Multi-GPU based Cluster System for CT Iterative Reconstruction Algorithm

Wan-li Lu[1, a *], Bin Yan[2,b], Jian-lin Chen[3,c], Ai-long Cai, Lei Li

[1] National Digital Switching System Engineering & Technology R & D Center

Zhengzhou, China

[a]15201469521@163.com

Keywords: CT; reconstruction algorithm; IR; multi-GPU

Abstract. Iterative reconstruction (IR) techniques are able to provide better reconstruction results compared with analytic reconstruction methods in cone-beam CT (CBCT) with incomplete and noisy projection data. However, image reconstruction algorithm remains to be significantly time-consuming especially faced with the large data set which decreases the efficiency of imaging processing. Although graphic processing unit (GPU) has offered an attractive alternative platform to improve the computation efficiency, but the single graphic card is still hard to meet the practical requirement because of the limitation on video memory. This paper presents a distributed parallel method for iterative algorithm using multi-GPU based cluster system, which is designed by algorithm characteristics and hardware structures. The experiments show that the multi-GPU based cluster system is able to achieve the same precision with single node, meanwhile it will gain higher speedup with the increasing number of nodes or GPUs. The computing time of whole reconstruction reduces to almost half of the origin by doubling the computing devices.

Introduction

Computed tomography (CT) reconstructions of three-dimensional (3D) objects reconstructed from a set of integrals from projection data are becoming increasingly popular. CT has the advantages of high efficiency, fast scanning speed, flexible scanning scope, and high image resolution. Iterative reconstruction techniques are able to provide better reconstruction results compared with analytic reconstruction methods such as FDK in CBCT with incomplete and noisy projection data [1-3]. However, these methods demand huge computational cost because of requiring a number of iterations to get an acceptable image, so it may hard be implemented in the practical applications. Recently, graphics processing units (GPUs) [4-7] have been employed to accelerate the iterative reconstruction. Nonetheless, the lack of research in the clinical application is the field requiring in-depth study.

This paper reports our recent progress toward solving the aforementioned problems. A multi-GPU based cluster system for CT iterative reconstruction was developed. While using multiple GPUs is a straightforward idea, inter-GPU parallelization is not a trivial problem. Specifically, since different GPUs only hold their own memory, communication among GPUs should be handled with care to achieve satisfactory efficiency. From a parallel computing point of view, conventional memory organization in a parallel processing task is either shared memory, where all processing units share a common memory space (e.g., a GPU), or distributed memory, where each unit holds its own memory space for conducting inter unit data communication (e.g., a CPU cluster). CBCT iterative reconstruction on a multi-GPU platform, however, attains a hybrid structure of shared and distributed memories. Careful design of the data allocation and communication among the GPUs is necessary to maximally exploit the potential of all the GPUs, as will be shown in this paper.

Background

Imaging Model and Iterative Algorithm

CT scanners could be modeled using several imaging systems by the following linear equation:

$$\boldsymbol{W} f = \boldsymbol{P}. \tag{1}$$

Where $\boldsymbol{p} \in \boldsymbol{P}$ represents projection data measured at certain angles, $f \in F$ represents an image function to be reconstructed, and the system matrix $\boldsymbol{W} : f \to F$ is a projection operator corresponding to those angles. In practice, the discrete–discrete model is assumed. For a typical iterative algorithm, the linear equation is usually solved by an iterative algorithm. The typical iterative algorithm conducts two steps: forward projection of the current reconstructed image f and back-projection of the projections acquired by the scanner.

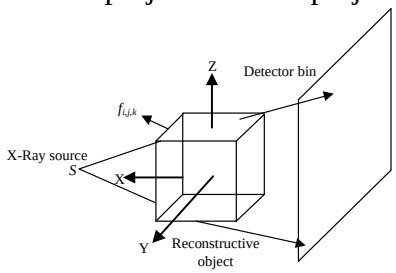

Fig.1. Diagram of cone-beam computer tomography system.

Theory of SART

As a major refinement of the algebraic reconstruction technique (ART), simultaneous algebraic reconstruction technique (SART) [8] was proposed in 1984, which has become one of the most popular iterative algorithm now. The iterative expression is (2).

$$v_j^{(k+1)} = v_j^{(k)} + \lambda \frac{1}{\sum_{i \in I_\varphi} w_{ij}} \sum_{i \in I_\varphi} w_{ij} \left(p_i - \sum_{n=1}^{N} w_{in} v_n^{(k)} \Big/ \sum_{n=1}^{N} w_{in} \right) \tag{2}$$

$v_j^{(k)}$ and $v_j^{(k+1)}$ represent the value of j th voxel in the process of k and $k+1$ iteration. p_i represents the value of projection from i th ray. λ represents the relaxation factor. I_φ represents the set of ray at the angle of φ.

