UM MODELO PARALELO PARA O APRENDIZADO EM REDES NEURONAIS BASEADO EM ALGORITMOS GENÉTICOS

Mauricio Solar[1/2]
Sueli B. T. Mendes[2]

[1]USACH: Universidad de Santiago de Chile
Depto. de Ingenieria Informática - Casilla 10233-STGO
Avda. Ecuador 3659 - Estación Central - Santiago de Chile
FAX: (562) 7763511 e-mail: MSOLAR@USACHVM1.BITNET

[2]COPPE/UFRJ
Programa de Engenharia de Sistemas e Computação
Centro de Tecnologia - COPPE - UFRJ - Caixa Postal 68511
Rio de Janeiro - RJ - CEP 21945 - Brasil

RESUMO

Na ciência da computação onde os algoritmos seriais são usualmente paralelisados com a ajuda de truques e contorções, não deixa de ser uma ironia que os Algoritmos Genéticos (AGs) (altamente paralelos) são efetuados em forma serial com os mesmos truques pouco naturais e sem elegância. O artigo apresenta um modelo paralelo para um Algoritmo Genético Conexionista (AGC) mapeado naturalmente sobre uma máquina paralela (iPSC da Intel e Rede de Transputers da INMOS) com bons resultados em termos de tempo de comunicação, tempo de processamento e tempo de resposta. A implementação do modelo AGC mostra que é adequado para tarefas que operam num meio ambiente dinámico. O modelo é baseado nos mecanismos de busca dos AGs e em resultados provenientes das Redes Neuronais Artificiais (RNAs). As propriedades e os parâmetros do AGC permitem ao sistema se adaptar às mudanças do meio ambiente, respondendo adequadamente às situações para as quais não foi treinado. O AG é capaz de superar as limitações de aprendizado automático das RNAs, permitindo alterar o conhecimento na medida que o meio ambiente muda. O AGC apresenta propriedades desejáveis de ambos os campos, ou seja capacidade de generalização das RNAs e capacidade de adaptação dos AGs. O AGC opera com uma familia (população) de RNAs (indivíduos) com arquitetura idêntica, cujo conhecimento armazenado nos pesos é diferente.

1.- INTRODUÇÃO

As Redes Neuronais Artificiais (RNAs) [HOPFIELD, 82; PAO, 89] têm duas fases de operação: aprendizado e consulta. No aprendizado, a RNA constrói representações internas complexas de seu meio ambiente, sendo este um dos maiores objetivos da pesquisa de RNA (obter procedimentos eficientes de aprendizado [MICHALSKI et alii, 86]).

O algoritmo de aprendizado descobre os pesos que fazem a RNA apresentar o comportamento desejado, com base num conjunto de exemplos. O algoritmo deve ser capaz de modificar os pesos das conexões de forma que as unidades internas que não são parte da entrada ou da saída venham a representar aspectos importantes do domínio do problema. O conjunto de exemplos (o qual não é simples de determinar), geralmente não representa todos os padrões da entrada, o qual piora com um meio ambiente dinámico (mais complexo), e na consulta, a RNA não é capaz de responder as mudanças do meio ambiente [MUHLENBEIN, 90]. Esta limitação é produto da dificuldade destas em capturar todos os padrões no aprendizado [HINTON, 89]. Daqui surge a necessidade de dispor de um sistema que possa se adaptar às mudanças do meio ambiente.

Várias propostas têm sido apresentadas como uma solução a este problema. Uma delas é o uso dos Algoritmos Genéticos (AGs) [DeJONG, 80; GREFENSTETTE, 86; HOLLAND, 86; DAVIS, 87; BOOKER et alii, 89; GOLDBERG, 89;], os quais têm mostrado ser uma boa alternativa pela sua flexibilidade na adaptação. Estes AGs "puros" não possuem um mecanismo de aprendizado eficiente que oriente a busca, dado que sua forma de avaliar as soluções (função objetivo) é complexa de obter. Outro problema que apresentam estes AGs "puros" é o tamanho inicial da população [GOLDBERG, 85], o qual chega a ser tão grande, que é inviável a sua implementação pelo espaço de memória e tempo de processamento necessários [ROBERTSON, 88].

Uma das soluções proposta em SOLAR (92) e apresentada neste trabalho é um modelo de Algoritmo Genético Conexionista (AGC) [OOSTHUIZEN, 87; MONTANA & DAVIS, 89; WHITLEY et alii, 90; PALAGI, 91; MACHADO et alii, 92] que combina ambas as técnicas, onde é possível modificar os pesos das RNAs de tal forma que o conhecimento representado nas conexões, seja alterado na medida que o meio ambiente exija. Isto permite herdar as propriedades desejáveis de ambas as técnicas.

