International Journal of Trend in Scientific Research and Development (IJTSRD)
International Open Access Journal | www.ijtsrd.com
ISSN No: 2456 - 6470 | Volume - 2 | Issue – 6 | Sep – Oct 2018

Energy Efficient For Cloud Based GPU Using DVFS With Snoopy Protocol

Prof. Avinash Sharma[1], Anchal Pathak[2]
[1]Associate Professor, [2]PG Scholar
Department of CSE, Millennium Institute of Technology and Science,
Bhopal, Madhya Pradesh, India

ABSTRACT
GPU design trends show that the register file size will continue to increase to enable even more thread level parallelism. As a result register file consumes a large fraction of the total GPU chip power. It explores register file data compression for GPUs to improve power efficiency. Compression reduces the width of the register file read and writes operations, which in turn reduces dynamic power. This work is motivated by the observation that the register values of threads within the same warp are similar, namely the arithmetic differences between two successive thread registers is small. Compression exploits the value similarity by removing data redundancy of register values. Without decompressing operand values some instructions can be processed inside register file, which enables to further save energy by minimizing data movement and processing in power hungry main execution unit. Evaluation results show that the proposed techniques save 25% of the total register file energy consumption and 21% of the total execution unit energy consumption with negligible performance impact.

Keyword: *GPU, Energy Consumption, Data Redundancy, Compression, Cloud Computing, General Purpose Graphics Processing Units.*

1. INTRODUCTION

Performance and energy efficiency are major concerns in cloud computing data centers. More often, they carry conflicting requirements making optimization a challenge. Further complications arise when heterogeneous hardware and data center management technologies are combined. For example, heterogeneous hardware such as General Purpose Graphics Processing Units (GPGPUs) improved performance at the cost of greater power consumption while virtualization technologies improve resource management and utilization at the cost of degraded performance.

Distributed computing is picking up fame because of economies of scale and high adaptability. In any case, because of incessant equipment invigorating and substitutions, the basic design winds up heterogeneous. Heterogeneity results from various processor designs, memory, stockpiling, control administration, and stage structures. All the more as of late, high-throughput models like GPUs are turning into a staple in distributed computing offices and in this manner adding another level of heterogeneity to the server farm. In planning, critical wasteful aspects can result if heterogeneity isn't thought about since workloads run all the more productively on equipment with coordinating qualities. Be that as it may, heterogeneity in cloud server farms isn't the main worry with new designs; for instance, GPU control utilization is more noteworthy contrasted with multi-center models. GPUs expend generous measure of intensity as an end-result of execution, in this manner raising worry about power and vitality utilizations in server farms. It has been evaluated that server farms in the US expended 100 billion kWh in 2011 at a cost of $7.4 billion every year. An ongoing report demonstrated that overall utilization for 2010 surpassed 250 billion kWh, just about 1.5% of the world's aggregate power utilization [16]. To grow administrations and bolster developing applications, huge endeavors are being applied to move conventional private High Performance Computing (HPC) to the cloud. Developing applications like remote workstations and cloud gaming are cases of

promising markets. The primary obstacles confronting these business sectors are execution and administration issues. The execution concern is because of virtualization overhead. Virtualization is fundamentally used to enhance usage in server farms through combination yet additionally has benefits in administration, security, and live relocation. Virtualization overhead is expected to the hypervisor (additionally called a Virtual Machine Monitor, VMM) that goes about as an interface amongst applications and the hidden equipment. Answers for enhance execution depend on bypassing the hypervisor to accomplish close local task [7]. Bypassing the hypervisor furnishes VMs with guide access to I/O and equipment and enhances execution altogether. This engineering enables a GPU to be appended or gone through to a VM with least punishment on execution.

Cloud specialist organizations like Amazon Elastic Compute Cloud (EC2), Microsoft Windows Azure and Google Compute Engine are putting forth distributed computing administrations and frameworks at scales that have never been conceivable. For instance, Amazon EC2 furnishes clients with levels called occurrences custom fitted to the clients' necessities regarding virtual figure control, memory measure, stockpiling, organize transmission capacity, and bunch GPU cases. These assets are designated to the client straightforwardly, powerfully and on request. Be that as it may, in such conditions, noteworthy wasteful aspects can result if applications are not mapped to the best fitting stage. GPUs and virtualization advances have been very much examined and assessed independently. We consolidate those two advances to plan and execute, on genuine equipment, a system for upgrading virtual asset utilization and expanding vitality productivity of servers. Since assets are allotted powerfully, two issues are of concern: server control spending plan and provisioning of assets.

