V Escola de Redes Neurais, Promoção: Conselho Nacional de Redes Neurais
pp. c091-c121, 19 de julho, 1999 - ITA, São José dos Campos - SP

V Escola de Redes Neurais, ITA, 19/Julho/1999

ALGORITMOS GENÉTICOS EM REDES NEURAIS ARTIFICIAIS

Prof. Fernando Mendes de Azevedo, Dr.
azevedo@gpeb.ufsc.br

Grupo de Pesquisas em Engenharia Biomédica (GPEB)
Departamento de Engenharia Elétrica (EEL)
Universidade Federal de Santa Catarina (UFSC)

O objetivo do presente minicurso é de apresentar, em quatro horas, os conceitos básicos de Algoritmos Genéticos e de Redes Neurais Artificiais e como os primeiros podem ser usados para resolver alguns problemas relacionados aos segundos. Apesar de que algumas soluções, aqui apresentadas, constituem-se no estado da arte, o texto não pretende ser profundo ou formal, visto que o público alvo deve ser constituído, em princípio, por estudantes de Engenharia Elétrica ou recém graduados.

O texto relativo a este minicurso foi escrito baseado em trabalhos realizados durante os programas de doutorado realizados no Grupo de Pesquisas em Engenharia Biomédica do EEL – UFSC pelos Engenheiros Lourdes Mattos Brasil, Roberto Célio Limão de Oliveira, Mauro Roisenberg e João da Silva Dias nos últimos cinco anos. Os Doutores L. M. Brasil e R. C. L. de Oliveira foram orientados pelo Prof. Fernando Mendes de Azevedo, do Departamento de Engenharia Elétrica (GPEB – EEL) e co-orientados pelo Prof. Jorge Muniz Barreto, do Departamento de Informática e Estatística, (INE), ambos departamentos da UFSC. O Doutor M. Roisenberg foi orientado pelo Prof. Jorge Muniz Barreto e co-orientado pelo Prof. Fernando Mendes de Azevedo. O Dr. João da Silva Dias foi orientado pelo Prof. Jorge Muniz Barreto.

Além dos trabalhos dos quatro doutores acima referenciados, serviram de base, também, para este minicurso, alguns trabalhos do próprio autor, Prof. Fernando Mendes de Azevedo e alguns do Prof. Jorge Muniz Barreto.

Todas as citações, tanto destes autores citados, como de outros, encontram-se nas Referências Bibliográficas.

ALGORITMOS GENÉTICOS EM REDES NEURAIS ARTIFICIAIS

I. FUNDAMENTOS DE ALGORITMOS GENÉTICOS

I.1. Introdução

No final da década de 50, alguns pesquisadores buscaram na natureza inspiração para novas técnicas de busca de soluções. O motivo para a atenção ter se voltado para a natureza deve-se ao fato desta conseguir resolver, de forma satisfatória, problemas altamente complexos, como é o caso da sobrevivência das espécies. Aliado a este fato, é possível explicar a grande maioria dos seres vivos através de poucos processos de natureza estatística (cruzamento, mutação e seleção), agindo sobre uma população de uma espécie [Fogel95] [Back97].

A tentativa de imitação do cérebro humano na expectativa de um comportamento emergente deu origem as Redes Neurais Artificiais (RNA). Já a tentativa de imitar a evolução dos seres vivos na natureza originou a Computação Evolucionária (CE).

Computação Evolucionária é o nome genérico, dado a métodos computacionais, inspirados na teoria da evolução. Os algoritmos usados, em computação evolucionária, são conhecidos como Algoritmos Evolucionários (AE) [Barreto96].

Atualmente, os AE mais conhecidos são: Algoritmos Genéticos (AG), Programação Evolucionária (PE) e Estratégias Evolucionárias (EE). Todos compartilham de uma base conceitual comum, que consiste na simulação da evolução de estruturas individuais, via processos de seleção e os operadores de busca, referidos como Operadores Genéticos (OG), tais como, mutação e "crossover" (cruzamento ou recombinação). O processo depende do "fitness" (aptidão), atingido pelas estruturas individuais, frente a um ambiente. A seleção é focalizada nos indivíduos com um alto grau de aptidão, explorando então, a informação da aptidão disponível. O "crossover" e a mutação perturbam estes indivíduos, fornecendo heurística geral para a exploração.

O AG foi proposto inicialmente por John H. Holland em 1975 no trabalho intitulado "Adaptation in Natural and Artificial Systems" [Holland75]. Holland inspirou-se no mecanismo de evolução das espécies, tendo como base os trabalhos de Darwin sobre a origem das espécies [Darwin81] e na genética natural, devido principalmente a Mendel [Dunn50]. De acordo com a teoria Darwiniana de evolução das espécies, uma população sujeita a um ambiente qualquer, sofrerá influências desse, de tal forma que os mais aptos terão maior probabilidade de sobreviver a tal ambiente. Já os trabalhos de Mendel mostram como o material genético dos pais pode ser passado para os descendentes. Desta forma, a cada geração haverá uma população mais adaptada ao ambiente em questão [Holland75][Goldberg89].

I.2. Base Biológica

Charles Darwin e Alfred Russel Wallace foram os precursores, através de suas evidências para a teoria de evolução, em 1858, na revolução tanto do pensamento biológico quanto da

filosofia humana. Desde então, esta teoria é uma das mais aceitas pelo mundo científico, originando o chamado paradigma Neo-Darwiniano, proveniente da teoria evolucionária Darwiniana clássica, combinada com o selecionismo de Weismann e a genética de Mendel [Fogel95].

O Neo-Darwinismo afirma que a história de existência de vida, em nosso mundo, é atribuída completamente a alguns poucos processos estatísticos, que agem sobre populações e espécies. Estes processos são: reprodução, mutação, competição e seleção. Reprodução é uma propriedade óbvia de toda vida. Mas, similarmente, mutação é garantida para algum sistema, no qual, ela própria se reproduz continuamente, em um universo positivamente entrópico. Competição e seleção, tornam-se as conseqüências inevitáveis de alguma população expandida que esteja restrita a uma área finita. Evolução é, então, o resultado desses processos estatísticos, interagindo, fundamentalmente, nas populações, geração após geração [Fogel95].

A evolução, na natureza, ou em qualquer lugar, não é um processo proposital ou dirigido, isto é, não há evidências de que a meta da evolução seja a produção do homem. No entanto, os processos da natureza parecem se constituir em diferentes indivíduos, competindo por recursos no ambiente. Alguns são melhores do que os outros. Aqueles que são melhores estão mais habilitados a sobreviver e propagar sua carga genética.

Na natureza, a perpetuação da informação genética é mantida pelo genoma, de modo que ela é realizada através da reprodução assexuada e sexuada. A codificação é um sistema igual em todos os seres vivos, porém com variações, e ocorre no DNA, o que confere a diversidade.

Na reprodução sexuada, a produção de descendentes se dá pela união de dois seres diferentes (cujo material genético é organizado em pares de unidades mais simples, os cromossomos), da mesma espécie, denominados macho e fêmea, os quais, diferem pela carga genética que contêm os cromossomos sexuais, de forma a produzirem células chamadas de gametas. O macho produz gametas masculinos e a fêmea produz gametas femininos. Um novo ser é produzido pela união de um gameta masculino com um gameta feminino, que se divide, sucessivamente, até formarem-se todos os órgãos e sistemas do indivíduo. Assim, um grupo dessas células, chamadas germinativas, é capaz de produzir um novo macho ou uma fêmea. É o processo de reprodução dos seres mais elevados na escala filogenética.

Por outro lado, na reprodução assexuada não há diferença genética entre seres da mesma espécie, no que se refere ao seu papel na reprodução. Na descendência o material genético parental é trocado entre seus cromossomos (“crossover”), resultando em cromossomos filhos, cujo material genético é a combinação dos materiais dos dois progenitores. Este processo é o imitado pela maioria dos AE.

Convém ainda notar, quanto a evolução biológica, que esta exige diversidade. Na natureza, a diversidade deriva de mutações.

I.3. Algoritmos Genéticos (AG)

AG constituem uma técnica de busca, inspirada no processo de evolução dos seres vivos, baseada na seleção natural de Darwin.

Considerando os sistemas biológicos como um todo, observa-se que os mesmos desenvolveram, ao longo da sua evolução, estratégias de adaptação de comportamento, que possibilitaram a sua sobrevivência e a perpetuação de suas espécies. As pressões do ambiente fizeram com que estas estratégias tivessem um forte impacto sobre os organismos biológicos, gerando profundas mudanças nos mesmos. Manifestações destas mudanças podem ser observadas nas especializações estruturais e funcionais, na organização da informação e nas representações internas do conhecimento [Austin90].

Baseado nesta analogia com o processo de evolução biológica das espécies, chamada de metáfora biológica, os AG mantêm a informação sobre o ambiente, acumulando-a durante o período de adaptação. Eles utilizam tal informação acumulada para podar o espaço de busca e gerar novas soluções plausíveis dentro do domínio.

Entre os principais fatores que têm feito do AG uma técnica bem sucedida destacam-se [Silva95][Goldeberg89][Tanomaru95]:

- simplicidade de operação;
- facilidade de implementação;
- eficácia na busca da região onde, provavelmente, encontra-se o máximo global;
- aplicável em situações onde não se conhece o modelo matemático ou este é impreciso e também em funções lineares e não lineares.

Os AG podem ser enquadrados, em grande parte dos problemas científicos a serem formulados, como problemas de busca e otimização. O problema de otimização pode ser solucionado por meio de métodos numéricos, enumerativos e probabilísticos, ou por hibridismo destes métodos.

Os métodos numéricos podem ser divididos em analíticos, cuja função *f(x)* é explicitamente conhecida e derivável, ou pode ser aproximada, por alguma função derivável até o grau desejado de precisão, enquanto que, nos baseados em cálculo numérico, caso o espaço de busca seja linear, técnicas de Programação Linear, como o método *simplex*, são suficientes [Tanomaru95]. Contudo, em ambientes não-lineares, técnicas de gradiente ou de estatística de ordem superior são geralmente empregadas. Já os métodos enumerativos de otimização examinam cada ponto do espaço de busca, um por um, em busca dos pontos ótimos. Por outro lado, os métodos probabilísticos são métodos que empregam a idéia de busca probabilística, isto é, descrevem a variação de sistemas, que se realizam, essencialmente, sob condições inalteradas. Esses sistemas são chamados de sistemas aleatórios, de forma que a teoria de probabilidade permite modelar seu comportamento.

Os AG fazem parte da classe correspondente aos métodos probabilísticos de busca e otimização, apesar de não serem aleatórios. Os AG usam o conceito de probabilidade, mas

não são simples buscas aleatórias. Pelo contrário, os AG tentam direcionar a busca para regiões onde é provável que os pontos ótimos estejam.

Além disso, em relação as técnicas de busca convencionais, os AG diferem nos seguintes pontos:

- A busca da melhor solução para o problema é feita sobre uma população de pontos, e não sobre um único ponto, reduzindo sensivelmente o risco da solução recair sobre um máximo (ou mínimo) local;

- Os AG realizam uma busca cega. A única exigência é o conhecimento do valor da função de custo (ou objetivo) de cada indivíduo. Não há necessidade de qualquer outra informação, ou heurística, dependente do problema.

- Os AG usam operadores estocásticos e não regras determinísticas para guiar uma busca altamente exploratória e estruturada, onde informações acumuladas nas iterações (gerações) anteriores são usadas para direcionar essa busca.

