31st Annual International Conference of the IEEE EMBS
Minneapolis, Minnesota, USA, September 2-6, 2009

Compressed Sensing MRI with Multi-Channel Data Using Multi-Core Processors

Ching-Hua Chang and Jim Ji

Department of Electrical and Computer Engineering, Texas A&M University

***Abstract*—Compressed sensing (CS) has emerged as a promising method in the field of magnetic resonance imaging. Taking advantage of the signal sparsity in certain domain via L_1 minimization, CS requires only reduced k-space data to reconstruct an image. Since most clinical MRI scanners are equipped with multi-channel receiver systems, integrating CS with multi-channel systems may not only shorten the scan time but provide a better image quality. However, significant computation time is required to perform CS reconstruction. Furthermore, this burden will be scaled by the number of channels. In this paper, we proposed a reconstruction procedure, which uses multi-core processors to accelerate CS reconstruction from multiple channel data. The performance was tested in terms of comparing to different image sizes and using different number cores of CPU. Experimentally, it shows that the maximum efficiency benefits from parallelizing the CS reconstructions, pipelining multi-channel data on multi-core processors and choosing the numbers of channels as multiple numbers of cores.**

***Keywords*—Multi-channel Phased Array, Compressed Sensing, Multi-core Processors, Image Reconstruction**

I. Introduction

MAGNETIC resonance imaging (MRI) using conventional Fourier imaging is relatively slow as compared with other imaging modalities because the frequency domain data are sequentially sampled. A breakthrough of the sampling theory, i.e., Compressed Sensing (CS) [1,2,3], allows a sparse signal being reconstructed from a set of highly incomplete samples. CS theory lays a foundation for reducing MRI samples and increasing imaging speed. In [4], Lustig et al have made a success of applying CS to single imaging MRI to reduce the number of samples and have reported impressive results.

Moreover, since the first implement of phased array system in early 1990s [5], significant efforts have gone into the development and applications [6]. Since the array coils receive signals from multi-channel simultaneously, they offer parallelism. This important property can be exploited when reconstructing images by using parallel computing.

Because of the limited growth in processing clock and the problem of power consumptions, it caused a trend towards multi-core designed. The multi-core processor offers a platform for implementing parallel algorithms. In [7], a conjugate gradient CG solver was implemented on NVIDIA's G80 Graphics Processing Unit (GPU). It showed that the multi-core processor can accelerate the processing speed. Also, the implementation of CS reconstruction algorithm on a multi-core processor is proposed by Borghi et. al [8], who used the multi-core platform to solve the CS reconstruction problem, which involves L_1 minimization. The work on implementing MRI reconstruction algorithms with multi-core processors is proposed in [9,10]. They accelerated the image reconstruction speed with NVIDIA's GPU Quadro FX 5600.

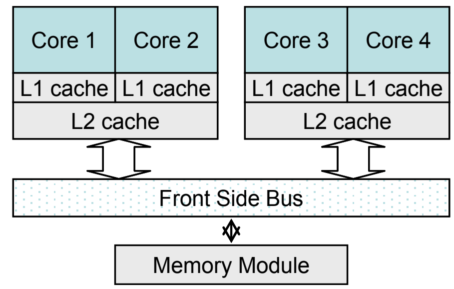

Fig. 1. The architecture for Intel core 2 quad CPU.

In this paper, we proposed a reconstruction procedure for combining CS with the phased-array receiver system, which can easily and efficiently use multi-core processors to accelerate the time for image reconstruction. We implement CS reconstruction algorithms with Intel's Core 2 Quad Q8200 2.33 GHz CPU. As shown in our results, the implementation of the reconstruction algorithm can be benefited from the parallel computing and multi-core architecture.

This paper is organized as follows. Section II introduces the methodology including the review of CS, phased Array multi-core processors, and the proposed method. Section III demonstrates the results, and finally the conclusions are drawn in Section IV.

