ACCELERATING BIG DATA PROCESSING CHAIN IN IMAGE INFORMATION MINING USING A HYBRID HPC APPROACH

Kuldeep R. Kurte[1], Ujwala M. Bhangale[1], Surya S. Durbha[1], Roger L. King[2], Nicolas H. Younan[2]

[1]Centre of Studies in Resources Engineering, IIT Bombay, Powai, Mumbai - 400076, India,
[2]Department of Electrical and Computer Engineering, Mississippi State University, Starkville, USA
[1]{kuldeep.iitb, ujwala.bhangale, sdurbha}@iitb.ac.in, [2]{rking,younan}@ece.msstate.edu

ABSTRACT

The recent development in sensor technology shows the unprecedented growth of Remote Sensing (RS) data archives-*Big Data*. However, this growth in RS archives has resulted in many processing challenges. The three V's of big data- Volume, Velocity and Variety is highly relevant in situations such as flood, earthquake disaster, where real/near real time processing of data from different RS data sources is vital to deploy rescue operations. In this work, we have demonstrated a high-performance analytics approach- Message Passing Interface (MPI) along with the emerging Graphics Processing Units (GPUs) (i.e. hybrid MPI+GPU) technology to overcome the big data processing limitation. The different processing/analysis stages of our Spatial Image Information Mining (SIIM) system are parallelized using the above approach. The experimental results for parallel segmentation process show the applicability of MPI+GPU hybrid approach in disaster scenario.

Index Terms— Big Data Analytics, High-Performance Analytics, Parallel Computing, MPI, GPU, Meanshift, CUDA, HPC

1. INTRODUCTION

The recent development in sensor electronics enables the RS satellites to capture data in high spatial and spectral resolutions. This is leading to the cumulative growth in remote sensing (RS) archive size- *Big Data*. This growth of the RS data is also leading to new challenges such as, rapid big data processing, big data storage techniques etc. The three V's that characterize big data i.e. Volume, Velocity and Variety are highly relevant to Earth Observation (EO) data systems. These become even more pertinent when dealing with time critical events such as disasters (e.g. floods, forest fires, earthquakes, etc.). Since, these events are dynamic and as they unfold, there is a need to respond in a rapid manner to minimize the losses to human lives and properties. From a big data perspective, the volume of data that needs to be analyzed during these events can become enormous given the variety of sensors (Spaceborn and Airborn) sending data at high temporal resolution (velocity). It can quickly become challenging to analyze such big data in a standalone desktop (CPU) kind of environments, and needs to be distributed across many computing nodes (processors, CPUs, GPUs) for rapid processing. Hence, it is imperative that emerging approaches for big data analytics, which addresses the above issues from a distributed high performance computing perspective should be adopted. This enables a rapid high velocity processing of high volumes of data coming from a variety of RS sensors for disaster data analysis.

Currently, to exploit such a massive RS data, the earth observation (EO) IIM systems have evolved over the last decade, such as, KIM [1], I3KR [2], GeoIRIS [3] etc. The EO-IIM systems allow automatic/semi-automatic extraction of the focused information from huge EO archives using various analytical techniques [4]. However, in disaster scenarios - the rapid (high velocity) identification of flooded builtup, flooded agriculture and in post-earthquake- the extraction of destructed buildings, damaged roads from voluminous archives requires big data processing techniques based on distributed high performance computing approaches to deal with processes such as image decomposition, feature extraction and classification.

The EO-IIM systems, mentioned earlier, broadly contain two modules: *Offline Module*- it contains big data analytics process chain, which is responsible for image ingestion, image decomposition, feature extraction and classification, database generation etc. and *Online Module*: which allows querying large RS image databases. This paper presents our work towards developing the rapid big data processing chain of offline module (refer section 2) in our Spatial Image Information Mining (SIIM [5]) system using high-performance analytics techniques. The traditional parallel processing techniques MPI and an emerging hardware graphics acceleration techniques- GPUs are explored and a hybrid MPI+GPU, a high performance computing (HPC) approach is used for accelerating the offline process chain.

