Chapter 4

Optimization of Image processing application on VAX/VMS system

1. Introduction.................................................................................................................. 2
2. IRS Data-products........................................................................................................ 3
   2.1 Data Products System.............................................................................................. 3
   2.2 Data products generation scheme ........................................................................... 4
3. Software Optimization.................................................................................................. 4
   3.1 Optimization philosophy.......................................................................................... 5
   3.2 Case study and Results ............................................................................................. 8
4. Operating under scheduler environment.................................................................... 13
5. Conclusion..................................................................................................................... 14
1. Introduction

The massive task of developing data products software for Indian Remote sensing Satellites (IRS) was going on when the study began. It consists of various applications like radiometric correction, geometric correction, swath modeling, cloud cover estimation, data quality evaluation, etc. As a preliminary study, it was decided to study the virtual parallel processing capabilities of VAX/VMS and work out the guidelines for the software optimization. The guidelines may be established by optimizing more than one application. The guidelines should be generic in nature so that these may be applied to all the applications. The optimization task for an application is a difficult job. Moreover working out the generic guidelines that could be useful to many applications is a real challenging job.

To start with, the Radiometric correction software (RADCOR) was examined [6]. The VAX/VMS operating system, and its architecture was studied in detail [10,11]. As the software was developed in FORTRAN, the efficiency aspect of FORTRAN was studied in detail [12,13]. The radiometric correction software was optimized from 10 minutes to 2.75 minutes for an image [I].

The guidelines were used to optimize the geometric correction (GEOCOR) and photo processing (PCTGEN) applications and got a similar efficiency [I]. The optimized software has been used for IRS-IB and IRS-P2 successfully.

This chapter presents the detailed description of the work carried out.

Section-2 gives a brief overview of the IRS data-products. Section-3 explains the optimization philosophy (guidelines). The section-4 compares the scheduler efficiency with the optimization efforts. It gives a set of useful guidelines for IRS data-products generation software.
2. IRS Data-products

2.1 Data Products System

The VAX/VMS system configuration is used to support the IRS data products software starting from data reception, attitude determination, data product generation, Information Management System (IMS) and photo writing. The IRS data product generation software system configuration is centered around microVAX with five disk drives & four tape units under Hierarchical Storage Controller (HSC), two FPS-5000 array processors and Frame Sync Decom Unit (FSDU). The data-product generation software was developed using VAX/VMS Fortran, system services and FPS software support. The overview of data product generation system configuration is shown in figure-4.1.
2.2 Data products generation scheme

The raw data transmitted and recorded on High Density Digital Tape (HDDT), is prone to radiometric and geometric errors. It is being corrected at various stages of processing before producing final data products. The radiometric errors are mainly sensor non-uniformity, sensor response degradation over time, data transmission loss and so on. The geometric errors are mainly due to platform attitude variations, earth rotation, earth curvature, satellite velocity, and so on. The output data requires to be mapped to the standard maps for easy referencing.

The IRS data product generation is carried out using four basic software components. The DISKLOAD software downloads the data from HDDT using specially developed hardware (FSDU). The RADCOR software does the radiometric correction whereas GEOCOR software does geometric corrections. The PCTGEN software generates the User Computer Compatible Tapes (UCCT) which can be used by the application scientists. It also generates the Photo Compatible Tapes (PCT) for photo writing devices to generate photographic products. The RADCOR software is a mixture of I/O bound and compute bound processing, GEOCOR is highly computational whereas PCTGEN is only I/O processing. The RADCOR and PCTGEN use only VAX/VMS processing power and peripherals whereas GEOCOR additionally uses FPS-5000 for its high computation needs. Different flavors of above software are created by the image-processing scientists to support different types of IRS data-products. The scheduler system was developed for a smooth and efficient execution of the above processes [18].

3. Software Optimization

The IRS data-product generation software (DPGS) was developed by a group of scientists in FORTRAN. Only the device drivers & real time applications had been implemented in assembly language. The DPGS requires real time processing, different types of image processing algorithms, complex mathematics, satellites behavior modeling, sensor performance calibration, data transmission loss corrections, multiple types of I/O media handling, etc. See appendix-1 for more details. Since the processing power requirements were very low during early days, the software concentrated only on generating accurate data products. It did not bother about the efficient implementation.

