First question arises is that which part of the model should be accelerated for better improvement of the whole model. In their case, they found that WSM5 is the highly computational intensive module in WRF. In a single invocation, 2400 fp multiply-equivalent operations are required in WSM5. As the WRF model is originally implemented in fortran90, they ported the WSM5 module on Compute Unified Device Architecture (CUDA).[1]

Parallelisation. In WRF, the whole atmosphere in a particular geographical region is represented with the help of 3- Dimentional grid as shown in figure 1.

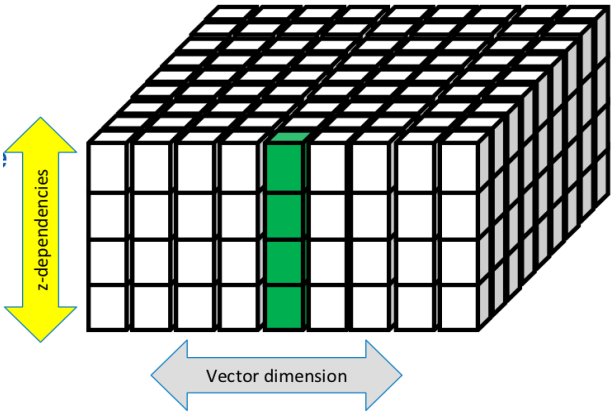

Figure 1: Dependencies in Z direction

The Cartesian Coordinate system in this grid is equally spaced in horizontal directions *x* and *y*. The third dimension *z* shows the vertical level in a pressure based terrain following coordinates. In the column physics, the computations are processed along vertical access i.e. *z* axis. So, this shows that the *x* and *y* axis are fully independent to each other as dependencies are shown over *z*-axis. This shows that each column can be processed individually; this helps in the parallelisation process where each column is processed by different processors.

3. Results

The *Storm of the Century* (SOC) test is used to analyse the performance of the WSM5 GPU implementation. SOC covers 30km area with 27 vertical levels. It contains 115,000 cells which covers grid of 71x58 at horizontal resolution. The whole implementation is tested on different configuration for performance comparison which includes 2.8 GHz Pentium machine, 1.9 GHz Power5, and 3.0 GHz Xeon processor and GPU. Figure 2 shows the performance comparison on different platforms.

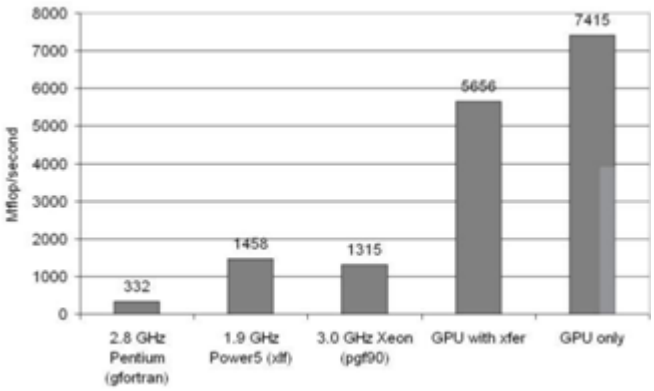

Figure 2: Performance Comparisons [1]

Figure 2 show that GPU accelerate the WSM5 module with more than 17x speedup. Use of GPU increased the total speedup of WRF model by the factor of 1.3x. This work is also shown in the NVIDIA official website. The statistic of NVIDA official website is shown below, which shows that speedup is 1.25x.

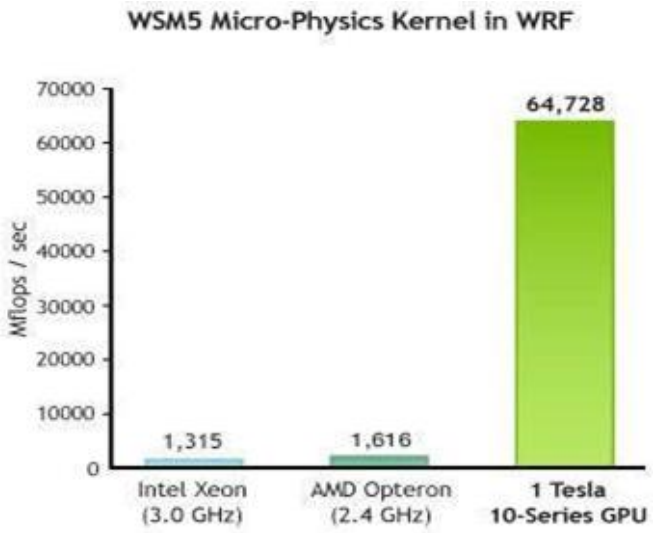

Figure 3: WRF acceleration using CUDA[7]

