Proceedings of the VI Brazilian Conference on Neural Networks - VI Congresso Brasileiro de Redes Neurais
pp. 331-335, June 2–5, 2003 – Centro Universitário da FEI, São Paulo - SP - Brazil

Algoritmo Genético Paralelo com Diferentes Comportamentos Populacionais para o Treinamento de Perceptrons de Múltiplas Camadas

Ana Claudia M. L. Albuquerque, Jorge D. de Melo, Adrião D. Dória Neto
Departamento de Engenharia de Computação e Automação
Universidade Federal do Rio Grande do Norte
Campus Universitário s/n - 59072-970 - Natal - RN - Brazil
ana_c@engcomp.ufrn.br, jdmelo@dca.ufrn.br, adriao@dca.ufrn.br

Abstract

In this paper, a parallel genetic algorithm used in the learning process of a Multilayer Perceptron network is presented. An analysis of the different populational behavior is shown, and aspects such as task definition and communications profile were taken into account. An application in approximation of functions illustrates the capacity of these new procedures when compared to the classical approach and to other parallel implementations.

1. Introdução

Ao longo dos anos, as redes neurais Perceptron de Múltiplas Camadas foram propostas para a solução de vários problemas utilizando-se o algoritmo de treinamento por retropropagação de erro (*back-propagation*) [1], [2] e [3]. Neste trabalho, no entanto, ao invés de utilizar o algoritmo clássico de retropropagação de erro no processo de aprendizagem da rede neural, foi desenvolvida uma abordagem diferente fazendo-se uso de algoritmos genéticos na etapa de treinamento da rede neural.

Sabe-se, entretanto, que o processo de aprendizagem das redes neurais Perceptron de Múltiplas Camadas pode ser bastante lento, principalmente quando são utilizadas topologias complexas, podendo prejudicar, assim, o desempenho de incontáveis aplicações [4], [5] e [6].

Portanto, visando minimizar o tempo de treinamento das redes neurais e melhorar o desempenho das aplicações, técnicas de paralelismo foram aplicadas ao algoritmo de treinamento desenvolvido. Além disso, foram criados vários processos paralelos, cada um deles correspondendo a uma população diferente. Ao todo, foram desenvolvidos quatro comportamentos de evolução distintos, de modo que as populações criadas evoluam seguindo diferentes critérios de seleção e reprodução. Além disso, tem-se que ao final de um número predeterminado de vezes, os processos criados se comunicam a fim de trocarem informações relacionadas aos melhores indivíduos selecionados por cada um deles. Desta maneira, o sinal de erro produzido pela rede neural é minimizado mais rapidamente. Algumas aplicações em aproximação de funções foram realizadas com o intuito de evidenciar a efetividade do algoritmo genético paralelo desenvolvido.

2. Treinamento de Perceptrons de Múltiplas Camadas com Algoritmos Genéticos

Um algoritmo genético consiste em um método de busca dinâmico baseado na Teoria de Seleção Natural, podendo ser usado com sucesso no processo de aprendizagem de uma rede neural Perceptron de Múltiplas Camadas [7]. Neste caso, tem-se que cada indivíduo da população irá codificar uma rede neural específica. Portanto, em cada população existirá um número finito de indivíduos apresentando a mesma topologia da rede neural com pesos sinápticos distintos. No fim do processo de treinamento, o material genético do indivíduo selecionado irá conter a configuração final da rede neural.

Os passos principais da implementação do algoritmo genético encontram-se ilustrados no fluxograma da Figura 1.

O algoritmo genético desenvolve-se da seguinte maneira: uma população inicial de indivíduos é gerada ao acaso. Cada material genético de um indivíduo, consistindo de um vetor de 0's e 1's conterá uma descrição de uma rede neural específica. Posteriormente, os indivíduos são decodificados e avaliados de acordo com uma função de custo predefinida. Os melhores indivíduos são aqueles com função de custo próxima de zero.