Four steps can be divided from the process of iteration, which are forward projection, revise, back-projection and update. The first two steps can be shown as (3).

$$c_i = p_i - \sum_{n=1}^{N} w_{in} v_n^{(k)} \Big/ \sum_{n=1}^{N} w_{in} \tag{3}$$

And the expression of the last two steps is (4).

$$v_j^{(k+1)} = v_j^{(k)} + \lambda \sum_{i \in I_\varphi} c_i w_{ij} \Big/ \sum_{i \in I_\varphi} w_{ij} \tag{4}$$

So the acceleration of SART can be divided into the acceleration of forward projection and backward projection.

Implementation Details.

Our cluster system is comprised of several workstations connected by high-speed switch. One of them is assigned as the root node and the others are computing nodes. Each node has one or more graphic cards. Considering the limitation on power supply, the number of graphic cards in a node is less than that of nodes.

Thus the parallel strategy can be divided into three levels according to the structure of this cluster system as follow.

Parallelization Strategy

Generally speaking, CBCT reconstruction involves two datasets: one is the image dataset and the other is the projection dataset. When it comes to multi-GPU, at least one of the two, or both, needs to be partitioned with each GPU. Nevertheless, there are an infinite number of ways to divide the datasets in theory. Considering the reality, the one performing better than others can be found.

One possible partitioning method is to divide the CBCT volume into sub-volumes. Considering the conveniences of calculation and storage, planes parallel to the coordinate plane are used to divide the volume. Let the rotation axis be the z axis. One is making the division plane parallel to X or Y coordinate plane, leading to the storage of the whole range of projections in each node. To get the right projection, which is used for backward projection, each node will gather all the other projections. The other is making the division plane parallel to the Z coordinate plane. Corresponding to the sub-volume, a part range of projection is enough, which reduces the amount of calculation and storage.

Another possible way to partition is to divide the dataset of projection by angle. a subset of projections at certain angles is stored in each GPU, as well as the whole volume data. For the forward projection, each GPU can perform independently of each other. However, because of the requirement of whole projections, which are accumulated from all GPUs, a large amount of communication is required.

Taking into account the storage, a sub-volume and a part of projection is stored for the first method, but each GPU stores a subset of projections and the whole volume data using the second method. The size of volume is large bigger than the size of projection, so the first method is the best choice.

Forward projection and Back-projection

In this study, we consider a hybrid strategy for data division in the cluster system. The main idea of data division strategy is the first method aforementioned. Each node stores and calculates one sub-volume, as well as a part range of projections at whole angles. Fig.2 shows diagram of data division. A sub-volume maps a range of projection. For the i th sub-volume named v_i, its range of projection is p_i. The projection of v_i is located in p_i, and the projection in p_i is enough to reconstruct v_i.

There are overlapped rows between the adjacent projection, and the right value in overlapped rows is the summation of projection over and under it. Make sure the right value in overlapped rows, the sub-volume can be reconstructed accurately. So the communication between adjacent projections is necessary after forward projection. The whole process is described in Fig.3.

Not all the pixels in a projection are required to transmit to the current computing GPU for the particular sub-volume inbackprojection operation. The range of needed pixels in the detector can be

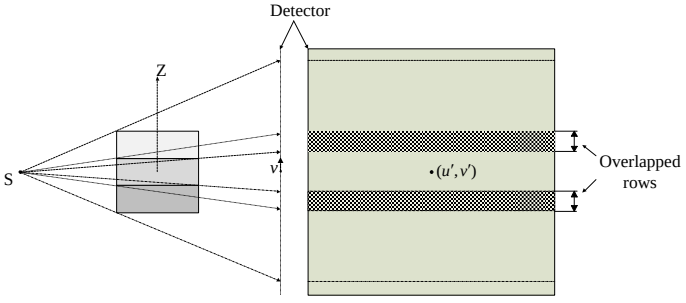

Fig.2. Diagram of data division.

obtain from geometric relationship of CT system. For the i th sub-volume, whose minimum and maximum coordinates in Z direction are z_i and z_{i+1}, then the corresponding range of pixels in direction v is $[v_i^L, v_i^R]$, where

$$\begin{cases} v_i^L = \max\left[\dfrac{D}{\varepsilon}\cdot\dfrac{z_i}{d-L/2}+v', \dfrac{D}{\varepsilon}\cdot\dfrac{z_i}{d+L/2}+v'\right] \\ v_i^R = \min\left[\dfrac{D}{\varepsilon}\cdot\dfrac{z_{i+1}}{d-L/2}+v', \dfrac{D}{\varepsilon}\cdot\dfrac{z_{i+1}}{d+L/2}+v'\right] \end{cases} \tag{5}$$

and D represents the distance from source to detector, ε represents the size of pixel, d represents the distance from source to the center of volume, L represents the max distance along the x axis in the volume at all angles, v' represents the central coordinate of the detector.