Muitos componentes do modelo do AGC apresentado, incluindo as RNAs, ajuste dos pesos e mecanismos de aprendizado são inherentemente paralelos [HOLLAND, 86; PETTEY et alii, 87; ROBERTSON, 87]. Isto indica que uma implementação paralela deste modelo pode ser mapeado em forma natural sobre uma arquitetura paralela.

Uma questão importante que está em aberto no estudo dos AGs é o tamanho inicial ótimo da população. O problema está no compromiso entre o montante da busca genética que pode ser efetuada e o montante disponível de tempo real. Se a população é muito pequena, então é possível que o AG não ache uma boa solução pela insuficiencia de esquemas na população [BOOKER, 87]. Se a população é muito grande, é possível que o tempo usado para efetuar todas as avaliações seja inaceitável [ROBERTSON, 88]. Uma implementação paralela do AGC pode resolver o compromiso entre busca genética e tempo real. O AGC paralelo opera sobre uma grande população de vários grupos de indivíduos distribuidos num multiprocessador.

BETHKE (81) calculou várias estimações de complexidade para um mapeamento particular de AGs em máquinas paralelas. Conclui que o cálculo da adaptabilidade da média da população é o principal gargalo serial em implementações de AGs desses tempos. Mas não simulou nem implementou um AG paralelo. GR\EFENSTETTE (81) examinou várias implementações paralelas de AGs, classificando quatro protótipos: (1) Mestre-Servidor síncrono; (2) Mestre-Servidor semi-sincrono; (3) Concorrência assincrona distribuída; e (4) Redes.

HOLLAND (86) mostra a natureza paralela do paradigma reprodutivo e a eficiência inherente do processamento paralelo. Mostra um tipo de computador celular para mapear a reprodução.

RIOLO (86) introduz o primeiro AG em aprendizado (Sistema de Classificação), o qual ROBERTSON (87) implementou na "Connection Machine" [HILLIS, 86]. PETTEY et alii (87) apresentam um AG paralelo que consiste de um grupo de nós de AG idênticos. Cada nó do AG mantêm uma pequena população, e em cada geração, cada nó se comunica com os seus vizinhos. Esta comunicação consiste em enviar os melhores indivíduos da população local para os vizinhos e recever os melhores indivíduos de cada vizinho, inserindo os novos indivíduos na população local.

GOLDBERG (89) propõe dois procedimentos paralelos orientados a objetos para os AGs: a "Comunidade" e a "Planta de Polinização". Sendo cada subprocesso um objeto, é simples isolar o processamento, o requerimento de memória e a largura de banda da comunicação requerida para cada objeto. Esta perspectiva pode permitir um mapeamento mais eficiente dos AGs em computadores paralelos.

O trabalho define os elementos básicos do AGC, suas características, princípios e parámetros. Depois mostra as características do modelo paralelo em termos dos parámetros do modelo, tais como tamanho da população do AG, ordem da RNA e comunicação necessária. No fim apresenta as conclusões do modelo paralelo implementado no hipercubo iPSC da Intel e numa Rede de Transputers da INMOS.

2.- CONCEITOS GERAIS

O modelo foi implementado num sistema de multiprocessamento com memória distribuída, sob um esquema de computação MIMD [SOUCEK & SOUCEK, 88]. As memórias e processadores estão conectados por uma rede de interconexão ponto-a-ponto; descrita em termos de nós e enlaces (do tipo "full-duplex"). Cada processador pode usar todos os seus enlaces ao mesmo tempo (comunicação limitada-pelo-enlace), porque é o número de enlaces os que limitam a comunicação [seleção do sistema multiprocessador aparece em SOLAR (92)].

Os multiprocessadores que compartilham memória têm dificuldade no acesso concorrente à memória. No entanto, se a memória é também distribuída, a transferência de dados depende da topologia, para o qual é necessário escolher a topologia. O esquema MIMD permite diferentes tipos de topologias a serem usadas. Na escolha da topologia qualquer decisão é um compromiso entre custo e desempenho [HILLIS, 86], pelo qual tem-se usado diferentes tipos de topologías: hipercubo, anel, grade, etc. O modelo apresentado baseia-se na topologia de Hipercubo, na qual é facil mapear outras topologias, tais como uma grade, uma árvore, etc.

Os processadores do hipercubo se comunicam pela troca de mensagens, mas o esquema de comunicação depende do problema. No modelo desenvolvido são usados os esquemas de comunicação de Difusão e Convergência [DINNING, 89] para sinconizar os processos. O tempo de comunicação T para enviar uma mensagem desde um processador para um vizinho é modelado como constante. Muitas pesquisas mostram que o custo elemental T depende do comprimento L da mensagem. No AGC desenvolvido a grande maioria das mensagens são curtas (1 byte), para o qual se usou o modelo constante (T=1) ao inves do modelo linear. No modelo constante as mensagens podem ser particionadas e recombinadas sem mudar o tempo de propagação.