Apesar de sua simplicidade, os resultados obtidos com a aplicação do método, segundo Goldberg [Goldberg89], permitem concluir que os AG são um método de busca robusto, eficiente e eficaz em uma grande variedade de problemas.

I.3.1. Conceitos Fundamentais de AG

A nível biológico, um indivíduo é formado por um conjunto de cromossomos. No entanto, pode-se fazer uma analogia, neste contexto, entre indivíduo e cromossomo, tendo em vista que um indivíduo pode ser formado por apenas um cromossomo, o que é comum em AG. Por isso, os dois termos são utilizados indistintamente, neste contexto.

Assim, um indivíduo é definido por um "string", usando-se um alfabeto finito, de modo que cada "string" represente um conjunto de valores para o conjunto de parâmetros do problema. Um exemplo de alfabeto é o conjunto {0,1}, ou o conjunto de números inteiros. Deste modo, cada posição do "string" representa um gene.

O cromossomo é composto de genes sendo que cada gene possui um local fixo no cromossomo, local este denominado de "locus". Cada gene pode assumir um certo valor, pertencente a um certo conjunto de valores, os quais, são denominados de "alelo" [South93]. Em termos de AG, o gene é denominado de "bit" e o "locus", de posição do "bit" no indivíduo. Já o termo alelo, refere-se ao conjunto de valores possíveis de serem atribuídos a um determinado "bit".

Ao conjunto de cromossomos, genes e alelos, denomina-se de genótipo, e as características conferidas por este, denomina-se de fenótipo. Em termos de AG, o genótipo é a variável independente, x, e o fenótipo, a variável dependente ou função, $f(x)$ [Dias99].

Normalmente, os AG trabalham com um conjunto de indivíduos (população)[1], no qual, cada elemento é candidato a ser a solução desejada. Cada indivíduo é codificado em uma cadeia de bits, denominada de cromossomo. A cadeia de cromossomos, representada por números binários é de comprimento fixo. A função a ser otimizada é o ambiente, no qual a população inicial vai ser posta. Espera-se que, através dos mecanismos de evolução das espécies e a genética natural, somente os mais aptos se reproduzam e, também, que cada nova geração esteja mais apta ao ambiente (função a ser otimizada).

O grau de aptidão de cada indivíduo é obtido pela avaliação de tal 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 da função, a aptidão será inversamente proporcional ao valor da função [Tanomaru95].

Assim, quando já se tem realizado o teste de todos os indivíduos da população, na função a ser otimizada, obtem-se a aptidão para cada um, ou seja, o seu grau de aptidão.

A próxima geração será uma evolução da anterior e, para que isso ocorra, os mais aptos, os de melhor aptidão, deverão possuir maior probabilidade de serem selecionados para dar origem à nova geração. Com isso, 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 dos indivíduos da geração anterior, que vão participar da formação da nova geração, pode ser realizada através da roleta ponderada. Na roleta ponderada, os indivíduos que obtiveram melhor valor de aptidão, recebem maior nota. Além disso, os valores são acumulativos, como é exemplificado na Tabela 1.

Tabela 1 – Roleta Ponderada

Elemento	Nota	Nota Acumulada
X_3	5	5
X_2	4	9
X_1	3	12
X_4	2	14
X_5	1	15

Como pode ser visto na Tabela 1, ao mais apto foi dado a nota máxima, 5, e ao menos apto, foi dada a nota mínima, 1. Em valores acumulados, tem-se 5 para o mais apto e 15 para o menos apto. O ponto crucial da roleta ponderada é a diferença entre as notas acumuladas dos elementos da população. Nota-se, que a diferença do mais apto, com nota acumulada de 5, para o segundo, com nota acumulada de 9, é de 4 unidades de espaçamento. Por outro lado, tomando o menos apto, com nota acumulada de 15, e o penúltimo, com nota acumulada de 14, nota-se que a distância entre eles é de somente 1 unidade. Tal característica torna maior a chance dos mais aptos serem sorteados em relação aos menos aptos. Nesse caso, na razão de 4:1 [Dias99].

[1] No caso de se utilizar AG para otimização da topologia de rede neural, o conjunto de indivíduos será representado por uma população de redes.

Além disso, na roleta ponderada o sorteio é realizado pela geração de um número aleatório, segundo uma distribuição uniforme ou distribuída, dependendo do tipo de aplicação. A escolha dos melhores indivíduos para participar da reprodução, faz com que a média da população caminhe na direção mais promissora da solução desejada [Machado92a][Machado92b][Tanomaru95].

Realizada a seleção, o próximo passo é a aplicação dos mecanismos de busca, também conhecidos como OG. Entre tais mecanismos, os mais comumente empregados em AG, são: "crossover" e mutação. Estes operadores serão descritos com maiores detalhes, na próxima seção.

Um outro ponto relevante a ser mencionado, diz respeito aos parâmetros do AG, ou seja, os valores que influenciam o desempenho do AG. Seguindo a relação proposta por S. Austin [Austin90], estes parâmetros são: tamanho da população, taxa de operadores, intervalo de geração, seleção de estratégia e fator de escalada.

O tamanho da população de cromossomos afeta o desempenho global dos AG. Uma população pequena é insuficiente para cobrir o espaço de busca do problema. Uma população grande é mais representativa do domínio, além de evitar a convergência prematura para soluções locais, em vez de soluções globais.

As taxas de operadores medem a freqüência com que cada tipo de OG é utilizado. Representam, também, a influência que cada tipo de OG exerce sobre a população de cromossomos. Além do que, se a taxa de operadores de "crossover" for muito alta, alguns cromossomos de bom desempenho podem ser removidos mais rapidamente do que a seleção possa desenvolvê-los. Se a taxa de "crossover" for muito baixa, a busca pode estagnar. Entretanto, se a taxa dos operadores de mutação for baixa, evita-se que uma dada posição estabilize-se em um único valor. Uma taxa de mutação alta resulta essencialmente numa busca aleatória.

O intervalo de geração controla o percentual da população, a ser substituído durante cada ciclo de geração. Por exemplo: N x G cromossomos da população P(t) são escolhidos para serem substituídos na população P(t + 1). Se o valor de G for igual a 1, significa que toda a população é substituída durante cada geração.

As estratégias de seleção correspondem aos critérios utilizados para a escolha de cromossomos durante a reprodução. Um exemplo de estratégia de reprodução, baseado em S. Austin [Austin90], é a seleção pura, onde os cromossomos são reproduzidos em função da sua aptidão.

O fator de escalada mede a manutenção da diversidade genética da população de cromossomos durante a evolução. Um cromossomo ou um grupo de cromossomos pode ter uma aptidão bastante forte, a ponto de dominar o processo de reprodução, reduzindo-se a diversidade da população. Uma maneira de se controlar este processo é ordenando os cromossomos, escalonando o seu desempenho, para refletir sua aptidão relativa dentro da população, e utilizando as operações genéticas de mutação para reduzir a homogeneidade da população de cromossomos.

I.3.2. Operadores Genéticos

Indivíduos novos são criados usando-se dois principais operadores de recombinação genética, conhecidos como “crossover” e mutação.

O “crossover” se dá pela aproximação dos cromossomos dos dois indivíduos (pais), que trocam entre si partes de seus cromossomos. Isso resulta em dois cromossomos diferentes que, porém, ainda guardam influências dos pais [Barreto96]. Há várias formas possíveis de se fazer o cruzamento [Dias99][Kozek93][Park94][Srinivas94]. O operador “crossover“, mais simples, é o chamado “crossover” de um ponto (One-Point), onde, primeiro um local de cruzamento é escolhido com probabilidade uniforme sobre o comprimento do cromossomo, sendo, então, os “strings” correspondentes permutados, como é mostrado na Figura 1. Há, ainda, muitas outras técnicas de “crossover”, como é o caso do “crossover” de dois pontos (Two-Point), e dos tipos uniformes [Garis92][Srinivas94]. Contudo, não há consenso sobre qual é a melhor técnica a ser usada.

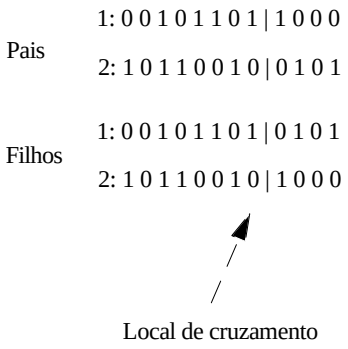

Figura 1 – Operador “crossover“ de um ponto

A mutação consiste em perturbações na cadeia dos cromossomos dando origem a uma nova cadeia, que guardará pouca ou nenhuma informação da cadeia mãe. Na realidade, mutação é a denominação dada a vários mecanismos de alteração genética, os quais têm em comum o fato de fazerem o novo cromossomo apresentar pouca informação dos pais [Algarve97]. Esta alteração ocorre de forma que cada gene em cada cromossomo é um candidato à mutação, enquanto que a seleção é determinada pela probabilidade de mutação. Esta probabilidade é mantida, usualmente, em um valor baixo, para evitar-se a perda de um número grande de cromossomos bons [Park94]. O operador de mutação pode ser implementado de várias maneiras. A codificação binária de “string” é o modo mais fácil para executá-la.

A tarefa da mutação em AG tem sido a de restituir a perda ou material genético inexplorado na população, com o objetivo de prevenir a convergência prematura do AG para soluções sub-ótimas [Srinivas94].

Dentre os principais mecanismos de alteração genética, que recebem a denominação global de mutação, destacam-se: troca simples, translocação, inversão, deleção e adição.

Na adição, ocorre a inserção de mais um gene na cadeia, e na deleção, é justamente o oposto, ou seja, ocorre a retirada de um gene da cadeia. Geralmente, estes mecanismos não são utilizados em algoritmos genéticos, pois alteram o comprimento da cadeia do cromossomo.

A troca simples consiste de um erro de cópia, de um ou mais genes da cadeia. Se um gene for considerado como sendo um bit com valor lógico 1, a ocorrência de troca simples levaria este bit (gene) para nível lógico 0 e vice-versa (Figura 2). Já a inversão consiste na retirada e inserção de um pedaço da cadeia, porém, na ordem inversa da que foi retirada. Ao contrário da inversão - onde um pedaço do código é retirado e colocado no mesmo local com ordem inversa -, a translocação retira uma parte do cromossomo e coloca em outra posição do mesmo cromossomo. Estes três últimos mecanismos não alteram o comprimento original da cadeia e, como a maior parte dos trabalhos em algoritmo genético utilizam cadeia de comprimento fixo, estes são os mais comumente utilizados [Algarve97]. No entanto, como na maioria dos trabalhos com AG, este também usa o termo mutação, como sinônimo de troca simples [Algarve97][Dias99][Holland75][Soucek91].

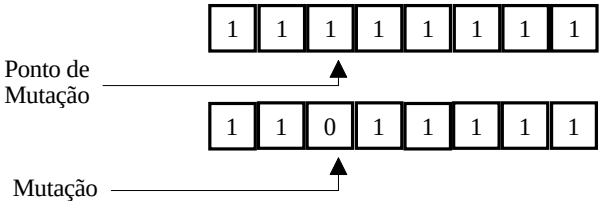

Figura 2 – Exemplo de mutação (troca simples)

Por último, após a seleção e a aplicação dos OG, tem-se uma nova geração, a qual deve ser avaliada, visando comparar o seu grau de aptidão em relação a geração anterior. Caso tal geração não esteja apta o suficiente, deve-se repetir o processo de seleção e reprodução, até que o grau de aptidão seja aceitável [Algarve97][Dias99].

