceitualmente uma rede separada que faz interface a uma rede de telecomunicação em diversos pontos separados”[71].

A TMN possui três arquiteturas distintas:

- Uma funcional, as funções de gerência da TMN encontram-se na recomendação M.3300;
- Uma física, definindo conceitos de Sistemas de Operação, Elementos de Rede e Redes de Comunicação de Dados, uma variedade de dispositivos de mediação, interfaces e adaptadores de funções que não possuam interface com a TMN. As normas M.3020 e M.3200 definem as interfaces e serviços, respectivamente; e,
- Uma arquitetura de informações, definido pela norma M.3010.

Como o trabalho não será dirigido para área de telecomunicações, não serão apresentados maiores detalhes sobre este protocolo. No entanto, existe a possibilidade de futuros trabalhos aplicando a técnica aqui proposta à área de telecomunicações.

## Capítulo 3

# Redes Neurais Artificiais (RNAs)

As Redes Neurais Artificiais (RNAs), ou Sistemas Conexionistas, foram inspiradas em sistemas biológicos onde um grande número de células nervosas funcionam individualmente de forma lenta e imperfeita. No entanto coletivamente, são capazes de realizar tarefas que muitos computadores não tem capacidade de fazê-las. São normalmente formadas por diversos processadores simples interconectados com vários elementos de memória, cujos pesos das conexões são ajustados por experiência. Este ajuste de pesos caracteriza a capacidade de aprendizado das redes neurais [4][21][40][45][59][60]. Existem atualmente, diversos tipos de redes neurais, tais como *Percéptrons*, *Adalines*, Redes de Hopfield, Mapas de Kohonen, BAM (*Birectional Associative Memories*), entre outras.

As RNAs, como mencionado anteriormente, possuem inspiração biológica muito forte. A maioria delas tentam simular o sistema biológico das células nervosas. Da mesma forma que em nosso organismo, os neurônios artificiais realizam tarefas mais complexas atuando coletivamente.

Entre as características mais importantes destacam-se [4][40][45][59][60]:

- Processamento e memória distribuída ao longo da estrutura da rede,
- Normalmente, as redes são treinadas para realizar uma determinada tarefa, e não programadas;
- Os neurônios são amplamente interligados de forma que o estado de um neurônio afeta o potencial de um grande número de neurônios,
- Os pesos das conexões são adaptativos, se ajustam a medida que a rede aprende,
- As unidades de processamento contém funções de ativação do tipo não linear;
- Generalização.

### 3.1 Modelos de neurônio

É comum encontrar na literatura diversos tipos de modelos de neurônios artificiais.

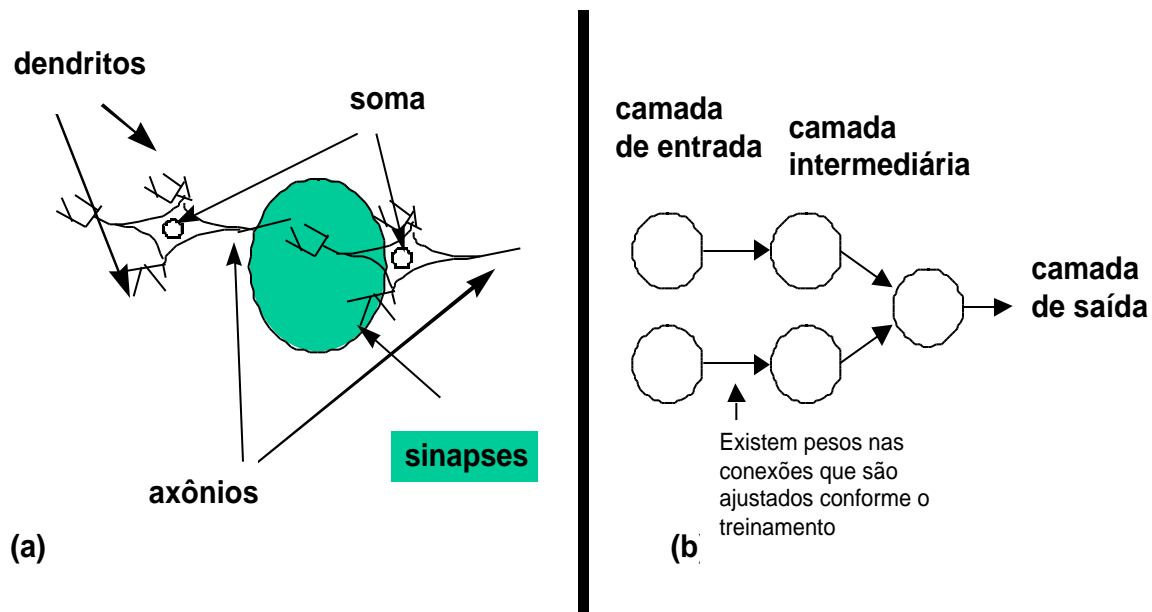


Figura 2 - (a) Neurônios Biológicos e (b) Neurônios Artificiais.

A Figura 2 ilustra algumas semelhanças entre os neurônios biológicos e os neurônios artificiais. Os dendritos correspondem as entradas, a soma corresponde ao somatório e os axônios às saídas do neurônio artificial. A maior diferença concentra-se nas sinapses, que nos neurônios biológicos não existe uma ligação física (as trocas de informações são realizadas através de neurotransmissores), enquanto que nos neurônios artificiais existe uma ligação física e o armazenamento das informações dependem dos pesos sinápticos. Assim como, nos sistemas biológicos não é possível determinar exatamente em quais neurônios as informações estão armazenadas, elas encontram-se distribuídas ao longo da rede neural artificial.

O primeiro modelo de neurônio artificial foi apresentado por McCulloch e Pitts (1943) [49], e em 1959 foram apresentados simultaneamente, os *Percéptrons* por Rosenblatt e os *Adalines* por Widrow e Hoff [3]. Tanto os *Percéptrons* quanto os *Adalines* possuíam regras de aprendizado. Existe prova da convergência do algoritmo de



aprendizado associado aos percéptrons. No *Adaline* é possível calcular pesos pelo método dos mínimos quadrados, apesar de ter sido apresentado com aprendizado iterativo.

A Figura 3 ilustra um modelo de neurônio artificial simples, composto de três entradas, denominadas  $x_0$ ,  $x_1$  e  $x_2$ , em que  $x_0$  pode ser considerada uma polarização de valor 1. Os pesos sinápticos  $w_0$ ,  $w_1$  e  $w_2$ , inicialmente são atribuídos valores aleatórios entre -0.1 e 0.1. Estes valores, normalmente são ajustados ao longo do treinamento. O somatório  $_$  calculado através da soma das entradas multiplicadas pelos pesos de cada uma irá ativar a saída.

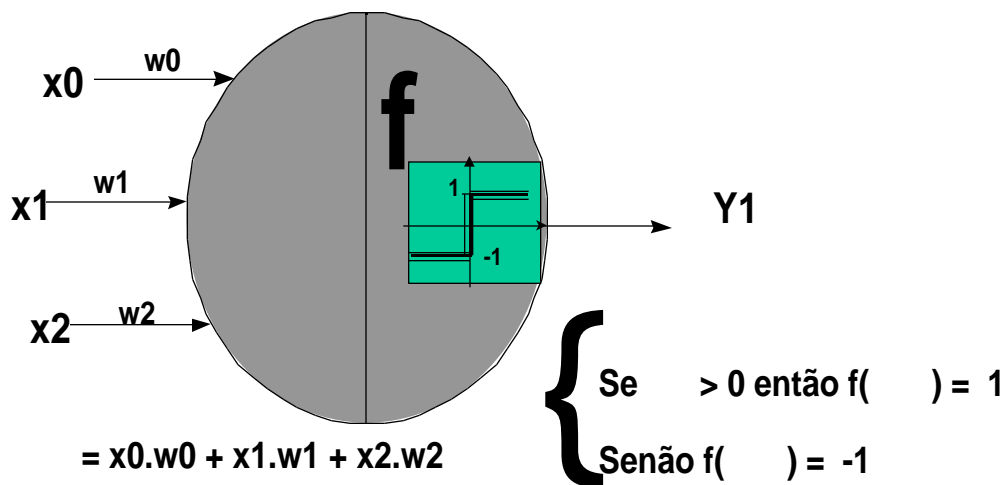


Figura 3 - O neurônio artificial.

A saída, denominada  $Y_1$  irá depender da função de ativação  $f$ , que no exemplo é uma função bipolar. Estas funções são determinadas no início de um projeto de redes neurais, e na maioria dos casos devem ser utilizadas funções sigmoidais (no caso de redes com algoritmo de *backpropagation* sempre deverão ser utilizadas funções deste tipo).

Uma definição formal, é descrita por De Azevedo em [21], o autor apresenta o modelo geral de neurônio por com base na Teoria de Sistemas.

## 3.2 Topologias das RNAs

As redes neurais artificiais podem ser subdivididas em redes diretas e redes recorrentes. As redes diretas são aquelas que o fluxo das informações segue em uma única direção, enquanto que, as redes recorrentes (ou de realimentação) podem se realimentar de uma saída demonstrando troca de estado.

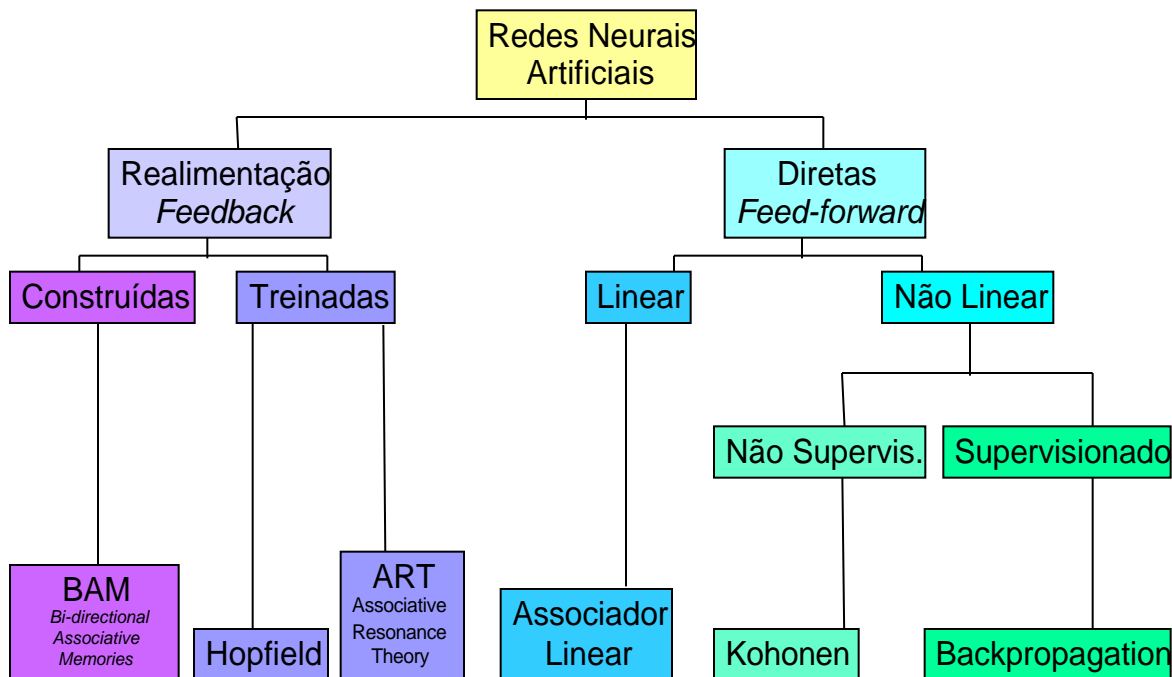


Figura 4 - Classificação de Redes Neurais Artificiais.

A Figura 4 [40] apresenta um diagrama que classifica as redes neurais em dois grandes grupos, conforme sua topologia: Realimentação ou Diretas. No caso de redes com realimentação (ou *feedback*) podem ainda ser subdivididas em Construídas (por exemplo, redes do tipo BAM - *Bi-directional Associative Memories*) ou Treinadas (Redes do tipo Hopfield ou do tipo ART - *Associative Resonance Theory* - por exemplo). As redes diretas podem ser subdivididas em redes lineares e não lineares. As redes não lineares podem ainda ser subdivididas conforme o algoritmo de aprendizado utilizado: Supervisionado e Não Supervisionado.

## 3.3 Algoritmos de treinamento

Os algoritmos de treinamento ou regras de aprendizado ditam como são feitos os ajustes dos pesos sinápticos para que a rede adquira experiência ao longo do treinamento. Existem dois tipos: Supervisionado e Não Supervisionado.

### 3.3.1 *Supervisionado*

As regras ou algoritmos de aprendizado do tipo supervisionado atuam com o auxílio de um professor. Neste caso, a rede precisa conhecer o conjunto de entrada e o conjunto de saída. A saída obtida pela rede é calculada conforme a saída desejada, calcula-se um erro, que é utilizado para corrigir os pesos sinápticos.

Existem numerosas regras deste tipo, no entanto a maioria são variações da regra de Hebb e da Regra Delta. A mais de 30 anos atrás Donald O. Hebb teorizou que a memória associativa biológica concentra-se nas conexões sinápticas das células nervosas, e que o processo de aprendizado e armazenamento de memória envolve trocas nas forças com as quais os sinais nervosos são transmitidos através das sinapses individuais. A primeira regra de Hebb diz que se um par de neurônios estão ativos simultaneamente existe uma troca dos pesos sinápticos, reforçando esta conexão.

### 3.3.2 *Não Supervisionado*

Regras de aprendizado do tipo não supervisionado atuam de forma competitiva ou auto supervisionadas. Para este tipo de regras são fornecidas a rede apenas o conjunto de entrada. Os neurônios da rede competem entre si fornecendo uma classificação como saída. Existe, além da competição, uma cooperação entre os neurônios da rede. Entre os algoritmos mais conhecidos, destacam-se: *o vencedor leva tudo* (“the winner take all”) e a Inibição Lateral.

A Figura 5 ilustra um exemplo de algoritmo competitivo. Abaixo, tem-se a seguinte entrada (1 0 0 1). É uma rede altamente conectada, onde os neurônios da mesma camada competem entre si, o que possuir maior somatório, recebe valor 1, os demais recebem 0.

Assim para todas as seqüências de entrada, o algoritmo termina quando todas as entradas tenham sido classificadas.

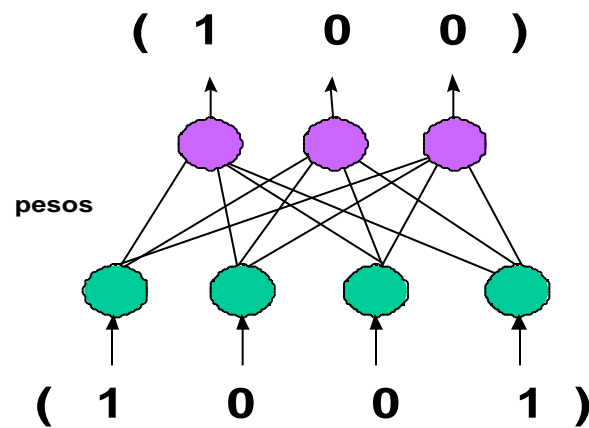


Figura 5 - Exemplo de algoritmo competitivo.

### 3.3.3 Por Reforço

O aprendizado por reforço (“reinforcement learning”) é o método por tentativa e erro. Existe um índice de desempenho, chamado de sinal de reforço, que é utilizado para otimização. Este paradigma de aprendizado tem profunda inspiração biológica em que os comportamentos satisfatórios são reforçados e os insatisfatórios geram alterações nos valores correspondentes às conexões [5].

## 3.4 Redes Diretas X Redes Recorrentes

O objetivo desta comparação é apresentar as vantagens do uso de redes dinâmicas. Talvez por falta de informação ou porque as redes diretas são mais divulgadas, o número de trabalhos que utilizam redes estáticas para solução de problemas dinâmicos é muito grande. A verdade é que as redes diretas podem funcionar para um caso particular do problema, mas não para todas as instâncias.

A Tabela I apresenta um quadro comparativo entre as redes neurais diretas (*feedforward*) e as redes recorrentes (ou com realimentação, *feedback*).

Tabela I - Principais diferenças entre RNAs Diretas e Recorrentes.

Características	RNAs Diretas	RNAs Recorrentes
Quanto ao estado	Estáticas – não existe troca de estados.	Dinâmicas – existe troca de estados conforme o tempo.
Quanto à topologia	Não possui ciclos. O fluxo dos dados possui apenas uma direção, da entrada para a saída.	Possui ciclos com realimentação. A saída de uma das camadas pode realimentar a entrada de dados (o fluxo pode ser da entrada para a saída, das camadas intermediárias para a entrada, ou da camada de saída para a entrada).
Quanto ao número de camadas	Para solucionar um problema <b>LINEAR</b> : duas camadas uma de entrada e uma de saída. <b>NÃO LINEAR</b> : tem que ter no mínimo uma camada intermediária.	Como existem ciclos com realimentação, pode haver ou não camada intermediária, depende da ordem do predicado.
Quanto a sua construção	<b>Treinadas</b> ou de forma <b>supervisionada</b> (como é o caso do algoritmo de treinamento Backpropagation) ou <b>não-supervisionada</b> (como é o caso dos mapas de Kohonen).	<b>Construídas</b> : exemplos de redes neurais construídas são BAM(Kosko) e Hopfield . <b>Treinadas</b> : normalmente são mais complicadas, porque o ajuste dos pesos deve ser feito nos dois sentidos para frente e para trás. Exemplo deste tipo de rede ART (Adaptative Resonance Theory – Grossberg & Carpenter).