Uma função que converte um vetor de 0's e 1's em um número real foi criada para decodificar as informações de cada cromossomo do indivíduo.

A função de custo, por sua vez, é calculada para cada conjunto de amostras de treinamento adicionando-se todos os sinais de erro obtidos para cada elemento localizado na saída da rede neural e dividindo o resultado pelo número de amostras aplicadas à rede. O sinal de erro, por outro lado, é o resultado da subtração entre uma resposta desejada previamente definida e a resposta real obtida pela rede:

$$e_j(n) = d_j(n) - o_j(n) \quad (1)$$

Se a condição de parada ainda não for alcançada, o processo de aprendizagem continua e um certo número de indivíduos é escolhido, de acordo com alguns critérios de seleção, para serem os pais da próxima geração. No algoritmo genético desenvolvido neste artigo, existem quatro formas diferentes de reprodução, conforme será detalhados na Seção 3. Em geral, pode ser dito que para formar uma nova população, os indivíduos são selecionados de acordo com sua capacidade de minimizarem a função de custo. Deste modo, os indivíduos que obtiveram o menor sinal de erro têm uma chance melhor de reprodução, enquanto que os outros são mais prováveis de desaparecerem.

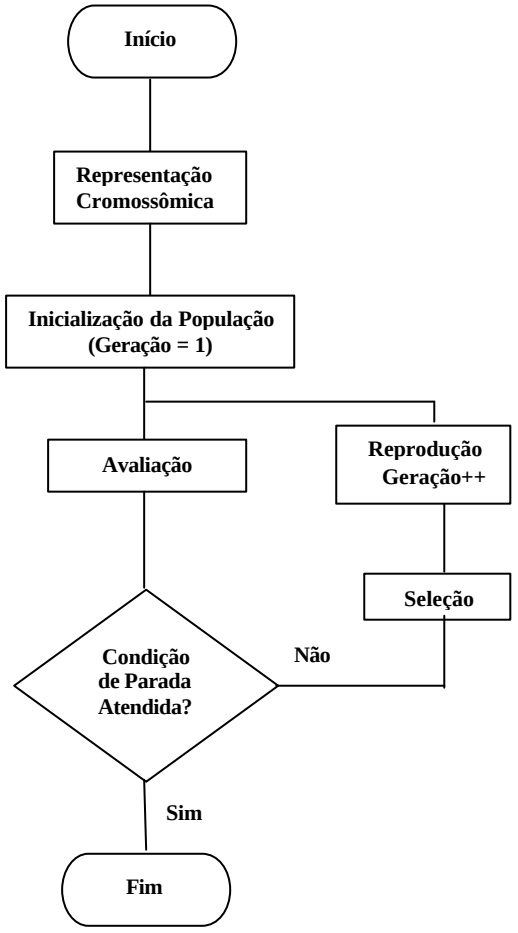

Figura 1: Fluxograma básico do algoritmo genético

3. Algoritmo Genético Paralelo com Diferentes Comportamentos Evolutivos

O algoritmo genético usado no processo de aprendizagem da rede neural Perceptron de Múltiplas Camadas tem como objetivo encontrar, numa determinada população, os indivíduos que produzem os sinais de erro mais baixos, mantendo sempre a diversidade das mesmos. Sendo assim, os diferentes tipos de população crescem paralelamente, segundo diferentes critérios de reprodução. A seguir, são apresentados quatro critérios distintos de reprodução utilizados no desenvolvimento deste algoritmo.

O primeiro critério implementado, ilustrado na Figura 2, consiste na ordenação dos indivíduos da população conforme o resultado fornecido pela função de custo dos mesmos. Feito isso, seleciona-se a primeira metade dos indivíduos, e, entre eles, são escolhidos, randomicamente, os pais da próxima geração. Sendo assim, após a reprodução da outra metade que foi esquecida, nascerá uma nova geração. Um pequeno número de mutações, também obtido de maneira aleatória, é introduzido na nova população.

