Disaster Monitoring using Grid Based Data Fusion Algorithms

Cătălin NAE*

*Corresponding author
INCAS - National Institute for Aerospace Research "Elie Carafoli"
Bdul Iuliu Maniu 220, Bucharest 061126, Romania
cnae@incas.ro
DOI: 10.13111/2066-8201.2010.2.4.19

Abstract: *This is a study of the application of Grid technology and high performance parallel computing to a candidate algorithm for jointly accomplishing data fusion from different sensors. This includes applications for both image analysis and/or data processing for simultaneously tracking multiple targets in real-time. The emphasis is on comparing the architectures of the serial and parallel algorithms, and characterizing the performance benefits achieved by the parallel algorithm with both on-ground and in-space hardware implementations. The improved performance levels achieved by the use of Grid technology (middleware) for Parallel Data Fusion are presented for the main metrics of interest in near real-time applications, namely latency, total computation load, and total sustainable throughput. The objective of this analysis is, therefore, to demonstrate an implementation of multi-sensor data fusion and/or multi-target tracking functions within an integrated multi-node portable HPC architecture based on emerging Grid technology. The key metrics to be determined in support of ongoing system analyses includes: required computational throughput in MFLOPS; latency between receipt of input data and resulting outputs; and scalability, processor utilization and memory requirements. Furthermore, the standard MPI functions are considered to be used for inter-node communications in order to promote code portability across multiple HPC computer platforms, both in space and on-ground.*

Key Words: Grid technology, data fusion algorithms, HPC, earth observation.

1. INTRODUCTION

Data fusion involves forming useful relationships between data from different sources to provide salient information which is more readily assimilated. This is a domain of major interest and constant development [1]. On one hand the information age imposes a certain degree of information overload, and on the other hand it can provide the filters, "digesters", and human interfaces, both to highlight key relationships and to suppress extraneous data. These have direct application to economic optimization by decision-makers and their staffs, and by organizations which provide input to decision-makers. Therefore, data fusion is an important part of harnessing information technology, and to obtaining better information for other endeavors.

The major objective in multi-modal fusion is to integrate multiple modalities of information in order to infer knowledge that could not have been supplied by any single modality alone. Information is acquired via some type of sensory devices that have its own interpretation of the sensed environment and thus provides a particularly distinctive set of properties.

There is also important applicability of data fusion to national security and warfighting. Highlighting key relationships, and simultaneously suppressing extraneous data can have

direct application to tactical and strategic decisions. Other key applications are in automatic target recognizers and other automated aids help to both reduce the vulnerability and increase precision.

This paper addresses parallel implementations of multi-modal fusion method in Grid environment, using latest IT technologies, the final result being enhancements in both the interpretation of the scene/environment using parallel algorithms capable of integration/fusing information and in assessment of existing middleware and current implementation for Grid computing. Basic analysis is for typical Earth Observation applications as in Figure 1.

