DISTRIBUTED COMPUTING IN GROUND PROCESSING

Belay T. Beshah [a], John Welter [b], Kristian Morin [b]

[a] Leica Geosystems GIS & Mapping, LLC, 10840 Thornmint Rd. Suite 100, San Diego CA 92127, USA - belay.beshah@gis.leica-geosystems.com
[b] North West Geomatics Inc., Suite 212, 5438 11st NE, Calgary, Alberta, CANADA T2E 7E9 - (jwelter, kmorin)@nwgeo.com

Commission III, WG III/8

KEY WORDS: Distributed, Automation, Processing, Digital, Sensor, Rectification, Matching, Performance

ABSTRACT:

In the recent years, airborne digital imaging sensors have gained acceptance in the photogrammetric workflow. However, the processing and management of data acquired by these sensors requires an enormous computational effort, which is often too high for single processor architectures. This demand for processing power stems from the large amount of data being generated and the high rate of automation possible in ground processing. Distributed computing, the method of dividing large processing problems into smaller tasks that can run on individual systems, has emerged as a key enabling technology for digital photogrammetric workflows.

Using networks of workstations in distributed computing to solve large problems has become popular owing to the proliferation of inexpensive, powerful workstations. Clusters offer a cost-effective alternative to batch processing and an easy entry into parallel computing. The main advantage is the potential for future performance enhancement that results from the high rate of advances seen in computer and network hardware, scalability, fault tolerance and rapid development of applications. This paper summarizes a range of distributed computation technologies surveyed, design criteria used for choosing a solution and the results obtained in the ground processing workflow of the Leica Airborne Digital Sensor, especially in rectification and automated point matching. We conclude by presenting the results from real applications indicating timesaving and benefits of the distributed computing model in a photogrammetric production department.

1. INTRODUCTION

The digital sensor revolution happening right now will result in an information explosion. The processing and management of data acquired by digital sensors requires an enormous computational effort. The Aerial Digital Sensor (ADS40) from Leica Geosystems GIS & Mapping (LGGM) is already ahead of the curve in generating tremendous amount of data. Pretty soon other digital sensors will also be creating large quantity of data and face the same problems. This demand for large processing power stems from the large amount of data being generated and the high rate of automation possible in the ground processing. Using single computers one could not exploit the full automation possibility of digital acquisitions.

Leica's Airborne Digital Sensor is a high performance digital sensor, capable of delivering photogrammetric accuracy with multi-spectral remote sensing quality imagery. The ADS40 is a three-line-scanner that captures imagery looking forwards, nadir and backwards from the aircraft. Every portion of the ground surface is imaged multiple times. With its simultaneous capture of data from three panchromatic and four multi-spectral bands, the ADS40 provides unparalleled qualities of image and position data. Digital workflow from flight planning through acquisition to product generation is one of the major advantages of the sensor. The design principles and advantages compared to film cameras are discussed in detail in (Sandau, 2000).

The ADS40 generates around 100GB of raw data per hour if all the CCD lines are active. This data passes through different levels of processing before products are generated; these processes are well documented in (Tempelmann, 2000). The first step in the workflow is downloading the data and adding the geo-positioning information using data supplied by a Position and Orientation System from Applanix Corporation. The geo-referenced images are then rectified to a plane to create stereo-viewable and geometrically corrected Level 1 images. The Level 1 images are also useful for the next process which is automatic point matching. Based on the accuracy requirements, the images are then triangulated using ORIMA, LGGM's triangulation package. Finally, ortho-photo images using existing DTM or DTM generated from the triangulated images are created. Due to the large amount of data being processed this workflow could take an extended time. GPro is the ground processing software for the ADS40 that controls the workflow described. It provides the full functionality to download, generate geo-referenced and ortho-rectified images from the recorded imagery and positioning data. It consists of highly optimized and threaded applications that are designed to handle large data sets. In addition, GPro also allows the user to perform various automated process and data management tasks.