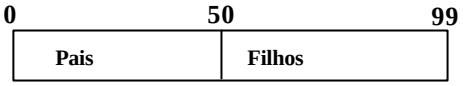

Figura 2: Ilustração do primeiro critério de seleção

No segundo critério de seleção, os indivíduos da população também são ordenados de acordo com a função de custo produzida. Entretanto, ao invés de mantermos a primeira metade dos indivíduos para a próxima geração, foi escolhido, randomicamente, um número **r** localizado no intervalo de zero e o número total de indivíduos menos um para indicar os pais da próxima geração. Por exemplo, se o número total de indivíduos é igual a 100 e **r** é igual a 20, como ilustrado na Figura 3, os primeiros 20 indivíduos serão escolhidos aleatoriamente para serem os pais dos novos indivíduos. Sendo assim, os 80 indivíduos restantes morrerão, enquanto 80 novos indivíduos serão criados, aleatoriamente, por combinação do material genético de dois indivíduos selecionados entre os 20 primeiros. Um pequeno número de mutações também é inserido na próxima geração.

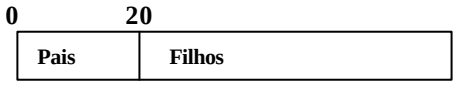

Figura 3: Ilustração do segundo critério de seleção

No terceiro critério de reprodução implementado, dois números, **r1** e **r2**, são escolhidos aleatoriamente para indicar os pais da próxima geração. Conforme ilustrado na Figura 4, se **r1** é igual a 30 e **r2** é igual a 60, os primeiros 30 indivíduos podem ser, aleatoriamente, selecionados para serem os pais dos indivíduos localizados no intervalo de 30 a 59. Após a reprodução, os indivíduos localizados neste intervalo são escolhidos, aleatoriamente, para serem os pais dos indivíduos localizados no intervalo de 60 a 99. Mutação é introduzida dentro da nova geração.

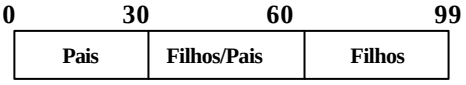

Figura 4: Ilustração do terceiro critério de seleção

No quarto critério de seleção, os indivíduos da população são também ordenados de acordo com os

valores da função de custo produzida. Desta forma, os indivíduos localizados na primeira metade, isto é, aqueles com os sinais de erro mais baixos serão, aleatoriamente, escolhidos para serem os pais da próxima geração, conforme ilustrado na Figura 5.

Após a reprodução da outra metade que foi esquecida, nasce uma nova geração e dois ou mais números são selecionados, de maneira aleatória, para representar a quantidade de *crossing-over*. Na Figura 5, dois números, 30 e 60, são escolhidos aleatoriamente para indicar os indivíduos enquanto os números 2 e 5 são selecionados para indicar o intervalo do material genético que será permutado. Além disso, mutação também é introduzida dentro da nova geração.

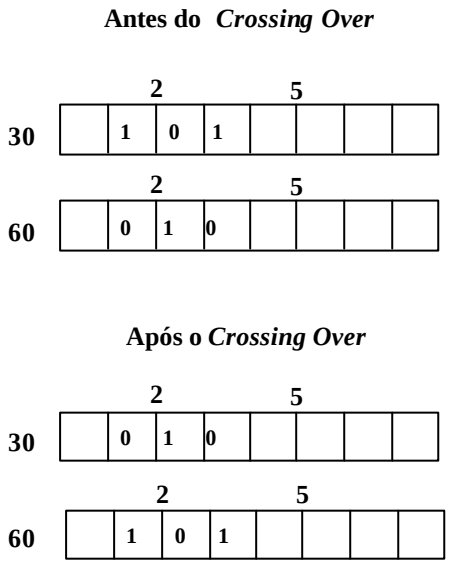

Figura 5: Ilustração do quarto critério de seleção

4. Aplicações em Aproximação de Funções

O algoritmo genético paralelo desenvolvido foi utilizado na etapa de treinamento da rede neural Perceptron de Múltiplas Camadas para aproximar a função $f(x) = 1/x$.