II. Methodology

A. *Compressed Sensing MRI*

Based on the CS theory, Compressed Sensing MRI (CS-MRI) reconstructs an image x from a reduced set of incomplete k-space data y, where $y = \Phi x$ with Φ being the randomly-sampled Fourier transform operator implemented by the phase-encoding and frequency-encoding gradients. Generally, for better image reconstructions, a sparsifying transform Ψ is needed. Typically, Ψ is a wavelet or gradient transform (total variation). Then, the under-sampled data can be recovered by the following constrained optimization problem.

$$\min\|\Psi x\|_1 \quad \text{Subject to} \quad \|y-\Phi x\| < \varepsilon \qquad (1)$$

Note that ε is a parameter corresponding to the data fidelity. Nevertheless, solving the optimization problem is computational and memory intensive when the equations are in a large scale.

B. The Phased Array MR Receiver System

The phased array MR receiver system associates with a set of decoupled receiver coils and separate receiver channels. In principle, with phased array technology, an increase in imaging speed equal to the number of parallel coils can be achieved. Besides, since coil sensitivities are typically unknown, optimal and artifact-free image reconstruction is a challenge. The most commonly used method for image reconstruction is so-called "sum-of squares" (SOS), which effectively computes the root mean square average of images associating with different coils.

C. Multi-Core Processors

Because of the limited growth in processing clock and the problem of power consumptions, increasing in the performance of single-core processors slowed down. It caused a trend towards multi-core designed, so that Intel places multiple processors on a single chip. A multi-core CPU is a standard superscalar and each core is composed of vector computing units. As depicted in Figure 1, cores are clustered by 2 and they share a common memory of level 2 cache. Besides, a front side bus interconnecting 2 clusters is also used to access the memory module (random access memory). As a result, this architecture can reduces memory access latency. However, in most cases of running programs, the usage of CPU is not always fully utilized. Thus, how to use processors more efficiently on combining the parallel imaging and compressed sensing will be addressed on next section.

D. The Proposed Method

In this paper, we proposed a reconstruction procedure by using multi-core processors to accelerate the combination of multi-channel phased array and the compressed sensing MRI. The flow control is illustrated as Fig. 2. First, if the data received from different channels are properly sampled, by applying the CS theory, the reconstruction is obtained by solving eq.(1) individually. Moreover, the eq.(1) can be recast as minimizing

$$J(\hat{x}_k) = \|\Phi\hat{x}_k - y_k\|_2^2 + \lambda\|\Psi\hat{x}_k\|_1 \qquad (2)$$

Note that Ψ represents the total variation and the regularization parameter λ associated with data fidelity is experimentally set to 0.001. Since there are many existing solvers for convex optimization, here, we use software

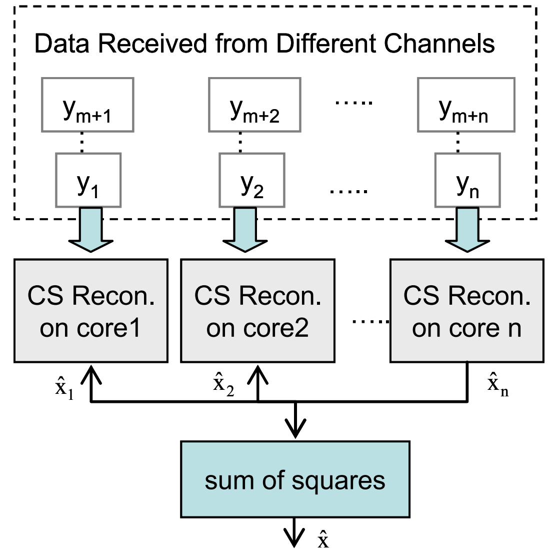

Fig. 2. The reconstruction procedure for combining CS with the phased array MR receiver system.

package of sparseMRI [11], whose algorithm are based on the non-linear conjugate gradient method. Second, the data and their corresponding CS reconstructions can be parallelized into multi-cores CPU because they are data-independent. It can also be implemented in Matlab (Mathworks, Natick, MA) and the implementation takes advantage of the limitation that each Matlab process can only run in a single core. Thus, the CS reconstructions in Fig. 2 can be parallelized into different Matlab processes. We directly use the existing package of parallel processing on multiple cores, provided by M. Buehren [12]. Finally, all the reconstruction images are combined by the SOS addressed as the following equation.