By profiling differing cloud workloads for control utilization and execution, we propose strategies to relieve the impacts of utilizing GPUs in server farms. We make the accompanying commitments:

- We demonstrate that over-provisioning GPU VMs with virtual CPUs (VCPUs) does not enhance execution and squanders assets.
- To enhance execution of multi-strung applications, we propose to redistribute VCPUs to VMs that can profit by the extra assets.
- To lessen server control spending plan and acquire greatest execution, we propose to reallocate obtained capacity to GPU VMs.
- By joining the two strategies, we demonstrate that server control spending plan can be diminished while keeping up execution of multi-strung VMs.

2. BACKGROUND

Vitality and execution of parallel frameworks are an expanding worry for new substantial scale frameworks. Research has been produced because of this test pointing the maker of more vitality proficient frameworks. In this unique circumstance, this paper proposes streamlining strategies to quicken execution and increment vitality effectiveness of stencil application utilized in conjunction to calculation and GPU memory qualities. The enhancements we created connected to GPU calculations for stencil applications accomplish an execution change of up to 44.65% contrasted and the read-just form. (Matheus S. Serpa, Emmanuell D. Carreño, Jairo Panetta, Jean-François, 2018)

Vitality proficiency has turned out to be one of the best plan criteria for current processing frameworks. The Dynamic Voltage and Frequency Scaling (DVFS) have been broadly received by smart phones and cell phones to moderate vitality, while the GPU DVFS is still at a specific early age. This paper goes for investigating the effect of GPU DVFS on the application execution and power utilization, and besides, on vitality protection. (Xinxin Meia, Qiang Wanga, Xiaowen Chua, 2017)

This paper depicts blame tolerant structure for dispersed stencil calculation on cloud-based GPU bunches. It utilizes pipelining to cover the information development with calculation in the radiance district and additionally parallelizes information development inside the GPUs. Rather than running stencil codes on customary bunches and supercomputers, the calculation is performed on the Amazon Web Service GPU cloud, and uses its spot occasions to enhance cost-proficiency. (Jun Zhou, Yan Zhang and Weng-Fai Wong, 2017)

GPU configuration patterns demonstrate that the enroll document size will keep on increasing to empower considerably more string level parallelism. Thus enroll record expends an extensive portion of the aggregate GPU chip control. This paper investigates enroll document information pressure for GPUs to

enhance control productivity. Pressure lessens the width of the enlist document read and composes tasks, which thusly decreases dynamic power. (Sangpil Lee, Keunsoo Kim, Gunjae Koo, Hyeran Jeon, Murali Annavaram and Won Woo Ro, 2016)

3. PROBLEM IDENTIFICATION

The recognized issue in existing work is as per the following:

1. During read and compose process in enlist record, execution time is high, because of this reason GPU misfortune influence productivity.
2. The execution is corrupted and execution time is high, when pipeline stages progress toward becoming increment.
3. When pipeline stages increment then GPU speed might be diminished. Pipeline may enhance errand execution.
4. Recovery overhead high because of expanded pipeline stages.

4. METHODOLOGY

The method is Dynamic Voltage Frequency Scaling with Snoopy Protocol (DVFS-SP), which is described through following point.

[b, f] = DVFS-SP (T)

// b is block in task, f is frequency level in task and T is the set of task

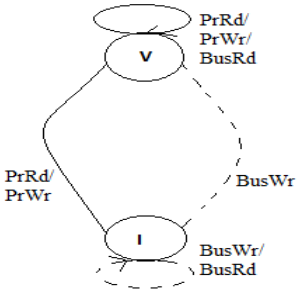
Input: Task set T with transform task period
Output: Assign speed levels to h schedule
Step 1: Whenever task complete early
Step 2: Compute the critical speed S_c using Critical Speed ()
Step 3: if $U_c^r \leq 1$ then assign S_c to the whole h schedule
Step 4: Else $\varepsilon^c = \left(U_c^r - 1\right) \times P_n$
Step 5: $\delta = \frac{\varepsilon^c \times S_{c-1}}{S_{c-1} - S_c}$

