UNIVERSIDADE FEDERAL DE SANTA CATARINA
PROGRAMA DE PÓS-GRADUAÇÃO EM
CIÊNCIA DA COMPUTAÇÃO

LÍDIO MAURO LIMA DE CAMPOS

METÁFORAS BIOLÓGICAS COMBINADAS PARA PROJETO DE REDES NEURAIS ARTIFICIAS

Dissertação submetida à Universidade Federal de Santa Catarina como parte dos requisitos para a obtenção do grau de Mestre em Ciência da Computação

Prof. Mauro Roisenberg, Dr.

Florianópolis, outubro de 2001

METÁFORAS BIOLÓGICAS COMBINADAS PARA PROJETO DE REDES NEURAIS ARTIFICIAS

Lídio Mauro Lima de Campos

Esta Dissertação foi julgada adequada para a obtenção do título de Mestre em Ciência da Computação, Área de Concentração (Sistemas de Conhecimento) e aprovada em sua forma final pelo Programa de Pós-Graduação em Ciência da Computação.

Banca Examinadora

Prof. Mauro Roisenberg, Dr. – UFSC.

Prof.Fernando A .Ostuni Gauthier, Dr. – UFSC.

Prof. João Bosco da Mota Alves, Dr. – UFSC.

Prof.Roberto Célio Limão de Oliveira, Dr. – UFPA.

Profa.Silvia Modesto Nassar, Dra. – UFSC.

AGRADECIMENTOS

A Deus por me dar força, sabedoria e persistência.

Ao meu orientador, Prof. Dr. Mauro Roisenberg, pela sua dedicação, apoio e motivação. Seu bom humor, sua capacidade intelectual e sua disponibilidade em sempre atender, ouvir e aconselhar foram fundamentais para a realização deste trabalho. Para mim ele é além de um orientador, um amigo e conselheiro.

Gostaria de agradecer também ao Prof. Dr. João Bosco da Mota Alves por todo o apoio e por ter me incentivado a fazer o mestrado, pela maneira brilhante como conduziu a coordenação do mestrado e pelo estímulo e motivação transmitida a todos os alunos do curso.

Ao Prof. Msc. Gustavo Augusto Lima de Campos também pelo trabalho realizado na coordenação local do mestrado e pela sua paciência e vontade de atender os alunos, na sua casa ou no Cesupa, orientando e incentivando os mestrandos.

Agradeço a todos os membros que compuseram a banca de defesa da dissertação, pela paciência em ler o trabalho e pelas valiosas sugestões e comentários.

Especial agradecimento aos meus pais, pelo apoio, formação e incentivo constante e sincero. Motivos que não permitiram que este caminho fosse desviado.

Agradeço à Universidade Federal de Santa Catarina e a todos os professores do curso e ao Centro de Ensino Superior do Pará.

Aos colegas do mestrado em especial ao Luís, Andréa, Dio, Marta e Fabiano do grupo de Inteligência Artificial pelo interesse e amizade.

PUBLICAÇÕES

CAMPOS, Lídio Mauro Lima de; ROISENBERG, Mauro; CAMPOS, Gustavo Augusto Lima de. Evolutionary Approach to Design of Artificial Neural Networks. **In Frontiers in Artificial Intelligence and Applications**. Amsterdam: IOS Press, 2001. p. 35-42.

CAMPOS, Lídio Mauro Lima de; CAMPOS, Gustavo Augusto Lima de; OLIVEIRA, Luis Otávio Lacerda de; NASSAR, Silvia Modesto. Redes Bayesianas Aplicadas ao Planejamento de Operações Agrícolas. **Saber**. Ciências exatas e tecnologia: revista do Centro de Ensino Superior do Pará, Belém, v.2, n. 1-2, p.27-34, janeiro/dezembro. 2000.

CAMPOS, Lídio Mauro Lima de; CAMPOS, Gustavo Augusto Lima de; OLIVEIRA, Luis Otávio Lacerda de; . Redes Bayesianas Aplicadas ao Planejamento de Operações Agrícolas. In: I Congresso de Lógica Aplicada a Tecnologia / LAPTEC2000, 1., 2000. **Anais**... São Paulo-SP: EP, 2000. p. 125-134.

CAMPOS, Lídio Mauro Lima de; CAMPOS, Gustavo Augusto Lima de . Aplicação de Redes Neurais em um Problema de Seleção de Tecnologias para a Manufatura Integrada por Computador. In : I Simpósio Catarinense de Computação , 1., 2000. **Anais**... Santa Catarina: ρρρ Edições, 2000. p. 206-215.

GATO, R.F.G.; GRANA, A.J. de A; SANTOS,A.S.; SILVA,A W.F. da; RODRIGUES, D.M & De CAMPOS, L.M.L. (1997). "Estudo de demanda por Informação tecnológica pelo setor produtivo agro-industrial no estado do Pará". In: ENCONTRO NACIONAL DE PESQUISA EM CIÊNCIA DA INFORMAÇÃO, 3., Rio de Janeiro, RJ. Resumos. Rio de Janeiro: Associação Nacional de Pesquisa em Ciência da Informação, 1997, p.12-13.

SUMÁRIO

4 REDES NEURAIS

5 METODOLOGIA DA PROPOSTA

6 SIMULAÇÕES

7 CONSIDERAÇÕES FINAIS

ANEXOS

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS

AE Algoritmos Evolucionários

AG Algoritmos Genéticos

BAM BiDirecional Associative Memory

CBI Computador Baseado em Instruções

CC Ciência da Computação

CE Computação Evolucionária

CRN Computadores Baseados em Redes Neurais

DNA Ácido Desoxirribonucléico

EE Estratégias Evolucionárias

ERM Média dos erros dos padrões na saída da rede

IA Inteligência Artificial

NA Neurônio Artificial

NCIH Número de Neurônios na Camada Intermediária

NRP Número de Regras de Produção que gerem uma rede

PDP Parallel and Distributed Processing

PE Programação Evolucionária

REC Número de Recorrências

RNAs Redes Neurais Artificiais

RNA Ácido Ribonucléico

RNN Redes Neurais Naturais

RNR Redes Neurais Recorrentes

RP Roleta Ponderada

NRP Número de Regras de Produção

RS Roleta Simples

SL Sistemas de Lindenmayer

SNC Sistema Nervoso Central

SNP Sistema Nervoso Periférico

OG Operadores Genéticos

RESUMO

A Computação Evolucionária (CE) tem sido utilizada na área de Redes Neurais Artificiais (RNAs) para evolução de três grandes constituintes: pesos das conexões, arquiteturas e regras de aprendizado. A evolução de arquiteturas possibilita o projeto automático de arquiteturas de Redes Neurais Artificiais (RNAs), permitindo adaptá-las para diferentes tarefas sem a intervenção humana. O objetivo desta pesquisa é introduzir uma metodologia a mais plausível biologicamente possível, que permita automaticamente gerar Redes Neurais Artificiais (RNAs) com boa capacidade de generalização, pequeno erro e grande tolerância a ruídos. Para isso três metáforas biológicas foram usadas: Algoritmos Genéticos, Sistemas de Lindenmayer e (RNAs), Testou-se quatro classes de problemas:XOR, paridade, problema das lâmpadas e botões e as linguagens de Tomita. O método é superior em relação aos outros, pois aumenta o paralelismo implícito do algoritmo genético e pelos aspectos de plausibilidade biológica.O sistema gera arquiteturas mínimas satisfatórias que resolvem determinadas tarefas, reduzindo os custos de projeto e aumentando o desempenho das redes neurais obtidas. Finalmente sugerem-se estratégias racionais que podem fornecer uma eficiência adicional ao Algoritmo genético (AG) tradicional.

ABSTRACT

Evolutionary Computation has been introduced into Artificial Neural Networks (ANNs) for the evolution of three different levels: connection weights, archictectures, and learning rules. The evolution of architectures enables Artificial Neural Networks (ANNs) to adapt their topologies to different taks without human intervention, and thus provides an approach to automatic ANNs design. The objective of this research is to introduce an mthodology system biologically plausible, as far as possible that can automatically generate Artificial Neural Networks (ANN) with good generalization capacity, smaller error and larger tolerance to noises. To this aid, three biological metaphors were used: Genetic algorithms (GA), Lindenmayer Systems (L-System) and ANN. The method is better than other ones because it increases the level of implicit parallelism of genetic algorithm and the aspects of biological plausibility. The system generates the minimum satisfactory architecture that solves a specific task, reducing the project costs and increasing the performance of the neural networks obtained. Finally it is suggested rational strategies that can supply an additional efficiency to the traditional Genetic Algorithm.

1 INTRODUÇÃO

1.1 Motivação

A inspiração biológica tem sido ferramenta cada vez mais utilizada dentro da Computação e em especial, no que diz respeito a área da Inteligência Artificial. Na busca por sistemas inteligentes, a abordagem, que procura fazer a inteligência emergir, usando, como elemento de processamento, uma metáfora do neurônio biológico, tem crescido de maneira extraordinária nos últimos anos. Esta abordagem da Inteligência Artificial (IA) é conhecida como abordagem Conexionista ou mais especificamente como Redes Neurais Artificiais (RNAs).

As aplicações atuais que usam a tecnologia de RNAs variam desde reconhecimento de padrões, sistemas de controle, previsão financeira, medicina, sensoriamento remoto, agricultura e sistemas de apoio à decisão. Algumas perguntas surgem quando se usa a tecnologia de RNAs para resolver um determinado problema são: Que arquitetura de Rede Neural Artificial é mais apropriada para determinada classe de problema? Quantos neurônios serão necessários para representar a solução do problema com a precisão desejada?.

A definição da arquitetura de uma rede é um parâmetro importante na sua concepção, uma vez que ela restringe o tipo de problema que pode ser tratado. RNAs com uma camada única de neurônios, por exemplo, só conseguem resolver problemas linearmente separáveis. Redes Neurais Recorrentes (RNR), por sua vez são mais apropriadas para resolverem problemas que envolvem processamento temporal. Fazem parte da definição da arquitetura os seguintes parâmetros: número de camadas, número de nodos em cada camada, tipo de conexão entre os nodos e topologia.

A determinação de uma Rede Neural Artificial otimizada, bem como dos parâmetros de seu algoritmo de treinamento (taxa de aprendizado, momentum) melhora em muito o seu desempenho, isto é, velocidade e exatidão do aprendizado, tolerância a ruído e a capacidade de generalização. Entretanto esse problema não é simples, não existe um método para determinar uma arquitetura satisfatória para um dado problema. O espaço de busca entre as RNAs válidas é muito grande. Além disso, ele é deceptivo e multimodal : deceptivo, pois duas arquiteturas similares podem apresentar desempenhos

muito diferentes e multimodal, porque duas arquiteturas muito distintas podem ter desempenhos semelhantes MILLER, TODD e HEDGE (1989).

Existem técnicas que utilizam conhecimentos empíricos para projeto. Uma abordagem muito utilizada na prática consiste na construção de redes com arquiteturas padronizados, já utilizadas em outros sistemas. Assim, testa-se a mesma para a função em estudo e altera-se a estrutura e os parâmetros desta, até que se tenha uma arquitetura razoável para a aplicação desejada. Abordagem como esta, tem um custo muito elevado e não apresenta resultados confiáveis. Assim como não é possível garantir a otimização da solução, já que o critério de desempenho é baseado em uma combinação complexa de fatores.

O projeto automático de Redes Neurais Artificiais proporciona uma forma mais eficiente de busca de arquiteturas de RNAs do que os métodos convencionais. Sendo os Algoritmos Genéticos (AG) uma excelente ferramenta para esses tipos de problemas, pois os mesmos trabalham com uma população de indivíduos candidatos a solução do problema e não somente com um único, além disso, é possível direcionar a busca do AG, de acordo com a função custo adotada, possibilitando obter RNAs otimizadas satisfatórias que resolvam tarefas específicas.

A maioria dos trabalhos que utilizam AG para evolução de arquiteturas de RNAs utiliza como função custo o inverso do erro médio quadrático obtido durante o treinamento. Assim, a busca do AG é direcionada para as RNAs que apresentam o menor erro. Entretanto esse critério não garante que redes com arquiteturas mínimas sejam obtidas. Além disso, os trabalhos são mais voltados para obtenção de redes neurais diretas deixando de privilegiar as recorrentes, que representam atualmente um campo aberto a pesquisas, fato que também motiva o desenvolvimento desta pesquisa.

Alguns trabalhos que utilizam algoritmos evolucionários para obtenção e otimização de arquiteturas de RNA são: MILLER, TODD e HEDGE (1989), com um método de codificação estrutural, nesta mesma linha de pesquisa pode-se citar, VICO e SANDOVAL (1991). O método de KITANO (1990) utiliza gramáticas para geração de grafos que representam RNAs. BOERS e KUIPER (1992) e VAARIO (1993) buscaram inspiração na natureza e utilizaram Algoritmos Genéticos e os Sistemas de Lindenmayer

(SL), propostos por LINDENMAYER (1968), com o objetivo de gerar arquiteturas modulares de RNAs. Esses trabalhos diferem basicamente pelo método de codificação que utilizam, o tipo de arquitetura que geram e o grau de inspiração biológica. A preocupação com a questão da plausibilidade biológica na área de Inteligência Artificial não é compartilhada por vários pesquisadores. Outros são mais fiéis às inspirações biológicas BOERS e KUIPER (1992), VAARIO (1993) e ROISENBERG (1998). Ainda não se chegou a um consenso sobre qual seja a melhor metodologia de projeto, o que depende de diversos fatores tais como: flexibilidade, facilidade de implementação, robustez, tempo de processamento, desempenho das redes obtidas e do grau de inspiração biológica.

Vale ressaltar que nem sempre a inspiração na natureza é perfeita, normalmente os modelos adotam simplificações da realidade, sendo a robustez dos modelos o grande desafio de pesquisas nessa área. Acredita-se que um maior grau de fidedignidade nos modelos naturais quando trazidos para o ambiente computacional pode ser uma abordagem extremamente interessante quando se procura obter RNAs satisfatórias para resolver determinado problema, pois foi exatamente assim que a natureza operou para produzir os sistemas nervosos dos seres vivos que lhes permitem sobreviver e prosperar.

A inspiração biológica que foi adotada no desenvolvimento desta pesquisa considerou as seguintes premissas: O Algoritmo Genético como metáfora do processo evolucionário que procura criar/selecionar os seres vivos mais adaptados ao ambiente. As RNAs como metáfora do Sistema Nervoso deste seres que interagem com o ambiente fazendo o controle das atividades dos mesmos.Os Sistemas de Lindenmayer como metáfora da codificação cromossômica e do crescimento de estruturas dentre elas as RNAs.

1.2 Objetivo Geral

Esta dissertação tem como objetivo principal desenvolver uma ferramenta de projeto automático de RNAs, com inspiração na natureza, integrando três metáforas biológicas: *Algoritmos Genéticos (AG) e Sistemas de Lindenmayer (SL)*, que permita gerar de maneira automática arquiteturas satisfatórias de *Redes Neurais Artificiais*

(RNAs) diretas e recorrentes, que providenciem maior capacidade de generalização, menor erro e maior tolerância a ruídos.

1.3 Objetivos Específicos

- Adicionar à metodologia alguns aspectos de plausibilidade biológica observados na natureza em relação à Evolução Biológica, a transmissão das características hereditárias, ao processo de desenvolvimento biológico e a modularidade do cérebro, preocupação não compartilhada por muitos pesquisadores na área de Inteligência Artificial (IA).
- Fazer uma análise de plausibilidade biológica de três metodologias de projeto automático de arquiteturas de RNAs: Codificação Direta, Codificação Gramatical e Sistemas de Lindenmayer.
- Avaliar o efeito da função custo e a influência que elas exercem no processo de busca realizada pelo Algoritmo Genético.
- Avaliar a eficiência das técnicas de treinamento baseadas no método do gradiente descendente para as redes recorrentes de Jordan. Estudar a capacidade de generalização no sentido da extrapolação das redes recorrentes de Jordan.
- Obter uma gramática geral baseada em Sistemas de Lindenmayer, capaz de gerar grafos que representem redes neurais diretas e recorrentes.

1.4 Estrutura da Dissertação

O segundo capítulo apresenta fundamentos de Evolução Biológica e Seleção Natural, bem como os mecanismos que permitem a transmissão das características hereditárias de geração em geração. Adicionalmente os conceitos de Computação Evolucionária como metáfora da Evolução e finalmente uma análise de plausibilidade biológica dos Algoritmos Genéticos .

Modelos de Desenvolvimento Biológico e Artificial (Sistemas de Lindenmayer) são apresentados no terceiro capítulo, bem como a análise de plausibilidade biológica do mesmo.

No quarto capítulo são discutidos os modelos de Redes Neuronais e Redes Neurais Artificiais, bem como a modelagem matemática dos modelos utilizados nesta pesquisa.

O quinto capítulo é destinado a apresentar a metodologia da proposta utilizada na dissertação, descrevendo em detalhes a inspiração biológica apresentada anteriormente e uma análise de plausibilidade biológica de três metodologias para projeto automático de RNAs.

Os resultados obtidos por simulação, para diferentes classes de problemas, são apresentados no sexto capítulo.

Finalmente no sétimo capítulo, descreve-se as conclusões obtidas e recomendações para trabalhos futuros.

2 EVOLUÇÃO

2.1 Introdução

Ao discutir Evolução é importante considerar dois aspectos: primeiro a sua existência, e segundo os mecanismos da mesma. Ao observar-se a natureza, tenta-se entender como a mesma resolveu o problema, quais os mecanismos utilizados para a criação de criaturas cada vez mais complexas e melhores adaptadas a sobreviverem em determinado ambiente.

Estes mecanismos são a Evolução e a Seleção Natural. O primeiro é o mecanismo pelo qual se obtém a diversidade de organismos biológicos interagindo com o ambiente, sendo o mesmo uma arena finita, em conseqüência disso, existe uma competição pelos recursos disponíveis. As criaturas que apresentam os comportamentos menos eficientes ou que não são capazes de alterarem o mesmo em um ambiente dinâmico, têm uma tendência de serem eliminados na luta pela sobrevivência. Assim de geração em geração, os organismos aumentaram a sua chance de sobrevivência através do aumento de seu repertório de comportamentos e de sua capacidade de adaptação (ROISENBERG, 1998).

Em um sentido mais amplo a evolução é meramente mudança, a evolução biológica (ou evolução orgânica) é a mudança nas propriedades de organismos que transcendem o período de vida de um único indivíduo. As mudanças nas populações que são consideradas evolutivas são aquelas herdadas via material genético. Sendo assim, a natureza deve possuir também mecanismos pelo qual as criaturas mais aptas sejam capazes de transmitir a seus descendentes as características genéticas exibidas por eles por meio do código genético.

Este capítulo, inicialmente, apresenta fundamentos da evolução a seguir os fundamentos biológicos da célula ao DNA e resumidamente os mecanismos pelos quais as características genéticas são transmitidas aos descendentes por meio do código genético. Finalmente, são apresentados os principais métodos computacionais

inspirados nos mecanismos biológicos de evolução e uma análise de plausibilidade biológica dos AG.

2.2 Evolução Biológica

Jean Baptiste de Lamarck (1744-1829) apresentou no livro "Philosophie Zoologique" a primeira abordagem Evolucionista tratada cientificamente. Ele focalizou a transmissão dos caracteres adquiridos durante a existência de um indivíduo. Segundo as idéias dele, a evolução se dava através da hereditariedade dos caracteres adquiridos de forma direta, ou seja, o indivíduo herdava as características ao seu patrimônio genético e desta forma poderia transmiti-la a seus descendentes (DIAS, 1998).

Uma revolução no pensamento biológico e até mesmo na filosofia humana iniciou quando Charles Darwin e Alfred Russell apresentaram suas evidências para a teoria da Evolução perante a "Linnean Society of London" em primeiro de julho de 1858. Sendo que a teoria completa de Darwin só foi publicada em 24 de novembro de 1859 em seu livro "On the Origin of Species" (DARWIN, 1859).

A retórica e a tomada de posições a respeito das teorias de Darwin, vêm de longa data, desde o choque do século XIX entre o bispo Wilbeforce e T.H. Huxley, por causa da origem das espécies passando pelo julgamento do Scopes no Tennesse*, em 1925, até os criacionistas científicos da década de 1980, que tentaram legislar sobre o ensino das doutrinas bíblicas em contraposição ao darwinismo. Entre as teorias modernas o darwinismo é a única que tem incomodado muita gente fora do mundo acadêmico. Mesmo assim prossegue como movimento científico (ROSE, 2000).

Darwin voltou-se para um mecanismo de evolução baseado na Seleção Natural. Ele observou que, enquanto os descendentes herdavam semelhanças dos pais, os mesmos não são idênticos aos mesmos. Notou também que algumas diferenças entre pais e filhos não eram devidas somente ao meio-ambiente, mas também elas próprias

---

*Levado aos tribunais em julho de 1925, esse foi um dos processos judiciais mais divulgados da história norte-americana do século XX. Acusação foi que John Thomas Scopes, professor de ciências em um ginásio de Dayton, violara a lei estadual do Tennesse (promulgada em março de 1925) que proibia o ensino, nas escolas públicas, de qualquer teoria que negasse a criação divina do homem, tal como ensinada na Bíblia. A base da natureza sensacionalista do julgamento foi a crescente preocupação dos fundamentalistas cristãos com o desafio que a ciência e a teoria evolucionista representavam para a interpretação literal das Escrituras.

transmitidas por herança pelos pais. Darwin argumentou que na natureza, os indivíduos com qualidades que os tornava mais bem ajustados aos seus ambientes, ou lhes permitiam maior capacidade de reprodução tenderiam a deixar mais descendentes. Esses indivíduos eram considerados como tendo maior adaptabilidade. As partes mais discutidas da teoria de Darwin de difícil comprovação científica eram as inferências a respeito da hereditariedade das características, dado o conhecimento da época.

Em 1865 o monge Gregor Mendel (1822-1884) chegou à conclusão que cada característica de um indivíduo era determinada por um par de fatores hereditários. No momento de formar os gametas, os fatores se separavam, de modo que cada gameta era portador de apenas um fator relativo a cada característica.

As teorias de Darwin e Mendel não foram suficientes para explicar a evolução dos seres vivos, quando os mesmos interagem com o ambiente. Darwin já tinha observado que isso acontecia, chegando a admitir a hereditariedade dos caracteres diretamente adquiridos, de acordo com Lamarck, mas numa escala reduzida.

Em 1896 James M. Baldwin apresentou o trabalho "A new factor in evolution" (BALDWIN, 1896). Ele acreditava, da mesma forma que Lamarck, que o aprendizado adquirido pelos indivíduos interagindo com o meio poderia ser transmitido às gerações futuras. Entretanto Baldwin divergia de Lamarck no sentido de que as características adquiridas poderiam ser herdadas indiretas, e não diretamente transferidos para o patrimônio genético como achava Lamarck. Isto pela conscientização de que a capacidade em aprender e do que se aprende depende fortemente do meio. Assim, segundo Baldwin, as interações com o meio fazem com que os indivíduos que tenham maior habilidade em aprender a sobreviver neste meio, tenham o seu grau de adaptação melhorado. Desta forma, o indivíduo também terá uma probabilidade maior de sobrevivência e de reprodução. Consequentemente se houvesse reprodução viável, o patrimônio genético dos mais hábeis estaria sendo preservado para as gerações futuras.

Por outro lado, pelo fato das características adquiridas serem diretamente herdadas, a proposta de Lamarck requer o mapeamento inverso, do fenótipo para o genótipo, o que não é plausível biologicamente. Já a proposta de Badwin é Darwiniana,

pois não houve o mapeamento inverso. As características adquiridas são herdadas indiretamente.

Outra descoberta importante e que, aliada às Leis de Mendel, deu maior sustentação à teoria de Darwin, foi a mutação. O botânico holandês Hugo Marie De Vries (1848-1935) além de redescobrir as Leis de Mendel, foi o primeiro a utilizar o termo mutação. De Vries, em 1903, publicou o livro "The mutation theory", descrevendo seus estudos sobre o fenômeno da variação e mutação das plantas, onde esperava que amplas variações pudessem produzir uma espécie nova em uma única geração, discordando desta forma de Darwin, que enfatiza o desenvolvimento lento de novas espécies por diferenças individuais quase imperceptíveis.

Em janeiro de 1947 foi realizada em Princeton (EUA) uma conferência internacional, onde estavam presentes geneticistas, naturalistas e paleontólogos, com o objetivo de reunir todas estas teorias em torno de uma denominada de "neodarwinismo" ou também de "Teoria Sintética da Evolução". Tal teoria defende a idéia de que a história de todas as criaturas vivas pode ser explicada através de processos estatísticos atuando sobre populações e espécies. Tais processos são: reprodução, mutação, competição e seleção. A Competição e a Seleção tornam-se conseqüências inevitáveis para qualquer população que se expanda em um ambiente de recursos finitos. Assim sendo a Evolução pode ser entendida como o resultado destes processos estocáticos fundamentais à medida que eles atuam sobre populações, geração após geração (DIAS, 1998).

Supondo válida a teoria da evolução é interessante notar que existe uma tendência geral de que a finalidade da evolução é gerar organismos em graus crescentes de complexidade. Não há evidências de que o objetivo da mesma seja produzir a humanidade ou qualquer outra espécie. A Evolução resulta da diversidade de organismos biológicos interagindo com o meio ambiente, retirando deste suas necessidades de vida e modificando-o. Pelo fato de haver diversidades entre os organismos, alguns encontram condições de vida e uma possibilidade de maior de reprodução do que outros, sendo assim seu número tende a aumentar pelo processo de seleção, o que pode ser entendido como uma busca constante de maior eficiência. Os

mecanismos pelos quais a seleção atua possibilitando a evolução são: parcimônia, diversidade e morte (ROISENBERG, 1998).

- *Parcimônia* significa que o organismo tende a se organizar de modo a minimizar o esforço para viver.
- *Diversidade* significa que a natureza deve possuir um mecanismo de exploração de possibilidades na busca pela geração de organismos mais parcimoniosos. Este mecanismo é o da variação de fenótipos através de alterações nos genótipos dos organismos. Estas alterações se dão através de reproduções sexuadas e mutações.
- *Morte* para dar lugar a organismos mais evoluídos – sem morte não há evolução. Na natureza, morte não é o fim do ser, mas tal como uma regra de produção é a substituição de um elemento por seu descendente, mantendo vivo o elo da informação genética.

O processo da evolução é extremamente complexo e envolvendo uma série de fatores. A aptidão de um organismo de se adaptar ao ambiente não é sempre fixa, pois na sua luta pela sobrevivência existe uma interação do organismo com o próprio ambiente alterando-o. Com isso modifica-se também o grau de aptidão do mesmo. Considerando que o comportamento é o elemento de principal interação do organismo com o ambiente, sendo este determinado tanto por fatores genéticos como pelo grau de inteligência dele, o que lhe permite o aprendizado de novos comportamentos, pode-se dizer que: a capacidade de aprendizado também é um fator que influencia o processo de evolução. Este efeito é chamado de Baldwin (ROISENBERG, 1998).

2.3 Fundamentos Biológicos: Da Célula ao DNA

2.3.1 A Célula

A célula pode ser definida como a menor parte de um organismo vivo capaz de desenvolver, de forma autônoma, as funções básicas que caracterizam a vida: a reprodução e o crescimento. As mesmas podem assumir diferentes formas, tornando-se especializadas no desempenho de diferentes funções. As modificações que ocorrem na mesma, a fim de torná-la mais eficiente na realização de uma função, alteram sua forma e conteúdo, o que resulta na diferenciação celular. Aquelas que se diferenciam para

realizar a mesma função formam um tecido, os mesmos se organizam em órgãos para formar o indivíduo (FARAH, 1997).

Uma célula típica contém duas partes principais: o núcleo e o citoplasma. Entre os dois existe uma membrana nuclear, Figura 2.1. Na célula são produzidas as substâncias químicas necessárias para o desempenho de suas tarefas fundamentais, ou seja, crescer e se reproduzir em novas células (FARAH, 1997). O núcleo controla as reações químicas que ocorrem na célula, as divisões celulares e contêm grande quantidade de ácido desoxirribonucléico, constituídos de material cromático os genes. Estes controlam todas as funções da célula sendo também responsáveis pela transmissão das características hereditárias de célula para célula (BARRETO, 1999), o que será discutido em detalhes no próximo capítulo.

O citoplasma é composto de várias organelas celulares com funções específicas. Como exemplos pode-se citar as *mitocôndrias*, responsáveis pela produção de energia, os *lisossomos*, onde os alimentos são ingeridos, o *complexo de Golgi*, que concentra e secreta as proteínas para o exterior da célula e, de maior interesse, os *ribossomos*, onde as proteínas são fabricadas e que geralmente, aparecem ligadas a um sistema de membranas do citoplasma o *retículo endoplasmático*.

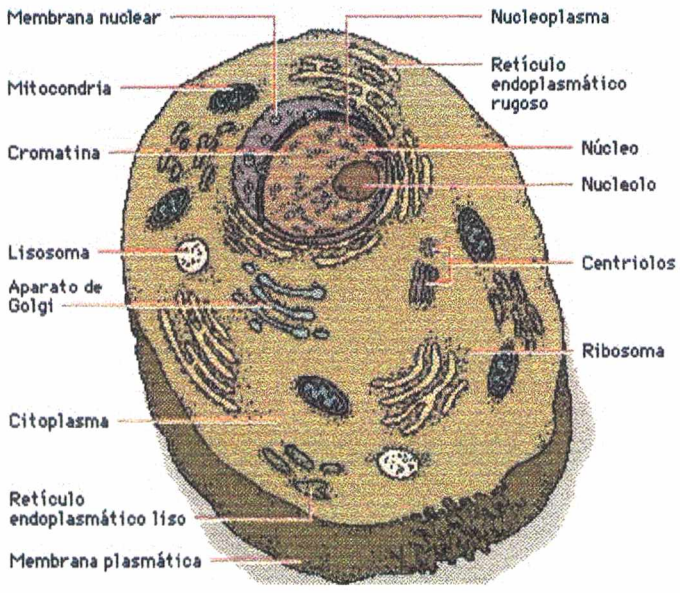


Figura 2.1 Célula Animal, adaptado de (FARAH, 1997).

2.3.2 As Proteínas

A função das proteínas nas células varia. Algumas são *estruturais* e compõem as membranas celulares ou formam as fibras musculares. Outras atuam defendendo o organismo contra elementos indesejáveis como as bactérias e os vírus. Um bom exemplo são os *anticorpos* que dão proteção contra doenças infecciosas. Entretanto as que interessam a esta discussão são as enzimas, praticamente todas as reações químicas que ocorrem na célula exigem a participação de uma *enzima,* que aceleram a taxa de uma reação química sem propriamente participar dela. Praticamente, cada uma é capaz de catalisar somente uma reação química. A especificidade deve-se ao fato de que, para a mesma atuar, ela deve se encaixar nas moléculas que participam da reação química chamada substratos (FARAH, 1997).

2.3.3 Os Ácidos Nucléicos

Há dois tipos de ácidos nucléicos o DNA e o RNA que controlam de forma integrada a síntese de proteínas. O DNA, como qualquer outro polímero, é formado por nucleotídeos. Sendo cada um composto por um grupo fosfato, um açúcar e uma base nitrogenada que pode ser: Adenina (A), Guanina (G), Citosina (C) e Timina (T). O açúcar e o fosfato são componentes invariáveis nos nucleotídeos e apresentam uma função unicamente estrutural na molécula de DNA. A parte que guarda a informação importante sobre os programas para a síntese de proteínas são as bases nitrogenadas (FARAH, 1997).