4. Implementation of WDM5 module on Many-Core GPU

Similar approach is done by Jun Wang, Mitchell D. Goldberg and team in 2011[8]. WDM5 is the WRF Double Moment-5 module of WRF model, which is used to predict mixing ration of hydrometeors and their concentrations for warm rain species including rain and clouds. Jun Wang and team published a paper in 2011, in which they ported the original Fortran code of WDM5 on the highly parallel CUDA C program.

Analysis. The experiments were performed on NVIDIA Fermi personal supercomputer with 2048-cores and 3.20GHz Intel core i7 970 CPU with 12 cores and two NVIDIA GeForce GTX 590 GPUs. The specifications are shown in table below:

Table 1: NVIDIA GeForce GTX 590 Specifications

Number of thread processors	2*512
Frequency of thread processors	1.215GHz
Single Precision Floating Point Capability	512FMAops/clock
Double Precision Floating Point Capability	256FMAops/clock
Global Memory	2*1.5GBGDDR5
Global Memory Bandwidth	163.86GB/s
Shared Memory Per streaming multi Processor(SM)	64KB (L1/shared memory)
Number of 32-bit register per SM	32K
Max power consumption	365W

With the above specifications and CUDA implementation of WDM5 module, the total speedup obtained over the CPU based single threaded code was up to 147x with asynchronous data transfer. The results are shown in the table below:

Table 2: Speedup results of WDM5 using Asynchronous Data Transfer

	Asynchronous+ Non-coalesced	*Asynchronous+ coalesced*
CPU time(ms)	*19341*	*19241*
GPU+I/O (ms)	*131*	*160*
Speed up with I/O	*147x*	*121x*

This work can be further improved with the scalable implementation on GPU.

5. Improved GPU implementation of Numerical Weather prediction

As discussed earlier, Michlakes and Manish Vachharajani implemented the WSM5 module of WRF model on GPU; the improvements are done on this implementation in 2012 by Jarno Mielikainen and team [9]. They also implemented the WSM5 Fortran module in CUDA, but with some improvements. Some improvements includes that they scalarized more temporary arrays. The use of coalesced memory access is one of the most important improvements. Another improvement done by them is the use of asynchronous memory transfer. This implementation achieved the speedup of 206x with I/O and 389x without I/O using a single GPU. Earlier the speedup was 17x only. The speedup without I/O increases linearly with the increase in number of GPUs. Maximum number of GPUs used here are 4, which shows the speedup of 1556x. Thus, this shows that GPU implementation is the feasible way to accelerate the Weather Research and Forecasting Computation.
All these researches are done by implementing different modules of WRF on GPU using CUDA.

6. Implementation of WRF on GPU using OpenCL

So far we have seen the implementation of WRF model on GPU using CUDA, and we also conclude that the GPU implementation is the feasible way to accelerate the performance of the WRF model. So, now a question arises that, what else we can do to optimize the GPU implementation of WRF model?

In [10], a comprehensive performance comparison of CUDA and Open Computing Language (OpenCL) is done. Like CUDA, OpenCL is also a parallel programming framework used to write programs which are executed on different heterogeneous platforms which consists of CPUs, GPUs, digital signal processors (DSPs) and other processors. It has been adopted by Intel, Qualcomm, Apple, Advanced Micro Devices (AMD), NVIDIA etc.