$$\hat{x}(i,j) = \sqrt{\sum_{k=1}^{mn} |\hat{x}_k(i,j)|^2} \qquad (3)$$

Note that there are some overhead on saving and loading temporary files and communicating between processes. However, comparing to the reconstruction algorithms of compressed sensing, the overhead become little part of total computations. The proposed method is tested in computer simulations. Without scarifying the image quality, the performance was evaluated by the execution time of reconstructions.

III. Results

In order to prove the effectiveness of our approach and compare the benefits and advantages of parallelizing the reconstruction using multi-core CPU, we work on an experimental platform that consists of an standard Intel Core 2 Quad Q8200 2.33 GHz with 4 MB L2 cache and 4GB

DDR2. Preliminarily, we simulate the input data with the phantom of 'Shepp-Logan' and it is sub-sampled in frequency domain with the pattern of radial sampling. The sub-sample factor in each channel is about 40% and we limit the receivers to 4-channel data input (m*n = 2 in Fig.2). Besides, the operations on each core in Fig. 2 contain a full solver to find an optimum solution of CS reconstruction, which basically associates with minimizing eq. (2). Comparing the package of L_1-magic, it has less sparse-matrix vector multiplications, which may easily cause CPU stall due to the limited memory bandwidth especially for large image sizes. The performance was tested in terms of execution time, image sizes, and the number usage of cores.

Table 1 lists the execution time, which mainly counts the optimization function of CS reconstructions in Fig.2. In our computer simulation, we exclude the time of preparing multi-channel input data. The only overheads of the processor operations are listed in Table. 2, which include the runtime in saving and loading temporary files and Table 3 contains the stalled time that processors may wait for each others before continuing on the next operation (sum of square in Fig.2). Note that if multiple cores are working simultaneously, we only record the maximum stalled time and the maximum wasted time on reading/writing files among processors. First, the time spent on saving and load temporary files is roughly

size \ cores	1	2	3	4
16*16	12.31	7.7	7.44	6.24
32*32	17.78	13.24	11.3	10.05
64*64	40.31	26.76	26.28	22.45
128*128	125	76.76	81.22	59.3
256*256	566.98	325.61	334.63	248.9

Table. 1. The total execution time counted in CS reconstruction modules in Fig.2

size \ cores	1	2	3	4
16*16	1.555	1.73	1.17	1.26
32*32	1.176	1.47	1.28	1.4
64*64	1.123	1.15	1.56	1.7
128*128	1.41	1.11	1.42	1.9
256*256	1.496	1.39	1.88	1.34

Table. 2. The overhead on saving and loading temporary files.

size \ cores	1	2	3	4
16*16	0	1.37	1.17	0.96
32*32	0	1.404	1.28	0.934
64*64	0	1.044	6.91	1.293
128*128	0	0.867	15.36	7.8
256*256	0	0.742	27.43	7.73

Table. 3. The stalled time of processors waiting for each other.

1~2 seconds shown in Table 2, so that it is not very important

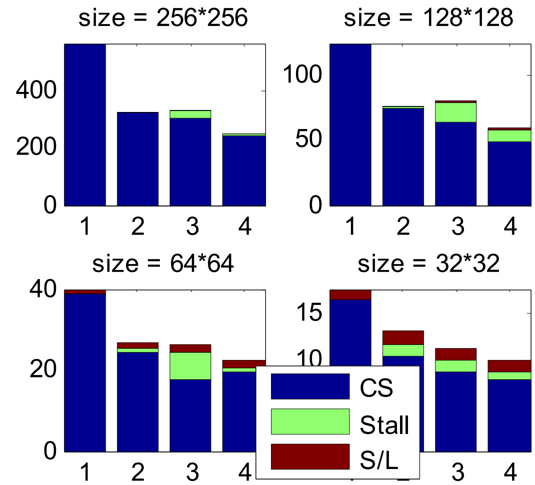

Fig. 3. The proportion CS, the stalled time, and overhead in the total execution time. Horizontal axis represents the numbers of core in use, and vertical axis represents the computation time of each portion.