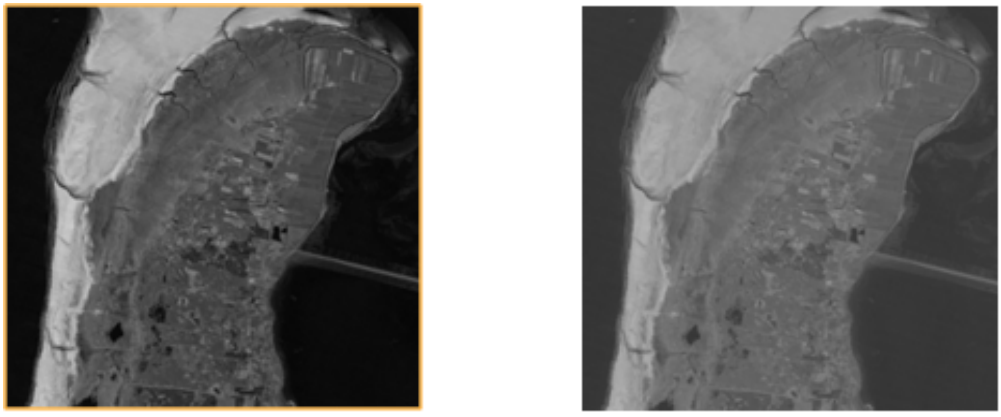

Fig. 1 – Typical LandSAT images in different spectra

2. DATA FUSION MODELS

There are two principles used to integrate multi-modal information. The "Knowledge Source Aggregation" is an accumulation process which tends to maximize the final belief in a given proposition if either the knowledge sources supports the occurrence of this proposition. This enables the sources to corroborate and to ensure that the given feature from either knowledge sources to be represented at the output. Elimination must also be confirmed by the other source, so a feature not detected by one sensor but detected by the other will still be present at the output. The "Belief Enhancement/Withdrawal" principle, on the other hand, adjusts the belief in the second knowledge source by maximizing the similarities between the two. If there is a feature present, then one source of knowledge should validate the existence of that feature which was given by the other knowledge source. The two analytically formulated constraints are the foundation to the fusion technique which shall increase the ability to interpret information available [2].

Our objective is to combine or fuse information M1, provided by one knowledge source S1 with information M2 provided by a second source S2. We assume M1 and M2 are continuous in x-y space, registered and correlated to some extent. The fusion of these two functions can be performed by determining an unknown analytic functional. The fused result can be written as a function of the two inputs as an expansion of converging Taylor series in terms of the powers of the inputs as:

$$M(x,y) = H\left[M_1(x,y), M_2(x,y)\right] = \sum_{i=0}^{\infty}\sum_{j=0}^{\infty} c_{ij} M_1^i(x,y) M_2^j(x,y) \tag{1}$$

where c_{ij} are the unknown coefficients to be determined. For present formulation we consider only the first order approximation to this series. Since all contributions to the output must come from at least one of the inputs we have $c_{ij} = 0$. This leads to a first order approximation given by the following equation:

$$M(x,y) = c_{10} \cdot M_1(x,y) + c_{01} \cdot M_2(x,y) \tag{2}$$

Since both M_1 and M_2 can be normalized, we can use coefficients c_{ij} to normalize M(x,y) as follows:

$$M_{norm}(x,y) = \alpha \cdot M_1(x,y) + \beta \cdot M_2(x,y), \text{ where } \begin{array}{l} \alpha = c_{10} / (c_{10} + c_{01}) \\ \beta = c_{01} / (c_{10} + c_{01}) \\ \alpha + \beta = 1 \end{array} \tag{3}$$

Using the two principles mentioned above, a regularized formulation of the fusion problem can be put in the following form, using Lagrange multiplier technique [3]:

$$\nabla^2 \alpha + A \cdot \alpha_x + B \cdot \alpha_y + C \cdot \alpha = D \tag{4}$$

The solution to the first order approximation becomes that of solving the partial differential equation (4). The fusion result can then be calculated from :

$$M_{norm}(x,y) = \alpha(x,y) \cdot M_1(x,y) + (1 - \alpha(x,y)) \cdot M_2(x,y) \tag{5}$$

The equation (4) needs to be solved using advanced numerical techniques. In practice this consists of high-order matrices whose reachable solution requires careful consideration if the answer is to be obtained in any reasonable amount of time. The choice of the method is often problem-dependent and the problem size will always be a critical factor.

Efficient solution cannot be achieved using exact solution for equation (4). Iterative methods are the correct choice in terms of computational efficiency. Also, important speed-ups can be achieved using preconditioning techniques to enable faster convergence rate.

3. DATA FUSION ALGORITHM

The conjugate gradient method has been a particularly useful and popular method for solving large systems of linear equations. It was first introduced by Hestenes and Stiefel [5]. Parallel CG algorithms involve an additional level of complexity. Aykanat at al. [6] have developed a technique for restructuring the basic CG algorithm for a parallel implementation with minimal communication costs.

The CG method is an iterative technique which starts with an initial vector $\vec{\alpha}_0$ and generates a sequence of vectors $\vec{\alpha}^{(k)}$ by searching the vector space in such a way as to minimize the objective function $\Psi(\vec{\alpha})$. The objective function is considered in the quadratic form as:

$$\Psi(\vec{\alpha}) \approx c - \vec{b} \cdot \vec{\alpha} + \frac{1}{2} \vec{\alpha} \cdot (A\vec{\alpha}) \tag{6}$$

where A is an NxN positive-definite symmetric coefficient matrix, $\Psi(\vec{\alpha})$ will be a convex function having a global minimum where its gradient vector vanishes. Computing the gradient of (6), the CG algorithm converges to the stationary point $\vec{\alpha}^{(k)}$ where:

$$\nabla\Psi(\vec{\alpha}^{*}) = -\vec{b} + A\vec{x} = \vec{0} \Rightarrow A\vec{x} = b \tag{7}$$