Milhares de nucleotídeos empilham-se em forma linear para constituir uma cadeia de molécula de DNA. O grupo fosfato liga o carbono 3′ do açúcar de um nucleotídeo ao carbono 5′ do açúcar do nucleotídeo seguinte, por meio de uma ligação fosfodiéser. Sendo este constituído de duas cadeias de nucleotídeos que se enrolam originando uma estrutura de *dupla-hélice,* como mostrado na Figura 2.2 . É interessante notar que essas duas cadeias dispõem-se de modo antiparalelo, ou seja, enquanto uma vai da direção 5′par 3′ a outra ocorre na direção oposta. Ambas as cadeias são unidas por *pontes de hidrogênio* (FARAH, 1997).

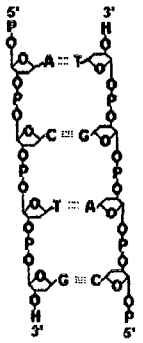


Figura 2.2 – Estrutura do DNA adaptado de (FARAH, 1997).

A estrutura do RNA é semelhante à do DNA, a primeira diferença, entretanto, é que o açúcar presente no mesmo é a ribose, daí seu nome ácido ribonucléico. Outra diferença é que a base timina (T) do DNA é substituída no RNA pela base Uracila (U). Além disso, o RNA é formado por uma única cadeia de nucleotídeos. Existem três tipos de RNA: o RNA mensageiro que carrega uma mensagem específica do DNA, até o citoplasma. O RNA ribossômico que se liga às proteínas para compor os ribossomos e o RNA transportador que carrega os aminoácidos no citoplasma que se juntarão para formar uma cadeia polipeptídica (GUYTON e HALL, 1988).

2.3.4 O Código Genético

Em 1966 o código genético foi completamente decifrado. Ficou então claramente demonstrado que: (1) todos os aminoácidos nas proteínas são codificados por uma combinação de três bases do DNA, formando um tríplex chamado códon; (2) um mesmo aminoácido pode ser definido por mais de um códon, sendo por isso o código degenerado, (3) Existem três códons que, ao invés de determinarem a entrada de um aminoácido específico na cadeia polipeptídica, funcionam como o fim de um programa, determinando a interrupção da cadeia que está sendo utilizada. O código genético Tabela 2.1, costuma ser designado pelas bases correspondentes ao RNA mensageiro (FARAH, 1997).

Vale a pena lembrar aqui que o RNA mensageiro não é fabricado como uma cópia exata do DNA, mas sim como uma molécula complementar que respeita as mesmas regras de ligação entre as bases nitrogenadas, sendo somente a base Timina (T) do DNA substituída pela base uracila (U) do RNA. A seguir aparece exemplificado

como uma mensagem presente em uma das cadeias do DNA será transcrita para o RNA e, observando-se a Tabela 2.1, pode-se traduzir o código determinando-se quais aminoácidos seriam adicionados na molécula de proteína, como mostra a Figura 2.3. Na próxima seção apresenta-se a síntese de proteína detalhadamente.

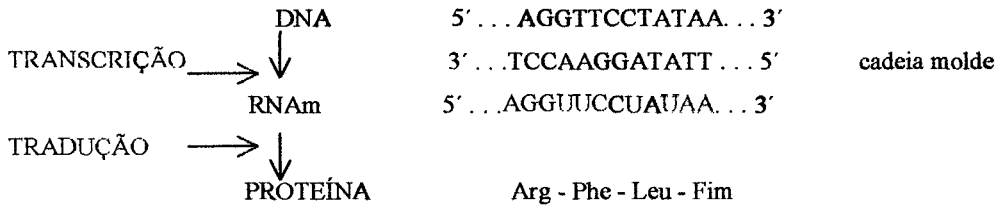


Figura 2.3 – Processo de Transcrição e Tradução

2.3.5 Genes e Cromossomos

Um gene é um segmento de DNA que contem o arquivo completo da seqüência de aminoácidos para fabricar uma cadeia polipeptídica específica. No caso de uma proteína ser composta de mais de uma cadeia polipeptídica, existe um arquivo para cada uma das cadeias e, portanto, tal proteína seria codificada por mais de um gene. O DNA de uma célula humana apresenta o comprimento total de quase dois metros. Desde que existam aproximadamente 10^{13} células no corpo humano, isso significa que, alinhando-se ponto a ponto o DNA de todas às células de um ser humano, o total de DNA seria de aproximadamente $2x10^{13}$ m. Provavelmente para facilitar a organização desses quase 2 metros de DNA dentro do núcleo de cada célula, o DNA é dividido em vários elementos distintos chamados cromossomos. Cada cromossomo é formado por uma molécula de dupla hélice de DNA condensada com proteínas (WOLPERT, 2000).

2.3.6 A Síntese de Proteínas

O processo de síntese de proteínas inicia quando um sinal que vem de dentro ou de fora da célula, avisa que é hora de sintetizar uma proteína particular. Pode-se fazer uma analogia do processo da seguinte forma: Um dos arquivos do DNA (como sendo um computador) é então copiado no RNA (disquete). A mensagem está contida na seqüência de bases de uma das duas cadeias que formam a dupla hélice do DNA. Para que esta mensagem seja transcrita para o RNA é necessário, antes de tudo, que a dupla hélice se abra separando as duas cadeias. O RNA é então formado copiando a mensagem do DNA de forma complementar.

Por razões não completamente esclarecidas, nos organismos eucariotos, incluindo o homem, um arquivo do DNA com a mensagem para a síntese de uma cadeia polipeptídica é, geralmente, interrompido por certas posições que não têm função codificante denominada íntrons, enquanto que as regiões com a informação codificantes são os exons. Quando um arquivo é ativado, o RNA copia fielmente tanto os íntrons como os exons, entretanto, antes do RNA nuclear deixar o núcleo, as porções correspondentes aos íntrons são retiradas. Dessa forma, no RNA mensageiro, aquele que efetivamente carrega a mensagem para o citoplasma estão presentes somente as porções que correspondem aos exons (RIDLEY,2001).

No citoplasma, o RNA mensageiro liga-se ao ribossomo e, à medida que o arquivo começa a ser traduzido, a cadeia polipeptídica vai sendo montada. Pequenas moléculas de RNA presentes no citoplasma transportam os aminoácidos corretos de acordo com a seqüência de códons. Os aminoácidos juntam-se por meio de ligações peptídicas para compor a cadeia polipeptídica, como mostra a Figura 2.4. Quando a cadeia está completa, ela se desprende do ribossomo (FARAH, 1997).

2.3.7 Genótipo e Fenótipo

O código genético não descreve qual será a forma final de um organismo, mas sim codifica um certo número de regras que, quando seguidas, resultarão na forma final. Ou seja, ao invés de um diagrama ou planta de como será o organismo, os genes dos cromossomos equivalem a uma receita e esta é chamada de genótipo. O processo de crescimento e manutenção de um organismo resulta na "execução" dessa receita pelas células e na reprodução destas. Assim, o material genético vai produzir um fenótipo, resultado do genótipo interagindo com o ambiente. Isto significa que não existe uma correspondência de um para um entre o que está codificado no gene e o que será o organismo final (ROISENBERG, 1998).

Assim, num processo evolucionário, não são os organismos finais, mas as receitas que os construíram é que são combinadas para a evolução. Apenas as regras (proteínas) que tem a capacidade de funcionarem conjuntamente para a produção de um novo organismo irão sobreviver. Este processo evolucionário não leva apenas a mistura de receitas, mas faz com que certas receitas sejam executadas mais vezes, ou seja,

aquelas que deram certo podem ser levadas a serem repetidas mais vezes durante o processo de crescimento de novos organismos (ROISENBERG, 1998).

Provavelmente foi isto que ocorreu no caso do cérebro humano. O mesmo é resultado de um processo evolucionário durante o qual o cérebro dos nossos ancestrais foi se tornando maior. Se o cérebro humano for comparado com o cérebro de um gato ou macaco, a maior diferença está no tamanho, que no caso dos humanos é significativamente maior, em particular os hemisférios cerebrais, que são responsáveis pelas funções cognitivas. A idéia então, é que, o processo evolucionário fez com que a receita que determina como um cérebro deve ser feito, seja executada mais vezes no processo embrionário de um homem, do que em um gato (ROISENBERG, 1998).

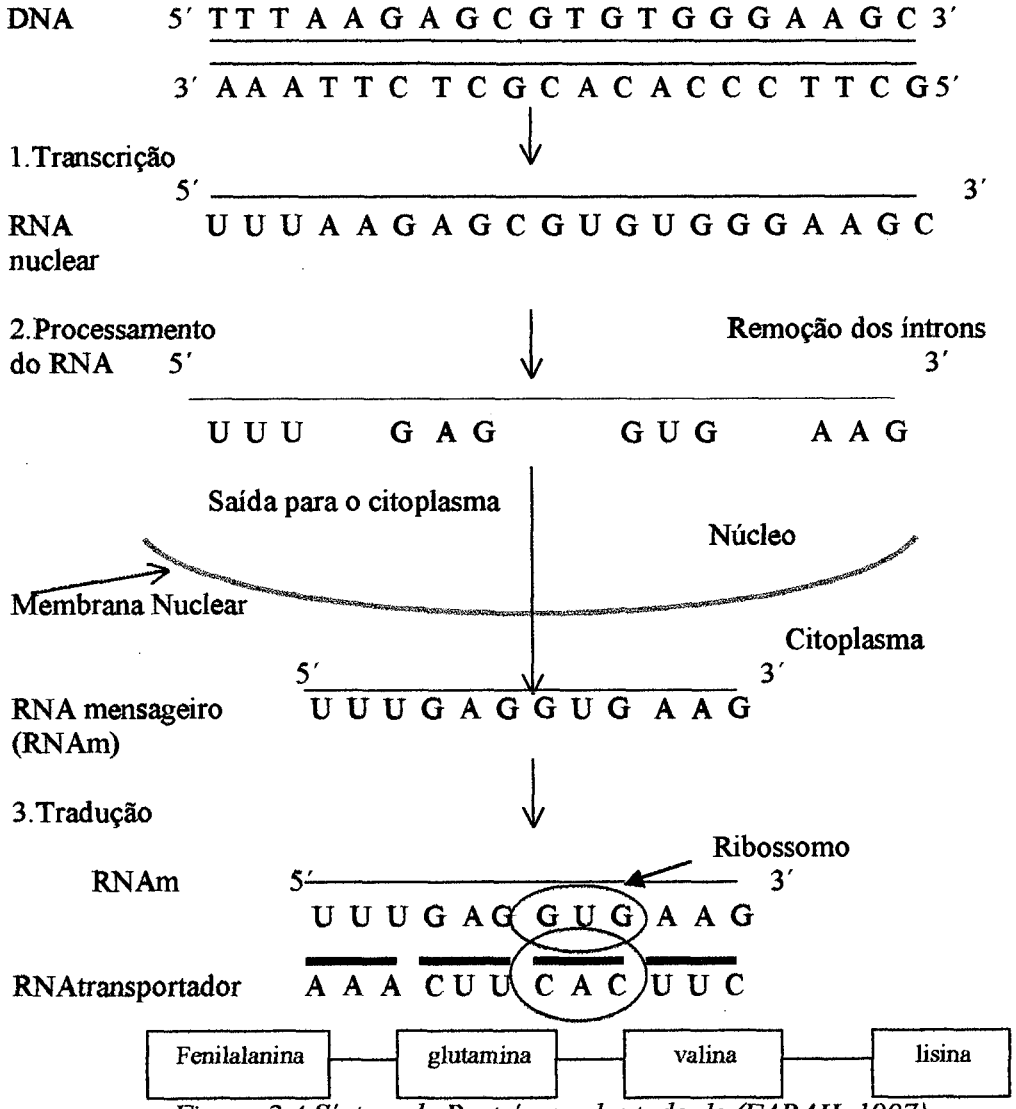


Figura 2.4 Síntese de Proteínas adaptado de (FARAH, 1997)

Tabela 2.1 – O Código genético universal: constituído por 64 trincas diferentes (codóns e seus aminoácidos correspondentes).

		SEGUNDA LETRA					
		A	G	T	C		
PRIMEIRA LETRA	A	Fenilalanina	Serina	Tirosina	Cisteína	A	TERCEIRA LETRA
		Fenilalanina	Serina	Tirosina	Cisteína	G	
		Leucina	Serina	Fim	Fim	T	
		Leucina	Serina	Fim	Tritofano	C	
	G	Leucina	Prolina	Histidina	Arginina	A	
		Leucina	Prolina	Histidina	Arginina	G	
		Leucina	Prolina	Glutamina	Arginina	T	
		Leucina	Prolina	Glutamina	Arginina	C	
	T	Isoleucina	Treonina	Aspargina	Serina	A	
		Isoleucina	Treonina	Aspargina	Serina	G	
		Isoleucina	Treonina	Lisina	Arginina	T	
		Metionina	Treonina	Lisina	Arginina	C	
	C	Valina	Alanina	Aspartato	Glicina	A	
		Valina	Alanina	Aspartato	Glicina	G	
		Valina	Alanina	Glutamato	Glicina	T	
		Valina	Alanina	Glutamato	Glicina	C	

Fonte: FARAH, Solange Bento. (1997). DNA Segredos e Mistérios. SAVIER, São Paulo.

2.4 Computação Evolucionária

Computação Evolucionária (CE) é o nome genérico dado aos métodos computacionais que se inspiram na teoria da Evolução. A Biologia Evolucionária é uma forte fonte de inspiração para resolução desses de problemas, pois a própria natureza teve de realizar uma busca num enorme conjunto de soluções possíveis, que são seqüências genéticas, para desenvolver organismos aptos a viverem e a se reproduzirem em seus ambientes (MITCHELL, 1996).

Os Algoritmos usados em CE são chamados Evolutivos ou Evolucionários (AE). Estes incluem os Algoritmos Genéticos (AG), a Programação Evolucionária (PE) e as Estratégias Evolucionárias (EE). A seguir apresenta-se de maneira introdutória os conceitos de AG, pois os mesmos foram utilizados no desenvolvimento desta dissertação.

O AG foi proposto inicialmente por John H. Holland em 1975 no trabalho intitulado "Adaptation in Natural and Artificial Systems" (HOLLAND, 1975). Holland

inspirou-se no mecanismo de evolução das espécies, tendo como base os trabalhos de Darwin sobre a origem das espécies e na genética natural devido principalmente a Mendel. Atualmente, há também uma variação desta, a programação genética (PG), onde além da presença dos operadores de seleção e mutação há também a do cruzamento.

Conforme a definição de KOZA (1990), um AG é um algoritmo matemático altamente paralelo que transforma populações de objetos matemáticos individuais em novas populações utilizando operações genéticas, como a reprodução sexual (recombinação), e a reprodução proporcional a adaptabilidade, que é o princípio Darwiniano da sobrevivência do mais apto.

É conveniente mencionar que existem outras teorias para a origem da vida e para os mecanismos da evolução biológica. Apesar dos conceitos de Darwin serem os mais conhecidos, tal hipótese não deve ser considerada como a única válida (BARRETO, 1999).

As principais diferenças entre os (AG) e os algoritmos tradicionais de busca, segundo GOLDBERG (1989), são:

- AG trabalham com uma codificação de parâmetros em vez de trabalhar com os parâmetros em si;
- AG trabalham com uma população de pontos, em vez de um único ponto;
- AG utilizam a informação de uma função custo , e não utilizam conhecimentos auxiliares ou derivados;
- AG utilizam regras de transição probabilísticas em vez de regras determinísticas.

2.5 Conceitos Fundamentais de Algoritmos Genéticos

A nível biológico, um ser vivo (indivíduo) é formado por um conjunto de cromossomos. Em AG emprega-se indistintamente os dos dois termos (indivíduos e cromossomos). Entretanto fazendo uma analogia da biologia com os problemas a serem otimizados pelo AG, isto só é válido se o problema for uma única variável. Caso o problema seja uma função de várias variáveis, todas passíveis de otimização, então o indivíduo é composto de vários cromossomos. Este geralmente é o caso das (RNAs).

Em AG é comumente utiliza-se o termo "string" como sendo sinônimo de cromossomo e indivíduo. Isso se deve ao fato de que na representação algorítmica do cromossomo, em AG, geralmente ser implementada pelo alfabeto binário {0, 1}, fazendo analogia com a natureza que utiliza quatro bases: a Adenina representada por **A**, a Guanina representada por **G**, a Timina representada por **T** e por último a Citosina representada por **C** (FOGEL, 1995).

O cromossomo é composto de genes, sendo que cada um possui um local fixo no mesmo. Este local é denominado de lócus. Cada gene pode assumir certo valor dentro de um domínio de valores, os quais são denominados de alelo. Em termos de AG, o cromossomo corresponde ao indivíduo, e este é representado por uma "string" de comprimento finito. O termo gene é denominado de "bit". O termo alelo refere-se ao conjunto de valores possíveis de serem atribuídos a um determinado "bit", ou seja, o alfabeto binário {0, 1}. Dessa forma o valor de um "bit" (alelo) refletido no fenótipo depende da posição que este ocupa na "string" (lócus). Ao conjunto de cromossomos, genes e alelos denomina-se genótipo e as características conferidas por este denomina-se fenótipo. Em termos de AG, o genótipo é a variável independente *(x)* e o fenótipo é a variável dependente ou função, *f(x)* (DIAS, 1998).

No algoritmo genético trabalha-se com um conjunto de indivíduos (população) no qual cada elemento é candidato a ser a solução desejada. A função a ser otimizada representa o ambiente no qual a população inicial vai ser posta. Espera-se que, através dos mecanismos de evolução das espécies e da genética natural, os indivíduos mais aptos tenham maior probabilidade de se reproduzirem e que a cada nova geração estejam mais aptos ao ambiente (função a ser otimizada).

A aptidão, em inglês "fitness", é obtida pela avaliação do indivíduo através da função a ser otimizada. Se o objetivo for maximizar, a aptidão é diretamente proporcional ao valor da função. Caso o objetivo seja a minimização, a aptidão será inversamente proporcional ao valor da função.

Contudo, o termo minimização não é bem aceito por alguns pesquisadores por não ter inspiração biológica, haja vista que os indivíduos mais aptos é que deverão ter maiores chances de sobreviverem (TANOMARU, 1995).

Após ter sido realizado o teste de todos os indivíduos da população na função a ser otimizada, obtém-se a aptidão para cada um. A adaptação de um indivíduo, no contexto biológico, designa o desempenho deste no ambiente em que se encontra inserido (HOLLAND , 1975). A aptidão é a quantificação da adaptação do indivíduo, ou seja, é o valor obtido com a aplicação do indivíduo à função custo.

A aptidão é um valor que exprime quão adaptado está o indivíduo ao meio; quanto maior a aptidão, maior é a probabilidade de o indivíduo se reproduzir. Assim, a aptidão é obtida através da função $f(x)$, a qual é uma aproximação do meio onde se encontra inserido o indivíduo x que deverá competir com os demais indivíduos da população. Dependendo do meio e do genótipo do indivíduo x, ao longo do tempo pode haver uma variação na sua adaptação, Δx. A magnitude desta variação é denominada de grau de adaptação. Isso quer dizer que ela possui um alto grau de adaptação, quando considerados estes dois meios (DIAS, 1998).

Conforme MITCHELL (1996) um AG simples é composto pelos seguintes passos:

1) Iniciar uma população de cromossomos, geralmente gerada aleatoriamente;
2) Calcular a adaptabilidade de cada cromossomo;
3) Repetir os seguintes passos até o n-ésimo descendente ser criado;
   a) Selecionar dois cromossomos como pais, através do operador de seleção;
   b) Aplicar a operação de recombinação sobre os dois cromossomos selecionados no passo anterior, gerando os descendentes;
   c) Aplicar a operação de mutação, sobre os dois descendentes gerados no passo anterior;
4) Trocar a população antiga pela nova;
5) Voltar ao passo 2.

2.6 Operação de Seleção

Finalizada a avaliação dos indivíduos da geração atual, espera-se que a próxima geração seja uma evolução da anterior. Para que isso ocorra, os mais aptos deverão possuir maior probabilidade de serem selecionados para dar origem à nova geração. Contudo, alguns poucos não muito aptos também poderão ser selecionados. Ao

mecanismo responsável por está escolha seletiva, denomina-se de seleção. Desta forma, se o processo for bem conduzido, espera-se que a nova geração seja, em média, melhor do que a que lhe deu origem.

A seleção tem basicamente por objetivo fazer com que os indivíduos mais aptos da geração anterior tenham maior probabilidade de participarem do processo que irá gerar a nova população.O processo de seleção tem início após a verificação de aptidão de cada individuo e análise de não convergência dos valores.

O processo de validação fornece os elementos da população em ordem de adaptação. Uma das formas mais comumente empregadas para o processo de seleção é a roleta (denominada doravante de roleta simples). No entanto, conforme demonstrado a seguir, está técnica promove uma rigorosa pressão seletiva, podendo reduzir bruscamente a diversidade das populações iniciais. Assim é grande a chance de a convergência ser guiada para um máximo local. Como forma de amenizar este problema utiliza-se uma variante da roleta simples (RS), também conhecida como roleta ponderada (RP). No caso da RS, cada individuo da população anterior terá uma probabilidade de ser sorteado proporcional a sua aptidão, conforme mostra o exemplo da Tabela 2.2. A soma da aptidão dos 4 indivíduos é 220. Portanto, o primeiro indivíduo apresenta 91% de adaptação relativa a somatória da adaptação da população. Na Figura 2.5, mostra-se o domínio do primeiro indivíduo sobre os demais (DIAS, 1998).

A roleta ponderada também apresenta os indivíduos ordenados conforme desempenho. Contudo, a distância entre os indivíduos próximos é reduzida, ou seja, a pressão seletiva é atenuada. Na RP a cada individuo é atribuído um posto, conforme sua ordem na população.

A Figura 2.6 mostra a RP construída conforme os dados da Tabela 2.3. Uma análise comparativa das duas técnicas de seleção, RS e RP, conclui-se que a pressão seletiva causada pela RP é menor do que a da RS. Isso porque despreza o valor da distância entre as aptidões de dois vizinhos, considerando apenas que tem aptidões diferentes.

Tabela 2.2 – Método da roleta simples

Indivíduos Ordenados	Aptidão	
	Valor	% do total
X4	201	91
X3	10	5
X2	7	3
X1	2	1
TOTAL	220	100%

Fonte: BARRETO, Jorge Muniz. (1999). Inteligência Artificial no limiar do século XXI. Florianópolis.

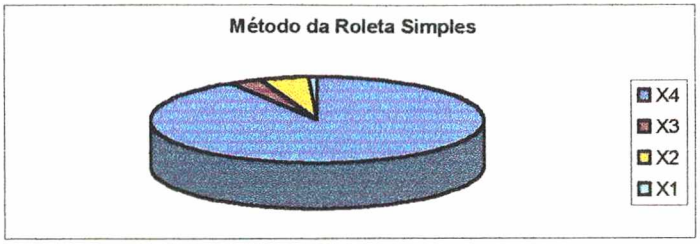


Figura 2.5 – Participação de cada indivíduo pelo método da roleta simples adaptado de (DIAS, 1998).

Os exemplos apresentados nas Tabelas 2.2 e 2.3 são bastante representativos. Principalmente quando nas primeiras gerações aparece um individuo com um alto valor de aptidão em relação aos demais, por exemplo, um máximo local. Neste caso, pode-se finalizar a comparação entre as técnicas evidenciadas que no caso da RS, o melhor elemento terá uma relação de 91:1 de ser sorteado em relação ao pior, enquanto que no caso da RP a relação é de 4:1.

Realizada a seleção, o próximo passo é a aplicação dos mecanismos de busca, também conhecidos como Operadores Genéticos. Entre tais mecanismos empregados em AG, são: cruzamento e mutação. Esses mecanismos serão descritos a seguir, bem como os valores que influenciam o desempenho do AG.

2.7 Operadores Genéticos

a) Operador de Recombinação

Também conhecido como "crossover" ou cruzamento, a operação de recombinação consiste na escolha aleatória de um lócus, e na troca das partes anteriores ao lócus entre dois cromossomos pais, para gerarem descendentes (MITCHELL, 1996).

A operação de recombinação mais usual utiliza apenas um ponto de corte, ou seja, os cromossomos pais são divididos em duas partes, e estas partes são trocadas entre eles. A Figura 2.7 mostra a operação de recombinação com um ponto de corte. Se

a taxa de cruzamento for muito alta, alguns cromossomos de bom desempenho podem ser removidos mais rapidamente do que a seleção possa desenvolvê-los, por outro lado se for muito baixa a busca pode estagnar (LEHRER, 2000).

Tabela 2.3 – Método da roleta ponderada

Indivíduos Ordenados	**Aptidão**		
	Valor	**Posto**	**Soma dos Postos**
X4	201	4	10
X3	10	3	6
X2	7	2	3
X1	2	1	1

Fonte: BARRETO, Jorge Muniz. (1999). Inteligência Artificial no limiar do século XXI. Florianópolis.

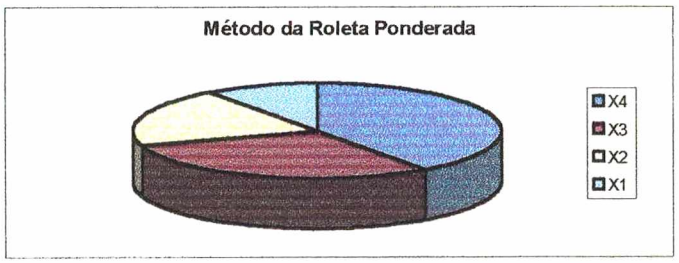


Figura 2.6– Participação de cada indivíduo pelo método da roleta ponderada, adaptado de (DIAS, 1998)

b) Operador de Mutação

A operação de mutação é um operador base, consistindo na realização de pequenas mudanças aleatórias no cromossomo. Isso assegura que todas as partes do espaço de busca sejam teoricamente pesquisadas. A operação de mutação altera aleatoriamente os valores de alguns genes do cromossomo, podendo ocorrer em cada gene uma probabilidade muito pequena, usualmente 0.001 (MITCHELL, 1996). A Figura 2.8 exemplifica o funcionamento da operação de mutação. Se a taxa de operadores de mutação for baixa, evita-se que uma dada posição estabilize-se em um único valor e uma taxa de mutação muito alta resulta em uma busca aleatória (LEHRER, 2000).

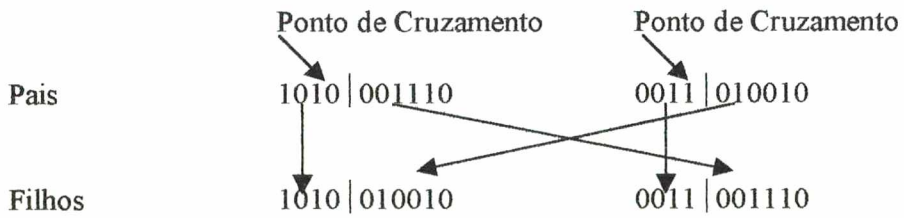


Figura 2.7 – Operação de recombinação com um ponto de corte adaptado de (DIAS, 1998)

Cromossomo Original	1010**0**10010
Cromossomo Alterado	1010**1**10010

Figura 2.8 – Operação de mutação adaptado de (DIAS, 1998)

2.8 Considerações Adicionais Sobre Algoritmos Genéticos

Nesta seção apresenta-se a importância da escolha da função aptidão e os problemas que podem ocorrer no desempenho do AG. Adicionalmente apresenta-se um estudo comparativo entre o paradigma biológico e o artificial.

2.8.1 Função Custo

Junto com o esquema de codificação usado, a função custo é o aspecto mais crucial de qualquer AG. A mesma pode ser uma função matemática um experimento ou mesmo um jogo. Sendo a saída dela modificada para um conjunto de valores apropriados de entrada. No início de execução de qualquer AG, define-se um conjunto de cromossomos ou um vetor de valores para serem otimizados. Por exemplo, se o cromossomo tem N parâmetros a função custo é função dos mesmos dado por: *custo=f(p1,p2,p3,....,pN).*

Freqüentemente a escolha da função custo é complicada, pois se deve analisar quais parâmetros são essenciais para a solução do problema e aqueles que não têm impacto e não devem ser incluídos, o que não é trivial para alguns problemas. Idealmente se deseja uma função homogênea e regular. Para muitos problemas de interesse, infelizmente, não é possível construir uma função ideal. Não obstante, para que o AG (ou qualquer técnica de otimização) seja eficiente, é necessário achar modos de construir uma que não tem muitos máximos locais, ou um máximo global muito isolado.

Dependendo da Função custo escolhida alguns problemas podem acontecer durante a execução do AG. No inicio do processo os valores para cada gene para membros diferentes da uma população são distribuídos uniformemente. Por conseguinte, há uma expansão larga de aptidão individual. A partir de determinado

ponto da execução do mesmo, valores particulares para cada gene começam a predominar, com isso o alcance da aptidão na população reduz. Esta variação ao longo da execução conduz freqüentemente aos problemas de convergência prematura, o que não é desejável.

Uma vez que a população converge, a habilidade do AG em continuar procurando soluções melhores é eliminada efetivamente, os cruzamentos entre cromossomos quase idênticos produzem poucas variações, a mutação pode levar a uma busca aleatória. Uma técnica utiliza para manter o AG efetivo é modificar o modo de selecionar os indivíduos para reprodução, controlando as oportunidades de cruzamento que os indivíduos adquirem.

2.8.2 Do Paradigma Biológico ao Artificial

Apesar de os AG serem inspirados na natureza esta inspiração não é perfeita. As questões relativas à plausibilidade biológica dos mesmos devem-se em parte as terminologias utilizadas nos dois paradigmas, como apresentado na seção 2.5. Além disso, para analisar a plausibilidade biológica, isso não é suficiente, é preciso entender os mecanismos naturais, que permitem aos indivíduos transmitirem aos seus descendentes as características genéticas por ele exibidas.

Nos AG normalmente o algoritmo já inicia a população igualmente distribuída no espaço de soluções produzindo genótipos diferentes na população inicial. Entretanto na natureza não é bem assim que acontece. Os processos evolucionários naturais sempre tentam primeiro achar soluções mais simples e depois evoluem para soluções mais complexas e caras. Um exemplo é a evolução do sistema nervoso, primeiro surgiram seres primitivos com sistemas neurais simples e depois evoluíram até aparecer o homem (BARRETO, 1999).

Nos AG os genes dos cromossomos codificam parâmetros de uma função aptidão a ser otimizada e não receitas, como mencionado na seção 2.3.7, o que não é plausível biologicamente. A hereditariedade é descrita hoje em termos de informação, mensagens, código. O que se transmite de geração em geração, são as "instruções" que especificam as estruturas moleculares. São os planos arquitetônicos do futuro organismo, são também os meios para executar esses planos e coordenar as atividades

do sistema. Portanto, cada ovo contém, nos cromossomos recebidos dos pais, todo o seu futuro, as etapas do seu desenvolvimento, a forma e as propriedades do ser que surgirá dele. O organismo torna-se assim a realização de um programa prescrito pela hereditariedade (JACOB, 1983).

Normalmente nos AG, é raro uma mutação cujo efeito seja nulo. Na natureza isso é possível, se as espécies são estáveis é porque o programa é rigorosamente recopiado, signo por signo, de geração em geração. Se elas variam, é porque de tempos em tempos o programa se modifica (JACOB, 1983).

No Paradigma Biológico quando se fala de programa genético, dois conceitos devem ser levados em consideração: projeto e memória, por projeto entende-se o plano que dirige minuciosamente a formação de um organismo e por memória entende-se a lembrança dos pais que a hereditariedade grava no descendente. Entretanto esses dois temas estão no centro de muitas controvérsias, como a que diz respeito aos caracteres adquiridos. Achar que o meio ensina a hereditariedade significa uma confusão, entre dois tipos de memória a genética e a nervosa (JACOB, 1983).