when the image size becomes larger. In addition, Table 3 shows that there is no processor stalling since only one core is used. Moreover, using 2 cores also gives less stalled time in processors (about 1~2 sec) because 2 cores can almost start running at same time and provide converged results if signals is received under same channel conditions. It is worth to mention that when we increase the number usage of CPU cores to 3, there is no benefit from 3 cores comparing to the usage of 2 cores. Because, in our experiments, we test the procedure of reconstruction with 4-channel data sequentially waiting for reconstruction, there are 2 processors stalling when 3 channel data is reconstructed and 1 channel data is left. Besides, as the image size gets larger, the stalled time increase significantly. We can observe that using 4 cores also has long stalled time for image sizes of 128 and 256. Intuitively, the main reason for contributing the stalled time comes from processors waiting for others to finish jobs since 4-channel data pipeline into 4 cores and they may not start or finish the job simultaneously.

We subtract all the overhead and stalled time from Table 1, and plot them as a 'CS'(blue) bar in Fig. 3. Comparing sub-plots for large image sizes with that of image size 32, the overhead and stalled time eliminate the advantage of the parallelization and lead to a little increase on computation time. However, when images become larger, the percentage of overhead and stalled time is significantly reduced and the parallelization of CS reconstructions gains much benefits.

To further illustrate the efficiency and the performance, we plot the execution time of 'CS'(blue) bar in Fig. 3 in terms of cores vs. execution time and images sizes vs. execution time shown in Fig. 4 and Fig. 5. Note that in figure 4, 'p' represents the total length of image size and the execution time excludes the overhead and stalled time. Surprisingly, using 2 cores has significant reduction in execution time. The average reduction factor is constantly about 1.684 no matter what image size is. Fig. 4 and Fig. 5 obviously shows that using 3 cores have almost the same runtime as using 2 cores; we have explained previously that when we use 3 cores and the total channels are 4, there are only 3-channel data that can be pipelined and dealt by 3 processors. Therefore, the most efficient way to parallelize the reconstruction of CS running on different cores is to choose the numbers of channels as multiple numbers of cores. Moreover, using 4 cores give us the shortest reconstruction time. Comparing with using 1 core, the average reduction factor is constantly about 2.338; Comparing with using 2 cores, the average reduction factor is constantly about 1.321. The efficiency is not double as we double the number usage of cores since the memory bandwidth and the memory size are fixed and limited.

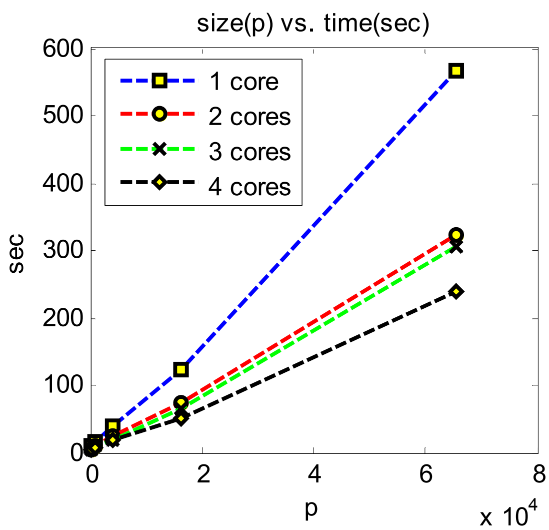

Fig. 4. Numerical results for comparing different numbers of cores.