### 3.5 Redes Recorrentes

É característica das redes recorrentes possuir o grafo de sua topologia com ciclos e trabalharem com o tempo discreto e síncrono. Existem, no entanto, redes com ciclos que não devem ser chamadas de recorrentes, como é o caso de dinâmica em que o neurônio é representado por uma equação diferencial [5][60]. É possível encontrar uma rede recursiva a partir de um autômato finito definido por sua equação matemática (Figura 6):

$$S = \{U, Y, X, x_0, \dots\}$$

Onde:

U é um conjunto finito de entradas;

$Y$  é um conjunto finito de saídas;

$X$  é um conjunto de estados ou espaço de estado;

$x_0 \in X$  é o estado inicial;

$f: U \times X \rightarrow X$  é a função de transição de estado;

$g: U \times X \rightarrow Y$  é a função saída.

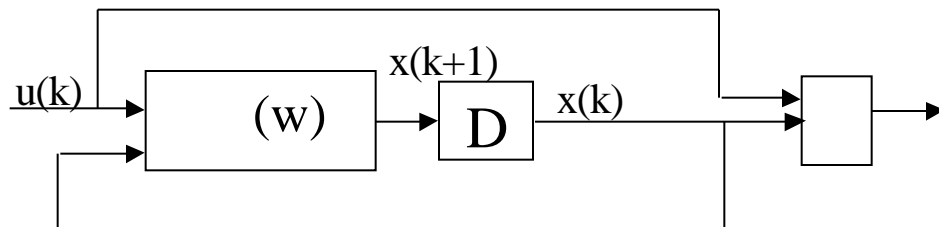


Figura 6 - Implementação neural de autômato finito.

$$x(k+1) = f(x(k), u(k), W)$$

$$y(k) = g(x(k), u(k))$$

A Figura 6 [5] apresenta a implementação das equações acima com uma camada de neurônios munidos de retardo (transformando  $k+1$  em  $k$ ) implementando (ou aproximando) a função  $f$  que tem por entradas  $x(k)$  e  $u(k)$ . Um segundo conjunto de neurônios implementando a função  $g$  (podendo ser neurônios estáticos, portanto sem retardos).

### 3.5.1 Aprendizagem em RNAs Recorrentes

Nos sistemas dinâmicos, normalmente, os valores de entrada e saída são conhecidos, no entanto, nestes sistemas não estão disponíveis o conjunto de estados. Neste caso, a rede recorrente tenta estimar o estado com base no conjunto de valores de entrada e saída apresentados durante o treinamento. O objetivo do aprendizado é encontrar uma regra para ajuste dos pesos durante o treinamento [5][60].

Roisenberg [60] descreve bons exemplos de aprendizado em RNAs recorrentes e propõe um algoritmo baseado em retropropagação quando se dispõe do estado. Durante o treinamento, a linha de atraso entre os neurônios de saída dos estados da Máquina de Estados Finitos (MEF) e os neurônios da camada intermediária foram desligadas, obtendo

uma rede direta com neurônios de entrada que correspondem aos entrada de sensores no tempo  $k$  e neurônios de entrada que correspondem a estados da MEF no tempo  $k$ . Também foram utilizados neurônios de entrada que correspondem a entradas de sensores, a seqüência temporal de todas as possíveis entradas da MEF, de tal forma que passe por todos os estados possíveis da MEF. Então, apresenta-se a rede os dados de entrada  $u(k)$  e o estado  $x(k)$  da MEF, e propagou-se na rede, obtendo o próximo estado calculado  $x(k+1)$ . Depois desta etapa é possível aplicar o algoritmo de retropropagação tradicional, calculando o erro a partir do estado desejado da MEF  $x'(k+1)$ .

### 3.6 Teoria de Problemas

Antes de abordar a computabilidade e complexidade de problemas resolvidos por RNAs, é interessante definir o que é um problema. Esta interrogação preocupa a mente humana desde os antigos filósofos gregos. Destacam-se Descartes com o seu método analítico baseado na filosofia *dividir para conquistar*, e deste século Polya [56] sugere que se responda as seguintes perguntas antes de solucionar um problema:

- Quais são os dados?
- Quais são as possíveis soluções?
- O que caracteriza uma solução satisfatória?

O problema pode ser formalizado como um objeto matemático do tipo  $P = \{D, R, q\}$ , consistindo de dois conjuntos não vazios,  $D$  os dados e  $R$  os resultados possíveis e de uma relação binária  $q \subseteq D \times R$ , a condição que caracteriza uma solução satisfatória, associando a cada elemento a solução única desejada. Desta forma o problema é representado como uma função.

Segundo Barreto [5] existem diversas formas de definir uma função, dentre elas destacam-se enumeração exaustiva, declarativamente, por um programa ou através de exemplos. Quando um problema não é completamente definido para todo valor de seus dados, em que conhece-se apenas a definição do problema para um subconjunto dos dados possíveis. Neste caso, a solução não é única: todas as funções que forem iguais dentro da

região em que o problema é definido são válidas. Neste caso, é melhor ter uma solução aproximada do que os dados para definir a função. Este trabalho defende que resolver o problema será então encontrar um modo de implementar a função ou aproximá-la com as ferramentas disponíveis. Ou seja, através de exemplos treinar a rede e obter-se valores estimados da solução para os outros valores, utilizando a propriedade de generalização das RNAs. O autor apresenta ainda, uma classificação dos problemas que podem ser tratados pela neurocomputação, como problemas estáticos e dinâmicos, descrita a seguir.

### *3.6.1 Problemas estáticos linearmente separáveis*

São problemas que envolvem a implementação de uma função (porque são estáticos) e podem ser resolvidos por uma RNA direta sem camada intermediária.

Problemas estáticos não linearmente separáveis: problemas envolvendo a implementação de uma função e que podem ser resolvidos por uma rede direta, com neurônios estáticos, exigindo ao menos uma camada intermediária.

### *3.6.2 Problemas dinâmicos com dinâmica finita*

Os problemas com dinâmica finita são aquelas com duração da resposta do sistema após a entrada do sistema dura um tempo finito. Este tipo de problema pode ser resolvido por uma rede direta com neurônios dinâmicos.

### *3.6.3 Problemas dinâmicos com dinâmica infinita*

São problemas que a duração da resposta do sistema após uma entrada pode durar um tempo infinito. Esses problemas podem ser resolvidos por rede com retroação e com neurônios dinâmicos ou rede estática com um conjunto de retardos. No entanto, a questão da estabilidade da rede, ou seja, se ela encontrará ou não uma solução e quanto tempo levará, fica em aberto.



### 3.7 O que são Problemas Distribuídos

Na seção anterior foi abordado o conceito de problema. No entanto, nada foi dito quanto aos problemas distribuídos. Problemas Distribuídos podem ser entendidos como problemas que possuem as informações dispostas fisicamente em localizações diferentes. Pode-se afirmar que o grau de complexidade desse tipo de problema cresce de acordo com o número de nodos envolvidos. No caso de redes de computadores, os problemas distribuídos são mais visíveis, os equipamentos compartilham informações dispostas em localizações físicas diferentes, existem equipamentos locais e remotos. É claro que a complexidade do problema dependerá da extensão da rede. Uma rede local com dez estações de trabalho interligadas com barramento Ethernet não pode ter problemas de complexidade superior a uma rede Metropolitana.

Por exemplo, no caso de falha em um segmento de uma rede de computadores como a da UFSC (semelhante a da Figura 7). Muitas vezes não é necessário que toda a rede seja influenciada pela falha, porém os equipamentos que fazem parte do segmento com problema provavelmente serão atingidos. No caso da falha estar nas linhas do NPD, que ligam o resto da rede com o mundo externo, a falha atingirá boa parte dos usuários, pelo menos aqueles que estiverem acessando páginas da Web, transferências de arquivo (que não sejam entre servidores locais da UFSC) e correio eletrônico. No caso de falhas deve-se obter o maior número de informações sobre o problema.

- Determinar a exata localização do problema é muito importante;
- Determinar a extensão do problema – a complexidade da solução;
- Isolar a área afetada;
- Tomar providências.

No entanto, surge outra questão importante: como solucionar de forma eficiente esses problemas? Tem sido freqüente o uso de redes diretas para solução de problemas distribuídos, principalmente em gerência de redes de computadores. Normalmente, tratam-se de sistemas dinâmicos com problemas dinâmicos [39][75]. Informalmente, sistemas

dinâmicos possuem mudança de estado, no capítulo 4, será apresentada a descrição formal de sistemas dinâmicos.

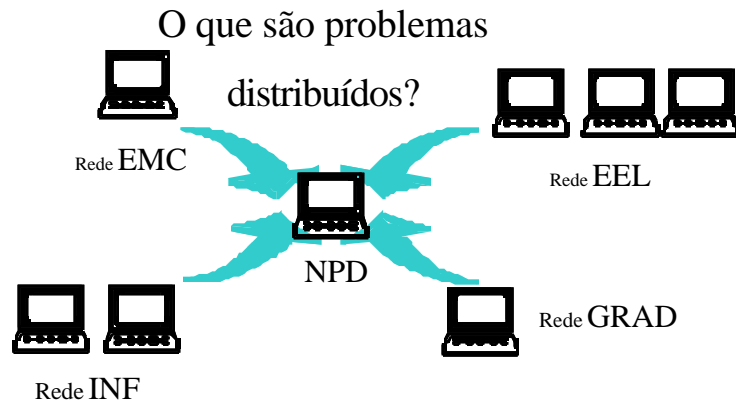


Figura 7 - Problemas Distribuídos.

Desta forma, pode-se afirmar que problemas dinâmicos possuem características dinâmicas. E neste caso, deve-se propor uma solução dinâmica. No entanto, não são todos os problemas da área de gerência dinâmicos. Existem problemas com características estáticas e portanto, bem resolvidos por redes neurais diretas. Este trabalho pretende apresentar um método para desenvolver agentes autônomos com as ferramentas adequadas (redes neurais artificiais diretas ou recorrentes) para solucionar problemas distribuídos na área de gerência de redes.

### 3.7.1 Problemas comuns em redes de computadores

Em [19] são apresentadas quatro categorias de problemas de redes de computadores relacionadas com as diferentes partes da rede:

- Hardware da rede : problemas de conectividade física são os mais comuns desta categoria, devem-se ao fato do estresse ambiental e tempo de vida dos cabos, inclui quebra de cabos, cabos curtos, cabos danificados, quebras do circuito de hardware, controlador de interface de má qualidade;
- Equipamentos de Interconexão: como o crescimento das redes é muito freqüente produtos de interconexão são fontes de problemas comuns que

umentam com a mesma intensidade. Erros de configuração também são muito comuns. Nesta categoria deve-se verificar se as filas de processamento não cresceram de forma anormal.

- Protocolos de Rede, esse tipo de problema é possível apenas identificar porque normalmente estão relacionados a falta de compatibilidade de diferentes fabricantes.
- Aplicações de Rede, esse tipo de problema só é possível resolver se a própria empresa ou universidade escreveu a aplicação, caso contrário deve-se contatar o fabricante para reportar o problema ocorrido (“bug” em inglês).

### 3.8 Computabilidade e Complexidade de RNAs

Um problema é dito computável se for possível construir uma Máquina de Turing para resolvê-lo. Quando a solução de um problema é abordada por um computador baseado em instruções (CBI), isto corresponde a existência de um algoritmo, composto de uma seqüência finita de instruções, capaz de solucionar tal problema. No entanto, um computador construído com redes neurais é equivalente a uma Máquina de Turing, e em consequência, tem capacidade para resolver qualquer problema computável e somente eles, tem despertado grande interesse científico. Esta equivalência foi inicialmente provada por McCulloch e Pitts [49] usando manipulações de lógica e por Arbib [3] com argumento bastante intuitivo.

Mas se as redes neurais não resolvem problemas que antes não se sabia resolver, qual seu interesse? A resposta vem da facilidade de resolver problemas e dos recursos necessários. A isto chama-se complexidade. Quando se usa um computador digital esta complexidade é geralmente expressa em quantidade de memória e tempo. Quando se usa a abordagem conexionista deve-se se falar em topologia da rede (número de neurônios e como são ligados) e como são os neurônios.

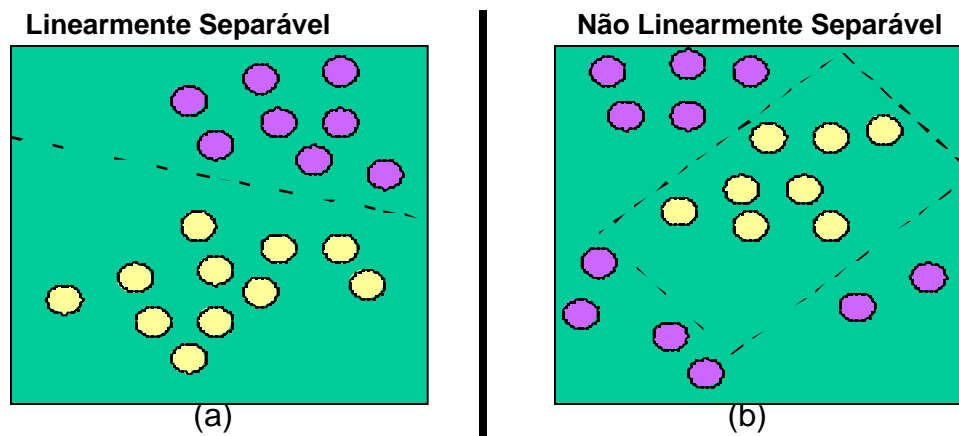


Figura 8 – Problemas linear e não linearmente separáveis.

A maior contribuição de Minsky e Papert [50] ao estudo da complexidade de problemas resolvidos por RNAs foi a classificação dos problemas em (Figura 8): linearmente separáveis; e, não linearmente separáveis. Se um problema é linearmente separável ele pode ser resolvido com uma camada de entrada e saída de neurônios, como é o caso dos circuitos lógicos E, OU e Não. Se o problema é não linearmente separável ele precisa de uma camada intermediária de neurônios entre a entrada e a saída da rede, como é o caso do OU-Exclusivo e da paridade de uma seqüência de 0 e 1.

### 3.8.1 Confluência

As redes neurais artificiais são compostas de neurônios artificiais como visto nas seções anteriores, e possuem uma função de ativação semelhante a:

$$Y_i = f(w_{ij} \cdot X_j)$$

Linearmente não separável pode ser não-linearmente separável.

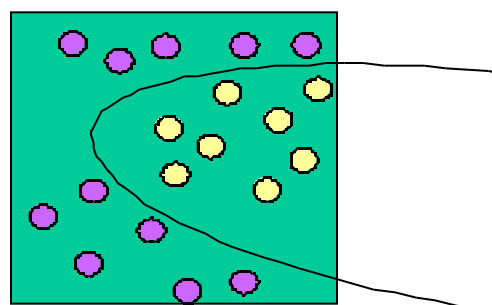


Figura 9- Exemplo sobre confluência com parábola.

O conteúdo da função  $f$  chama-se confluência, normalmente toma-se o produto escalar como confluência:

$$W_{i1} \cdot X_1 + W_{i2} \cdot X_2 + \dots + W_{in} \cdot X_n$$

No entanto, podem ser utilizadas outras operações, tais como: neurônios *fuzzy* que utilizam máximos e mínimos formando redes neurais do tipo OU/E nebulosas [9]. Se for utilizada uma superfície com grau proporcional ao produto com duas dimensões, isto é, se a confluência for uma parábola com parâmetros abertura e vértice ajustados, de modo a ter capacidade de separar não linearmente um conjunto de pontos não separável (Figura 9).

## Capítulo 4

# Teoria de Sistemas

A Figura 10 ilustra um esquema sobre os tipos de sistemas que serão abordados. O sistema geral possui a mais alta abstração de um sistema, pela sua definição é possível afirmar que um sistema geral é constituído por um conjunto de relações entre suas entidades e o mundo exterior. O sistema orientado possui conjunto de entrada e saídas. O Sistema Temporal apresenta seus conjuntos em função do tempo, enquanto que o Sistema Funcional em função de estados. Estes dois últimos sistemas unidos formam Sistemas Dinâmicos. Finalmente, Sistemas Complexos são caracterizados por um grupamento de sistemas dinâmicos. A seguir estes conceitos serão discutidos em maiores detalhes porque serão necessários para a compreensão das próximas seções deste trabalho [60][62].