It is experienced that, the image segmentation stage plays a central role in big data processing of RS data, which forms the basis for further analysis such as, feature extraction, classification and spatial relations extraction in

SIIM. However, for big RS archives (e.g. high resolution RS archive) segmentation becomes a time consuming process. Hence, a high-performance analytics approach (hybrid MPI+GPU) for image segmentation module is presented. The meanshift segmentation algorithm [6] is used in SIIM for decomposition of an ingested RS scene into homogeneous regions/objects.

The paper is organized as follows: Section 2 describes the big data processing chain of offline module in SIIM followed by the description of meanshift segmentation algorithm. The high-performance analytics approach based on hybrid MPI+GPU for fast segmentation is presented in Section 3. Section 4 describes the experiments carried out to show the performance of this hybrid MPI+GPU based high performance analytics approach. Section 5 concludes the paper and describes the future work.

2. OFFLINE MODULE IN SIIM

This paper presents a high-performance computing approach for accelerating the big data analytics chain of offline module in our SIIM system.

2.1. Offline process chain

The offline module in SIIM system contains four modules- image tiling, image segmentation, feature extraction and regions adjacency building (refer Fig. 1). The tilling module decomposes the ingested RS imagery into a set of tiles. These tiles are then undergoes to the segmentation process which will further decompose each image tile into a set of homogeneous regions/objects. The set of color, texture and shape features are extracted from each region of

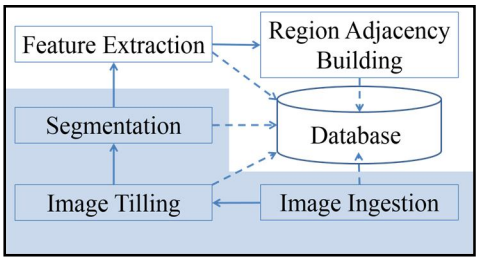

Figure 1. Big data process chain of offline module in SIIM.

a segmented image tile. Next, the Region Adjacency Graph (RAG) for a tile will be built and the generated information at each stage in offline module will be stored into database; and once prepared, the SIIM database is ready for user's queries.

It is observed that, a single big scene of high resolution RS imagery (e.g. panchromatic image from worldview-2 with 0.46 meter spatial resolution) produced large number of image tiles. The segmentation process then repeats for each image tile and later the feature extraction will be done followed by the classification for every region. Hence, this offline process chain has become very much time consuming. So there is a necessity of high-performance analytics approaches (based on both task and data level parallelism) in the offline module of SIIM.

2.2 Meanshift segmentation

The segmentation process is an important first step in high resolution RS image analysis. Especially, in disaster scenario it allows to extract affected regions/objects from the high resolution image of the disaster affected areas, which then be used to identify newly evolved LULC classes such as, flooded builtup, flooded agriculture, destructed buildings.

The meanshift is an iterative, non-parametric and robust-to-noise clustering approach, which uses the kernel density technique to identify dense regions in the feature space [7][8][6]. It uses three parameters (*SR*, *RR* and *M*). The spatial radius (*SR*, aka spatial bandwidth) specifies the spatial window to be used during segmentation. The range radius (*RR*, aka spectral bandwidth) is the spectral window to be used during segmentation. The meanshift segmentation algorithm iteratively determines the density at each point having constraint on spatial and spectral window. Then all the points will move towards the nearest dense region (using gradient ascent approach) and finally converges to points of dense region in the feature space. The parameter *M* can be used to eliminate or merge the neighboring regions smaller than size *M* (refer [6] for more information on meanshift segmentation).

As an iterative approach, the meanshift segmentation is computationally intensive, which is a bottleneck for the high resolution RS imagery, however, it is very much suitable for the parallel processing [9]. Also, the recent work presented in [10][11] describes the parallel nature of meanshift technique and its GPU based acceleration.

3. HIGH-PERFORMANCE ANALYTICS APPROACH FOR RAPID OFFLINE PROCESSING IN SIIM

The work presented in [12] lists different techniques such as, MPI, GPU, Field Programmable Gate Array (FPGA) etc. for rapid processing of large RS data. Recently, MPI is used to accelerate smart analytic techniques such as, Support Vector Machine (SVM) for image classification [13]. As mentioned earlier, this paper demonstrates the MPI+GPU based hybrid approach for rapid segmentation (meanshift) of RS image tiles.