The last two years have shown that digital sensors, especially the ADS40, are able to meet the accuracy and radiometry required for large-scale ortho-photo generation. However, the quick turnaround time these sensors provide has been a challenge to realize to its full potential using a single computer. High volume production businesses like North West Geomatics (NWG) have already risen the bar for flight to product

turnaround times using the ADS40. This problem can only be addressed using High Performance Computing (HPC). High Performance Computing is defined as the technology that is used to provide solutions to problems that either require significant computational power or need to process very large amounts of data quickly. In HPC, enormous processing power, fast networks, and huge amounts of data storage are applied to solve complex problems by utilizing industry standard, volume components. HPC clusters exploit recent performance advances in commodity processor and networking technology to make supercomputing affordable and accessible.

This paper endeavors to summarize the challenges of dealing with the considerable amount of data being generated by digital sensors and the need for fast turn around times. It also shows how the computationally intensive processes in ADS40 ground processing are distributed to an HPC cluster. We conclude by indicating our current processing capacity and the long term plans of how to meet these challenges.

2. SOLUTION

North West Geomatics were one of the early adopter of the ADS40. They were quick to point out that if they were to fulfill their business plan for fast turn around of projects they needed more than a highly optimized solution. To address this problem Leica started looking at various technologies for parallel or distributed computing. The following criteria were used to select the best HPC technology that addresses our customer's problems and give us fast to market solutions:

- Minimum changes to current software
- Easy to set up and configure on the client's computer
- Be able to schedule jobs
- Work on Microsoft Windows platform
- Prefer to use idle cycles of workstations

The minimum changes to the current software requirement results from the fact that we already had a working solution and didn't want to burden a simple user with the set up and configuration of a cluster of computers. GPro's applications are highly threaded and optimized, so for production shop that dealt only with a small amount of data at a time they could continue using the already existing solution with out complicating their workflow. The next requirement, ease of deployment was a major criteria in our selection process. Most of our users do not have a large IT department with the knowledge and the budget to deploy, configure and fine-tune a dedicated cluster. One of our main goals in testing has been to see how easily the solution works out of the box, what the minimum hardware and software requirements are.

As in any time consuming process, being able to schedule jobs for processing, setting their priorities, ability to cancel them is critical in fully utilizing a computational resource. Large volume production projects come in stages and there is a large contention for the resources. A good job scheduling tool will allow easy prioritization of jobs, monitoring and above all fault tolerance since hardware failure is imminent in a cluster environment. In addition, since Microsoft Windows is our main development environment the tool selected for distribution has to be mature and well supported on this platform. Most of the tools in HPC are available only for Linux and finding mature implementations of libraries for Windows was a challenge. Lastly, production shops have high-end workstations that are used for triangulation, feature collection and quality control. These high-end machines are busy during the day when the operators are working but sit idle during the night. It was considered an added bonus if the solution was able to take advantage of this computational power during off hours.

Based on these criteria's the following technologies were selected for investigation:

- MPI: Message Passing Interface
- PVM: Parallel Virtual Machine
- Condor: High Throughput Job Scheduler
- DCOM: Distributed Component Object Model

There are varieties of other distributed and parallel computing technologies not covered here. The above were selected since they are extensively used in the scientific community to solve large memory and CPU intensive numerical problems and are most relevant for the Microsoft Windows platform.

There were two choices in using the different parallel/distribution technologies. The first was to write proxies that use these technologies to distribute the load across machines and continue to use the current programs with the minimum change as possible. This, however, implied that the granularity of distribution has to be large enough to fit the existing applications that work on image-by-image basis. The other solution was to rewrite all our computationally intensive application using parallel methodologies. Parallization will have resulted in the largest scalability since it allows a production shop to reduce the turn around time from flight to product generation by just adding more computation hardware. However, this revolutionary approach was rejected from the start because of time constraints since it will require a large redesign and implementation effort.