I.3.3. Algoritmo Genético Simples

O Trabalho original de Holland (1975) propõe os seguintes passos principais para um algoritmo genético simples [Sirivas94]:

- Geração da população inicial;
- Validação dos elementos da população e análise de convergência;
- Seleção;
- Manipulação genética.

Geração da População Inicial

A população inicial pode ser obtida através da geração aleatória de indivíduos, obedecendo condições de contorno previamente estabelecidas pelo usuário. O usuário estabelece estas condições, tendo em vista o seu conhecimento prévio do problema a ser otimizado. Quanto mais restringente forem as condições de contorno, mais rápida será a convergência, isso porque os valores gerados aleatoriamente estarão mais próximos da solução desejada [Dias99].

O número de elementos, que comporá a população, ainda é motivo de estudos, mas existem várias heurísticas, ou seja, depende muito da experiência do usuário e do seu conhecimento prévio sobre a função a ser otimizada [Davidor92]. É claro que, quanto maior o número de elementos na população, maior é a probabilidade de convergência, tendo em vista que aumenta a probabilidade da solução desejada ser constatada entre os elementos da população. Em contrapartida, o tempo de processamento também aumenta. Já, no caso da população inicial ser muito pequena, ela terá o problema da perda de diversidade, isto é, o espaço de busca seria muito pequeno para ser avaliado. Desta forma, a solução obtida poderia não estar dentro do ótimo global. Conseqüentemente, a convergência seria prematura.

A população inicial não precisa, necessariamente, de ser gerada aleatoriamente, tendo em vista que o objetivo é gerar uma população dentro de certo intervalo onde se acredita estar a resposta. Também, pode-se obter a população inicial através de um escalonamento do número de indivíduos, que compõem no intervalo especificado, isto é: se a população é de 50 indivíduos e o intervalo inicial é de 0 a 10, os indivíduos da população inicial deverão ser distribuídos uniformemente neste intervalo.

O número de elementos na população, a probabilidade de ocorrer cruzamento e a probabilidade de acontecer mutação, são denominados de parâmetros de controle dos AG [Sirivas94].

Validação dos Elementos da População e Análise de Convergência

A validação é o processo de expor cada elemento da população a função de custo (objetivo) e, ao final, ordená-los de acordo com a aptidão à esta função.

Na convergência, analisa-se o desempenho da população para ver se o objetivo foi atingido. Isto pode ser feito através de vários fatores, tais como: valores máximo, mínimo e médio da função de aptidão. Também, é relativamente comum utilizar-se o desvio padrão dos valores da função de aptidão, como forma de análise da convergência [Goldberg89].

Como o AG é regido por população, se na população inicial tiver um elemento que seja a resposta exata do problema, o AG ainda assim não finalizará o processo de busca da solução. A finalização ou convergência só ocorrerá quando a aptidão média da população estiver suficientemente estável, ou seja, quando houver pouca variação da aptidão média da população atual em relação a anterior. Isto indica que a população se adaptou ao meio, isto é, os elementos da população levam a função ao valor otimizado/desejado [Sirivas94].

Utiliza-se memorizar o indivíduo mais apto, independentemente deste fazer, ou não, parte da população atual. Assim, ao final, este será o resultado esperado [Tanomaru95].

Contudo, na utilização de AG pode ocorrer uma rápida convergência prematura para uma solução sub-ótima, porém não o esperado ótimo global. Este problema é denominado convergência prematura, podendo ocorrer devido a população reduzida ou a má distribuição da população inicial, em torno do ponto sub-ótimo. Ou seja, um indivíduo próximo de um ótimo local, possui um valor de aptidão superior aos demais indivíduos da população.

Conseqüentemente, o processo de seleção fará com que este indivíduo tenha grande chance de dominar a próxima geração e, assim sucessivamente, se não aparecerem outros indivíduos com melhores valores de aptidão [Tanomaru95][South93].

Conforme pode ser visto, a convergência prematura pode ocorrer devido a uma má distribuição dos indivíduos no espaço de busca. Esta má distribuição, também recebe a denominação de perda da diversidade [Tanomaru95][Goldberg89]. Segundo Júlio Tanomaru [Tanomaru95], o conceito de diversidade indica o grau em que as mais diversas regiões estão representadas no espaço de busca. Este problema pode ser amenizado através da escolha criteriosa do número de indivíduos na população, melhora da distribuição dos indivíduos da população inicial no espaço de busca e, também, impedindo a perda de diversidade nas primeiras gerações.

Seleção

A seleção tem por objetivo fazer com que somente os elementos mais aptos da geração anterior participem do processo que irá gerar a nova população.

O processo de seleção tem início após a verificação do grau de aptidão de cada elemento à função de custo e a verificação da não convergência dos valores.

O processo de validação fornece os elementos da população, em ordem de aptidão. Uma das formas empregadas na seleção, para pegar somente os mais aptos, é o da roleta ponderada [Fogel95]. Na roleta ponderada, imagina-se uma roleta em que cada casa, corresponde a um indivíduo, sendo a área da casa proporcional ao valor de aptidão de cada indivíduo, de modo que os indivíduos mais aptos têm maior probabilidade de serem selecionados. Desta forma, a aptidão de cada indivíduo é usada para aumentar sua probabilidade de sobrevivência, e não utilizada de forma determinística [Barreto96].

Manipulação Genética

A etapa de manipulação genética consiste na aplicação de OG, isto é dos operadores "crossover" e/ou mutação, somente em alguns elementos que tiveram maior valor de aptidão, quando sorteados por meio da roleta ponderada. Ao término desta etapa terá sido gerada uma nova população, que deverá repetir os passos anteriores até que a aptidão da população seja aceitável.

Inicialmente é necessário estabelecer alguns pontos importantes, tais como: manter o tamanho da população fixo e garantir que uma parcela da nova população seja composta por elementos obtidos por seleção da população anterior, além do que, o complemento seja de elementos manipulados pelos OG.

Como exemplo, se a população for composta de 40 elementos, na etapa de seleção deverão ser sorteados, com o auxílio da roleta, 20 elementos (caso seja escolhida uma renovação de 50%) que passam diretamente a fazer parte da nova geração, enquanto isso, os outros 20 elementos poderão sofrer manipulação pelos OG. Com o auxílio da roleta, torna-se claro que entre os 40 elementos sorteados os mais aptos, ou melhor, os de maiores notas acumuladas aparecerão mais vezes.

Quanto aos OG, costuma-se executá-los em seqüência nos AG simples, isto é, inicialmente aplica-se o OG de "crossover" e, após, o de mutação [Goldberg89].

A aplicação do "crossover" implica na composição de 10 casais a partir dos 20 elementos, sendo que alguns serão acasalados e outros não. Para tanto gera-se dois números aleatórios: o primeiro, entre 0 e 1, indicará a probabilidade de ocorrer "crossover" e, o segundo, o local da realização do "crossover". Caso o primeiro número gerado seja inferior ao definido pelo usuário, como probabilidade de "crossover", realiza-se o "crossover" propriamente dito, caso contrário copia-se os pais para a nova geração [Tanomaru95]. O segundo número aleatório, gerado no caso da ocorrência de "crossover", indicará a posição de corte do cromossomo para efetuar o "crossover". Para tanto o número aleatório gerado deverá estar entre *1* e *g - 1*, onde *g* é o número de genes ou bits do cromossomo.

Para a aplicação do OG de mutação, há necessidade de gerar um número aleatório para cada bit de cada indivíduo. Este número randômico é denominado de probabilidade de mutação e deverá ser comparado com a probabilidade de mutação estipulada pelo usuário, para o problema em questão. Caso seja inferior a esta, executa-se a mutação, caso contrário repete-se o processo para o próximo bit do indivíduo, até que todos os indivíduos tenham sido analisados [Goldberg89][Tanomaru95].

A probabilidade de ocorrer mutação é sempre bem menor que a de ocorrer "crossover". Segundo M. Sirivas e L.M. Patnaik [Sirivas94], existe um compromisso entre os três parâmentros de controle do AG simples: tamanho da população, probabilidade de "crossover" e probabilidade de mutação. Muitos autores têm proposto valores para estes parâmetros visando garantir uma boa performance do AG, porém, estes valores ainda fazem parte de várias heurísticas [Jason93].

I.4. Aplicações de AG para Redes Neurais

AG podem ser aplicados às Redes Neurais Artificiais (RNA) em três modos:

- treinamento de RNA (ajuste dos pesos);
- otimização da topologia de RN (número de neurônios da camada intermediária);
- geração tanto da topologia como dos pesos das conexões.

A maioria dos trabalhos em que AG são utilizados tratam do problema de treinamento, ou seja, da geração dos pesos das conexões da rede [Dias99][Tanomaru95][Jason93]. Neste tipo de aplicação, dada uma estrutura para uma rede, um AG é utilizado para achar os pesos que resultam nos menores valores de erro, ao invés de algoritmos de treinamento convencionais, como o "Backpropagation" (retropropagação). Por exemplo, Montana [Montana89] mostrou como é possível treinar RNA usando AG. Muhlenbein [Muhlenbein89] abordou a dinâmica da evolução combinada com aprendizado, dando os primeiros passos para a compreensão dos dois mecanismos agindo simultaneamente [Muhlenbein91]. Prado [Prado92] e Porto [Porto95] exploraram o treinamento de RNA diretas através de AG.

Trabalhos em que os AG são utilizados para a escolha da topologia da RNA melhor adaptada à solução de um problema são mais raros. Nestes trabalhos geralmente os genes codificam a topologia da rede, especificando que conexões estão presentes. Os pesos são, então, determinados por outros métodos. Como exemplo de pesquisas nesta área pode-se citar os trabalhos de Garcia [Garcia95][Vico91][Brasil99].

Quanto a terceira alternativa, utilizando os AG tanto para a escolha da topologia quanto para o treinamento da RNA, alguns trabalhos nesta área foram publicados por Karunanithi, Das & Whitley [Karunanithi92] e Angeline [Angeline93].

II. FUNDAMENTOS DE REDES NEURAIS

II.1. Introdução

Redes neurais são sistemas complexos constituídos por elementos representando algumas das características dos neurônios que constituem o sistema nervoso de seres vivos e permitindo sua interação com o ambiente que os cerca. Trata-se de assunto tipicamente interdisciplinar interessando à engenheiros, matemáticos, fisiologistas, psicólogos, etc. Conseqüentemente não existe um normatização relativa à nomenclatura e notação, entre outras.

Neste trabalho, baseado em [de Azevedo97], apresenta-se, uma proposta para modelos formais de Neurônios Artificiais e Redes de Neurônios Artificiais na direção da referida normatização permitindo uma linguagem unificada que sirva aos pesquisadores com as mais diversas formações. Por outro lado, devido ao fato dos referidos modelos utilizarem-se de conceitos de Teoria de Sistemas e de Teoria de Grafos, supõe-se que os leitores serão, ao menos em sua maioria, conhecedores de alguns aspectos matemáticos que serão explorados. Para maiores conhecimentos sobre a abordagem de Teoria de Sistemas, aqui utilizada, recorra a [Barreto96] [Barreto97] e [de Azevedo93].