Now check 2 states of energy function in each cache
Step 6:

1. Valid state: Block can be read and written
2. Invalid state: Block has no data

Step 7: Write invalid data all other cache copy that is each task can have multiple simultaneous reader or block but write invalid data them
Step 8: Event can be issued from processor or observed on the bus through following signal

PRd = Process Read, PWr = Process Write, BusRd = Bus Read, BusWr = Bus Write



Step 9: Assign Speed Sc of each block (Valid State) to the interval $[0, P_k - \delta]$
Step 10: Assign Speed Sc of each block (Valid State) to the interval $[P_k - \delta, P_k]$
Step 11: Update b_i and f_i of each task.

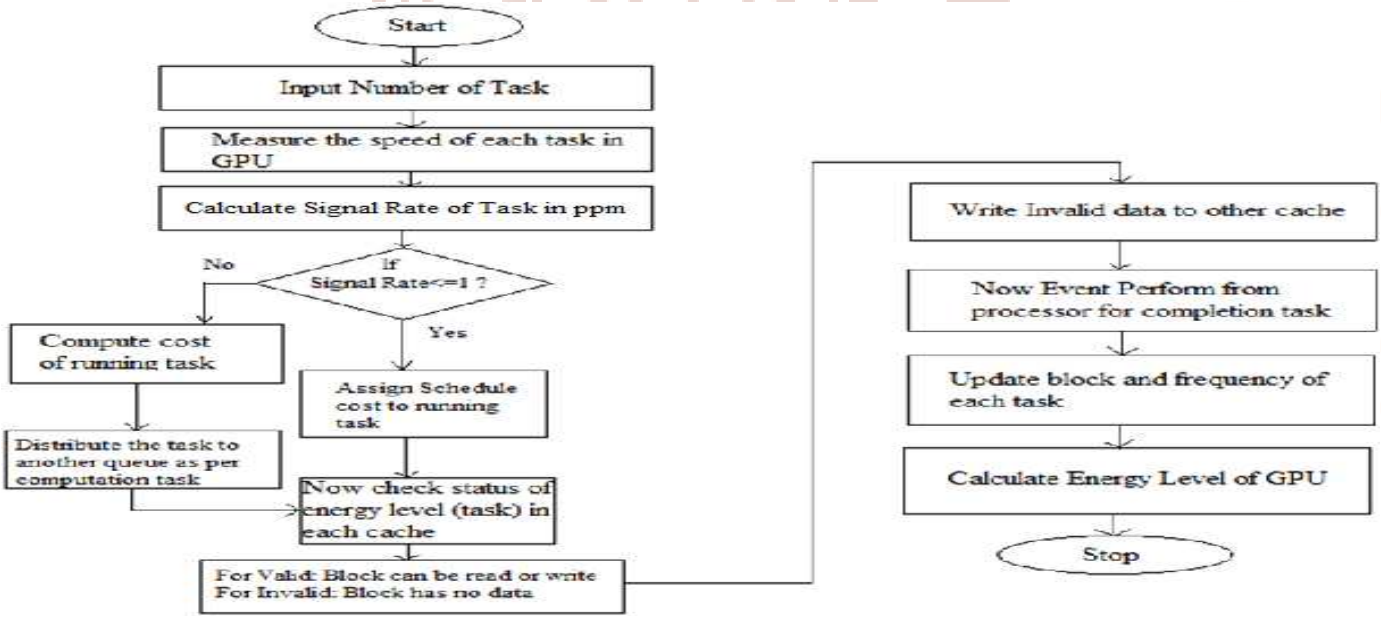


Figure 1: Flowchart of Dynamic Voltage Frequency Scaling with Snoopy Protocol (DVFS-SP)

5. RESULTS AND ANALYSIS

The analysis performs on basis of radius and chunk size. The execution time of the pipeline stages are corresponding to piece size and range. We profiled the execution time of each phase with various lump sizes and sweeps on 27 hubs masterminded in a 3 x 3 x 3 network, each having an issue size of 512x512x1024. We took the profile of the hub at position (1; 1; 1) in the hub lattice as it has correspondence with each of the six neighbors.