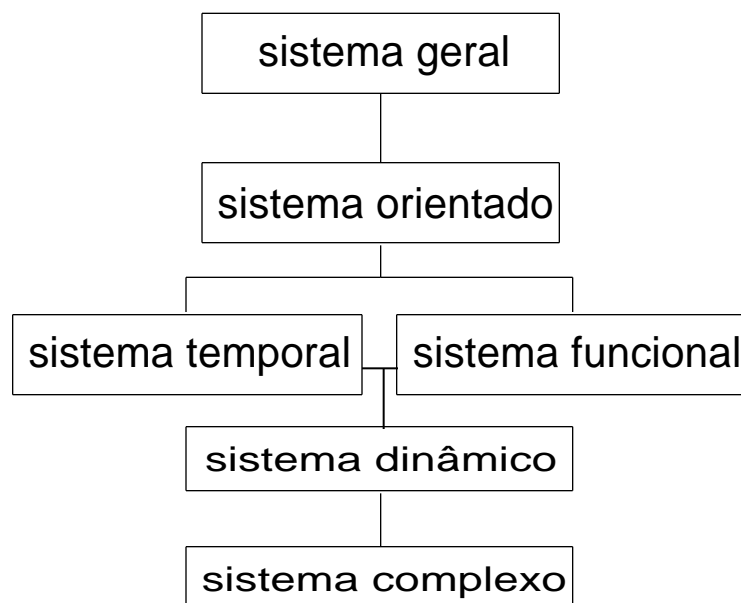


Figura 10 - Teoria de Sistemas

## 4.1 Sistema Geral

Em uma primeira definição um Sistema Geral pode ser considerado como o menor nível de descrição possível, em que somente as interações com o mundo exterior são retidas. Ou seja, o conceito de sistema geral é o mais alto nível de abstração sobre o conceito de um sistema, que pode ser visto como uma caixa preta com indicações de interação com o mundo exterior (Figura 11).

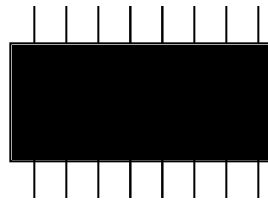
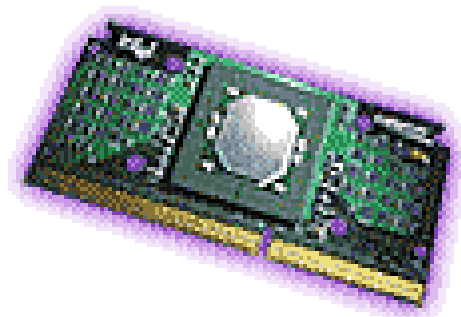


Figura 11 - O sistema geral visto como uma caixa preta.

### 4.1.1 Exemplos de Sistema Geral

Podemos imaginar esta caixa preta como um microprocessador, onde apenas é possível enxergar os conectores de contato com a placa de um computador. Neste nível de descrição não é necessário saber qual a função desses conectores, apenas que são os pontos de integração com o mundo externo.



Outro bom exemplo, é um modem em que existem sinais de transmissão e recepção de dados, no entanto neste nível de descrição não é necessário conhecer os detalhes. Um outro exemplo de sistema geral que não é orientado é constituído por uma tabela em que cada linha corresponde a um casal. Assim:

João	Sofia
Carlos	Rosa
Luiz	Neuza
...	...
José	Angela

O sistema faz a correspondência entre um dos cônjuges e o outro sem haver precedência.

**Definição 1:** um sistema geral  $g$  é definido por um conjunto de relações entre as entidades que caracterizam as interações com o mundo exterior. Então  $g = I$ , onde  $I$  são todas as interações.

## 4.2 Sistema Orientado

Caso deseje-se definir quais das variáveis serão usadas como entrada e quais serão a saída, tem-se um sistema orientado. Seguindo a analogia da caixa preta, o sistema orientado poderia ser visto como a caixa preta com indicações das entradas e saídas (Figura 13).



Figura 13 - Sistema orientado.

No caso do processador seria descrever os conectores que o compõem, as entradas e as saídas



### 4.2.1 Exemplos de Sistema Orientado

Um exemplo que faz parte do cotidiano é o próprio computador. Neste nível de descrição é possível indicar as entradas e saídas de forma clara. Na Figura 14, as entradas são o mouse, o joystick, a câmera sobre o vídeo. Como saída tem-se o monitor.



Figura 14 - Computador como exemplo de sistema orientado.

Outro exemplo interessante seria um sistema de transferência de arquivos, onde é possível indicar as operações de entrada e saída de forma explícita.

**Definição 1.2:** um sistema orientado pode ser caracterizado por uma relação entre o conjunto de entradas e saídas.

$$O \quad X$$

onde:

$O$  é o sistema de transferência de arquivos;

$X$  é o conjunto de entradas admissíveis, ou seja  $X$  é o conjunto de operações que podem ser solicitadas ao sistema. Onde cada uma das operações são representadas por  $x$ , de forma que  $X = \{\text{append, bye, delete, dir, cd, lcd, open, quit, user}\}$ .

$Y$  é o conjunto de saídas admissíveis, ou seja, a resposta para cada operação fornecida na entrada do sistema. Se for colocado na entrada uma operação `ftp <endereço>`, tem-se:

```
C:\>ftp ftp.ufsc.br
Connected to ftp.ufsc.br.
220 ftp.ufsc.br FTP server (Version wu-2.4.2-academ[BETA-15](1) Tue Jun 9
15:44:
20 GRNLNDDT 1998) ready.
User (ftp.ufsc.br:(none)): anonymous
331 Guest login ok, send your complete e-mail address as password.
Password:
230 Guest login ok, access restrictions apply
```

Outra operação: colocar como entrada `dir` tem-se:

```
ftp> dir
200 PORT command successful.
150 Opening ASCII mode data connection for /bin/ls.
total 48
d--x--s--x  2 0      0      512 May 10 1998  bin
dr--r-Sr--   3 0      0      512 May 26 1998  etc
drwxr-s--x  30 0      0     2048 Jun  2 11:07  incoming
dr-xr-xr-x   2 0      0      512 May 26 1998  lib
dr-xr-xr-x  33 0      0     1024 Jun  1 10:12  pub
dr-xr-xr-x   3 0      0      512 May 26 1998  usr
226 Transfer complete.
369 bytes received in 0.16 seconds (2.31 Kbytes/sec)
```

## 4.3 Sistema Temporal

Os dois conceitos apresentados, sistemas geral e orientado, não incluem o tempo como intrínseco ao sistema. Entretanto, freqüentemente quando lidando com sistemas que são abstrações do mundo real, o tempo é um parâmetro relevante. O sistema temporal é o sistema orientado com o acréscimo dos tempos. Pode-se imaginar uma entrada na caixa preta no tempo inicial ( $t_0$ ), e após uma quantia de tempo denotado por  $t_{i+1}$  é gerada uma saída

(  
Figura 15).

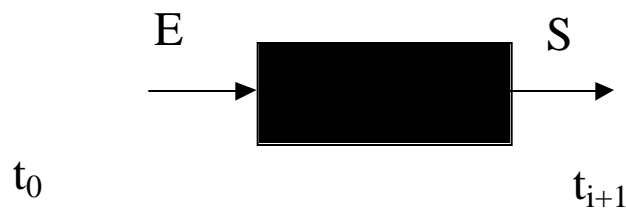


Figura 15 - Sistema temporal.

### 4.3.1 Exemplos de Sistema Temporal

A utilização de uma linha (“link”) de uma rede Ethernet pode suportar altas rajadas de tráfego, no entanto se a carga normal exceder a 40% o segmento de rede terá seu desempenho comprometido. Para determinar a utilização de uma linha é necessário examinar

a quantidade de bits transmitidos e de bits recebidos sobre a largura de banda disponível.

Tem-se então:

$$\text{Utilização} = \text{bits\_recebidos} + \text{bits\_transmitidos} / \text{largura\_de\_banda}$$

**Definição 1.3:** um sistema orientado onde  $f$  e  $g$  são funções do tipo:  $f: T \rightarrow U$  e  $g: T \rightarrow Y$ , onde:

$U$  é o conjunto dos valores de entrada, valores dos bits transmitidos e recebidos no tempo inicial.

$Y$  é o conjunto dos valores de saída, passados o tempos  $t+1$ , será valores dos bits transmitidos e recebidos no tempo  $i+1$  e o valor da utilização neste período.

$T$  é um conjunto ordenado com um primeiro elemento, normalmente denotado por  $t_0$  (às vezes é usual considerar  $t_0$  como - ) que é chamado de conjunto  $T = \{t_0, t_1, t_2, \dots, t_n\}$ .

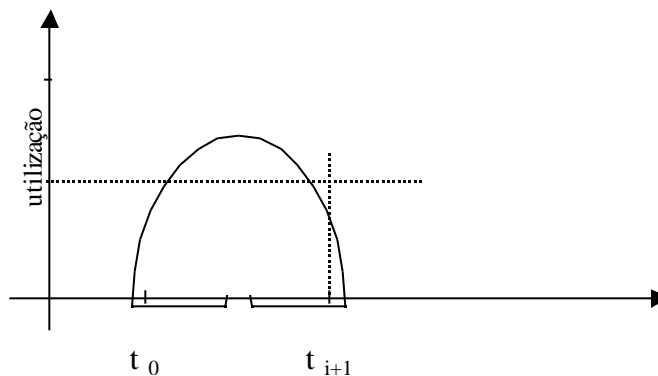


Figura 16 – Um Sistema Temporal.

Note que um sistema temporal cujas funções  $f$  e  $g$  são constantes, ou seja, a imagem será os mesmos elemento de  $U$  e  $Y$  é dito ser um Sistema Estático. No exemplo do servidor de uma rede, se não houver alteração na taxa de utilização em um determinado período, ou seja se for constante, ou é porque houve alguma falha ou o sistema permaneceu estável naquele período.

## 4.4 Sistema Funcional

Conceito de Estado: o conceito de estado é freqüentemente usado em várias disciplinas, por exemplo, na teoria de autômatos finitos.

Figura 17 - Sistema funcional.

No caso da caixa, seria o que aconteceu dentro da caixa, algum evento interno mudou do estado inicial ( $e_0$ ) para um estado final ( $e_f$ ) (Figura 17).

#### 4.4.1 Exemplos de Sistema Funcional

Através de um exemplo bem simples, considere uma agenda telefônica, em que a entrada é o nome da pessoa e a saída é o telefone. E se houver mais de um telefone o de casa e o celular? Qual o telefone aparecerá? É possível definir um conjunto de estados  $X = \{\text{telefone\_casa}, \text{telefone\_celular}, \text{telefone\_escritório}\}$  e aí definir a função:

$f: X \rightarrow Y$

onde:

$f$  função busca de número de telefone conforme o número;

$X$  é o conjunto das entradas admissíveis, ou seja, nome da pessoa a ser procurada na agenda,  $Y$ , de forma que

$X$  é o conjunto dos estados,  $X = \{\text{telefone\_casa}, \text{telefone\_celular}, \text{telefone\_escritório}\}$ , que poderá mudar automaticamente, e esta mudança poderá ocorrer de diversas formas. Se  $h = 8 - 12$  e  $14 - 18$  então  $\text{telefone\_trabalho}$ ; se  $h = 18 - 8$  e  $12 - 14$  então  $\text{telefone\_casa}$ . Se  $\text{telefone\_casa}$  não responde depois de 20 chamadas então  $\text{telefone\_celular}$ .

$Y$  é a saída, conjunto de telefones da lista, em que  $y$  é um telefone,



Figura 18 - Exemplo de Sistema Funcional.

## 4.5 Sistema Dinâmico

Um sistema pode ser funcional e temporal ao mesmo tempo caracterizando um Sistema Dinâmico, com o estado do sistema variando conforme o tempo. Em um sistema dinâmico, descreve-se um sistema como se estivesse descrevendo o mecanismo de como ele trabalha (internamente), especificando como o conjunto de estados varia conforme o tempo. Na analogia com a caixa preta seria, como é o comportamento interno desta caixa conforme o tempo. Para gerar uma descrição comportamental basta fazer o mecanismo funcionar para cada entrada desejada gerando o sistema orientado correspondente (Figura 19).

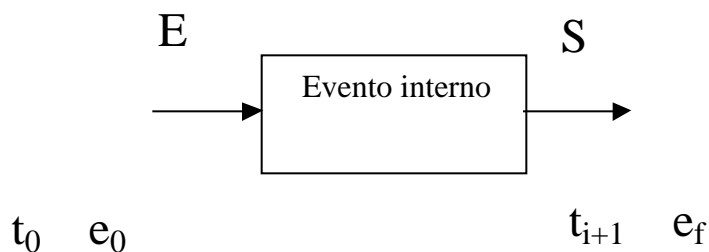


Figura 19 - Sistema dinâmico.

Formalmente, tem-se:

Definição 1.6: um sistema dinâmico é o objeto matemático.

$$S = \{T, U, Y, X, ; \}$$

Onde:

T é o conjunto dos tempos,

é o conjunto das funções de entrada  $= \{ :T \rightarrow U \}$ ,

U é o conjunto dos valores de entrada,

$Y$  é o conjunto dos valores de saída,

$f$  é o conjunto das funções de saída  $f = \{ f : T \rightarrow Y \}$ ,

$X$  é o conjunto dos estados,

$\delta$  é a função de transição dos estados:  $\delta : T \times T \times X \rightarrow X$ ,

$g$  é a função de saída:  $g : T \times X \rightarrow U \rightarrow Y$ .

#### 4.5.1 Exemplos de Sistema Dinâmico

Um modem “half-duplex” pode encontrar-se em um dos três estados: Inativo, Transmitindo ou Recebendo. A troca de estado dependerá se existe ou não mensagens na fila de transmissão ou se existe alguma indicação para recepção de mensagens, caso contrário permanece inativo.

Formalizando o exemplo tem-se:

$$S = \{ T, U, Y, X, \delta, g \}$$

Onde:

$T$  é o conjunto dos tempos  $T$ , em que  $T = \mathbb{R}^+$ ;

$U$  é o conjunto das funções de entrada  $U = \{ f : T \rightarrow \{0,1\} \}$ , se existem mensagens na fila de transmissão ou se existe sinal para recebimento de mensagens;

$Y$  é o conjunto dos valores de entrada, as mensagens a serem transmitidas e recebidas;

$Y$  é o conjunto dos valores de saída,  $Y = \{ \text{sinal de reconhecimento, sinal de erro} \}$ .

$f$  é o conjunto das funções de saída  $f = \{ f : T \rightarrow Y \}$ , emite o sinal de reconhecimento se a mensagem foi recebida corretamente, caso contrário envia um sinal de erro;

$X$  é o conjunto dos estados do sistema,  $X = \{ \text{Inativo, Transmitindo, Recebendo} \}$ ;

$\delta$  é a função de transição dos estados:  $\delta : T \times T \times X \rightarrow X$ ;

$g$  é a função de saída:  $g : T \times X \rightarrow U \rightarrow Y$ .

#### 4.5.2 Modelo Geral de Neurônio como Exemplo de Sistema Dinâmico

Utilizando teoria de sistemas, o modelo formal de um neurônio pode ser apresentado como um sistema dinâmico, conforme segue. Com efeito, as variáveis de definição de um sistema dinâmico são identificáveis em um neurônio formal. Observe o seguinte objeto matemático:

$$S = \{T, U, Y, X, ; \}$$

Onde:

$T$  é o conjunto dos tempos, conjunto munido de uma ordem completa. Exemplos de conjuntos de tempo usuais são os números naturais e os reais.

$U$  é o conjunto das funções de entrada  $= \{ :T \rightarrow U \}$ , provenientes dos sensores dos neurônios ou da saída de outros neurônios,

$X$  é o conjunto dos valores de entrada, que geralmente são vetores de entrada cujos componentes são os valores dos sinais nas entradas dos dendritos,

$Y$  é o conjunto dos valores de saída, corresponde ao valor no axônio no neurônio biológico, que é a frequência de pulsos resultantes dos potenciais de ação,

$$\text{é o conjunto das funções de saída} = \{ :T \rightarrow Y \}$$

$X$  é o conjunto dos estados, conhecidos como estados de ativação dos neurônios,

$$\text{é a função de transição dos estados: } : T \times T \times X \times X,$$

$$\text{é a função de saída: } : T \times X \times U \rightarrow Y.$$

Quando o conjunto dos estados for igual a um, o modelo de neurônio é dito *estático*, que é um caso particular do neurônio dinâmico.

## 4.6 Sistema Complexo

Um conjunto de sistemas interconectados formam um sistema complexo. Utilizando o exemplo das caixas, seriam várias caixas de sistema dinâmico trabalhando como módulos paralelos para atingir um objetivo comum (denotado na Figura 20 pela seta Saída).

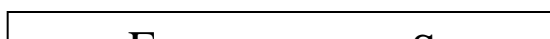


Figura 20 - Sistema Complexo.

**Definição 1.7:** Um sistema complexo  $c$  é definido como um conjunto de sistemas denotados por  $d$  interconectados.

$$d \times d \quad c$$

#### 4.6.1 Exemplos de Sistema Complexo

As plataformas de gerência de redes, tanto de computadores quanto de telecomunicações, são exemplos de sistemas complexos. É possível identificar as áreas funcionais de gerência (Seção 2.2) como sistemas dinâmicos que compõem tais plataformas. Por exemplo, um sistema para gerenciamento de uma rede local deverá ter controle de acesso dos usuários (Gerência de Segurança), contabilizar o uso dos equipamentos (Gerência de Contabilização), incluir equipamentos ou refazer a topologia da rede (Gerência de Configuração), controlar o desempenho (Gerência de Performance), prevenir e solucionar falhas na rede (Gerência de Falhas). A maioria destas, entre outras funções não citadas, devem ser feitas em paralelo sobre um sistema também dinâmico (nesse caso, a rede local).

