Análise de desempenho da biblioteca Theano com GPU sob a ótica do CUDA Profiler

Felipe Melo[1], **Raul S. Ferreira**[1,2], **Fellipe Duarte**[1,2], **Marcelo Zamith**[1]

[1]Instituto Multidisciplinar
Universidade Federal Rural do Rio de Janeiro (UFRRJ)
Nova Iguaçu – RJ – Brazil

[2]COPPE - Graduate School and Research in Engineering
Universidade Federal do Rio de Janeiro (UFRJ)
Rio de Janeiro – RJ – Brazil

{felipemelo, duartefellipe, mzamith}@ufrrj.br, raulsf@cos.ufrj.br

Abstract. *Increasingly, graphics cards (GPUs - Graphics Processing Units) have been used to increase the performance of computational algorithms, and, in the last years, several libraries have appeared in order to create a abstraction layer to developers in using GPUs. However, few studies evaluate the efficiency of these libraries in using GPUs. This article aims to present an analysis of the performance of one of these libraries on some GPUs through matrix basic operations contained in Deep Neural Networks (RNP), using CUDA Profiler, NVIDIA tool profiling GPUs, which shows in detail how the execution was performed on GPUs. We present the performance results and conclusions of the use of the library on different GPUs.*

Resumo. *Cada vez mais as placas gráficas (GPUs - Graphics Processing Units) vêm sendo usadas para aumentar a performance dos algoritmos computacionais e, nos últimos anos, diversas bibliotecas surgiram para facilitar o desenvolvimento de programas que fazem uso dessas placas (Theano, TensorFlow, Torch entre outras). Porém poucos trabalhos avaliam a eficiência destas bibliotecas no processamento das GPUs. Este artigo tem como objetivo apresentar uma análise da performance de uma destas bibliotecas em algumas GPUs através de operações matriciais contidas em Redes Neurais Profundas (RNP), utilizando o CUDA Profiler, ferramenta da NVIDIA, para mostrar detalhes na execução das GPUS. Apresentamos os resultados de desempenho e as conclusões do uso da biblioteca entre as diferentes GPUs.*

1. Introdução

O campo da Inteligência Artificial ganhou um grande aliado com as aceleradoras gráficas (GPUs – *Graphics Processing Units*). Este tipo de hardware trabalha como um verdadeiro cluster, disponibilizando um grande poder computacional e acelerando o processamento de diversas técnicas de aprendizagem como por exemplo Redes Neurais Profundas [Krizhevsky et al. 2012, Sainath et al. 2013].

Além disto, algumas bibliotecas vêm sendo desenvolvidas a fim de disponibilizar uma camada de abstração para os desenvolvedores, permitindo utilizar operações de

álgebra linear, que são empregadas em técnicas de Inteligência Artificial, tanto em CPU multicore quanto em GPUs.

Embora a utilização de diferentes arquiteturas de processadores estejam disponíveis nessas bibliotecas, não é claro quão eficiente as bibliotecas são em relação a estas arquiteturas de hardware, especialmente com relação ao uso das GPUs. Em geral, os trabalhos que avaliam a performance, o fazem medindo o tempo total de processamento [Bergstra et al. 2010].

Para isso, foram escolhidas operações básicas de álgebra linear[Kovács 2002], que são utilizadas em Redes Neurais, a fim de analisar a performance destas operações em diferentes GPUs. E, o CUDA Profiler foi escolhido como ferramenta de coleta das métricas da GPU. Esta ferramente registra com maior acurácia os tempos de execução[Mayanglambam et al. 2010], número de operações de ponto flutuante executadas, eficiência atingida na execução destas operações de ponto flutuante, considerando o pico teórico da GPU.

2. Experimentos

Para avaliar a biblioteca Theano em diferentes arquiteturas de GPU[Owens et al. 2008] NVIDIA, foi proposto um teste que avalia as operações básicas utilizadas em RNP. A Tabela 1 apresenta as características das GPU utilizadas nos testes. Foram escolhidas estas três placas com o objetivo de executar os testes em diferentes arquiteturas e também porque representam diferentes utilizações de GPUs, ou seja, a GT710M é uma GPU para notebook, a GTX980 é utilizada em computadores Desktop e, por fim, a K40 é uma placa voltada para servidores.