Para a biologia moderna, uma característica fundamental dos seres vivos é a sua capacidade de conservar a experiência passada e de transmiti-la. Neste contexto comportamentos inatos já vem pré-fiados na memória genética. Por outro lado àqueles que são adquiridos ou aprendidos ocupam o que se chama memória nervosa. Entretanto estes comportamentos aprendidos só podem ser passados aos descendentes, se houver comunicação entre os mesmos, caso contrário, eles têm de ser apreendidos novamente pela próxima geração (JACOB, 1993).

Algumas variações do AG permitem que o aprendizado Baldwiniano seja agregado ao AG. O mesmo troca apenas o grau de adaptação dos indivíduos, com isso a pressão seletiva é direcionada para os indivíduos que apresentarem maior desempenho no momento da busca. Ao contrário da pressão seletiva de origem puramente evolutiva, a pressão seletiva devido ao aprendizado leva em consideração o desempenho evolutivo do indivíduo, além do próprio aprendizado (DIAS, 1998). O aprendizado Lamarckiano, quando incorporado ao AG, não é plausível biologicamente, pois o mesmo considera

que o aprendizado adquirido deve ser incorporado ao genótipo, com isso existe um mapeamento inverso entre genótipo e fenótipo.

Não há como discutir que os aspectos aqui discutidos são de extrema importância em termos de plausibilidade biológica. O que se pretende nos próximos capítulos é explorar algumas destas idéias e trazer o AG mais para perto do que ocorre na natureza. No capítulo 3 apresentam-se modelos de desenvolvimento biológico e o artificial, o segundo é baseado em Sistemas de Lindenmayer, que foram inicialmente utilizados para modelar o crescimento de plantas e que foi utilizado nesta pesquisa como metáfora da codificação cromossômica e do crescimento de RNAs, sendo as mesmas tratadas no quarto capitulo. No quinto capítulo apresenta-se a metodologia da proposta que permite obter arquiteturas satisfatórias de RNAs, partindo de estruturas simples evoluindo para arquiteturas mais complexas e que aprendam determinadas tarefas.

3 MODELOS DE DESENVOLVIMENTO BIOLÓGICO

3.1 Introdução

O desenvolvimento de organismos multicelulares a partir de uma única célula, o ovo fecundado é um brilhante triunfo da evolução. Durante o desenvolvimento embrionário, ocorre a divisão do ovo que dá origem a muitos milhões de células, que formam estruturas tão complexas e variadas como olhos, braços, coração e cérebro.

Este capítulo inicialmente apresenta os conceitos de desenvolvimento biológico . Adicionalmente introduz-se os Sistemas de Lindenmayer, que foram utilizados inicialmente para modelar o desenvolvimento de plantas e nesta pesquisa como modelos de desenvolvimento de RNAs .

3.2 A Divisão Celular

No capítulo anterior foi apresentado de forma simplificada, como o DNA codifica as proteínas celulares. Entretanto o mesmo autoduplica-se durante o processo de divisão celular. Sabe-se que o crescimento de organismos é dado pela multiplicação de células, sendo que milhões de divisões celulares ocorrem ao longo da vida de um organismo. No momento em que a célula se divide a molécula de DNA multiplica-se para continuar a síntese protéica nas células filhas.

É sabido que, em organismos superiores com reprodução sexuada, um indivíduo é formado a partir de uma célula ovo, a qual é resultante da fusão de um óvulo e um espermatozóide, os *gametas*. Da célula ovo até se formar um organismo adulto, que possui aproximadamente 10 trilhões de células, ocorrem inúmeras mitoses. Entretanto a formação dos gametas requer um outro tipo de divisão celular. Para manter o número cromossômico da espécie, o óvulo e o espermatozóide devem conter a metade do número de cromossomos presentes nas outras células. A divisão celular que é capaz de reduzir o número cromossômico à metade é chamada *meiose*, (Figura 3.1 b).

Na meiose primeiro ocorre uma divisão na qual os cromossomos homólogos se separam, os dois elementos que formam um par cromossômico. As duas células, resultantes dessa primeira divisão, já têm a metade dos cromossomos da célula inicial.

Em uma segunda etapa, separam-se as cromátides-irmãs. Assim na meiose, para cada duplicação do DNA ocorrem duas divisões celulares. Dessa forma, uma célula com dois cromossomos (formando um par) dará origem, por meio da meiose, a 4 células com 1 cromossomo cada uma (FARAH, 1997).

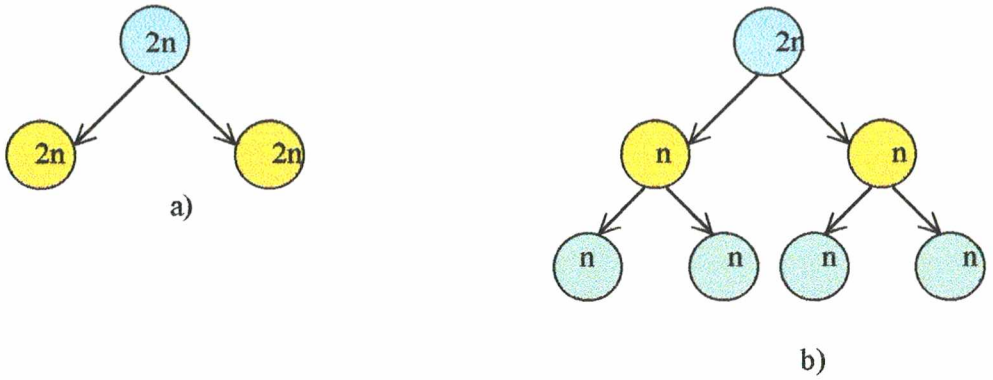


Figura 3.1 Divisão celular a) Mitose, b)Meiose, adaptado de (FARAH, 1997)

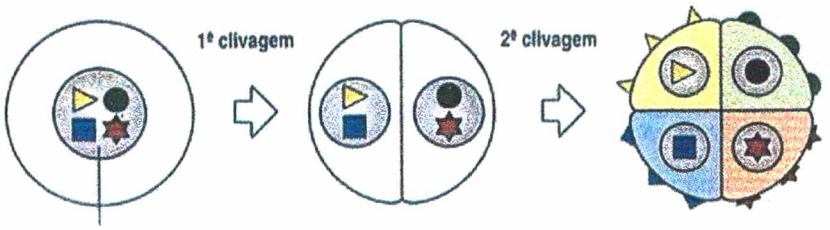
O DNA costuma ser referido como o material genético ou hereditário da célula, pois o conjunto dos genes ali presente representa todo o material que um organismo herda dos seus antepassados, o qual é necessário e suficiente para o controle de seu desenvolvimento. Sendo o DNA que define o tipo de organismo que se desenvolverá.

3.3 Fundamentos de Biologia do Desenvolvimento

O desenvolvimento consiste no surgimento de estruturas organizadas a partir de um grupo inicialmente bastante simples de células. É conveniente distinguir cinco processos de desenvolvimento (fertilização, formação de padrão, mudança de forma, diferenciação celular e crescimento), mesmo que na realidade eles se sobreponham e exerçam influência uns sobre outros (WOLPERT, 2000).

A ***fertilização*** é seguida por um período de rápida divisão celular, onde o ovo divide-se em um grupo de células menores. Essas divisões são conhecidas como clivagens. O ciclo celular, durante a clivagem, consiste simplesmente das fases: replicação do DNA e mitose, sem nenhum estágio interveniente de crescimento celular. Assim o estágio de clivagem da embriogênese rapidamente divide o embrião em um grupo de diversas células, cada uma delas contendo uma cópia do genoma, o processo de clivagem é mostrado na Figura 3.2.

A ***formação do padrão*** é o processo pelo qual um padrão especial e temporal de atividades celulares é organizado dentro do embrião, de modo que uma estrutura bem ordenada se desenvolve. No desenvolvimento do braço, por exemplo, a formação do padrão é o processo que capacita as células à "saber" se elas devem fazer parte do braço ou dos dedos ou onde os músculos devem ser formados. Não existe uma única estratégia ou mecanismo de padronização universal, isso é conseguido por uma variedade de mecanismos celulares e moleculares em diferentes organismos e em diferentes estágios de desenvolvimento (WOLPERT, 2000).



Fatores nucleares especiais ou determinantes

Figura 3.2 – Divisões Celulares: Clivagens, adaptado de (WOLPERT, 2000).

A formação do padrão inicialmente envolve o estabelecimento do *plano corporal* geral, definindo os eixos principais do embrião, de modo que as extremidades anterior e posterior e os lados dorsal e ventral do corpo sejam especificados. Pelo menos um eixo principal do corpo pode ser distinguido em todos os organismos multicelulares. Em animais isso se refere ao eixo que vai da "cabeça" à "cauda". Uma característica interessante destes eixos é que eles estão quase sempre perpendiculares um em relação ao outro. Os dois eixos podem, por isso, ser visualizados como formando um sistema de coordenadas (WOLPERT, 2000), (Figura 3.3).

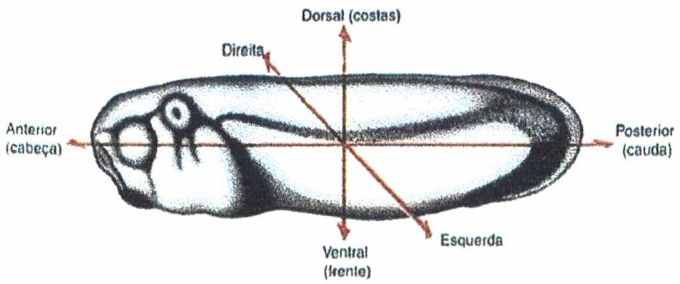


Figura 3.3 – Embrião de Xenopus Laevis no estágio de brotamento da cauda, adaptado de (WOLPERT, 2000).

O próximo estágio na formação de padrão em embriões de animais é o posicionamento de células em diferentes camadas germinativas, a ectoderme, a mesoderme e a endoderme. Durante as etapas restantes da formação de padrão, as células das camadas germinativas adquirem diferentes identidades, de modo que surgem padrões espaciais organizados de diferenciação celular, como o arranjo de pele, músculo e cartilagem, no desenvolvimento de membros, e o arranjo de neurônios no sistema nervoso, Figura 3.4.

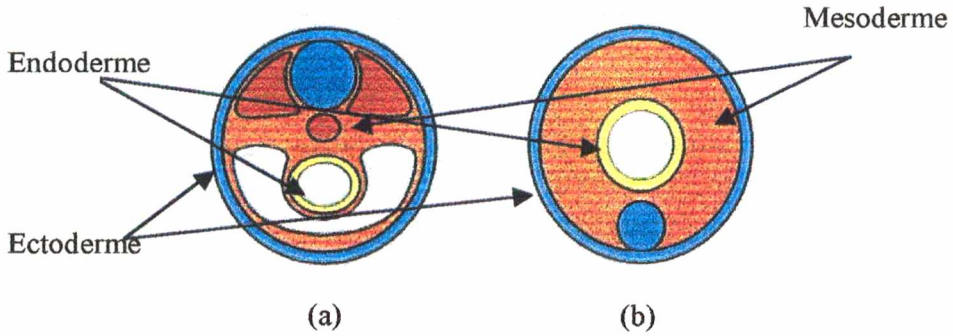


Figura 3.4 – Camadas germinativas a) vertebrados b) insetos, adaptado de (WOLPERT, 2000).

Tabela 3.1 – Camadas Germinativas e órgãos

Camadas Germinativas	**Órgãos**	
Endoderme	Intestino, fígado, pulmões	Intestino
Mesoderme	Esqueleto, músculo, rim coração	Músculo, coração, sangue
Ectoderme	Pele, sistema nervoso	Cutícula, sistema nervoso

Fonte: WOLPERT, Lewis; et. al. (2000) Princípios de Biologia do Desenvolvimento. Trad. Henrique Bunselmen Ferreira. Porto Alegre: Artes Médicas Sul.

O terceiro processo importante do desenvolvimento é a ***mudança na forma ou morfogênese***. Os embriões passam por mudanças marcantes na sua forma tridimensional. Em certos estágios do desenvolvimento ocorrem mudanças características e dramáticas na forma, das quais a gastrulação é a mais notável. Quase todos os embriões de animais passam por esse processo, durante o qual o intestino é formado e os planos principais do corpo emergem (WOLPERT, 2000).

O quarto processo no desenvolvimento é a ***diferenciação celular***, no qual as células se formam estrutural e funcionalmente diferentes umas das outras, acabando como tipos celulares distintos, como células do sangue, de músculo ou de pele. No

homem, o ovo fertilizado dá origem a pelo menos 250 tipos de células claramente distintos (WOLPERT, 2000).

O quinto processo é o ***crescimento***, ou aumento do tamanho. Em geral, há pouco crescimento durante o desenvolvimento embrionário inicial e o padrão básico e a forma do embrião são estabelecidos em pequena escala, sempre com tamanho menor do que uns poucos milímetros. O crescimento subseqüente pode acontecer de diversas maneiras: multiplicação celular, aumento do tamanho das células e deposição de materiais extracelulares como osso e concha (WOLPERT, 2000).

3.4 Sistemas de Lindenmayer

3.4.1 Introdução

Com o objetivo de modelar o desenvolvimento biológico de plantas, o biólogo Aristid Lindenmayer (LINDENMAYER, 1968) desenvolveu uma formalização matemática denominada Sistemas de Lindenmayer, que são um novo tipo de mecanismo de reescrita de cadeias, uma espécie de gramática. A reescrita é uma técnica de definição de objetos complexos, sucessivamente repassando partes de um objeto inicial, usando um conjunto de regras de produção. A diferença essencial entre as gramáticas de Chomsky e os Sistemas de Lindenmayer é a maneira de aplicação das regras de produção. Nas gramáticas de Chomsky (CHOMSKY, 1956) elas são aplicadas seqüencialmente, enquanto que nos Sistemas de Lindenmayer as mesmas são aplicadas em paralelo e simultaneamente repassam todas as letras em uma palavra dada (PRUSINKIEWICZ e HANAN, 1990).

3.4.2 Definição de Sistemas de Lindenmayer

Os Sistemas de Lindenmayer podem ser definidos como uma gramática **G** de uma linguagem **L**, $G=\{\Sigma, \Pi, \alpha\}$, onde Σ é o conjunto finito de símbolos ou alfabeto da linguagem. O Π é o conjunto finito de regras de produção, $\alpha \in \Sigma^*$ é a cadeia de caracteres de início (axioma) de **L** onde $\Pi=\{\pi|\pi:\Sigma\rightarrow\Sigma^*\}$ é o conjunto de regras de produção (BOERS, 1995).

3.4.3 Sistemas de Lindenmayer Livres do Contexto

Esses sistemas são os tipos mais básicos de "L-systems". As regras de produção são da forma **Predecessor→Sucessor**, a interpretação é da seguinte forma: o predecessor é substituído pelo sucessor.

Como exemplo ilustrativo da utilização de Sistemas de Lindenmayer livres do contexto, considere a gramática $G=\{\Sigma, \Pi, \alpha\}$ com $\Sigma=\{A,B,C\}$, $\Pi=\{A\rightarrow BA, B\rightarrow CB, C\rightarrow AC\}$ e $\alpha=ABC$, considerando-se as regras de produção aplicadas ao axioma, obtem-se a linguagem **L={ABC, BACBAC, CBBAACCBBAAC}**.

O exemplo apresentado anteriormente mostra que cada regra de produção é aplicada em paralelo em passos consecutivos de reescrita. Após cada passo de reescrita, todos os símbolos da cadeia de caracteres original, que sejam idênticos aos do lado esquerdo das regras de produção, são substituídos pelos sucessores das regras de produção, como definido pelo lado direito das mesmas. A linguagem L é o conjunto de todas as cadeias que são geradas aplicando-se as regras de produção.

3.4.4 Sistemas de Lindenmayer Sensíveis ao Contexto

Uma extensão para os Sistemas de Lindenmayer, apresentados em 3.4.3 é o sensível ao contexto, que é usado para gerar regras de produção condicionais. A regra de produção $A<B\rightarrow C$ expressa que **B** deve ser reescrito somente se é precedido por **A**, em geral as regras de produção sensíveis ao contexto terão a forma: $L<P>R\rightarrow S$ com $P \in \Sigma$ e $L,R,S \in \Sigma^{+}$. **P** é o predecessor e **S** o sucessor, são o que anteriormente foi chamado de contexto esquerdo e direito respectivamente da regra de produção, o que determina se a regra de produção será aplicada ao símbolo **P** na cadeia de caracteres. O símbolo **P** será reescrito somente se tiver **L** a sua esquerda e **R** a sua direita (BOERS, 1995).

Como exemplo, suponha a seguinte cadeia de caracteres **ABBACAADBAABBAC**, com as seguintes regras de produção: $CA<A\rightarrow EG$, $A<A>B\rightarrow BE$ e $B\rightarrow RT$. A cadeia de caracteres que resulta após uma iteração é: **ARTRTACAEGDRTABERTRTAC.**

3.4.5 Sistemas de Lindenmayer Estocásticos

Os Sistemas de Lindenmayer Estocásticos são sistemas onde entre as regras de produção são aplicadas dependendo de valores de probabilidade. Os valores probabilísticos são listados em cima da seta →. A forma geral da regra de produção é:

$$\text{Predecessor} \xrightarrow{0.10} \text{Sucessor}$$

Os Sistemas de Lindenmayer Estocásticos são usados para alcançar variações entre modelos, por exemplo, variações entre espécies de plantas (VARRIO, 1999).

3.4.6 Aplicações de Sistemas de Lindenmayer

a) Simulação do Desenvolvimento de Filamento de Bactéria

Uma aplicação muito interessante de aplicação dos Sistemas de Lindenmayer livres de contexto é a que simula o desenvolvimento de um filamento multicelular como encontrado na bactéria "Anabaena catenula" e várias algas (MITCHISON e MICHAEL, 1972). Os símbolos "a" e "b" representam estados citológicos da célula e o tamanho das mesmas. Os símbolos "l" e "r" indicam a polaridade da célula, especificando as posições nas quais as células filhas do tipo "a" e tipo "b" serão produzidas. O desenvolvimento é governado pelas regras de produção (VAARIO, 1999).

P1:ar→albr P2:al→blar P3:br→ar P4:bl→al

Sendo w=ar o axioma, as regras de produção geram a seguinte seqüência de palavras.

ar , albr, alarar, alalbralbr, blarblararblarar

Olhando com o microscópio, a célula aparece como cilindros de vários comprimentos. As células tipo "a" são maiores que as células tipo "b". A imagem esquemática do filamento desenvolvido é mostrada na Figura 3.5 (VAARIO, 1993).

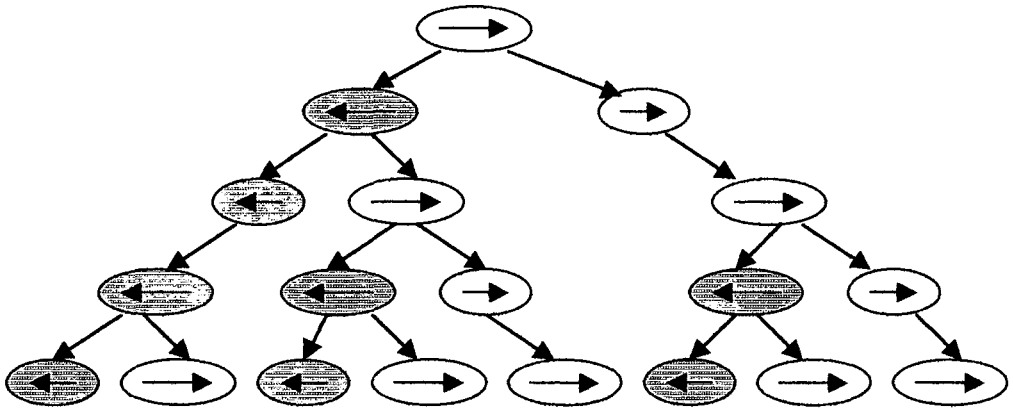


Figura 3.5 – Desenvolvimento de um filamento "Anabaena catenula", adaptado de (VAARIO, 1993).

b) Interpretação do Movimento de Tartaruga

O nome da aplicação interpretação de tartaruga é devido o desenvolvimento de um modelo que pode ser entendido como o movimento de uma tartaruga em um espaço bidimensional ou tridimensional (ABELSON e diSESSA, 1982)

O estado da tartaruga é definido pela tripla (x,y,α), onde a coordenada cartesiana (x,y) representa a posição da tartaruga, e o ângulo α que indica a direção da mesma. Dados o passo de comprimento d e o ângulo de incremento δ, a tartaruga pode responder aos seguintes comandos, mostrados na Tabela 3.2. Um exemplo é mostrado na Figura 3.6.

Tabela 3.2 Comandos para o movimento da Tartaruga

	Ação
F	Mover adiante um passo de comprimento d. O estado da tartaruga muda para (x',y',α), onde $x'=x+d.\cos(\alpha)$ e $y'=y+d.\text{sen}(\alpha)$. Uma linha é desenhada entre os segmentos (x,y) e (x',y').
f	Mover adiante um passo de comprimento d sem desenhar uma linha.
+	Virar à direita por um ângulo de δ . O próximo estado da tartaruga $(x,y, \alpha+\delta)$ É assumido aqui que a orientação positiva do ângulo Virar à direita por um ângulo de δ, como sendo no sentido horário.
-	Virar à esquerda por um ângulo de δ . O próximo estado da tartaruga $(x,y, \alpha-\delta)$

Fonte: PRUSINKIEWICZ, P. e HANAN, J. (1990); Lindenmayer systems, fractals and plants. Springer-Verlag, New York.

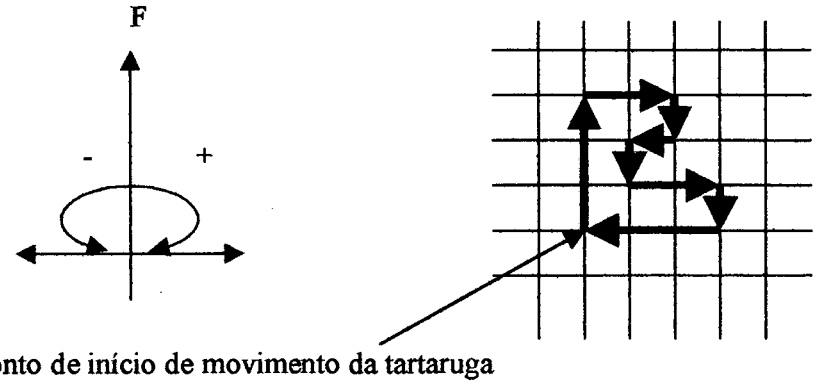


Figura 3.6 (a) Interpretação dos símbolos F, -, +. (b) Interpretação da seqüência de símbolos FFF+FF+F+F-F-FF+F+FFF, para um ângulo de incremento $\delta = 90^{o}$ *, adaptado de* (PRUSINKIEWICZ, P. e HANAN, J. , 1990).

c) Sistemas de Lindenmayer com Memória

Em 1968 (LINDENMAYER, 1968) introduziu uma notação para armazenar estados, com o objetivo de modelar ramificações encontradas em algumas plantas, desde de algas até árvores. Para isso dois novos símbolos foram adicionados a representação apresentada em 3.4.6 (b), o que é mostrado a seguir.

'[' Armazenar o estado atual em uma pilha, a posição da tartaruga e a orientação, assim como outros atributos como a cor e a largura das linhas desenhadas.

']' Recuperar o estado da pilha e fazer dele o estado corrente. Não é desenhada linha, entretanto em geral a posição da tartaruga muda.

Como exemplo de utilização considere a seqüência F[-F][+F[+F]F]F[-F][+F], a interpretação para a seqüência é mostrada na Figura 3.7. Outros exemplos de utilização de Sistemas de Lindenmayer com memória para modelamento do crescimento de plantas são mostrados na Figura 3.8.

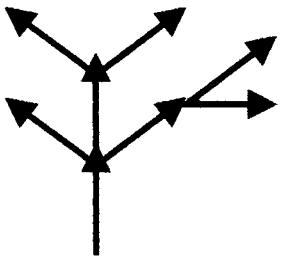
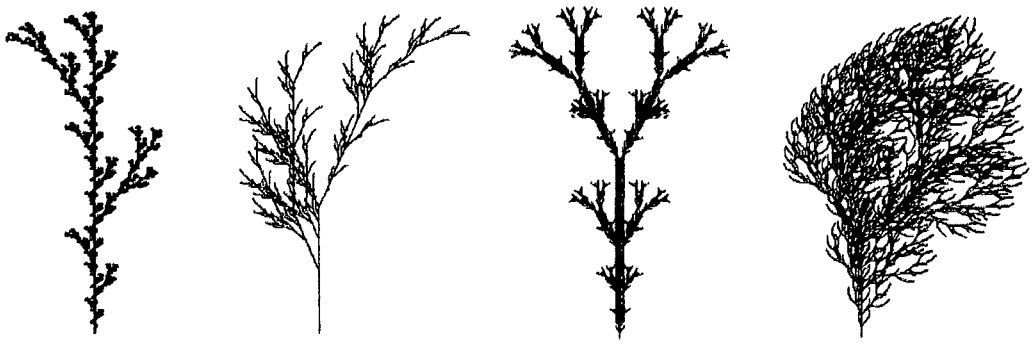


Figura 3.7 Exemplo de Sistemas de Lindenmayer com memória, adaptado de (PRUSINKIEWICZ, P. e HANAN, J. , 1990).



a)
n=5, δ=25.7°
F
F→F[+F]F[-F]F

b)
n=5, δ=22.5°
X
X→F-[[X]+X]+F[+FX]-X
F→FF

c)n=6, δ=25.7°
Y
Y→YFX[+Y][-Y]
X→X[-FFF][+FFF]FX

d) n=7, δ=22.5°
F
F→FF+[+F-F-F]-[-F+F+F]

Figura 3.8 Sistemas de Lindenmayer com memória para modelamento de plantas.

3.5 Do Modelo de Desenvolvimento Biológico para o Artificial

O processo de desenvolvimento biológico como um todo é desafiador e complexo, para se computar o embrião, talvez fosse necessário computar o desenvolvimento de todas as células que o constituem. Os modelos que tentam simular o desenvolvimento artificialmente adotam simplificações, no sentido de encontrar um nível de comportamento celular que explique adequadamente o mesmo, mas que não precise levar em consideração o comportamento pormenorizado de cada célula.

No quinto capítulo apresenta-se um modelo simplificado que simula o desenvolvimento de RNAs, considerando alguns aspectos aqui discutidos. O mesmo foi baseado nos trabalhos de (BOERS e KUIPER, 1992) que utilizaram Sistemas de Lindenmayer para geração de topologias de grafos que gerem redes diretamente alimentadas e (BOERS, 1995) que desenvolveu uma interpretação generalizada, permitindo a construção de grafos com ciclos, que possibilita a obtenção de geração de redes recorrentes, essa construção é denominada gramática para geração de grafos "G2L-Systems".

4. REDES NEURAIS

4.1 Introdução

Diante da complexidade do seu objeto de investigação a neurociência é a ciência que estuda o sistema nervoso e a suas manifestações, sendo dividida em várias áreas de especialização: neuroanatomia, neurofisiologia, neuropsicologia, psicologia, sendo as mais importantes. A neurociência cognitiva procura unir a neurociência e a ciência cognitiva (ciência que se propõe a modelar os processos de percepção e cognição) com metodologias emprestadas da psicologia, inteligência artificial e modelos matemáticos associados (KOVÁCS, 1997).

Neste capítulo, inicialmente, apresenta-se o modelo biológico do neurônio, seu funcionamento e a organização interna do cérebro, o que permite avançar milimetricamente o entendimento dos mecanismos do sistema nervoso. Em seguida são apresentados o modelo artificial de neurônio e de RNAs, estáticas e dinâmicas, bem como os mecanismos de aprendizado utilizados. Procura-se relacionar as arquiteturas de redes neurais apresentadas com a capacidade de resolução de diferentes classes de problemas.

4.2 O Neurônio Biológico: Caracterização e Funcionamento

O tecido nervoso seja ele do encéfalo, da medula espinhal ou dos nervos periféricos, contém dois tipo de células: os **neurônios** que conduzem os sinais pelo sistema nervoso. Existem 100 bilhões dessas células em todo o sistema nervoso central e as células de suporte ou de isolamento, que mantém os neurônios em suas posições e evitam que os sinais sejam dispersos entre os neurônios e suas estruturas intercelulares; são em conjunto chamadas de **neuroglia**. No sistema nervoso periférico são chamadas as **células de Schawann** (KOVÁCS, 1997).

Existem vários tipos de neurônios, cada um com sua função, forma e localização específica, entretanto os componentes principais de um neurônio de qualquer tipo são: o corpo do neurônio, chamado soma, os dendritos e o axônio como ilustrado na Figura 4.1.

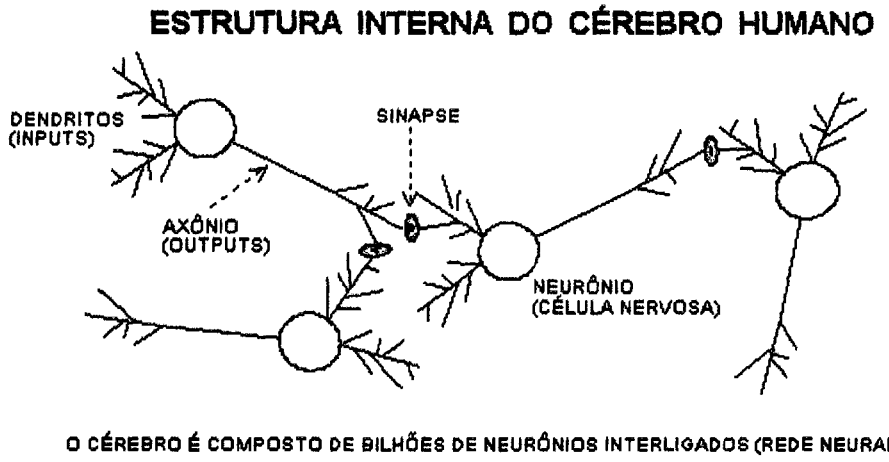


Figura 4.1 – Neurônio Biológico adaptado de (GUYTON, 1998)

O **corpo do neurônio** mede entre 5 a 10 mm de diâmetro e contém o núcleo da célula. A maioria das atividades bioquímicas necessárias para manter a vida do neurônio, tais como síntese de enzimas, são realizadas no seu corpo.

Os **dendritos** atuam como canais de entrada para os sinais externos provenientes de outros neurônios e o axônio atua como canal de saída e transmissão. Os dendritos formam a árvore dendrítica, uma estrutura que se espalha em torno do corpo do neurônio dentro de uma região de até 400mm em raio. O **axônio** se estende do corpo do neurônio e é relativamente uniforme em diâmetro. O axônio pode ser tão curto quanto 10mm (no caso dos interneurônios, neurônios que funcionam apenas como pontos de ligação com outros neurônios) ou tão longos quanto 1m (no caso dos neurônios motores, usados para estimular músculos, neurônios que conectam o dedo do pé à espinha vertebral). O axônio também se espalha em uma estrutura de árvore com vários ramos, mas geralmente apenas no seu fim, enquanto que os dendritos se espalham muito mais perto do corpo do neurônio (KOVÁCS, 1997).