2.1.- Arquitetura do Hipercubo H_d [FOX, 88]

Um hipercubo H_d é um grafo com $N=2^d$ vértices. Cada vértice é marcado por um número binário de d-bits. As arestas estão entre dois vértices cuja marca difere em precisamente um bit. O hipercubo pode ser definido recursivamente como: um hipercubo de 1-dimensão é uma aresta com um vértice marcado 0 e outro vértice marcado com 1. Um hipercubo de (d+1)-dimensões é construido apartir de dois hipercubos de d-dimensão, H_d^0 e H_d^1, pela soma de arestas desde cada vértice em H_d^0 para vértices em H_d^1 que possuem a mesma marca e logo prefixando todas as marcas em H_d^0 com um 0 e todas as marcas em H_d^1 com um 1. O hipercubo é usado em vários computadores paralelos (iPSC da INTEL) por seu pequeno diámetro $D=d=\log_2 N$ e pelo seu pequeno grau $\triangle=d=\log_2 N$. Desafortunadamente, possui $(N \log_2 N)/2$ arestas.

3.- IMPLEMENTAÇÃO PARALELA DO ALGORITMO GENÉTICO CONEXIONISTA

No modelo AGC tem-se N ($N=2^d$) AGs locais e independentes com memória distribuida, operações genéticas e funções objetivos independentes. Os N processos operam normalmente, com a exceção que os melhores indivíduos de cada geração são difundidos para as outras subpopulações através da rede de comunicação. Com uma comunicação relativamente intermitente, o largo de banda das ligações é reducido se comparado com outros esquemas. A confiança deste modelo é alta, pela autonomia dos processos independentes.

3.1.- Suposições do Modelo AGC

O AG opera com uma população de indivíduos, que representa as soluções do problema no qual está imerso. O modelo de AGC tem como população uma familia de RNA com uma arquitetura fixa. Cada RNA é representada como uma cadeia, cujos componentes correspondem aos pesos das conexões entre os neurônios. O AGC opera sobre esta população escolhendo aqueles que melhor se adaptam à entrada proveniente do meio ambiente, gerando uma saída. Se na população de RNA não existir nenhuma RNA que responda adequadamente, então os operadores genéticos combinam algumas destas RNAs para obter novas soluções (com idêntica arquitetura, mas com pesos diferentes) e elimina os piores individuos da população (substituição elitista). O modelo do AGC se concentra nas seguintes suposições:

- O meio ambiente, as entradas e saídas podem ser representadas por uma RNA de três camadas: nós de entrada, nós escondidos e nós de saída. Cada RNA corresponde a uma cadeia de símbolos de um comprimento fixo (L bits) sobre um alfabeto A. Este alfabeto binário é {0,1}. No modelo de AGC cada cadeia representa uma arquitetura de RNA com seus pesos;
- Cada ponto no espaço do problema (espaço total: $N=2^L*2$) pode ser considerado como uma combinação de pesos de uma RNA representado unicamente dentro do sistema por uma cadeia gerada do alfabeto A. Nesta cadeia (cromossoma) do modelo de AGC cada posição específica (locus) representa um bit da conexão da RNA, e seus valores (alelos) indicam um bit do peso de uma conexão;
- O sistema mantém uma população P(t) fixa de RNAs em cada processador representando as soluções atuais do problema;
- O tempo é medido em intervalos discretos chamado gerações.

O AGC conhece as regras para executar a tarefa específica, a qual é definida por exemplos apresentados ao sistema, e a função objetivo determina o desempenho do sistema. Cada RNA tem associado um peso (dado pela função objetivo) que indica a eficiência dela em executar a tarefa desejada.

3.2.- Princípios do Modelo AGC

O AGC usa a familia de RNAs como representação, e a estrutura do conceito é modelada pela organização, variabilidade e distribuição dos pesos entre os neurônios de cada RNA. O modelo se baseia nos seguintes princípios:

Codificação do Cromossoma (Indivíduos): Um indivíduo é uma RNA com uma topología definida e com conhecimento armazenado nos pesos das conexões. Uma conexão da RNA é codificado como um número real de 8 bits (-12.7 até 12.8). Cada indivíduo do AGC é composto por uma cadeia de comprimento fixo (L), onde a cadeia representa as conexões entre os nós de entrada, os nós de saída e os nós escondidos da RNA. Uma posição nesta cadeia (*locus*) corresponde a um bit da conexão entre dois neurônios (nó). O valor armazenado nesta posição (*alelo*) é um número binário tomado do alfabeto {0,1} que indica um bit do peso da conexão.