Radiometric correction software was examined thoroughly [6]. The VAX/VMS operating system, its architecture was studied in detail [10,11]. The efficiency aspect of FORTRAN was studied in detail [12,13]. GEOCOR and PCTGEN s/w was overviewed at the same time for guideline generalization.
Optimization of image process application on VAX/VMS system

The following optimization guidelines (philosophy) have been worked out from the above study.

3.1 Optimization philosophy

3.1.1. Paralleling I/O and computation

The VAX/VMS had separate I/O controller, offloading CPU for processing [15]. The operating system provided the system services (API in today’s terminology) to use them in parallel [16,17]. Following are the two important system services for IO on VAX/VMS:

SYS$QIOW(event flag, io channel, function to perform, status, ...)

SYS$QIO(event flag, io channel, function to perform, status, ...)

The QIOW issues a read request to I/O controller and waits for request to satisfy. The QIO simply issues a read request to I/O controller along with an event flag. It does not wait for the request to be satisfied; instead it returns the control back to program to continue. The software needs to wait for event flag at synchronization point. It supports SYS$WAITFR(event flag) system services for synchronizing. This service waits for the event flag. (It will proceed further if the flag is already set). The I/O controller will set the event as soon as it completes the request. The program will continue further. Each request has status information too.

The multiple-buffer technique is useful to explore the above capability; one for input, one for processing and one for output. Thus, data input, output and processing can be paralleled. The buffers are toggled at the synchronization point.

3.1.2. MACRO implementation

The VAX FORTRAN is considered a highly optimized compiler. However, assembly coding is normally efficient than the FORTRAN code. Nevertheless, it is quite difficult to implement the entire code in assembly. In general, 90 % of the processing time is taken by 10 % of the total code. Hence, the software can be optimized by writing this small 10% part of the code in assembly language.

The throughput of any image-processing algorithm can be expressed by equation

\[ T_p = T \times N + 8T \]

Where,
Optimization of image process application on VAX/VMS system

\[ T_p = \text{Total time taken by program.} \]
\[ N = \text{size of the data.} \]
\[ T = \text{time taken by a set of instructions executing most frequently.} \]
\[ \delta T = \text{Time taken by rest of the code.} \]

The task of the optimization is to minimize \( T \). \( T \) is a function of number of instruction and time taken by each instruction. One has to optimize between number of instructions and their speed. The assembly language reference manual gives the required details [14].

3.1.3. Selection of Buffer length

The data input/output time is a function of the head-seek time and read/write time. Hence, usage of larger buffer for input/output can make great difference in time due to less head-seek overheads. Furthermore, the buffer size has to be consistent with the computation. The input/output should be over, before the computation at the synchronization point, so that CPU utilization is maximized. This buffer length varies from application to application. It can be obtained by experiment.

The VAX/VMS supports maximum buffer length equal to 64 KB. Hence, one has to issue multiple QIO, incase the buffer length is higher than 64 KB. This will improve the IO time provided the data is not fragmented. The VAX/VMS provides the facility to store the data in continuous form.

3.1.4. Efficient Usage of array processor

The array processor (AP), FPS 5000, has 256 KWs system common memory in addition to the 8 KWs table memory RAM and 8 KWs table memory ROM. It provides six instructions, which can be executed in parallel. The array processor can be used efficiently by adopting instruction pipeline processing mechanism. The overhead of calling mathematical library routines can be reduced largely by using VFC (vector function chainer).

The MACRO (assembly language) utilization philosophy adopted for VAX processor holds good for array processor in optimizing among AP executable functions, AP mathematical library and APAL (Array Processor Assembly Language) code. The usage of AP should also be optimized with respect to the data transfer overheads between VAX and AP. The array processor being a dedicated processor for floating point computation, it should be used maximally for such
Optimization of image process application on VAX/VMS system

processing, leaving VAX processor for other jobs. Both, VAX processor and AP, usage should be optimized in case the software has to be executed in standalone mode.