O final dos ramos do axônio tem a forma de um botão com diâmetro de 1mm é conectado ao dendrito de outro neurônio. Tal conexão é chamada de **sinapse**. Normalmente a mesma não é física, mas eletroquímica. Existe um pequeno espaço entre eles chamado de fenda sináptica de 200 Å A 500 Å (1 Å $= 10^{-10}$ m). As sinapses influenciam as atividades de outras células da seguinte maneira:

O corpo do neurônio pode gerar uma atividade elétrica na forma de pulsos de voltagem chamados de **potencial de ação**. O axônio transporta o potencial de ação do corpo do neurônio até as sinapes, onde certas moléculas, chamadas neurotransmissores, são liberadas. Essas moléculas atravessam a fenda sináptica e modificam o potencial da membrana do dendrito. Dependendo do tipo de neurotransmissor predominante na sinapse, o potencial da membrana do dendrito pode aumentar (sinapse excitatória) ou diminuir (sinapse inibidora). Esses sinais recebidos pelos dendritos de vários neurônios são propagados até o corpo do neurônio onde são aproximadamente somados. Se essa soma dentro de um pequeno intervalo de tempo for superior de um determinado limite, o corpo do neurônio gera um potencial de ação que é transmitido pelo axônio dos outros neurônios (KOVÁCS, 1996).

O potencial de ação tem um valor de pico em tomo de 100 mV e duração em torno de 1 ms. Em repouso, ou seja, sem receber sinais de outros neurônios, o corpo do neurônio tem um potencial próximo de –70 mV em relação ao exterior da célula e o valor limite para geração do potencial se situa entre –30 mV e –60 mV. Depois que o potencial de ação é gerado (Tn), existe um período refratário (Ta) quando o neurônio não gera outro potencial de ação, mesmo que receba uma grande excitação. Seguindo esse período de refração absoluta, persiste um período de refração relativa (Tr) correspondendo a uma elevação no limiar de disparo, que assintóticamente retoma ao seu valor normal (KOVÁCS, 1996).

A velocidade de propagação do potencial de ação ao longo do axônio varia de 0,5 a 130 m/s dependendo do diâmetro do axônio e da existência **da bainha de mielina** (uma substância isolante) (KOVÁCS, 1996).

4.3 Redes de Neurônios – O Cérebro

Os neurônios agrupam-se para formar o Sistema Nervoso Central (SNC). Como já foi visto, cada neurônio possui um comportamento bioquímico e bio-elétrico bastante complexo, entretanto, se acredita que seu princípio computacional é simples: ele soma suas entradas, e realiza uma operação de limiar e cada neurônio é capaz de ajustar sua saída como uma função relativamente simples de suas entradas. Acredita-se que a

capacidade do cérebro em produzir comportamentos complexos, deve-se à cooperação e interação de uma grande quantidade de neurônios operando em paralelo (KOVÁCS, 1997).

O cérebro humano possui em torno de 10^{11} neurônios e cada neurônio recebe sinais de 10^4 outros. Descrições mais pormenorizadas sobre a estrutura do cérebro, em particular do cérebro humano, podem ser encontradas em (KOVÁCS, 1996).

Nas próximas seções apresenta-se a abordagem conexionista que foi inspirada no modelo biológico. Inicialmente apresenta-se o neurônio artificial e os dois modelos de RNA utilizados nesta pesquisa.

4.4 Modularidade do Cérebro

O processo de desenvolvimento de sistemas computadorizados biologicamente inspirados envolve a verificação da funcionalidade e da arquitetura do cérebro com ênfase nas atividades de processamento da informação. Os mesmos utilizam evidências experimentais para criar sistemas de computadores inspirados na neurociência.

Várias restrições têm limitado o grau de progresso dos sistemas inspirados biologicamente. Esses modelos apresentam simplificações em relação ao que eles tentam realizar em comparação ao que ocorre no cérebro. Apesar do atual avanço de conhecimento dos níveis de processamento neuronal e conectividade do cérebro, existe muita coisa que não é conhecida sobre o que acontece nos vários níveis do sistema (FRIEDERICI, 2000).

A última década viu um significante aumento de interesse pelo estudo do cérebro, a razão para isso é possibilidade de explorar a inspiração no mesmo possibilitando assim criar sistemas inspirados biologicamente com melhor desempenho (BORISYUK et.al. ,2001). Existem várias características de processamento de informação e arquiteturais que podem ser incluídas em sistemas de computadores que possibilita aos mesmos adquirirem melhor desempenho tais como: organização modular, robustez e memórias de armazenamento.

A modularidade do cérebro desenvolveu-se por milhões de anos de evolução com o objetivo de executar funções cognitivas usando uma estrutura compacta e adquiriu várias formas de organização: neurônios, colunas, regiões e hemisférios. O

cérebro pode ser visto como várias redes de neurônios em diversas regiões que podem realizar processamento paralelo para executar funções cognitivas específicas (DODEL et. al., 2001), (KNOBLAUCH e PALM., 2001). O mesmo pode ser comparado como um grupo de especialistas colaborando que executam as funções cognitivas dividindo a tarefa em elementos menores (REILLY, 2001).

O córtex cerebral é a maior parte do cérebro humano sendo organizado em muitas regiões que são responsáveis por uma modularidade de alto nível. Uma característica da modularidade é a divisão de atividades requeridas para executar uma função cognitiva entre diferentes hemisférios do cérebro. Técnicas de mapeamento do cérebro permitem obter uma grande quantidade de informações de regiões associadas com funções cognitivas (GAZZANIGA et. al., 1998), (BINDER, 1995).

Com objetivo de estudar mais detalhadamente a estrutura modular do cérebro GUIGON et.al. (1994) baseou-se no fato de que o córtex cerebral é composto basicamente de blocos de módulos repetidos denominados colunas corticais com basicamente a mesma estrutura de seis camadas. De acordo com REILLY (2001) as colunas podem ter 0.03mm de diâmetro e incluem em tomo de 100 neurônios, o desenvolvimento dessa estrutura foi resultado de uma necessidade evolucionária por uma melhor funcionalidade e largura de banda do sistema motor. O que se observa no córtex é uma estrutura em camadas e uma abordagem de processamento extremamente recorrente (DOZA,1999).

Analisando-se características gerais de organização estrutural do cérebro, pode-se identificar uma estrutura horizontal e outra vertical. A primeira é identificada onde o processamento no mesmo é executado por camadas hierárquicas de neurônios subseqüentes. Dessa forma o processamento da informação multi-estágio pode, por exemplo, ser identificado no sistema visual primário onde simples características de imagens visuais como linhas e arcos são representadas em camadas de neurônios simples as mesmas são combinadas e representadas por neurônios em camadas subseqüentes que possuem capacidades representacional mais complexas, formando imagens com maior complexidade que são interpretadas pelo córtex (BINDER, 1995).

A estrutura vertical do cérebro permite o processamento separado de diferentes tipos de informação. Um bom exemplo é encontrado no sistema visual, onde diferentes tipos de estímulos visuais tais como forma, coloração, movimento e posição são processados em paralelo por sistemas neurais anatomicamente separados organizados no caminho magno celular e parvo celular. Estruturas convergentes integram esta informação visual processada separadamente em níveis hierárquicos superiores para produzir uma percepção unitária (BOERS e KUIPER, 1992).

Uma vantagem funcional importante da separação anatômica de diferentes funções é a minimização da interferência mútua entre os processos simultâneos e execução de tarefas diferentes, necessárias para executar habilidades complexas como, por exemplo, dirigir um carro. Algumas tarefas são processadas em módulos separados e não se interferem. Outras requerem processos simultâneos em módulos únicos sendo dessa forma difíceis de executar em paralelo.

A modularização pode ser dividida em duas partes distintas, a modularização repetitiva e a diferenciada. A primeira refere-se ao uso do mesmo tipo de módulo em tempos distintos e a segunda ao uso de módulos distintos para organização de um todo.

O cérebro humano trabalha dessa forma, onde vários processos podem estar sendo executados em paralelo e hierarquicamente. Por exemplo, quando se escreve um livro seja manuscrito ou digitado, trabalha-se com as mãos, mas as idéias estão sendo processadas pelo cérebro. Conforme as idéias vão surgindo o cérebro "dá" o comando para o centro o motor mover as mãos e escrever as letras. Nesse caso existem dois processos distintos executados paralelamente, pode acontecer de vários processos estarem sendo executados simultaneamente.

4.5 Abordagem Conexionista

4.5.1 Introdução

Na atualidade, as pesquisas na área de arquitetura de computadores estão empenhadas em desenvolver estruturas altamente paralelas com a finalidade de otimizar os tempos de busca e execução de instruções.

Apesar do alto grau de paralelismo existente nestas arquiteturas, elas se baseiam no princípio de execução de instruções para realizar o processamento desejado, adotando uma abordagem algorítmica para soluções de problemas. De acordo com a metodologia proposta por Barreto (BARRETO, 1990) e desenvolvida por de Azevedo (de AZEVEDO, 1996), um Computador Baseado em Instruções (CBI) é qualquer computador que funcione baseado na execução de instruções para a realização do processamento. A abordagem algorítmica pode ser muito eficiente para solução de problemas onde já se conhece a seqüência precisa de instruções a serem executadas.

Por outro lado, apesar de toda tecnologia desenvolvida tanto a nível de "software" como de "hardware" atualmente, existem muitos problemas que os seres vivos, e em particular os seres humanos, resolvem de maneira inata, ou seja, é muito difícil programar computadores para automatizar tarefas que os seres humanos podem realizar com pouco esforço, tais como dirigir um carro ou reconhecer faces e sons nas situações encontradas na vida real, visto que o algoritmo ou conjunto de regras usados pelos seres humanos para realizar estas tarefas não é conhecido (o conhecimento não está explícito). Mesmo problemas mais corriqueiros, tais como simples locomoção autônoma no mundo real é tarefa ainda mais desafiadora para ser implementada da maneira algorítmica em CBIs.

As RNAs fornecem um enfoque alternativo a ser aplicado em problemas onde os métodos algoritmos não são julgados muito adequados. Como a unidade básica de funcionamento do sistema nervoso parece ser uma estrutura celular chamada neurônio e o comportamento inteligente parece emergir da interconexão entre estas células, esta nova abordagem é chamada neural ou conexionista.

Os computadores desenvolvidos utilizando esta abordagem são chamados de Computadores Baseados em Redes Neurais (CRN). Ainda segundo De AZEVEDO (1996), é importante diferenciar RNAs de CRN. Pode-se dizer que uma Rede Neural Artificial esta para um CRN, assim como um processador esta para um CBI, ou seja, em termos de computador é necessário ainda considerar todas as interfaces com o mundo exterior. Além disso, um CRN não é composto de apenas de uma Rede Neural Artificial, mais sim por uma série de Neurônios Artificiais (NA) que podem possuir

diferentes funções de ativação e de saída e que podem, ainda, ser interconectados segundo diferentes topologias.

4.5.2 Histórico

Em 1943, Warren McCulloch e Walter Pitts estabeleceram as bases da neurocomputação, mostrando que qualquer função lógica poderia ser configurada através de um sistema de neurônios interconectados (um dos primeiros computadores digitais – O ENIAC surgiu em 1946), concebendo procedimentos matemáticos análogos ao funcionamento dos neurônios biológicos. Esse desenvolvimento foi puramente conceitual, uma vez que estes autores não sugeriram aplicações práticas a partir do seu trabalho, mesmo porque os sistemas propostos não tinham a capacidade de aprendizado (SIMPSON, 1990) (GORNI, 1994).

Em 1949, Donald Hebb sugeriu no seu livro "Organization of Behavior", um modo de se proporcionar capacidade de aprendizado às redes neurais artificiais. Através de experiências com animais, ele propôs que as mudanças nas forças das sinapses são proporcionais às ativações das mesmas. Este princípio, traduzido matematicamente, viabilizou o desenvolvimento de redes neurais artificiais eficazes. Esses princípios foram aplicados por Marvin Minsky na construção do Snark, o primeiro neurocomputador que se tem notícia, em 1951. Tecnicamente ele foi um sucesso, uma vez que ajustava automaticamente os pesos entre as diversas sinapses, ou seja, demonstrava ao menos teoricamente, que tinha capacidade de aprendizado. Contudo este dispositivo nunca executou qualquer função útil no campo de processamento da informação, constituindo-se apenas em uma curiosidade acadêmica (GORNI, 1994).

.

Em 1958, Frank Rosenblatt concebeu um dispositivo denominado de perceptron, que era uma rede neural com duas camadas de neurônios capaz de aprender de acordo com as regras propostas por Hebb. Tal aparelho tinha capacidade de treinamento supervisionado e foi utilizado para reconhecimento de caracteres. Pela primeira vez, as redes neurais artificiais foram utilizadas com sucesso em uma aplicação prática. O êxito conseguido por esta abordagem fez com que muitos considerem Rosenblatt como o verdadeiro pai da neurocomputação (ROISENBERG, 1998).

Bernard Widrow, em 1960, desenvolveu um tipo diferente de processador para redes neural, denominado ADALINE, o qual dispunha de uma poderosa estratégia de aprendizado.

Contudo, após esses espetaculares desenvolvimentos, a neurocomputação entrou em uma grave crise com a publicação do trabalho de MINSKY e PAPERT (1969) que demonstrava, de maneira formal, que o modelo de Rosenblatt era incapaz de aprender padrões não-linearmente separáveis (o famoso problema do OU-EXCLUSIVO). Minsky & Papert foram extremamente pessimistas em relação ao futuro das redes neurais artificiais.

Rosenblatt propôs como solução aumentar o número de camadas, mas, apesar de toda a sua visão e perspicácia neste campo, não logrou desenvolver um método de aprendizado eficaz para estas redes neurais mais avançadas (GORNI, 1994).

Apesar do descrédito gerado sobre a área de neurocomputação, entre 1967 e 1982 os estudos neste campo continuaram, ainda que englobadas em outras linhas de pesquisa, como processamento adaptativo de sinais, reconhecimentos de padrões, modelamento biológico, etc. Este trabalho, ainda que silencioso, construiu as bases necessárias para que o desenvolvimento das redes neurais pudesse continuar de forma consistente.

Em 1974 aconteceu um fato que viria, mais tarde, a proporcionar o renascimento do interesse geral pelas potencialidades das redes neurais, foi quando Paul Werbos lançou as bases do algoritmo de retro-propagação ("backpropgation"), que permitia que redes neurais com múltiplas camadas apresentassem capacidade de aprendizado. Contudo, a potencialidade desse método tardou a ser reconhecida (GORNI, 1994).

Em 1982 John Hopfield publicou um estudo que chamava a atenção para as propriedades associativas de uma classe de redes neurais que apresentava fluxos de dados multidirecionais e comportamento dinâmico. Primeiramente ele mostrou que o sistema possuía estados estáveis e, posteriormente, que tais estados poderiam ser criados alterando-se os pesos das conexões entre os neurônios.

No entanto, os primeiros resultados levaram a retomada de desenvolvimento sobre redes neurais foram publicadas em 1986 e 1987, através dos trabalhos do grupo PDP (Parallel and Distributed Processing), MCCLELLAND, et. al. (1986), onde ficou consagrada a técnica de treinamento por "backpropagation". Nesta fase destacaram-se o surgimento do treinamento não-supervisonado, proposto por Teuvo Kohonen em 1984 e as novas topologias de redes neurais como a proposta por Bart Kosko, em 1987 (BAM-BiDirecional Associative Memory) (ROISENBERG, 1998).

4.5.3 O Neurônio Artificial

De maneira geral, os elementos básicos que compõem um neurônio artificial, podem ser vistos na Figura 4.2.

Examinando a Figura 4.2, ao nível de rede, cada entrada do neurônio, possui um conjunto de pesos associados. Este conjunto de pesos pode ser entendido como uma "força" da conexão sináptica quando comparada com o neurônio biológico. As entradas de um neurônio podem ser as saídas de outros, entradas externas um "bias" ou qualquer combinação desses elementos. O somatório de todos as entradas da origem ao chamado "net" de um neurônio.

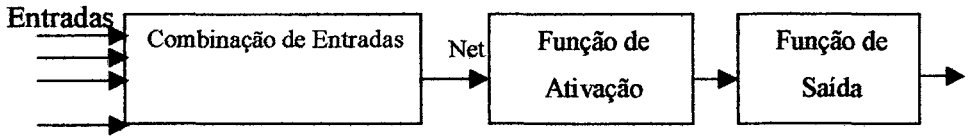


Figura 4.2 – Elementos básicos que compõem um neurônio artificial, adaptado de (ROISENBERG, 1998).

Na notação usualmente adotada na literatura é comum encontrar-se todas as entradas de um neurônio incorporadas sobre a representação uk e os pesos por wij com i=1 até n. A equação do "net" fica então.

$$net_i(t) = \sum_{j=1}^{n} w_{ij} u_j(t) \qquad (4.1)$$

Onde wij é o número real que representa a conexão sináptica de entrada com o $i^{ésimo}$ neurônio de saída do $j^{ésimo}$ neurônio. A conexão sináptica é conhecida como excitatória se wij>0 e inibitória caso wij<0. Após a determinação do net, o valor da

ativação do neurônio é atualizado através da função de ativação e finalmente, o valor de saída do neurônio é produzido através da função de saída. Analiticamente, a ativação e a saída dos neurônios são dadas pelas seguintes relações.

$$x(t+h)=\phi(x(t),net(t),t,h) \tag{4.2}$$

$$y(t)=\eta(x(t),net(t),t) \tag{4.3}$$

A partir destas relações formais, pode-se obter diferentes modelos a partir de modificações na função de ativação e de saída.

A Função de ativação ϕ

Examinando as equações (4.2) e (4.3) pode-se ver que os estados futuros de um neurônio são afetados pelo estado atual do neurônio e pelo valor do net de entrada. Este tipo de neurônio, que possui "memória" é conhecido como "neurônio dinâmico". Por outro lado, se a função é constante, tem-se um neurônio que não possui "memória", ou seja, o estado atual é igual aos estados anteriores e, portanto o neurônio é conhecido como "neurônio estático". As Redes Neurais são sistemas dinâmicos nos quais dependendo da sua função de ativação, podem ser classificados como neurônios estáticos e neurônios dinâmicos, como mostra a Figura 4.3.

A Função de saída η

Essencialmente, qualquer função contínua e monotonicamente crescente tal que $x \in \Re$ e $y(x) \in [-1,1]$ pode ser utilizada como função de saída na modelagem neural. Existem, no entanto, uma série de funções mais comumente utilizadas como funções de saída em neurônios. Estas funções são:

- A função linear $y(x)=ax$; (4.4)
- A função logística, que é a função unipolar mais popular

$$y(x)=\frac{1}{1+e^{-kx}} \tag{4.5}$$

- A função tangente hiperbólica, que é a função bipolar mais popular

$$y(x)=\tanh(kx)=\frac{1-e^{-kx}}{1+e^{-kx}} \tag{4.6}$$

4.6 Redes Neurais Artificiais

Informalmente uma Rede Neural Artificial é um sistema composto por vários neurônios de modo que as propriedades de sistemas complexos sejam usadas. Estes

neurônios estão ligados por conexões, chamadas conexões sinápticas. Alguns neurônios recebem excitações do exterior e são chamados de neurônios de entrada e correspondem aos neurônios dos órgãos dos sentidos. Outros têm suas respostas usadas para alterar de alguma forma o mundo exterior e são chamados neurônios de saída e correspondem aos motoneurônios que são os neurônios biológicos que excitam os músculos. Os neurônios que não são nem entrada nem saída são conhecidos como neurônios internos. Os neurônios internos são importantes por dois aspectos: biologicamente e matematicamente.

A importância biológica corresponde a uma atividade do sistema nervoso que pode apresentar uma independência de excitações externas. Com efeito, se entre estes neurônios houver ligações formando ciclos, e considerando ainda um certo tempo de resposta de um neurônio, após cessar toda a excitação exterior pode haver nestes neurônios internos uma evolução de um vetor representativo da excitação destes neurônios. Esta excitação pode provocar uma evolução durante um tempo relativamente longo e pode ser interpretada como uma metáfora da mente, através dos quais os pensamentos vêm e voltam, sem estímulo anterior. Com relação à importância matemática, ficou provado que sem estes neurônios é impossível uma Rede Neural Artificial resolver problemas classificados como linearmente não separáveis.

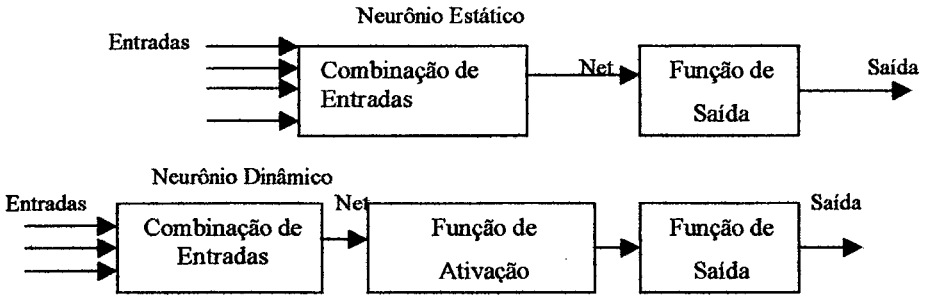


Figura 4.3 – Modelos de neurônios estáticos e dinâmicos, adaptado de (ROISENBERG, 1998)

4.7 Topologias de Redes Neurais

A maneira como os neurônios estão interconectados e as características dinâmicas dos neurônios formam as diferentes topologias de (RNAs) e estão intimamente relacionadas com o algoritmo de aprendizado utilizado para ensinar a rede

e com a capacidade de resolver certas classes de problemas. A seguir serão apresentadas topologias de redes neurais utilizadas no desenvolvimento experimental desta pesquisa.

a) Redes Diretas

As Redes Diretas ("Feedforward") são representadas por grafos que não possuem ciclos. As mesmas apresentam-se organizadas em camadas sendo por isso denominadas redes em camadas. Uma camada é constituída por um conjunto de neurônios que recebem entradas de um mesmo local e tem como saídas também um mesmo local. A camada de entrada é assim denominada, pois recebe todas as entradas do meio exterior e saída nos outros neurônios de forma análoga à camada de saída. Se os neurônios internos se organizam em grupos, cada um recebendo entradas somente de um grupo anterior (entrada) e suas saídas vão para outro grupo, tem-se uma rede em camadas (BOERS e KUIPER, 1992). A Figura 4.4 mostra uma rede direta com três camadas de neurônios, onde as conexões sinápticas são representadas apenas por setas. Estas redes são atualmente as mais populares, principalmente por existirem métodos de aprendizado fáceis para essas redes, um método que é bastante usado é o "backpropagation", cuja formulação matemática é apresentada no ANEXO A. Além disto, estas redes são capazes de aproximar, com maior ou menor precisão, dependendo do número de neurônios da rede, qualquer função não linear.Entretanto, mesmo no caso de usarem neurônios dinâmicos, tem uma dinâmica muito limitada não podendo representar todos os sistemas dinâmicos. Onde U é o conjunto dos valores de entrada, ϕ é a função de transição de estados, λ a função de saída, V os pesos calculados e Y o conjunto das saídas obtidas.

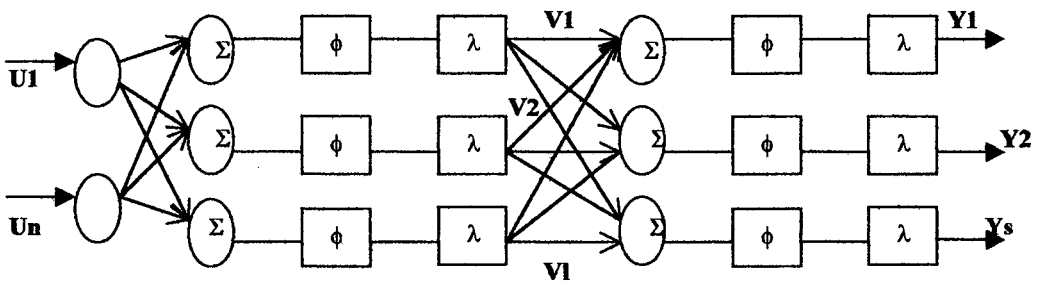


Figura 4.4 Rede Neural Artificial direta multicamada

b) Redes Recorrentes

Redes em ciclo (ou com realimentação, ou com retroação, ou com "feedback") são aquelas cujo grafo de conectividade contém ao menos um ciclo.Estas redes podem ser compostas por uma ou mais camadas sendo que cada neurônio fornece o seu sinal de saída como entrada para cada um dos outros neurônios (BARRETO, 1999) (BOERS e KUIPER, 1992). Também pode não conter uma camada intermediária (escondida).

Exemplos destas redes são as propostas por Hopfield, as Redes Bidirecionais, as Redes de Jordan (o ANEXO B, apresenta a formulação matemática de um algoritmo baseado na regra delta generalizada para redes de Jordan) e as Redes Recorrentes de Tempo Real.

Quando, além disso, envolve neurônios dinâmicos são chamadas recorrentes.As realimentações envolvem a utilização de ramos particulares compostos por atrasos unitários denotados por z^{-1}, o que resulta em um comportamento dinâmico não-linear em virtude da natureza não-linear dos próprios neurônios. Esta característica de dinâmica não-linear da estrutura permite que este tipo de rede neural seja utilizado em problemas e aplicações que necessitem representar estados, tais como: processamento de voz, controle industrial, processamento adaptativo de sinais, predição de séries temporais. A Figura 4.5 ilustra uma topologia recorrente.

As Redes Neurais diretas são capazes de modelar unicamente sistemas estáticos. Nos sistemas estáticos a saída é função apenas de suas entradas. Já as redes neurais recorrentes são por sua própria característica capazes de modelar sistemas dinâmicos. Nestes sistemas, a saída não depende apenas das suas entradas, mas também de variáveis de estado. Estas por sua vez, se alteram em função das entradas do sistema e do seu estado anterior. As redes diretas são capazes de implementar apenas comportamentos reflexivos, ou seja, onde a resposta imediata do ser vivo a um estímulo ambiental e, uma vez cessado o estímulo, cessa também a sua saída correspondente. Entretanto as redes recorrentes são capazes de implementar comportamentos mais complexos, como por exemplo, os comportamentos reativos e instintivos, onde a reação do ser vivo não é função apenas dos estímulos ambientais, mas, também de um estado interno ou "memória" ROISENBERG (1996), BARRETO (1999).

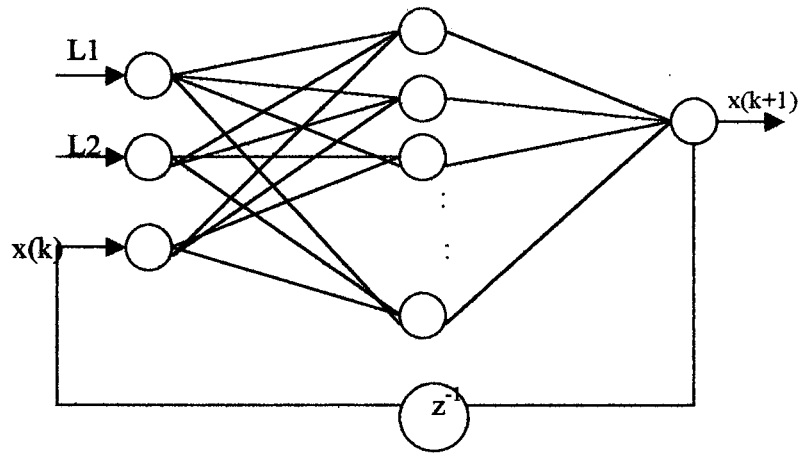


Figura 4.5 Rede Neural Artificial recorrente multicamada, adaptado de (SILVA et. al, 2000)

4.8 Do Modelo Biológico para o Artificial

A construção de RNAs tem inspiração nos neurônios biológicos e nos sistemas nervosos. Entretanto, é importante compreender que atualmente as RNAs estão muito distantes das Redes Neurais Naturais e que freqüentemente as semelhanças são mínimas (BARRETO, 1999).

Muitos pesquisadores têm tentado simular o funcionamento do cérebro e de seus neurônios em computadores digitais, baseados em princípios biológicos. Entretanto o grande número de neurônios e suas conexões tornam praticamente impossível simular o cérebro humano em sua totalidade de detalhes. Estima-se que a taxa de transmissão de informação no cérebro humano é da ordem de 10^{13} bits por segundo e que o total de operações aritméticas necessárias para simular o funcionamento do mesmo em todos os seus detalhes em 10^{8} por segundo, operando sobre uma memória de 10^{16} bytes (BOERS e KUIPER, 1992). Assim, a maioria dos pesquisadores procura desenvolver modelos que simulem apenas aspectos considerados relevantes ou que sejam capazes de validar ou fornecer informações sobre teorias de funcionamento de algum aspecto do cérebro.

As pesquisas que vêm sendo desenvolvidas na área de sistemas de computadores, biologicamente inspirados, tentam identificar a maneira como os módulos no cérebro interagem SPITZER (1999). Em relação à organização modular apresentadas anteriormente, REILLY (2001) identificou rotas diretas e com

realimentação unindo os módulos executando uma função cognitiva particular. Os sistemas biologicamente inspirados procuram, por exemplo, estudar o grau de associação entre essas regiões identificadas como responsáveis por uma tarefa (TAYLOR, 2001).

As abordagens tradicionais de projeto de RNAs não têm o desempenho, a flexibilidade dos sistemas de processamento de informação neurais. Em razão os sistemas biologicamente inspirados, procuram beneficiar-se de várias características do cérebro tais como: organização modular, robustez e memórias de armazenamento. No próximo capítulo apresenta-se uma metodologia de projeto que contempla algumas dessas características.

5 METODOLOGIA DA PROPOSTA

5.1 Introdução

Neste capítulo apresenta-se uma metodologia de projeto automático de RNAs, inspirada biologicamente na Evolução, "Seleção Natural", nos mecanismos que possibilitam a transmissão das características genéticas de geração em geração, no processo de desenvolvimento dos seres vivos e no princípio de organização modular do cérebro. O que se pretende é incorporar a mesma alguns aspectos que foram discutidos nos capítulos 2, 3 e 4, tentando-se aproximá-la do que se acredita que acontece na natureza em relação a esses processos. Inicialmente apresenta-se uma análise de plausibilidade biológica de três métodos de codificação, que juntamente com a função aptidão é essencial para qualquer aplicação que utiliza AG, após isso se detalha a metodologia da proposta utilizada nesta pesquisa.

5.2 Plausibilidade Biológica dos Métodos de Codificação

5.2.1 Método De Codificação Direta

O método de codificação direta é apresentado em detalhes no (ANEXO D, seção D.1). Na verdade, não se sabe como estão codificados nos genes, os esquemas de interconexão entre os neurônios. Entretanto de alguma forma eles estão, já que alguns tipos de comportamentos, são inatos, como é o caso do choro da criança quando nasce. Acredita-se que genes dos cromossomos codificam uma "receita" (genótipo), o processo de crescimento e manutenção de órgãos e do sistema nervoso, resulta na "execução" dessa receita pelas células e na reprodução destas. Com isso, o material genético vai produzir um fenótipo, que é resultado do genótipo interagindo com o ambiente. O processo evolucionário seria guiado, pela obtenção das melhores "regras" (proteínas), que têm a capacidade de funcionarem em conjunto. Neste processo, receitas que deram mais certo, podem ser levadas a serem repetidas mais vezes que outras.

O método de codificação direta tem um grau de plausibilidade baixo. No mesmo, o AG é empregado para evoluir uma população de cromossomos ("strings de bits"), que

equivalem à presença ou não de conexões entre os neurônios, converte-se então cada cromossomo para uma rede neural artificial, inicializa-se os pesos randomicamente, e treina-se a mesma, para calcular a aptidão de cada indivíduo, com isso buscam-se as melhores arquiteturas de RNAs. Entretanto, não é isso o que se acredita que ocorre na natureza, de acordo com o que foi descrito no parágrafo anterior e na seção 2.3.7. O método utiliza um Algoritmo Genético que gera Redes Neurais Artificiais, entretanto é como se os genes dos cromossomos codificassem o fenótipo (esquema de interconexão dos neurônios) ou a forma final do organismo, uma espécie de "mapa" que dá origem ao sistema nervoso deste seres, o que não é plausível biologicamente.