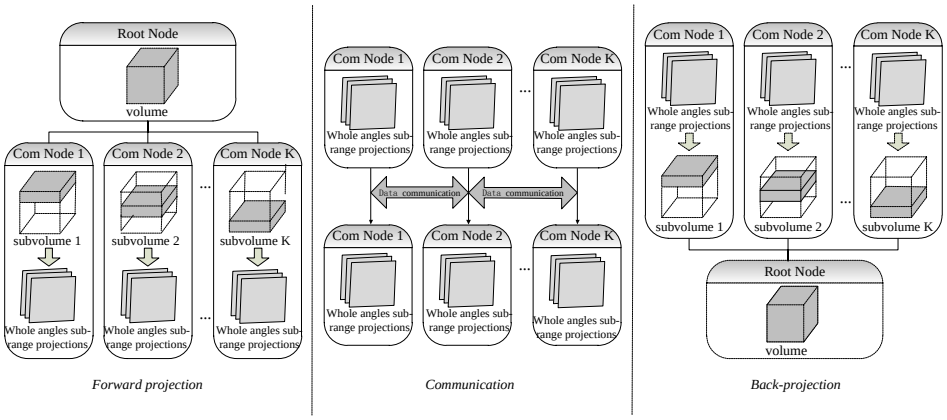

Fig.3. Algorithm framework.

OpenMP and GPU

For each node, OpenMP programming model is used to control the GPU. The parallelism between GPUs can be described as following: the projections or reconstructed volume are sent to video memory after received from root node. Then forward projection and backprojection are done in the GPUs. After the processing is finished, the reconstruction volume or projections will be transmitted back to the host memory and then combined into a whole. Suppose the number of GPU is N, the size of sub-volume is M_1 and the size of projections is M_2 in one node. For forward projection, each GPU in a node will have M_2 / K projections to operate and each GPU will have the whole sub-volume to be projected. For backprojection, the sub-volume is divided into N parts in a node and the projection is invariant. This strategy can avoid the communication among GPUs but it will need the accumulated time from all GPUs, leading to extra cost of time.

For each GPU, the computation is achieved by employing the forward projection and backprojection formula derived in Gao's paper [9] for high efficiency. GPU is perfectly expanded and adapted to general computation because of its many-core architectures. CUDA programming model is used for thread level parallelism. For different GPUs, appropriate dimension of blocks must be designed first; subsequently the reconstruction volume is assigned to each thread for parallel execution.

Besides, some optimized methods can also significantly improve the efficiency of the algorithm. The wisdom can be summarized following: 1) Make use of constant memory. Some repeated constants calculation in the kernel waste the computing resources of GPU. These constants can be calculated in CPU first for once and then transmitted to GPU. Constant memory is the best choice to save them for its low access latency. 2) Reduce the amount of global memory access. To reduce the amount of global memory access, more than one projection are back-projected in the kernel. A variable of register is applied for saving the intermediate back- projection result and then copied to global memory.

Experimental Results

To improve the computational efficiency, we have developed a multi-GPU based cluster system. The system contain four NVIDIA Tesla K20c GPU cards. These GPUs are labeled as GPU 1 through 4 in the rest of this paper. For each GPU, there are 2496 CUDA Cores, each of which attains a clock speed of 0.71 GHz. All processors on a GPU share 4800 MB global memory. For the host, each node has an Intel Xeon 2.60GHz CPU, 32GB RAM and 7200 RPM hard disk. The operating system is Windows 7 64bit with the compiler of Visual C++ 2010 and MPICH version 1.41, OpenMP version 2.0 and CUDA version 5.0.

The experiment is carried out on three different type of environments: Single PC with one GPU; Cluster No.1: 2 nodes with 2 GPUs per node; Cluster No.2: 4 nodes with a single GPU per node. To validate our method for improving the computational efficiency, we perform simulation data experiment. Volume size of 256^3 and 512^3 with the corresponding projections 360×256^2 and 360×512^2 are included, which are typing of 32-bit floating point. Both time cost and reconstruction quality are concerned in the experiment.