Uma representação cromossómica que obedeça à hipotese dos blocos de armar garante a busca do ótimo da solução [BOOKER et alii, 89]. Os blocos de sinapses que têm efeito sobre um único neurônio da camada de saída devem ser contiguos aos outros blocos que tem influencia sobre o mesmo neurônio. Isto indica que a estrutura deve conter contiguamente todos os blocos importantes ao aprendizado de cada neurônio de saída.

Seja uma RNA do tipo "feedforward" com três camadas, onde a camada de entrada possui l neurônios N_i, $1<=i<=l$, a camada escondida possui m neurônios N_j, $1<=j<=m$, e a camada de saída possue n neurônios N_k, $1<=k<=n$.

Um bloco de sinapses é uma estrutura criada com o objetivo de reunir coerentemente as sinapses que influenciam um determinado neurônio de saída. Define-se o bloco de sinapses que influencia o neurônio N_k da camada de saída conectado ao neurônio N_j da camada intermediária pela lista resultante da concatenação da sinapse que une N_j a N_k com todas as sinapses que convergem para N_j [PALAGI, 91]:

$$\text{Bloco}(k,j) = [w_{kj}, [w_{ji}]_{i=1}^{t}]$$

onde o simbolo $[w_{ji}]_{i=1}^{t}$ significa a concatenação dos valores A_i, $1<=i<t$.

A escolha de uma RNA "feedforward" se baseia principalmente na existência do Teorema de KOLMOGOROV sobre o Mapeamento em Redes Neuronais, mostrado a seguir e cuja prova está em HECHT-NIELSEN (87):

TEOREMA DE KOLMOGOROV: Dada qualquer função contínua $f:[0,1]^n \longrightarrow R^m$, $f(x)=y$, f pode ser implementada exatamente por uma RNA "feedforward" de três camadas, tendo n elementos de processamento na primeira camada (x-entrada), (2n+1) elementos de processamento na camada intermediária, e m elementos de processamento na última camada (y-saída) [HECHT-NIELSEN, 90].

A existência deste teorema indica que a representação escolhida além de cumprir a teoria dos blocos de armar, permite mapear qualquer função contínua sobre a arquitetura de RNA no cromossoma do AGC. O comprimento L do cromossoma é dado por:

$$L = (n * m * (1 + 1) * 8) \text{ bits}$$

Segundo o comprimento do cromossoma binário é possível determinar o tamanho, ou melhor a complexidade do problema como:

$$N = 2^L * 2$$

Função Objetivo: A função objetivo para o modelo AGC deve guiar o aprendizado da RNA na direção do erro mínimo das suas saídas, quando existe um reforço que vem do meio ambiente externo. Para possibilitar a solução do aprendizado com AGs faz-se necessário definir uma função objetivo de forma que seu máximo represente o menor erro de aprendizado. Pode-se definir o erro de aprendizado de uma RNA como o erro médio quadrático entre a resposta efetiva (o) e a resposta desejada (t) de cada neurônio N_k da camada de saída somado para cada um dos v padrões a serem aprendidos [PALAGI, 91]. Ou seja:

$$E = \sum_{p=1}^{v} \sum_{k=1}^{n} 1/2(t_{pk} - o_{pk})^2$$

Para normalizar o erro, pode-se dividir pelo máximo erro possível nv/2 do aprendizado dos v padrões, obtendo-se o erro médio, sendo a média tomada sobre os n neurônios e sobre os v padrões:

$$E_N = 2E/nv$$

A função objetivo deve maximizar a solução, para o qual se usa o inverso do erro de aprendizado (o qual deve ser minimizado), estabelecida por:

$$f(E_N) = 1/(E_N)^c$$

onde c tem o papel de controlar a intensidade desta relação.

Heurísticas de Aprendizado: No AGC a adaptação de um conjunto fixo de RNAs é feito para que elas respondam melhor à tarefa desejada. Isto é efetuado pelo AG criando novas RNAs da mesma arquitetura (e pesos diferentes) e é executado pelos mecanismos de heurística dos Operadores Genéticos.

3.3.- Parâmetros do Modelo de AGC

É possível elaborar vários tipos de AGC envolvendo variações, tais como probabilidade de aplicar os operadores genéticos, tamanho fixo ou variável da população, percentagem da população sujeita a mudança, etc. A seguir são apresentadas as características dos parâmetros do AGC [GREFENSTETTE, 86]:

Tamanho da População N (TP): GOLDBERG (85) desenvolveu uma teoría para o ótimo do TP para AGs com código binário baseada no comprimento do cromossoma. Para um cromossoma com L=27, o TP ótimo é 77. Um cromossoma com L=27 representa uma RNA extremamente pequena (menor que uma RNA de 1X2X1 neurônios). Com um L=60, o qual ainda representa uma RNA muito pequena (2x2x2), o TP ótimo é 10.200, que é muito grande. Para uma RNA de tamanho médio é necessário um cromossoma de L=500, cujo TP é excessivamente grande, o qual não permitiria um processamento em tempo real. A teoría de GOLDBERG foi desenvolvida baseada num AG puro, ou seja o problema a ser resolvido é único. No caso das RNAs a quantidade de subproblemas a serem resolvidos depende da quantidade de padrões a serem apresentados à RNA. Portanto, por cada padrão novo, o TP ótimo deve incrementarse proporcionalmente. Se se desejam ensinar 4 padrões numa RNA com L=60, o TP ótimo é 4*10.200=40.800, o qual piora ainda mais o processamento em tempo real.