5.2.2 Método de Codificação Gramatical

Neste método, que é apresentado em detalhes no (ANEXO D, seção D.2), os Algoritmos Genéticos são utilizados para a geração de RNAs, entretanto os genes dos cromossomos codificam uma "receita" contendo regras de produção que serão transformados na especificação estrutural da rede.

KITANO (1990) discutiu que o esquema de codificação indireto é biologicamente mais plausível que o direto. O presente método já se preocupa com a questão de plausibilidade biológica, com o desenvolvimento do genótipo para o fenótipo sendo controlado por regras de produção, o método pode vagamente lembrar o desenvolvimento do embrião, controlado pelo DNA na natureza (PUJOL, 1999).

Em relação à plausibilidade biológica o método de KITANO (1990) é deficiente nos seguintes aspectos: A gramática utilizada na metodologia possui regras de produção que são aplicadas seqüencialmente, porém VAARIO (1993) considera que a motivação biológica para o uso dos Sistemas de Lindenmayer sensíveis ao contexto deve-se ao fato de que: as regras de produção são aplicadas em paralelo e, simultaneamente, repassam todas as letras em uma palavra dada, sendo destinadas a capturar múltiplas divisões celulares em organismos multicelulares ocorrendo ao mesmo tempo.

As regras de produção utilizadas por KITANO (1990), não exploram recursividade, que pode ser vista como divisões celulares (clivagens) que acontecem ao mesmo tempo na fase de desenvolvimento de organismos, onde ocorrem várias ao mesmo tempo. Além disso, as regras de produção localizam-se em posições fixas do cromossomo, bem como todas as posições do cromossomo codificam partes de uma regra de produção, que pode ser o predecessor ou sucessor. É como se todo o DNA codificasse proteínas, entretanto sabe-se que apenas 2% a 3% do mesmo é codificável, sendo a função restante ainda não muito clara (FARAH, 1997).

5.2.3 Método de Codificação Gramatical Sensível ao Contexto: Sistemas de Lindenmayer

BOERS e KUIPER (1992) foram mais fiéis às inspirações biológicas e utilizaram três metáforas biológicas. No trabalho deles, o Algoritmo Genético busca as melhores regras de produção, capazes de funcionarem conjuntamente e que gerem redes neurais artificiais capazes de aprender determinadas tarefas, a função custo utilizada foi o inverso do erro médio quadrático. As regras de produção são sensíveis ao contexto e implementam o crescimento de RNAs, as mesmas são aplicadas a um axioma por um certo número de vezes, e a lista resultante é transformada na especificação estrutural de uma rede neural. O erro resultante após o treinamento da mesma é então transformado na medida de aptidão, sendo então retornada para o AG. As melhores regras de produção serão mantidas, enquanto outras serão eliminadas. Os membros mantidos pela seleção sofrerão modificações em suas características através de mecanismos como mutações e cruzamentos gerando melhores descendentes para as próximas gerações.

O modelo de desenvolvimento artificial proposto por BOERS e KUIPER (1992) exploram como inspiração biológica a questão da modularidade do cérebro. As RNAs geradas pelo sistema evolucionário que eles criaram possuem uma estrutura modular, possibilitando obter redes satisfatórias que possuem um melhor desempenho que as obtidas por metodologias tradicionais de projeto.

A metodologia de projeto automático de RNAs utilizada nesta pesquisa baseou-se nos trabalhos de BOERS E KUIPER (1992), VARRIO (1993) e BOERS (1995). A principal motivação foi a questão da maior plausibilidade biológica defendida por estes pesquisadores.

5.3 METÁFORAS BIOLÓGICAS COMBINADAS

Nos capítulos 2, 3, 4 apresentou-se três sistemas inspirados biologicamente como se fossem peças de um quebra cabeça, que podem ser utilizadas isoladamente para a solução de vários tipos de problemas: De CAMPOS (2000), LEHRER (2000) , (SILVA et. al, 2000) . Nesta seção apresenta-se os mesmos, de forma integrada, como componentes de uma metodologia de projeto automático de RNAs.

5.3.1 Algoritmo Genético

Na presente metodologia utilizou-se um AG inspirado no processo evolucionário que procura criar/selecionar os seres vivos mais adaptados ao ambiente. O AG manipula um conjunto de regras de produção, sendo cada membro da população uma "string" binária consistindo de uma ou mais regras de produção de um o Sistema de Lindenmayer. Para determinar a aptidão de cada indivíduo, as regras de produção são extraídas dos cromossomos e o Sistema de Lindenmayer reescreve as mesmas aplicadas a um axioma. A cadeia de caracteres resultante é interpretada como uma rede a ser treinada por um algoritmo baseado no algoritmo de retropropagação, e que foi também adaptado para redes recorrentes de Jordan, o AG busca arquiteturas satisfatórias que aprendam tarefas específicas.

Com o objetivo de aproximar o Algoritmo Genético do modelo biológico de evolução, incorporou-se ao mesmo alguns aspectos observados na natureza e que foram discutidos na seção 2.3. O primeiro deles foi o esquema de codificação adotado, onde os genes dos cromossomos codificam "receitas" (regras de produção), seção 2.3.7. Acredita-se que no processo evolucionário, não são os organismos finais, mas as receitas que os construíram é que são combinadas para a Evolução. Aquelas que possuem a capacidade de

funcionarem conjuntamente na formação de um organismo têm maiores chances de sobreviver. Além de o processo evolucionário conduzir a mistura de receitas, algumas são executadas mais vezes do que outras aquelas que deram mais certo podem ser levadas a serem repetidas um maior número de vezes (ROISENBERG, 1998).

Como inspiração biológica da codificação cromossômica os genes de uma população de cromossomos do AG são inicializados aleatoriamente, cada um contendo um valor 0 ou 1. É como se o os cromossomos acondicionam-se o genoma (conjunto de genes da espécie humana) de cada indivíduo. Entretanto existe uma simplificação, pois se sabe que os seres humanos possuem 23 pares distintos de cromossomos (RIDLEY,2001). Entretanto em AG, costuma-se usar os termos indivíduos e cromossomos indistintamente (seção 2.5), sendo assim o número de variáveis que serão otimizadas correspondem ao número de cromossomos. Como será discutido na seção 5.4 a função custo utilizada nesta pesquisa apresenta quatro variáveis.

Na seção 2.3.5 mencionou-se que o DNA é dividido em vários elementos distintos chamados cromossomos, onde os genes correspondem a segmentos de DNA, que contém as seqüências de aminoácidos para fabricar as proteínas aqui vistas como regras de produção de um Sistema de Lindenmayer. Para a obtenção das mesmas utilizou-se algoritmo de extração de regras, cujos passos são apresentados a seguir:

1. O cromossomo é lido seis bits de cada vez.
2. Cada grupo de seis bits é convertido para uma vírgula, ou para um caractere do alfabeto, de acordo com a Tabela 5.1, o que resultará em uma cadeia de símbolos. A ordem de leitura da Tabela 5.1 é a seguinte: para determinar a seqüência de bits que codifica o caractere A, segue-se os seguintes passos: determina-se qual das quatro linhas da esquerda corresponde aos dois primeiros bits, no caso (00), então se escolhe a coluna de acordo com os dois bits do meio, que são (10) e finalmente escolhe-se uma linha da direita usando os dois últimos bits (00), ou seja, o caractere A é codificado por 001000.

3. Encontra-se todas as subcadeias que codificam uma regra de produção, adotando-se a representação utilizada na Tabela 5.2.
4. Rejeitar todas as subcadeias que não satisfaçam a representação utilizada na Tabela 5.2.
5. Repetir os passos1-5, não somente começando a ler a cadeia de bits somente a partir do primeiro bit do cromossomo, mas também em outras ordens. Nesta metodologia adotou-se as seguintes posições de leitura: seis da esquerda para a direita começando nas posições 1,2,3,4,5,6 e seis da direita para a esquerda, começando nas posições 512, 511, 510, 509, 508, 507. Cada cromossomo tem um tamanho de 512 bits. O algoritmo é repetido para todos os membros da população.

O Algoritmo apresentado anteriormente foi inspirado no processo de síntese de proteínas, seção 2.3.6. A Tabela 5.1 é a metáfora do código genético apresentado na Tabela 2.1. O passo 4 do algoritmo corresponde a remoção do íntrons do RNAm e da saída do mesmo para o citoplasma, onde estão presentes somente as porções que corresponde aos exons o que é mostrado na Figura 5.2.

Tabela 5.1 – Conversão de caracteres em binário

	00	01	10	11	
00	B	2	A	B	00
	D	,	,	C	01
	D	B	,	,	10
	B	A	,	B	11
01	D	D	B	2	00
	D	A	D	,	01
	C	B	D	C	10
	B	,	C	D	11
10	A	B	2	C	00
	B	B	B	B	01
	,	D	D	D	10
	A	C	B	A	11
11	B	B	C	B	00
	B	A	D	C	01
	B	2	D	C	10
	,	B	,	B	11

Tabela 5.2 – Representação para as Regras de Produção

Regras de Produção	**Representação utilizada**
α= A,B,C	A,B,C
A→AB|ϵ	AAB
B>C→DB|ϵ	BCDB
B<C→CD|ϵ	BCCD
D<D→D2	DDD2

Exemplificando-se o processo de leitura de um cromossomo, cuja representação simplificada é mostrada na Figura 5.1, considere que a mesma começa na posição 1 lendo-se o cromossomo de seis em seis bits, tem-se: 011010=D; 100101=B; 001101=C, 100110=D, o que resulta na seqüência de caracteres (D,B,C,D).

0	1	1	0	1	0	1	0	0	1	0	1	0	0	1	1	0	1	1	0	0	1	1	0

Figura 5.1 Representação simplificada de um cromossomo

Considerando que após a leitura de um cromossomo em doze posições específicas obteve-se a cadeia de caracteres, mostrada na Figura 5.2. Em azul aparecem as subcadeias válidas de acordo com a Tabela 5.2. O processo de remoção dos íntrons consiste em remover todos os caracteres representados por preto. Analisando a Figura 5.2 isso resultaria em: A,B,CAABBCDBBCCDDDD2 o que corresponde ao processamento do RNAm etapa 2 da Figura 2.4. O processo de tradução do RNAm em proteínas, etapa 3 da Figura 2.4, corresponde à tradução da cadeia A,B,CAABBCDBBCCDDDD2 em regras de produção, o que é realizado utilizando-se a Tabela 5.2. Sendo a segunda coluna da mesma (representação utilizada) convertida na primeira coluna (regras de produção). Somente são válidas as seguintes cadeias obtidas por transcrição: A,B,CAABBCDBBCCD e A,B,CAABBCDBBCCDDDD2 e que correspondem respectivamente, às seguintes regras de produção **(A→AB|ϵ, B>C→DB|ϵ, B<C→CD|ϵ)** e **(A→AB|ϵ, B>C→DB|ϵ, B<C→CD|ϵ, D<D→D2)**. O algoritmo de extração de regras reproduz a idéia básica da síntese protéica, mostrada em 2.3.6.

ABCDBCAD2BDCAA,B,CDCBCDADBACBDCAABABCDADCB2,ADBC2,,ADCBA2BCDBABCD
AB,2,,ADCB,ABCDA2,,2AABCCDABCDB2CDA2ABCD2,ABDCA2,ABCBDDD2ABCDA,2DACB,A
DCBDA,ABDCB2,,ACBD2,,ADBDC2,ABCD2,ACBDCACBDB,ABABACDCCD,2,ABDC,2,2ABC

Figura 5.2 Representação simplificada da extração de regras

A inspiração biológica que se adotou anteriormente considera basicamente que os cromossomos são estruturas celulares que contem o DNA, ou seja, os genes. Esses por sua vez carregam o código para que as células fabriquem as proteínas (regras de produção) que um organismo precisa para se desenvolver e viver.

5.3.2 Sistemas de Lindenmayer

Após à primeira etapa são geradas regras de produção de um Sistema de Lindenmayer . O mesmo pode ser formalmente definido como uma gramática **G={Σ, Π, α}**, onde: **Σ={A;B;C;D;2; ,}**, o axioma é **α=A,B,C** e as regras de produção utilizadas são de duplo contexto, contendo quatro elementos, o predecessor (**P**), o sucessor (**S**), o contexto à esquerda (**L**) e o contexto à direita (**R**) e tem o formato geral **L<P>R→S** onde **Π={A→AB|ϵ, B>C→DB|ϵ, B<C→CD|ϵ, D<D→D2}**. O significado das mesmas é mostrado na Tabela 5.3. A próxima etapa corresponde ao modelo de desenvolvimento de Redes Neurais, por meio do qual são geradas especificações estruturais de RNAs.

Tabela 5.3 – Regras de produção utilizadas no sistema, considerando o axioma α=A,B,C

A→AB|ϵ	A é substituído por AB ou por ϵ
B>C→DB|ϵ	Se **B** é sucedido por **C**, é substituído por **DB** ou ϵ (vazio).
B<C→CD|ϵ	Se **C** é precedido **B**, é substituído por **CD** ou ϵ (vazio).
D<D→D2	Se **D** é precedido por **D** é substituído por **D2**.

Como mencionado anteriormente o controle do desenvolvimento é realizado pelos genes, sendo que os mesmos participam passivamente do processo, comparados com as proteínas que eles codificam e que determinam o comportamento de cada célula. Um gene, que codifica uma proteína, constitui-se de uma porção de DNA que tem uma região codificante, que contém as instruções para a síntese protéica, além disso, o mesmo contém regiões de controle pelas quais o gene é ligado e desligado. A ativação e desativação dos genes corretos é a questão central do desenvolvimento dos organismos, sendo o mesmo dado pela multiplicação de células. Milhões de divisões celulares ocorrem ao longo da vida de um organismo da célula ovo até se formar um organismo adulto, ocorrem inúmeras mitoses (WOLPERT, 2000).

Como visto em 3.3 o desenvolvimento é composto de cinco fases, entretanto o modelo artificial que foi utilizado nesta pesquisa, apresenta simplificações no sentido de encontrar um nível de comportamento celular que explique o desenvolvimento, mas que não precise levar em consideração o comportamento detalhado de cada célula. Muitos dos processos de crescimento do cérebro ainda não são totalmente conhecidos, dado a heterogeneidade do mesmo, devido a grande diferenciação celular que existe nele em comparação com outros órgãos. Mas complexo ainda é saber a rota que leva os genes a uma estrutura complexa como o Sistema Nervoso.

O modelo artificial que é apresentado a seguir adota apenas a idéia básica do desenvolvimento biológico: Os genes controlando a síntese protéica (regras de produção), que quanto interagem com outras, modificam o comportamento celular, de acordo com os genes expressos em cada uma. O que leva a um processo de diferenciação celular responsável pela formação do cérebro.

O modelo proposto considera que a célula carrega uma cópia do código genético em forma de uma árvore. Cada nó representa uma instrução de um programa. O desenvolvimento começa com uma célula ancestral com conexões para célula de entrada e células de saída. As divisões celulares que ocorrem podem ser seqüenciais ou paralelas.

Em uma divisão seqüencial, denotado por "S", a célula pai divide-se em outras duas células (mitose), apresentada em 3.2, tal que o primeiro filho herda todas as conexões de entrada dos pais e o segundo todas as conexões de saída, sendo o primeiro conectado ao segundo, a divisão seqüencial ocorre na vertical.

Em uma divisão paralela, denotada por "P", a célula pai divide-se em duas células (mitose) que herdam todas as entradas e saídas dos pais. Em uma divisão paralela as duas células são colocadas lado a lado, sendo divisões na horizontal. O caractere "T" na árvore representa um símbolo terminal, e o "M" representa morte celular nesse caso o neurônio é eliminado bem como suas conexões que chegam e partem dele. A morte neuronial é comum no desenvolvimento do sistema nervoso de vertebrados, parece que inicialmente

são produzidos neurônios em excesso e que somente aqueles que estabelecem conexões apropriadas sobrevivem (WOLPERT, 2000).

As regras de produção do Sistema de Lindenmayer, mostradas na Tabela 5.3, são aplicadas ao axioma α são capazes de gerar redes diretas e qualquer rede recorrente de Jordan, com arquiteturas modulares. Com as restrições de não gerarem conexões diretas da entrada para a saída e realimentações partindo da camada intermediária.

- O processo de desenvolvimento começa com α, Figura 5.3 (a). A notação para α→ **A,B,C** , considera duas divisões celulares seqüências a primeira "**A,B**" Figura 5.3 (b) e a segunda "**B,C**", Figura 5.3 (c).
- Aplicando-se a regra **(A→AB|ϵ)** à palavra **(A,B,C)** repetidamente um número de vezes igual ao número de entradas da RNA desejada, inicialmente gera-se **(AB,B,C)**, Figura 5.3 (d) e na segunda **(ABB,B,C)**, Figura 5.3 (e). Considerando-se a segunda opção da regra **(A→ϵ)** a palavra obtida será **(BB,B,C)**.
- Utilizando-se a regra **(B>C→DB|ϵ)** aplicada à **(BB,B,C)** duas vezes. Na primeira gera-se **(BB,DB,C)**, Figura 5.3 (f) e na segunda **(BB,DDB,C)** , utilizando-se a segunda opção da regra obtêm-se **(BB,DD,C)**, Figura 5.3 (g).
- Considerando-se a regra **(B<C→CD|ϵ)** aplicada à **(BB,DD,C)** um número de vezes igual ao número de saídas da rede desejada, igual à por exemplo 1, resulta em **(BB,DD,CD)**, fazendo-se uso da segunda opção da regra gera-se a palavra **(BB,DD,D)**, Figura 5.3 (h).

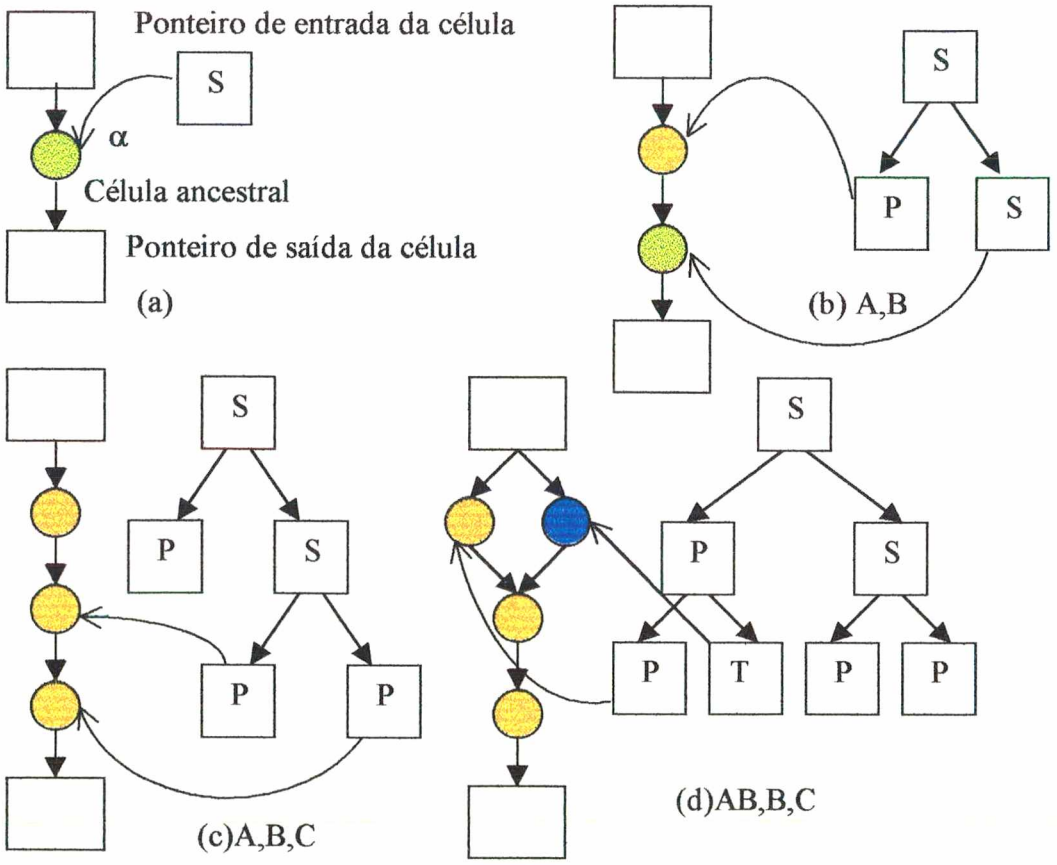
A Rede Neural gerada é mostrada na Figura 5.4 (a) a vírgula representa um nível acima e separa os módulos. A noção de contexto aqui apresentada é diferente daquela utilizada em 3.4.4. Aqui se considera o mesmo em relação à um nível acima ou abaixo analisando-se cada palavra gerada. Por exemplo, considerando-se a regra de produção **(B>C→DB|ϵ)** aplicada à **(BB,B,C)**, percebe-se que na Figura 5.3 (e) que C sucede B no terceiro nível, olhando-se de cima para baixo.

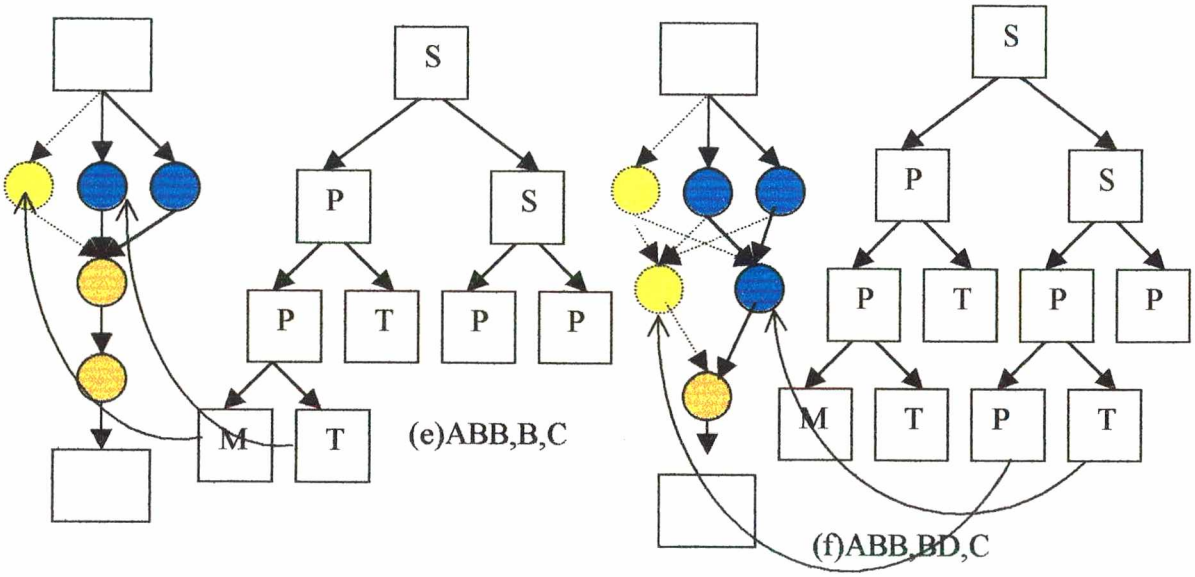
A Rede Neural Recorrente da Figura 5.4 (b) é obtida de maneira análoga à anterior, considerando ainda a regra de produção **D<D→D2** aplicada à **BB,DD,D**

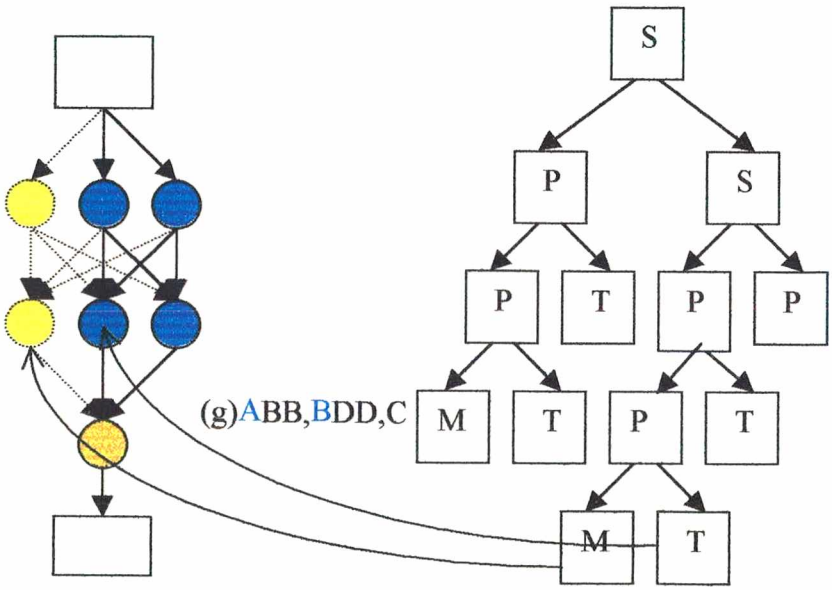
resulta em **BB,DD,D2.** O último **D2** significa recorrência em relação a dois níveis abaixo. A rede da Figura 5.4 (c) é obtida aplicando-se a regra **B>C→DB|ϵ** três vezes ao invés de duas como realizado na segunda etapa apresentada anteriormente.

5.3.3 Redes Neurais Artificias

As redes neurais que são geradas pelo Sistema de Lindenmayer foram biologicamente inspiradas no princípio de organização modular do cérebro, onde o mesmo pode ser visto como várias redes de neurônios em diversas regiões que podem realizar processamento paralelo para a execução de funções cognitivas específicas. O córtex cerebral é composto basicamente por módulos repetidos denominados colunas corticais com basicamente a mesma estrutura em seis camadas. Além disso, a estrutura em camadas do mesmo apresenta uma abordagem de processamento extremamente recorrente, esses aspectos foram tratados detalhadamente na seção 4.5.




S
P
S
P
T
P
P
M
T
(e)ABB,B,C
S
P
S
P
T
P
P
M
T
P
T
(f)ABB,BD,C


S
P
S
P
T
P
P
(g)ABB,BDD,C
M
T
P
T
M
T

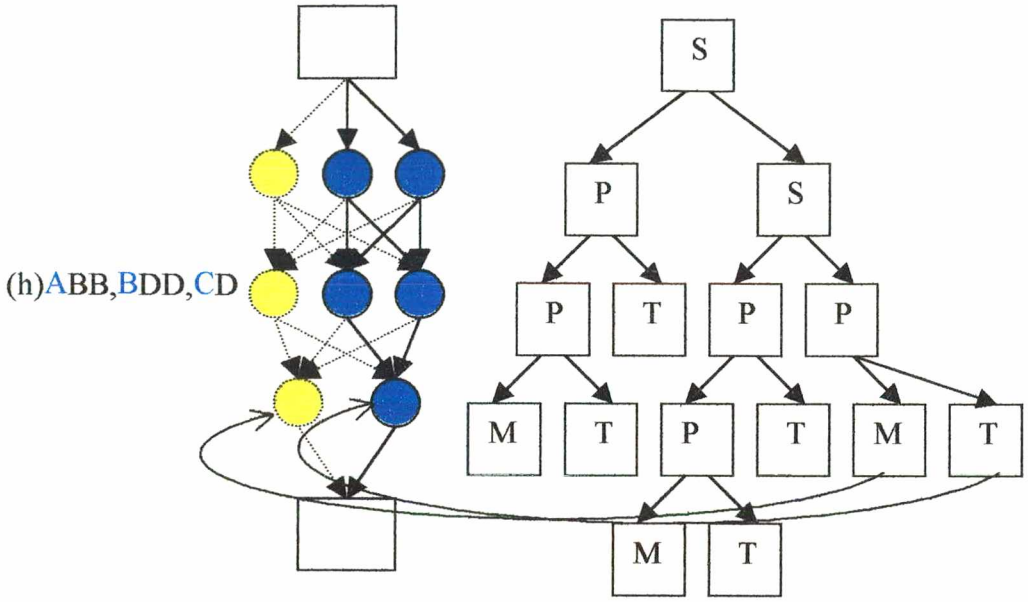


Figura 5.3 – Fases do Desenvolvimento Biológico(a,b,c,d,e,f,g,h)

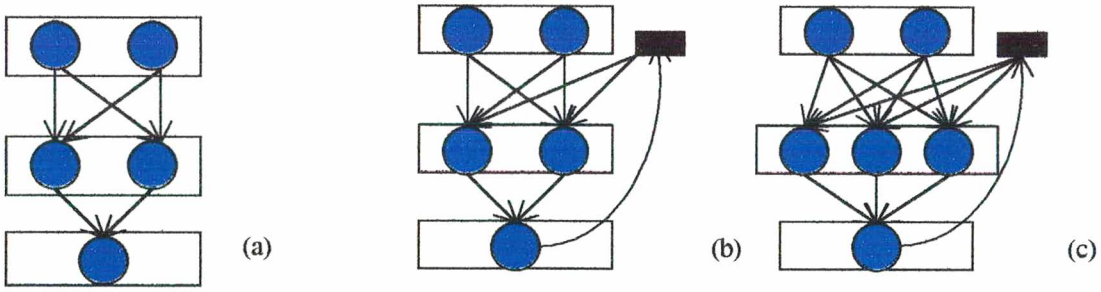


Figura 5.4 Redes Neurais geradas pelo Sistema de Lindenmayer

As Redes Neurais Artificias tentam, em princípio, modelar o sistema nervoso de seres vivos que interagem com o ambiente, realizando o controle das atividades dos mesmos. A modelagem matemática do modelo de rede direta é apresentada no ANEXO A e para rede recorrente de Jordan no ANEXO B, os dois foram baseados na regra delta generalizada. À medida que as especificações estruturais vão sendo geradas as mesmas são treinadas pelo simulador de Rede Neural. Após o treinamento procede-se a avaliação das regras de produção, com o objetivo de selecionar aquelas que tem capacidade de funcionarem conjuntamente.

5.4 Função Custo

Uma das etapas críticas de qualquer AG é a escolha da função custo. Para muitos problemas já existe uma função definida a priori. Entretanto, para o caso específico da otimização de arquiteturas de RNAs, a mesma teve de ser criada de acordo com a definição do problema. Pelos objetivos desta pesquisa, deseja-se obter redes satisfatórias (com um número mínimo de neurônios na camada intermediária, geradas com um número mínimo de regras de produção, com um número mínimo de recorrências e com um mínimo erro aceitável). A obtenção da função custo baseou-se na soma dos fatores que se deseja minimizar, entretanto considerou-se o inverso dos mesmos (com o objetivo de maximizá-los), por ser esta a ordem que se acredita que a natureza utiliza, o que é mostrado nas equações (5.1) e (5.2).

$FunçãoCusto = [A1/ERM + A2/NCIH + A3/NRP + A4/(REC+1)]$, Se ERM < = Valor Aceitável.

(5.1)

$FunçãoCusto = [A1/ERM + A2/NCIH + A3/NRP + A4/(REC+1)]*0.1$, Se ERM > Valor Aceitável.

(5.2)

ERM= Média dos erros dos padrões na saída da rede, **NCIH**=Número de Neurônios na Camada Intermediária, **NRP**=Número de Regras de Produção que gerem uma rede, **REC**=Número de Recorrências.