Característica/GPU	**GT710M**	**K40**	**GTX980**
Arquitetura	Fermer	Kepler	Maxwell
CUDA cores	96	2.880	2.048
Freq. core	1,55GHz	745MHz	1.241 MHz
Freq. memória	900 MHz	3.004GHz	3.505GHz
RAM (Capacidade)	2GB	12GB	4GB
Cache L2	0,125MB	1,5MB	2MB
Banda passante	14GB/s	288GB/s	224GB/s

Tabela 1. Características das GPUs testadas

Foram escolhidas operações básicas que são utilizadas em RNPs. Para cada operação, foram analisados o tempo de execução dos kernels utilizados e também a quantidade de operações de ponto flutuante (*flops*) executados por estes kernels. Assim, é possível calcular *flops* por segundo alcançado por cada placa.

A fim de coletar estes dados com uma maior precisão, utilizou-se o CUDA profiler[1] modo console com os parâmetros: i) `--print-gpu-trace -u s` para obter o tempo de computação dos *kernels*. ii) `--metrics flop_count_sp, flop_sp_efficiency, flop_count_dp, flop_dp_efficiency` para registrar a quantidade de *flops* e sua eficiência, tanto em precisão simples quanto em precisão dupla. A eficiência representa uma taxa que a placa alcança em relação ao seu pico teórico

[1]http://developer.download.nvidia.com/compute/cuda/2_1/cudaprof/cudaprof.html

de *flops*, embora os testes mostrem que os *kernels* trabalham com precisão simples e para medir essa eficiência, seis operações básicas (que são operações matriciais) são testadas:

- Adição (add): Operações de adição são bastante usadas em RNPs, pois sempre procuramos trabalhar entrada e pesos de uma rede neural na forma matricial, ela se dá na forma $C_{mn} = A_{mn} + B_{mn}$, onde $c_{ij} = a_{ij} + b_{ij}$.
- Produto vetorial (dot): A operação de produto entre matrizes difere um pouco das demais operações escolhidas, pois não é realizada ponto a ponto, seu desempenho no paralelismo pode variar com a implementação, o que não ocorre tanto com as demais operações escolhidas, ela se dá na forma $C_{mo} = A_{mxn} * B_{nxo}$, onde $c_{ij} = \sum_{k=1}^{o} a_{ik} * b_{kj}$.
- Multiplicação (mul): A multiplicação matricial ponto a ponto(ou produto hadamard) é menos conhecida porém também está presente nas rede neurais principalmente nas rede convolucionais, pois a convolução pode ser vista como um filtro que é aplicado à entrada na forma de $C_{mn} = A_{mn}xB_{mn}$, onde $c_{ij} = a_{ij} * b_{ij}$.
- Produto escalar: Produto entre uma matriz e um escalar faz parte de qualquer rede neural, um exemplo está na taxa de aprendizado. A implementação é simples e altamente paralelizável. Ela se dá na forma $C_{mn} = s * A_{mn}$, onde $c_{ij} = s * a_{ij}$.
- Sigmoid: O operação de sigmoid é uma função não linear comum como saída dos perceptrons das RNP, sua implementação segue a forma de $C_{mn} = \frac{1}{1+e^{-A_{mn}}}$, onde $c_{ij} = \frac{1}{1+e^{-a_{ij}}}$.
- Tanh: Da mesma forma que a operação de sigmoid a função de tangente hiperbólica é uma função não linear presente como saída dos perceptrons das RNPs, sua implementação se dá na forma: $Cmn = \frac{e^{A_{mn}} - e^{-A_{mn}}}{e^{A_{mn}} + e^{-A_{mn}}}$, onde $c_{ij} = \frac{e^{a_{ij}} - e^{-a_{ij}}}{e^{a_{ij}} + e^{-a_{ij}}}$.

As Tabelas 2 e 3 apresentam os resultados das operações ADD, MUL, DOT, Scalar, Sigmoid e Tanh. A coluna *MFlops/s* indica a quantidade de operações de ponto flutuante por segundo, enquanto a eficiência destes flops em relação ao pico teórico é apresentada na coluna *Efic. (%)* e representa um percentual, a coluna MB representa o tamanho da quatidade de dados processados pelas operações em Megabytes.

