# Rede Perceptron de Múltiplas Camadas para Sistema Híbrido Reconfigurável

**Vitor C. F. Gomes[1], Élcio H. Shiguemori[2], Andrea S. Charão[3], Haroldo F. C. Velho[1]**

[1] Instituto Nacional de Pesquisas Espaciais – INPE

[2]Departamento de Ciência e Tecnolodia Aeroespacial – DCTA

[3]Universidade Federal de Santa Maria - UFSM

{vitor.gomes, haroldo}@lac.inpe.br, elcio@ieav.cta.br, andrea@inf.ufsm.br

***Resumo.*** *Redes Neurais Artificiais são usadas na solução de problemas em diversas áreas do conhecimento. Apesar de poder reduzir o custo de computação, seu uso em grades volumes de dados requer o uso intensivo de recursos computacionais. No contexto de aplicações críticas, este trabalho apresenta o projeto de uma MLP para FPGA e a implementação em um sistema híbrido reconfigurável. Essa solução é testada e comparada com uma aplicação em software.*

**Palavras-chave:** *Redes Neurais, FPGA, Cray XD1*

## 1. Introdução

Redes Neurais Artificiais (RNAs) são sistemas de processamento de informações inspirados na rede neural cerebral, que, nos últimos anos, têm sido utilizados com sucesso na soluções de aplicações em diferentes áreas do conhecimento[Skoda et al. 2011, Liu e Liang 2005, Gadea et al. 2000].

A implementação de RNAs é tradicionalmente realizada em software em máquinas sequenciais. Apesar de não ser uma limitação, esse fato deixa de aproveitar o paralelismo inerente desse modelo computacional. Além disso, quando operadas sobre grandes volumes de dados, as RNAs requerem o uso intensivo de recursos computacionais, indicando a necessidade de alternativas para a sua execução. Diversos trabalhos trazem soluções de alto desempenho para essa computação, envolvendo ambientes paralelos, uso de GPUs ou FPGAs. Desses recursos, FPGAs apresentam maior afinidade com as características de paralelismo apresentadas pelas RNAs.

FPGAs são dispositivos lógicos programáveis compostos por blocos lógicos configuráveis que podem ser reconfigurados diversas vezes. Sua utilização visa obter a flexibilidade de soluções baseadas em software com o desempenho de soluções implementadas em hardware [Chamberlain et al. 2008].

Apesar do potencial de FPGAs, aplicações que envolvem simulações complexas e grande volume de dados estão fortemente vinculadas a software. Para essas situações, onde deseja-se usar o poder de computação de FPGAs combinado com CPUs, existem sistemas híbridos reconfiguráveis. Esses sistemas incorporam FPGAs em suas arquiteturas permitindo a utilização conjunta com processadores de propósito geral.

Seguindo essa tendência, este trabalho apresenta o projeto e implementação de uma MLP para um sistema híbrido reconfigurável. Ao longo deste trabalho, são apresentados detalhes sobre Redes Neurais e Cray XD1. Na sequência, é apresentado o projeto da MLP

para FPGA e os detalhes da sua implementação. Na seção 6, é apresentada uma avaliação do projeto, enquanto a seção 7 apresentada as conclusões.

## 2. Redes Perceptron de Múltiplas Camadas (MLP)

Redes Neurais Artificiais são modelos de computação baseados em neurônios artificiais diretamente análogos aos neurônios biológicos. A Figura 1 mostra a representação esquemática de um neurônio artificial. Seus elementos fundamentais são: (a) entradas $\{x_1, x_2, \ldots, x_N\}$; (b) conexões sinápticas com pesos associados $\{w_1, w_2, \ldots, w_n\}$; e (c) função de ativação $\varphi$ que relaciona a atividade interna $u$ do neurônio com o sinal de saída $y$ [Skoda et al. 2011].

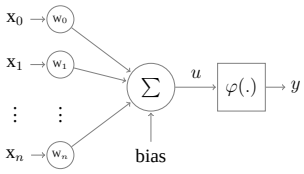



**Figura 1. Neurônio artificial.**

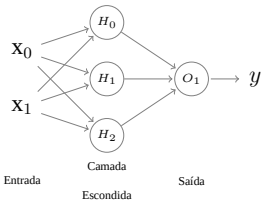



**Figura 2. MLP**

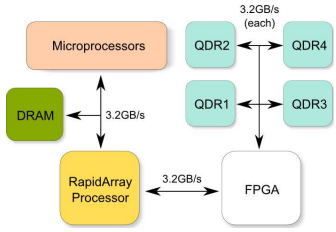



**Figura 3. Blade XD1**

Através da combinação de neurônios artificiais, pode-se gerar diferentes arquiteturas de redes. Uma das mais utilizadas na literatura é a Rede Perceptron de Múltiplas Camadas que é alvo de estudo neste trabalho. A Figura 2 ilustra uma MLP com uma camada escondida.

