Parallel Processing Imaging Algorithm for Synthetic Aperture Radar based on Pipeline

Hua Zhili
Shandong Academy of Sciences Institute of Oceanographic Instrumentation,
Shandong Provincial Key Laboratory of Ocean Environment Monitoring Technology,
Qingdao, China
hua_zhili@yahoo.cn

Hui Chao
Shandong Academy of Sciences Institute of Oceanographic Instrumentation,
Shandong Provincial Key Laboratory of Ocean Environment Monitoring Technology,
Qingdao, China
benhc@163.com

Li Hongping
College of Information Science and Engineering,
Ocean University of China,
Qingdao, China
lihp2005@yeah.net

Abstract—**Parallel processing is an effective way to process Synthetic Aperture Radar (SAR) data quickly and realize real-time imaging. Based on the traditional serial Range-Doppler algorithm, a parallel SAR imaging processing scheme is present by using of pipeline technology, and partition approach of computing nodes in both range and azimuth compression is discussed. Computing results show that, when the ratio of nodes is 2/4 or 2/5, the efficiency of parallel processing can be satisfied, in which 2/4 refers to the best efficiency and 2/5 to the best speedup. The ability of real-time SAR imaging is also tested, from which the ultimate computing time of a frame image is about 0.5 s, smaller than the forming time of 0.92 s.**

Keywords- Synthetic Aperture Radar; Imaging; Parallel; Pipeline

I. INTRODUCTION

Synthetic aperture radar (SAR) is a system which can produce images from radar signals obtained using a relatively short antenna that transmits long duration pulses. By time domain convolution in both range and azimuth direction, SAR can produce high resolution images using special signal processing implemented on wide purpose architectures[1]. One of the traditional problems with SAR as a remote sensing tool is the very large computation and data storage requirements to form an image from the raw data. Due to the high cost of the special purpose SAR processors, high performance computing platforms are becoming popular for SAR processing.

The SAR processing algorithm used in the analysis in this paper is the range-Doppler (RD) algorithm, which is one of the most widely used SAR algorithm first developed in 1979 for the processing of SEASAT data[2,3]. Some obvious partition characteristics of this algorithm, such as fairly simple data dependencies and synchronization requirements make it ran on multiple processors easier and more efficient[4]. Hence, several parallel techniques based on this algorithm have been used in the past[5,6].

The algorithm used in this paper is suitable for using pipeline mechanism which is an efficient architecture to obtain high throughput performance, and has been test on a high performance cluster system. By the computing results, the most appropriate partitioning approach is discussed, and real time imaging ability by using of this algorithm is proved to realize.

II. PARALLEL SAR IMAGING ALGORITHM

All this time the RD algorithm has been the basic of most precision SAR processors. The core steps included in the algorithm mainly are pulse compression in range direction, match filtering in azimuth direction and corner turn.

The basic SAR signal processing sequence is shown in Fig.1, in which P refers to computing nodes, n is the total number of computing nodes, R means compression in the range direction, and A means compression in the azimuth direction. Solid line in the figure describes the data flow of the processing, and dashed line means tasks assigned to each nodes. During the SAR imaging process, processors are partitioned into two groups, namely ($P_1 \ldots P_m$) with m processors performs the pulse compression in range direction, and ($P_{m+1} \ldots P_n$) with n-m processors performs the match filtering in azimuth direction. Since computational requirement for each group is different, the number of processors assigned to each group is different. The operations of the two groups are pipelined, the first group receives data and process it, and then sends the processed data to the second group. So in this algorithm, the number of communication steps is $m \times (n-m)$.

On the other hand, the original data source should also be partitioned during the data transmission. The SAR image data are first partitioned into m non-overlapped horizontal strips. Each data strip has $[M/m] \times N$ rows, where M is the number of samples in a column. Each data strip is assigned to one of m processors which perform pulse compression in row dimension. Then, each horizontal data strip is partitioned into (n-m) blocks. The size of each block is $M \times [N/(n-m)]$, where N is the number of samples in a row. Each block in a processor is sent to a different processor for subsequent processing.

A major problem in performing SAR signal processing on high performance computing platform is the cost of communication; partially processed data should be moved between processors for subsequent processing. To send data

Supported by the National Natural Science Foundation of China (Grant No. 41275013)

from a processor to another in a message passing environment, the two processors first need to set up a communication channel, and then send and receive data, namely per unit transmission time is the cost of transferring a message of unit length through the network. Therefore, how the input data is partitioned to minimize the unit transmission time is important for corner turn between the two groups' processors.