The code snippet in Fig. 2 shows how MPI is used to utilize the available system resources such as, processors, cores per processor, GPU devices to distribute the segmentation task. The degree of parallelism (*degOfPar*) is calculated using available system resources (refer line 4 and line 10 in Fig. 2). The comments on line 4-11shows the *degOfPar* for CPU based distributed meanshift segmentation using MPI. The rank 0 MPI process is the master process, which is responsible to do the tilling of

```
1.  ……
2.  nNodes=1;
3.  nGPUsPerNode=2;
4.  degOfPar=nNodes*nGPUsPerNode;
5.  /*//for CPU based meanshift segmentation using MPI
6.  nNodes=1;
7.  nProcPerNode=2
8.  nCoresPerProc=4;
9.  nThreadsPerCore=2; // hyperthreading enabled
10. degOfPar=nNodes*nProcPerNode
11.         *nCoresPerProc*nThreadsPerCore; */
12. //MPI intitialization
13. MPI_Status stat;
14. MPI_Init(&argc,&argv);
15. MPI_Comm_size(MPI_COMM_WORLD,&numprocs);
16. MPI_Comm_rank(MPI_COMM_WORLD,&rank);

17. if(rank == 0){
18.  doTiling(/*Path of the ingested RS scene in archive*/);
19.  tileCount=getTheTilsCount();
20.  tileCountBuff[0]=tileCount;
21. //here, broadcast archive path to all slave processes
22. MPI_Bcast(tilesArchivePath, 100, MPI_INT, 0 , comm);
23.  for(i=1;i<numprocs;i++){
24.   // here, send tilecount to all slave processes
25.   MPI_Send(tileCountBuff, 1, MPI_INT, i, TAG,
    MPI_COMM_WORLD);
26.  }
27. }
28. else {
29.  /* receive archivePath of tiles from rank 0: */
30.  MPI_Recv(tileArchivePath, 1, MPI_INT, 0, TAG,
    MPI_COMM_WORLD, &stat);
31.
32.  /* receive tileCount from rank 0: */
33.  MPI_Recv(tileCountBuff, 1, MPI_INT, 0, TAG,
    MPI_COMM_WORLD, &stat);
34.  tileCount=tileCountBuff[0];
35.  nTilesPerProcess=tileCount/degOfPar+1;
36.  fromTileId=(rank-1)*nTilesPerProcess;

37.  if(fromTileId<=tileCount-1){
38.   toTileId=fromTileId+nTilesPerProcess-1;
39.   if(toTileId>tileCount-1){
40.    toTileId=tileCount-1;
41.   }
42.  }
43.  else{
44.   exit(0);
45.  }
46. //here, call segmentation function
47. //cpuMeanshift(fromTileId,toTileId, SR, RR, density);
48. gpuMeanshift(fromTileId,toTileId, SR, RR, density, rank-1);
49. }
50. ……
```

Figure 2. Code snippet showing MPI based parallel segmentation of the tiles generated after in SIIM

ingested RS imagery. Then the RS image archive path of generated tiles and *tileCount* will be sent to all slave processes using MPI_Bcast, MPI_Send (line 22, Fig. 2). Upon receiving all the information from master process, each slave process will calculate the range of tile ids to be processed based on its own rank (refer line 35-49, Fig. 2 for this logic). Fig. 3 shows this task distribution where slave processes are assigned to the available cores in the system. Each slave process will execute the GPU version of the meanshift segmentation in parallel.