II.2. Modelos de Neurônios

A construção de redes neurais artificiais (RNA) tem inspiração nos neurônios biológicos e nos sistemas nervosos[2]. Entretanto, é importante compreender que, atualmente, as RNAs estão muito distantes das redes neurais naturais (RNN) e, freqüentemente, as semelhanças são mínimas. Se é verdade que o primeiro modelo de neurônio, proposto por McCulloch e Pitts, em 1943 [McCulloch43] é, também, um modelo simples, cabe ressaltar que a intenção era de imitar a realidade biológica, preocupação não compartilhada pelos muitos pesquisadores atuais. De fato, dois fatores diferentes motivam a pesquisa hoje em dia:

O primeiro é modelar o sistema nervoso com suficiente precisão de tal modo a poder observar um comportamento emergente que, sendo semelhante ao comportamento do ser vivo modelado, possa servir de apoio às hipóteses usadas na modelagem; o segundo é construir computadores com um alto grau de paralelismo.

[2] Para uma apresentação dos conceitos de fisiologia, indispensáveis à compreensão das RNA, consultar os bons livros de fisiologia, por exemplo, o clássico Guyton [Guyton76].

O trabalho na modelagem do sistema nervoso começou há um século, aproximadamente. Depois dos trabalhos de McCulloch e Pitts [McCulloch43], Hebb [Hebb49] e Rosenblatt [Rosenblatt58], muitos cientistas se interessaram pelo campo. O desejo de construir neurocomputadores (ou Computadores baseados em Redes Neurais - CBRN [Barreto90] [de Azevedo93]) é mais recente [Hecht88].

II.2.1. Modelo de McCulloch-Pitts

Warren S. McCulloch era um fisiologista preocupado com problemas filosóficos e, também, poeta. Ele costumava chamar sua especialidade de epistemologia experimental. Uma coleção dos seus trabalhos se encontra no volume "Embodiments of Mind" [McCulloch65] com introdução de S. Papert.

Sendo um fisiologista e conhecendo as ondas de potencial de membrana, ele interpretou o funcionamento do neurônio como sendo um circuito binário. Seu modelo é, portanto, binário e é apresentado na Figura 3.

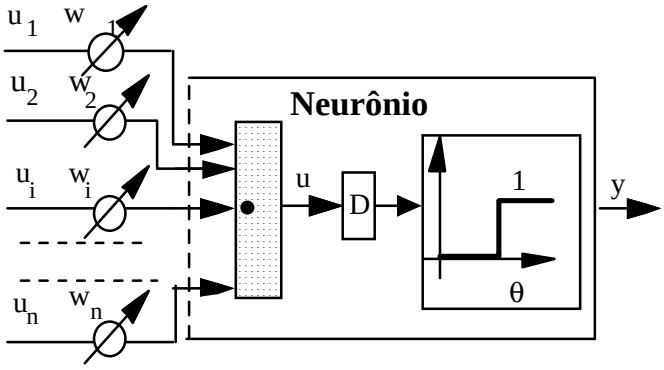

Figura 3 - Modelo de McCulloch e Pitts (Figura de [Barreto96a] com permissão)

A entrada do neurônio é, também, binária e as várias entradas são combinadas por uma soma ponderada, produzindo a entrada efetiva do neurônio (também conhecida como "net_input"):

$$entrada\_efetiva = net\_input = \sum_{1}^{n} w_i u_i \qquad (1)$$

O resultado na entrada efetiva[3] sofre um retardo *D*. Algumas vezes, este retardo, que é correspondente, principalmente, à difusão através da sinapse dos neuro-transmissores, é desprezado e serve de argumento a uma função chamada de função de transferência. Neste caso, de saída binária {0 1}, para dar a resposta do neurônio.

Usando este modelo de neurônio ele provou, juntamente com seu aluno Walter H. Pitts (um matemático), usando argumentos lógicos, a equivalência de suas redes com a máquina de

[3] Ver, por exemplo, as equações referentes ao teorema III em "A Logical Calculus of Ideas Immanent in Nervous Activity" [McCulloch43].

Turing. Entretanto, muitos não ficaram satisfeitos com esta prova, principalmente devido à complexidade da demonstração. Posteriormente outros pesquisadores, entre eles Arbib [Arbib64] [Arbib87], apresentaram provas mais didáticas.

II.2.2. Modelo geral de neurônio

O modelo geral de neurônio é uma generalização do modelo de McCulloch e Pitts. Este modelo foi proposto, em 1993, por de Azevedo [de Azevedo93] e, depois utilizado, com modificações, em suas teses de Doutorado, por Roisenberg [Roisenberg98] e de Oliveira [de Oliveira99].

Define-se, então, o modelo geral de neurônio como um sistema dinâmico, conforme segue:
Teorema: O neurônio formal é um sistema dinâmico.
Prova: Basta identificar as várias variáveis e funções com as da definição de sistema dinâmico. Assim, na definição de sistema dinâmico, se considera o seguinte objeto matemático:

$$S = \{T, U, \Omega, \mathrm{Y}, \Gamma, \mathrm{X}, \Phi, \lambda\} \tag{2}$$

onde:
T é o conjunto dos tempos;
Ω é o conjunto das funções de entrada $\omega \in \Omega = \{\omega: \mathrm{T} \rightarrow \mathrm{U}\}$;
U é o conjunto dos valores da entrada;
Y é o conjunto dos valores da saída;
Γ é o conjunto das funções de saída $\gamma \in \Gamma = \{\gamma: \mathrm{T} \rightarrow \mathrm{Y}\}$;
X é o conjunto dos estados;
Φ é a função de transição dos estados: Φ: T x Tx X x $\Omega \rightarrow$ X; e
λ é a função de saída[4]: λ: T x X x U $\rightarrow$ Y

Identificando as variáveis e funções do neurônio tem-se:
T, conjunto dos tempos, é geralmente o conjunto dos inteiros;
Ω, o conjunto das funções de entrada ou provenientes dos órgãos sensores ou da saída de outros neurônios;
Γ é o conjunto das funções de saída cujos valores são representativos das freqüências de descarga dos neurônios;
U é o conjunto dos valores da entrada, valores representativos das saídas dos órgãos sensores ou dos valores de freqüências de descarga de outros neurônios;
Y é o conjunto dos valores da saída, valor representativo da freqüência de descarga do neurônio;
X é o conjunto dos estados possíveis e representa a excitação do neurônio;
Φ é a função de transição dos estados: Φ: T x T x X x $\Omega \rightarrow$ X, e mostra como a excitação do neurônio evolui; e
λ é a função de saída: λ: T x X x U $\rightarrow$ Y que, freqüentemente, é tomada como uma função cujo valor é sempre o valor de excitação do neurônio.

[4] A função λ freqüentemente não depende de U.

Tendo identificado todas as variáveis e funções envolvidas na definição de neurônio, pode-se dizer, então, que o modelo de neurônio é um exemplo de sistema dinâmico.

Nota: é importante observar que, muitas vezes, o modelo de neurônio é estático. Este é um caso particular de nossa definição quando a cardinalidade do espaço de estados é um.

Nota: Outra observação importante é que nada é dito, nesta definição, acerca dos pesos das conexões entre os diferentes neurônios constituintes de uma RNA. Estes (os pesos sinápticos) aparecerão quando da definição de redes neurais artificiais. No entanto, pode-se, desde já, considerar que os neurônios têm, quando conectados entre si para formar uma RNA, um peso associado a cada uma de suas entradas.

Pode-se, por conseguinte, representar o neurônio formal conforme a Figura 4.

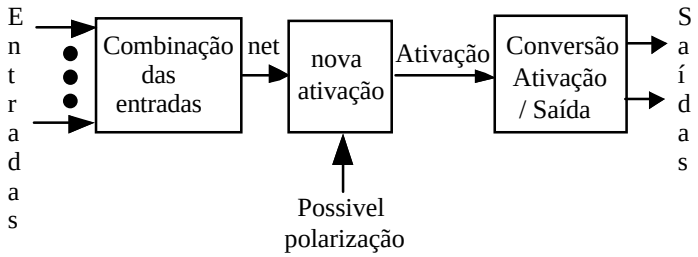

Figura 4 - Neurônio artificial

Cada entrada tem um peso associado (ao nível de rede). Estas entradas podem ser saídas de outros neurônios, entradas externas, uma polarização (bias) ou qualquer combinação delas. A associação de todas as entradas dá origem a entrada total efetiva net_i, que é representada da seguinte forma:

$$net_i(t) = \sum_{j=1}^{n} a_{ij} y_j(t) + \sum_{k=1}^{m-1} b_{ik} u_k(t) + b_{im} bias_i \qquad (2)$$

onde y_j são as saídas de outros neurônios, u_k entradas externas, $bias_i$ uma polarização, e a_{ij} e b_{ik} os pesos correspondentes.

Nota: O "bias" é um parâmetro opcional que pode ser obtido pela simples adição de uma entrada constante (normalmente com valor unitário) e um peso apropriado.

Todavia é comum incorporar a constante $bias_i$ e as saídas y_j com as entradas u_k.. Os pesos correspondentes a_{ij} e b_{ik} são representados por w_{ij} com $j = 1$ para $n+m$ ou, simplesmente $j = 1$ para n. A nova equação é, então:

$$net_i(t) = \sum_{j=1}^{n} w_{ij} u_j(t) \qquad (3)$$

onde:
w_{ij} é um número real que sumariza a conexão sináptica da entrada do $i^{ésimo}$ neurônio para a saída do $j^{ésimo}$ neurônio. A conexão sináptica é conhecida por excitatória se $w_{ij} > 0$ e por inibitória se $w_{ij} < 0$.

Depois da determinação de net_i, a ativação é atualizada através da função de ativação, Φ, para produzir um novo estado de ativação do neurônio que, através da função λ, vai produzir a saída do neurônio (correspondente à freqüência de descarga do neurônio biológico). Um valor auxiliar θ é, geralmente, usado para representar uma polarização, valor abaixo do qual a saída é nula.

Note-se que isso poderia, também, ser obtido por escolha adequada da função λ, mas seria mais difícil de trabalhar. Note-se, mais uma vez, *que as conexões sinápticas são consideradas como externas ao modelo do neurônio*, tal como ocorre no sistema nervoso biológico e não como fazendo parte do neurônio, como usado por alguns autores. Se este detalhe pode ter pouca importância aparente no estudo de uma RNA, proporciona a possibilidade de interpretar a matriz de conexões como a matriz de pesos de um grafo, o grafo representativo da rede neural, como sugerido por de Azevedo em [de Azevedo93].

Apesar deste modelo ser uma generalização do modelo de McCulloch-Pitts, conforme já citado, ele apresenta, no entanto, uma diferença conceitual importante: enquanto no modelo de McCulloch-Pitts o estado excitado interpreta a despolarização da membrana celular, (ao menos é o que aparenta), neste modelo geral fica bem claro que a resposta do neurônio, podendo variar de modo contínuo entre zero e um valor máximo, pode ser interpretada como a freqüência de descarga do neurônio biológico e, portanto, este modelo geral pode servir para modelar processos biologicamente plausíveis.

Nota: Geralmente net_i é a soma das entradas. Algumas vezes, o produto. Raramente, uma outra função, se bem que isto seja possível.

Este modelo permite, usando conceitos de Teroria de Sistemas, definir os vários tipos de neurônios existentes. Assim:

Definição: O neurônio é linear se Φ e λ são funções lineares.

Definição: O neurônio é não estacionário se as funções Φ e/ou λ são funções do tempo. Neste caso deve-se escrever:

$$x = \Phi (w_i, u_i, t)$$
$$y = \lambda (x, t)$$

Definição: O neurônio é dito dinâmico se, para o cálculo de *x* em um determinado instante, é necessário o conhecimento de *x* em um instante anterior.