Table 1: Execution time (ms) of Copy In Data in between of FTCS [3] and DVFS-SP (Proposed) where r=1

CHUNK SIZE	FTSC[3]	DVFS-SP(PROPOSED)
32	1117	1089
64	2450	2213
128	6225	6081

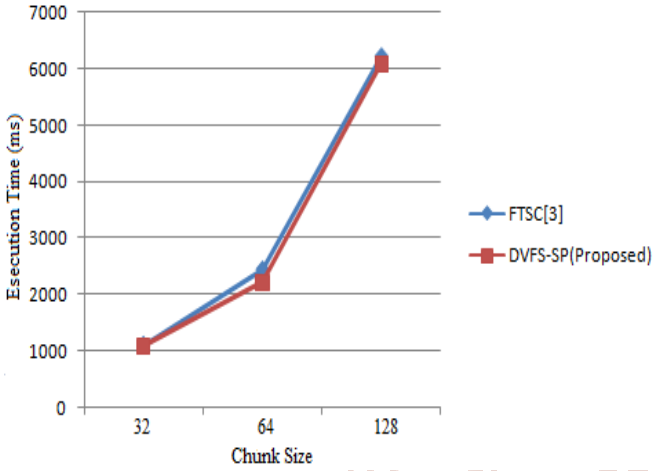


Figure 2: Analysis of execution time of Copy In Data in between FTSC [3] and DVFS-SP (Proposed) where radius = 1

Above figure shows the execution time of each stage in Copy In Data stage with different chunk size c and fixed radius of r = 1. From the result, we can see that the execution time is proportional to chunk size c.

Table 2: Execution time (ms) of computation in between of FTSC [3] and DVFS-SP (Proposed) where r=1

CHUNK SIZE	FTSC[3]	DVFS-SP(PROPOSED)
32	656	407
64	1307	1182
128	2688	2396

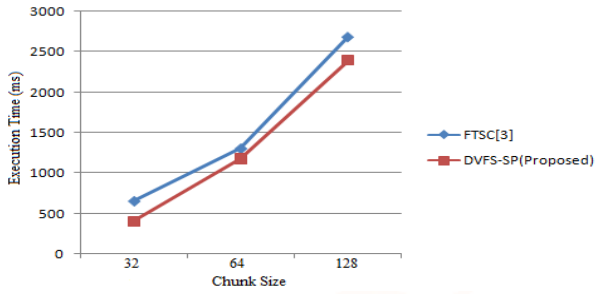


Figure 3: Analysis of execution time of Computation in between FTCS [3] and DVFS-SP (proposed) where radius = 1

Above figure demonstrates the execution time of each phase in Computation arrange with various lump estimate c and settled sweep of r = 1. From the outcome, we can see that the execution time is corresponding to lump estimate c.

Table 3: Execution Time (ms) of Copy Out Data in between of FTSC [3] and DVFS-SP (Proposed) where r=1

Chunk Size	FTSC[3]	DVFS-SP(Proposed)
32	140	128
64	251	226
128	478	398

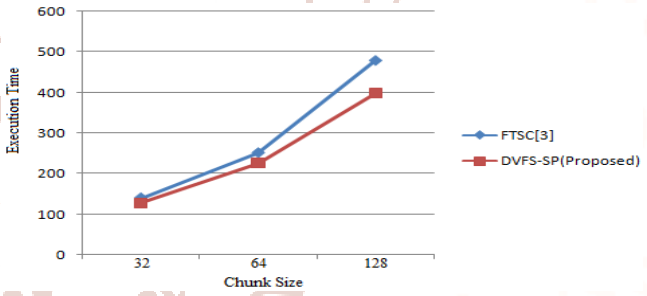


Figure 4: Analysis of execution time of Copy Out Data in between FTCS [3] and DVFS-SP (Proposed) where radius = 1

Above figure demonstrates the execution time of each phase in Copy out Data organize with various piece measure c and settled span of r = 1. From the outcome, we can see that the execution time is relative to piece measure c.