Baseado no trabalho de ROBERTSON (88), no modelo paralelo do AGC, é possível mostrar o seguinte: no modelo sequencial do AGC um indivíduo é selecionado baseado no seu desempenho na população completa, mas no modelo paralelo um indivíduo é selecionado baseado no seu desempenho na população local. Esta diferença mostra três aspectos interessantes de antecipar: (1) O modelo paralelo do AGC é independente do TP, ocupando pouco espaço de memória; (2) A convergência do modelo é mais rápida, permitindo sua aplicação em tempo real; (3) Produz melhores resultados, dado que intercambiam os melhores indivíduos locais entre as diferentes populações.

O modelo implementado usou um TP=50 indivíduos, independente da quantidade de processadores usados.

Procedimento de Inicialização: Os pesos das RNAs da população inicial são escolhidos em forma aleatória com uma distribuição de probabilidade dada por $e^{-|x|}$. Isto é diferente da distribuição de probabilidades inicial dos pesos usualmente usados em "backpropagation", a qual é uma distribuição uniforme entre 0.0 e 1.0. A função de distribuição de probabilidades usada é o refleto das observações empíricas dos pesquisadores de que as soluções ótimas tem tendência de conter pesos com valores absolutos arbitrariamente grandes [MONTANA e DAVIS, 89].

Escolha dos Pais: A escolha dos pais é feita a partir da população ordenada em forma ascendente (proporcional à função objetivo). De acordo com o valor da função objetivo são escolhidos pares de pais em forma probabilistica para gerarem novos filhos. Aquelas RNAs com melhor desempenho têm maior probabilidade de serem selecionadas como pais da seguinte geração.

Probabilidade do Operador

Razão de Cruzamento C (RC): Em cada população nova, são cruzados C*N indivíduos. No AGC se efetuaron varios testes com a RC, usando os valores 0.5 até 0.9 com incremento de 0.1;

Razão de Mutação M (RM): ocorrem aproximadamente M*N*L mutações por geração. O valor de M no AGC foi escolhido empiricamente igual a 0.04;

Intervalo de Geração G (IG): a percentagem da população que será substituida em cada geração, N*G indivíduos de P(t), é variável dependendo da adaptabilidade da população. A adaptabilidade é determinada pela soma do valor da função objetivo para todas as RNAs, não excedendo o 50% da população;

Fator de Graduação W (FG): No AGC é importante manter a diversidade da população, para o qual o FG usado pode chegar até 0.5 da população sem perda de informação;

Estratégia de Seleção S (E$): O AGC usa a ES elitista; primeiro efetua uma ES pura e logo os com melhor desempenho são escolhidos para sobreviver na próxima geração, permitindo modificá-los por cruzamento, mutação ou erro de mostra.

3.4.- Operadores Genéticos

Com o objetivo de estudar o comportamento dos diferentes operadores nas diferentes situações, foram usados um grande número de operadores genéticos, selecionando o melhor conjunto de operadores para a versão final do modelo. Os operadores podem ser agrupados em duas categorias básicas: mutação e cruzamento. As variações do operador de mutação são: mutação tendenciosa, mutação não tendenciosa, mutação de nós e mutação dos nós fracos. As variações do operador de cruzamento são: cruzamento dos pesos e dos esquemas. A seguir descreve-se as variantes dos operadores que foram usados:

Mutação tendenciosa: para cada entrada do cromossoma, o operador substitui com uma probabilidade fixa p=0.04 um valor aleatório escolhido da distribuição de probabilidades de inicialização;

Mutação não tendenciosa: para cada entrada do cromossoma, o operador soma com uma probabilidade fixa p=0.02 um valor aleatório escolhido da distribuição de probabilidades de

inicialização. A mutação tendenciosa é melhor que a não tendenciosa pela seguinte razão: os pais são escolhidos com a tendencia de serem melhores que a média, portanto, os pesos dos pais têm tendencia de serem melhores que uma escolha aleatória. Usando uma distribuição de probabilidades que tem tendência pelos valores presentes dos pesos dá melhores resultados que uma distribuição de probabilidades centrada em zero;
Mutação de nós: este operador seleciona n nós de não entrada da RNA. Para cada conexão de entrada nestes nós, o operador soma ao peso da conexão um valor aleatório tomado da distribuição de probabilidades de inicialização. Então codifica a nova RNA no cromossoma filho;
Mutação dos nós fracos: o conceito de peso dos nós é diferente do conceito de erro usado em "backpropagation". Por exemplo, um nó pode ter erro zero se todas as suas conexões de saída estão em zero, mas este nó não está contribuindo nada positivo para a RNA e não é um nó forte. A definição de peso de um nó escondido numa RNA "feedforward" é a diferença entre a avaliação da RNA intacta e a avaliação da RNA com aquele nó lobotomizado (com suas saídas em zero).