The focus is to formulate an objective function which is a representative of the system and the to optimize this function in agreement with its control variables. Then we use Polak-Ribiere [7] implementation of the CG method to develop a minimizing sequence as follows:

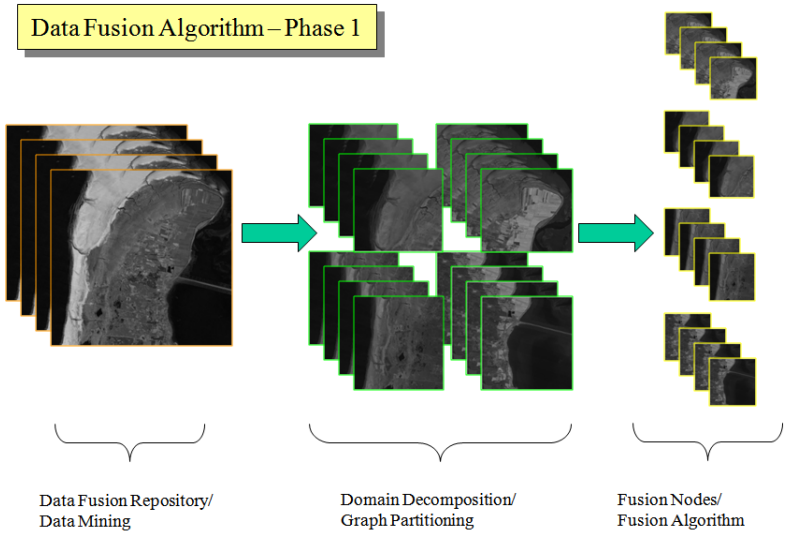

Fig. 2 – Data Fusion Strategy - Phase 1

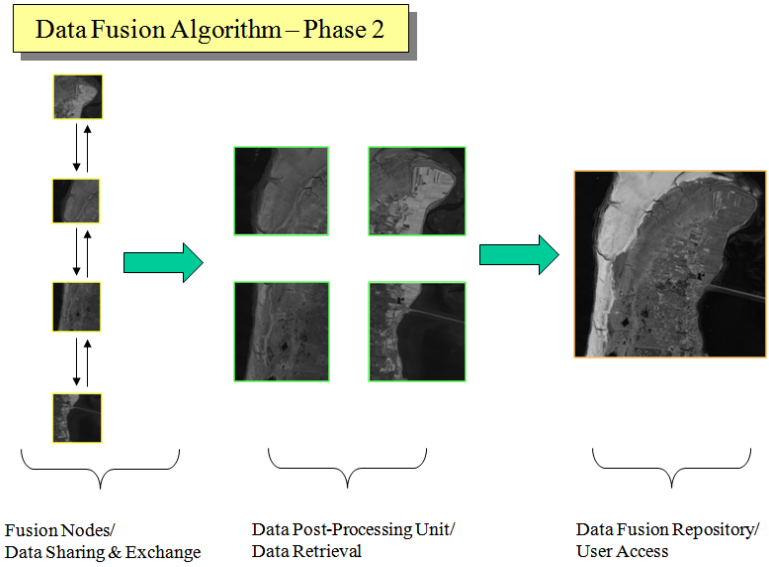

Fig. 3 – Data Fusion Strategy - Phase 2

Step 1 Choose an initial approximation vector $\vec{\alpha}^{(0)}$
If $\nabla\Psi\left(\vec{\alpha}^{(0)}\right)=\vec{0}$, STOP;
ELSE continue

Step 2 Set k=0 and $\vec{g}^{(0)}=\vec{d}^{(0)}=-\nabla\Psi\left(\vec{\alpha}^{(0)}\right)$

Step 3 Compute $\delta^{(k)}>0$ such that $\Psi\left(\vec{\alpha}^{(k)}+\delta^{(k)}\vec{d}^{(k)}\right)=\min\left\{\Psi\left(\vec{\alpha}^{(k)}+\delta^{(k)}\vec{d}^{(k)}\right)|\ \delta\geq 0\right\}$