As regras de produção são capazes de gerar tanto redes diretas quanto recorrentes. Entretanto dependendo da classe de problema, determinada arquitetura pode não ser adequada para o mesmo. Com isso, as regras de produção que gerarem redes não satisfatórias não receberão uma boa avaliação e aquelas que gerarem redes satisfatórias receberão uma avaliação melhor. Com o objetivo de tratar esse tipo de situação, considerou-se na função custo o valor ERM que é comparado com um valor de erro aceitável, que garante uma boa capacidade de generalização da rede e considerando a precisão desejada para a classe de problema estudada. As equações (5.1) e (5.2) dividem

em dois conjuntos as regras de produção, aquelas que geram redes com valor de ERM menor ou igual ao valor aceitável e que são as desejadas, avaliadas pela equação (5.1), e outras cujos valores de ERM sejam superiores ao valor aceitável, e que não são desejadas para determinado problema, avaliadas pela equação 5.2. O fator número de recorrências foi incrementado de 1 nas duas equações, com o objetivo de generalizá-las para redes diretas e recorrentes. Percebe-se que regras de produção que gerem redes diretas apresentam REC=0, com isso esse termo não poderia ser considerado no denominador. No capítulo seguinte, apresentam-se maiores detalhes e discussões a respeito da função custo e dos resultados obtidos.

Após o treinamento de todas as redes, realiza-se a ordenação das mesmas o que é realizado em ordem crescente de acordo com o desempenho das mesmas. Na seleção utilizou-se o método da roleta ponderada, onde os indivíduos melhor avaliados terão maiores oportunidades de serem escolhidos em relação aos outros. A etapa seguinte corresponde a aplicação dos operadores genéticos. Os parâmetros taxa de cruzamento, taxa de mutação e número de pontos de corte, bem como os valores de A1,A2,A3 e A4 serão discutidos em pormenores no próximo capítulo. A nova população é obtida após a aplicação dos operadores Genéticos. Todas as etapas discutidas anteriormente desde a extração de regras do cromossomo até a obtenção de uma nova geração são repetidas por 100 gerações. O sistema evolucionário cuja metodologia foi apresentada anteriormente foi desenvolvido em linguagem C, pelo fato de a mesma ser amplamente utilizada no meio científico. Vale ressaltar que o mesmo não teve o objetivo de ser um produto de software. A plataforma utilizada nas simulações foi um Pentium III, 550 MHZ com 64MB RAM DIMM. No próximo capítulo apresentam-se os resultados de simulação para várias classes de problemas.

6 SIMULAÇÕES

Neste capítulo apresentam-se os resultados das simulações, utilizando-se a metodologia foi apresentada no capítulo anterior. As classes de problemas estudadas foram: XOR, Paridade, Problema das Lâmpadas e Botões e Linguagens de Tomita.

6.1 XOR

A função XOR é uma função lógica boolena de duas variáveis x e y, e que é freqüentemente utilizada na literatura como problema básico para modelos de Redes Neurais, (BOERS e KUIPER, 1992). As variáveis x e y são as entradas da rede e para cada ponto cartesiano (x,y) existe um valor de saída correspondente. A Tabela 6.1 ilustra a mesma. Os parâmetros da função custo utilizados na simulação foram A1=0.000001, A2=100, A3=1 e A4=100, o valor de erro aceitável foi de 10^{-2}. O número de épocas usadas foi de 20000 para 100 gerações, o tamanho do cromossomo como mencionado anteriormente é de 512 bits, a taxa de cruzamento de 0.6 e a de mutação de 0.001. Adotou-se cinco pontos de corte duplos fixos, nas posições 32, 235, 130, 256 e 432 do cromossomo. A Figura 6.1(a) mostra a rede satisfatória obtida para o problema e Tabela 6.2 os valores dos pesos das conexões obtidos para a mesma. Os testes realizados para o XOR são mostrados na Tabela 6.3.

Tabela 6.1 Função lógica XOR

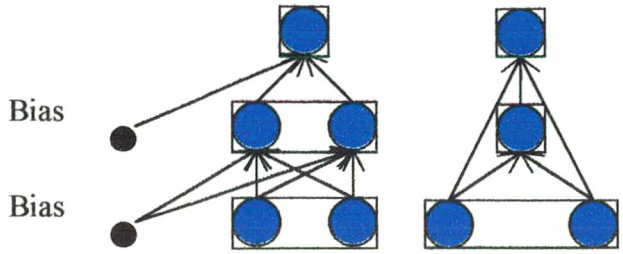
x	y	Saída
0	0	0
0	1	1
1	0	1
1	1	0

Tabela 6.2 Valores de pesos obtidos para o XOR

w1[0][1]=3.512511	w1[0][2]=9.161925	w1[1][1]=-8.043105
w1[1][2]=-6.129583	w1[2][1]=-8.065246	w1[2][2]=-6.132899
w2[0][0]=-6.884906	w2[1][0]=-14.182601	w2[2][0]=14.161345

Tabela 6.3 Operação da rede após o treinamento para o XOR

x	y	Saída Obtida	Saída Desejada	|Erro|
0	0	0.001516	0	0.001516
0	1	0.998461	1	0.001539
1	0	0.998459	1	0.001541
1	1	0.00188	0	0.00188



(a) (b)

Figura 6.1 – Arquiteturas de Redes Neurais Artificiais para a função lógica XOR

Existe uma outra solução para o XOR, com um neurônio na camada escondida e um na saída, mas com a propagação direta das entradas para a saída Figura 6.1(b), o que não é uma estrutura convencional. Como mencionado anteriormente na metodologia da proposta, a gramática utilizada nesta pesquisa não gera conexões diretas da entrada para a saída, sendo que para uma estrutura "feed-forward" completamente conectada são necessários dois neurônios na camada intermediária como ilustra a Figura 6.1(a).

6.2 A Paridade Como um Problema de Ordem Infinita

O problema da paridade consiste em apresentar na entrada da rede neural uma "string" de bits e obter na sua saída o valor da paridade do 'string" apresentado. O grafo correspondente a Máquina de Mealy, que implementa a paridade, é mostrado na Figura 6.2, onde a segunda componente de cada estado representa a saída, ou seja o bit que indica a paridade de uma string de bits.

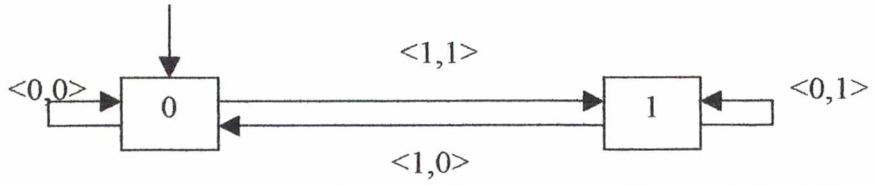


Figura 6.2- Grafo da correspondente Máquina de Mealy ME={∑,Q, δ, qo, F, Δ}=({0,1},{0,1},δ,0,{0,1},{0,1}}, que implementa o cálculo de paridade de uma string.

Analogamente, pode-se obter a Máquina de Moore que simula a Máquina de Mealy da Figura 6.2 o que é mostrado na Figura 6.3, onde a segunda componente de cada transição representa a saída, ou seja, o bit que indica a paridade de uma string de bits. A inserção do novo estado <0,ε> é necessário, pois, senão antes de começar o processamento,

no estado inicial, a máquina estaria escrevendo 0 na saída. Com a introdução deste estado, ela continua escrevendo, só que, desta vez, a cadeia vazia.

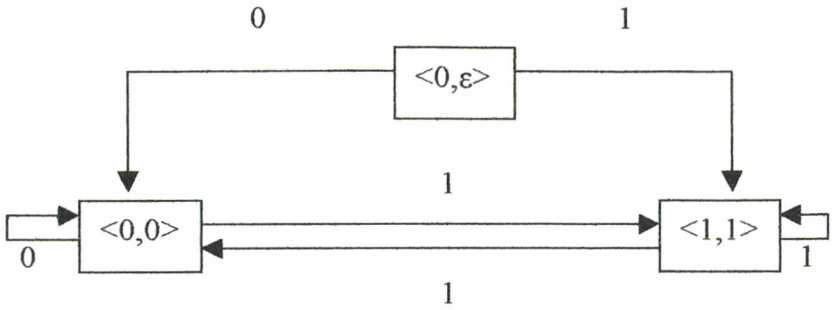


Figura 6.3- Grafo da correspondente Máquina de Moore MO={Σ,Q, δ, qo, F, Δ,δs}=({0,1},{0,1},δ,0,{0,1},{0,1}, δs }, que implementa o cálculo de paridade de uma string.

O saída 0 significa paridade par e a saída 1 corresponde à paridade ímpar. Assim se o autômato estava no estado 0 e o próximo bit do "string" for 1, a paridade passa a ser impar (saída 1), caso o próximo bit fosse 0, a paridade permaneceria sendo par (saída 0). Se o autômato estivesse no estado 1 e o próximo bit da "string" for 1, a paridade passa a ser par (saída 0), caso o próximo bit fosse 0, a paridade permaneceria sendo ímpar (saída 1).

Uma maneira de fazer com que uma rede neural fosse capaz de entender o "princípio de funcionamento" da questão da paridade seria ensinar para ela, por exemplo, o próprio autômato (Máquina de Mealy ou Moore) que determina a paridade, da Figura 6.2 ou Figura 6.3.

Uma maneira de treinar esta rede é utilizar o algoritmo baseado na retropropagação do erro para redes recorrentes, apresentado no (ANEXO B). O conjunto de treinamento é apresentado na Tabela 6.4.

Tabela 6.4- Conjunto de treinamento para o autômato que implementa o cálculo da paridade

Entrada u(k)	Estado x(k)	Estado x(k+1)
0	0	0
1	0	1
0	1	1
1	1	0

Ao analisar a Tabela 6.4, contendo o conjunto de treinamento, percebe-se claramente que o conjunto de treinamento representa a função lógica "OU-EXCLUSIVO".

Os parâmetros da função custo utilizados na simulação foram A1=0.000001, A2=100, A3=1 e A4=100, o valor do erro aceitável para o qual se garante uma boa capacidade de generalização no sentido da extrapolação foi de 10^{-2}. Os valores foram escolhidos por heurística. Adotou-se cinco pontos de corte duplos, nas posições 32, 235,130, 256 e 432 do cromossomo, o número de épocas usadas foi de 30000 para um total de 100 gerações, o tamanho do cromossomo foi de 512 bits, a taxa de cruzamento de 0.6 e a de mutação de 0.001. A rede satisfatória obtida é mostrada na Figura 6.4.

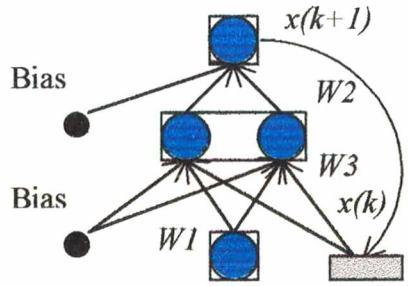


Figura 6.4 Arquitetura de Rede Neural Artificial obtida para a paridade

Os valores de pesos obtidos são mostrados na Tabela 6.5 onde W3 são os pesos das conexões recorrentes, W1 os pesos das conexões da camada de entrada para a camada intermediária e W2 os pesos das conexões da camada intermediária para a camada de saída, o índice 0 dos vetores de pesos da Tabela 6.5, representa o primeiro elemento de cada camada, sendo que para a de entrada e a intermediária o mesmo corresponde ao "Bias" e para a camada de saída ao primeiro neurônio da esquerda para a direita . A avaliação média e as melhores avaliações são mostradas na Figura 6.5. Percebe-se que a média da população caminhou em direção as melhores avaliações, permanecendo abaixo da mesma em torno de 30%.

Pelos resultados obtidos e mostrados na Figura 6.5, percebe-se que a função custo direcionou a busca do AG para uma rede satisfatória. A rede com um neurônio na camada intermediária não aprende o problema da paridade, pelos resultados obtidos a mesma teve uma avaliação bem abaixo da rede da Figura 6.4, o valor de ERM obtido para a Rede(1,2,1,2) foi de 1.321216E-05 e para a Rede(1,1,1,1)=0.56707, sendo o valor maior do que o valor aceitável estimado em 10^{-2}. O relatório final das simulações para o problema da paridade é mostrado no ANEXO E. Em azul aparecem as redes que obtiveram ERM maior

que 10^{-2} e em preto todas as que obtiveram valores de ERM menor ou igual ao valor aceitável. Os cálculos dos valores de aptidão obtidos por cada rede foram realizados pelas equações (5.1) e (5.2). Exemplificando-se para a arquitetura encontrada e mostrada na Figura 6.4, $FunçãoCusto = [0.000001/1.321216E - 5 + 100/2 + 1/4 + 100/(2+1)] =$ 83.659021, o mesmo processo é realizado para todas as outras redes.

Na população inicial predominaram regras de produção simples, capazes de gerar arquiteturas de redes diretas, o que ocasionou inicialmente uma aptidão média baixa. O processo evolucionário possibilitou que melhores regras de produção fossem obtidas, capazes de gerar redes recorrentes com isso a aptidão média da população aumentou também. Após 100 gerações as mesmas predominaram na população.

A segunda etapa teve por objetivo testar o aprendizado da rede apresentada na Figura 6.4 e comprovar a capacidade de extrapolação das redes recorrentes, os resultados são mostrados nas Tabelas 6.6, Tabela 6.7 e Tabela 6.8.

Tabela 6.5 – Pesos obtidos para a Rede Neural Artificial da Figura 6.4

W1[0][1]=-3.9739	W1[0][2]=-3.744871	W1[1][1]=-7.627204
W1[1][2]=7.220986	W2[0][0]=-6.726833	W2[1][0]=13.743343
W2[2][0]=13.678354	W3[0][1]=7.328455	W3[0][2]=-15.238744

Para a string de bits 0,1,0,1,1,0,1 apresentada na entrada da rede a saída desejada (ou seqüência de estados) seria 0,1,1,0,1,1,0, o que equivale a paridade par, pois a última saída é 0. As saídas obtidas e os respectivos erros são mostrados abaixo na Tabela 6.6.

Tabela 6.6 – Saídas obtidas e desejadas para a entrada 0,1,0,1,1,0,1

Saídas Desejadas	Saídas Obtidas	|Erro|
0	0.022707	0.022707
1	0.987723	0.012277
1	0.987723	0.012277
0	0.016073	0.016073
1	0.984752	0.015248
1	0.987743	0.012257
0	0.016074	0.016074

O segundo experimento considerou como entrada a string 0,0,1,0,1,0,1 a saída desejada é 0,0,1,1,0,0,1, o que representa paridade ímpar, pois a última saída é 1. O que é mostrado na Tabela 6.7.

O terceiro experimento, apresentou-se a seguinte string de bits na entrada da rede 0,0,1,0,1,0,1,1,0,1,0 a saída desejada é 0,0,1,1,0,0,1,0,0,1,1, ou seja uma string que tem paridade ímpar, os resultados são mostrados na Tabela 6.8.

Pelos resultados obtidos pode-se perceber que a rede é capaz de calcular a paridade de um número ilimitado de bits de entrada. O que não seria possível utilizando-se uma rede direta, pois se a entrada da rede tivesse n bits, uma rede com n neurônios na camada de entrada, n na camada escondida, 1 neurônio da camada de saída e uma amostra representativa de 2^n possíveis vetores de entrada como conjunto de treinamento, seria possível resolver o problema . Esta rede deveria essencialmente aprender a contar o número de bits no conjunto de treinamento. Após a rede treinada a rede seria capaz de interpolar o exemplo apresentado, chegando a acertar o valor da paridade para todos os 2^n valores possíveis de serem questionados.Entretanto ela não é capaz de resolver qual a paridade para um vetor de entrada maior que n bits. O que é possível com a utilização de redes recorrentes que possuem uma excelente capacidade de generalizar no sentido de extrapolação (ROISENBERG, 1998).

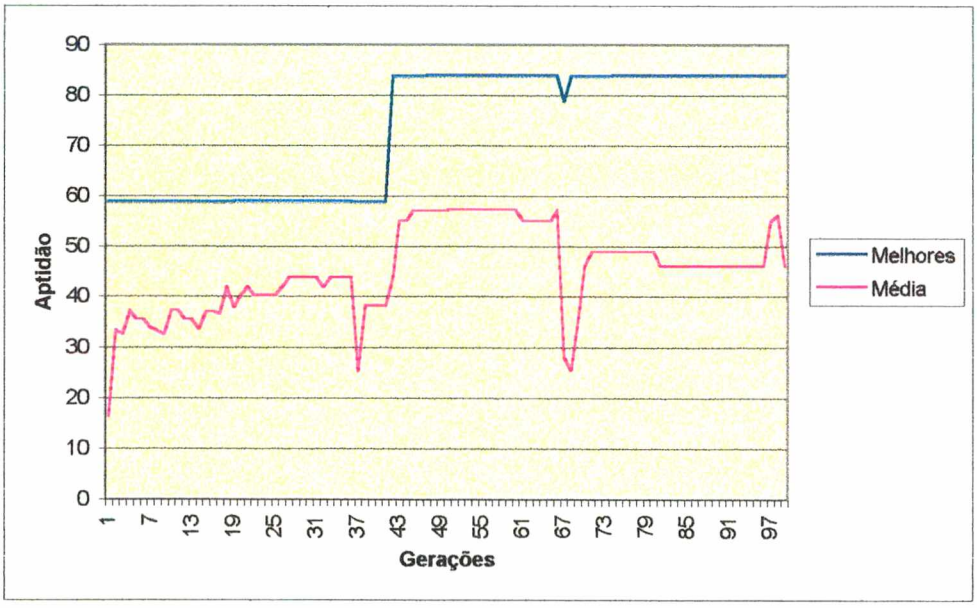


Figura 6.5 – Melhores avaliações e avaliação média para o problema da paridade

Tabela 6.7 – Saídas obtidas e desejadas para a entrada 0,0,1,0,1,0,1

Saídas Desejadas	Saídas Obtidas	|Erro|
0	0.022707	0.022707
0	0.002135	0.002135
1	0.984533	0.015467
1	0.987728	0.012272
0	0.016074	0.016074
0	0.02168	0.02168
1	0.984519	0.015481

Tabela 6.8 – Saídas obtidas e desejadas para a entrada 0,0,1,0,1,0,1,1,0,1,0

Saídas Desejadas	Saídas Obtidas	|Erro|
0	0.013282	0.013282
0	0.01286	0.01286
1	0.990632	0.09368
1	0.992627	0.007373
0	0.009866	0.009866
0	0.012957	0.012957
1	0.99063	0.00937
0	0.009821	0.009821
0	0.012958	0.012958
1	0.99063	0.00937
1	0.992627	0.007373

6.3 Lâmpadas e Botões

Suponha-se um agente bastante simples cuja entrada sensorial é capaz de perceber a condição de duas lâmpadas. Se a primeira estiver acesa, o primeiro sensor é acionado, caso a segunda lâmpada esteja acesa, o segundo é acionado. O agente possui também um atuador capaz de agir sobre dois botões. Se o valor do atuador for 0.5, o primeiro botão é acionado, se o valor do atuador for 1, o segundo botão é acionado. Uma ou nenhuma das lâmpadas podem estar acesas em determinado instante, mas não as duas simultaneamente. O comportamento desejado é que o agente pressione o botão correspondente a lâmpada que estiver ligada, se eventualmente uma delas estiver. Caso nenhuma das lâmpadas esteja acesa, o agente deve pressionar o botão correspondente à lâmpada que estava acesa mais recentemente.

O AEF mostrado na Figura 6.6 descreve formalmente esse comportamento. Note-se que neste caso, o próprio valor do estado corresponde de maneira unívoca aos valores de saída do AEF. O conjunto de treinamento utilizado para treinar a rede é apresentado na Tabela 6.9. Aqui o valor 0.5 equivale a lâmpada desligada e o valor 1 corresponde a lâmpada ligada. O estado inicial é indeterminado e corresponde ao valor 0.

Os parâmetros da função custo utilizados na simulação foram A1=0.1, A2=100, A3=1 e A4=100, o valor do erro aceitável foi de 0.0999. O número de épocas usadas foi de 30000 para 100 gerações, o tamanho do cromossomo é de 512 bits, a taxa de cruzamento de 0.6 e a de mutação de 0.001. Adotou-se cinco pontos de corte duplos fixos, nas posições 32, 235, 130, 256 e 432 do cromossomo. A Figura 6.8 mostra a rede satisfatória obtida para o problema e Tabela 6.11 os valores dos pesos das conexões. Os testes realizados para o problema são mostrados na Tabela 6.10.

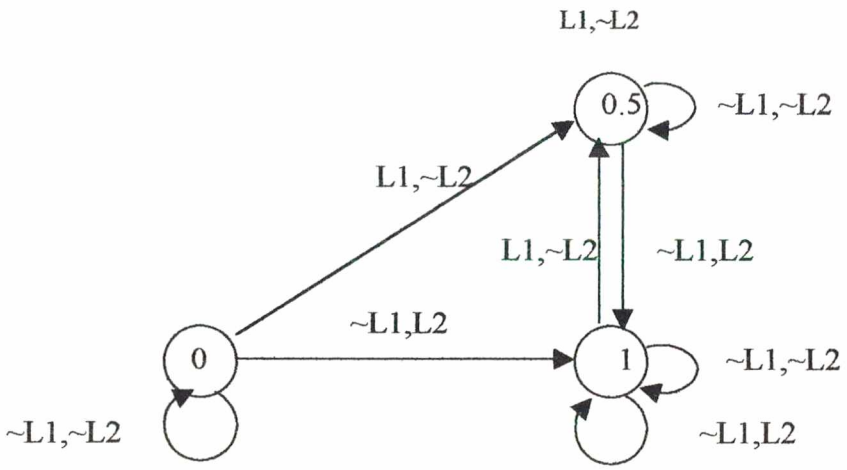


Figura 6.6 – Autômato Finito, (~L1 – lâmpada 1 desligada, ~L2 – lâmpada 2 desligada, L1 – lâmpada 1 ligada, L2 lâmpada 2 ligada), que implementa o agente para o problema das lâmpadas e botões.

A Figura 6.7 mostra as melhores avaliações e a avaliação média para o problema de lâmpadas e botões, percebe-se que a média da população caminha em direção dos melhores resultados permanecendo, nas gerações finais, abaixo da mesma cerca de 20%. Redes com número de neurônio superior a dois na camada intermediária aprendem melhor o problema simulado, entretanto como a tarefa não exige um nível de precisão muito grande, a rede

com dois neurônios na camada intermediária foi adequada para o aprendizado e testes realizados para o problema das lâmpadas e botões.

Tabela 6.9 - Conjunto de treinamento para o problema lâmpadas e botões

Entradas u(k)		Estado x(k)	Estado x(k+1)
Lâmpada 1	Lâmpada 2		
0.5	0.5	0	0
1	0.5	0	0.5
1	0.5	0.5	0.5
0.5	0.5	0.5	0.5
0.5	1	0.5	1
0.5	1	1	1
0.5	0.5	1	1
1	0.5	1	0.5
1	0.5	0.5	0.5
0.5	1	0.5	1
0.5	0.5	1	1
0.5	1	1	1
1	0.5	1	0.5
0.5	0.5	0.5	0.5
0.5	0.5	0.5	0
0.5-Lâmpada desligada,1 -Lâmpada desligada		0.5 – Botão 1 acionado,1 - Botão 2 acionado 0 - nenhum botão acionado	

Tabela 6.10- Operação da rede para o problema lâmpadas e botões

x(k+1) Obtido	x(k+1) Desejado	|Erro|
5.545238E-6	0	5.545238E-6
0.495596	0.5	0.004404
0.50207	0.5	0.00207
0.427402	0.5	0.072598
0.999997	1	0.000003
0.999998	1	0.000002
0.999998	1	0.000002
0.50641	0.5	0.00641
0.50207	0.5	0.0207
0.999998	1	0.000002
0.999998	1	0.000002
0.999998	1	0.000002
0.50641	0.5	0.00641
0.459515	0.5	0.040485
0.168672	0	0.168672

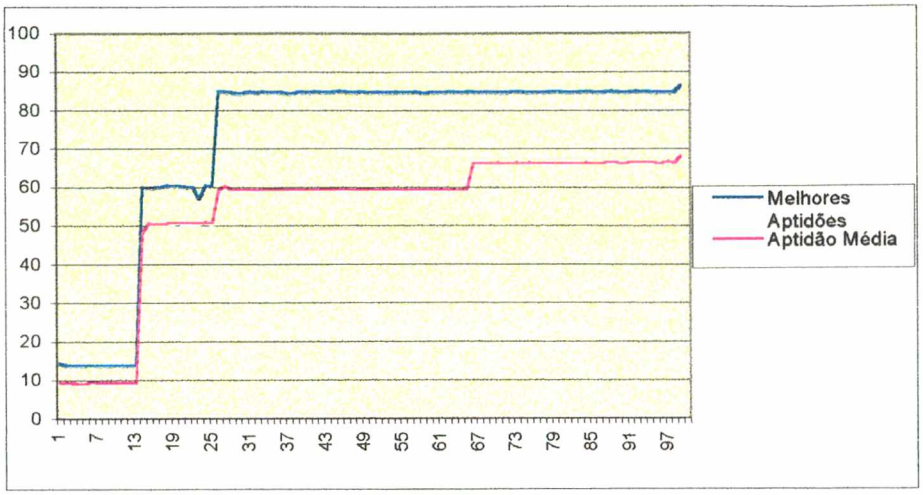


Figura 6.7 – Melhores avaliações e médias para o problema Lâmpadas e Botões.

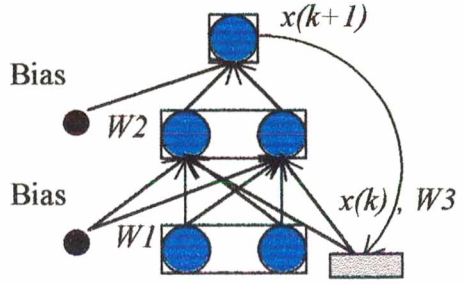


Figura 6.8 Arquitetura de Rede Neural Artificial obtida para o problema lâmpadas e botões

Tabela 6.11 – Pesos obtidos para a RNA da Figura 6.8

W1[0][1]=-9.261272	W1[0][2]= 21.670431
W1[1][1]=-32.309912	W1[1][2=-25.861577
W1[2][1]= 19.358637	W1[2][2]=-3.910781
W2[0][0]= 0.007756	W2[1][0]= 13.277248
W2[2][0]=-12.124556	W3[0][1]= 25.253488 , W3[0][2]=-43.800223

6.4 Linguagens de Tomita

Um tópico de pesquisa bastante interessante e que tem despertado o interesse de pesquisadores na área conexionista é desenvolver modelos de RNA com a capacidade de induzir autômatos finitos baseados apenas em exemplos positivos e negativos de "strings" ou cadeias, que pertencem e que não pertencem à linguagem reconhecida pelo mesmo.

Uma coleção explícita de exemplos positivos e negativos, mostrados na Tabela 6.13, e que colocam dificuldades específicas para a indução da linguagem pretendida é apresentada em (TOMITA, 1982). A caracterização das linguagens propostas é apresentada na Tabela 6.12. Percebe-se que os conjuntos de treinamentos não são balanceados, ou seja, são incompletos e variam enormemente na sua capacidade de definir a linguagem pretendida.

Os valores adotados para as entradas da Rede Neural Artificial forma (0.5,1,0) respectivamente para os valores (0,1, ε) e (1,0) para as saídas correspondendo a exemplos (+,-). A seguir apresentam-se os resultados obtidos por simulação para as linguagens de Tomita, bem como os parâmetros utilizados e testes da capacidade de extrapolação.

Os parâmetros da função custo utilizados na simulação para a linguagem 1 de Tomita e Linguagem 2 de Tomita, foram A1=0.0001, A2=100, A3=1 e A4=100. O valor do erro aceitável foi de 10^{-2} . O número de épocas usadas foi de 30000 para 100 gerações, a taxa de cruzamento de 0.6 e a de mutação de 0.001. Adotou-se cinco pontos de corte duplos fixos, nas posições 32, 235, 130, 256 e 432 do cromossomo.

A Figura 6.9 mostra a rede satisfatória obtida para a Linguagem 1 de Tomita e a Figura 6.12 para a Linguagem 2. As Tabelas 6.14 e 6.16 mostram os pesos obtidos. Finalmente nas Tabelas 6.15 e 6.17, mostram-se os testes de operação das redes para as duas linguagens.

Tabela 6.12 – Caracterização de algumas linguagens propostas por Tomita

Linguagem	Descrição
1	1*
2	(10)*
3	Nenhuma cadeia com número ímpar de 0´s é permitido depois de uma cadeia com números ímpar de 1´s.
4	Não mais do que dois 0´s em uma cadeia
5	(número de 1´s-número de 0´s) mod 3=0
6	0*1*0*1*

Tabela 6.13 – Exemplos positivos e negativos de algumas linguagens investigadas por Tomita

Linguagem	**Exemplos Positivos**	**Exemplos Negativos**
1	ε,1,11,111,1111,11111,111111, 1111111,11111111	0,10,01,00,011,110,000, 11111110,10111111
2	ε,10,1010,101010, 10101010,10101010101010	1,0,11,00,101,100 10001010,10110,110101010
3	ε,1,0,01,11,00,100,110,111,000, 100100,110000011100001, 111101100010011100	10,101,010,1010,1011,111010, 1001000,1111100, 0111001101,11011100110
4	ε,1,0,10,01,00,100100, 001111110100,0100100100,11100,010	000,11000,0001,000000000,00000 ,0000,11111000011, 1101010000010111,1010010001
5	ε,10,01,1100,111,000000 0111101111,100100100	1,0,11,00,101,011,11001,1111, 00000000,0101111,10111101111, 1001001001
6	ε,1,0,10,01,11111,000,00110011, 0101,0000100001111,00100, 011111011111,00	1010,00110011000,0101010101, 1011010,10101,010100,101001, 100100110101

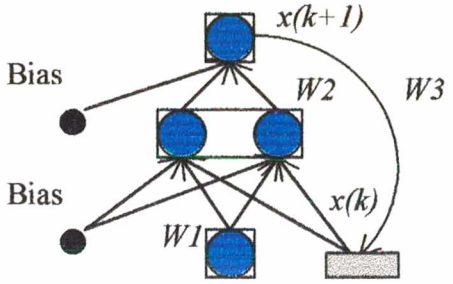


Figura 6.9 Arquitetura de Rede Neural Artificial obtida para a linguagem 1 de Tomita.