O operador de mutação de nós fracos calcula os pesos de cada nó escondido de uma RNA. Logo seleciona os m nós mais fracos e realiza a mutação em cada uma das suas conexões de saída e de entrada. Esta mutação não é tendenciosa se o peso do nó é negativo e é se o peso do nó é positivo. Logo codifica a nova RNA no cromossoma como um filho.

Cruzamento dos pesos: este operador coloca um valor dentro de cada posição do cromossoma filho selecionando aleatoriamente um dos pais e usando o valor na mesma posição que o cromossoma pai;
Cruzamento dos esquemas: os diferentes nós numa RNA tem diferentes papeis. Para uma RNA com camadas totalmente interligada, o papel de um nó dado pode depender só da camada onde se encontra e não da posição dentro da camada. De fato, pode-se trocar o papel de dois nós A e B na mesma camada de uma RNA como segue: passa por todos os nós C ligados (pela entrada ou saída) com A (portanto com B). Intercambiar os pesos das conexões entre C e A com aquele da conexão entre C e B. Ignorando a estrutura interna, a nova RNA é idêntica com a anterior (dada uma entrada, a saída é exatamente a mesma).

Os filhos produzidos pelos operadores de cruzamento acima mencionados afeta grandemente as estruturas internas dos pais. O operador de cruzamento de esquemas reduz esta dependencia da estrutura interna fazendo o seguinte: para cada nó no primeiro pai, tenta achar um nó no segundo pai, o qual tem o mesmo papel observando o número de entradas de ambas as RNAs e comparando a resposta dos diferentes nós. Então combina o segundo pai de tal forma que os nós que tem o mesmo papel estejam na mesma posição.

4.- OPERAÇÃO PARALELA do Modelo AGC

O modelo AGC foi implementado na linguagem C para o hipercubo iPSC [INTEL CORP., 88], e na linguagem OCCAM para uma rede Hipercubica baseada em Transputers. Ambas as implementações deram bons resultados em termos de tempo de comunicação, sendo que na Rede de Transputers é necessário realizar o controle das mensagens enviadas entre os processadores. No iPSC é possível explorar as primitivas de comunicação proporcionadas pela máquina, tendo-se uma implementação mais simples de sincronizar. A operação do modelo é resumida no seguinte algoritmo:

```
ALGORITMO AGC;
INICIO
  Inicialização; (O Mestre inicializa cada nó Servidor)
  REPETIR
    Recebe Entrada; (O Mestre envia a Entrada aos Servidores)
    Transmite Entrada para os AGs Locais;
    REPETIR em paralelo para cada AG Local
      PARA cada RNA REALIZAR
       INICIO
         Dominio AG --> Dominio de RNA;
         Testar RNA com Padrão de Entrada;
         Obter Erro Quadratico Médio;
         Obter Valor da Função Objetivo;
       FIM;
      Ordenar População segundo Desempenho;
      Aplicar Operadores Genéticos;
      Individuo de Melhor Desempenho --> Piores Individuos;
      Transmitir Melhor Individuo ao Processo Mestre;
      Receber Melhor Individuo do Processo Mestre;
      Inserir Individuo recebedo na População Local;
    ATÉ Convergência Local;
    Aplica Saída;
  ATÉ Fim do Processo;
FIM.
```

A Inicialização consiste em adquirir os dados iniciais do problema e inicializar cada população local do AGC. Segundo a ordem da RNA definida pelo usuario é calculado o comprimento do cromossoma e sua representação no domínio do AGC no processo Mestre, o qual envia os dados de inicialização para os processos locais. O Mestre adquire os dados do meio ambiente (o processo indica uma referência que determina a direção do controle). A entrada é considerada como um padrão que a RNA deve aprender. Logo o Mestre envia a entrada para os processos locais (Recebe Entrada).

Dado que o AG opera com uma representação cromossómica da RNA é necessário leva-la para o dominio das RNAs, no qual se realiza a consulta da RNA para verificar o erro médio desta com os padrões apresentados. Este valor

indicará a ordem da função objetivo de cada RNA. Uma vez realizada a consulta das RNAs e já calculada a função objetivo de cada indivíduo, é feita uma ordem descendente (do melhor até o pior) da população segundo seu desempenho. Isto é realizado para selecionar com maior probabilidade aqueles individuos de melhor desempenho.