We choose the middle slice of the reconstruction volume to estimate the image quality, shown in Fig.4 and Fig.5. Meanwhile, the root mean square error (RMSE) between the phantom and four different reconstruction volumes are calculated (Table 1). From the Table 1, we can find that the RMSE of the four results are all the same both the volume size of 256^3 and 512^3, which means that the communication between the nodes or GPUs doesn't bring any error.

Table 2 shows the time cost of the reconstruction by three types. While obtaining the same reconstruction results by the three types, the time with GPUs in cluster can is much less than the single one. The whole process is segmented into four procedures listed in the left column. The results show that the ratio of time cost in forward projection and backprojection is larger than others. With the increasing of nodes, the time of them can fall considerably. When the number of nodes is two, the overlapped rows of projection are none, so the cost of communication can be ignored. While the ratio of communication increase obviously on account of overlapping data with four nodes. Even so, the performance of cluster is much better than single one.

RMSE	Single workstation		Cluster system 1		Cluster system 2	
	256^3	512^3	256^3	512^3	256^3	512^3
Iteration 1	0.052936	0.053034	0.052936	0.053034	0.052936	0.053034
Iteration 10	0.003747	0.004691	0.003747	0.004691	0.003747	0.004691
Iteration 30	0.003234	0.004034	0.003234	0.004034	0.003234	0.004034
Iteration 50	0.003052	0.003518	0.003052	0.003518	0.003052	0.003518

Table 1. RMSE in iteration 1, 10, 30, 50.

Time	Single workstation		Cluster system 1		Cluster system 2	
	256^3	512^3	256^3	512^3	256^3	512^3
Forward projection[seconds]	172.09	1796.22	52.35	483.57	55.62	490.91
Backprojection[seconds]	593.27	6045.49	178.96	1591.21	180.38	1605.18
communication[seconds]	—	—	4.52	10.78	59.73	576.32
other[seconds]	218.12	2333.63	58.62	602.37	50.41	534.89
Total[seconds]	983.48	10175.34	293.93	2675.15	346.14	3207.30
Speedup	—	—	3.35	3.78	2.84	3.17

Table 2. Time cost in the reconstruction process by iteration 50.

(a) Single workstation (b) Cluster system 1 (c) Cluster system 2

Fig.4. Middle slice of reconstruction volume: 256*256*256

(a) Single workstation (b) Cluster system 1 (c) Cluster system 2

Fig.5. Middle slice of reconstruction volume: 512*512*512

References

[1] Bian J, Wang J, Han X, et al, Optimization-based image reconstruction from sparse-view data in offset-detector CBCT[J], Physics in Medicine and Biology. 58 (2013) 205–230.

[2] Sidky E Y, Jørgensen J H, Pan X, Convex optimization problem prototyping for image reconstruction in computed tomography with the Chambolle-Pock algorithm[J], Physics in Medicine and Biology. 57 (2012) 3065-3091.

[3] Wang L, Cai A, Zhang H, et al, Distributed CT image reconstruction algorithm based on the alternating direction method[J], Journal of X-Ray Science and Technology. 23 (2015) 83–99.

[4] C. X. Ma, H. Jiang, B. Yan and L. Li, "Design and realization of cone beam CT 3D image reconstruction based on FPGA," International Conference on Electric Information and Control Engineering. China, 2011, pp.3139-3142.

[5] Yan H, Wang X, Shi F, et al, et al, Towards the clinical implementation of iterative low-dose cone-beam CT reconstruction in image-guided radiation therapy: Cone/ring artifact correction and multiple GPU implementation[J], Medical Physics. 2014, 41: 111912.

[6] Zhang H, Yan B, Lu L, et al, High Performance Parallel Backprojection on Multi-GPU[A]. Fuzzy Systems and Knowledge Discovery (FSKD), 2012 9th International Conference on[C], 2012: 2693-2696.

[7] Y. N. Zhu, Y. S. Zhao and X. Zhao, "A multi-thread scheduling method for 3D CT image reconstruction using multi-GPU," J. X-ray Sci. Technology. 20 (2012) 187-197.

[8] Andersen A H, Kak A C. simultaneous algebraic reconstruction technique (SART): asuperior implementation of the ART algorithm [J]. Ultrasonic Imaging. 6 (1984) 81-94.

[9] Gao H. Fast parallel algorithms for the X-ray transform and its adjoint [J]. Medical Physics, 39 (2012) 7110-7120.