Tabela 6.14 – Pesos obtidos para a RNA da Figura 6.9

W1[0][1]= 3.597285	W1[0][2]= 3.108498	W1[1][1]=-10.213931
W1[1][2]=-3.736722	W2[0][0]=-17.382507	W2[1][0]=-21.773534
W2[2][0]= 33.999864	W3[0][1]= 4.240194	W3[0][2]= 38.629033

Tabela 6.15 - Operação da rede para o problema linguagem 1 de Tomita

x(k+1) Obtido	x(k+1) Desejado	|Erro|
0.993624	1	0.006376
1	1	0
1	1	0
1	1	0
0.00578	0	0.00578
0.026471	0	0.026471
0.978195	1	0.021805
1	1	0
0.00578	0	0.00578
0.026471	0	0.026471
0.005959	0	0.005959

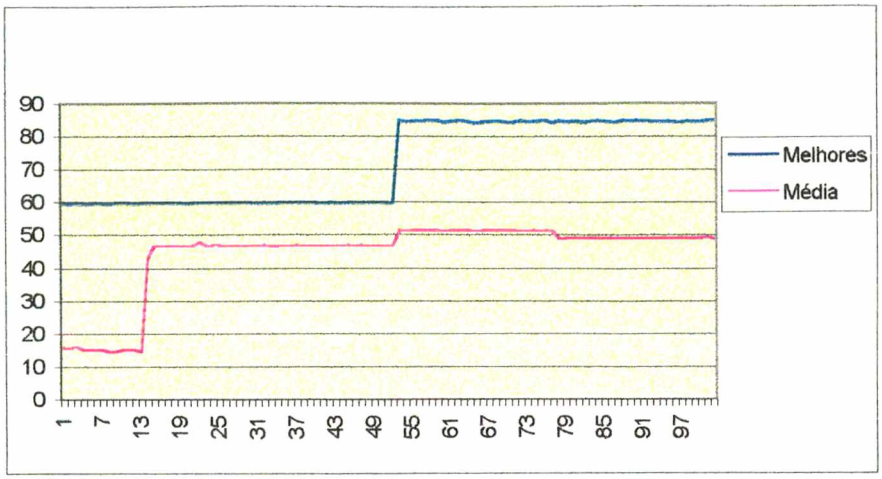


Figura 6.10- Melhores avaliações e médias para o problema da paridade para a linguagem 1 de Tomita

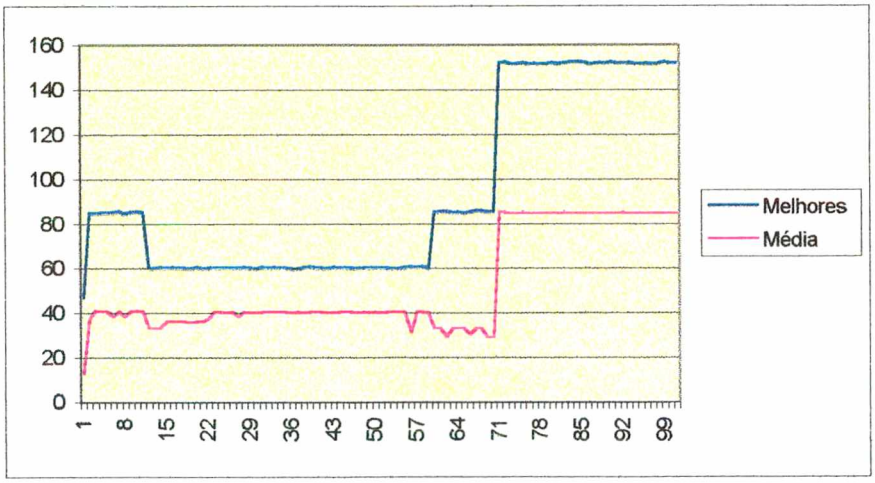


Figura 6.11 - Melhores avaliações e médias para o problema da paridade para a linguagem 2 de Tomita

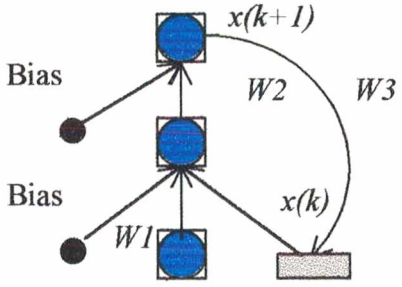


Figura 6.12 Arquitetura de Rede Neural Artificial obtida para a linguagem 2 de Tomita

Tabela 6.16 – Pesos obtidos para a Rede Neural Artificial da Figura 6.12

W1[0][1]=-1.124018	W1[1][1]=5.540576	W2[0][0]= 4.127925
W1[1][0]=-6.500178	W2[0][1]= 6.583464	

Tabela 6.17 - Operação da rede para o problema linguagem 2 de Tomita

x(k+1) Obtido	x(k+1) Desejado	|Erro|
0.760134	1	0.239866
0.092165	0	0.092165
0.873081	1	0.126919
0.092139	0	0.092139
0.873109	1	0.126891
0.092139	0	0.092139
0.873109	1	0.126891
0.092139	0	0.092139
0.873109	1	0.126891
0.098142	0	0.098142
0.096041	0	0.096041
0.868848	1	0.131152
0.096201	0	0.096201
0.86867	1	0.13133
0.09214	0	0.09214
0.873109	1	0.126891

No próximo capítulo apresentam-se as conclusões finais deste trabalho e recomendações de melhorias à metodologia e pontos em aberto, possibilitando assim o desenvolvimento de trabalhos futuros.

7 CONSIDERAÇÕES FINAIS

7.1 Conclusões

Este capítulo tem por objetivo apresentar as principais conclusões obtidas, bem como algumas melhorias que poderiam ser acrescentadas à metodologia proposta nesta pesquisa e que de certa forma ampliariam o conhecimento aqui gerado.

Nesta pesquisa utilizou-se uma metodologia de projeto de RNAs inspirada na Evolução Biológica, em Modelos de Desenvolvimento Biológico e no princípio de modularidade do cérebro. Pelos resultados obtidos por simulação percebe-se que para todas as classes de problemas estudadas, a metodologia possibilitou a obtenção de redes satisfatórias, sendo que para cada problema heurísticas próprias foram necessárias, de acordo com o que foi mostrado no capítulo 6. O que nem sempre garante a mínima arquitetura, pois podem existir outras redes com a mesma topologia, número de neurônios por camada, número de recorrências e geradas como mesmo conjunto de regras de produção, porém com configurações de pesos diferentes e que resolvem o problema a contento.

Em relação à atuação dos operadores genéticos, chegou-se a conclusão que os mesmos representam papel fundamental no processo de busca de regras de produção que gerem arquiteturas otimizadas de RNAs. No caso específico da metodologia apresentada no capítulo 5, podem acontecer mutações cujo efeito seja nulo, ou seja, que não tem importância. Observou-se que algumas regras de produção permanecem inalteradas durante gerações posteriores, subitamente são produzidas novas formas que diferem nitidamente dos seus pais, sendo a mutação o mecanismo que permite a variabilidade das mesmas, o que se realiza por saltos. Na natureza os mecanismos de mutações são raros, durante gerações sucessivas, a grande maioria dos indivíduos não se modifica, a forma mutante aparecem subitamente, sem que sejam esperadas. Vale ressaltar que uma mutação pode representar tanto uma progressão quanto uma regressão (JACOB, 1983).

Pelos resultados obtidos por simulação percebe-se que a metodologia aumenta o grau de paralelismo do AG, pois se adota várias posições de leitura no cromossomo. O método reduz os custos de projeto e aumenta o desempenho das RNAs obtidas. Além disso,

o mesmo é superior em relação ao método de codificação direta (ANEXO D.1) e ao de codificação gramatical (ANEXO D.2) em relação a: plausibilidade biológica, arquiteturas de redes geradas e pelas classes de problemas simuladas. O tempo médio das simulações foi de 48 horas, utilizando-se um Pentium III, 550 MHZ com 64MB RAM DIMM.

A metodologia de projeto automático de RNAs utilizada nesta pesquisa constitui uma importante ferramenta para a obtenção de arquiteturas de redes neurais otimizadas. A utilização do AG permite que se trabalhe com uma população de RNAs ao invés de uma única rede, permitindo que se selecione uma arquitetura satisfatória e que solucione determinada tarefa, com certo grau de precisão. Percebe-se que a média da população de soluções candidatas se situa em torno de 70% para os problema da paridade, lâmpadas e botões e Linguagem 1 de Tomita, a 60% para a Linguagem 2 de Tomita. A metodologia apresentada é superior aos métodos convencionais de projetos, onde uma arquitetura não otimizada acarreta uma carga computacional que pode limitar a aplicação prática da Rede Neural ou inviabilizar a sua implementação, por exemplo, em "hardware".

7.2 Recomendações para trabalhos futuros

Em trabalhos futuros sugere-se melhorar a gramática proposta no capítulo 5, seção 5.3.2, possibilitando assim a obtenção de redes com arquiteturas não totalmente interconectadas, aumentando assim o grau de inspiração biológica em relação a interconexão dos neurônios do cérebro. Com isso será possível a obtenção de arquiteturas modulares não totalmente conectadas. Para isso, deve-se adotar uma gramática mais flexível, o que certamente dificultará o processo de decodificação e obtenção de regras de produção válidas.

Outra sugestão para futuras pesquisas seria comparar o aprendizado baseado no "backpropagation" para redes recorrentes, apresentado no ANEXO-B, com outras técnicas que utilizam computação evolucionária para evolução dos pesos das conexões, analisando-se o desempenho dos mesmos. Outro tópico interessante é o desenvolvimento de um modelo de rede totalmente conectada considerando recorrências da camada de saída e da camada intermediária.

Em trabalhos futuros sugere-se também investigar a utilização de cromossomos de menor comprimento, sendo que na presente pesquisa já se obteve uma melhoria em relação ao trabalho de (BOERS e KUIPER, 1992), que utilizaram cromossomos de 1024 bits e nesta pesquisa diminui-se o tamanho do mesmo para 512 bits.

Com objetivo de melhorar o desempenho do AG proposto na presente pesquisa, sugere-se em futuras dissertações ou teses empregar conceitos de Teoria de Jogos Evolucionários, buscando estratégias que possam fornecer uma eficiência adicional ao AG na busca de soluções satisfatórias para o problema de otimização de arquiteturas de RNAs. Neste caso, os operadores tradicionais do AG, seleção, recombinação e mutação, não ficariam sujeitos somente a critérios aleatórios para realizar a exploração da superfície adaptativa.

Usar outra tabela de translação, a distribuição de caracteres da Tabela 5.1 foi realizada aleatoriamente. Sugere-se investigar experimentos feitos com diferentes distribuições, variando o número de cada caractere. Uma comparação mais exaustiva com outros métodos que utilizam Sistemas de Lindenmayer e verificar a convergência do AG e o desempenho das redes obtidas.

Outro estudo que poderia ser realizado seria adotar pontos de corte aleatórios nos cromossomos durante a operação de cruzamento e comparar os resultados obtidos com os obtidos nesta pesquisa, que adotou pontos fixos.

REFERÊNCIAS BIBLIOGRÁFICAS

ABELSON, H. e DISESSA, A.A. (1982).Turtle geometry. M.I.T.Press, Cambridge.

BALDWIN, J.M. (1896), A new factor in evolution American Naturalist, nº 30, pp. 441-451, 1896.

BARRETO, Jorge Muniz. (1999). Inteligência Artificial no limiar do século XXI. Florianópolis.

BARRETO, Jorge Muniz (1990). Neural Network Learning : A new programing paradigm. In ACM International Conference : Trends and Directions in Expret Systems, Florida, USA, Oct. 1990. ACM.

BEASLEY, David; BULL, David R.; MARTIN, Ralph R. (1993). "An Overview of Genetic Algorithms: Part 2, Research Topics." In University Computing 15(4), 170-181.

BINDER, J. (1995). Functional magnetic resonance image of langauge cortex. International Jornal of Imaging System an technology, 6:280-288.

BOERS .E.J.W. and KUIPER.H. (1992) "Biological metaphors and the desing of modular artificial neural networks".Msc. dissertation, Leiden University, the Netherlands, Departments of Computer Science and Experimental an Theoretical Psychology.

BOERS, E.J.W (1995). Using L-Systems as Graph Grammar: G2L-Systems,Technical Report 95-30.

BORISYUK, R.; BORISYUK, G. e KAZANOVICH, Y. (2001). Temporal structure of neural activity and modelling of information processing in the brain. In. S. Wermter, J.Austin and D. Willshaw, editors, Emergent Neural Computational Architectures based on Neuroscience. Spring-Verlag, Heidelberg, Germany.

CHOMSKY, N. (1956). Three models for the description of language. IRE trans. On Information Theory, 2(3):113-124.

De CAMPOS, L.M.L; De CAMPOS, G.A.L (2000). "Aplicação de Redes Neurais em um Problema de Seleção de Tecnologias para a Manufatura Integrada por Computador, I Simpósio Catarinense de Computação / UNIVALI, Santa Catarina-SC.

DARWIN, Charles Robert. (1859). On the origin of species by mean of natural selection, Londres:Murray.

DAWKINS, RICHARD. (1998). A Escalada do monte improvável, tradução Suzana Sturlini Couto. – São Paulo : Companhia das Letras.

De AZEVEDO, Fernando Mendes (1993). Contribution to the study of Neural Networks in Dynamical Expert Systems. PhD thesis, Institut d'Informatique, FUNDP, Namur, Belgium.

DODEL, S. ; HERRMANN, J.M. e GEISEL T. (2001) .Stimulus-independent data analysis for fRMI. In. S. Wermter, J.Austin and D. Willshaw, editors, Emergent Neural Computational Architectures based on Neuroscience. Spring-Verlag, Heidelberg, Germany.

DIAS, JOÃO DA SILVA. (1998). Sensibilidade Paramétrica como Guia para Treinamento híbrido de Redes Neurais. Tese de Doutorado. Universidade Federal de Santa Catarina, Departamento de Engenharia e Elétrica, UFSC, Florianópolis, Santa Catarina.

DOZA, K. (1999).What are the computations of the cerebellum, the basal ganglia and the cerebral cortex? Neural Networks, 12(7-8):961-974.

FARAH, Solange Bento. (1997). DNA Segredos e Mistérios. SAVIER, São Paulo.

FOGEL, David B. (1995). Evolutionary Computation – Toward a new philosophy of machine inteligence, Piscataway, NJ: IEEE Press.

FREEMAN, J.A. ; SKAPURA, D.M. (1991). Neural networks:algorithms, applications and programming techniques, Addilson-Wesley, Reading.

FRIEDERICI, A. (2000). The developmental cognitive neuroscience of language : A new research domain. Brain and Language, 71:65-68.

GAZZANIGA, M.; IVRY R. e MANGUN, G. (1998). Cognitive Neuroscience: The Biology of the Mind. W.W.Norton Company Ltd, New York.

GOLDEBERG, David E. (1989). Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley Publishing Company, Inc.

GORNI, A. A. (1994). Redes Neurais Artificiais – Uma Abordagem Revolucionária em Inteligência Artificial. *Microsistemas*, 13(333):14-25.

GUIGON, E.;GRAMDGUILLAUME, P; OTTO, I.; BOUTKHIL, L. e BURNORD, Y. (1994). Neural network models of cortical functions based on the computational properties of cerebral córtex, Journal of Physiology (Paris), 88:291-308.

GUYTON, ARTHUR C. ; HALL, JOHN E. (1998). FISIOLOGIA HUMANA E MECANISMOS DAS DOENÇAS. Guanabara Koogan, Rio de Janeiro.

HARP, S.A.; SAMAD, T. and GUHA, A. (1990). Desigining application-specific neural networks using the genetic algorithm. In Advances in Nural Information Processing Systems 2 (D. S. Touretzky, ed.), pp. 447-454, Morgan Kaufman, San Mateo, CA.

HAUSEN, P., RIEBESELL, H. (1991) The Early Development of Xenopus Laevis. Berlin : Springer – Verlag.

HOLLAND, J.H. (1975). Adaptation in Natural and Artificial Systems, USA:Univ.Michigan Press.

JACOB, F RANÇOIS. (1983). A Lógica da Vida. Uma história da hereditariedade. Edições Graal, Rio de Janeiro.

KITANO.H.(1990) "Designing neural networks using genetic algorithms with graph generation system, " Complex Systems, vol.4, no. 4, pp.461-476.

KOZA, John R. (1990). "Genetically breending populations of computer programs to solve problems in artificial intelligence.". In Proceedings of the Second International Conference on Tools for AI, 819-827. Los Alamitos, CA: IEEE Computer Society Press.

KOVÁCS, Zsolt László. (1996). REDES NEURAIS ARTIFICIAIS: Fundamentos e Aplicações: um texto básico – 2. Ed. – São Paulo: Edição Acadêmica.

KOVÁCS, Zsolt László. (1997). O Cérebro e a sua mente: uma introdução à neurociência computacional. São Paulo. Edição Acadêmica.

KNOBLAUCH, A. ; PALM, G. (2001) .Spiking associative memory and scene segmentation by synchronization of cortical activity. In. S. Wermter, J.Austin and D. Willshaw, editors, Emergent Neural Computational Architectures based on Neuroscience. Spring-Verlag, Heidelberg, Germany.

LEHRER, Cristiano. (2000). Operador de Seleção para Algoritmos Genéticos baseados no Jogo HAWK-DOVE. Dissertação de Mestrado, Universidade Federal de Santa Catarina, Departamento de Estatística e Informática, UFSC, Florianópolis, Santa Catarina.

LINDENMAYER, A. (1968) ."Mathematical models for celular interaction in development, parts I and II". In Journal of theorical biology, 18, 280-315.

MCCLELLAND, J.L. ; RUMELHART, D.E. e PDP Group (1986). Parallel Distributed Processing, volume 2, Psycological and Biological Models. The MIT Press, Cambridge, Massachusetts.

MCCLELLAND, J.L. ; RUMELHART, D.E. e PDP Group (1986). Parallel Distributed Processing, volume 1, Psycological and Biological Models. The MIT Press, Cambridge, Massachusetts.

MENEZES, PAULO BLAUTH (1998). Linguagens Formais e Autômatos, 2ª edição, Porto Alegre.

MICKLOS, D.A.e FREYER, G. A . (1990). DNA Science : A First Course in Recombinant DNA Technology. Cold Spring Harbor Press, New York, p. 477.

MILLER .G.F., TODD P.M., and HEDGE. S. U.(1989), "Designing neural networks using genetic algorithms, " in Proc. Of the third Int'l Conf. On Genetic Algorithms and Their Applications (J. D. Schaffer, ed.), pp 379-384, Morgan Kaufman, San Mateo, CA.

MINSKY, M.L. e PAPERT, S. A . (1969). Perceptrons : an introduction to computational geometry. The MIT Press, Cambridge, Massachusetts.

MITCHELL, Melaine. (1996). An Introduction to Genetic Algorithms. Cambridge: MIT Press.

MITCHISON, G.J e MICHAEL Wilcox. (1972). Rules governing cell division in Anabaena, Nature, 239:110-111.

MURPHY, Michael P.; O'NEILL, Luke A.J. (1997)." O que é vida?" 50 anos depois. Especulações sobre o Futuro da Biologia. Unesp/Cambridge.São Paulo.

MURRAY, D. (1994). Tuning neural network with genetic algorithm. AI Expert, 9:27-31.

PRUSINKIEWICZ, P. e HANAN, J. (1990); Lindenmayer systems, fractals and plants. Springer-Verlag, New York.

PUJOL, J. C. F. (1999). Evolution of artificial neural networks using a two-dimensional representation. Doctorate thesis, School of Computer Science the University of Birmingham.

REILLY, R. (2001). Collaborative cell assemblies: Building blocks of cortical computation. In. S. Wermter, J.Austin and D. Willshaw, editors, Emergent Neural Computational Architectures based on Neuroscience. Spring-Verlag, Heidelberg, Germany.

RIDLEY, Matt. (2001). Genoma; tradução de Ryta Vinagre. Rio de Janeiro: Record.

ROISENBERG, Mauro; BARRETO, Jorge Muniz, e de AZEVEDO, Fernando Mendes. (1996).Biological Inspirations in Neural Networks Implementations of Autonomous Agents. In D.L. Bores and C.A.A.Kaestner, editors, Advances in artificial inteligence: proceedings/13th Brazilian Symposium on Artificial Inteligence, number 1159 in lecture notes in computer science; Lectures notes in artificial intelligence, p.211-220 Spring-Verlag, Berlin, Oct. 1996.

ROISENBERG, M.. Emergência da Inteligência em Agentes Autônomos através de Modelos Inspirados na Natureza", Tese de Doutorado, Universidade Federal de Santa Catarina, Florianópolis, 1998.

ROSE, Michael Robertson .(2000). O Espectro de Darwin: a teoria da evolução e suas implicações no mundo moderno / tradução Vera Ribeiro; revisão técnica, Francisco M. Salzano. – Rio de Janeiro: Jorge Zahar Ed.

SILVA, Flávio de Almeida e, ROISENBERG, Mauro, BARRETO, Jorge M. *"Redes Neurais Hierárquicas para Implementação de Comportamentos em Agentes Autônomos"*. XIII CONGRESSO BRASILEIRO DE AUTOMÁTICA - CBA2000, Florianópolis. 2000.

SIMPSON, P.K. (1990). Artificial Neural Systems – foudantions, paradigms, applications and implementations. Neural Networks : Research and Applications. Pergamon Press, Elmsford, NY.

SPITZER, M. (1999) .The Mind Within the Net: Models of Learning, Thinking and Acting. MIT Press, Cambridge, MA.

TANOMARU. J. (1995). Motivação, fundamentos e aplicações de algoritmos genéticos. II Congresso Brasileiro de Redes Neurais e III Escola de Redes Neurais. Curitiba, Brasil.

TAYLOR, J. (2001). Images of the mind:Brain images and neural networks. In. S. Wermter, J.Austin and D. Willshaw, editors, Emergent Neural Computational Architectures based on Neuroscience. Spring-Verlag, Heidelberg, Germany.

TOMITA, M. (1982). Dynamic construction of finite automata from examples using hill-climbing. In Proceedings of the Fourth Annual Conference of the Cognitive Science Society, p.105-108, Ann Arbor, Michigan.

VAARIO. J. (1993)."An Emergent Modeling Method for Artificial Neural Networks." (1993) Doctor dissertation of engineering, Aeronautic an Astronautica Engineering Course, The University of Tokyo, Japan.

VICO, F.J. ; SANDOVAL, F. (1991). Use of Genetic algorithms in neural networks definition. In A.Prieto, editor.Proc.of the International Workshop on Artificial Neural Networks. IWANN'91, p.196-203, Granada, Spain, 17-19 Sep 1991. Spring-Verlag.

WOLPERT, Lewis; et. al. (2000) Princípios de Biologia do Desenvolvimento. Trad. Henrique Bunselmen Ferreira. Porto Alegre: Artes Médicas Sul.

YAO, X. and SHI, Y. (1995). A preliminary study on designing artificial neural networks using co-evolution, in Proc. Of the IEEE Singapore Intl Conf on Intelligent Control and Instrumentation, (Singapore), pp. 149-154, IEEE Singapore Section, June 1995.

ANEXO A-ALGORITMO RETROPROPAGAÇÃO

Os passos do algoritmo de retropropagação são descritos abaixo:

1. Seja A o número de unidades da camada de entrada, conforme determinado pelo comprimento dos vetores de entrada de treinamento. Seja C o número de unidades da camada de saída. Seja B o número de unidades da camada oculta. As camadas de entrada e oculta, têm, cada uma, uma unidade extra usada como limite; portanto as unidades dessas camadas, às vezes, serão indexadas pelos intervalos (0,....,A) e (0,.....,B). Denota-se os níveis de ativação das unidades da camada de entrada por xj, da camada oculta por hj e da camada de saída por oj. Os pesos que conectam a camada de entrada a camada oculta são denotados por $w1_{ij}$, onde i indexa as unidades de entrada e o j, as unidades ocultas. Da mesma forma, os pesos que conectam a camada oculta à camada de saída são denotados por $w2_{ij}$, com i indexando as unidades ocultas e j as unidades de saída.

2. Inicializar os pesos da rede. Cada peso deve ser ajustado aleatoriamente para um número entre –0.1 e 0.1.

$w1_{ij}$=aleatório(-0.1;0.1), $w2_{ij}$=aleatório(-0.1;0.1) para todo i=0,.....,A; j=1,.....,B;

para todo i=0,.....,B; j=0,.....,C.

3. Inicializar as ativações das unidades limite. Seus valores nunca mudam.

$$x_o=1.0 \text{ e } h_o=1.0$$

4. Escolher um par de padrão de entrada-saída. Supondo que o vetor de entrada seja x_i e o vetor de saída seja y_j. Atribuem-se níveis de ativação às unidades de entrada.

5.Propagar a ativação das unidades da camada de entrada para as unidades da camada oculta, usando a função de ativação.

$$h_j = \frac{1}{1+e^{-\sum_{i=0}^{A} w1ij.xi}} \text{ , para todo j=1,...,B} \qquad \text{(A.1)}$$

6.Propaga-se as ativações das unidades da camada oculta para as unidades da camada de saída, usando a função de ativação.

$$o_j = \frac{1}{1+e^{-\sum_{i=0}^{B} w2ij.xi}} \text{ para todo j=1,.....,C} \qquad \text{(A.2)}$$

7. Computar os erros das unidades da camada de saída, denotados por $\partial 2_j$. Os erros baseiam-se na saída real da rede (o_j) e na saída (y_j).

$$\delta 2_j = o_j(1-o_j)(y_j - o_j) \text{ para todo j=1,.....,C} \quad \text{(A.3)}$$

8. Computar os erros das unidades da camada oculta, denotados por δ1j.

$$\delta 1_j = h_j(1-h_j)\sum_{1=0}^{C} \delta 2_i.w2_{ij} \text{ para todo j=1,.....,B} \quad \text{(A.4)}$$

9. Ajuste dos pesos entre a camada oculta e a camada de saída. O coeficiente de aprendizagem é denotado por η, sua função é a mesma de na aprendizagem por perceptrons. $\Delta w2_{ij}(t+1) = \Delta w2_{ij}(t) + \eta\delta 2_j.h_i$ para todo i=0,.....,B; j=1,.....,C (A.5)

10. Ajuste os pesos entre a camada de entrada e a camada oculta

$$\Delta w1_{ij}(t+1) = \Delta w1_{ij}(t) + \eta\delta 1_j.x_i \text{ para todo i=0,.....,A; j=1,.....,B} \quad \text{(A.6)}$$

11. Vá para a etapa 4 e repita. Quando todos os pares entrada-saída tiverem sido apresentados à rede, uma época terá sido completada. Repita as etapas de 4 a 10 para tantas épocas quantas forem desejadas.

ATUALIZAÇÃO DOS PESOS DA CAMADA INTERMEDIÁRIA E CAMADA DE SAÍDA PARA REDES DIRETAS

Esta seção contém a dedução do algoritmo "backpropagation" (regra delta generalizada RDG). A explicação aqui apresentada é mínima, o leitor interessado pode encontrar um estudo mais aprofundado em ou FREEMAN e SKAPURA (1991). A dedução será apresentada para uma rede de três camadas e pode ser facilmente generalizada para rede de mais de três camadas.

O erro para um nó de saída durante o treinamento é δm=(ynm-om), com oj a saída obtida e yj a saída desejada para o nó j de saída. O erro que é minimizado pela regra delta generalizada é dado pela equação (A.7).

$$\varepsilon_n(t) = 1/2\sum_{m=0}^{C} \delta_m{}^2 \quad \text{(A.7)}$$

com C o número de nós de saída. Deseja-se minimizar ε, logo o melhor caminho é ir no sentido contrário ao gradiente, já que este aponta para o sentido crescente da função.

$$\varepsilon_n(t) = 1/2\sum_{m=0}^{C}(y_{nm} - o_m)^2 = 1/2\sum_{m=0}^{C}\varepsilon_{nm} \tag{A.8}$$

onde: n=número de padrões que serão apresentados durante o treinamento.As atualizações dos pesos são dadas pelas equações (A.9) e (A.10).

$$\Delta w1_{jk}(t+1) = \Delta w1_{jk}(t) - \eta\frac{\partial\varepsilon_n}{\partial w1(t)_{jk}} \tag{A.9}$$

$$\Delta w2_{km}(t+1) = \Delta w2_{km}(t) - \eta\frac{\partial\varepsilon_n}{\partial w2(t)_{km}} \tag{A.10}$$

$$\text{onde } S_{jk}(t) = \frac{-\partial\varepsilon_n}{\partial w1_{jk}(t)},\ r_{km}(t) = \frac{-\partial\varepsilon_n}{\partial w2_{km}(t)} \tag{A.11}$$

Deseja-se determinar as variações dos pesos relativos aos nós de saída, calcula-se o gradiente negativo $-\nabla E$ com relação aos pesos $w2_{km}$. Considerando cada componente de ∇E, separadamente obtém-se a equação (A.12).

$$\frac{\partial\varepsilon_n(t)}{\partial w2_{km}(t)} = \frac{1}{2}\frac{\partial\varepsilon_{nm}}{\partial o_m}\frac{\partial o_m}{\partial w2_{km}} \tag{A.12}$$

$$\frac{\partial\varepsilon_{nm}}{\partial o_m} = -2(y_{nm} - o_m) \tag{A.13}$$

$$\frac{\partial o_m}{\partial w2_{km}} = \frac{\partial}{\partial w2_{km}}\left[\frac{1}{1+e^{-\sum_{K=0}^{B} w2_{km}.hk}}\right] \tag{A.14}$$

$$, \ u = 1+e^{-\sum_{K=0}^{B} w2_{km}.hk} \quad \text{(A.15)}, \quad o_m = \frac{1}{u}, \quad u\text{-}1 = e^{-\sum_{K=0}^{B} w2_{km}.hk} \tag{A.16}$$

$$\frac{\partial o_m}{\partial w2_{km}} = \frac{\partial o_m}{\partial_u}.\frac{\partial_u}{\partial w2_{km}} \tag{A.17}$$

$$\frac{\partial o_m}{\partial u} = -\frac{1}{u^2},\ \frac{\partial u}{\partial w2_{km}} = (u-1)(-h_k) \tag{A.18),(A .19}$$

$$\frac{\partial o_m}{\partial w2_{km}} = h_k.\frac{(u-1)}{u^2},\ \frac{\partial o_m}{\partial w2_{km}} = \frac{(1/o_m - 1)}{1/o_m^{\,2}} = (1-o_m)o_m h_k \tag{A.20}$$

$$\frac{\partial\varepsilon_n}{\partial w2_{km}(t)} = \frac{1}{2}.-2(y_{nm} - o_m)(1-o_m)o_m.h_k \tag{A.21}$$

$$\frac{\partial \varepsilon_n}{\partial w2_{km}(t)} = -(y_{nm} - o_m)(1-o_m).o_m.h_k \tag{A.22}$$

O cálculo para os pesos relativos a camada intermediária é mostrado a seguir:

$$\frac{\partial \varepsilon_n}{\partial w1_{jk}(t)} = \frac{1}{2}\sum_{m=0}^{C} \frac{\partial \varepsilon_{nm}}{\partial o_m}\frac{\partial o_m}{\partial h_k}\frac{\partial h_k}{\partial w1_{jk}} \tag{A.23}$$

$$\frac{\partial \varepsilon_{nm}}{\partial o_m} = -2(y_{nm} - o_m) \tag{A.24}$$

$$\frac{\partial o_m}{\partial h_k} = \frac{\partial}{\partial h_k}\left[\frac{1}{1+e^{-\sum_{K=0}^{B} w2_{km}.hk}}\right] \text{(A.25)} \qquad \text{u=}1+e^{-\sum_{K=0}^{B} w2_{km}.hk} \tag{A.26}$$

$$o_m = \frac{1}{u}\text{ , u-1=}e^{-\sum_{K=0}^{B} w2_{km}.hk}\text{ , }\frac{\partial o_m}{\partial h_k} = \frac{\partial o_m}{\partial u}.\frac{\partial u}{\partial h_k} \tag{A.27}$$

$$\frac{\partial o_m}{\partial u} = -\frac{1}{u^2}\text{ , }\frac{\partial u}{\partial h_k} = (u-1)(-w2_{km}) \tag{A.28}$$

$$\frac{\partial o_m}{\partial h_k} = \frac{-1}{u^2}.\frac{(u-1)(-w2_{km})}{} = \frac{(u-1)(w2_{km})}{u^2} \tag{A.29}$$

$$\frac{\partial o_m}{\partial h_k} = \frac{(1/o_m - 1)(w2_{km})}{1/o_m{}^2} = (1-o_m).o_m.w2_{km} \tag{A.30}$$

$$\frac{\partial h_k}{\partial w1_{jk}} = \frac{\partial}{\partial w1_{jk}}\left[\frac{1}{1+e^{-\sum_{j=0}^{A} w1jk.xnj}}\right] \tag{A.31}$$

$$\text{u=}1+e^{-\sum_{j=0}^{A} w1jk.xnj}\text{ , u-1=}e^{-\sum_{j=0}^{A} w1jk.xnj} \tag{A.32}$$

$$h_k = \frac{1}{u}\text{ , }\frac{\partial h_k}{\partial w1_{jk}} = \frac{\partial h_k}{\partial u}.\frac{\partial u}{\partial w1_{jk}} \tag{A.33}$$

$$\frac{\partial h_k}{\partial u} = -\frac{1}{u^2}\text{ , }\frac{\partial u}{\partial w1_{jk}} = -(u-1)(x_n) \tag{A.34}$$

$$\frac{\partial h_k}{\partial w1_{jk}} = \frac{(u-1)(x_{nj})}{u^2} = \frac{(\frac{1}{h_k}-1)}{\frac{1}{h_k^2}}.x_n \tag{A.35}$$

$$\frac{\partial h_k}{\partial w1_{jk}} = (1-h_k).h_k.x_{nj}i \tag{A.36}$$

$$\frac{\partial \varepsilon_m}{\partial w1_{jk}(t)} = \frac{1}{2}\sum_{m=0}^{C}\frac{\partial \varepsilon_{nm}}{\partial \theta_m}\frac{\partial \theta_m}{\partial h_k}\frac{\partial h_k}{\partial w1_{jk}} \tag{A.37}$$

$$\frac{\partial \varepsilon_n}{\partial w1_{jk}(t)} = \frac{1}{2}\sum_{m=0}^{C} -2(y_{nm}-o_m)(1-o_m).o_m.w2_{km}.(1-h_k)h_k.x_{nj} \tag{A.38}$$