An experiment was conducted to study the optimum utilization of two array processors. An application was written to process an image using both the array processors simultaneously. Another option tried out was to use two array processors for two different images independently. The second option gives a linear gain, as it does not call for any synchronization overheads. The second option was found to be approximately 30% more efficient compared to the first option. Furthermore, it reduces the application complexity. The experiment further ensured that the scheduler environment does not have any overhead for such scheduling.

3.1.5. Avoiding memory data transfer

The image processing application deals with large volume of data. It processes the data using multiple buffers one by one. Many applications require some overlap between the two successive buffer operations. The buffer-to-buffer data transfer mechanism is normally used for this purpose. Such an overhead can easily be curtailed using pointer mechanism. Fortran does not support pointers. It can be simulated over a single large buffer instead of small multiple buffers.

The following VAX/VMS run time library routines support the dynamic allocation of virtual memory:

LIB$GET_VM(No of bytes, base address,...)
LIB$FREE_VM(No of bytes, base address,...)

This base address can be toggled in Fortran program, if one wants to use multi buffers.

3.1.6. Parallel usage of multiple disk drives

The capability of multiple disk drives working in parallel can be exploited by using separate disk drives for input data and output data. This can give better efficiency due to parallel operation and less head-seek time. This guideline is very useful for stand-alone applications. The probability of all disk usage is normally equal for multi processing environment. However, the operations manager should monitor the disk usage and suggest the optimum utilization plan to the designers.
Optimization of image process application on VAX/VMS system

3.2 Case study and Results

The basic optimization philosophy was first implemented on RADOCR software. The results were very encouraging. The philosophy was verified with the implementation of GEOCOR and PCTGEN.

3.2.1 Radiometric Correction

The RADOCR optimization was carried out in different phases using the above philosophy. The total throughput gained is given in table 4.1 and throughput achieved during the different phases of optimization is listed in table 4.2.

<table>
<thead>
<tr>
<th>Table 4.1: Total throughput of RADCOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Software Version</td>
</tr>
<tr>
<td>Un optimized</td>
</tr>
<tr>
<td>Optimized</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Table 4.2: Throughput achieved during optimization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sr</td>
</tr>
<tr>
<td>----</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>4</td>
</tr>
</tbody>
</table>

3.2.1.1 Paralleling I/O and computation

Triple buffer mechanism was used to make input, computation and output to function in parallel as far as possible. The three small buffers were simulated into a single large buffer using three pointers to avoid internal data buffer transfers. The
total time was reduced from 10 minutes to 7 minutes by adopting this technique.

3.2.1.2 Macro Implementation

During the study of the RADCOR software, a portion, which consumed the largest time, was identified. This identification was carried out by

- Monitoring the program execution
- Inserting lib$date_time function in application
- FORTRAN code walk through.

Seven-line code found in the inner most loop of the small portion is given in table 4.3(a).

<table>
<thead>
<tr>
<th>Table 4.3: Assembly code comparison</th>
</tr>
</thead>
<tbody>
<tr>
<td>(a) Fortran code</td>
</tr>
<tr>
<td>lut_pnt = raw_video(src_pnt) + (j + 1)*128 + base_adr</td>
</tr>
<tr>
<td>src_pnt = src_pnt + 1</td>
</tr>
<tr>
<td>gray = rad(lut_pnt)</td>
</tr>
<tr>
<td>raw_video(des_pnt) = lgray(1)</td>
</tr>
<tr>
<td>des_pnt = des_pnt + 1</td>
</tr>
<tr>
<td>hist_pnt = hist_pnt + gray</td>
</tr>
<tr>
<td>hist_gram(hist_pnt) = hist_gram(hist_pnt) + 1</td>
</tr>
</tbody>
</table>