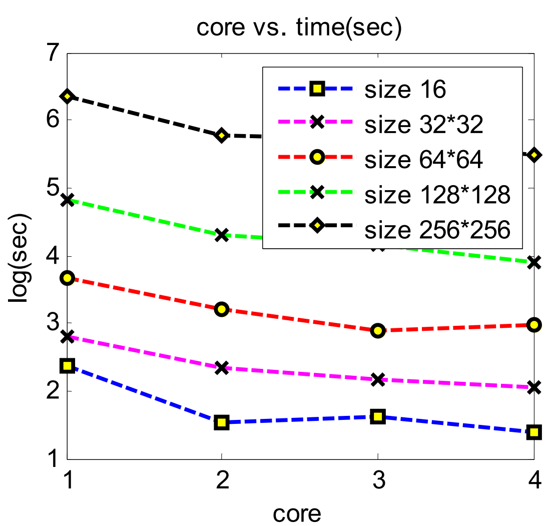

Fig. 5. Numerical results for comparing the computation of different image sizes. The time is in logarithmic scale.

IV. Conclusions

We proposed an efficient reconstruction procedure for compressive sensing MRI with phased array MR receiver system. By utilizing multi-core architecture of CPU, it significantly shortens the reconstruction time. In our experiments with 4-channel data, using 2 cores of CPU gives maximum reduction of time on reconstruction; while using 4 cores gives the fastest reconstruction over multi-channel CS MRI. Moreover, it appears that the maximum efficiency was gained by choosing the numbers of channels matching with the numbers of cores. Further research is required to maximize the efficiency of multi-core processors and other parallel processors such as Graphics Processing Units (GPU) for parallel MRI and CS imaging with channel numbers normally larger than 4.

Acknowledgement

This work was supported in part by the National Science Foundation under Award number 0748180. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect those of the National Science Foundation

References

[1] Emmanuel Candès, Justin Romberg, and Terence Tao, "Stable signal recovery from incomplete and inaccurate measurements," Communications on Pure and Applied Mathematics, 59(8), pp. 1207-1223, August 2006.

[2] E. J. Candès, Justin Romberg, and Terence Tao, "Robust uncertainty principles: Exact signal reconstruction from highly incomplete frequency information," IEEE Trans. on Information Theory, 52(2) pp. 489 - 509, February 2006.

[3] David. Donoho, "Compressed Sensing," IEEE Trans. on Information Theory, 52(4), pp. 1289 - 1306, April 2006.

[4] M. Lustig, D. Donoho and J.M. Paul, "Sparse MRI: the application of compressed sensing for rapid MR imaging," Magn. Reson. Med. Vol. 58, No. 6, pp.1182-1195, 2007.

[5] P.B Roemer, W.A.Edelstein, C.E. Hayes, S.P. Souza, O.M. Mueller, "The NMR Phased Array," Magnetic Resonance in Medicine 16, 192-225, 1990.

[6] S.M.Wright, L.L. Wald, "Theory and Application of Array Coils in MR Spectroscopy," NMR in Biomedicine 10, 394-410, 1997.

[7] W.A. Wiggers, V. Bakker, A.B.J. Kokkeler, and G.J.M. Smit, "Implementing the conjugate gradient algorithm on multi-core systems," Proceedings of the International Symposium on System-on-Chip, pp. 11-14.,19-21 Nov, 2007.

[8] Alexandre Borghi, Jerome Darbon, Sylvain Peyronnet, Tony F. Chan and Stanley Osher, "A Simple Compressive Sensing Algorithm for Parallel Many-Core Architectures," Unpublished article, September, 2008.

[9] Sam Stone, Justin Haldar, Stephanie C. Tsao, Wen-mei Hwu, Zhi-Pei Liang, and Bradley P. Sutton, "Accelerating Advanced MRI Reconstructions on GPUs," Proceedings of 5th International Conference on Computing Frontiers, ACM, May, 2008.

[10] Sam Stone, Haoran Yi, Wen-mei Hwu, Justin Haldar, Bradley Sutton, Zhi-Pei Liang. "How GPUs Can Improve the Quality of Magnetic Resonance Imaging". Unpublished article, October, 2007

[11] http://www.stanford.edu/~mlustig/SparseMRI.html

[12] http://www.mathworks.com/matlabcentral/fileexchange/13775)