Table 4: Execution time (ms) of Copy In Data in between of FTCS [3] and DVFS-SP (Proposed) where c=128

RADIUS	FTSC[3]	DVFS-SP(PROPOSED)
1	6225	5719
2	7522	6837
3	8959	8009
4	11393	10038

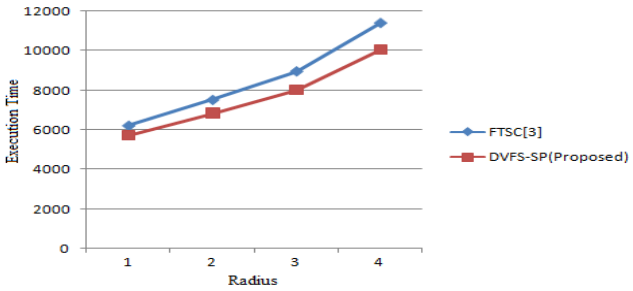


Figure 5: Analysis of execution time of Copy In Data in between FTCS [3] and DVFS-SP (Proposed) where c = 128

Above figure shows the execution time of each stage in Copy In Data stage with different radius r and fixed chunk size of c = 128. From the result, we can see that the execution time is proportional to chunk size r.

6. CONCLUSIONS AND FUTURE WORK

The GPU DVFS for vitality protection. We center around the most up and coming GPU DVFS innovations and their effect on the execution and power utilization. We outline the philosophy and the execution of existing GPU FTCS models. As a rule, the nonlinear demonstrating procedure, for example, the ANN and the changed SLR, has better estimation exactness. What's more, we direct certifiable DFS/DVFS estimation probes the NVIDIA Fermi and Maxwell GPUs. The trial result proposes that both the center and memory recurrence impact the vitality utilization altogether. Utilizing the most noteworthy memory recurrence would dependably save vitality for the Maxwell GPU, which isn't the situation on the Fermi stage. As per the Fermi DVFS tests, downsizing the center voltage is crucial to monitor vitality. Both the study and the estimations spotlight the test of building a precise DVFS execution show, and besides, applying proper voltage/recurrence settings to moderate vitality. We go away these for our outlook investigation. In addition, it is another essential heading to coordinate the GPU DVFS into the substantial scale bunch level power administration later on. It will intrigue investigate how to adequately join GPU DVFS with other vitality preservation systems, for example, errand planning, VM solidification, control execution mediating and runtime control observing.

Bunch size and range is corresponding to each other the season of all the three phases (Copy In Data, Computation and Copy Out Data) are relative to the lump estimate (c) and sweep (r). At the point when the sweep is expanded, the calculation and correspondence workload increments at various rates. The technique perform on NVIDIA based GPU. The power effectiveness is kept up to 256 pieces measure and furthermore decrease excess checkpoint in enroll record. The execution of GPU is better for various pipelines stages like pieces size and sweep of group information record. The execution time for various pieces sizes and sweeps are essentially up to 31 % and 8% separately. The execution time of GPU might be lessening fundamentally and GFLOPS is enhancing for information calculation in cloud. Subsequently, recuperation overhead might be diminished.

1. GPU implementation might be enhanced through heaviness plan of put away in sequence in cloud.
2. Computation process in cloud can be diminishing through MIMD processor setup in server farm.
3. GPU execution can be investigated if there should be an occurrence of dispersed framework.

7. REFERENCES

1. Matheus S. Serpa, Emmanuell D. Carreño, Jairo Panetta, Jean-François, “Improving Performance and Energy Efficiency of Stencil Based Applications on GPU”, IEEE Conference on Green Energy, June, 2018.
2. Xinxin Meia, Qiang Wanga, Xiaowen Chua,b, “A survey and measurement study of GPU DVFS on energy conservation”, Elsivier Journal of Digital Communications and Networks, 2017.
3. Jun Zhou, Yan Zhang and Weng-Fai Wong, “Fault Tolerant Stencil Computation on Cloud-based GPU Spot Instances”, IEEE Transactions on Cloud Computing, 2017.
4. Sangpil Lee, Keunsoo Kim, Gunjae Koo, Hyeran Jeon, Murali Annavaram and Won Woo Ro, “Improving Energy Efficiency of GPUs through Data Compression and Compressed Execution”, Journal of IEEE Transactions on Computers, August 2016.
5. S. Huang, S. Xiao and W. Feng, “On the Energy Efficiency of Graphics Processing Units for Scientific Computing”, IEEE Conference on Cloud Computing, 2016.
6. Qing Jiao, Mian Lu, Huynh Phung Huynh, Tulika Mitra, “Improving GPGPU Energy-Efficiency through Concurrent Kernel Execution and DVFS”, IEEE/ACM International Symposium on Code Generation and Optimization, 2015.