The FORTRAN compiler provides the facility to convert the FORTRAN code into assembly code. This assembly code for the small portion was obtained using the same. The Fortran compiler generated 11 instructions in the innermost loop. It is given in table 4.3(b).
Table 4.3 (b) **Compiler generated assembly code**

```
CVTBL raw_video 1[R8], R0
SUBL3 #1, R10, R1
ASHL #7, R1, R1
ADDL2 R1, R0
ADDL2 R2, R0
INCL R8
CVTBL rad 1[R0], gray[R11]
MOVB lgray[R11], raw_video 1[R7]
INCL R7
ADDL3 gray[R11], R4, R0
INCL hist_gram 4[R0]
```

These 11 lines were reduced to 5 in case of hand coded assembly language. The tests were conducted and the most efficient instructions were selected. The instructions used in the hand-coded assembly language were simple. Those were taking less time than that of assembly language instruction generated by the compiler. A list of these 5 hand-coded instructions is given in table 4.3 (C).

Table 4.3 (C) **Hand coded assembly code**

```
MOVB (R4)+, R1 extracting raw data
MOVB (R9)[R1], R1 applying LUT
MOVB R1, (R6)+ updating raw data
INCL (R5)[R1] incrementing histogram
ADDL2 R2, R9 setting next LUT base address
```

Approximately 3 minutes of elapsed as well as CPU time was gained during this 4.10
Optimization of image process application on VAX/VMS system

phase of optimization (see table 4.2).

3.2.1.3 Buffer length selection

RADCOR software was modified in such a fashion that the input, computation and output buffer size could be selected dynamically for target system. The effect of the optimum buffer size over the throughput was worked out.

The buffer length of the VAX I/O controller is 64 KB. Hence, it was expected to be best. However, the FSDU design restricted the buffer length to 48KB. The study was carried out amongst various buffer sizes starting from 48KB 96KB, 144Kb and larger. Multiple QIOs of 48KB size were issued in all the cases. The size of 48 KB was enforced by the RADCOR data file format. Otherwise, 64 KB size would have been more efficient. It was found that the optimum buffer size was 96KB.

The 45 seconds of elapsed time and 20 second of CPU time were gained using 96KB data size in comparison with the second best.

3.2.1.4 Parallel usage of multiple disk drives

The elapsed time was improved by 17 seconds by using separate disks for input and output.

3.1.1. Geometric Correction

The geometric correction software (GEOCOR) was optimized using the above philosophy by image processing scientists (Mr. Mritunjay and Mr. Nikunj Darji) with the help of author, (Haresh Bhatt). The total throughput gained is given in table 4.4

| Table 4.4: TOTAL Through put of GEOCOR (4 band PHOTO PRODUCT) |
|-----------------------|-----------------------|
| S/W Version           | Time (hh:mm:ss.mm)    |
|                       | Elapsed               | CPU                        |
| Old unoptimized GEOCOR S/W | 00:41:09 03 | 00 23:33 32 |
| New optimized GEOCOR S/W | 00:14:23.16 | 00:05:19.03 |

4.11
Optimization of image process application on VAX/VMS system

The following sensitive areas were selected for GEOCOR optimization:

a. Efficient utilization of array processor:

a.1. Implementation of Cubic Convolution resampling using full 256 Kilo Words (KW) of Array Processor (AP) main data memory and 8 KW of Table Memory™ RAM. Look Up Table (LUT) approach is employed for implementing the cubic convolution algorithm by using the TM RAM as pre calculated coefficients.

a.2. Interpolation to generate secondary points is done in AP, which was earlier done on VAX.

a.3. Merging of enhancement and camera normalization look up tables to one look up table and then applying it in array processor itself saved some precious minutes in GEOCOR.

b. Concurrent execution of I/O and computation was achieved.

c. Left and right annotation patching was performed in dual buffer mode using pointer-toggling mechanism.

d. Integration of interpolation module and resampling module in to one VFC (vector function chainer) call module for array processor.