The following sections will point out the advantages and disadvantages for each proposal. Except for Distributed Component Object Model, which was purely based on literature survey, all other options were installed and experimented with.

2.1 Message Passing Interface (MPI)

MPI addresses the message-passing model of parallel computation, in which processes with separate address spaces synchronize with one another and move data from the address space of one process to that of another by sending and receiving messages (Sterling, 2002). MPI is a standardized interface (not an implementation) that is wildly used to write parallel programs. It was designed by a forum that sought to define an interface that attempts to establish a practical, portable, efficient, and flexible standard for message passing. There are a lot of MPI implementations ranging from free open source multi-platform libraries to highly optimized ones for a specific hardware. This range of choices allows an MPI program to be portable across multiple HPC platforms from simple cluster to a supercomputer.

Writing proxies based on MPI turned out to be easy for the simplistic approach of distributing the work based on individual images or strips. The prototypes were developed using the public domain implementation MIPICH (Ashton, 2002) for windows. Configuration and set up of clusters using MPI was straightforward for the Windows platform. MPICH even had a plain GUI for feedback and control of running jobs. Image based job distribution was carried out with minimum

modification to the existing software. This implementation was also instrumental in introducing us to the large potential available in parallel computing. One major disadvantage of the MPI implementation was that it required dedicated nodes for computation, which renders most of the idle high-end workstations unusable. Even though, simple programs could be written easily it requires a significant learning curve to get the maximum benefit out of parallel computing model of MPI. Some of the issues that need to be addressed in parallel programs are minimization of communication, load balancing and fault tolerance. The scheduling, monitoring and managing of jobs are not addressed and would require integration with another tool.

2.2 Parallel Virtual Machine (PVM)

PVM is an integrated set of software tools and libraries that emulates a general purpose, flexible, heterogeneous parallel computing framework on interconnected computers of varied architecture (Sterling, 2002). It is a portable message-passing programming system, designed to link separate host machines to form a ``virtual machine" which is a single, manageable computing resource (Geist, 1994).

The biggest advantage of PVM is that it has a large user base supporting the libraries and tools. Writing our proxies using PVM allowed us to leverage this knowledge that resulted in the least amount of modification to the existing applications. It also introduced us to parallel programming. The disadvantages were that it was not easy to set up and not mature in the Windows platform. It has also been superseded by MPI and is considered not as fast. As in the MPI case, job scheduling and management is not addressed.

2.3 Condor: High Throughput Job Scheduler

Condor is a sophisticated and unique distributed job scheduler developed by the Condor research project at the University of Wisconsin-Madison Department of Computer Science (Sterling, 2002). *"Condor is a specialized workload management system for compute-intensive jobs. Like other full-featured batch systems, Condor provides a job queuing mechanism, scheduling policy, priority scheme, resource monitoring, and resource management. Users submit their serial or parallel jobs to Condor, Condor places them into a queue, chooses when and where to run the jobs based upon a policy, carefully monitors their progress, and ultimately informs the user upon completion."* (Condor Team, 2003). It provides a high throughput computing environment that delivers large amounts of computational power over a long period of time even in case of machine failures. Condor has more extensive features that are well documented in the manual and appear in many publications from the research team.

Condor proxies were developed to submit jobs to the Condor pool and monitor its progress using the Condor command line executables. The most impressive thing about Condor was that set up was a breeze and everything worked right out of the box. The ability to effectively harness wasted CPU power from idle desktop workstations was an added benefit that gave Condor an edge over the others. The scheduling, monitoring and inbuilt fault tolerances were also features that could not be matched by any of the other HPC models. Moreover, since it could control both serial and parallel (MPI, PVM) programs it provides us with growth potential.