Outro exemplo fácil de imaginar é o organismo humano que possui características cognitivas, como visão, audição, tato, paladar e olfato. Até os dias de hoje não puderam ser



reproduzidas pela máquina. O mais próximo é o robô desenvolvido pelo MIT, chamado COG onde tenta-se reproduzir a visão humana.

## Capítulo 5

# Inteligência Artificial Distribuída (IAD)

A inspiração biológica na área da IA sempre se faz presente, e não existe maneira melhor de explicar o que é Inteligência Artificial Distribuída (IAD). Imagine uma colônia de formigas que tem sobrevivido aos rigores da evolução por cerca de bilhões de anos. Se tivermos a visão de apenas uma formiga não será possível compreender a imensidão de uma colônia. Por outro lado, do ponto de vista da colônia podemos ver a distribuição do trabalho como uma linguagem muito mais poderosa. Estas colônias só sobrevivem porque a distribuição tem um significado, que provavelmente é invisível aos níveis inferiores [34][35].

A IAD é composta por sistemas cujo o funcionamento depende de um determinado conjunto de partes menores para resolver, de modo cooperativo, um determinado problema. Segundo Barreto [5], entre os principais motivos para utilizar a IAD destacam-se:

- Inspiração biológica, além do exemplo das formigas citado anteriormente, podemos pensar em outros seres vivos que para sobreviver vivem em bandos, para proteger os seus filhotes, por exemplo. O próprio ser humano, agrupa-se em nacionalidades, sociedades, comunidades para solucionar seus problemas maiores, formam uma estrutura tão complexa que existe uma área destinada ao seu estudo que é a *sociologia*;
- Decompor problemas complexos em problemas de menor escala, cada um com suas características distintas. Costuma-se dividir os problemas em problemas

menores e dividi-los entre os agentes autônomos para solucioná-los, porém não é a única abordagem para implementação da IAD;

- Usufruir dos ambientes distribuídos, podendo determinar vários agentes paralelos para trabalhar de forma cooperativa.

Além disso, em [59] são apresentadas três contribuições importantes que arquiteturas paralelas e distribuídas podem oferecer ao estudo de sistemas inteligentes. A Modelagem Psicológica, visto que o ser humano trabalha em paralelo. A Modularidade, é mais fácil operar sobre módulos independentes. E consequentemente, o Aumento da Eficiência porque nem todos os conhecimentos são necessários a todos os módulos.

Uma das primeiras implementações em IA Distribuída foi o Modelo do Quadro Negro, apresentado no projeto de compreensão da fala HEARSAY-II. No entanto existem vários outros métodos. Bem mais recente, são os agentes inteligentes, ou agentes autônomos de software, que serão analisados na próxima seção.

## 5.1 Agentes Autônomos

“Agentes são sistemas computacionais que operam em ambientes dinâmicos e imprevisíveis. Eles interpretam dados obtidos pelos sensores que refletem eventos ocorridos no ambiente e executam comandos em motores que produzem efeitos no ambiente. O grau de autonomia de um agente está relacionado à capacidade de decidir por si só como relacionar os dados dos sensores com os comandos dos motores em seus esforços para atingir objetivos, satisfazer motivações, etc.” [60].

De uma forma geral um agente autônomo (AA) é composto basicamente por: sensores, atuadores e por uma entidade cognitiva responsável pela autonomia do agente. No caso de um agente de software os sensores são as entradas, e s atuadores as saídas. Por autonomia entende-se que é a capacidade de governar-se por si próprio.

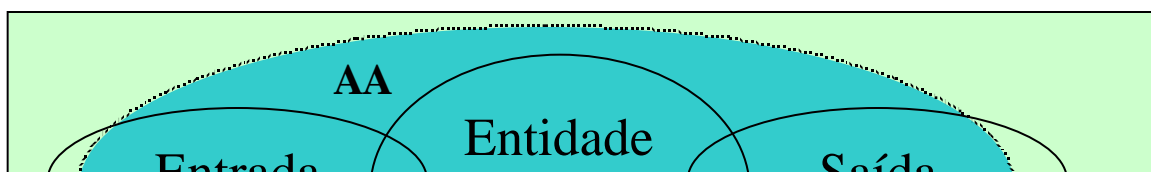


Figura 21 - Agente Autônomo em seu ambiente.

A Figura 21 ilustra os componentes básicos de um AA inserido em seu ambiente. Existem outros conceitos para composição de AAs que envolvem desde arquiteturas genéricas, até comportamentos e aprendizado de AAs [5][7][8] [31][36][37][45][46][47][53][55][60][61][62][63][65][72][73].

Além da autonomia, outro conceito importante para trabalhar-se com AAs é o comportamento. O comportamento é a resposta fornecida pelo AA aos estímulos recebidos do ambiente [7][31][36][37][45][46][73]. Os estímulos são recebidos pelos sensores, mapeados pela entidade cognitiva e refletidos no ambiente através dos atuadores. Este comportamento está diretamente ligado ao grau de autonomia do agente.

## 5.2 Modelo de Agente

Existem outras definições para Agente Autônomos na literatura. No entanto, este trabalho resolveu adotar uma definição formal que utiliza conceitos da Teoria de Sistemas (Capítulo 4), estas definições foram apresentadas por Roisenberg em [60][61][62][63]. O modelo formal apresentado de Roisenberg, descreve agente autônomo e o ambiente como sistemas dinâmicos.

### 5.2.1 Modelo Multi-agentes

Considerando o que foi abordado até o momento, é possível definir um sistema de IAD, dentro do escopo do trabalho, como um sistema multi-agentes. Portanto, cada entidade assume um papel dentro da solução do problema. Outras abordagens sobre sistemas multi-agentes podem ser encontrados em [7][8][37][45][63][65][72][73].

Figura 22 - Sistema multi-agentes.

Então, da mesma forma que De Azevedo [21] apresentou um formalismo para redes neurais utilizando a teoria de sistemas, e Roisenberg [60] apresentou um modelo de agentes autônomos utilizando sistemas dinâmicos é possível definir um sistema multi-agentes como um sistema complexo.

Segundo a definição, um sistema complexo  $S_c$  é definido como um conjunto de sistemas denotados por  $S_d$  interconectados. Esses sistemas podem ser dinâmicos ou não. Então define-se um sistema multi-agentes como um sistema complexo composto de diversos agentes autônomos. Cada agente é um processo dinâmico e possui um certo grau de inteligência e de autonomia.

### 5.3 Por que utilizar IAD

O quadro da Tabela II a seguir apresenta as principais diferenças de trabalhar com sistemas distribuídos e utilizar sistemas monolíticos. A partir deste quadro é possível fazer uma analogia sobre as diferenças entre a IA e a IAD.

Tabela II - Diferenças entre sistemas distribuídos e monolíticos.

Características	Distribuído	Monolítico
Modularidade	Totalmente modular, os sistemas distribuídos caracterizam-se por	Não existe modularidade, normalmente é um sistema único que

	cooperativa.	
Eficiência	Devido a modularidade, este tipo de sistema é mais rápido.	Mais lento, corre o risco de por alguma falha parar no meio do processamento.
Processamento	Paralelo. No entanto, corre o risco de ficar dependendo de um processo que entre em <i>deadlock</i> para terminar o processamento.	Sequencial, porém não depende de outros processos para chegar ao final do processamento.
Comunicação	Deve haver mecanismos necessários para a comunicação entre processos.	Não existe tal preocupação.
Controle	Deve haver mecanismos necessários para o controle dos processos.	Não existe tal preocupação.
Coordenação	Deve haver mecanismos necessários para a coordenação	Não existe tal preocupação.

Aproveitando-se das vantagens da distribuição de sistemas podemos apontar que utilizar IAD torna a solução de problemas modular e eficiente. A modularidade permite que um problema seja dividido em problemas menores e a cada um seja atribuída uma entidade para solucioná-la. Estas entidades são os agentes autônomos ou inteligentes, conceituados nos itens anteriores. Pode-se afirmar que um sistema distribuído de forma inteligente poderá convergir a uma solução mais rápido do que um sistema monolítico.

Entre as desvantagens, é que um sistema de IAD deverá possuir um mecanismo de mantenha esta mesma eficiência para garantir a coordenação, comunicação e cooperação entre suas entidades. A literatura traz alguns destes mecanismos (como é o caso do Modelo do Quadro negro), porém dependendo do problema pode ser interessante criar um mecanismo adequado.

## 5.4 Sistema de Raciocínio Distribuído (SRD)

Na literatura é comum encontrar soluções de problemas efetuadas por sistemas compostos de agentes inteligentes. Em Rich [59], esses sistemas são chamados de SRD, e é definido como:

*Sistema de Raciocínio Distribuído é aquele composto por um conjunto de módulos separados (em geral chamados de agentes) já que cada módulo assume o papel de uma entidade de solução de problemas) e por um conjunto de caminhos de comunicação entre eles”.*

Estes sistemas podem ser completamente distribuídos ou ter mecanismo de controle centralizado.

#### *5.4.1 Vantagens de um SRD*

Um conjunto de vantagens para o uso de SRDs são apresentados por Rich [59], alguns desses itens foram discutidos na Seção 4.4 de forma comparativa entre a IA com a IAD. A seguir são apresentados em relação ao trabalho que será desenvolvido:

- Modularidade do sistema, o sistema proposto deverá ser composto por módulos menores sob forma de agentes autônomos baseados em RNAs recorrentes;
- Eficiência, nem todos os conhecimentos são necessários para todos os agentes do sistema, cada um terá um grau de autonomia diferenciado;
- Rapidez, como trata-se de um sistema distribuído os agentes serão processados em paralelo;
- Raciocínio heterogêneo, poderão ser realizados métodos diferentes de soluções de problemas em cada AA;
- Múltiplas perspectivas, é a possibilidade de mais de um AA encontrar a solução de um problema;
- Problemas distribuídos, na área de redes a grande maioria dos problemas tem característica distribuída;
- Confiabilidade, se a solução está distribuída em uma série de agentes diferentes, o problema poderá ser resolvido mesmo que um dos agentes apresente falhas.

Em contrapartida às vantagens apresentadas no item anterior, um sistema de raciocínio distribuído deve oferecer:

- Como existem vários módulos para solucionar o problema, deve existir um mecanismo de coordenação para assegurar que as atividades dos AAs sejam realizadas e que o sistema atinja seus objetivos;
- Um mecanismo de comunicação que permita a troca de informações entre os agentes e o mecanismo de coordenação

## 5.5 Mecanismos de Coordenação e Controle

O maior problema de se adotar uma solução distribuída é sem dúvida como Controlar e Coordenar os processos permitindo que atuem de forma Cooperativa. Entre as abordagens para solucionar os problemas de controle e coordenação destacam-se [5][59]: planejamento de execução multi-agentes, planejamento e negociação: redes de contrato e planejamento distribuído, as demais encontradas normalmente são variações destas três.

### 5.5.1 *Planejamento de execução multi-agentes*

Este tipo de raciocínio decompõe o problema e delega os subproblemas de forma independente aos AAs. Possui um agente-mestre.

### 5.5.2 *Planejamento e negociação: redes de contrato*

Um único agente efetua a decomposição do problema e negocia com os outros as atribuições das tarefas. Existem agentes-gerentes e agentes-contratados.

### 5.5.3 *Planejamento distribuído*

Não existe um controlador central. Nesse caso é assumido que um agente é racional se ele se comporta de forma considerada ideal em termos de seus objetivos. Outro conceito dessa abordagem é a racionalidade limitada, propriedade do agente que se comporta de forma próxima à ideal com relação a seus objetivos, dentro do que seus recursos permitem.

## 5.6 Mecanismos de Comunicação

Entre as soluções para o problema de comunicação as abordagens mais utilizadas são: a do Quadro Negro (QN) e o Sistema de Transmissão de Mensagens. O terceiro mecanismo apresentado será a Linguagem KQML. Foi utilizado em [39] para resolver o problema de comunicação de AAs para gerência de redes.



### 5.6.1 *Modelo Quadro Negro*

Este modelo foi apresentado no projeto de compreensão da fala HEARSAY-II. Naquela ocasião foi apresentado como uma das primeiras implementações de IAD. Atualmente é um modelo utilizado para solucionar o problema de comunicação entre AAs. O modelo compreende [5][59]:

- Um conjunto de módulos independentes contendo o conhecimento sobre o assunto em pauta;
- Uma estrutura de dados compartilhada por diversas fontes de conhecimento, chamada Quadro Negro (QN);
- Um sistema de controle que determina como as várias fontes do conhecimento operam inserindo dados no QN.

### 5.6.2 *Sistema de Transmissão de Mensagens*

É outra alternativa para realizar a comunicação em um SRD. Nessa abordagem, os agentes tendem a saber mais uns sobre os outros do que no modelo do QN. Esse conhecimento permite que os agentes redirecionem as mensagens para aqueles com maior capacidade de alcançar o objetivo. Em [59], é descrito o sistema distribuído MACE, que fornece uma estrutura genérica de SRD e utiliza esse mecanismo para a troca de informações dos módulos do sistema.

### 5.6.3 *Linguagem KQML*

A linguagem KQML (*Knowledge Query and Manipulation Language*) foi projetada para suportar interações entre agentes inteligentes. Ela foi desenvolvida pelo ARPA (DARPA - *Defense Advanced Research Projects Agency*), no programa de compartilhamento de informações e implementada de forma independente por diversos grupos de pesquisa. A comunicação existe em vários níveis. O conteúdo da mensagem é apenas uma parte da comunicação [29][30]

## 5.7 Aplicação da IAD em Gerência de Redes

A idéia de ter uma única estação de gerência para controlar a rede é um pouco ultrapassada. Hoje em dia tem-se objetos distribuídos na gerência de redes. Distribuir o controle nos diversos nós da rede ajuda a diminuir o número de informações de controle que transitam na rede. Nada mais adequado do que fornecer uma solução distribuída e inteligente aos problemas de gerência.

Em Barreto [5] é feito um breve comentário sobre a aplicação de AAs de software em gerência de redes. Transcrevendo suas palavras: *“Uma das mais promissoras aplicações é na gerência de redes de computadores. Neste caso cada nó estaria munido de um agente funcionando de modo autônomo, e contendo inteligência suficiente para desempenhar sua função de gerir o funcionamento do nó do melhor modo possível.”*

Assim, no próximo capítulo serão apresentadas as técnicas disponíveis para implementar os agentes autônomos para gerência de redes de computadores, respeitando as características estáticas e dinâmicas encontradas nas cinco áreas funcionais.

## Capítulo 6

# Agentes Autônomos para Gerência de Redes

Considerando a importância da distribuição, como apresentado no caso das formigas na introdução do capítulo anterior. Questiona-se: qual a melhor maneira de automatizar um problema extremamente complexo e distribuído? A utilização de agentes (processos individuais que realizam tarefas e as comunicam a outros processos chamados gerentes), como tem sido realizado por vários anos na área de gerência de redes. No entanto, esses agentes são passivos, não realizam nenhum tipo de tomada de decisão e não possuem autonomia. Normalmente, coletam informações e alteram informações da MIB ordenados por um processo gerente, que por sua vez é ativado por um gerente humano conhecido como administrador da rede. Existem trabalhos que tentam transformar a abordagem clássica de gerência de redes em uma IAD. Para isso, são propostos mecanismos adaptativos para gerência de redes [70].

Este trabalho apresenta uma metodologia para desenvolver agentes autônomos para gerência de redes empregando as técnicas de IA adequadas. Nem sempre o método de treinamento mais popular, o “backpropagation”, aplicado a redes diretas é o mais adequado. Em muitos casos, como problemas com caráter dinâmico, este método é ineficiente sendo necessário o uso de ferramentas com características dinâmicas. Defende-se que em primeiro lugar deve-se analisar o problema: verificar se possui características estáticas ou dinâmicas, e a partir daí buscar as soluções apropriadas. A gerência de redes, tanto de computadores quanto de telecomunicações, é um problema extremamente complexo e dinâmico quando vista como um todo. No entanto, aplicando o método analítico de resolução de problemas

de Descartes “dividir para conquistar”, a gerência de redes pode ser dividida nas cinco áreas funcionais propostas pelo modelo OSI (apresentado na Seção 2.2). No decorrer deste capítulo serão apresentadas as análises de resolução dos problemas de cada uma destas áreas.

Existem, portanto, duas maneiras de desenvolver agentes inteligentes para solucionar problemas de gerência de redes. Considere um exemplo simples: um sistema para diagnóstico de falhas em uma rede local com 50 computadores e que se deseje construir um sistema de diagnóstico de falhas com características inteligentes. Se o ambiente possui um administrador que já conhece os problemas do ambiente e suas possíveis soluções é mais prático construir uma base de dados sobre as falhas da rede. Uma espécie de sistema de “Troubleshooting” e aplicar técnicas de IA na construção deste sistema, tais como regras de produção e redes diretas. É um ambiente de comportamento reativo, após ocorrer a falha e o administrador busca as soluções. Isto pode ser implementado com uma rede neural direta onde as entradas da rede são os possíveis sintomas e a saída o diagnóstico do problema. No entanto, se for o caso de construir um sistema preventivo o comportamento é chamado de pró-ativo. O sistema de diagnóstico deve funcionar para impedir que haja degradação no funcionamento da rede [22][25][26].