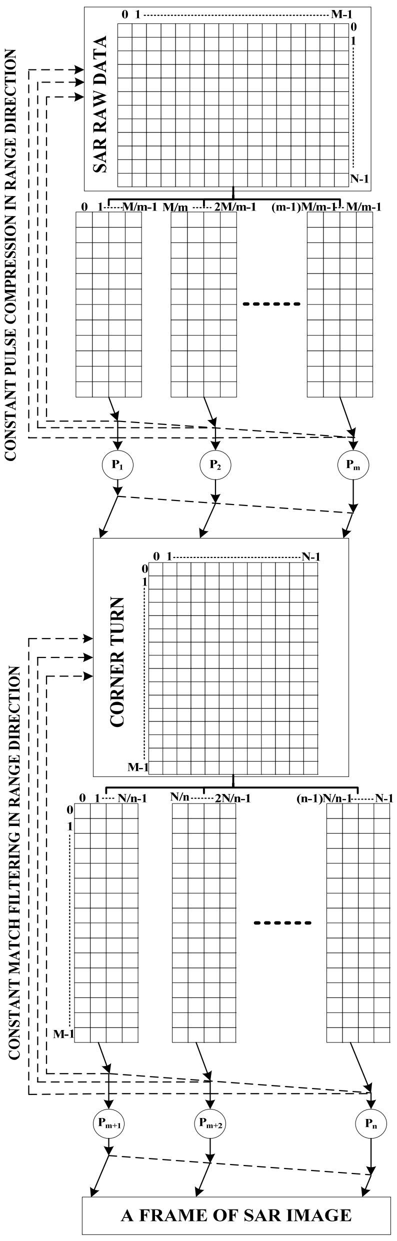
CONSTANT PULSE COMPRESSION IN RANGE DIRECTION
SAR RAW DATA
0 1 M-1
N-1
0 1 M/m-1 M/m 2M/m-1 (m-1)M/m-1 M/m-1
P1
P2
Pm
CORNER TURN
0 1 N-1
M-1
CONSTANT MATCH FILTERING IN RANGE DIRECTION
0 1 N/n-1 N/n 2N/n-1 (n-1)N/n-1 N-1
Pm+1
Pm+2
Pn
A FRAME OF SAR IMAGE

Figure 1. Schematic of Parallel SAR Image Algorithm

III. Experiment results and analysis

We use raw data to implement parallel SAR processing on IBM cluster system. The original data is 2048×512 pixels, parallel program is written on Message Passing Interface (MPI) software platform and we use mathematic library to do FFT.

TABLE I. SAR system parameters

Parameter	Value
Instrument Type	Airborne Ka-band Synthetic Aperture Radar
Center frequency	33.56 GHZ (Ka-Band)
Polarization	HH/HV/VV
Swath	375 m
Swath range	7.26 km
PRF	556 Hz
Velocity	127.1 m/s
Resolution	0.3 m
Range sampling	2048
Azimuth sampling	512
Grid number	2048*512*4 (consist of 4 frames)

To achieve the high throughput performance by using a software task pipeline with an efficient architecture, the total time includes both computing and communication time for range and azimuth compression should almost be the same in theory. Through test we have found the ratio of compression time is 1/2, that is the range and azimuth compression costs is 42.76 s and 90.54 s respectively. Then, we are going to analysis the partition approach of the computing nodes. First, the number of range compression nodes is fixed to 2, and different node arrangements of 2/n, such as 2/2, 2/3, 2/4, 2/5, 2/6 and 2/7 are performed to evaluate corresponding results including computing time, speedup and efficiency which are shown in Table 2.

TABLE II. Computing results for partition approach of 2/n

Parameter	Value					
Total No. of nodes	4	5	6	7	8	9
Range nodes	2	2	2	2	2	2
Azimuth nodes	2	3	4	5	6	7
Ratio value	1	0.67	0.5	0.4	0.33	0.29
Total computing time /s	46.10	30.85	22.95	**22.05**	22.10	22.08
Expectation /s	33.4	26.72	22.27	19.08	16.7	14.84
Efficiency /%	72.5	86.6	**97.0**	86.5	75.6	67.2
Speedup	2.9	4.33	5.82	6.06	6.06	6.06
Input throughout /Mbps	0.35	0.52	0.7	0.73	0.72	0.72
Output throughout /Mbps	0.69	1.04	1.4	1.45	1.45	1.45

Results in Table 2 show that as the number of the computing nodes increases, total computing time would reduce gradually until the ratio value arrives at 2/5, then the computing time drives to steady. Computing time trend compared with theory value is shown in Figure 2. Figure 3 shows the throughout variation of both input and output data, the trends correspond well with the computing time, which also appears as a increasing at the beginning and gradually stabilized.