The authors have concluded that OpenCL can be a good alternative to CUDA. It is inconvenient for programmers that every programming framework has different application development methods. The programmers are forced to learn the new languages which quickly become outdated. So, this leads to the establishment of OpenCL, which gives the programmers a freedom to use various platforms to implement their parallel programs. While comparing the performance of CUDA and OpenCL, performance of CUDA is slightly better than OpenCL, but OpenCL gives the power of portability, which is most important. So, from this it can be concluded that OpenCL can be a good alternative of CUDA for GPU implementation of WRF model.

This approach is implemented by Wim Vanderbauwhede and Tetsuya Takemi in the year 2013 [11]. The authors have done the profiling of WRF model and analysed that physics and dynamics modules are together taking more than 85% of total running time. As previously in [1], works are done on physics module, the authors have choose the dynamic module, so they choose the scalar advection module to port on OpenCL. After porting to OpenCL, the compute performance of the ported code is compared with the original WRF code on Intel CPU and the code is compiled with PGI compiler. The results show that the GPU implemented code shows the speedup of 7x. Previously, in [1], the speedup was 1.3x approx. The results show that only porting the original code to OpenCL, gives the significant speedup in the multicore CPU environment, without using any GPU.

So, for the future work, we can say that, implantation of different modules of WRF model on GPU using OpenCL can also be feasible. Further, we can think of implementing different physics modules in OpenCL to get better speedup.

7. Conclusion

This paper gives a brief review of work related to the GPU implementation of Weather Research and Forecast model. From the work done in past few years, it can be concluded that GPU implementation is a feasible way to accelerate the performance of WRF model.

The GPU implementation using CUDA and OpenCL both are feasible and worthwhile. On the basis of performance, the performance of CUDA is slightly better than OpenCL, but the limitation in CUDA code is the lack of portability. CUDA codes can we executed on NVIDIA GPU cards only. This makes the OpenCL implementation more preferable. OpenCL programs can be run on various platforms. For the future work we can think of implementing the different modules of WRF on GPU using OpenCL.

References

[1] J. Michalakes and M. Vachharajani, "GPU acceleration of numerical weather prediction," in Parallel and Distributed Processing, 2008. IPDPS 2008. IEEE International Symposium on. IEEE, 2008, pp. 1–7.
[2] C. Ahrens, Meteorology Today, 9th ed. Independence, KY: Brooks Cole, 2009.
[3] R. Bradley and P. Jones, *Climate Since A.D. 1500*. London, U.K.: Routledge, 1992.
[4] J. Nickolls and W. J. Dally, "The GPU computing era," *IEEE Micro*, vol. 30, no. 2, pp. 56–69, 2010, doi 10.1109/MM.2010.41.
[5] The Weather Research & Forecasting Model Website: www.**wrf-model**.org/
[6] J. Michalakes, J. Hacker, R. Loft, M. O. McCracken, A. Snavely, N. Wright, T. Spelce, B. Gorda, and B. Walkup. Wrf nature run. In proceedings of the 2007 ACM/IEEE conference on Supercomputing, pages 1–6, 2007.
[7] NVIDIA Official Website Weather Forecasting Results: http://www.nvidia.com/object/weather.html
[8] J. Wang, B. Huang, A. Huang, and M. D. Goldberg, "Parallel computation of the weather research and forecast (wrf) wdm5 cloud microphysics on a many-core gpu," in Parallel and Distributed Systems (ICPADS), 2011 IEEE 17th International Conference on, dec. 2011, pp. 1032-1037.
[9] J. Mielikainen , H. Bormin , H. A. Huang and M. D. Goldberg "Improved GPU/CUDA based parallel weather and research forecast (WRF) single moment 5-class (WSM5) cloud microphysics", *IEEE J. Sel. Topics Appl. Observ. Remote Sens.*, vol. 5, no. 4, pp.1256 -1265 2012
[10] J. Fang , A. L. Varbanescu and H. Sips "A comprehensive performance comparison of CUDA and OpenCL", *Proc. Int. Conf. ParallelProcess.*, pp.216 - 225 2011.
[11] W. Vanderbauwhede and T. Takemi, "An investigation into the feasibility and benefits of GPU/multicore acceleration of the weather research and forecasting model," in High Performance Computing and Simulation (HPCS), 2013 International Conference on, 2013, pp. 482-489.