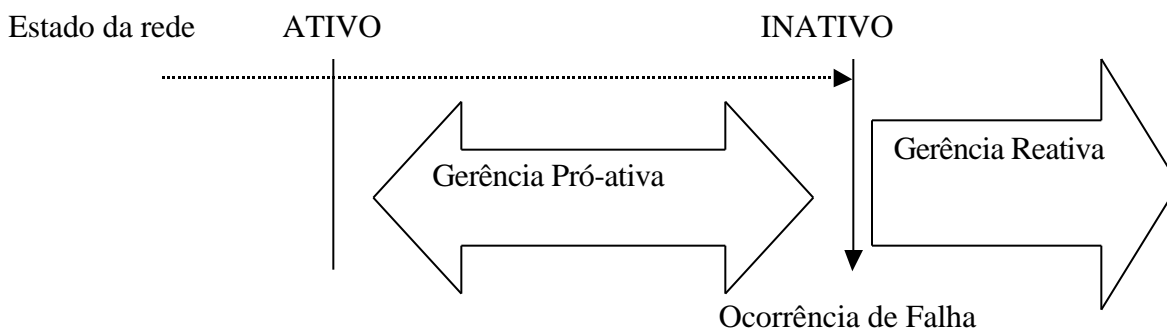


Figura 23 - Comportamento da Gerência de Redes.

A outra forma de resolver o problema de gerência no caso de contar com um especialista é através de exemplos. Neste caso, segundo Barreto [6], apenas a definição do problema é conhecido para um subconjunto de dados possíveis. A partir daí deseja-se

conjunto de dados, mesmo aqueles que não estão incluídos na definição da função. Este é o caso em que se usam exemplos para treinar uma rede neural com algoritmo supervisionado e obtém-se valores estimados da solução para os outros valores, utilizando a propriedade de generalização.

## 6.1 Metodologia

Como desenvolver os agentes autônomos para gerência de redes? Relembrando, no capítulo anterior foram descritos os componentes dos AAs: entrada, entidade cognitiva e saída e que são necessários mecanismos de controle, comunicação e coordenação. Existem diversos tipos de mecanismos que podem ser aplicados, o modelo do quadro-negro é um deles. No entanto, como extrair o conhecimento necessário para a solução de problemas distribuídos, mais precisamente de gerência de redes? Existem duas maneiras como mencionado no início deste capítulo.

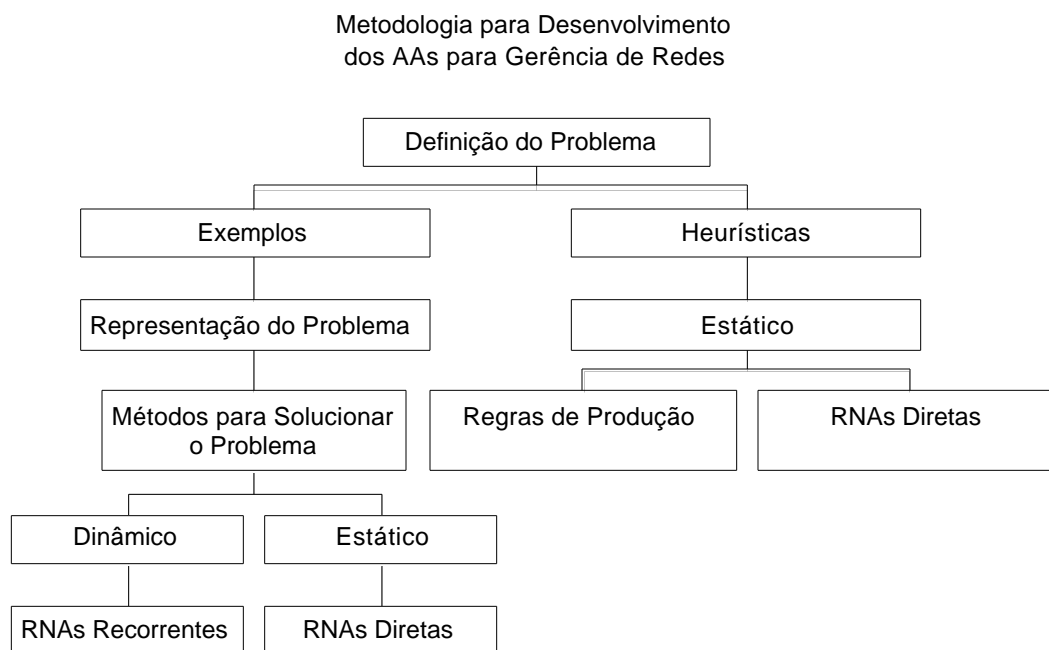


Figura 24 - Metodologia Híbrida para solução de problemas distribuídos.

A primeira, é através de heurísticas. Ou seja, uma forma declarativa de resolver uma função, dando as propriedades que devem ser satisfeitas para solucionar o problema [6], neste caso, é necessário auxílio de um especialista. O especialista que será um administrador de redes, deverá indicar quais os sintomas, para diagnosticar o problema e sugerir as possíveis soluções. Com essas informações pode-se construir uma base de dados (tipo “troubleshooting”) para um sistema de diagnóstico de falhas. Este sistema poderá ter características de IA aplicando-se regras de produção (paradigma simbólico). Outra técnica que pode ser aplicada para implementar a entidade cognitiva de agentes para gerência de redes utilizando heurísticas é através de RNAs diretas.

A segunda maneira, como pode ser observada pelo lado esquerdo do organograma acima, é através de exemplos. Os exemplos formam uma representação do problema, ou seja, apenas a definição do problema é conhecido para um subconjunto de dados possíveis. A partir daí deve-se analisar este subconjunto e verificar se é um problema estático ou dinâmico. Cada tipo de problema possui um método de solução diferente. Um problema estático pode ser resolvido por RNAs diretas. Porém o dinâmico, só é “bem resolvido” se forem utilizadas RNAs com características dinâmicas. Deseja-se então, conhecer os elementos do conjunto de respostas admissíveis para todos os elementos do conjunto de dados, mesmo aqueles que não estão incluídos na definição da função. Os exemplos, portanto, são utilizados para treinar uma rede neural com algoritmo supervisionado e obter os valores estimados da solução para os outros valores, utilizando a propriedade de generalização.

As duas abordagens possuem vantagens e desvantagens. A ação dispensável do especialista é uma das principais vantagens da abordagem dos exemplos. A extração do conhecimento de um especialista é uma tarefa muito complicada. Muitas vezes, a solução de um problema é realizada com a intuição, e fica impossível indicar porque determinada ação foi realizada. Como desvantagem tem-se a não garantia da convergência de uma rede neural. Não existem métodos para garantir que uma rede neural chegue a uma solução. No entanto, podem ser fornecidos bons exemplos para que a probabilidade de que a rede neural aprenda um determinado padrão seja boa.

## 6.2 Gerência de Falhas

Uma das questões mais interessantes no tratamento de redes de computadores é porque a rede pode falhar. Se uma rede falha, muitos ou milhares de usuários podem ficar descontentes. Construir um sistema com múltiplos agentes autônomos que aumentassem a tolerância à falhas das redes de computadores e telecomunicações seria de grande importância. Aliar os dois tipos de comportamentos de gerência, reativo e pró-ativo seria ideal. É possível dividir a Gerência de Falhas como lógica e física. A gerência física é aquela que diz respeito aos componentes da rede. Neste caso, muitas vezes é impossível prever que um determinado cabo ou transistor seja danificado. É o típico comportamento reativo, se foi danificado um elemento da rede ele deverá ser trocado. A gerência lógica diz respeito as aplicações da rede. Significa administrar as aplicações da rede. Isto pode ser realizado de forma pró-ativa. Monitorar as diversas taxas de erros da rede (erro de alinhamento de pacotes ou o número de conexões recusadas) é um exemplo de comportamento pró-ativo.

Normalmente, existem heurísticas sobre as falhas da rede. Então não é necessário desperdiçar tempo treinando uma RNA para chegar a soluções que são conhecidas. Aplicando a metodologia e se existem heurísticas para solucionar o problema: pode-se implementar as heurísticas através de regras de produção ou RNAs diretas, conforme grafo da Figura 25.

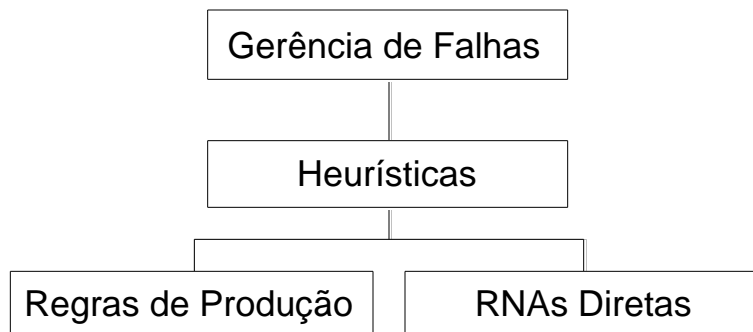


Figura 25 - Gerência de falhas.

### 6.2.1 Exemplo de Diagnóstico de Falhas

Um sistema de diagnóstico de falhas é semelhante a um sistema de diagnóstico médico. Segundo Barreto [6] diagnóstico é o processo de encontrar um defeito em um sistema. É uma das mais populares aplicações de sistemas especialistas, principalmente no domínio médico. Observe a Figura 26 para construção de um sistema para diagnosticar falhas na rede.

Em [19] é apresentado um sistema de diagnóstico de falhas com as seguintes etapas:

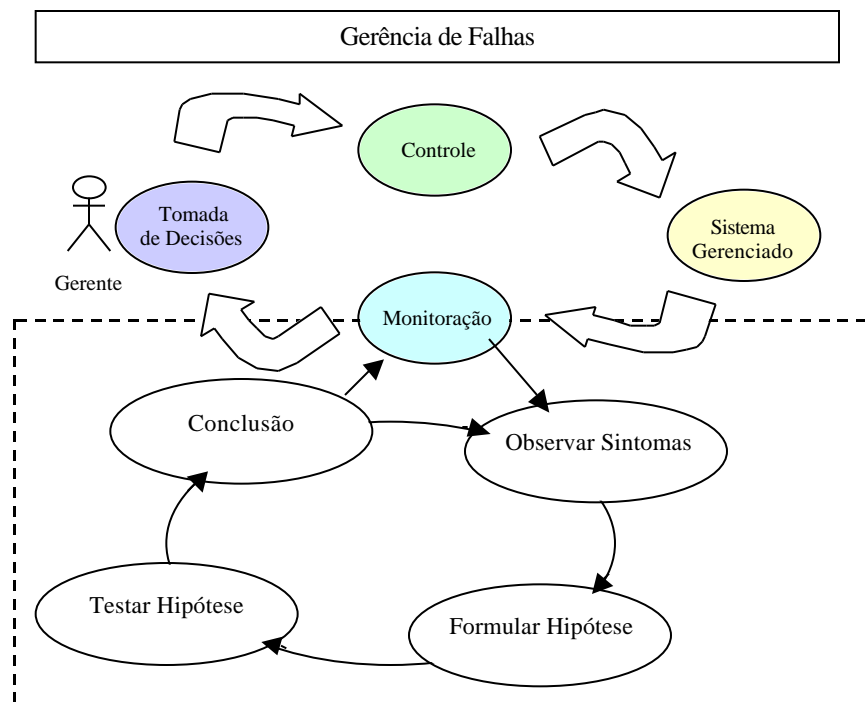


Figura 26 - Sistema de Diagnóstico de Falhas.

#### Observação de Sintomas

Esta é a etapa mais importante desse processo, pois se os sintomas não forem descritos corretamente o diagnóstico poderá ter uma margem grande de erro (da mesma forma que um sistema de diagnóstico médico). Considere que os protocolos de redes normalmente são desenvolvidos para não expor de forma explícita os seus problemas. A maioria dos protocolos incorporam mecanismos de recuperação de falhas e de repetição, o que resulta em um sintoma óbvio tempo longo de resposta da rede, ou mais comum em sistemas bancários temporariamente sem comunicação. Embora esses mecanismos aumentem a



confiabilidade nas redes, eles dificultam o processo de diagnóstico. Portanto, deve-se descobrir o máximo número de sintomas antes de passar para a formulação da hipótese.

Outros itens a descobrir são:

- Extensão e escopo dos sintomas, ou seja, o problema afeta a todos, todos em certa área, ou em indivíduos aleatórios?
- O tempo de manifestação do problema. Se é contínuo ou intermitente e se ocorre regularmente.
- É sempre importante verificar se houve trocas recentes na configuração da rede.
- Descrever todas as variáveis que fazem parte do ambiente, placas de interface de rede, pontes, roteadores, aplicações e sistemas.

### **Formulação da hipótese**

Para construir uma hipótese válida deve-se conhecer o perfil da rede, para saber se o que foi observado é incomum. É impossível ter duas redes idênticas mesmo se forem configuradas iguais sempre manterão características próprias, até mesmo por causa do ambiente em que forem instaladas. Delinear um perfil de uma rede chama-se “baselining”, e existem várias formas de determiná-lo [22][25][26][27][28][66]. Existem vários aspectos a saber sobre a rede para determinar um esboço do seu perfil. São exemplos de informações sobre o perfil da rede:

- Qual a média de utilização da rede, incluindo variações no horário comercial em que normalmente as linhas telefônicas ficam mais congestionadas.
- Quais os tipos de aplicações mais utilizadas e quais as versões?
- Quais os protocolos de rede que são utilizados e características de performance?
- Características do hardware da rede também são importantes, conhecer os controladores de interface, as unidades de acesso ao meio e outros equipamentos de conexão e características de performance.
- Os equipamentos de interconexão entre redes, tais como repetidores, roteadores e servidores da rede, quais fabricantes, quais versões e quais as suas características de performance.

Após identificar os sintomas e possuir um conhecimento sobre o funcionamento normal da rede, formula-se a hipótese. Esta etapa dependerá da experiência de quem trabalha com a rede, ou seja, a heurística sobre falhas na rede. Conhecimento sobre protocolos e aplicações também são importantes nessa fase. Por exemplo, muitas colisões em um barramento Ethernet são frequentemente um resultado de congestionamento ou tráfego excessivo, mas também pode ser por causa de mal funcionamento nos conectores de rede (“transceivers”, em inglês).

### **Testar a hipótese formulada**

Testar a hipótese é o próximo passo. Existem ferramentas para esse propósito, são os analisadores de redes, alguns deles oferecem testes pré-programados, em que cada experimento pode testar uma ou mais hipóteses [19].

### **Elaborar conclusões**

Após realizar os testes necessários é possível realizar a etapa final que é a elaboração de conclusões. Normalmente, se os outros passos são realizados corretamente este torna-se mais simples. Eventos não familiares devem ser expandidos ou revisar os sintomas observados. Este processo é cíclico e pode ter o seu tempo reduzido segundo Dauber [19]. Para isso é necessário possuir um perfil da rede bem elaborado e a cada experimento novo, documentar os sintomas e hipóteses para acrescentar ao histórico do processo de diagnóstico.

### *6.2.2 Exemplos prático*

Para um estabelecimento remoto de conexão, como por exemplo, o acesso doméstico da UFSC. Na maioria das vezes, falha ou por causa de ruído na linha telefônica, (devido ações do meio ambiente) ou por falta de servidor (largura de banda do serviço indisponível). Situando o exemplo dentro do processo de diagnóstico através de heurísticas, tem-se:

#### **Tentativa de estabelecimento de conexão do acesso doméstico.**

<b>Ação: ESTABELECIMENTO DE CONEXÃO DO ACESSO DOMÉSTICO DA UFSC</b>
---

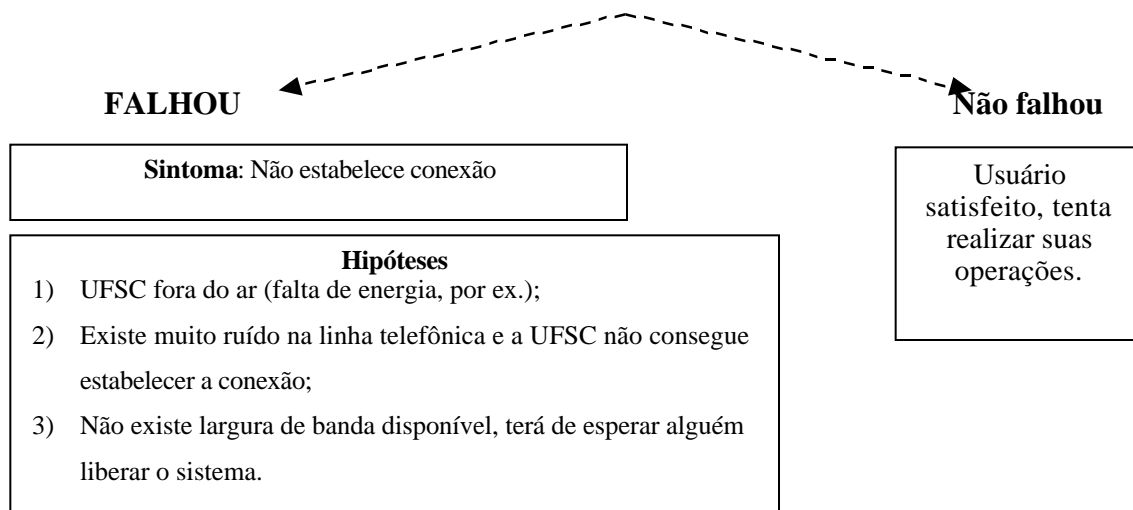


Figura 27 - Exemplo de diagnóstico de falhas.