Nota: Por esta definição nota-se que o modelo de neurônio proposto por McCulloch e Pitts é um sistema dinâmico se o retardo *D* não for nulo.

II.2.3. A função de transição de estados Φ

A função de transição de estados é uma função dos estados anteriores, das entradas e dos pesos sinápticos (considerando ao nível de rede). Examinemos a equação:

$$x(t+h) = \Phi(x(t), net(t), t, h) \quad (4)$$

$$y(t) = \lambda(x(t), net(t), t) \quad (5)$$

Esta equação mostra que os estados futuros, do neurônio, são afetados pelo estado atual bem como pelo net_i. Dado um "valor inicial" $x(t_0)$ de uma variável de estado $x(t)$ no tempo $t=t_0$, é possível predizer os futuros valores $x(t+h)$ de $x(t)$. A equação de estado 4 é uma equação de diferenças (equação discreta). Por outro lado, se nós fazemos o incremento de tempo h menor e menor, o somatório torna-se uma integral, x variando continuamente com o tempo t e a equação 4 torna-se uma equação diferencial como segue:

$$\frac{dx}{dt} = \Phi'(x, net, t) \quad (6)$$

Os dois tipos de neurônios discutidos acima são conhecidos como "neurônios dinâmicos". Conseqüentemente, eles apresentam memória. Por outro lado, conforme mencionado anteriormente, se nós considerarmos a função Φ como uma constante, os neurônios são "neurônios estáticos" significando que o novo estado é igual ao anterior. Estes neurônios não apresentam memória. Eles podem ser considerados como um caso particular dos neurônios dinâmicos quando o espaço de estados é um "singletão".

Exemplos de funções de transição de estados podem ser encontrados no trabalho de Hunt [Hunt92], entre outros.

II.2.4. A Função de saída λ

As mais comuns funções de saída usadas em neurônios individuais são a função linear, a função logística e a função tangente hiperbólica. Todavia, essencialmente qualquer função monotonicamente crescente e contínua tal que $x \in \Re$ e $y(x) \in [-1,1]$ pode ser usada em modelagem neural. A seguir nós discutimos algumas dessas funções.

- a função linear:

$$y(x) = x \quad (7)$$

- a função logística, que é a mais popular função unipolar:

$$y(x) = \frac{1}{1+e^{-kx}} \quad (8)$$

onde k é um escalar positivo.

Outras funções podem ser derivadas pelo ajuste da constante k. Quanto mais k cresce, mais abrupta a função é nas proximidades de $x = 0$. Note que quando $k \to \infty$, a função é definida por:

$$y(x) = \begin{cases} +1 & \text{se } x > 0 \\ 0 & \text{se } x < 0 \end{cases}$$

A Figura 5 mostra, diversas funções logísticas de ativação para k = 0.5, 1, 2, 4 e 8.

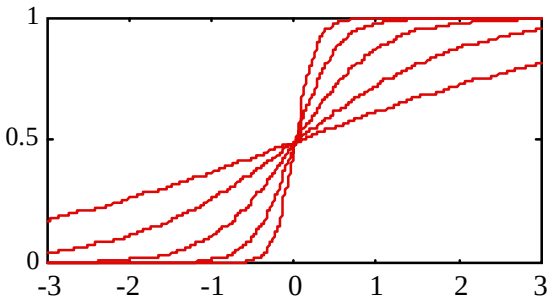

Figura 5 - Funções de ativação unipolar

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

$$y(x) = \tanh(kx) = \frac{e^{kx} - e^{-kx}}{e^{kx} + e^{-kx}} \tag{9}$$

onde k é um escalar positivo.

De novo outras funções podem ser derivadas pelo ajuste da constante k. Quanto mais k cresce mais abrupta a função é nas proximidades de $x = 0$. Note que quando $k \to \infty$, a função torna-se a função *sgn(x)* que é definida por:

$$y(x) = \begin{cases} +1 \text{ se } x > 0 \\ -1 \text{ se } x < 0 \end{cases}$$

A Figura 6 mostra diversas funções tangente hiperbólica para k = 0.5, 1, 2 e 4.

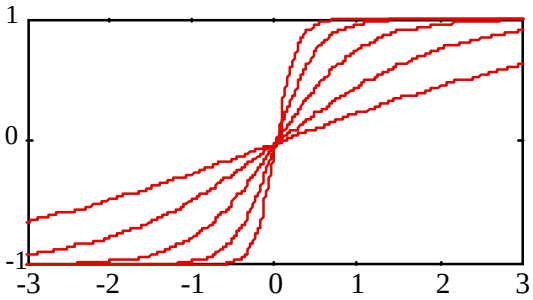

Figura 6 - Funções de ativação bipolar

Estas funções são, também, denominadas de "com características sigmoidais", e quando $k \to \infty$ elas se tornam "hard-limiting functions", as quais descrevem modelos de neurônios discretos.

Nota: As funções sigmoidais, como as acima citadas, são necessárias para alguns algoritmos de aprendizado, tais como o algoritmo "backpropagation".

Da discussão acima nós podemos ver que os neurônios, mesmo sendo elementos computacionais bastante interessantes, não são muito poderosos do ponto de vista de

computação e representação. Esta é a razão pela qual se usa conjuntos de neurônios para melhorar as referidas características. Estes conjuntos de neurônios, denominados Redes Neurais Artificiais (RNA), apresentam algumas propriedades interessantes. Por exemplo, Hopfield, em seu trabalho publicado em 1982 [Hopfield82], mostrou a existência de capacidades computacionais emergentes, ao nível de redes, que não são predizíveis ao nível do simples neurônio.

II.3. Topologias das RNA

De forma a definir as Redes Neurais Artificiais nós poderíamos, em princípio, estabelecer (e provar) um teorema mostrando que elas se constituem em sistemas dinâmicos, da mesma forma que foi feito para os neurônios. Todavia, um problema surgiria aqui: nada seria dito acerca dos pesos das conexões entre os diferentes neurônios da rede. Uma outra abordagem seria a de considerar uma rede neural como um sistema dinâmico complexo, onde:

Definição: Um sistema dinâmico complexo é uma rede de sistemas interconectados.

Da definição apresentada decorre que um sistema complexo pode ser representado por um grafo direcionado ou dígrafo, onde os vértices representam os sistemas componentes (subsistemas) e os arcos as interações entre subsistemas. Esta será a abordagem utilizada aqui.

Nota: Observa-se que, considerando que, em princípio, qualquer dígrafo possa dar lugar a uma topologia de RNA, esta abordagem vem sendo utilizada em textos surgidos nos últimos anos, como por exemplo [Haykin94], entre outros. No entanto, de Azevedo [de Azevedo93] utilizou esta abordagem ainda em 1993.

Definição: Uma Rede Neural Artificial, RNA, é um Sistema Dinâmico Complexo representado por um grafo arco rotulado em que cada vértice é um Neurônio Artificial NA.

Nesta definição, rótulos são, naturalmente, valores numéricos. Eles correspondem aos valores das conexões entre os diferentes neurônios. Todavia, eles podem ser interpretados, também, como os valores "fuzzy" entre as conexões. Neste caso, eles devem pertencer a um conjunto, que na maioria dos casos, é o conjunto [0, 1] (Outros intervalos de valores podem ser, também, considerados para conjuntos "fuzzy"). Ambas interpretações são válidas para nossos propósitos. Todavia, se nós escolhermos a segunda interpretação nós poderíamos repensar a definição de Grafos e, por conseqüência, a de Redes Neurais, conforme segue:

Definição: Um Grafo "Fuzzy" é um Grafo Arco Rotulado onde os rótulos são valores de um conjunto "fuzzy".

Definição: Uma Rede Neural "Fuzzy" é uma rede neural representada por um grafo "fuzzy".

Tendo estabelecido definições precisas para Redes Neurais e Redes Neurais "Fuzzy" nós podemos definir diferentes tipos de redes. Isto é feito através de escolhas particulares dos conjuntos e funções envolvidas na definição de Redes Neurais como Sistemas Dinâmicos. Tem-se, por conseguinte:

Definição: Uma Rede Neural Contínua no Tempo é uma rede neural definida em um subconjunto contínuo do eixo do tempo, $T = \Re$.

Definição: Uma Rede Neural Discreta no Tempo é uma rede neural definida em um subconjunto discreto do eixo do tempo, $T = Z$.

Definição: Uma Rede Neural Invariante no Tempo ou Rede Neural Estacionária é uma rede neural em que a função de transição Φ depende de um único elemento de T e a função de saída λ é independente de T.

Neste trabalho nós consideramos ambos os tipos de redes, contínuas e discretas. Todavia, todas são invariantes no tempo para permitir uma fácil tratabilidade matemática.

Até agora nós propusemos definições matemáticas para NA e RNA. Estas definições permitem o estudo de diferentes tipos particulares de RNA como sistemas dinâmicos. A abordagem dinâmica para RNA serve como um guia para o estudo da capacidade de memória e para formular idéias no sentido de uma Teoria da Computabilidade adaptada a RNA. A seguir serão apresentadas as topologias de RNA que podem ser derivadas de nossos modelos formais.

II.3.1. Redes diretas ("Feedforward")

Definição: Redes Diretas são aquelas cujo grafo não tem ciclos.

Freqüentemente é comum representar estas redes em camadas e, neste caso, são chamadas redes em camadas. Neurônios que recebem sinais de excitação são chamados da camada de entrada, ou primeira camada. Neurônios que têm sua saída como saída da rede pertencem a camada de saída ou última camada. Neurônios que não pertencem nem a camada de entrada nem a de saída são neurônios internos à rede podendo se organizar em uma ou mais camadas intemediárias ("hidden layers").

A Figura 7 mostra uma rede direta com 3 camadas de neurônios. Observe que, nesta figura, os neurônios são apresentados com os seus diversos elementos constituintes conforme a Figura 4. Estas redes são atualmente as mais populares, principalmente por existirem métodos de aprendizado bastante difundidos e fáceis de usar. Um método bastante usado é o "backpropagation". Por esta razão alguns autores chegam mesmo a chamar, impropriamente, este tipo de rede de "backpropagation". 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 (equação diferencial de primeira ordem ou a uma diferença finita), têm uma dinâmica muito limitada não podendo representar todos os sistemas dinâmicos.

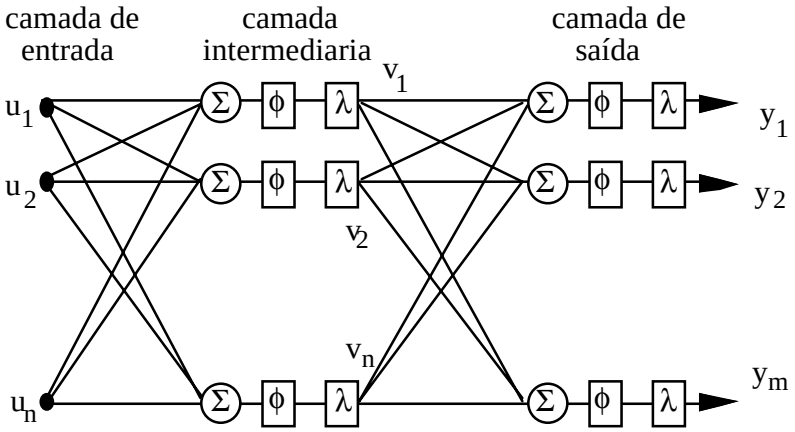

Figura 7 - Uma rede direta com 3 camadas de neurônios

II.3.2. Redes com ciclos

Definição: Redes com ciclos (ou com realimentação, ou com retroação, ou com "feedback") são aquelas cujo grafo de conectividade contém, ao menos, um ciclo.
Um exemplo bem conhecido de rede com ciclos é a proposta por Hopfield [Hopfield82].