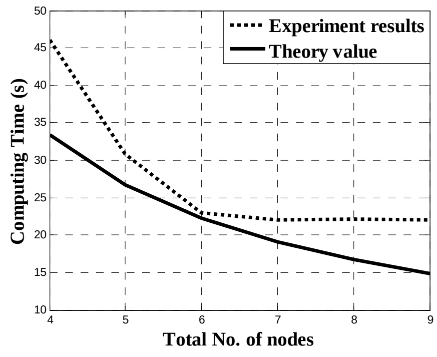

Figure 2. Computing time trend for partition approach of 2/n

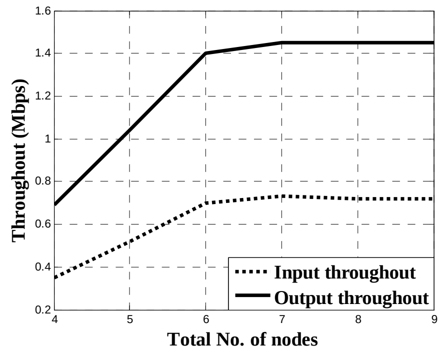

Figure 3. Throughout trend for partition approach of 2/n

Above results show that the best computing time comes from the ratio of 2/5, while the most efficient configuration is 2/4. To achieve a better speedup and efficiency, 2/4 is the choice in this paper. Computing test is also introduced to the case that the number of range compression nodes is 4, and ratio of nodes still remain the same. Tab.3 shows the computing time which is similar to that of 2/n. Trend of computing time, throughout of both input and output data are respectively shown in Figure 4 and 5.

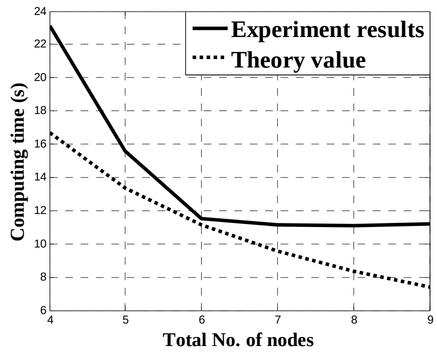

Figure 4. Computing time trend for partition approach of 4/n

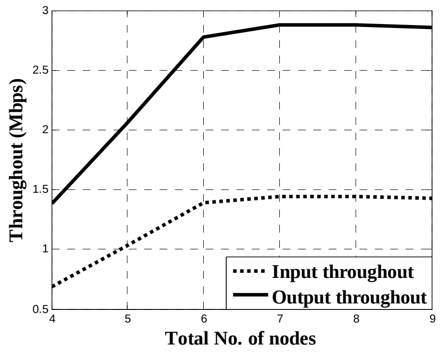

Figure 5. Throughout trend for partition approach of 4/n

TABLE III. COMPUTING RESULTS FOR PARTITION APPROACH OF 4/N

Parameter	Value					
Total No. of nodes	8	10	12	14	16	18
Range nodes	4	4	4	4	4	4
Azimuth nodes	4	6	8	10	12	14
Ratio value	1	0.67	0.5	0.4	0.33	0.29
Total computing time /s	23.13	15.56	11.52	11.13	11.10	11.2
Expectation value /s	16.7	13.36	11.13	9.54	8.35	7.42
Efficiency /%	72.2	85.9	96.6	85.7	75.2	66.3
Speedup	5.78	8.59	11.6	12.	12.04	11.93
Input throughout /Mbps	0.69	1.03	1.39	1.44	1.44	1.43
Output throughout /Mbps	1.38	2.06	2.78	2.88	2.88	2.86

Direct comparison of efficiency for the partition approach of 2/n and 4/n is shown in Figure 6, from which we can see efficiency overall decreases somewhat when total number of nodes doubling. That is because efficiency depends on the ratio of computing time and communication time that make up the total computing time. To a fixed ratio value, as the number of nodes increase, computing time would have a cutoff of 1/2, but corresponding communication time would reduce somewhat less owing to the bandwidth limitation and data latency. Then the proportion of computing time in the total time gradually decreases and finally cuts the efficiency off.