O próximo passo é testar as hipóteses: isto pode ser feito por tentativa/erro, o sistema pode ter voltado após poucos segundos. Ou alguém pode ter liberado um terminal. Se nenhuma das duas funcionar pode-se trocar o cabo telefônico. Após as tentativas será possível concluir porque a tentativa de estabelecimento da conexão falhou.

Como é possível perceber é um problema que pode ser resolvido de forma declarativa. Existem três hipóteses de porque a rede falhou. Portanto, uma rede neural direta ou um sistema de regras de produção são adequados para solucionar o problema.

### 6.3 Gerência de Desempenho

A Gerência de Desempenho compreende essencialmente dois grupos de atividades: um deles considerado instantâneas (“snapshots”) e outro chamado coleta de estatísticas. O grupo de atividades instantâneas é como ocorre em uma fotografia: em um segundo de tempo, captura-se e congela-se um valor para auxiliar a detectar falhas. O outro grupo de atividades compreende um longo período de coleta de informações, normalmente sobre o tráfego de pacotes, o número de erros de alinhamento, as taxas de utilização de linha, a carga do processador, etc.

Figura 28- Gerência de Desempenho.

O desempenho da rede é um problema dinâmico que depende de diversos parâmetros. É possível realizar a gerência com comportamento pró-ativo onde escolhe-se parâmetros para construir um perfil da rede (“baseline”) [21][26][27]. Estes parâmetros podem servir de entrada de uma RNA com características dinâmicas. As saídas da RNA deverão refletir o comportamento da rede conforme as excitações de entrada da rede. Além do estado ativo e do inativo, é possível considerar um estado intermediário para caracterizar a dinâmica do problema. Ou seja, um estado intermediário que ao ser atingido significa que o sistema poderá ser degradado conforme alguma excitação.

### 6.3.1 Exemplo de Controle de Rotas de Pacotes

Um exemplo prático é construir um grupo de agentes autônomos utilizando redes recorrentes para determinar rotas para transmissão de pacotes. Empregando a idéia de um estado intermediário tem-se:

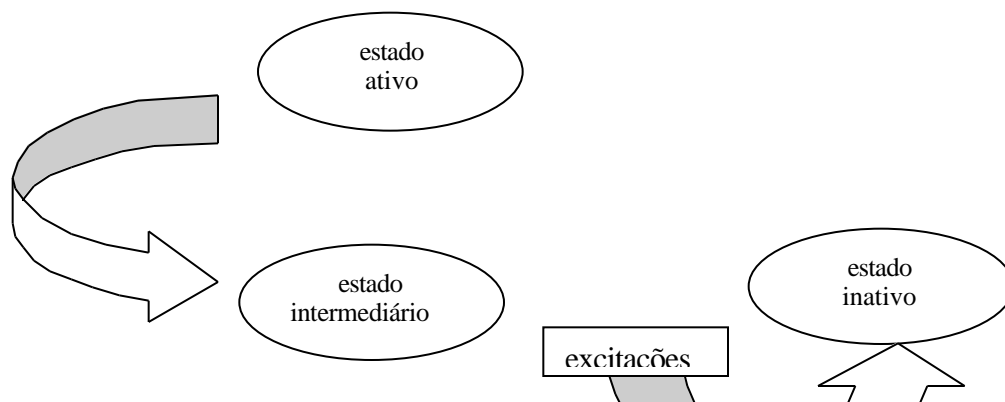


Figura 29 - Gerência dinâmica da rede.

O estado ativo é a rota de transmissão sendo bem utilizada sem taxas de erros. Os valores destas taxas podem crescer e o estado da rede passar a ser o intermediário, Isto pode significar que um grupo de agentes autônomos podem ser disparados para buscar rotas alternativas e impedir que as excitações (que poderão ser: a duração do tempo de resposta ou a taxa de erros) levem a rede para o terceiro estado, o inativo. Ou seja, antes mesmo de uma rota apresentar problemas, as rotas alternativas devem ser testadas.

## 6.4 Gerência de Configuração

A gerência de configuração envolve o controle e o restabelecimento de informações de todos os objetos que fazem parte da rede, sejam eles gerenciáveis ou não. Basicamente, são necessários conhecimentos sobre todos os elementos que fazem parte da rede, uma espécie de inventário. Entre as informações destacam-se a topologia da rede em questão, quais as conexões, o tipo de barramento, a velocidade de operação, quantidade de usuários suportáveis. Estas características aplicam um certo grau de dinâmica ao sistema de gerência, porque o estado da rede sofrerá alterações.

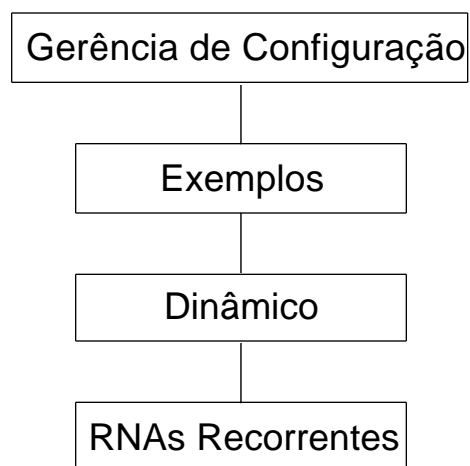


Figura 30- Gerência de configuração.

Considere o fato de acrescentar novas linhas a topologia atual de uma rede. Muitas destas características serão alteradas e a rede não será mais a mesma. Outras alterações de configuração podem ser percebidas com uma simples instalação de um novo aplicativo. Se em um computador pessoal já ocorrem alterações de configuração devido instalações de softwares, você pode imaginar em uma rede de cinquenta ou (muito) mais elementos o que pode acontecer.

#### 6.4.1 Exemplo de Gerência de Configuração

Um exemplo bem original é a criação de um novo laboratório na universidade. Como o controle da rede é feito pelo NPD quando existe a instalação de um novo laboratório, ele deve ser acrescentado ao “backbone” e a rede reconfigurada para reconhecer o novo segmento de rede. Qualquer elemento que seja acrescentado a uma rede, qualquer aplicação e até mesmo a criação de novas contas de usuários possuem características de problema dinâmico.

### 6.5 Gerência de Contabilização

Para compreender a gerência de contabilização basta imaginar uma rede telefônica. Contabilizar seus recursos significa ter o controle do uso de cada recurso de cada usuário. Quando discamos um número no telefone e a ligação é completada começa a tarifação por parte da provedora do serviço. Cada tipo de ligação, normalmente, tem um custo diferente, existem outros fatores que mudam os custos das ligações como horário de utilização da linha. Porém são informações que não alteram o estado da rede.

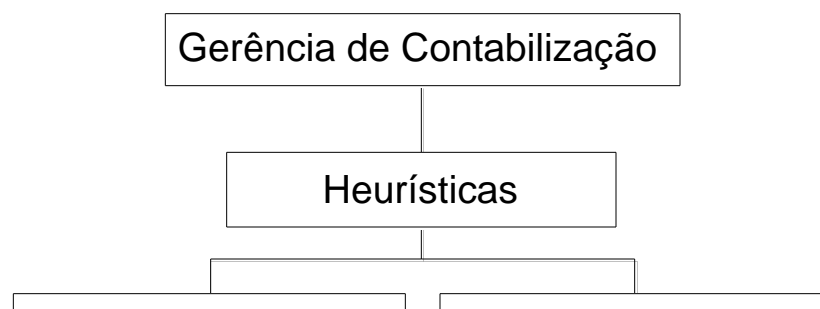


Figura 31- Gerência de Contabilização.

Portanto, pode-se afirmar que problemas de gerência de contabilização não requerem recursos dinâmicos. Um exemplo de controle de recursos em uma rede de computadores pode ser a utilização do disco que está sendo gasta por cada usuário da rede.

## 6.6 Gerência de Segurança

Envolve a criação e monitoração de vários mecanismos de segurança de uma rede. Cabe ressaltar a importância desses mecanismos que possuem duas características distintas. Manter a integridade das informações que atravessam as redes. E garantir o controle de acesso dos recursos disponíveis, já que existe um grau de prioridades quanto a certos recursos da rede. Por exemplo, o cadastro de senhas da rede, só pode ser alterado pelo administrador da rede ou por aquele que possua a senha de acesso.

A

Figura 32 ilustra a aplicação da metodologia. Serão necessários agentes de características dinâmicas para o problema da gerência de segurança.

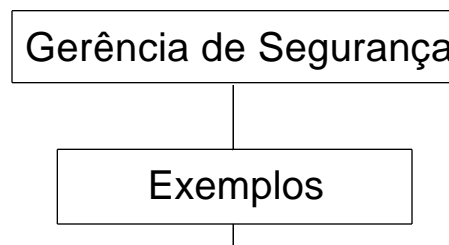


Figura 32- Gerência de segurança.

## 6.7 Classes de Agentes

A partir do estudos realizados algumas classes de agentes autônomos para gerência de redes são propostas:

- Agentes para Tomada de Decisões: uma classe composta por agentes com um alto grau de autonomia, o suficiente para fornecer uma solução para o problema;
- Agentes para Controle da Rede: esta classe de agentes deve alterar valores que controlam a rede, seguindo ordens dos agentes de tomada de decisões;
- Agentes para Monitoração de Parâmetros: uma classe de agentes monitores das informações, essa classe deveria verificar se os valores estão em parâmetros normais ou não;
- Agentes Coletores de Dados: agentes coletores de dados, essa classe de agentes poderia ter baixa autonomia, pois sua função deverá ser coletar as informações sobre o elementos gerenciados.

No entanto, apenas essas informações não são suficientes. Da mesma forma que os agentes da área de gerência de redes, os agentes inteligentes poderão ser locais ou remotos. Poderão estar ou não interligados de alguma forma, deverão possuir algum mecanismo de comunicação, poderão ter bases de dados compartilhadas ou não, e ainda, poderão estar organizados de forma hierárquica, poderão assumir o papel de gerentes. Estas questões deverão ser respondidas durante a fase de implementação do trabalho e serão de grande importância para descrever como solucionar problemas distribuídos.



Assim como os modelos de gerência são formados por agentes e gerentes, e cada um possui características próprias. O agente é responsável por um conjunto de objetos de um nó da rede. E o gerente deve realizar as funções de gerência, monitorando e controlando os objetos através de um protocolo, no caso da Internet, o SNMP. Durante a segunda fase deste trabalho será possível haver alterações destas informações, mas seriam de grande importância ter estas classes implementadas em linguagem JAVA. As seguintes informações poderão ser incorporadas como atributos das classes de agentes autônomos para gerência de redes:

- **Localização:** se os agentes são locais ou remotos;
- **Autonomia:** qual o grau de autonomia que o agente recebeu ao ser criado, dependendo da classe que ele pertencer deverá ser maior ou menor;
- **Bases de Dados:** como a gerência de redes é toda realizada sobre informações da MIB, os agentes poderão compartilhar informações ou não;
- **Organização:** é possível desenvolver um modelo em que os agentes possuam uma organização hierárquica, ou dependendo do problema em questão não seja necessário.

## 6.8 Aspectos de Implementação

Existem duas etapas distintas de implementação do trabalho proposto: a primeira é a criação de agentes autônomos baseados em redes recorrentes que serão realizados a partir de exemplos práticos. A segunda etapa, mais ambiciosa, é a criação de um ambiente para desenvolvimento desses agentes. Estas etapas utilizarão a linguagem C e JAVA.

### 6.8.1 *Vantagens de Utilizar JAVA em ambientes distribuídos*

Para os desconhecedores das características que estas linguagens possuem em ambientes distribuídos, talvez a mais importante seja a portabilidade, é necessário apenas um *browser* para atuar em um ambiente de rede.

Transcrevendo as palavras de Nazim Agoulmine [1], “... a linguagem JAVA em companhia de suas classes, fornecem um ambiente de desenvolvimento portátil,

*interpretado, de alta performance, simples e orientado-a-objetos*”. Desde 1996, existe um esforço por parte das indústrias de integrar a gerência de redes através da Web. Especialmente, entre o SNMP e a linguagem JAVA, através de uma classe de APIs de gerência, chamada JMAPI - *JAVA Management API*.

O ambiente desta linguagem é naturalmente distribuído e integra de forma natural os aspectos de segurança. A vantagem principal é a possibilidade de definir os agentes de forma independente, não proprietária a nenhuma plataforma de gerência de redes, muito menos de sistema operacional e processador. Um programa escrito em JAVA e processado em um Pentium, ou em um Power PC ou até mesmo numa SPARC terá um código interpretável por um browser sobre qualquer um desses equipamentos. Além disso, JAVA é orientada-a-objetos e pode ser integrada com a linguagem C ++ [11][13][42].

Com todas essas características, esta deverá ser a linguagem de implementação dos agentes autônomos. Além disso, como JAVA é composta de pacotes com classes de objetos, poderá ser criada uma classe para agentes autônomos de gerência de redes.

## 6.9 Alguns Experimentos

No sentido de investigar a aplicabilidade das redes recorrentes foram desenvolvidos dois agentes em linguagem C. O primeiro agente tem caráter didático e foi implementado para se estabelecer as diferenças entre utilizar redes diretas e redes recorrentes em problemas dinâmico. O segundo agente, possui características de um interpretador de falhas de uma rede.

### 6.9.1 Agente Paridade

O primeiro agente, o agente paridade ilustra que é possível construir uma solução dinâmica para um determinado problema utilizando redes recursivas. Enquanto que com uma rede direta cria-se uma solução única, ou seja, apenas para uma dimensão do problema.

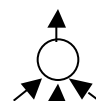
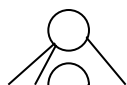


Figura 33 – Duas abordagens diferentes para solucionar o problema da paridade.

Suponha o problema da paridade de 7 bits. Este problema pode ser solucionado facilmente através de uma rede direta conforme a ilustração da Figura 33-B. Uma outra abordagem mais eficiente é utilizar uma rede recorrente capaz de aprender o conceito de paridade (Figura 33-A). Com uma rede neural recorrente, é possível apresentar entradas de problemas de paridade com dimensão variada e obter a resposta correta.

Para treinar a rede recorrente foram utilizados os pesos sinápticos obtidos através do treinamento de uma rede direta, e em seguida aplicado o algoritmo de retropropagação de Roisenberg, citado na Seção 3.5.1. Após o treinamento o conceito de paridade foi aprendido como ilustra a Figura 34. As letras X, Y e U significam o estado anterior, o estado atual da rede e a seqüência de entrada, respectivamente. As letras H1, H2, H3 e H4 apresentam os valores de ativação dos nodos da camada intermediária da rede. O resultado da paridade da seqüência 1 0 1 0 1 é 1.

```

Digite a quantidade da sequencia de entrada:5
Digite a entrada: 1 0 1 0 1
X:0 Y:0 U:1
H1: 0.0109306 H2: 0.985903 H3: 0.923973 H4: 0.0734135
X:0 Y:1 U:0
H1: 0.5 H2: 0.5 H3: 0.5 H4: 0.5
X:1 Y:0 U:1
H1: 0.00010109 H2: 0.900133 H3: 0.0659767 H4: 0.222908
X:0 Y:0 U:0
H1: 0.5 H2: 0.5 H3: 0.5 H4: 0.5
X:0 Y:0 U:1

```

X: 0    Y: 1
--------------

Figura 34 – Execução da rede neural com o conceito de paridade.

### 6.9.2 *Agente de Falhas*

Características de redes como dinâmica, ruído e não-estacionariedade tornam difícil definir o que é uma falha em um ambiente de rede. Diagnóstico é a identificação de uma condição através de sinais, sintomas ou características distintas [48]. No sentido de desenvolver uma IAD, pretende-se atacar o problema decompondo-o em problemas menores e com características distintas, desenvolvendo agentes autônomos que ajam de forma independente e de modo cooperativo na solução do problema de falhas na rede.

Um destes agentes autônomos está sendo desenvolvido para classificar um evento de rede como Crítico, Falha Simples ou Sem Falha. O agente funciona como um interpretador de eventos de rede e informa quando um problema é detectado ou através de registros da rede ou por *polling*. Encontrando a falha o agente poderá ter autonomia para agir ou simplesmente gerar uma alerta ao gerente ou usuário da estação (Figura 35).