Step 4 Set $\vec{\alpha}^{(k+1)}=\vec{\alpha}^{(k)}+\delta^{(k)}\vec{d}^{(k)}$

Step 5 IF $\nabla\Psi\left(\vec{\alpha}^{(k+1)}\right)=\vec{0}$, STOP;
ELSE
Set $\vec{g}^{(0)}=-\nabla\Psi\left(\vec{\alpha}^{(k+1)}\right)$
Set $\vec{d}^{(k+1)}=\vec{g}^{(k+1)}+\gamma^{(k)}\vec{d}^{(k)}$ with $\gamma^{(k)}=\dfrac{\left(\vec{g}^{(k+1)}-\vec{g}^{(k)}\right)\cdot\vec{g}^{(k+1)}}{\vec{g}^{(k)}\cdot\vec{g}^{(k)}}$
k=k+1
GOTO ***Step 3***

In order to use CG algorithm, we reformulate the objective function as a system with N^2equations and N^2 unknowns like :

$$\vec{\mu}=\nabla^2\vec{\alpha}+A\vec{\alpha}_x+B\vec{\alpha}_y+C\vec{\alpha}-\vec{D}=\vec{0} \tag{8}$$

We minimize this objective function by finding an $\vec{\alpha}$ which yield the closest value to zero for $\vec{\mu}$. Finally (using (k) for (x,y)) the objective function can be expressed as:

$$\Psi(\vec{\alpha})=\sum_{k=1}^{N^2}\mu_k^2=$$
$$\sum_{k=1}^{N^2}\left[\left(\alpha_{k-1}+\alpha_{k+1}+\alpha_{k-N}+\alpha_{k+N}-4\alpha_k\right)+\frac{A_k}{2}\left(\alpha_{k+N}-\alpha_{k-N}\right)+\frac{B_k}{2}\left(\alpha_{k+1}-\alpha_{k-1}\right)+C_k\alpha_k-D_k\right]^2 \tag{9}$$

Also, for an image of any practical size, in order to be very efficient in terms of number of operations, we compute the gradient for Ψ as :

$$\nabla\Psi(\vec{\alpha})=\left(\left.\frac{\partial\Psi(\vec{\alpha})}{\partial\alpha_\upsilon}\right|_{\upsilon=N+1}^{\upsilon=N(N-1)}\right)=$$
$$\left(\left(2-A_{\upsilon+N}\right)\mu_{\upsilon+N}+\left(2+A_{\upsilon-N}\right)\mu_{\upsilon-N}+\left(2-B_{\upsilon+1}\right)\mu_{\upsilon+1}+\left(2+B_{\upsilon-1}\right)\mu_{\upsilon-1}+\left(2C_\upsilon-8\right)\mu_\upsilon\Big|_{\upsilon=N+1}^{\upsilon=N(N-1)}\right) \tag{10}$$

Implementation of the algorithm making use of high performance computing facilities must be based on an optimum balance between two concepts:

- Algorithm parallelization, based on efficient implementation of the CG method on parallel computers;
- Domain decomposition, based on efficient input data division among existing computational resources based on graph partitioning methods.

Grid environment is able to provide all requested resources for an efficient solution for data fusion, both in terms of costs and reliability. Using this modern virtual environment for solving data fusion problems is now an affordable solution and can be used for future developments. The basic strategy we use is presented in Figure 2 and Figure 3.

4. GRID SYSTEM ARCHITECTURE

A Grid system is a synthetic architecture of hardware and software, composed of distributed, heterogeneous hardware and software, capable of resources sharing and distributed management, so that users can access in order to solve all kinds of complex computational problems effectively and unrestricted.