Podemos verificar que a grande vantagem desse algoritmo é permitir que populações diferentes com comportamentos evolutivos distintos possam evoluir de forma paralela. A Figura 6 representa o sinal de erro médio quadrado do algoritmo genético usual (seqüencial) com quatro populações evoluindo independentemente, onde cada linha representa o melhor desempenho individual de cada uma das quatro populações.

Portanto, como as populações evoluem independentemente, essa situação é equivalente àquela em que um algoritmo genético isolado é executado, isto é, existe uma população da qual surgem novas gerações, sempre buscando o indivíduo que melhor se adapta à solução. Não há troca de informação ou cooperação entre as populações existentes.

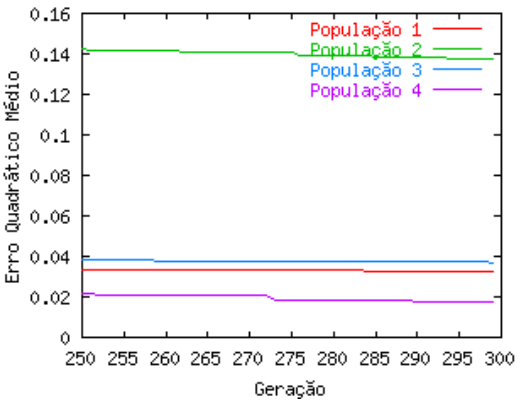

Figura 6: Sinal de erro médio quadrado do algoritmo genético com quatro populações evoluindo independentemente para a função $f(x)= 1/x$

A Figura 7, por outro lado, mostra a evolução de quatro populações, onde cada linha está associada a uma das quatro populações que está evoluindo, indicando o melhor desempenho individual de cada uma dessas populações. Pode-se observar que é possível obter um melhor desempenho com a paralelização do algoritmo genético de treinamento da rede neural.

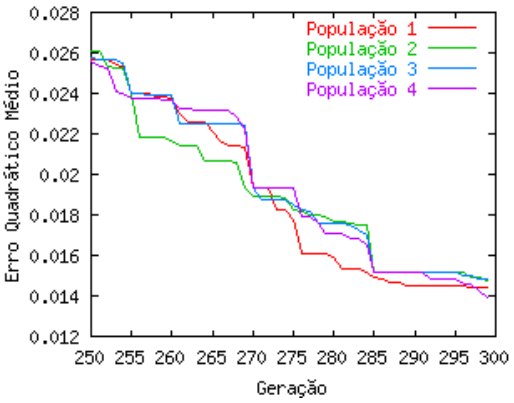

Figura 7: Sinal de erro médio quadrado do algoritmo genético com quatro populações para a função $f(x)=1/x$

Vemos, na Figura 7, que as quatro populações evoluem independentemente entre 250 e 255. Nessa geração, a População 4 contém a melhor seleção de todos os indivíduos, isto é, apresenta o sinal de erro mais baixo. Na geração 255, entretanto, todas as tarefas se comunicam e todas as populações recebem uma cópia do melhor indivíduo até então encontrado. Assim, todas as populações que inicialmente evoluíam independentemente, agora têm um elemento comum, o melhor indivíduo. A partir daí, as populações continuam novamente a evoluir independentemente entre 255 e 275. Entretanto, agora, o melhor indivíduo não é mais um membro da População 4, mas, ao invés disso, é um membro da População 1. Além disso, percebe-se claramente, através do gráfico da Figura 6 que muito provavelmente as populações estavam presas a um mínimo local. Por outro lado, analisando o gráfico da

Figura 7, onde utilizou-se o algoritmo genético paralelo implementado este fato não foi verificado uma vez que com as comunicações impedimos que os membros das populações fiquem presos em mínimos locais.