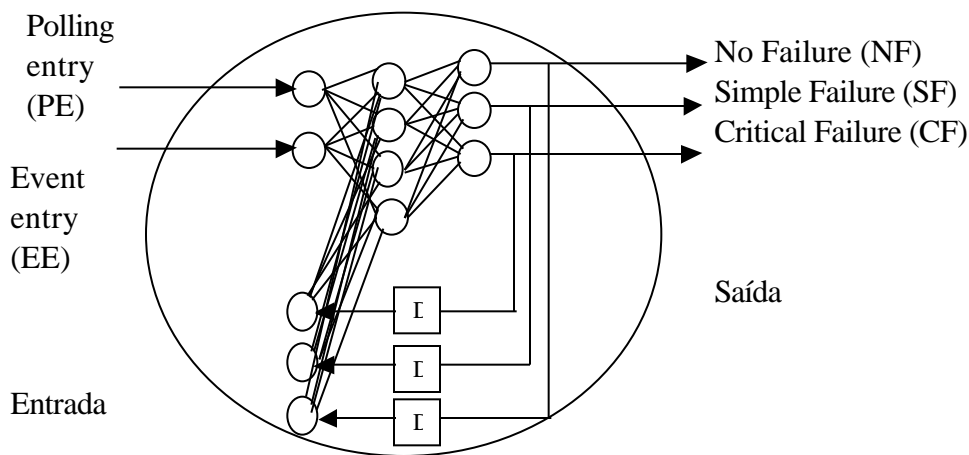


Figura 35 – Agente para detecção de falhas em uma rede.

Os sensores e os atuadores são o contato com o mundo exterior do agente. Respostas instantâneas e bem definidas a estímulos recebidos pelos sensores são conhecidos como taxias e reflexos. Esta forma de comportamento pode ser descrita por uma

função combinacional ou contínua apenas dos valores presentes nos sensores. Este comportamento pode ser implementado por uma rede neural direta, no nosso caso como a dinâmica é importante deve-se utilizar neurônios dinâmicos. A máquina de estados finita que representa o comportamento do agente está ilustrada pela Figura 36.

## 6.10 Outros AAs para Gerência de Redes

Koch [39] apresenta alguns tipos de agentes autônomos que seriam interessantes no auxílio à solução de problemas distribuídos, mais precisamente de gerência de redes de computadores. Porém, os agentes de gerência não foram implementados com redes neurais, o trabalho apresenta alguns AAs desenvolvidos com regras de produção. O autor apresenta ainda, uma pequena biblioteca para programação de uma rede neural direta, não apresentando resultados mais expressivos. Os agentes propostos são:

- Agentes facilitadores de comunicação, responsáveis pela comunicação entre agentes dentro do esquema de quadro negro utilizado naquele trabalho;
- Agentes para análise de dados, como exemplo, o autor propõe um agente para analisar tráfego de pacotes IP em concentradores, verificando os pontos de acesso com maior volume de informações;
- Agentes para consolidar dados, agentes para agrupar os dados de um ou mais coletores de dados com o objetivo de estipular estatísticas sobre a rede;
- Agentes para prever dados futuros, com a característica das RNAs serem utilizadas em predição de séries temporais, o autor sugere um agente para previsão de falhas;

- Agentes para classificação de valores, poderia ser criado um agente para classificar o fluxo de informações em alto, médio ou baixo.

## Capítulo 7

# Considerações finais

Este trabalho propõe uma metodologia para encontrar soluções adequadas aos problemas distribuídos na área da gerência de redes. Em muitos casos de uso de redes neurais, tem sido comum encontrar soluções estáticas para automatizar o processo. Dependendo do escopo do problema, tais soluções muitas vezes não são adequadas, pois existem problemas com características estáticas e outros com características dinâmicas. A essência do trabalho é explorar os dois tipos de soluções (dinâmicas ou estáticas) para problemas distribuídos, e como aplicar as técnicas de IA (Regras de Produção, RNAs diretas ou RNAs recorrentes) na construção de agentes autônomos

Conceitos importantes de gerência de redes tais como o modelo de gerência OSI enfatizando as cinco áreas funcionais de gerência (falhas, configuração, desempenho, contabilização e segurança) e o modelo utilizado na Internet foram apresentados. Os conceitos apresentados limitam-se ao escopo do trabalho. As cinco áreas funcionais foram utilizadas para distribuir os problemas da gerência. O modelo da Internet fornece conceito de agentes e gerentes de forma simplificada para compreender como funciona o processo de gerência de redes.

Em seguida foram apresentadas noções de redes neurais artificiais. Os modelos de neurônios mais conhecidos, algumas topologias de redes neurais, tipos de aprendizado, as redes neurais recorrentes. Além disso, foram apresentados conceitos sobre problemas com características estáticas, dinâmicas e problemas distribuídos. Ainda sobre redes neurais, questões sobre computabilidade e complexidade de RNAs foram apresentadas. Este estudo sobre redes neurais se faz necessário para compreender porque aplicar redes diretas ou recorrentes na solução de problemas.

Conceitos de inteligência artificial distribuída, tais como, agentes autônomos, modelos de agentes, sistemas de raciocínio distribuído, mecanismos de controle, coordenação e comunicação de agentes foram abordados. A importância de construir soluções de gerência de redes utilizando a abordagem distribuída da inteligência artificial também foi discutida.

Estes conceitos foram utilizados para construir uma metodologia para desenvolvimento de agentes autônomos para gerência de redes de computadores. Antes de propor uma solução o problema de gerência deve ser analisado e classificado como estático ou dinâmico. Além disso, deve ser considerado o comportamento das soluções de gerência que serão adotadas. Tais soluções poderão ser reativas ou pró-ativas. Gerência reativa, é quando as ações são determinadas após a degradação do sistema. Ao contrário, gerência pró-ativa envolve uma série de ações que previnem a rede da degradação do sistema. Cada um desses comportamentos exigem soluções apropriadas. Neste sentido a gerência de redes foi subdividida nas cinco áreas funcionais proposta pelo modelo OSI. Cada uma foi então analisada e proposta uma solução.

A gerência de falhas, normalmente, possui soluções declarativas (no caso da gerência reativa). Se existirem heurísticas sobre as falhas é mais apropriado utilizar RNAs diretas ou regras de produção para representação do conhecimento. No entanto, no caso de gerência pró-ativa, é preciso ter outra visão da situação. Se o objetivo é prevenir falhas na rede é preciso identificar os problemas antes que a rede seja afetada. Neste caso, é proposta uma solução dinâmica como o uso de redes recorrentes.

Para garantir a qualidade do serviço a gerência de desempenho deve ser pró-ativa. Além disso, possui características dinâmicas, existem atributos como o volume de informações transitando pela rede, ou o número de usuários utilizando a rede em determinado momento, ou ainda a quantidade de erros que se alteram a todo instante. Estes parâmetros podem alterar o estado da rede, oscilando entre bom, normal ou ruim, por exemplo (tratando de qualidade do serviço oferecido aos usuários). Nesse caso deve-se adotar uma solução com características dinâmicas.

Outra área com características dinâmicas é a de configuração. Configuração abrange funções de alteração de topologia da rede por eventuais trocas de equipamentos ou



expansão da rede. Como a tecnologia modifica a todo instante, e a necessidade por novos equipamento e nós da rede é uma constante, deve-se optar por soluções dinâmicas.

Para facilitar a compreensão do que trata a gerência de contabilização basta pensar na rede telefônica. Contabilizar os recursos é semelhante a tarifação processada pelo serviço telefônico. As ligações interurbanas e para telefones celulares tem um ônus maior do que para os telefones convencionais. Outro atributo considerado é o tempo de utilização. Estas funções não possuem características dinâmicas, portanto, são bem solucionadas por sistemas declarativos.

Para finalizar a área de gerência de segurança. Novamente, é possível verificar características dinâmicas. A mais óbvia talvez seja o número de usuários para o controle de acesso. É fácil verificar em uma rede universitária o fluxo de alunos novos e alunos que deixam a universidade, gerando assim uma quantidade variável de usuários que tem acesso à rede.

Nesse sentido propõe-se o desenvolvimento de agentes autônomos para a gerência de redes. Utilizando esta metodologia será possível desenvolver agentes com características inteligentes com ferramentas adequadas. Ou seja, problemas estáticos devem ser solucionados por ferramentas estáticas e problemas dinâmicos por ferramentas com características dinâmicas.

Os principais aspectos abordados que permitem estabelecer as bases sobre o qual se desenvolveu os pontos originais deste trabalho foram:

- A partir da análise de uma RNA, conclui-se que ao invés de um produto escalar for utilizada uma superfície com grau proporcional ao produto com duas dimensões, isto é, se a confluência for uma parábola com parâmetros abertura e vértice ajustados, ter-se-á a capacidade de separar não linearmente um conjunto de pontos não separável;
- Seguindo a linha de Azevedo [21], que utilizou a teoria de sistemas para formalizar as RNAs, e Roisenberg que utilizou a mesma ferramenta para apresentar um método formal de AAs. Foi formalizado o conceito de multi-agentes como um sistema complexo, que é um conjunto de sistemas dinâmicos (AAs na definição de Roisenberg) interligados.

- Além disso, foram analisadas as áreas de gerência e propostas soluções adequadas conforme o tipo de problema.

## 7.1 Cronograma

Para o próximo ano, seguem as seguintes atividades (Tabela III).

1. Definir o escopo do trabalho dentro da rede do GPEB;
2. Coletar exemplos de problemas distribuídos mais frequentes no ambiente de teste;
3. Definir os agentes que deverão ser implementados;
4. Implementá-los em JAVA e C e testá-los;
5. Observar e analisar o comportamento dos agentes;
6. Definir conceitos para a solução dos problemas distribuídos;
7. Publicar resultados;
8. Escrever a documentação final (tese);
9. Preparar apresentação.

Tabela III – Cronograma para as atividades restantes do trabalho.

2000	Jan.	Fev.	Mar.	Abr.	Mai.	Jun.	Jul.	Ago.	Set.	Out.	Nov.	Dez.
1												
2												
3												
4												
2000/2001	Jan.	Fev.	Mar.	Abr.	Mai.	Jun.	Jul.	Ago.	Set.	Out.	Nov.	Dez.
5												
6												
7												
8												

## Referências Bibliográficas

- [1] AGOULMINE, N. Les Architectures Distribuees de Gestion de Reseaux et de Services. 2o Seminário Franco Brasileiro de Sistemas Informáticos Distribuídos. In: Anais do SFBSID'97 – Arquiteturas Multimídias para as Telecomunicações, pp. 55-80, Fortaleza, CE, nov., 1997.
- [2] ANGELO, F.E.V. “Monitoração da RMON MIB utilizando a Plataforma de Gerência SunNet Manager”, Projeto de Conclusão de Curso (INE, UFSC), Florianópolis, SC, 1995.
- [3] ARBIB, M.A. Brains, Machines and Mathematics, McGraw-Hill,1964.
- [4] BARRETO, J.M, M. Roisenberg, F.M. de Azevedo. Developing Artificial Neural Networks for Autonomous Agents Using Evolutionary Programming. In: Proceedings of the IASTED International Conference Artificial Intelligence and Soft Computing. Cancun, México, May, 27-30, 1998. Pp. 283-286.
- [5] BARRETO, J.M. Inteligência Artificial No Limiar do Século XXI Abordagem Híbrida da Inteligência Artificial Simbólica, Conexionista e Evolucionária. Duplic Edições, Florianópolis, 1997.
- [6] BARRETO, J.M. “Inteligência Artificial No limiar do Século XXI Abordagem Híbrida Simbólica, Conexionista e Evolucionária. 2ª edição revista e aumentada. Florianópolis, 1999.
- [7] BAZZAN, A.L. Coordination Among Individually-Motivated Agents: An Evolutioary Approach. In: Advances in Artificial Intelligence. Proceedings of 13th Brazilian Symposium on Artificial Intelligence, SBIA'96, D.L. Borges, C.A.A. Kaestner (eds.), Curitiba, Brazil, October, 1996.
- [8] BELO, O. , RIBEIRO, A. A Distributed Multi-Agent System Environment A Web Approach. In: Proceedings of IASTED International Conference, Artificial Intelligence, Expert Systems and Neural Networks, Aug., 1996.
- [9] BRASIL, L. M. “Algoritmos de Extração de Regras de Redes Neurais Artificiais”. In: Anais do III Congresso Brasileiro de Redes Neurais, Santa Catarina, 1997.
- [10] BROOKS, R. A.. Artificial Life and Real Robots In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 3-9, 2nd print., 1994. The MIT Press.
- [11] CAMPIONE, M. , WALRATH, K. The JAVA™ Tutorial – Object-Oriented Programming for the Internet. Addison-Wesley,1996.

- [12] CARISSIMI, L.S, BASTOS, E.L., WESTPHALL, C.B. “Definição de Gerentes e Agentes para Gerência de Redes”. In: Anais do XXI Seminário Integrado de Software e Hardware, pp. 397-410, 1993.
- [13] CHAN, P. The Java™ Developers Almanac 1999. Addison-Wesley, 1999.
- [14] COLAJANNI, M., YU, P.S., DIAS, M. “Analysis of Task Assignment Policies in Scalable Distributed Web-Server Systems”, IEEE Transactions on Parallel and Distributed Systems, Vol. 9, No. 6, Jun., 1998.
- [15] COMER, D.E., STEVENS, D.L. Internetworking with TCP/IP vol. II: Design, Implementation and Internals. Prentice-Hall, 1991.
- [16] COMER, D.E., STEVENS, D.L. Internetworking with TCP/IP vol. III: Client-Server Programming and Applications. Prentice-Hall, 1993.
- [17] D.E. Comer. Internetworking with TCP/IP vol. I: Principles, Protocols and Architecture. Prentice-Hall, 1988.
- [18] DAO, B. V. , DUATO, J. , YALAMANCHILI, S. “Dinamically Configurable Message Flow Control for Fault-Tolerant Routing”. IEEE Transactions on Parallel and Distributed Systems, vol. 10, no. 1, Jan, 1999.
- [19] DAUBER, S. M. “Finding Fault”. BYTE, pp. 207-214, Mar., 1991.
- [20] DAVIDGE, R. Looking at Life. In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 448-455, 2nd print., 1994. The MIT Press.
- [21] DE AZEVEDO, F. M. “Uma Proposta de Modelos Formais de Neurônios e Redes Neurais Artificiais”. In: Anais do III Congresso Brasileiro de Redes Neurais, IV Escola de Redes Neurais, pp. 503-514, Florianópolis, SC, Julho, 1997.
- [22] DE FRANCESCHI, A.S.M. "Uma Aplicação de Desempenho para Validar a Gerência Pró-ativa de Redes", Dissertação de Mestrado (CPGCC, UFSC), Florianópolis, SC, Fev., 1996.
- [23] DE FRANCESCHI, A.S.M., BARRETO, J.M. Autonomous Agents based on Recurrent Neural Networks Applied to Computer Network Management. In: ISAS'99 – 5th International Conference on Information Systems, Analysis and Synthesis, Orlando, Florida, 31/07-04/08/1999.
- [24] DE FRANCESCHI, A.S.M., BARRETO, J.M. Distributed Problem Solving Based on Recurrent Neural Networks Applied to Computer Network Management In: ICT'99 IEEE/International Conference on Telecommunications, Cheju, Korea,15/06-18/06/1999.
- [25] DE FRANCESCHI, A.S.M., DA ROCHA, M. A., WEBER, H. L., WESTPHALL C. B., Proactive Network Management Using Remote Monitoring and Artificial Intelligence Techniques. In: Proceedings of IEEE International Symposium on Computer Communications, Alexandria, Egypt, 1997.
- [26] DE FRANCESCHI, A.S.M., DA ROCHA, M. A., WEBER, H. L., WESTPHALL C. B. Employing Remote Monitoring and Artificial Intelligence Techniques to Develop the Proactive Network Management. In: Proceedings of IEEE International Workshop on

- Application of Neural Networks in Telecommunications, Melbourne, Australia, 1997. pp.116-123.
- [27] DE FRANCESCHI, A.S.M., KORMANN, L. F., WESTPHALL, C.B. Performance Evaluation for Proactive Network Management. In: Proceedings of the IEEE/ICC'96 International on Communications Conference, Vol. I, Dallas, Texas, Jun., 1996.
- [28] DE FRANCESCHI, A.S.M., KORMANN, L. F., WESTPHALL, C.B. A Performance Application for Proactive Network Management In: Proceedings of the Second IEEE International Workshop on Management Systems, Toronto, Canada, Jun., 1996. Pp. 15-20.
- [29] FINN, T. LABROU, Y., MAYFIELD, J. KQML as an agent communication language, finn@cs.umbc.edu, USA, 1995.
- [30] FURTADO, V. H., VINÍCIUS, M. “Desenvolvimento de Novas Funções de Gerência utilizando a Plataforma de Gerência SunNet Manager”, Projeto de Conclusão de Curso (INE, UFSC), Florianópolis, SC, 1995.
- [31] GARDNER, R.A., GARDNER, B.T. Feedforward: the Ethological Basis of Animal Learning In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 399-410, 2nd print., 1994. The MIT Press.
- [32] GOLDSZMIDT, G., YEMINI, Y. Evaluation Management Decisions via Delegation. In: Proceedings of IFIP/ISINM'93 – International Symposium on Integrated Network Management, Apr., 1993.
- [33] HOFFNER, Y. The Management of Monitoring in Object-based Distributed Systems. In: Proceedings of Integrated Network Management, III, H.-G. Hegering and Y. Yemini(eds.), 1993.
- [34] HOFSTADTER, D. R. Fluid Concepts E Creative Analogies. Basic Books, 1995.
- [35] HOFSTADTER, D. R. Gödel, Escher, Bach: an Eternal Golden Braid. Vintage Books, may, 1989.
- [36] HORN, J. Measuring the Evolving Complexity of Stimulus-Response Organisms. In: In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 365-374, 2nd print., 1994. The MIT Press.
- [37] HUANG, R., MA, J. , TSUBOI, E. Communication Network Design via a Genetic Algorithm based Learning Algorithm. University of Aizu. In: Proceedings of IASTED International Conference, Artificial Intelligence, Expert Systems and Neural Networks, Aug., 1996.
- [38] KAELBLING, L.P. An Adaptable Mobile Robot. In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 41-47, 2nd print., 1994. The MIT Press.
- [39] KOCH, F. “Agentes Autônomos para Gerenciamento de Redes de Computadores”. Dissertação de Mestrado (CPGCC, UFSC), Florianópolis, out, 1997.
- [40] LAWRENCE, J. Untangling Neural Nets, Dr. Dobb's Journal, Apr., 1990. Pp.38-44.