## 3. Cray XD1

O Cray XD1 é um sistema híbrido composto por seis nós (*blades*) interconectados. Cada *blade* contém dois processadores AMD Opteron de 2.4 GHz e um FPGA Xilinx Virtex II Pro. A Figura 3 mostra a arquitetura de um *blade* do XD1.

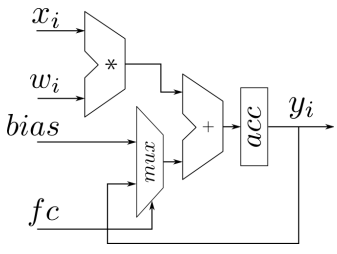



**Figura 4. MAC**

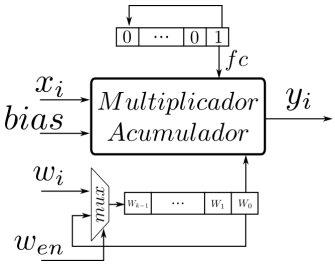



**Figura 5. Neurônio**

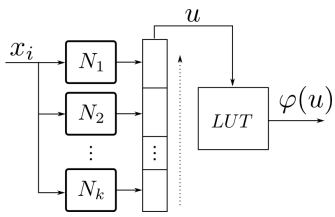



**Figura 6. Camada**

Para permitir a comunicação entre o FPGA e os processadores de um *blade*, a Cray disponibiliza a API RapidArray Transport Core. Essa API permite que a CPU realize envio e recepção de dados com o FPGA e que o FPGA leia e escreva em regiões de memória compartilhadas com a CPU. No Cray XD1, o FPGA tem acesso a diferentes níveis de memória. A DRAM, possui até 8GB e latência variável de leitura. A QDR II SRAM, com 4 blocos independentes de 4MB, pode ser acessada diretamente pelo FPGA com latência de 8 ciclos. Por fim, em menor quantidade, está disponível a memória interna ao FPGA que pode ser acessada em 1 ciclo.

## 4. MLP para Cray XD1

Considerando uma abordagem *bottom-up*, a seguir é descrito o funcionamento interno de cada módulo que compõe a MLP para FPGA. Cada módulo aparece novamente nos módulos superiores como componentes de computação.

A computação de multiplicação das entradas pelos pesos sinápticos e a acumulação da atividade interna juntamente com o bias é realizada pelo componente multiplicador acumulador MAC, ilustrado na Figura 4. Cada entrada $x_i$ é multiplicada por um peso $w_i$ e somado ao registrador $acc$ ou do $bias$. Para selecionar entre esses operandos, o sinal $fc$ é sinalizado a cada primeiro ciclo de operação.

O próximo módulo é o neurônio, que utiliza um MAC e estruturas de controle e tem seu diagrama apresentado na Figura 5. Nesse componente, o sinal $fc$ é gerado através de um registrador de deslocamento que possui um único bit 1. Para o gerenciamento dos pesos, são utilizados registradores interligados em uma fila circular. Os pesos são deslocados a cada entrada $x_i$.

O módulo computacional superior na MLP é uma camada, a qual é construída através da combinação de neurônios com entradas conectadas em um barramento único. A saída de cada neurônio é conectada a uma posição de um registrador de deslocamento com carga paralela. A Figura 6 ilustra essa estrutura. A partir desse momento, os neurônios já estão prontos para receberem novos dados, enquanto os resultados são encaminhados para uma *Lookup Table* (LUT), responsável por simular a operação da função de ativação.

Para finalizar o projeto da MLP para FPGA, as camadas podem ser concatenadas em série formando a rede neural. A entrada de cada camada é conectada diretamente a saída da camada anterior. Considerando uma camada como um módulo de computação, temos então uma sequência de operações em *pipeline*. A computação de cada camada pode acontecer independentemente das demais, permitindo que múltiplos conjuntos de dados possam sem computados sequencialmente defasados em uma camada de computação.