-	-	**ADD**		**MUL**		**DOT**		
MB	Placas	MFlops/s	Efic. (%)	MFlops/s	Efic. (%)	MB	MFlops/s	Efic. (%)
6	GT710M	979,11	**0,62**	971,99	**0,62**	5	133.021,16	**88,97**
	K40	10.502,79	0,22	10.202,14	0,22		1.527.464,91	34,89
	GTX980	12.261,28	0,23	12.432,37	0,25		3.533.837,89	77,97
12	GT710M	953,95	**0,62**	962,62	**0,61**	9	134.452,62	**88,98**
	K40	10.717,41	0,24	10.631,33	0,22		1.599.015,71	36,80
	GTX980	12.326,89	0,24	12.428,17	0,27		3.781.740,39	82,91
24	GT710M	949,30	**0,63**	945,05	**0,61**	17	135.342,34	84,77
	K40	10.752,33	0,24	10.752,33	0,24		1.627.485,14	37,65
	GTX980	12.619,50	0,25	12.695,52	0,27		3.849.025,26	**85,45**
48	GT710M	944,86	**0,63**	946,98	**0,62**	33	124.760,44	84,24
	K40	10.726,04	0,25	10.864,62	0,25		1.639.604,59	38,13
	GTX980	12.588,65	0,25	12.626,45	0,27		3.869.585,35	**87,32**
96	GT710M	941,58	**0,63**	941,58	**0,61**	65	120.861,45	80,10
	K40	10.726,58	0,25	10.767,84	0,25		1.652.063,72	38,43
	GTX980	12.842,37	0,26	12.862,04	0,28		3.869.462,63	**86,94**

Tabela 2. Tabela de Flops/s das operações: ADD, MUL e DOT

-	-	**Scalar**		**Sigmoid**		**Tanh**	
MB	Placas	MFlops/s	Efic. (%)	MFlops/s	Efic. (%)	MFlops/s	Efic. (%)
2	GT710M	1.448,31	0,93	18.031,07	11,35	14.651,79	9,28
	K40	17.593,56	0,25	101.424,76	1,90	92.563,65	1,76
	GTX980	23.616,58	0,43	196.257,54	3,16	145.738,51	2,87
4	GT710M	1.448,31	0,95	18.126,98	11,60	14.610,79	9,37
	K40	12.453,40	0,27	90.274,75	2,01	79.062,71	1,74
	GTX980	20.722,85	0,30	188.205,95	2,89	135.619,95	2,18
8	GT710M	1.434,44	0,95	17.912,60	11,63	14.591,55	9,42
	K40	13.025,79	0,28	92.731,21	2,12	81.778,02	1,84
	GTX980	14.483,09	0,30	156.170,89	3,05	115.593,41	2,39
16	GT710M	1.426,63	0,95	17.818,94	11,63	14.418,49	9,58
	K40	13.189,64	0,30	94.663,11	2,16	83.015,56	1,88
	GTX980	14.873,42	0,31	159.565,91	3,15	118.465,44	2,48
32	GT710M	1.412,22	0,95	17.703,23	11,51	14.371,11	9,62
	K40	13.315,25	0,30	95.158,73	2,18	83.014,95	1,90
	GTX980	15.141,89	0,31	161.319,38	3,18	119.759,27	2,52
64	GT710M	1.402,78	0,94	17.703,23	11,56	14.371,22	9,55
	K40	13.421,77	0,31	95.659,56	2,19	83.489,94	1,91
	GTX980	15.420,24	0,32	162.885,59	3,22	120.253,00	2,55
128	GT710M	1.392,88	0,94	17.640,23	11,52	14.334,47	9,49
	K40	13.421,77	0,31	95.659,56	2,20	83.488,03	1,92
	GTX980	15.534,46	0,32	163.111,82	3,24	121.753,38	2,57
256	GT710M	1.391,61	0,93	17.625,97	11,54	14.293,97	9,36
	K40	13.315,25	0,31	95.450,24	2,20	83.487,01	1,93
	GTX980	15.902,57	0,32	164.252,46	3,25	121.751,89	2,58
512	GT710M	1.385,92	0,91	17.569,18	11,51	14.273,29	9,34
	K40	13.236,46	0,31	95.554,79	2,20	83.586,34	1,93
	GTX980	15.864,98	0,32	164.252,46	3,26	121.963,37	2,58
1024	GT710M	1.384,27	0,90	17.635,59	11,51	14.251,05	9,31
	K40	13.206,51	0,30	95.429,36	2,20	83.635,90	1,93
	GTX980	16.468,43	0,33	165.118,47	3,27	129.976,29	2,58