Na Figura 8, a saída final da rede neural é reconstruída.

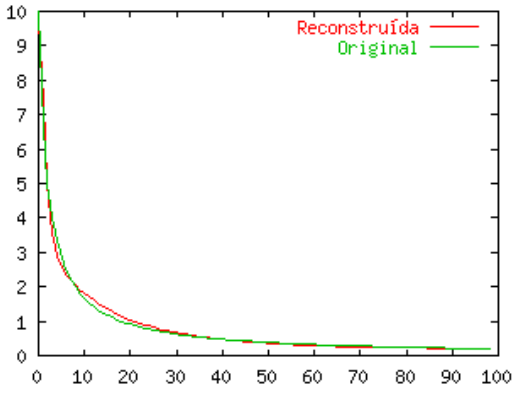

Figura 8: Reconstrução da saída da função *f(x)=1/x* obtida do Perceptron de Múltiplas Camadas

O algoritmo genético paralelo desenvolvido foi também utilizado no processo de aprendizagem da rede neural Perceptron de Múltiplas Camadas para aproximar a função *f(x)=sen(2px)*.

A Figura 9 representa o sinal de erro médio quadrado do algoritmo genético seqüencial com quatro populações evoluindo independentemente.

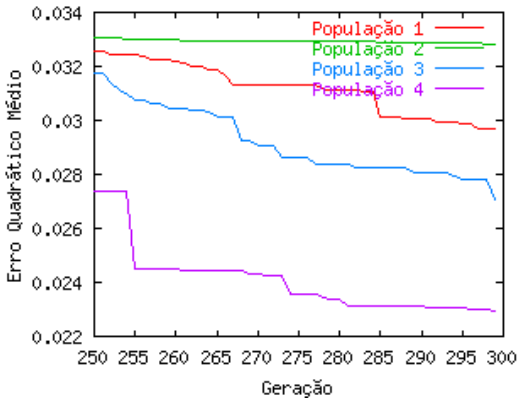

Figura 9: Sinal de erro médio quadrado do algoritmo genético com quatro populações evoluindo independentemente para a função *f(x)= sen(2px)*

Como foi dito anteriormente, existem quatro populações evoluindo independentemente, sem qualquer cooperação ou comunicação. As novas gerações nascem somente de suas respectivas populações.

Na Figura 10, por outro lado, há quatro populações que evoluem, onde cada linha representa o melhor indivíduo de cada uma das quatro populações. Pode-se verificar que a partir da geração 255 até a geração 270, a População 1 apresenta o sinal de erro mais baixo e, como resultado, a melhor seleção de indivíduos. Entretanto, após a troca de informação entre as populações, verifica-se que, a partir da geração 270, os sinais de erro produzidos por outras populações são altamente reduzidos. Como resultado, pode-se verificar uma grande melhoria na performance final do algoritmo.

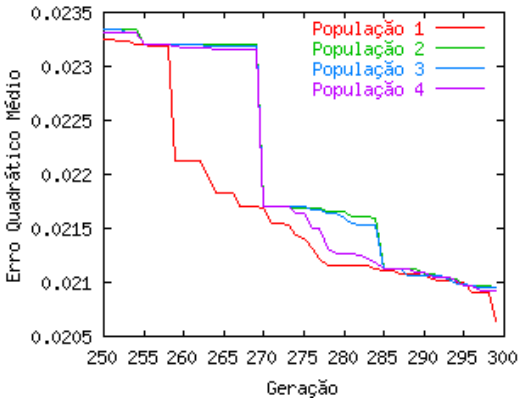

Figura 10: Sinal de erro médio quadrado do algoritmo genético com quatro populações para a função *f(x) = sen(2px)*

A Figura 11 ilustra a saída do Perceptron de Múltiplas Camadas.