The vanilla universe (which is Condor terminology for serial job management) directly matched our distribution model. In writing the proxies all we had to do were minor changes to each application to work on the subset of the input XML files and modify the job submission text files of Condor to kick of a batch file that sets up the shared drives. The major difficulty in using Condor was the absence of an API that allowed easy third party integration to monitor/manipulate jobs and machines. Another shortcoming was the lack of GUI's for job control and feedback. Clipped functionalities on the windows platform (no transparent authentication & failure handling, automatic drive mappings etc) were some of the minor problem we came across at the beginning that has been partially addressed since then in newer versions of Condor.

2.4 Distributed Component Object Model (DCOM)

DCOM is a protocol that is an extension of the Component Object Model (COM) from Microsoft that uses remote procedure calls to provide location independence by allowing components to run on different machine from the client. It enables software components to communicate directly over a network in a reliable, secure, and efficient manner. The theory of DCOM is great, but the implementation has proved less than stellar (Templeman, 2003). By the time we started evaluating distributed computing, DCOM has lost favor to newer technologies such as web services (Templeman, 2003).

The job management for COM components could be handled by Microsoft Application Center 2000 that is designed to enable Web applications to achieve on-demand scalability and mission-critical availability through software scaling, while reducing operational complexity and costs (Microsoft, 2001). As a full GUI based product, it appears to be uncomplicated to configure and set up Application Center clusters. However, it is mainly geared towards web service and our research did not turn up how it could be used for other types of applications. It was also not clear how one would write a component that could be load balanced. Moreover, this solution will also require a dedicated computation farm and has a high software price for each server added to the cluster.

2.5 Selected Solution

Condor with its ease of set up and high throughput computing was finally selected for implementation. Proxies were developed for each computationally intensive process. These proxies submit the job, get the status reported from each node and summarize the results as shown in Figure 1. The installation assumes that all the nodes have fast access to the file servers. The maximum output from this configuration will be utilized by high-end fiber array Storage Area Network (SAN) that provide high read and write performance. Using Windows Distributed File System (DFS) or different input and output locations from various file servers could also avoid I/O bottleneck.

The following section will show a typical set up that uses Condor and the proxies developed at LGGM. We will finally summarize the actual result attained for some projects.

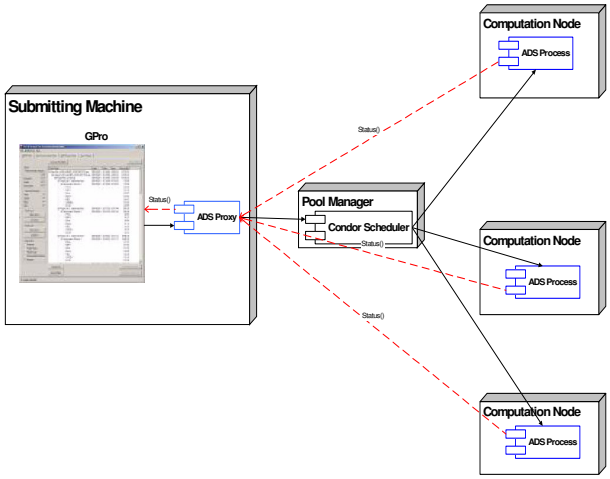

Figure 1. Distribution Model

3. SETUP

3.1 Requirements

The installation should be done on a network domain that has a valid domain controller to avoid authentication problems when mapping network drives. The different machines involved in Condor HPC solution are:

Pool Manager: acts as the central Condor pool manager that matches the jobs with the available machines. This is the most important machine and shouldn't run jobs in a production environment since it needs to be running all the time.

Submitting Machines: these are the machines that could submit jobs to the cluster. Another option is to use one machine as the gateway to the cluster and submit jobs only from that machine. The latter has the advantage of localizing the different run time option used to optimize the distribution.

Computation Nodes: these are the workers that do the computation. These nodes could be dedicated rack mounted machines or any machine in the office that is connected to the network. Workstations could also be configured to be both submitting and computation nodes.