Tabela 3. Tabela de Flops/s das operações: Scalar, Sigmoid e Tanh

As operações ADD, DOT e MUL trabalham com três arrays, onde um array de destino recebe o resultado da operação realizada entre os array de entrada. As operações Scalar, Sigmoid e Tanh atuam sobre duas matrizes ($m \times n$). A granularidade destas operações também é pequena e, portanto, não permitem as GPUs alcançarem um bom desempenho, considerando seu poder computacional teórico.

Em relação ao valor absoluto de flops/s alcançados, a placa da arquitetura mais recente apresenta o melhor resultado em todas as operações. E, como esperado, a GPU utilizada em notebooks (GT710M) apresenta a menor capacidade de processamento e a K40 obteve um resultado intermediário. Por outro lado, observando a eficiência destas GPUs em relação aos picos teóricos, foi possível determinar que a operação de DOT mostrou-se com uma granularidade maior, chegando atingir mais de 80% de eficiência nas GPUs GT710M e GTX980.

A K40 não apresentou a mesma eficiência na operação de DOT, atingindo uma eficiência em torno de 37%. A fim de explicar este resultado, executamos novamente a operação de DOT com 6MB nas três GPUs, coletando métricas relacionadas as instruções

e duas (i - falha no fetch de instruções [2] e ii - dependência das instruções [3]) mostraram valores bem diferentes entre a GT710M e GTX980 em relação a K40.

A K40 apresentou 35% de falha no fetch de instruções e 24% na dependência na execução destas instruções. A GT710M e GTX980 obtiveram uma taxa menor, ficando em torno de 12% para falha no fetch de instruções e 10% na dependência da execução destas instruções.

3. Conclusão

Neste trabalho fizemos a avaliação da biblioteca Theano no uso da GPU com operações básicas de álgebra linear contidas em RNPs. Para avaliar as GPUs, foi utilizado o CUDA Profiler, o que permitiu coletar os Flops de cada operação em cada GPU testada.

Os testes apresentaram resultados quantitativos do uso da GPU. Nas operações onde há uma granularidade fina do problema, a GPU gastou mais com a transferência dos dados do que com a computação em si. A operação DOT, que tem uma maior granularidade, alcançou 80% do pico teórico de flops da GT710M e GTX980, a K40 não apresentou a mesma performance, alcançando apenas 30% do pico teórico.

O gargalo que a K40 mostrou na execução do *kernel* sugere uma investigação detalhada (trabalhos futuros), pois fatores como a arquitetura da placa, ineficiência do compilador e quantidade de cores por processador podem influenciar causar dependência e retardos no fetch das instruções.

Referências

Bergstra, J., Breuleux, O., Bastien, F., Lamblin, P., Pascanu, R., Desjardins, G., Turian, J., Warde-Farley, D., and Bengio, Y. (2010). Theano: A cpu and gpu math compiler in python. In *Proc. 9th Python in Science Conf*, pages 1–7.

Kovács, Z. L. (2002). *Redes neurais artificiais*. Editora Livraria da Fisica.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

Mayanglambam, S., Malony, A. D., and Sottile, M. J. (2010). Performance measurement of applications with gpu acceleration using cuda. *Parallel Computing: From Multicores and GPU's to Petascale, B. Chapman et al., IOS Press*.

Owens, J. D., Houston, M., Luebke, D., Green, S., Stone, J. E., and Phillips, J. C. (2008). Gpu computing. *Proceedings of the IEEE*, 96(5):879–899.

Sainath, T. N., Mohamed, A.-r., Kingsbury, B., and Ramabhadran, B. (2013). Deep convolutional neural networks for lvcsr. In *Acoustics, speech and signal processing (ICASSP), 2013 IEEE international conference on*, pages 8614–8618. IEEE.

[2] `--metrics stall_inst_fetch`
[3] `--metrics stall_exec_dependency`