- [41] LEINWAND, A., CONROY, K.F. Network Management – A Practical Perspective. Addison-Wesley, 2nd ed., 1996.
- [42] LEMAY, L., PERKINS, C.L. Teachyourself JAVA in 21 days. SamsNet Publishing, 1996.
- [43] LÉVY, P. As Tecnologias da Inteligência - o futuro do pensamento na era da informática, Tradução Carlos Irineu da Costa, editora34, Rio de Janeiro, 1993.
- [44] MADRUGA, E.L. “Ferramentas de Apoio à Gerência de Falhas e Desempenho em Contexto Distribuído”. Dissertação de Mestrado (CPGCC, UFRGS), 1994.
- [45] MAES, P. Learning Behavior Networks from Experience. In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 48-57, 2nd print., 1994. The MIT Press.
- [46] MANDRICK, B. Selectionist Systems as Cognitive Systems. In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 441-447, 2nd print., 1994. The MIT Press.
- [47] MANIONE, R., MONTANARI, F. Validation and Extension of Fault Management Applications through Environment Simulation. In: Proceedings of IFIP/ISINM’95 – International Symposium on Integrated Network Management, 1995.
- [48] MAXION, R.A., FEATHER, F.E. A Case Study of Ethernet Anomalies in a Distributed Computing Environment, IEEE Transactions on Reliability, vol.39, no.4, Oct, 1990.
- [49] MCCULLOCH, W.S., PITTS, W.H. A Logical Calculus of Ideas Immanent in Nervous Activity, Bull. of Mathematical Biophysics, vol.5, pp.115-133, 1943.
- [50] MINSKY, M.L., PAPERT, S. A. Perceptrons: an introduction to computational geometry, MIT Press, 1988
- [51] MOTOROLA CODEX, “The Basics Book of OSI and Network Management”, Motorola University Press, Addison-Wesley, 1993.
- [52] MOURATA, M., TAKAGI, H. Two-Layer Modelling for Local Area Networks. IEEE Transactions on Communications, vol. 36, no. 9, sep., 1988. Pp. 1022-1034.
- [53] MOUSSALE, N.M., VICCARI, R.M., CORREA M. Intelligent tutoring systems modeled through the mental states. In: Advances in Artificial Intelligence, Proceedings of 13th Brazilian Symposium on Artificial Intelligence, SBIA’96, D.L. Borges,
- [54] NOGUEIRA, J.M.S. “O Estado Atual da Pesquisa e Desenvolvimento em Gerenciamento de Redes no Brasil”. 2o Seminário Franco Brasileiro de Sistemas Informáticos Distribuídos. In: Anais do SFBSID’97 – Arquiteturas Multimídias para as Telecomunicações, pp. 81-98, Fortaleza, CE, nov., 1997.
- [55] PFEIFER, R., VERSCHURE, P. Distributed Adaptive Control: A Paradigm for Designing Autonomous Agents. In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 21-29, 2nd print., 1994. The MIT Press.
- [56] POLYA, G. A Arte de Resolver Problemas. Interciência. Tradução de “How to solve it: A New Aspect of Mathematical Method”, Princeton University Press, Rio de Janeiro, 1975.

- [57] PRAS, A. Network Management Architectures. CTIT Ph. Thesis no. 95-02, Centre for Telematics and Information Technology, The Netherlands, 1995.
- [58] REILLY, D.L., COOPER, L.N. An Overview of Neural Networks: Early Models to Real World Systems. In: An Introduction to Neural and Eletronic Networks, Academic Press, 1990. Pp.227-247.
- [59] RICH,E., KNIGHT, K. Inteligência Artificial. Tradução Maria Claudia S. R. Ratto. 2a edição, McGraw-Hill : São Paulo, 1993.
- [60] ROISENBERG, M. “Emergência da Inteligência em Agentes Autônomos através de Modelos Inspirados na Natureza”. Tese de Doutorado (CPGEEL, UFSC), Florianópolis, 1998.
- [61] ROISENBERG, M., BARRETO, J.M, DE AZEVEDO, F.M, BRASIL, L.M. On a Formal Concept of Autonomous Agents. In: Proceedings of the 16th IASTED International Conference Applied Informatics, Germany, February, 1998.
- [62] ROISENBERG, M., BARRETO, J.M, DE AZEVEDO, F.M. “Uma Proposta de Modelização para Agentes autônomos Baseada na Teoria de Sistemas”. In: Anais do 3o Simpósio Brasileiro de Automação Inteligente, Vitória, ES, 3 a 5 de setembro, 1997. pp.500-507.
- [63] ROISENBERG, M., BARRETO, J.M, DE AZEVEDO, F.M. A Neural Network that Implements Reactive Behavored Autonomous Agents. In: Proceedings of the IASTED International Conference Artificial Intelligence, Expert Systems and Neural Networks. Honolulu, Hawaii, USA, Aug. 19-21, 1996. Pp. 245-248.
- [64] ROSE, M.T. The Simple Book – An Indroduction to Management of TCP/IP based Internets. Prentice-Hall, USA, 1994.
- [65] SCALABRIN, E.E., VANDENBERGHE, L., DE AZEVEDO, H. , BARTHÈS, J-P.A. A generic model of cognitive agent to develop open systems. In: Advances in Artificial Intelligence. Proceedings of 13th Brazilian Symposium on Artificial Intelligence, SBIA´96, D.L. Borges, C.A.A. Kaestner (eds.), Curitiba, Brazil, October, 1996.
- [66] SCHWEITZER, C.M. “Desenvolvimento de Baselines com o Uso de Simulação para Automação da Gerência de Redes”, Projeto de Conclusão de Curso (INE, UFSC), Florianópolis, SC, 1996.
- [67] SMITHERS, T. Taking Eliminative Materialism Seriously: A Methodology for Autonomous Systems Reasearch. In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 31-39, 2nd print., 1994. The MIT Press.
- [68] SPINNEY, B. Ethernet Tips and Techniques: for designinig, Installing and Troubleshooting your Ethernet Network, 2nd edition, CBM Books, 1995.
- [69] TANEMBAUM, A. S. “Computer Networks”, Prentice-Hall, 2<sup>nd</sup> edition.
- [70] VIEIRA, E.S., SARI, S.T. “Prototipação de um Subagente Adaptativo Baseado em Redes Neurais”. In: Anais do XIV SBRC – Simpósio Brasileiro de Redes de Computadores, pp. 140-156, Fortaleza, 1996.

- [71] VORUGANTI, R.R. A Global Network Management for the 90's. IEEE Communications Magazine, pp. 74-83, Aug., 1994.
- [72] WEBB, B. ,SMITHERS, T. "The Connection between AI And Biology in the Study of Behaviour". In: Proceedings of First European Conference On Artificial Life. F.J. Varela, P. Bourguine (eds), A Bradford Book, pp. 421-427, 2nd print., 1994. The MIT Press.
- [73] YOO, Y.-D. A Multi-agent Approach for a Multi-layered Student Model. In: Proceedings of the IASTED International Conference On Artificial Intelligence, Expert Systems and Neural Networks, Aug, 1996.
- [74] ZHENG, Q., SHIN, K.G. "Fault-Tolerant Real-Time Communication in Distributed Computing Systems". IEEE Transactions on Parallel and Distributed Systems, Vol. 9, N0.5, May, 1998.
- [75] ZOMAYA, A.Y., CLEMENTS, M., OLARIU, S. "A Framework for Reinforcement-Based Scheduling in Parallel Processor Systems". IEEE Transactions on Parallel and Distributed Systems, Vol. 9, No. 3, Mar, 1998.
- [76] ZURADA, J.M. Introduction to Artificial Neural Systems, West Publishing Co., USA, 1992.



# Glossário

## **Agente (gerência de redes)**

Existem várias definições para agentes em gerência de redes. Agente é um processo que troca informações de gerência entre a estação gerente e os objetos gerenciados. processos individuais que realizam tarefas e as comunicam a outros processos chamados gerentes

## **Agentes autônomos**

Agentes autônomos de software são programas, dotados de inteligência que agem como um parceiro melhorando a eficiência do trabalho do ser humano. Pode operar em um ambiente de apenas um computador ou estar distribuído, nesse caso vários agentes interagindo em um ambiente distribuído formam uma IA Distribuída. Em hardware, são sistemas que possuem uma interação duradoura com um ambiente dinâmico externo, estando normalmente instalados fisicamente em mecanismos. São dotados de rodas ou outros meios de locomoção e sensores projetados para funcionarem por longos períodos de tempo, operando em um ciclo que envolve aquisição de informações do ambiente e a geração de dados de saída.

## **Algoritmo de aprendizado**

O aprendizado em RNAs consiste no ajuste dos valores dos pesos das conexões. Quando esses valores são desconhecidos utiliza-se um algoritmo (ou regra) de aprendizado para aproximá-los. Entre as principais regras de aprendizado destacam-se: Aprendizado Hebbiano, Regra Delta, Retropropagação (ou “Backpropagation”, em inglês), Aprendizado competitivo, Aprendizado por Reforço ou Aprendizado Aleatório.

## **Barramento**

Meio físico que interconecta os equipamentos de uma rede de computadores. Ver Ethernet.

## **Bridge**

Roteador que conecta duas ou mais redes e transmite os pacotes de uma para a outra. Normalmente operam a nível físico da rede, por exemplo, uma ponte que liga duas redes Ethernet e transmite de um cabo para outro pacotes que não são locais. Pontes diferem de repetidores porque armazenam e enviam pacotes completos enquanto repetidores transmitem sinais elétricos.

## **CMIP (Common Management Information Protocol)**

Protocolo OSI para gerência de redes.

## **CMISE (Common Management Information Service)**

Serviço oferecido pelo CMIP.

## **Conexão**

Ligação lógica entre dois ou mais usuários de um serviço.

## **CSMA/CD (Carrier Sense Multiple Access with Collision Detection)**

É um dos protocolos de transmissão de redes locais mais utilizados, em que períodos de transmissão são seguidos de intervalos de contenção. Quando duas estações escutam o meio e “sentem” que está desocupado e tentam transmitir simultaneamente ambas detectam a colisão, param de transmitir e aguardam um período de contenção e tentam transmitir novamente.

## **DME (Distributed Management Environment)**

É um conjunto de especificações para produtos de gerência distribuída, proposta pela OSF (Open Software Foundation). Fornece uma estrutura que habilita um sistema consistente e um esquema de gerência para ambiente distribuído com produto de diversos fabricantes.

## **Ethernet**

Tecnologia de rede local inventada pela Xerox Corporation Palo Alto Research Center. Um Ethernet é um cabo coaxial passivo, em que as interconexões mantêm os componentes ativos. É um sistema de entrega do melhor esforço (best-effort delivery) que utiliza CSMA/CD. A Xerox Corporation, a Digital Equipment Corporation e a Intel Corporation publicaram padrões para Ethernet a 10Mbps.

## **Gateway (Roteador)**

Computador dedicado que liga dois ou mais redes e roteia pacotes de um para outro. Em particular, um gateway Internet roteia datagramas IP entre as redes que ele interconecta. Gateways também roteiam pacotes para outros gateways até que os pacotes sejam entregues no destinatário final diretamente através do meio físico. O termo normalmente é atribuído a máquinas que transferem informações de uma rede a outra, tal como “mail gateway”.

**Gerente (gerência de redes)**

É o processo que troca informações de gerência com o agente. Nas plataformas de gerência normalmente é ativado pelo administrador da rede para tomadas de decisão.

**Host (no ambiente de redes)**

É um sistema terminal (ou final de linha).

**IETF (Internet Engineering Task Force)**

Força tarefa encarregada das resoluções normativas da Internet. São os encarregados de elaborar e discutir os RFCs (Request for Comments).

**IP (Internet Protocol)**

Protocolo que oferece serviço não orientado a conexão correspondendo a camada de transporte do conjunto de protocolos da Internet.

**ISO (International Standards Organization)**

Organização internacional que RASCUNHA, discute, propõe e especifica padrões para protocolos de redes. A ISO é mais conhecida pelo modelo de referência das sete camadas que descreve uma organização conceitual de protocolos.

**JAVA**

Linguagem para desenvolvimento de aplicações para World Wide Web.

**MIB (Management Information Base)**

Uma coleção de objetos que podem ser acessados através do protocolo de gerência de redes.

**Neurônio**

No contexto biológico são células que compõem o sistema nervoso. É o sistema que comanda o funcionamento de todos nossos órgãos, aparelhos e sistemas. No contexto da Inteligência Artificial, são chamados de neurônios artificiais. É uma técnica de solução de problemas inspirada nos neurônios biológicos. Um neurônio apenas não pode ser utilizado para solucionar nada. Unidos entre vários formam as redes neurais (ou neuronais) artificiais.

**NIST (National Institute of Standards and Technology)**

Órgão do Departamento de Comércio dos EUA encarregado de padronizações. Antigo National Bureau of Standards.

## **OMNIPoints (Open Management Interoperability Points)**

Definem um conjunto de padrões que podem ser usados por implementadores e fabricantes de redes. Suas versões são liberadas de dois em dois anos. O OMNIPoint 1 oferece o SNMP e o CMIP como duas opções de implementação. No sentido de permitir uma maior flexibilidade os fabricantes são encorajados pelas resoluções dos OMNIPoints.

## **OSI (Open Systems Interconnection)**

Esforço internacional para facilitar as comunicações entre computadores de diferentes fabricantes.

## **Pesos sinápticos**

Excitações (podem ser positivas ou negativas) de entrada dos neurônios artificiais.

## **Plataforma de gerência**

Programas de gerência de redes que abrangem as cinco áreas funcionais do modelo de referência OSI.

## **Protocolo**

Descrição formal de formatos de mensagens e regras que duas ou mais máquinas devem seguir para trocar suas mensagens. Os protocolos podem descrever tanto baixos níveis de detalhamento quanto a nível de aplicação.

## **Regras de Produção**

Técnica declarativa e procedimental utilizada para representação do conhecimento em programas de Inteligência Artificial Simbólica. Normalmente, encontram-se na forma SE <condição ou condições> ENTÃO <realiza determinada ação>.

## **RFC (Request for Comments)**

Série de documentos que descrevem o conjunto de protocolos da Internet e experimentos relacionados.

## **Serviço**

Representa um conjunto de funções oferecidos ao usuário por um provedor. O serviço torna-se disponível através do SAP (Service Access Point). O provedor é visto como uma entidade abstrata e é representado pela interface do SAP. Os serviços podem ser orientado à conexão (possui três fases bem definidas, estabelecimento da conexão, transferência dos dados e

liberação da conexão). Ou não orientado à conexão que possui uma única fase de endereçamento adicionada a transferência de dados.

### **SNMP (Simple Network Management Protocol)**

Protocolo de aplicação que fornece serviço de gerência no conjunto de protocolos Internet.

### **SNMPv2 (Simple Network Management Protocol version 2)**

Versão menos popular do SNMP. Deve-se ao fato de existir MIBs do SNMP que não são compatíveis com a nova versão. Além disso, é mais complicado e com custos mais elevados para implantação. A vantagem é permitir de interações entre gerentes e novas operações de gerência, tal como, GetBulkRequest, que permite obter vários dados apenas com uma operação.

### **TCP (Transmission Control Protocol)**

Protocolo de transporte que fornece o serviço orientado a conexão no conjunto de protocolos da Internet.

### **Threshold (ou limiar)**

O threshold ou limiar é um importante conceito em gerência de redes. É uma forma de prevenir a rede de algum tipo de problema. Se este limiar for ultrapassado o gerente da rede é informado através de um alarme. Em alguns dispositivos de rede (tais como alguns modems) tem limiares de performance pré-determinados e alguns mecanismos de testes para assegurar que o desempenho do dispositivo seja satisfatório.

### **TMN (Telecommunications Management Network)**

Protocolo de gerência para redes de telecomunicações. A proposta da TMN é fornecer um conjunto de interfaces padrões para facilitar o gerenciamento de funções de operação, administração e manutenção e de elementos de rede.

### **UDP (User Datagram Protocol)**

Protocolo de transporte que fornece serviço não orientado à conexão no conjunto de protocolos da Internet.