Definição: Redes recorrentes são aquelas que, além de apresentarem ciclos, envolvem neurônios dinâmicos.
Por esta razão McCulloch chamou-as de "networks with cycles", ou redes com ciclos.

Duas destas redes têm particular importância: as redes propostas por Hopfield [Hopfield82][Hopfield84] e as redes bi-direcionais, de Kosko [Kosko88], que podem ser usadas em um dos dois principais paradigmas de sistemas especialistas: treinamento com exemplos de uma rede direta e representação do conhecimento de modo localizado pelo uso de rede com ciclos, geralmente uma rede simétrica. De um modo geral estes dois tipos de redes poderiam ser representados, no caso mais geral, pela Figura 8.

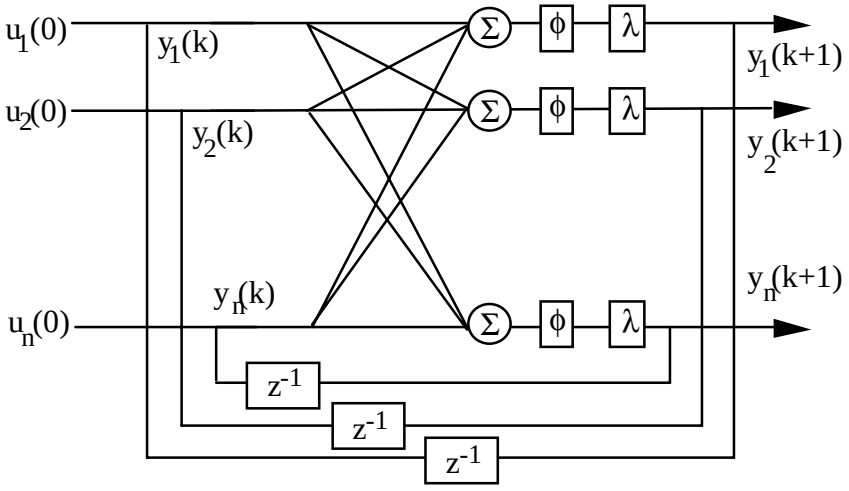

Figura 8 - Uma rede com realimentação e neurônios dinâmicos

II.3.3. Redes simétricas

Definição: Uma rede simétrica é aquela cuja matriz de conectividade é uma matriz simétrica.

Trata-se de um caso particular das redes com ciclos. Com efeito, os sistemas especialistas mencionados anteriormente e que usam redes com ciclos, usam redes simétricas. Isto foi feito para assegurar estabilidade do transitório da rede.

III. REDES NEURAIS: O PROBLEMA DO APRENDIZADO

Aprendizado em RNA é um processo altamente importante e, ainda nos dias de hoje, continua a ser submetido a intensas pesquisas em ambas as abordagens relacionadas as redes biológicas e neurais. Sabe-se, portanto, que o aprendizado não é um processo único. Há diferentes processos de aprendizado, isto é, cada um adequado a diferentes tipos de redes.

De uma maneira geral, o aprendizado pode ser classificado em dois tipos: associativo e não-associativo. O aprendizado associativo implica em aprender sobre o relacionamento que há entre pares de estímulos. Este tipo de aprendizado é um modelo para RNA supervisionadas. Quanto ao aprendizado não-associativo, não há estímulos secundários para associar com os estímulos primários. Neste tipo de aprendizado, a repetição de um estímulo fornece a oportunidade para aprender sobre suas propriedades. Assim, este tipo de aprendizado é um modelo para RNA não-supervisionadas [Lawrence92].

Nota: O aprendizado para redes dinâmicas foge do escopo deste trabalho mas foi explorado por Roisenberg [Roisenberg98] e de Oliveira [de Oliveira99] em suas Teses de Doutorado.

III.1. Aprendizado Supervisionado

Durante a sessão de treinamento de uma RNA, pares de entradas e saídas são apresentados à ela. A rede toma cada entrada e produz uma resposta na saída. Esta resposta é comparada com o sinal de saída desejado. Se a resposta real difere da resposta desejada, a RNA gera um sinal de erro, o qual é, então, usado para calcular o ajuste que deve ser feito para os pesos sinápticos da rede. Assim, a saída real casa com a saída desejada. Em outras palavras, o erro é minimizado. O processo de minimização de erro requer um circuito especial conhecido como um professor ou supervisor, daí o nome Aprendizado Supervisionado.

Para este tipo de aprendizado leva-se em consideração a importância de cálculos requeridos ao minimizar o erro, pois depende do algoritmo usado. Claramente, o algoritmo é puramente uma ferramenta matemática derivada de técnicas de otimização. Tais técnicas são muitíssimo usadas para os paradigmas conexionistas, porém, alguns parâmetros devem ser levados em conta, como: o tempo requerido por iteração, o número de iterações por padrão de entrada para o erro alcançar um valor mínimo durante a sessão de treinamento, se a RNA atingiu um mínimo local ou global, caso tenha alcançado um mínimo local se a rede

pode escapar dele, etc. Um dos algoritmos, que se utiliza dos parâmetros acima mencionados, é o de retropropagação (ou "backpropagation").

III.2. Aprendizado Não-Supervisionado

Em contraste ao aprendizado supervisionado, o aprendizado não-supervisionado[5] não requer um professor, isto é, não há saída desejada. Durante a sessão de treinamento, a RNA recebe em sua entrada excitações muito diferentes ou padrões de entrada e organiza, arbitrariamente, os padrões em categorias. Quando uma entrada é aplicada à rede, a RNA fornece uma resposta de saída indicando a classe a qual a entrada pertence. Se uma classe não pode ser encontrada para o padrão de entrada, uma nova classe é gerada. Este tipo de aprendizado é utilizado em sistemas classificadores.

III.3. Regras de Aprendizado

Para que se possa solucionar um problema, usando-se o modelo das RNA, o primeiro passo é estabelecer um conjunto de pesos para suas conexões, ativar um conjunto de unidades que correspondam a um padrão de entrada e observar o padrão para o qual a rede converge e em que se estabiliza. Se o padrão final não corresponder ao que se deseja associar, como resposta ao de entrada, é preciso fazer ajustes nos pesos e ativar novamente o padrão de entrada. Por causa de sua semelhança com o aprendizado humano, esse processo de ajustes sucessivos das RNA é chamado de aprendizagem.

Os paradigmas desenvolvidos, neste sentido, utilizam regras de aprendizado que são descritas por expressões matemáticas, chamadas de equações de aprendizado [Kartalopoulos96]. Estas equações descrevem o processo de aprendizado para o paradigma, o qual, na realidade, é o processo para auto-ajustar seus pesos sinápticos.

Uma das regras de aprendizado para RNA, bem conhecida, é a regra de Hebb [Hebb49][Hebb79]. Outra, é a regra delta [Widrow62], que é inspirada na regra de Hebb. Esta regra foi desenvolvida por Bernard Widrow e Ted Hoff, por isso, é conhecida também como regra de aprendizado de Widrow-Hoff ou como "Least Mean Square" (LMS), tendo em vista, que ela minimiza o erro médio quadrático [Lawrence92]. Além destas, há a regra delta generalizada, que será descrita a seguir.

III.3.1. Regra Delta Generalizada

Muitas redes usam alguma variação da fórmula para treinamento da regra delta. Uma delas, é a regra delta generalizada ou, também, conhecida por algoritmo de retropropagação.

O algoritmo de retropropagação foi desenvolvido por P. Werbos, em 1974 [Werbos74], e redescoberto, independentemente, por D. Parker (1982) [Parker82] e D.Rumelhart et al. (1986) [Rumelhart86][Kartalopoulos96]. Desde sua redescoberta, o algoritmo de retropropagação tem sido amplamente usado como um algoritmo de aprendizado para RNA, com topologia direta com múltiplas camadas.

[5] Para maiores detalhes sobre outros tipos de aprendizado, veja [Lawrence92][Kartalopoulos96].

Uma RNA que utiliza um algoritmo de retropropagação aprende de forma supervisionada (através de exemplos), em tempo discreto e utiliza um método de gradiente descendente para correção de erro, ou seja, o algoritmo de codificação executa um mapeamento entrada-saída, através da minimização de uma função de custo. A função de custo é minimizada, realizando-se iterativamente ajustes nos pesos sinápticos, de acordo com o erro quadrático acumulado[6] para todos os padrões do conjunto de treinamento[7]. Outras funções de custo podem ser utilizadas, mas independentemente disto, o procedimento de ajuste de pesos é realizado através do cálculo da mudança da função de custo, com respeito à mudança em cada peso (método do delta). O processo de evolução da redução gradativa de erro, que acompanha a minimização, pode levar a convergência. A medida que a rede aprende, o valor do erro converge para um valor estável, normalmente irredutível. O processo de aprendizagem prossegue, até que algum critério seja estabelecido, como por exemplo, um valor mínimo de erro global, ou uma diferença sucessiva mínima entre erros calculados para cada iteração.

IV. REDES NEURAIS: O PROBLEMA DO NÚMERO DE NEURÔNIOS NA CAMADA INTERMEDIÁRIA

O computador pode ser considerado como máquina de resolver problemas. Logo, é natural imaginar que tanto a possibilidade de resolver um problema específico, como quanto vai ser gasto em recursos na tarefa, dependem da máquina usada. Ao fato de que um problema possa ser resolvido com recursos finitos, chama-se Computabilidade [Kfoury82] e a quantidade de recursos envolvidos, Complexidade[8]. Fala-se, também, em Computabilidade Prática. Por exemplo, um problema que requeira um tempo de 100 anos do mais rápido computador disponível não é praticamente computável.

No caso de RNA, pode-se dizer que o problema crucial não reside na "computabilidade" e, sim, na "complexidade". Em RNA, complexidade diz respeito ao número de camadas utilizadas pela rede para resolver um dado problema, bem como ao número de neurônios em cada camada. Os trabalhos de Minsky e Papert [Minsky88] provaram que redes "feedforward" necessitam de camadas intermediárias para solucionar problemas "não linearmente separáveis". Posteriormente, ficou provado que tudo que uma rede pode aprender com n camadas intermediárias pode ser aprendido por uma rede de uma única camada intermediária. O número de neurônios nas camadas de entrada e de saída, normalmente, é função do problema em questão. O problema reside, então, no número de neurônios na camada intermediária: se for um número grande, a rede pode se especializar e perder a capacidade de generalização; se for um número pequeno, a rede pode não aprender. Este tem sido, até o presente momento, um problema em aberto. Heurísticas têm sido utilizadas para a definição deste número.

[6] Nem sempre o erro quadrático é acumulado. Este é acumulado quando o treinamento é por épocas.
[7] Eventualmente, o treinamento pode ser feito por padrão e não por época.
[8] Para uma apresentação destes conceitos em nível bastante claro ver Goldschlager [Goldschlager82].

No entanto, uma solução criativa é a de utilizar AG para a determinação deste número a partir de uma população inicial de redes com diferentes números de neurônios na camada intermediária. Brasil [Brasil99] explorou esta solução obtendo resultados encorajadores.