Cálculo de $S_{JK}(t)$ e $r_{KM}(t)$

$$S_{jk}(t) = \frac{-\partial \varepsilon_n}{\partial w1_{jk}(t)} = (1-h_k)h_k.x_{nj}\sum_{m=0}^{C}(y_{nm}-o_m)(1-o_m).o_m.w2_{km} \tag{A.39}$$

$$r_{km}(t) = \frac{-\partial \varepsilon_n}{\partial w2_{km}(t)} = (y_{nm}-o_m)(1-o_m).o_m.h_k \tag{A.40}$$

As equações (A.41) e (A.42) mostram as fórmulas par atualização dos pesos, da camada de saída e da camada intermediária, respectivamente.

$$\Delta w2_{km}(t+1) = \Delta w2_{km}(t) + \eta\ [(y_{nm}-o_m)(1-o_m).o_m].h_k \tag{A.41}$$

$$\Delta w1_{jk}(t+1) = \Delta w1_{jk}(t) + \eta\ [(1-h_k)h_k.\sum_{m=0}^{C}(y_{nm}-o_m)(1-o_m).o_m.w2_{km}]x_{nj} \tag{A.42}$$

ANEXO B – REDE RECORRENTE DE JORDAN

As saídas na camada intermediária são agora dadas pela equação (B.1).

$$hk(t)= \frac{1}{1 + e^{-\sum_{m=0}^{C} om\ (t-1).w3(t)_{mk} + \sum_{j=0}^{A} Xnj\ (t).w1(t)_{jk}}}, \tag{B.1}$$

para k>0 e t>0, h0=1, que é o valor do bias. O termo om(t-1) refere-se a cada saída que é realimentada e C é o número de neurônios da camada de saída, para t=0 o seu valor é om(t-1) =0. A atualização dos pesos é dada por:

$$\Delta w3_{mk}(t+1) = \Delta w3_{mk}(t) - \eta \frac{\partial \varepsilon_n}{\partial w3(t)_{mk}} \tag{B.2}$$

a atualização dos pesos da camada intermediaria e da camada de saída são dadas pelas equações (B.1) e (B.2). A atualização dos pesos da realimentação é calculada pela equação (B.3).

$$\frac{\partial \varepsilon_n}{\partial w3_{mk}(t)} = \frac{1}{2} \sum_{m=0}^{C} \frac{\partial \varepsilon_{nm}}{\partial o_m} \frac{\partial o_m}{\partial h_k} \frac{\partial h_k}{\partial w3_{mk}} \tag{B.3}$$

$$\frac{\partial \varepsilon_{nm}}{\partial o_m} = \frac{\partial \varepsilon_{nm}}{\partial o_m} = -2(y_{nm}(t) - o_m(t)) \tag{B.4}$$

$$\frac{\partial o_m}{\partial h_k} = (1 - o_m).o_m.w2_{km} \tag{B.5}$$

$$\frac{\partial h_k}{\partial w3_{mk}} = \frac{\partial}{\partial w3_{mk}} \left[\frac{1}{1 + e^{-\sum_{m=0}^{C} om\,(t-1).w3(t)_{mk} + \sum_{j=0}^{A} x_{nj}\,(t).w1(t)_{jk}}} \right] \tag{B.6}$$

$$u_k(t) = 1 + e^{-\sum_{m=0}^{C} om\ (t-1).w3(t)_{mk} + \sum_{j=0}^{A} X_{nj}\ (t).w1(t)_{jk}} \tag{B.7}$$

$$u_k(t) - 1 = e^{-\sum_{m=0}^{C} om(t-1).w3(t)_{mk} + \sum_{j=0}^{A} Xnj\,(t).w1(t)_{jk}} \tag{B.8}$$

$$h_k(t) = \frac{1}{uk(t)}, \frac{\partial h_k(t)}{\partial u(t)} = -\frac{1}{uk\ (t)^2}, \tag{B.9}$$

$$\frac{\partial h_k(t)}{\partial w3(t)_{mk}} = \frac{\partial h_k(t)}{\partial u_k(t)} \frac{\partial u_k(t)}{\partial w3(t)_{mk}} \tag{B.10}$$

$$\frac{\partial h_k(t)}{\partial w3(t)_{mk}} = (u_k(t) - 1)(-om(t-1)) \tag{B.11}$$

$$\frac{\partial h_k(t)}{\partial w3_{mk}(t)} = -\frac{1}{u_k(t)^2}(u_k(t)-1)(om(t-1)) \quad \text{(B.12)}$$

$$\frac{\partial h_k(t)}{\partial w3mk(t)} = \frac{(\frac{1}{h_k(t)}-1)}{(\frac{1}{h_k(t)})^2}.om\ (t-1) \quad \text{(B.13)}$$

$$\frac{\partial h_k(t)}{\partial w3_{mk}(t)} = (1-h_k(t))h_k(t).om\ (t-1)$$

(B.14)

$$\frac{\partial \varepsilon_n}{\partial w3_{mk}(t)} = (1-h_k(t))h_k(t)\sum_{m=0}^{C}(o_m(t)-y_{nm}(t))(1-o_m(t)).o_m(t).om(t-1).w2_{km}(t) \quad \text{(B.15)}$$

Logo

$$\Delta w3_{mk}(t+1) = \Delta w3_{mk}(t) + \eta(1-h_k(t))h_k(t)\sum_{m=0}^{C}(y_{nm}(t)-o_m(t))(1-o_m(t))o_m(t)om(t-1)w2_{km}(t) \quad \text{(B.16)}$$

ANEXO C - AUTÔMATO FINITO

Um autômato finito pode ser visto como uma máquina composta, basicamente, de três partes:

a) ***Fita*** - Dispositivo de entrada que contém a informação a ser processada.

b) ***Unidade de Controle*** – Reflete o estado corrente da máquina, possui uma unidade de leitura (cabeça da fita), a qual acessa uma fita cada vez e movimenta-se exclusivamente para a direita.

c)***Programa ou Função de transição*** – Função que comanda as leituras e define o estado da máquina.

Definição 1: Um autômato finito determinístico ou simplesmente autômato finito é uma quíntupla $M=(\Sigma,Q,\delta,qo,F)$ onde: Σ é o alfabeto de símbolos de entrada ; **Q** conjunto de estados possíveis do autômato o qual é finito; δ função programa ou função de transição; $\delta : Q \times \Sigma \rightarrow Q$, **qo** estado inicial tal que qo é elemento de Q; **F** conjunto de estados finais tal que F está contido em Q (MENEZES, 1998).

C.1 Autômato Finito com Saída

O conceito básico de Autômato Finito possui aplicações restritas, pois a informação de saída é limitada à lógica binária aceita/rejeita. Sem alterar a classe de linguagens reconhecidas. É possível estender a definição de AF incluindo a geração de uma palavra de saída. As saídas podem ser associadas às transições (Máquina de Mealy) ou aos estados (Máquina de Moore). Em ambas as máquinas, a saída não pode ser lida, ou seja, não pode ser usada como memória auxiliar, e é como segue:

- É definido sobre um alfabeto especial, denominado alfabeto de saída (pode ser igual ao alfabeto de entrada);
- A saída é armazenada em uma fita independente da de entrada;
- A cabeça da fita de saída move uma célula para direita a cada símbolo gravado;
- O resultado do processamento do autômato finito é o seu estado final (condição de aceita/rejeita) e a informação contida na fita de saída.

C.2 Máquina de Mealy

A máquina de Mealy é um Autômato finito modificado de forma a gerar uma palavra de saída para cada transição.

Definição 2: Uma máquina de Mealy M′ é um autômato finito determinístico com as saídas associadas às transições. É representada por uma sêxtupla: **M′=(Σ,Q,δ,qo,F,Δ)** onde: Σ é o alfabeto de símbolos de entrada, **Q** conjunto de estados possíveis do autômato o qual é finito, δ função programa ou função de transição, **qo** estado inicial tal que qo é elemento de Q, **F** conjunto de estados finais tal que F está contido em Q, Δ Alfabeto de símbolos de saída MENEZES (1998).

$$\delta : Q \times \Sigma \rightarrow Q \times \Delta^*$$

C.3 Máquina de Moore

A máquina de Moore possui uma segunda função, que gera uma palavra de saída (que pode ser vazia) para cada estado da máquina.

Definição 3: Uma máquina de Moore M′′ é um autômato finito determinístico com as saídas associadas aos estados. É representada por uma 7-upla: M′′=(Σ,Q,δ,qo,F,Δ, δs) onde: Σ é o alfabeto de símbolos de entrada, **Q** conjunto de estados possíveis do autômato o qual é finito, δ função programa ou função de transição $\delta : Q \times \Sigma \rightarrow Q$, **qo** estado inicial tal que qo é elemento de Q, **F** conjunto de estados finais tal que F está contido em Q, Δ Alfabeto de símbolos de saída, δs função de saída : $\delta s: Q \rightarrow \Delta^*$.

C.4 Equivalência das Máquinas Moore e Mealy

Teorema 1:Toda máquina de Moore pode ser simulada por uma Máquina de Mealy, para entradas não vazias.

Prova: Suponha MO=(Σ, Q, δMO, qo, F, Δ, δs), uma Máquina de Moore qualquer. Seja: ME=(Σ, Q∪{qe}, δME, qe, F, Δ) uma máquina de Mealy onde a função δME é definida como segue (suponha q um estado de Q e a um símbolo de Σ):

a) δME(qe,a)=(δMO(qo,a), δs(qo) δs((δMO(qo,a)))

b) δME(q,a)=(δMO(q,a),δs((δMO(q,a)))

Em b) é construída a função programa da Máquina de Mealy, a partir das funções de transição e de saída da Máquina de Moore. O estado qe introduzido em a) é referenciado somente na primeira transição a ser executada. Seu objetivo é garantir a geração da saída referente ao estado inicial qo de Moore.

Um exemplo de simulação de Máquina de Moore (Figura C.1) utilizando, utilizando a Máquina de Mealy (Figura C.2). Pela definição 3, de máquina de Moore, têm-se: MO=(Σ, Q, δMO, qo, F, Δ, δs) onde: Σ={ao,a1}, Q ={qo,q1}, δMO:QxΣ→Q, qo, F={qo,q1}, Δ={uo,u1}, δs= Q→Δ^*.

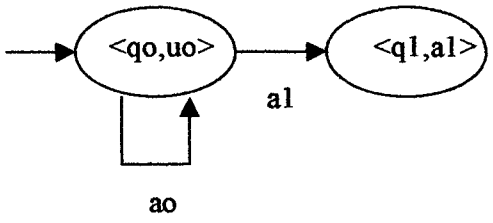


Figura C.1 – Máquina de Moore

δs={(qo,uo),(q1,u1)}

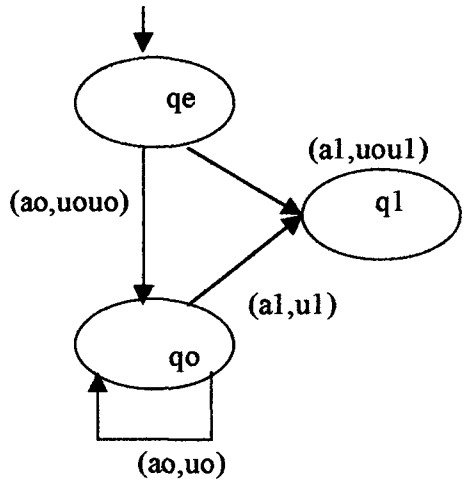


Figura C.2 Máquina de Mealy Equivalente

Tabela C.1 –Função programa da Máquina de Moore

δMO	Ao	a1
Qo	Qo	q1
q1	-	-

C.5 Equivalência das Máquinas Mealy e Moore

Teorema 2:Toda máquina de Mealy pode ser simulada por uma máquina de Moore.
Prova: Suponha ME=(Σ, Q, δME, qo, F, Δ), uma Máquina de Mealy qualquer. A máquina de Moore pode ser construída a partir de δME onde a função δMO é como segue:

- a)Para a em Σ, se δME(qo,a)=(q,u) então:δMO=(<qo,ε>,a)=<q, u>

- b)Para b em Σ, e para q em Q, se δME(q,b)=(p,v), então, para ai em Σ e para qi em Q tais que δME(qi,ai)=(q,ui), tem-se que:

  δMO=(<qo,ui>,b)=<p, v> e onde a função de saída δs é tal que, para o estado <q,u> de MO: δs=(<q,u>)=u

Como exemplo considere uma Máquina de Mealy ME=({a,β},{q,p},δME,q,{q,p}, {a,β}}, simulando a Máquina de Moore, que compacta os brancos de um texto onde a representa um símbolo qualquer do texto e β o símbolo branco, como ilustrado na (Figura C.3) (na etiqueta de uma transição, a primeira componente representa o símbolo lido e a segunda a palavra gravada).

A máquina de Moore construída conforme o Teorema 2 é MO==({a,β}, Q, δMO,<q,ε>,F,{a,β}, δs}), tal que Q=F={q,p}x{ε,a,β}, ilustrada na Figura C.4. onde a segunda componente de cada estado representa a saída.

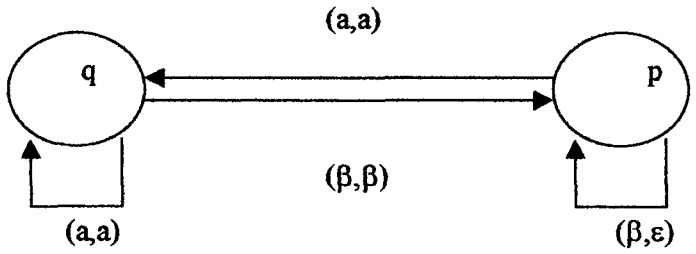


Figura C.3 Máquina de Mealy

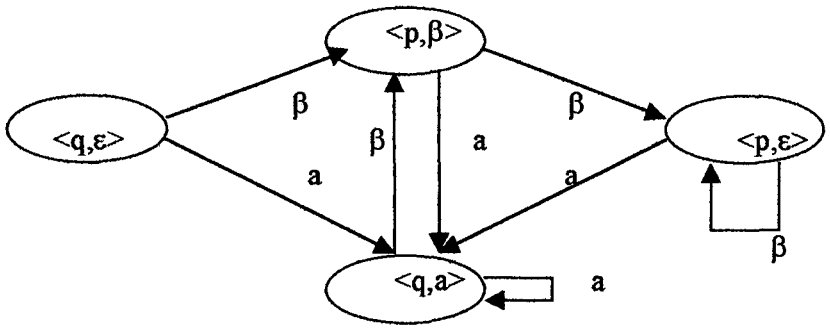


Figura C.4 Máquina de Moore

ANEXO D - MÉTODOS DE CODIFICAÇÃO

D.1) Método de Codificação Direta

No presente enfoque, cada conexão da arquitetura é diretamente especificada pela sua representação binária MILLER, TOOD e HEDGE (1989), VICO e SANDOVAL (1991), MURRAY (1994). Por exemplo, uma Matriz C=(cij) NxN, pode representar uma rede neural com N nós, onde cij indica a presença ou ausência de conexão do nó i para o nó j. Pode-se convencionar cij=1 para indicar uma conexão e cij=0 para indicar não conexão. Também cij, pode representar um valor de peso do nó i para o nó j, tal que a arquitetura e os pesos das conexões podem ser evoluídos simultaneamente.

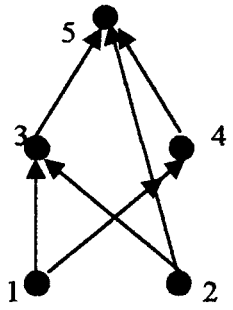
Cada matriz C tem um mapeamento de um-para-um para a rede neural . As Figuras D.1 e D.2 mostram dois exemplos do esquema de codificação direta de duas RNAs.

A Figura D.1 mostra uma rede neural direta, com duas entradas e uma saída. A matriz de conexão é dada na Figura D.1(b), onde cij indica a presença ou ausência de conexão do nó i para o nó j. Por exemplo, a primeira linha indica conexão do nó 1 para os nós 3 e 4. As duas primeiras colunas são zero, porque não existe conexão do nó 1 para ele mesmo e não tem conexão para o nó 2. Converter essa matriz de conectividade para o cromossomo como mostra a Figura D.1(c). Pode-se concatenar as linhas e colunas e obtendo-se: 00110 00101 00001 00001 00000.

A Figura D.2 representa uma rede neural recorrente. A representação é basicamente a mesma para redes neurais diretas, a única diferença é que não existe redução no tamanho do cromossomo, sendo possível, se desejado, explorar o espaço completo de conectividade. O algoritmo evolucionário usado para evoluir redes recorrentes pode ser o mesmo usado para evoluir redes diretas.

O método de codificação como descrito anteriormente é direto para implementar, é bastante satisfatório para uma busca precisa de uma arquitetura compacta de rede neural artificial, desde que uma conexão pode ser adicionada e

removida da rede neural artificial facilmente. Um problema potencial do esquema de codificação direta é a escalabilidade. Uma Rede Neural Artificial grande requer uma matriz muito grande para representar a estrutura da mesma, além disso, o tamanho do cromossomo e tempo de processamento aumentam.



(a)

0	0	1	1	0
0	0	1	0	1
0	0	0	0	1
0	0	0	0	1
0	0	0	0	0

(b)

Cromossomo

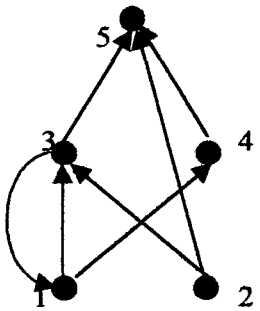
00110 00101 00001 00001 00000

Representação Reduzida

0110 101 01 1

(c)

Figura D.1 – Um exemplo de codificação direta de uma rede neural artificial direta. (a), (b) e (c), mostram a sua arquitetura, sua matriz de conectividade e a representação de sua string binária, respectivamente.



(a)

0	0	1	1	0
0	0	1	0	0
1	0	0	0	1
0	0	0	0	1
0	1	0	0	0

(b)

Cromossomo

00110 00100 10001 00001 01000

(c)

Figura D.2 – Um exemplo de codificação direta de uma rede neural artificial recorrente. (a), (b) e (c), mostram a sua arquitetura, sua matriz de conectividade e a representação de sua string binária, respectivamente.

D.2) Métodos de Codificação Indireta – Gramáticas

Com o objetivo de reduzir o tamanho da representação genotípica das arquiteturas, vários métodos de codificação indireta vêm sendo investigados por vários pesquisadores (HARP et. al., 1990), (KITANO 1990), onde somente algumas características da arquitetura são codificadas no cromossomo. Cada conexão na rede neural artificial são especificados por um conjunto de regras de produção.

Um método diferente de codificação direta apresentada em D.1 é o de codificação gramatical (KITANO, 1990), (YAO e SHI, 1995). O mesmo possibilita uma representação genotípica mais compacta para a evolução de arquiteturas de redes neurais. Um exemplo de regras de produção do método de codificação gramatical é apresentado na Figura D.3(a).

KITANO (1990) aplicou esta idéia geral ao desenvolvimento de redes neurais, usando um tipo de gramática chamada de "gramática geradora de grafos", da qual um exemplo simples é mostrado na Figura D.3. A matriz 8x8 da Figura D.3(b), pode ser interpretada como uma matriz de conexão que representa uma rede neural. Um 1 na linha i e coluna i(i=0,1,2,.....,7), significa que o neurônio i está presente na rede. Um 1 na linha i e coluna j, i≠j, significa que há uma conexão do neurônio i para o neurônio j. Conexões chegando ou partindo a neurônios inexistentes e conexões redundantes são ignorados. O resultado é a rede ilustrada na Figura D.3(c). A gramática utilizada foi: **G=({S,A,B,C,D.....Z,a,b,c,e,....p},{0,1},P,S}**, as regras de produção são mostradas abaixo, na Figura D.3(a).

O cromossomo é dividido em regras separadas, cada qual com 5 posições. A primeira posição (célula) é o lado esquerdo da regra, a segunda até a quinta são ocupadas pelos 4 símbolos da matriz do lado direito da regra. Os possíveis genes em cada célula são os símbolos A-Z e a-p, Figura D.4. A primeira posição do cromossomo é destinada a ser ocupada pelo símbolo inicial S, pelo menos uma regra conduzindo **S** a uma matriz 2x2 é necessária para iniciar o processo. Todos os outros símbolos são escolhidos ao acaso. Uma rede neural é gerada aplicando-se as regras gramaticais codificadas pelo cromossomo para um número pré-determinado de iterações. As regras que levam a-p "as 16 matrizes 2x2 nos símbolos terminais 0 e 1, são fixas e não são representados no cromossomo ", Figura D.3.

Alguns bons resultados têm sido conseguidos com o uso do método de codificação gramatical (KITANO, 1990). O tamanho do cromossomo utilizado é menor do que no método de codificação direta, desde que o AG trabalhe nas regras de produção da gramática para construir as redes neurais. Entretanto, o método tem algumas limitações: é necessário sempre predefinir o número de passos de reescrita,

além disso, não é muito bom evoluindo padrões de conectividade pormenorizados entre nós individuais. Embora a idéia de KITANO (1990) de evoluir gramáticas seja interessante, ainda não há comprovação de que o método seja robusto, pois os problemas abordados nos experimentos eram simples.

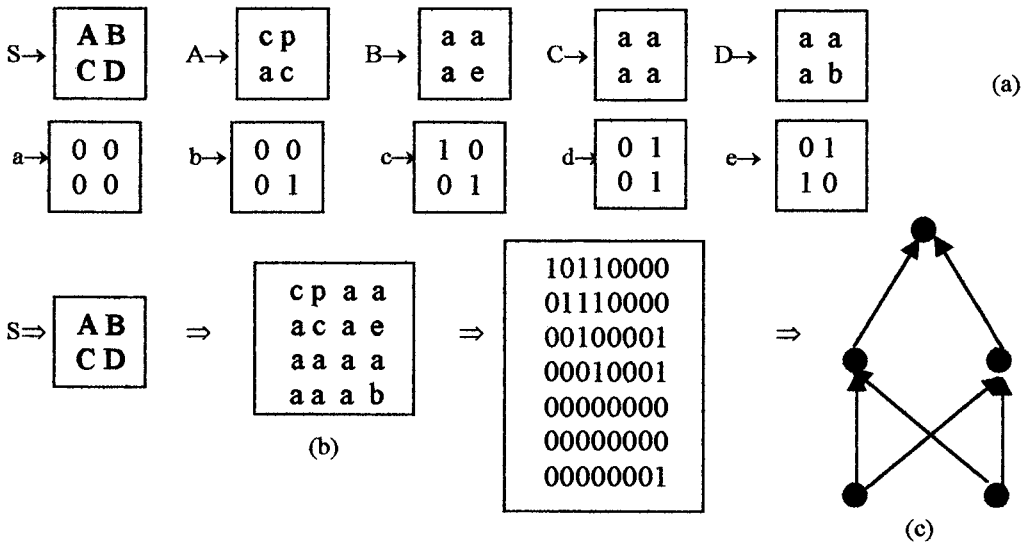


Figura D.3 – Ilustração do método de Kitano "gramática para geração de grafos". (a) Regras da gramática, (b) Matriz de conexão produzida da gramática, (c) A rede resultante.

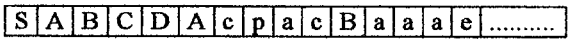


Figura D.4-Ilustração do cromossomo codificando as regras de produção

D.3) Métodos de Codificação Indireta – Sistemas de Lindenmayer

No trabalho de BOERS e KUIPER (1992) uma interpretação do uso de Sistemas de Lindenmayer para geração de topologias de grafos que gerem redes diretas é apresentada. Em BOERS (1995) uma interpretação generalizada é dada, permitindo a construção de grafos com ciclos, que permitem a geração de redes recorrentes, essa construção é denominada gramática para geração de grafos "G2L-Systems".

As regras de produção utilizadas no "G2L-Systems" são sensíveis ao contexto da forma L<P>R→S. Como exemplo considere o "G2L-System": $G=\{\Sigma g, \Pi, \alpha\}$, com $\Sigma=\{A,B,C,D\}$, As regras de produção (Π), são mostradas na Tabela D.1.

Tabela D.1 – Regras de Produção para Geração de Grafos

1) A→B1B1B	A é substituído por B1B1B.
2) B>B→[CD]	B é substituído pelo subgrafo [CD] se é sucedido por B.
3) B→C	B é substituído C
4) C<D→C	D é substituído por C se é precedido por C
5) D>D→C2	D é substituído por C2 se é sucedido por D

Inicialmente, tem-se o axioma A (Figura D.5(a)), após um passo de reescrita tem-se a string B1B1B (Figura D.5(b)). Durante o segundo passo de reescrita, o primeiro B é reescrito usando a regra 2, pois o primeiro B é conectado ao segundo (Figura D.5(c)).

O segundo B é também reescrito usando a regra 2 Figura D.5(d). O terceiro B é reescrito usando a regra 3, pois o terceiro B não é conectado a outro B (existe apenas uma conexão de outro B). Isto, finalmente, resulta na "string" [CD]1[CD]1C, mostrando na Figura D.5(e). Pelo fato de que todas as reescritas são realizadas em paralelo, as "strings" para as Figuras D.5(c) e D.5(d) não são criadas separadamente, mas a Figura D.5(b) é diretamente reescrita em D.7(e). Durante o terceiro passo de reescrita o primeiro D é reescrito usando a regra 5 pelo fato de que o primeiro D é conectado ao segundo D (Figura D.5(f)) e o segundo D é reescrito usando a regra 4, pois o primeiro C é conectado ao segundo D. Não são mais aplicadas nenhuma regra e a "string" final resultante é [CC2]1[CC]1C, mostrado na Figura D.5(g). Novamente a Figura D.5(f) é apenas para simplificar os passos de reescrita, na realidade a Figura D.7(e) é reescrita em paralelo na Figura D.5(g).

O método utilizado por BOERS e KUIPER (1995) permite a utilização de regras de produções recursivas, o que possibilita a obtenção de RNA com um tamanho maior do que o método de KITANO (1990), sendo por isso mais robusto. Apesar da metodologia de KITANO (1990) usar uma gramática livre de contexto a mesma permite a utilização de regras recursivas, entretanto isso não foi explorado no método razão da limitação do mesmo. A principal vantagem da metodologia utilizada por BOERS e KUIPER (1992) é questão da plausibilidade biológica e que foi discutido no capítulo 5, como desvantagens têm-se: o tamanho do cromossomo é de 1024 bits, o tempo de

processamento é elevado e as regras de produção necessitam ser decodificadas do cromossomo.

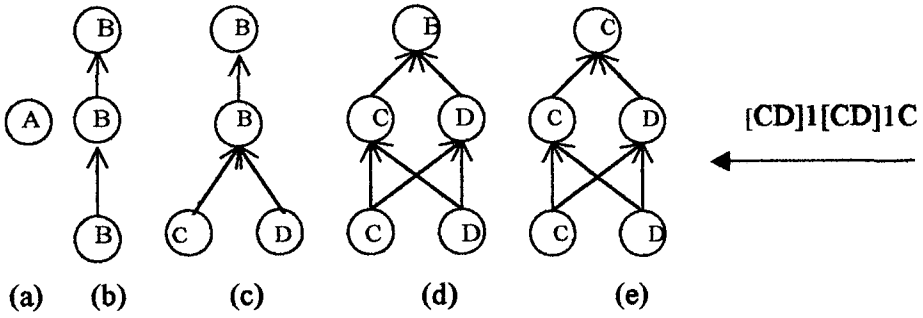


Figura D.5 – Grafos gerados pelo "G2L-Systems"

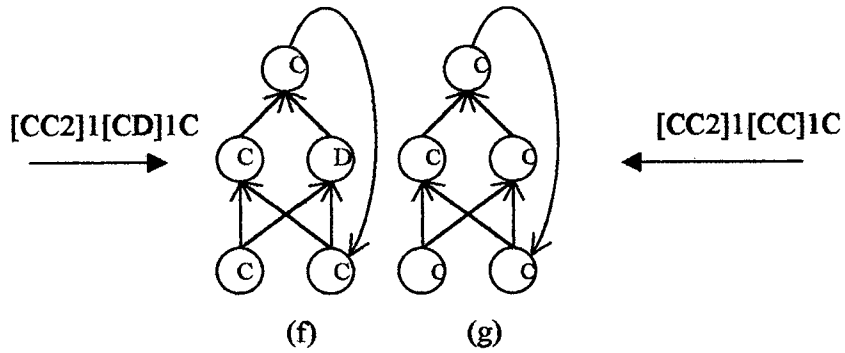


Figura D.5 – Grafos gerados pelo "G2L-Systems"

ANEXO E – RELATÓRIO FINAL PROBLEMA DA PARIDADE

REGRAS DE PRODUÇÃO

A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAAB%%%%BCCDDD2D	N
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAABBCDBBCCDDD2D	S
A,B,CAAB%%%%BCCDDD2D	N

OBS:O último campo "S" indica conjunto de regras de produção válidas e "N" indica conjunto de regras de produção não válidas.

<Ord>	<Fit>	<Posto>	<Soma dos Postos>	<ERM>	<Inter>	<RG>	<NREC>
6	83.659021	20	210	1.321216e-05	2	4	2
5	83.658543	19	190	1.329623e-05	2	4	2
9	83.658523	18	171	1.329971e-05	2	4	2
10	83.65833	17	153	1.333386e-05	2	4	2
14	83.656856	16	136	1.360131e-05	2	4	2
17	83.653586	15	120	1.423426e-05	2	4	2
16	58.685877	14	105	9.75194e-06	3	4	3
7	58.683257	13	91	1.000768e-05	3	4	3
18	58.680048	12	78	1.03397e-05	3	4	3
2	58.679011	11	66	1.045174e-05	3	4	3
0	58.669286	10	55	1.163431e-05	3	4	3
4	37.142619	9	45	4.425703e-06	5	4	5
8	15.025	8	36	0.56707	1	4	1
13	15.025	7	28	0.584497	1	4	1
11	15.025	6	21	0.585351	1	4	1
3	15.025	5	15	0.585359	1	4	1
1	15.025	4	10	0.585432	1	4	1
15	8.358333	3	6	0.723159	2	4	2
12	0	2	3	0	0	0	0
19	0	1	1	0	0	0	0

Fitness Medio=45.798415

Onde :

Ord= Número identificador da rede.

Fit= Avaliação após o treinamento da rede identificada por Ord.

Posto= Nota atribuída a rede, Soma dos Postos=Nota Acumulada (método da roleta ponderada).

Inter=Número de neurônios da camada intermediária.

RG= Número de regras de produção que geraram a rede.

NREC= Número de recorrências, ERM=Erro médio.