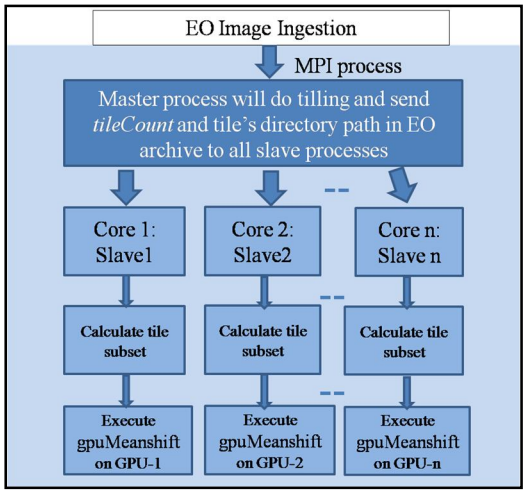

Figure 3. Diagrammatic representation of task distribution for parallel segmentation in SIIM

4. EXPERIMENTAL RESULTS

In this work, we have explored and compared both CPU implementation [14] and GPU (CUDA) implementation [15] of the meanshift segmentation algorithm for segmenting RS imagery of disaster affected area. The experiments in the presented work compares the execution timings of CPU and GPU implementations and their MPI versions i.e. cpuMeanshift, gpuMeanshift, cpuMeanshift_MPI and gpuMeanshift_MPI.

The dataset used for the experimentation is a scene captured by worldview 2 on September 16, 2014 of Srinagar flood (India). The experiments are done on a workstation which is having 2× Intel ® Xeon ® (2.40 GHz) processors, each with 4 cores and 2 threads per core (hyperthreading enabled), with system memory of 16 GB. The workstation also contains two NVIDIA's GPU devices: Tesla C2075 and GeForce GTS 450, which can be used for general purpose scientific computation. Further, an experiment (exp 6) is carried out on the GeoHPC cluster (4 nodes = 1 master + 3 compute nodes) having 6 NVIDIA Tesla K40 devices.

It is very clear from table 1 that the GPU (CUDA) meanshift segmentation (i.e. gpuMeanshift, exp 2, 3) outperforms and showed ≈84x speedup over the CPU version (i.e. cpuMeanshift, exp 1). Although the parallel MPI version of CPU meanshift (i.e. cpuMeanshift_MPI, exp 4) has shown ≈8x speedup over the CPU meanshift (i.e. cpuMeanshift, exp 1), it is still 10 times slower than the sequential GPU version (i.e. gpuMeanshift, exp 2, 3). So, the sequential GPU version i.e. gpuMeanshift (exp 2, 3) has shown ≈10x speedup over CPU MPI version i.e. cpuMeanshift_MPI (exp. 8). This clearly shows the

Table 1. Performance of different HPC scenarios

Exp no	Experiment (Number of image-tiles used = 1665 each of size 256 × 256)	Execution time (minutes ≈ hh mm ss)
1	Segmentation of tiles one after the other in sequence using 'cpuMeanshift'	181.60004 m ≈ 3h 1m 36s
2	Segmentation of tiles one after the other in sequence using 'gpuMeanshift' on device GeForce GTS 450	2.25080 m ≈ 2m 15s
3	Segmentation of tiles one after the other in sequence using 'gpuMeanshift' on device Tesla C2075	2.14420 m ≈ 2m 9s
4	Segmentation of tiles in parallel using 'cpuMeanshift_MPI'	22.94183 m ≈ 22m 57s
5	Segmentation of tiles in parallel using 'gpuMeanshift_MPI' using both devices GeForce GTS 450 and Tesla C2075	1.13213 m ≈ 1m 8s
6	Segmentation of tiles in parallel using 'gpuMeanshift_MPI' on a GeoHPC cluster having 6 Tesla K40 devices	≈ 24.7703s

computational power of GPU devices over CPU. Now, to take the advantage of computational power of both the available GPU devices, the hybrid MPI+GPU approach for segmenting image tiles is used (i.e. gpuMeanshift_MPI, exp 5), which distribute the segmentation task among both GPU devices in a single machine (as described in section 3). This achieves ≈2x speedup over the GPU meanshift (i.e. gpuMeanshift, exp 2, 3), which clearly shows the applicability of the MPI+GPU hybrid approach presented in this paper. Finally, this presented approach for parallel segmentation is tested on a GeoHPC cluster (6 Tesla K40 devices) (exp 6) to observe the performance improvement in multi-GPU environment. It took only 25 seconds (approximately) to do the parallel segmentation of the image tiles and showed a ≈3x speedup over the segmentation using 2 GPU devices (exp. 5). This shows how the presented hybrid (MPI+GPU) approach scales with the increased number of GPU devices and achieves speedup over other approaches.