V. ALGORITMOS GENÉTICOS NO APRENDIZADO DE REDES NEURAIS

Geralmente o treinamento de RNA com múltiplas camadas é realizado por algum método de gradiente, quase sempre uma variante da regra delta, usando o algoritmo de retropropagação para o cálculo de derivadas parciais [Rumelhart86]. Embora esse método seja eficaz, em muitos casos algoritmos de gradiente somente buscam um mínimo local da superfície de erros de uma RNA com múltiplas camadas e, geralmente, tal superfície é extremamente complexa e multimodal [Tanomaru95].

Com o objetivo de proporcionar uma busca mais global, AG podem ser empregados para determinação dos pesos de uma RNA com múltiplas camadas de configuração específica [Kitano90][Montana89][Whitley89]. Nesse caso, uma população de RNA com múltiplas camadas de mesma arquitetura é usada, ou seja, cada RNA com múltiplas camadas com seus pesos constitui um indivíduo. Os pesos podem ser representados por longas seqüências binárias ou números reais. A avaliação de cada indivíduo é feita com base num conjunto de padrões para teste, de modo que a aptidão seja uma função decrescente do erro quadrático.

O AG utilizado para este treinamento pode ser o AG simples, conforme descrito abaixo:
Etapas do Algoritmo:

1 – Criação da população inicial de pesos para uma RNA:

Inicia-se com todas as redes tendo a mesma arquitetura;

Cada indivíduo da população terá um conjunto de pesos sinápticos diferente;

Usa-se, para a criação desta população inicial, a distribuição uniforme ou distribuição Gaussiana;

Codificação da cadeia de cromossomos (fixa). No entanto, neste caso, cada gene é constituído de um "string": como cada gene corresponde a um valor de peso e este é um número real, torna-se necessária a codificação deste número em um "string", conforme mostrado na Figura 9;

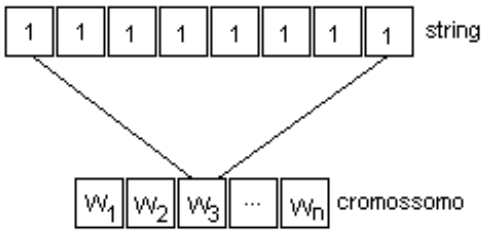

Figura 9 – Representação do cromossomo e dos genes

2 – Avaliação da população e teste de convergência:

Para esta avaliação uma função de aptidão deve ser escolhida. Normalmente esta função é o próprio erro acumulado por época para os padrões de treinamento.

Caso um indivíduo da população atual tenha atingido o erro esperado, considera-se que o algoritmo convergiu para uma solução e o treinamento termina.

3 – A seguir parte-se para a criação de novos indivíduos:
Emprega-se o AG para seleção, “crossover” e mutação.
Determina-se, por conseguinte, a nova geração.
4 – Volta-se ao passo 2.

A seguir pode ser realizada a avaliação da RNA vencedora usando-se um conjunto de padrões de teste que não aquele usado para o treinamento.

Testes conduzidos com problemas simples de reconhecimento de padrões produziram resultados inconclusivos. Em [Kitano90], Kitano mostrou experimentalmente que AG são mais lentos que algoritmos rápidos para treino de RNA com múltiplas camadas, como o “quickprop” [Fahlman88], e propôs usar AG somente para busca global no início do treino, bem como um método de gradiente convencional para busca local. Em resumo a rede seria treinada, inicialmente, pela técnica de AG e, após um certo número de épocas, um ajuste fino seria realizado, utilizando-se um algoritmo como o “backpropagation”, com os pesos iniciais dados pelo melhor indivíduo resultante da aplicação do AG.

Um outro problema diz respeito ao tamanho da RNA. Conforme a quantidade de neurônios que ela possua aumenta, o treinamento por AG pode tornar-se inviável.

De forma a resolver estes problemas, várias técnicas têm sido propostas. A idéia geral destas técnicas é a de se fixar todos os pesos a exceção de um. A seguir se treina este peso para todos os indivíduos da população, via AG, até se chegar ao que poderia ser considerado o melhor valor para esse peso. Encontrado este valor, ele é fixado, juntamente com todos os outros a exceção do “segundo” peso. Este seria treinado da mesma forma e, assim sucessivamente, até que todos os pesos tenham sido treinados. A esta classe de procedimentos denomina-se de buscas unimodais.

Uma solução criativa, dentre as implementações de busca unimodal, é aquela proposta por [Dias99]. Dias propõe a inclusão de uma Análise de Sensibilidade (AS), no algoritmo, de forma a ser possível determinar a importância de cada peso para o erro de época. No algoritmo implementado por Dias, denominado de Treinamento Sensibilidade Global, a sensibilidade do erro da época em relação a cada peso é avaliada. Assim, todos os pesos são processados individualmente pelo AG, conforme a ordenação dada pela AS. Os resultados obtidos por Dias mostram-se promissores.

VI. ALGORITMOS GENÉTICOS NA OTIMIZAÇÃO DO NÚMERO DE NEURÔNIOS NA CAMADA INTERMEDIÁRIA

Uma outra aplicação interessante de AG é a determinação de boas arquiteturas de RNA com múltiplas camadas para uma determinada tarefa.

Em [Hard89][Hard91], cada indivíduo corresponde a uma RNA, ou seja, o número de camadas, o número de nós por camada e o padrão de conectividade são representados por seqüências binárias e a busca opera, então, no espaço das arquiteturas de RNA com múltiplas camadas. Além disso, os parâmetros de aprendizagem, usando a regra delta,

também foram codificados nos cromossomos. Não somente a representação cromossômica e os OG se tornam relativamente complexos, mas o processo de avaliação de cada indivíduo consome tempo. Para a avaliação, as RNA com múltiplas camadas usando pesos gerados aleatoriamente são construídas para cada indivíduo e treinadas usando-se um conjunto de padrões de treino. Depois disso, cada RNA é avaliada usando-se um outro conjunto de padrões de teste. Somente depois, a aptidão de cada indivíduo é estimada usando-se informação referente à velocidade de convergência, erro de reconhecimento, etc. [Tanomaru95].

Mais recentemente, Brasil [Brasil99] desenvolveu um Sistema Especialista Híbrido para auxiliar na minimização da etapa de Aquisição de Conhecimento de um sistema especialista. Neste trabalho, o AG foi utilizado para auxiliar na escolha da melhor topologia para a RNA. Neste sentido, desenvolveu-se um algoritmo de aprendizado, chamado de "Genetic-Backpropagation Based Learning Algorithm" (GENBACK), inspirado no de Rumelhart et al. [Rumelhart86], mas diferente em vários aspectos: otimização da camada intermediária auxiliada pelo AG, incorporação de conectivos lógicos E/OU no local do somatório de pesos e tratamento das variáveis de entrada por lógica "fuzzy". A seguir é apresentada, resumidamente, as etapas do algoritmo GENBACK generalizado.

Etapas do Algoritmo GENBACK:

1 – Criação da população inicial de RNA:

Inicia-se com todas as redes tendo o mesmo número de neurônios de entrada e de saída;

Cada indivíduo (RNA) da população terá um número diferente de neurônios na camada intermediária;

Usa-se, para a criação desta população inicial, a distribuição uniforme ou distribuição Gaussiana;

Codificação da cadeia de cromossomos (fixa). Foi utilizada a representação binária do código genético como sendo igual a 8 "bits". Representação esta identificando o número de neurônios na camada intermediária da RNA;

2 – Treinamento da RNA: treina-se cada rede utilizando um algoritmo tipo "Backpropagation" para o mesmo número de épocas :

A partir da segunda geração, as redes novas devem ser treinadas para um número de épocas igual ao total de todas as iterações deste algoritmo. Ou seja, uma rede criada na geração número cinco deverá ser treinada cinco vezes o número de épocas estipulado;

3 – Avaliação da melhor topologia de rede:

Para esta avaliação uma função de aptidão deve ser escolhida. No trabalho original de Brasil [Brasil99] foram utilizadas as seguintes funções:

$$F_1 = N_{CH}/ERROR$$

$$F_2 = 1/(N_{CH} + ERROR)$$

onde F_1 e F_2 = função de aptidão ou função objetivo ou função de custo, N_{CH} = número de neurônios na camada intermediária da RNA, *ERROR* = erro na saída da RNA;

4 – A seguir parte-se para a criação de novos indivíduos:

Emprega-se o AG para seleção, "crossover" e mutação.

Determina-se, por conseguinte, a nova geração.

Este processo é repetido até que o conhecimento da solução do ótimo global seja descoberto, ou até algum critério de parada, por exemplo, o número de gerações;

5 – Volta-se ao passo 2.

A seguir pode ser realizada a avaliação da RNA vencedora usando-se um conjunto de padrões de teste que não aquele usado para o treinamento.

VII – PERPESCTIVAS FUTURAS

Uma linha de pesquisa ainda não tão explorada quanto a utilização dos AG para o treinamento propriamente dito de uma RNA e a escolha da melhor topologia de uma RNA, diz respeito a utilização dos AG tanto para o aprendizado da RNA quanto para a otimização da camada intermediária da RNA.

Referências Bibliográficas

[Algarve97] A.S. Algarve. "Simulaçào do Sistema Circulatório com Controle Neuronal Central", *Relatório Interno - GPEB*, UFSC, Florianópolis, Brasil, 1997.

[Angeline93] P.J. Angeline. "Evolucionary Algorithms and Emergent Intelligence". Ph.D. Dissertation, The Ohio State University, USA, 1993.

[Arbib64] M. A. Arbib. Brains, *Machines and Mathematics*, McGraw-Hill, 1964.

[Arbib87] M. A. Arbib. *Brains, Machines and Mathematics*, edition: Segunda, Springer, 1987.

[Austin90] S. Austin. "An introduction to genetic algorithms". *AI Expert*, vol.1, no.1, pp. 48-53, 1990.

[Back97] T. Bäck, U. Hammel, and H. Schwefel. "Evolutionary computation: comments on the history and current state". *IEEE Transactions on Evolutinary Computation*, vol.1, no.1, pp. 3-17, April 1997.

[Barreto90] J. M. Barreto. "Neural Networks Learnig: A New Programming Paradigm", in *ACM International Conference: Trends and Directions in Expert Systems*, Orlando, Florida, October 29 to November 3, 1990.

[Barreto96] J.M. Barreto. "Conexionismo e a Resolução de Problemas", *Titular Professor Contest Dissertation*, Departamento de Informatica e Estatística, UFSC, Florianópolis, SC, 1996.

[Barreto96a] J. M.Barreto. "Redes Neurais: aspectos matemáticos e computacionais", Universidade Federal do Maranhão, Escola de Matemática Computacional (SBMAC-96), Sociedade Brasileira de Matemática Aplicada e Computacional, 141 pg., agosto, 1996.

[Barreto97] J. M. Barreto. *Inteligência Artificial no Limiar do Século XXI*, Florianópolis, SC: DUPLIC – Prestação de Serviços, 1997.

[Brasil99] L.M. Brasil. "Proposta de Arquitetura para Sistema Especialista Híbrido e a Correspondente Metodologia de Aquisição do Conhecimento". Tese de Doutorado, Universidade Federal de Santa Catarina, UFSC, Brasil, 1999.

[Darwin81] C. Darwin. *The Origin of Species*. Fac-Simile da edição original – Charles W. Eliot, L. L.D. – 1981.