São aplicados os operadores genéticos sobre cada população local ordenada, gerándose uma nova geração da qual seram escolhidos os de melhor desempenho para substituir os de pior desempenho da geração anterior.

Caso nenhum AG local atinja a convergência, cada AG envia a sua melhor RNA para o Mestre, quem seleciona o melhor de todos e o difunde para todos os nós. Uma vez atingida qualquer convergência local, é aplicada a saída determinada pela RNA obtida como ótima na solução do problema.

5.- CONCLUSÕES E COMENTÁRIOS DO MODELO PARALELO PARA O AGC

A técnica de backpropagation é util nos problemas com treinamento simples. Quando é incrementada a complexidade do problema, o desempenho do backpropagation cai rapidamente. A técnica de busca pelo gradiente tende a chegar e ficar estagnada num mínimo local, do qual o backpropagation com um ganho suficiente pode fugir, mas sem conhecer se o próximo mínimo é melhor ou pior. Dado que o AGC é modelado usando as representações que codificam as soluções mais satisfatórias do problema respeitando a hipótese de armar blocos, os operadores geram melhores filhos dos bons pais, convergendo finalmente para resultados muito perto do ótimo global. Os resultados obtidos com o AGC foram muito bons enquanto a adaptabilidade [SOLAR, 92].

O sucesso de sistemas com meio-ambientes dinámicos sempre requerem de soluções adaptativas. O modelo de AGC tenta resolver os problemas do domínio pela acumulação de conhecimento do problema e usa esta informação para gerar soluções aceitáveis. Estas soluções nem sempre são ótimas, mas geralmente são satisfatórias. Alguns desafios típicos aparecem nas áreas de configuração de sistemas complexos, tais como alocação de tarefas (algoritmo de escalonamento), seleção de rota (problema do caixeiro viajante), e controle de processos. O AGC é uma solução adaptativa que direciona este desafio a modificar a base de conhecimento dinamicamente em resposta às variações do processo [GOLDBERG, 89].

Uma vantagem do modelo de AGC apresentado é a sua característica de operar com um tamanho fixo de população, que reduz a quantidade de memória necessária na implementação. O AGC opera sobre uma grande população de vários grupos de indivíduos distribuidos. A implementação

paralela do AGC resolve o compromiso entre busca genética e tempo real.

Na medida que as mudanças ocorrem, as RNAs são constantemente alteradas, as transformações são continuamente disparadas, e são formados os novos esquemas, dando como resultado uma RNA de "auto-aprendizado".

O AGC têm duas características inherentes que o faz altamente apropriado para o aprendizado por descoberta: mantem uma variabilidade suficientemente alta para prevenir a convergência para ótimos locais; e a categorização e recombinação produz um alto paralelismo implícito.

Da implementação na Rede de Transputers e no hipercubo iPSC da INTEL do modelo °AGC pode-se dizer que verifica que o paralelismo inherente é totalmente explorado em máquinas com baixa granularidade, mostrado pela independencia do tempo com o tamanho da população [pela falta de espaço não é possível inserir as curvas obtidas em SOLAR (92)].

Os testes feitos com problemas difíceis para o aprendizado, tais como OU exclusivo com múltiplas entradas, deteção de inversão e simetría deram excelentes resultados, mostrando a viabilidade do uso de técnicas de computação paralela nos AGs para o aprendizado em RNAs.

Uma pesquisa futura interessante de efetuar é propor uma metodologia de desenvolvimento de AGC para sistemas de aprendizado em tempo real. Isto permitiria desenvolver uma ferramenta de Meta-Algoritmos Genéticos, onde o usuário pode definir as entradas do meio ambiente (detectores), as saídas para o meio ambiente (efetuadores), e eventualmente a função objetivo para determinar o desempenho do sistema no meio ambiente.

REFERÊNCIAS BIBLIOGRÁFICAS

BETHKE, A. (81), Genetic Algorithms as Function Optimizers, *Ph.D. Dissertation*, Dept. of Computer Science, Vanderbilt Univ., Nashville, TN.

BOOKER, L. (87), Improving Search in Genetic Algorithms, in: L. DAVIS (Ed.), *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann, Los Altos, CA, 61-73.

BOOKER, L.; D. GOLDBERG & J. HOLLAND (89), Classifier Systems and Genetic Algorithms, *Artificial Intelligence*, Vol. 40, Sept., 235-282.

DAVIS, L. (87), *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann, Los Altos, CA.

DeJONG, K. (80), Adaptative System Design: A Genetic Approach, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. SMC-10, Num. 9, Sept., 566-574.

DINNING, A. (89), A Survey of Synchronization Methods for Parallel Computers, *IEEE Computer*, July, 66-76.