The detailed time gains for each criterion is shown in table 4.5

<table>
<thead>
<tr>
<th>Area of Optimization</th>
<th>Time-gain (mm:ss)</th>
</tr>
</thead>
<tbody>
<tr>
<td>LUT implementation algorithm for cubic convolution</td>
<td>6:03</td>
</tr>
<tr>
<td>Interpolation to generate points in AP</td>
<td>3:28</td>
</tr>
<tr>
<td>Paralleling disk I/O and Computation</td>
<td>3:21</td>
</tr>
<tr>
<td>Merging of enhancement and camera normalization LUT's and then application in AP</td>
<td>9:6</td>
</tr>
<tr>
<td>Dual buffer mechanism for patching left and right side annotation by pointer toggling mechanism</td>
<td>3:6</td>
</tr>
<tr>
<td>Single VFC module for AP for doing interpolation and resampling</td>
<td>1:04</td>
</tr>
</tbody>
</table>

Table 4.5 : GAIN IN ELAPSED TIME AT DIFFERENT STAGES
(for 4 band photo product)
Optimization of image process application on VAX/VMS system

3.1.2. PCTGEN Correction

The function of the PCTGEN software is to read the input data from disk, perform basic enhancement functions if required, and write the output on tape in a prespecified format for generating UCCT or PCT.

The maximum time consumed by this S/W is for disk input and tape output. The dual buffer technique was implemented for concurrent execution of the disk input and tape output. The elapsed time was improved from 15 minute to 5 minute in doing the same.

The above philosophy helps the image-processing scientists to visualize the efficient implementation model. It was also concluded that such implementation requires software expertise. One cannot expect it from the image-processing scientists. It was also observed during the optimization that the computer professionals would need the help of image-processing expert. The computer professionals in ISRO were involved mainly in H/W and system programming until date.

4. Operating under scheduler environment

The data product scheduler does the pipeline processing among the various processes. Hence, it was suspected, by some scientists, that the time gained by optimization would not be so effective under the scheduler environment. Both the software (optimized and unoptimized) were tested several times under the scheduler and non-scheduler environment several times. The timings achieved are listed in table 4.

<table>
<thead>
<tr>
<th>Type of product</th>
<th>Number of products</th>
<th>Session timings (Min)</th>
<th>Time-gain</th>
<th>%gain</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Non-scheduler</td>
<td>Scheduler</td>
<td></td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>1510</td>
<td>1035</td>
<td>475</td>
</tr>
<tr>
<td></td>
<td>40</td>
<td>3020</td>
<td>2045</td>
<td>975</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>7550</td>
<td>5075</td>
<td>2485</td>
</tr>
<tr>
<td>Standard Product Using Optimized S/W</td>
<td>10</td>
<td>244</td>
<td>164</td>
<td>80</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>488</td>
<td>311</td>
<td>177</td>
</tr>
</tbody>
</table>

Table 4.8: Session timings
Optimization of image process application on VAX/VMS system

Approximately 30% to 37% time with respect to non-scheduler version is gained under scheduler environment for both types of software (optimized and unoptimized). Infact, the optimized software gave better efficiency.

This percentage increases with number of products due to the typical pipeline-scheduling pattern.

Above results help us in making some useful combinations. Let us take the case where only GEOCOR software is optimized and RADCOR as well as PCTGEN is not optimized. It is expected to take almost a similar time as the RADCOR (10 minutes) & PCTGEN (15 minutes) will complete in parallel to GEOCOR (15 min). This means that the scheduler is able to explore the virtual parallel processing capabilities. This was tested for 20 products. The total time taken by the scheduler was 333 minutes. This gives the time-gain of 155 minutes or 31.76%. Thus, the scheduler is able to provide the better throughput even if only one critical module is optimized. The remote-sensing scientists should isolate the most time consuming portion of entire package and implement it efficiently.

5. Conclusion

The software optimization gave almost three-fold speed up. This is quite encouraging. However, it is also observed that such implementation requires skilled programmers. Hence, it cannot be expected from the remote-sensing scientists. Furthermore, some of the remote-sensing experts were concerned about their intellectual property rights too incase of involvement of skilled programmer in every application.

Furthermore, the scheduler system helps us in reducing the optimization effort. We need to optimize only the critical software.

It was decided to make a similar study for a parallel machine like transputer. It was also concluded to work out a generalized scheduler supporting most of the remote-sensing applications.