[Davidor92] Y. Davidor. "Genetic algorithms: a survey". *Dynamic, Genetic, and Chaotic Programming*, B. Soucek and The IRIS Group (eds.), John Wiley & Sons, Inc., New York, USA, 1992, pp.323-338.

[de Azevedo93] F.M. de Azevedo. "Contribution to the Study of Neural Networks in Dynamical Expert Systems", Ph.D. Thesis, Institut d'Informatique, FUNDP, Belgium, 1993.

[de Azevedo97] F.M. de Azevedo. "Uma Proposta de Modelos Formais de Neurônios e Redes Neurais Artificiais", *Anais do III Congresso Brasileiro de Redes Neurais, IV Escola de Redes Neurais*, Florianópolis, Brasil, 1997.

[de Oliveira99] R. C. L. de Oliveira. "Rede Neural Com Dinâmica Interna Aplicada A Problemas De Identificação E Controle Não-Linear", Tese de Doutorado, Universidade Federal de Santa Catarina (UFSC), Florianópolis, Brasil, 1999.

[Dias99] J. Dias. "Treinamento Híbrido de Redes Neurais para Processamento de Informações Biomédicas". Tese de Doutorado, Universidade Federal de Santa Catarina, UFSC, Brasil, 1999.

[Dunn50] L.C. Dunn and T.H. Dobzhansky. *Hereditary, race and Society*. Mentor Books – The New American Libray, 4a. ed., 1950.

[Fahlman88] S. Fahlman. "An empirical study of learning speed in back-propagation networks". *Technical Report CMU-CS-88-162*, Carnegie Mellon University, 1988.

[Fogel95] D.B. Fogel. *Evolutionary Computation: Toward a New Philosophy of Machine Intelligence*. New York,USA: IEEE Press, 1995.

[Garcia95] R. Garcia, F.M. Azevedo, and J.M. Barreto. "Genetic algorithms in the optimal choice of neural networks for signal processing". *In Proc. 38th. Midwest Symposium on Circuits and Systems*, Rio de Janeiro, Brazil, vol. 2, 13-14 August 1995, pp.1361-1364.

[Garis92] H. Garis. "Artificial embryology: the genetic programming of an artificial embryo", in *Dynamic, Genetic, and Chaotic Programming*, B. Soucek and The IRIS Group (eds.), John Wiley & Sons, Inc., New York, USA, 1992, pp.373-393.

[Goldberg89] D.E. Goldberg. "Genetic Algorithms in Search, Optmization, and Machine Learning". Addison-Wesley Publishing Company, Inc., Reading, Massachussets, 1989.

[Goldschlager82] L. Goldschlager and A. Lister. "Computer science: a modern introduction", New Jersey, USA: Prentice Hall, 1982.

[Guyton76] A. Guyton. *Textbook of Medical Physiology*. Philadelphia, USA: W.B.Sanders Company, 1976.

[Harp89] S.A. Harp. "Towards the genetic synthesis of neural networks", in *Proc Third International Conference Genetic Algorithms*, 1989, pp.360-369.

[Harp91] S.A. Harp and T. Samad. "Genetic synthesis of neural network architecture", in Handbook of Genetic Algorithms, L. Davis (ed.), Van Nostrand Reinhold, 1991, pp.202-221.

[Haykin94] S. Haykin. *Neural Networks: A comprehensive Foundation*", New York, USA: Macmillian College Publishing Company, Inc., 1994.

[Hebb49] D. Hebb. *Organization of Behavior*, New York, USA: John Wiley & Sons, 1949.

[Hebb79] D. Hebb. *Psicologia*, edition:2, Philadelphia, USA: W.B.Saunders Company, 1979.

[Hecht88] R. Hecht-Nielsen. "Neurocomputing: Picking the human brain". *IEEE Spectrum*, vol.25, no.3, pp.36-41, 1988.

[Holland75] J.H. Holland. *Adaptation in Natural and Artificial Systems*, Cambridge, Massachusetts: MIT Press, 1975.

[Hopfield82] J. Hopfield. "Neural Networks and Physical Systems with Emergent Collectives Computational Abilities", in *Proceedings of the National Academy of Sciences*, vol.79,1982, pp.2554-2558.

[Hopfield84] J. Hopfield. "Neurons with Graded Response Have Collective Computational Properties Like Those of Two-State Neurons", in *Proceedings of the National Academy of Sciences*, vol.81, 1984, pp.3088-3092.

[Hunt92] K. Hunt and D. Sbarbaro and R. Zbikowski and P. Gawthrop. "Neural Networks for Control Systems - A Survey", *Automatica*, vol.28, no.6, pp.1083-1112, 1992.

[Jason93] D.J. Jason and J.F. Frenzel. "Training product unit neural networks with genetic algorithms. IEEE Expert, vol.1, no.1, pp.26-33, 1993.

[Kartalopoulos96] S.V. Kartalopoulos. *Understanding Neural Networks and Fuzzy Logic: Basic Concepts and Applications*, New York, USA: IEEE Press, Inc., 1996.

[Karunanithi, Das & Whitley92] N. Karunanithi, R. Das, and D. Whitley. "Genetic cascade learning for neural networks". In L. D. Whitley and J.D. Schaffer, editors, *Proc. of the International Workshop on Combinations of Genetic Algorithms and Neural Networks CONGANN'92*, Los Alamitos, CA, IEEE Computer Society Press, 1992.

[Kfoury82] A. J. Kfoury, R. N. Moll, and M. A. Arbib. *A Programming Approach to Computability*, Springer Verlag, 1982.

[Kitano90] H. Kitano. "Empirical studies on the speed of convergence of neural network training using genetic algorithms", in *Proc. Eight National Conference Artificial Intelligence*, vol.2, 1990, pp.789-795.

[Kosko88] B. Kosko, "Bidirectional associative memories", *IEEE Transactions on Systems, Man, and Cybernetics*, vol.18, no.1, pp.49-60, 1988.

[Kozek93] T. Kozek and T. Roska and L.O. Chua. "Genetic algorithm for CNN template learning", *IEEE Transactions on Circuits and Systems - I: Fundamental Theory and Applications*, vol.40, no.6, pp.392-402, 1993.

[Lawrence92] J. Lawrence. *Introduction to Neural Networks and Expert Systems*, Nevada City, CA: California Scientific Software, 1992.

[Machado92a] R.J. Machado and A.F. Rocha, "A hybrid architecture for fuzzy connectionist expert systems", *in Hybrid Architectures for Intelligent Systems*, A. Kandel and G. Langholz (eds.), CRC Press, Inc., Florida, USA, 1992, pp.135-152.

[Machado92b] R.J. Machado and C. Ferlin and A.F. Rocha. Combining semantic and neural networks in expert systems", CCR-140, IBM Rio Scientific Center , pp. 21, Rio de Janeiro, Brazil, 1992.

[McCulloch43]W.S. McCulloch and W.H. Pitts. "A logical calculus of the ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics*, vol.5, no.1, pp.115-133, 1943.

[McCulloch65] W. S. McCulloch. "Embodiments of Mind", Cambridge, Massachusetts: The MIT press,1965.

[Minsky88] M. L. Minsky and S. A. Papert, "Perceptrons: an Introduction to Computational Geometry", The MIT Press, 3rd impressão (modificada) do original de 1969, 1988.

[Montana89] J. Montana and L. Davis. "Training feedforward neural networks using genetic algorithms". *In Proc. of the Eleventh International Joint Conference on Artificial Intelligence*, San Mateo, CA, 1989, pp.762-767.

[Muhlenbein89] H. Muhlenbein and J. Kindermann. "Parallel genetic algorithms and neural networks as learning machine". In DlJ. Evans, G. R. Joubert, and H. Liddell, editors, *in Proc. of the International Conference Parallel Computing'91*, London, UK, North-Holland, 1991, pp.91-103.

[Muhlenbein91] H. Muhlenbein. "The Dynamics of Evolution and Learning - Towards Genetic Neural Networks". *German National Reserach Center for Computer Science*, 1989.

[Park94] D. Park and A. Kandel. "Genetic-based new fuzzy reasoning models with application to fuzzy control". *IEEE Transactions on Systems, Man, and Cybernetics*, vol.24, no.1, pp.39-47, 1994.

[Parker82] D. Parker. "Learning Logic", *Invention Report*, Office of Technology Licensing, File 1, Stanford University, Stanford, California, pp. S81-64, 1982.

[Porto95] V.W. Porto, D. Fogel, and L. Fogel. "Alternative neural networks training methods". *IEEE Expert*, vol.10, no.3, pp.16-22, June 1995.

[Prado92] D.L. Prado. "New learning algorithm for training multilayered neural networks that uses genetic-algorithm techniques". *Electronics Letters*, vol.28, no.16, pp.1560-1561, 1992.

[Roisenberg98] M. Roisenberg. "Emergência de Inteligência em Agentes Autônomos Através de Modelos Inspirados na Natureza, Tese de Doutorado, Universidade Federal de Santa Catarina (UFSC), Florianópolis, Brasil, 1998.

[Rosenblatt58] F. Rosenblatt. "The Perceptron: a probabilistic model for information storage and organization in the brain". Psychological Review, vol.65, pp.386-408, 1958.

[Rumelhart86] D.E. Rumelhart and G.E. Hinton and R.J. Williams, "Learning internal representations by error propagation", in *Parallel Distributed Processing*, D.E. Rumelhart and J.L. McClelland and The {PDP} Group (eds.), MIT Press, Cambridge, Massachusetts, 1986, vol.1: Foundations, pp.319-362.

[Silva95] A. C. da Silva, C.C. Luiz e A.A.R. Coelho. "Projeto do controlador PID genético: algoritmo e aplicação". In Anais do II Congresso Brasileiro de Redes Neurais, Outubro 1995.

[Soucek91] B. Soucek and The IRIS Group. *Neural and Intelligent Systems Integration: Fifth and Sixth Generation Integrated Reasoning Information Systems*. John Wiley & Sons, Inc., New York, USA, 1991.

[South93] M.C. South and G.B. Wetherill and M.T. Tham. "Hitch-hiker's guide to genetic algorithms". *Journal of Applied Statistics*, vol.20, no.4, pp.153-175, 1993.

[Srinivas94] M. Srinivas and L.M. Patnaik. "Adaptive probabilities of crossover and mutation in genetic algorithm". *IEEE Transactions on Systems, Man, and Cybernetics*, vol.24, no.4, pp.656-667, 1994.

[Tanomaru95] J. Tanomaru. "Motivação, fundamentos e aplicações de algoritmos genéticos". *In Proc. do II Congresso Brasileiro de Redes Neurais*, vol.1, no.3, Curitiba, Brasil, 1995, pp.331-411.

[Vico91] F. J. Vico and F. Sandoval. "Use of genetic algorithms in neural networks definition". In A. Prieto, editor, in *Proc. of the International Workshop on Artificial Neural Networks IWANN'91*, Granada, Spain, Springer Verlag, September 1997, pp.196-203.

[Werbos74] P. Werbos. *Beyond Regression: New Tools for Prediction and Analysis in the Behavioral Sciences*, Harvard University, Cambridge, Massachusetts, 1974.

[Whitley89] D. Whitley. "Cellular genetic algorithms", *in Proc Fifth International Conference of the Genetic Algorithms*, 1993, pp.658.

[Widrow62] B. Widrow. "Generation and information storage in networks of adaline neurons", *in Self-Organizing Systems*, M. Yovits and G. Gacobi and G.Goldstein (eds.), Spartan Books, Washington, D.C., 1962, pp.435-461.