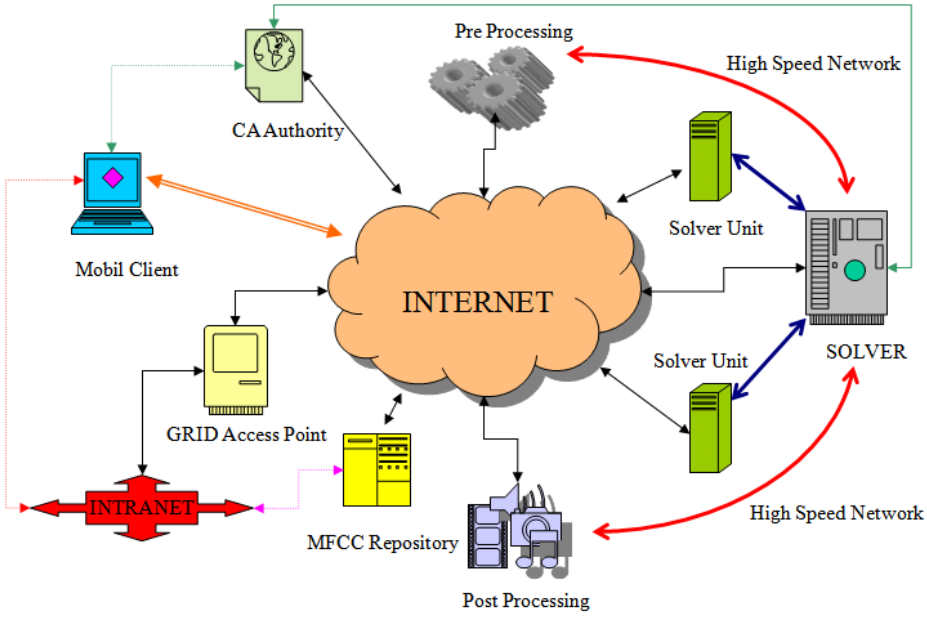

Fig. 4 – CFD system architecture - Mobile Grid Computing

Grid architecture can be looked on as a method that can decentralize essential modules of a system, appoint the purpose and function of system' modules, illuminate the mutual action among modules. At present, there are two architectures: the five layers sandglass architecture and Open Grid Services Architecture (OGSA). Five layers sandglass architecture would disperse the operation and management of share resources according to the distance from each component to share resources. Different from five layers sandglass architecture that center on protocol, OGSA centers on service. In OGSA, service conception includes all kinds of compute resource, storage resource, network, program, database and so on. In short, everything is service. In order to make service thinking more tangible and more concrete, OGSA defines the concept of "Grid Service". Grid Service provides a set of

interface to follow specific tradition to settle service detection, dynamic service establishment, lifecycle management and information etc. The model of OGSA which centers on grid service can achieve grid service through providing uniform kernel interface.

The architecture in Figure 4 is built on an open system paradigm. Source data are ingested, registered to a common geospatial reference, and stored as logically separate SHD (Sensor History Databases). This data is unmodified except for the registration transforms. The sensor data are available for processing by applications (e.g. change detectors, all source track & ID fusion). These algorithms operate in near-real time. Their derived products, along with the pedigrees relating process results to source data, are placed in a situation history data store, available for visualization by request.

Figure 5 explains the work spawning between the master and slave nodes. The master node is made of a AMD64 processor of speed 3.0 GHz, hard disk of 200GB and main memory of 2 GB. Slave nodes are of similar type of machines as the master, but the HDD memory is only 100 GB. All the machines are equipped with Gigabit Ethernet card. These nodes are connected in a network by a 24 ports Gigabit switch. All the machines are using CERN SL Linux OS and LCG middleware. Also, MPI is basic standard for all software developments which enables collection of computers to be cooperatively used for concurrent or parallel computation.

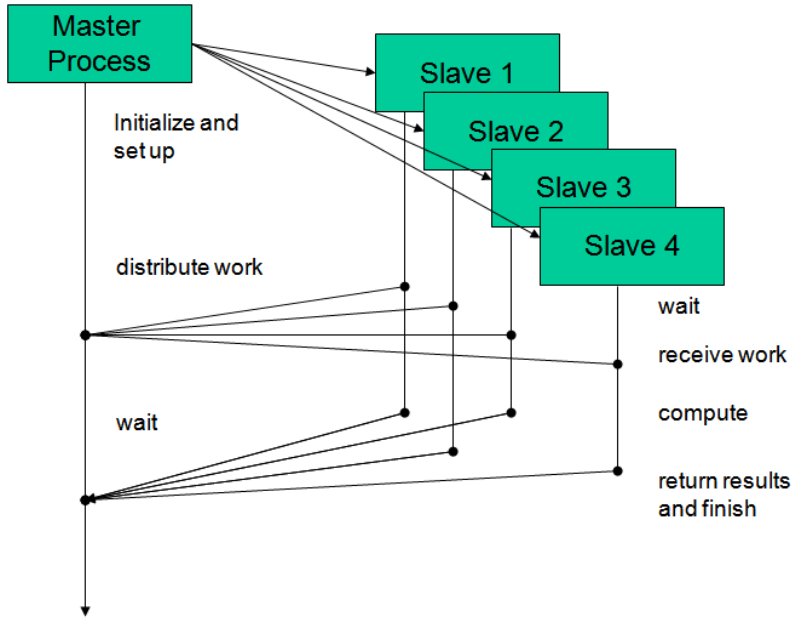

Fig. 5 – Work spawning

The proposed infrastructure consists in the following components:

– at the user nodes: the client codes and minimal facilities to access Grid infrastructure;
– at the repository nodes: the service codes and the catalog services that indexes Grid services and satellite data; a catalog of the virtual organization allows the access to

the institution catalogs; Grid middleware availability is mandatory;
– at the storage and computing nodes: the satellite data and image processing software; Grid middleware availability is optional.