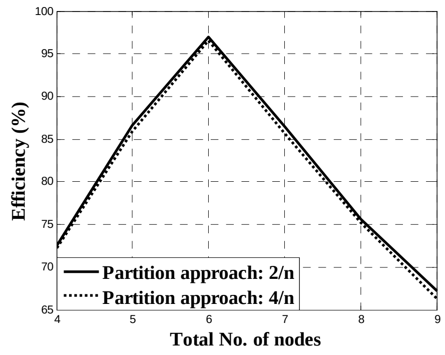

Figure 6. Comparison of Efficiency for the partition approach of 2/n and 4/n

Another important test items in this paper is the ability of SAR real-time imaging evaluated by ultimate processing power and corresponding cost. By the computing nodes partition approach of 2/4, parallel SAR imaging algorithm based on pipeline is run on a cluster platform with 112 computing nodes, input data source is original raw data. Tab.4 shows the ultimate computing time for a frame of image with 96 nodes is 0.5 s, smaller than the image forming time of 0.92 s which calculated from the SAR system parameters list in Table 1. That means SAR real time imaging is realized, and a frame picture is shown in Figure 7.

TABLE IV. ULTIMATE COMPUTING TIME TEST RESULTS

Parameter	Value		
Total No. of nodes	24	48	96
Range compression nodes	8	16	32
Azimuth compression nodes	16	32	64
Total computing time /s	5.98	3.25	2.0
Ultimate computing time of a frame image /s	1.5	0.82	**0.5**
Efficiency /%	92.9	85.5	70
Speedup	22.3	41	66.7

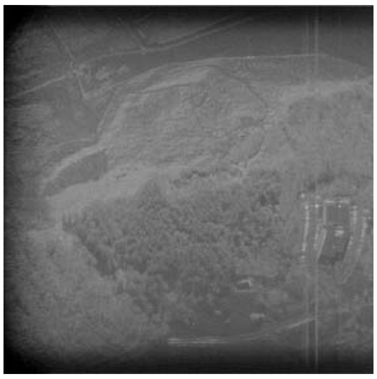
Figure 7. A frame of SAR Imaging Result

IV. CONCLUSION

Based on traditional serial RD imaging algorithm, a parallel processing scheme based on pipeline is tested on the high performance clustering system. With different ratio setting of range and azimuth compression nodes, the effect of this ratio change on final computing results is discussed. Computing results show that the best partition approach depends on the ratio of range and azimuth compression cost. The best efficiency is obtained with ratio of 2/4, and the best throughout and speedup with ratio of 2/5, which confirms the assumption that the respect time of range and azimuth compression should almost be the same. Then the ability of SAR real-time imaging is tested. A 0.5 s ultimate processing time, smaller than the forming time of a frame raw data is obtained, which realized the real time imaging ability.

REFERENCES

[1] J.C. Curlander and R.N. McDonough. Synthetic Aperture Radar Systems and Signal Processing. John Wiley & Sons, Inc, New York,1991

[2] I.G. Cumming and J.R. Bennett. Digital processing of SEASAT SAR data. IEEE International Conference on Acoustics, Speech and Signal Processing, 4(1979) 710-718

[3] C. Wu, K.Y. Liu and M. Jin. Modeling and a correlation algorithm for spaceborne SAR signals. IEEE Transaction on Aerospace and Electronic Systems, AES-18(1982) 563-575

[4] P.G. Meisl, M.R. Ito and I. G. Cumming. Parallel Synthetic Aperture Radar Processing on Workstation Networks. Parallel Processing Symposium, Proceedings of IPPS, 1996, 716-723

[5] M. Cafaro, I. Epicoco and S. Fiore et al. Near real-time parallel processing and advanced data management of SAR images in grid environments, Journal of Real-Time Image Processing, 4(2009), 219-227

[6] C.E. Nahum and H.Cantalloube. SAR image synthesis algorithm improvement on multi-processor/multi-core computers: Vectoring on massively parallel processors. Proc. IEEE Radar Symposium (IRS), 2011, 379-384.