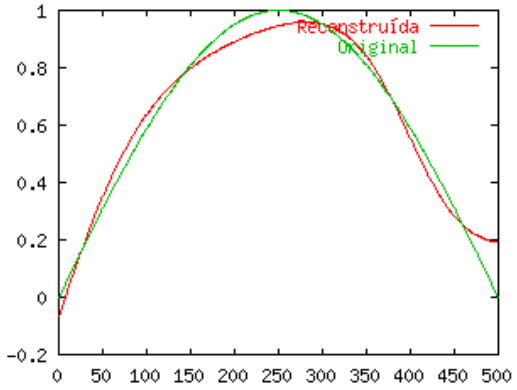

Figura 11: Saída reconstruída da função *f(x)=sen(2px)* obtida do Perceptron de Múltiplas Camadas

Portanto, uma das vantagens da paralelização do algoritmo deve-se ao fato de que várias populações evoluem independentemente até uma dada posição, na qual apresentam o melhor indivíduo em comum a todas populações. Daí em diante, continuam a evoluir independentemente, comunicando-se em um outro momento para trocarem informações sobre o melhor indivíduo e, assim, sucessivamente. Portanto, as populações que foram "capturadas" em sinais de erros altos passam imediatamente para o mesmo nível dos outros. Como mostrado na Figura 7, a População 2, entre as gerações 250 e 255, consiste na população que apresenta sinal de erro mais alto. Entretanto, da junção 255 até a junção 270, após o intercâmbio de

informações, torna a ser a população com sinal de erro mais baixo.

5. Conclusões

O algoritmo genético paralelo implementado para populações com comportamentos diferentes, e usado no processo de aprendizagem do Perceptron de Múltiplas Camadas para aproximar funções mostrou ser bastante eficiente quando comparado ao algoritmo genético implementado na forma seqüencial.

Verificou-se, da análise dos resultados, que a troca de informações entre populações diferentes representou um grande melhoramento no desempenho final do algoritmo. Assim, ao invés de um algoritmo genético puramente competitivo, com populações competindo entre si na procura do melhor indivíduo capaz de minimizar a função de custo, tem-se um algoritmo genético cooperativo em que populações diferentes cooperam entre si, na busca da solução ideal: o indivíduo que introduz a função de custo mínimo. Dessa maneira, após a troca de informações, todas as demais populações irão conter o melhor indivíduo até então encontrado como membro de sua população. Sendo assim, tem-se que populações associadas a sinais de erro elevados imediatamente passam para o mesmo nível das demais.

Por fim, vale ressaltar que o uso de comportamentos evolutivos diferentes em cada população conduz a uma grande diversificação dos indivíduos intensificando, assim, a busca por uma solução ideal.

Referências

[1] S. Haykin, "*Neural Networks: A Comprehensive Foundation*", Maxwell Macmillan International, 1994.

[2] J. Torresen, "Parallelization of Backpropagation Training for Feed-Forward NeuralNetworks", PhD Thesis, The Norwegian Institute of Technology, 1996.

[3] S. K. Foo, P. Saratchandran and N. Sundararajan, "Parallel Implementation of Backpropagation Neural Networks on a Heterogeneous Array of Transputers", *IEEE Trans. Systems, Man and Cybernetics - Part B*, Vol. 27:1, pp. 118-126, 1997.

[4] R. Alves, J. D. Melo, A. D. Dória Neto and A. C. M. L. Albuquerque, "New Parallel Algorithms for Back-Propagation Learning", IJCNN 2002, Honolulu, USA, 2002.

[5] T. G. Crainic and M. Gendreau, "Cooperative Parallel Tabu Search for Capacitated Network Design", Journal of Heuristics, 8, 601-627, 2002.

[6] A. Muhammad, A. Bargiela and G. King, "Fine-grained Parallel Genetic Algorithm: A Global Convergence Criterion", Int. Journal of Computer Mathematics, Vol. 73(2), 139-155, 1999.

[7] Z. Michalewicz, "*Genetic Algorithms+Data Structures=Evolution Programs*",Springer-Verlag, 1992.