In distributed image processing (and scientific computing in general), a great interest is dedicated to reuse of legacy code. In the recent past different technologies and methodologies have been developed to enable this process. The generally adopted solution is the legacy code encapsulation: the code is left in its environment and dynamically connected to the new technologies through a wrapper, allowing the software to perform in a client-server system. The server implements the most common image processing operations performed in a data parallel fashion. The parallelism is hidden from the users and is totally managed by the server, that also applies a transparent optimization policy.

For the testing purpose we used Globus Toolkit with its latest implementation of WSRF as Grid middleware, Java as programming language for the client and service codes, and also other software as image processing tool. It is important to mention batch mode software capability allowing to do image processing from the command line or acting as server component. The currently implemented client codes referring to the remote image processing facilities are described in what follows for PARDF algorithm (the used pattern is: command-name <parameters>):

createServInstance ***<FactoryRef>*** ***<EPRf>*** - A Grid service instance is created. An EPR (endpoint reference) is returned in the EPR file (EPRf). The service factory lying in a Globus container is referred as the first parameter. The Grid service will be activated on the repository node behind the same institution firewall as the image database that will be accessed. Using the Grid certificates and according to the VO policies the user will be mapped to a local user from that institution.

startPARDFServer ***<EPRf>*** ***<DatabaseRef>*** -Using the Grid service instance specified by the EPRfile the PARDF is started in the server mode on the storage-and-computing node specified by the second parameter. It is assumed that the Grid service code and the storage-and-computing node are currently behind the same firewall and PARDF can be started by simple commands like ssh. The communication between the Grid service instance and PARDF is done via sockets. The Grid service instance acts as client for the PARDF server. Different PARDF servers will be launched for different users.

clientPARDFAction ***<EPRf>*** ***<PARDFcommand>*** - The PARDF command is sent to the remote PARDF server. It contains a PARDF operation or script to be applied on specified remote image file(s). In the case of a script, this must lie on the remote computing node. The result can be one or more new image files saved on the remote node. Consecutive operations can be performed with references to the previous one since the service instance will remember the client and the PARDF server remains active until a specific PARDF command (e.g. quit) will stop the PARDF server and the Grid service.

clientGetFile ***<EPRf>*** ***<File>*** ***<DestDir>*** - The named file is transferred from the storage-andcomputing node to the destination directory from the user node. The file reference includes also the path. The file is copied with a simple command like “scp” to the repository node and from there using GridFTP to the user node.

5. SOME RESULTS

This section presents performance analysis of the parallel data fusion algorithm in our Grid environment. Several graphs are shown to depict speedups and efficiencies achieved for the parallel implementation. Different sizes of data sets are used. Timing results for several decomposition strategies are given. In order to show performance characteristics as a function of the problem size, multiple data sets from 256 x 256 (case A) to 2048 x 2048 (case E) were tested. Three common metrics that will be used to measure the parallel processor performance are:

- Elapsed time – τ
- Speedup – $s(p,N)=\dfrac{\tau(1,N)}{\tau(p,N)}$ (11)
- Efficiency – $\varepsilon(p,N)=\dfrac{s(p,N)}{p}$

A major factor in determining the measured performance levels is the fact that the fusion code was parallelized "as-is" without extensive optimization of either the FORTRAN compiler code or the algorithms employed. Both the serial and parallel fusion codes versions are written in straight FORTRAN code (matrix routines included), plus MPI calls for the parallel version. Neither uses library functions optimized for the AMD processors nor do they otherwise engage in cache management to speed the calculations.

In fact each of these considerations applies equally well to both serial and parallel code and the standard advice still remains true. Truly high performance begins with selection of a good serial algorithm, followed by optimization of it, followed then by parallelization. The initial serial data fusion code has been successfully parallelized through the use of standard MPI communication functions. Apart from debugging, the major components of effort were in the creation, declaration and definition of portable MPI derived data types to correspond to each of the complicated data structures present in the code. The result is a Parallel Fusion Tracker code written in standard FORTRAN and MPI that will be portable to other High Performance Computing architectures, such as the 64 node PC-based cluster at INCAS.