File Server(s): these are machine that provide the disk storage. The shared data files need to be on a Windows Server family OS or use Samba from Unix machines since Windows 2000/XP will only allow up to three clients to map its shared drives.

License Manager(s): machines used to provide the LGGM floating licenses to be used by the submitting and computational nodes. The opportunistic nature of GPro may sometimes result in more machines being available at one time than there are licenses. The Condor proxies of GPro correctly handle this by resubmitting the job until it gets a license and runs to completion.

Accordingly before installing a GPro Condor pool a user will have to select the pool manager machine, machines that are going to be used to submit GPro jobs (this are most of the time operators machines) and computation nodes. It is possible to combine the Pool/File/License Manager to the same machine if it has the capability.

3.2 Setup

From a users point of view, the work required to set up the HPC environment is minimal. Simply install Condor on the workstations and change the program names in the GPro preferences to their respective proxy equivalent. For example, the rectifier "ADSRectifier.exe" is changed to "ADSRectifyCondorProxy.exe". According to their needs users could mix and match which processes to distribute and which processes to run locally. The following steps explain how to install a HPC cluster for GPro using Condor.

Condor Setup:

1. Download Condor from "http://www.cs.wisc.edu/condor/".
2. First install Condor on a server that is going to be the Condor Pool Central Manager.
3. Install Condor on all the computation nodes.
    - Dedicated computational nodes should be configured so that their 'Job Start Policy' is "Always Run Condor jobs". This defines a very responsive pool.
    - All other workstations, for example that operators work on, should be configured using the default start job after 15 minutes of low CPU activity policy.
4. Test the condor installation using the examples distributed with Condor.

GPro Setup:
There is nothing special about the Condor pool used by GPro. The only assumption made by the installation is that all the nodes have access to shared file servers.

1. Select a shared file system to be mapped by all the computation nodes.
2. Install and configure the Leica licensing client on all workstations that run and submit jobs to the pool.
3. Install the distributed version of GPro that sets up the required proxy executables and configuration files to submit jobs.
4. Modify the default Condor job submission files to reflect the selected shared file system.
5. Change the program names to the proxies in the general preferences of GPro on all the submitting workstations
6. Start submitting jobs to the pool.

For easy upgrades GPro should only be installed on the server and not on each of the computational nodes. A typical distributed run is shown in Figure 2 below.

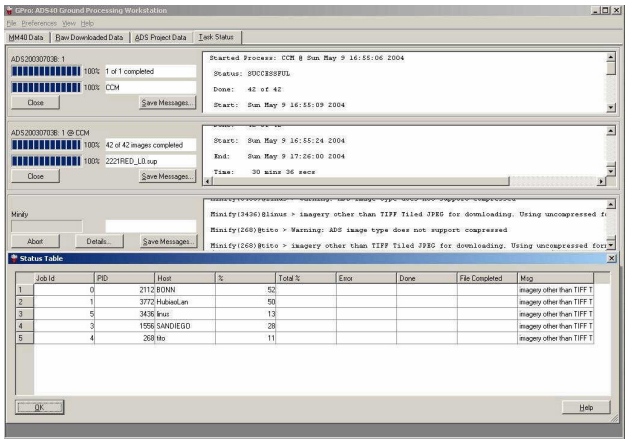

Figure 2. GPro run in a Condor cluster

4. PRACTICAL RESULTS

Two recent small projects were selected to show the timesavings in the computationally intensives processes. These projects were done using a portion of the cluster setup at NWG. The production cluster at NWG comprises of dedicated rack mounted systems in conjunction with operator's machines during off hours. All the single runs were performed on a dual Xeon 3.06 GHz with 3GB RAM running Windows XP. A RAID0 disk array with 10 disks of 146GB SCSI drives was used for all data input/output. Analysis of the average I/O queue during runs showed that disk contention was not a serious issue for a single machine run. The distributed runs were done on six nodes of same hardware specification as the single run. Interconnect between the nodes and file server was done using gigabit Ethernet. Disk I/O analysis for the distributed runs showed some disk contention.