5. CONCLUSION

A high-performance analytics (hybrid MPI+GPU) approach for accelerating big data analytics chain in SIIM is presented. In this paper, the hybrid implementation of the segmentation module is developed. The performance improvement achieved by this approach shows its applicability towards rapid processing of the RS archives in disaster situation. Similarly, the other modules in the offline process chain can also take advantage of such hybrid approach to accelerate the RS data processing. Moreover, the offline modules in SIIM can be pipelined to achieve a task level parallelism. Further, the CUDA aware MPI technique [16] will be explored for fast communication among GPU devices in the offline module.

6. REFERENCES

[1] M. Datcu *et al.*, "Information mining in remote sensing image archives: System concepts," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 41, no. 12, p. 2923–2936, 2003.

[2] S. S. Durbha and R. L. King, "Semantics-Enabled Framework l for Knowledge Discovery from Earth Observation Data Archives," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 43, no. 11, pp. 2563-2572, 2005.

[3] C. R. Shyu *et at.*, "GeoIRIS: Geospatial Information Retrieval and Indexing System-Content Mining, Semantics Modeling, and Complex Queries," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 45, no. 4, pp. 839-852, Apr. 2007.

[4] M. Quartulli and I. G. Olaizola, "A review of EO image information mining," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 75, pp. 11-28, Jan. 2013.

[5] K. R. Kurte, S. S. Durbha, R. L. King, N. H. Younan and R. Vatsavai, "Semantics-Enabled Framework for Spatial Image Information Mining of Linked Earth Observation Data", *IEEE Selected Topics in Applied Earth Observation and Remote Sensing*, doi: 10.1109/JSTARS.2016.2547992. (***In Press***)

[6] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Trans. Pattern Anal. Mach. Intell*., vol. 24, no. 5, pp. 603-619, 2002.

[7] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Trans. Pattern Anal. Mach. Intell.,* vol. 17, no. 8, pp. 790-799, 1995.

[8] K.-L. Wu and M.-S. Yang, "Mean shift-based clustering," *Pattern Recognit.,* vol. 40, no. 11, pp. 3035-3052, 2007.

[9] F. Wan and F. Deng, "Remote Sensing Image Segmentation Using Mean Shift Method," in *Advanced Research on Computer Education, Simulation and Modeling*, Berlin Heidelberg, Germany: Springer, 2011, pp. 86-90.

[10] M. Huang, L. Men and C. Lai, "Accelerating Mean Shift Segmentation Algorithm on Hybrid CPU/GPU Platforms," in *Modern Accelerator Technologies for Geographic Information Science*, US: Springer, 2013, pp. 157-166.

[11] J. Sirotkovic, H. Dujmic and V. Papic, "Accelerating mean shift image segmentation with IFGT on massively parallel GPU," in *Proceedings of 36th inter. Convention on Information Communication Technology Electronics Microelectronics*, May 2013, pp. 279-285.

[12] A. Plaza, Q. Du, Y.-L. Chang and R. L. King, "High performance computing for hyperspetral remote sensing," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 4, no. 3, pp. 528-544, 2011.

[13] G. Cavallaro *et al*., "Smart data analytics methods for remote sensing applications," *in IEEE Int. Conf. Geoscience and Remote Sensing Sym*., 2014, pp. 1405–1408.

[14] Blepo, "Blepo computer vision library." [online]. Available: http://www.ces.clemson.edu/~stb/blepo/.[Accessed: 04-Dec-2015].

[15] OpenCV, "Open Source Computer Vision Library." [online]. Available: http://opencv.org/. [Accessed: 06-Aug-2015].

[16] J. Kraus. (2013, Mar. 13). "An introduction to CUDA aware MPI" [Weblog entry]. *PARALLEL FORALL*. Available: http://devblogs.nvidia.com/parallelforall/introduction-cuda-aware-mpi/. [Accessed: 14-Mar-2014].