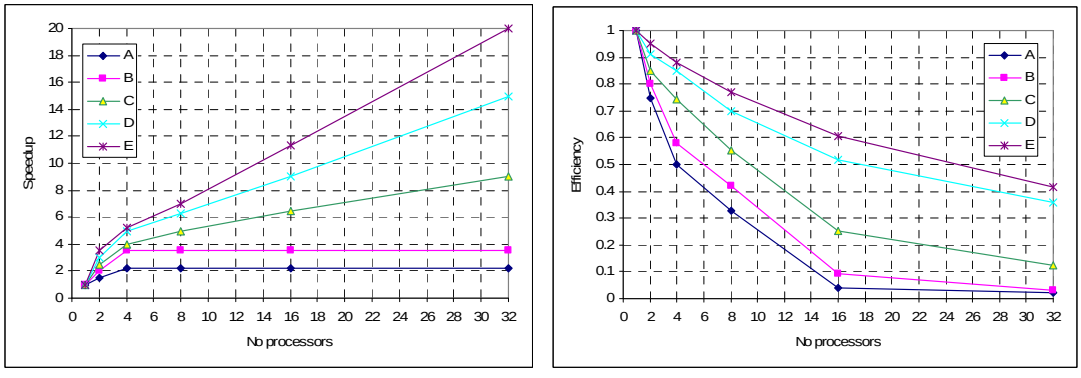

Fig. 6 – Performance charts

6. SOME CONCLUSIONS

The objective of this analysis was to demonstrate an implementation of multi-sensor data fusion and/or multi-target tracking functions within an integrated multi-node portable HPC architecture based on emerging Grid technology. The key metrics to be determined in support of ongoing system analyses includes: required computational throughput in MFLOPS; latency between receipt of input data and resulting outputs; and scalability, processor utilization and memory requirements. Furthermore, the standard MPI functions are considered to be used for inter-node communications in order to promote code portability across multiple HPC computer platforms, both in space and on-ground.

Centralized data fusion using a cluster-based parallel processing system is presented in this paper. The proposed method is useful in solving the high computational requirement in the fusion centre of the centralized architecture. Moreover it is useful if accurate tracking results are received. Using multiple sensors connected to the network, more coverage is possible using data fusion. The performance of the cluster-based centralized data fusion is presented in terms of relevant metrics.

REFERENCES

[1] C.L. Carvajal-Jimenez, W. Lugo-Beauchamp, W. Rivera. "Grid-HSI: using Grid computing to enable hyperspectral imaging analysis". M.H.Hamza (ed.), CIIT04, 583–588, 2004.
[2] M.A. Abidi, "Sensor Fusion: A new Approach and its Applications", Proc. SPIE Conf. on Sensor Fusion II: Human and Machine Strategies, pp. 235-246, 1989.
[3] C.J. Delcroix and M.A. Abidi, "Analytical Fusion of Edge Maps in Multisensory Images", Tech. Rep. ECE-1988-21, Univ. of Tenessee, Knoxville, TN, 1988
[4] M.R. Hestenes and E. Stiefel, "Methods of Conjugate Gradients for Solving Linear Systems", J. Res. N.B.S., vol **49**, p. 409, 1952
[5] C. Aykanat and F. Ozguner, "Large Grain Paralel Conjugate Gradient Algorithms on a Hypercube Multiprocessor", Proc. Of International Conf. on Parallel Processing, (S.K. Sahni ed.), pp. 641-644, The Pensilvania State University Press, University Park, 1987
[6] E. Polak, Computational Methods in Optimization: A unified Approach. New York, NY: Academic Press, 1971
[7] A. Clematis, D. D'Agostino, A. Galizia. An object interface for interoperability of image processing parallel library in a distributed environment, F. Roli, S. Vitulano (Eds.), ICIAP05, LNCS 3617:584-591, 2005.
[8] A. Galizia: "Evaluation of optimization policies in the implementation of Parallel Libraries", Technical Report IMATICNR-Ge 20, 2004.

Acknowledgement

This work has been initiated in GisHEO - PECS/2008 project and will continue in other ESA related activities.