## 5. Implementação

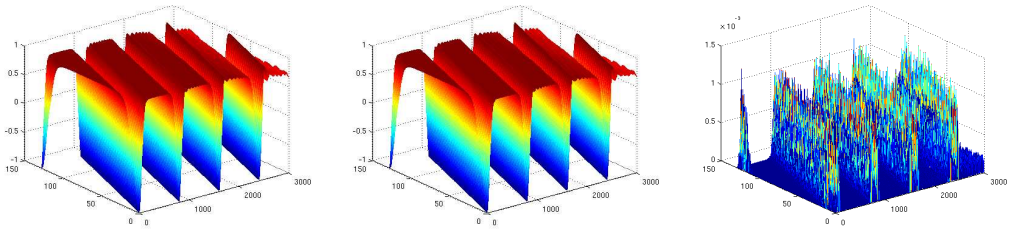
O projeto de MLP foi descrito em VHDL para o sistema híbrido reconfigurável Cray XD1. Devido a baixa afinidade dos FPGAs com operações em ponto flutuante, as computações foram feitas em ponto fixo de 16 bits.

Devido ao comportamento determinístico do FPGA, o total de ciclos necessários para realizar a computação de uma camada é calculado por: $ciclos = (E + 5) + (N + 8)$, onde (E + 5) é o total de entradas somado ao total de estágios do MAC e (N + 8) é o total de neurônios da camada somado à latência de leitura da QDR RAM. O total de ciclos necessários para executar uma MLP será a soma de ciclos de cada uma de suas camadas. Destaca-se ainda que, para fluxos contínuos de dados, ocorrerá uma saída a cada ciclo de relógio após o preenchimento de todos os estágios de computação da MLP.

Para evitar que a aplicação em software seja sobrecarregada com a conversão de ponto flutuante para ponto fixo, adicionou-se à entrada da MLP um conversor. À saída foi concatenado um módulo que converte o resultado em ponto fixo para ponto flutuante. Cada um desses módulos possui 6 estágios, adicionando 12 ciclos ao pipeline geral da aplicação.

## 6. Avaliação

Para avaliar nosso projeto, foi implementada uma versão em linguagem C para ser executada em CPU. A aplicação escolhida para os testes é a descrita em [Furtado et al. 2011], onde é utilizada uma MLP com 2 entradas, 3 neurônios na camada escondida e 1 neurônio de saída para a assimilação de dados atmosféricos. Para o teste, foi sintetizada uma rede para o FPGA e compilado um programa. Os resultados das execuções são apresentados nas Figuras 7(a), 7(b) e 7(c), onde trazem, respectivamente, a saída da execução em software, a saída da execução em FPGA e o quadrado da diferença ponto a ponto. O erro quadrático médio calculado é de $5,18 * 10^{-5}$ com variância de $1,79 * 10^{-8}$. O maior erro quadrático obtido foi de $1,30 * 10^{-3}$.



(a) Software (b) FPGA (c) Erro quadrático

**Figura 7. Resultados**

## 7. Conclusões e trabalhos futuros

Nesse trabalho, foi apresentado o projeto e a implementação de uma MLP para o sistema Cray XD1. Essa aplicação explorou o paralelismo em variados níveis de granularidade, no aspecto espacial e temporal da computação. Através da execução de testes, foi verificada variação numérica do uso de ponto fixo em relação ao uso do ponto flutuante. O erro quadrático máximo foi $1,30 * 10^{-3}$. Como trabalho futuro, planejamos verificar a viabilidade da execução das computações em ponto flutuante, para ter total compatibilidade com aplicações em software.

## Referências

Chamberlain, R. D., Lancaster, J. M., e Cytron, R. K. (2008). Visions for application development on hybrid computing systems. *Parallel Comput.*, 34(4-5):201–216.

Furtado, H. C. M., Velho, H. H. F., e Macau, E. E. N. (2011). Assimilação de dados com redes neurais artificiais em equações diferenciais. In *Anais do DINCON'2011*.

Gadea, R., Cerdá, J., Ballester, F., e Mocholí, A. (2000). Artificial neural network implementation on a single fpga of a pipelined on-line backpropagation. In *Proceedings of the 13th international symposium on System synthesis*, ISSS '00, pages 225–230, Washington, DC, USA. IEEE Computer Society.

Liu, J. e Liang, D. (2005). A survey of fpga-based hardware implementation of anns. In *Neural Networks and Brain, 2005. ICNN B '05. International Conference on*, volume 2, pages 915 –918.

Skoda, P., Lipic, T., Srp, A., Rogina, B. M., Skala, K., e Vajda, F. (2011). Implementation framework for artificial neural networks on fpga. In *MIPRO, 2011 Proceedings of the 34th International Convention*, pages 274 –278.