7. Amer Qouneh, Ming Liu, Tao Li, "Optimization of Resource Allocation and Energy Efficiency in Heterogeneous Cloud Data Centers", 44th International Conference on Parallel Processing, 2015.
8. Mohammed Sourouri, Johannes Langguth, Filippo Spiga, Scott B. Baden and Xing Cai, "CPU+GPU Programming of Stencil Computations for Resource-Efficient Use of GPU Clusters", IEEE 18th International Conference on Computational Science and Engineering, 2015.
9. Ashish Mishra, Nilay Khare, "Analysis of DVFS Techniques for Improving the GPU Energy Efficiency", Open Journal of Energy Efficiency, 2015.
10. Sparsh Mittal, "A Survey of Techniques For Improving Energy Efficiency in Embedded Computing Systems", International Journal of Computer Aided Engineering and Technology, 2014.
11. Juan M. Cebrian, Gines D. Guerrero and Jose M. Garcia "Energy Efficiency Analysis of GPUs", IEEE Journal of Cloud Computing, 2014.
12. Sparsh Mittal, Jeffrey S. Vetter, "A Survey of Methods for Analyzing and Improving GPU Energy Efficiency", ACM Computing Surveys, 2014.
13. Kai Sun, Zhikui Chen, Jiankang Ren, Song Yang, Jing Li, "M2C: Energy Efficient Mobile Cloud System for Deep Learning", IEEE Conference on Cloud Computing, 2014.
14. Jishen Zhao, Guangyu Sun, Gabriel H. Loh, Yuan Xie, "Optimizing GPU Energy Efficiency with 3D Die-Stacking Graphics Memory and Reconfigurable Memory Interface", ACM Transactions on Architecture and Code Optimization, December 2013.
15. Xiaohan Ma, Marion Rincon, and Zhigang Deng, "Improving Energy Efficiency of GPU based General-Purpose Scientific Computing through Automated Selection of Near Optimal Configurations", IEEE Conference on Parallel Computing, 2011.
16. C.-M. Liu, T. Wong, E. Wu, R. Luo, S.-M. Yiu, Y. Li, B. Wang, C. Yu, X. Chu, K. Zhao, et al., SOAP3: ultra-fast GPU-based parallel alignment tool for short reads, Bioinformatics 28 (6) (2012) 878–879.
17. K. Zhao, X. Chu, G-BLASTN: accelerating nucleotide alignment by graphics processors, Bioinformatics 30 (10) (2014) 1384–1391.
18. X. Chu, C. Liu, K. Ouyang, L. S. Yung, H. Liu, Y.-W. Leung, Perasure: a parallel cauchy reed-solomon coding library for GPUs, in: Proceedings of the IEEE International Conference on Communications, London, UK, 2015, pp. 436–441.
19. Q. Li, C. Zhong, K. Zhao, X. Mei, X. Chu, Implementation and analysis of AES encryption on GPU, in: Proceedings of the Third IEEE International Workshop on Frontier of GPU Computing, Liverpool, UK, 2012.
20. X. Chu, K. Zhao, Practical random linear network coding on GPUs, in: Proceedings of the 8th International Conferences on Networking, IFIP, Archen, Germany, 2009.
21. Y. Li, K. Zhao, X. Chu, J. Liu, Speeding up K-means algorithm by GPUs, in: Proceedings of the 10th IEEE International Conference on Computer and Information Technology, Bradford, UK, 2010, pp. 115–122.
22. R. Raina, A. Madhavan, A. Y. Ng, Large-scale deep unsupervised learning using graphics processors, in: Proceedings of the 26th ACM Annual International Conference on Machine Learning, 2009, pp. 873–880.
23. A. Coates, B. Huval, T. Wang, D. Wu, B. Catanzaro, N. Andrew, Deep learning with COTS HPC systems, in: Proceedings of the 30th International Conference on Machine Learning, 2013, pp. 1337–1345.
24. Q. V. Le, Building high-level features using large scale unsupervised learning, in: Proceedings of IEEE International Conference on Acoustics, Speech and Signal Processing, 2013, pp. 8595–8598.
25. D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of Go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.