The first dataset used is a small block from a project that was part of the Florida statewide program undertaken in 2003/2004 by North West Geomatics and Earth Data. This block is roughly one degree by one degree in size. It comprises of 256 USGS Digital Ortho Quarter Quads (DOQQs), delivered in both color and FCIR format. The numbers contained here are for color data only, but one could simply double the Ortho times for both datasets. This block was flown at 24,000 AGL with a 5-minute line spacing at 340 knots. Seven control points were used in the block to provide a datum shift. The DEM was the USGS format DEM converted into a 50m grid (flat terrain so the 50m grid is more than sufficient).

The second dataset is a small test area over Zolloffo springs in Florida, which is of environmental concern. It was flown on 850m line spacing to capture 0.13m data for a final product of 0.15m. Flight speed was 100 knots. Five control points were available in this area. Both color and FCIR ortho photos were also generated for this area. Table 1 summarizes the characteristics of the projects and Table 2 lists the processing time for the most time consuming processes of the workflow.

	Florida	Zolloffo
Capture GSD (m)	0.8	0.13
Output GSD (m)	1.0	0.15
Lateral Overlap (%)	30	30
Raw Data Size (GB)	110	21
Number of Lines	14	4
Average Line Length (km)	120	5
Project Area (km ^{2})	11500	4.5

Table 1: Dataset Summary

Process	Florida		Zollofo	
	1 Node	6 Nodes	1 Node	6 Node
Rectification	11.27	1.92	2.34	0.60
Automatic Point Matching	14.80	4.66	3.14	1.50
Ortho Generation	14.65	2.54	3.02	0.78

Table 2: Processing Times in Hours

As could be seen in the table, even though the processing time seem to decrease linearly, the cluster efficiency numbers drop for Zollofo processing due to only four nodes being utilized even though there were six available.

5. CONCLUSIONS

Distributed computing has enabled us to process large amounts of data in a reasonable time. It has empowered large mapping projects to be done in unprecedented turnaround times. Condor with it high throughput performance has enabled us to deliver products even in adverse condition when there have been node failures without human intervention. We plan to extend our usage of the Condor distribution model to other non-interactive applications so that they could benefit from all the advantages discussed above. However, in order to provide the scalability required we would have to increase our granularity and parallalize our applications.

6. REFERENCE

Ashton, D., Gropp, W., Lusk, E., 2002. Installation and User's Guide to MPICH, a Portable Implementation of MPI Version 1.2.5. The ch_nt device for workstations and clusters of Microsoft Windows machines, http://www-unix.mcs.anl.gov/mpi/mpich/ (accessed Feb. 2003).

Condor Team, 2003. Condor Manual, http://www.cs.wisc.edu/condor/manual/v6.4/, pp. 7-8 (accessed Feb. 2003).

Geist, A., et al., 1994. *PVM: Parallel Virtual Machine A Users' Guide and Tutorial for Networked Parallel Computing*, MIT Press, pp. 10-11.

Microsoft, 2001. Application Center 2000 Evaluation Guide, http://www.microsoft.com/applicationcenter/evaluation/productguide.asp, pp. 3-5 (accessed Feb. 2003).

Sandau, R., et al., 2000. Design Principles of the LH Systems ADS40 Airborne Digital Sensor. In: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Amsterdam, Netherlands, Vol. XXXIII, Part B1, pp. 258-265.

Sterling, T., 2002. *Beowulf Cluster Computing with Windows*, MIT Press, pp. 167-168, 235-236, 307-308.

Templeman, J., Mueller, P., 2003. COM Programming with Microsoft .NET, Microsoft Press, pp. 172-173.

Tempelmann, U., et al., 2000. Photogrammetric Software for the LH Systems ADS40 Airborne Digital Sensor. In: The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, Amsterdam, Netherlands, Vol. XXXIII, Part B1, pp. 552-559.