FOX, G. (88), *THE THIRD CONFERENCE ON Hypercube Concurrent Computers and Applications, Vol. 1 & 2. Architecture, Software, Computer Systems and General Issues*, ACM, California Inst. of Technology.

GOLDBERG, D. (85), Optimal Initial Population Size for Binary-Coded Genetic Algorithms, *Technical Report TCGA No. 85001*, The Clearinghouse for Genetic Algorithms, Univ. of Alabama.

GOLDBERG, D. (89), *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley.

GREFENSTETTE, J. (86), Optimization of Control Parameters for Genetic Algorithms, *IEEE Trans. on Systems, Man and Cybernetics*, Vol. SMC-16, Num. 1, Jan.-Feb., 122-128.

HECHT-NIELSEN, R. (87), Counterpropagation Networks, *Proc. of the Int. Conf. on Neural Networks*, II, 19-32, IEEE Press, New York, June.

HECHT-NIELSEN, R. (90), *Neurocomputing*, Addison-Wesley.

HILLIS, W. (86), *The Connection Machine*, Cambridge, MA, MIT Press.

HINTON, G. (89), Connectionist Learning Procedures, *Artificial Intelligence*, Vol. 40, Sept., 185-234.

HOLLAND, J. (86), Escaping Brittleness: The Possibilities of General Purpose Learning Algorithms applied to Parallel Ruled-Based Systems, in: R. MICHALSKI et alii (Eds), *Machine Learning: An Artificial Intelligence Approach. Vol. II*, Morgan Kaufmann, Los Altos, CA, 593-623.

HOPFIELD, J. (82), Neural Networks and Physical Systems with Emergent Collective Computational Abilities, *Proc. of the Nat. Ac. of Sc. USA*, Vol. 79, 2554-2558.

INTEL CORPORATION (88) *iPSC/2 User's Guide*, Intel Scientific Computers.

MACHADO, R.; C. FERLIN & A. ROCHA (92), Combining Semantic and Neural Networks in Expert Systems, *Relatório Técnico CCR-140 IBM Centro Científico Rio*, Rio de Janeiro, Brasil.

MICHALSKI, R.; J. CARBONELL & T. MITCHELL (86), *Machine Learning: An Artificial Intelligence Approach. Vol. II*, Morgan Kaufmann, Los Altos, CA.

MONTANA, D. & L. DAVIS (89), Training Feedforward Neural Networks using Genetic Algorithms, *Proc. of IJCAI-89*, Detroit, 762-767.

MUHLENBEIN, H. (90), Limitation of Multi-Layer Perceptrons Networks-Step towards Genetic Neural Networks, *Parallel Computing* 14, 249-260.

OOSTHUIZEN, G.D. (87), SUPERGRAN: A Connectionist Approach to Learning, Integrating Genetic Algorithms and Graph Induction, Dept. of Computer Science, Univ. of Strathclyde, Scotland, 132-139.

PALAGI, P. (91), Algoritmos Genéticos no Aprendizado de Redes Neuronais, *Tese M.Sc. do Instituto de Ciências Exatas, Universidade de Brasilia*, Julho, Brasilia-DF, Brasil.

PETTEY, C.; M. LEUZE & J. GREFENSTETTE (87), A Parallel Genetic Algorithm, *Vanderbilt Univ.*, Nashville, TN 37235, 155-161.

PAO, Y. (89), *Adaptive Pattern Recognition and Neural Networks*, Addison-Wesley, Reading, MA.

RIOLO, R. (86), CFS-C: A Package of Domain Independent Subroutines for Implementing Classifier Systems in Arbitrary, user-defined Environments, *Logic of Computers Group*, Division of Computer Science and Engineering, Univ. of Michigan, Ann Arbor.

ROBERTSON, G. (87), Parallel Implementation of Genetic Algorithms in a Classifier System, in: L. DAVIS (Ed.), *Genetic Algorithms and Simulated Annealing*, Morgan Kaufmann, Los Altos, CA, 129-140.

ROBERTSON, G. (88), Population Size in Classifier Systems, in J. LAIRD (Ed.), *Proc. of the Fifth Int. Conf. on Machine Learning*, Univ. of Michigan, Ann Arbor, 142-152.

SOLAR, M. (92), Modelos Paralelos para o Aprendizado em Redes Neuronais usando Algoritmos Genéticos, *Tese D.Sc. da COPPE/UFRJ*, a ser defendida em 1992, Rio de Janeiro-RJ, Brasil.

SOUCEK, B. & M. SOUCEK (88), *Neural and Massively Parallel Computers. The Sixth Generation*, Fleet Pub.

WHITLEY, D.; T. STARKWEATHER & C. BOGART (90), Genetic Algorithms and Neural Networks, *Parallel Computing 14*, 347-361.