International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 2 Issue 8, August - 2013

Investigation of Rocks cluster software for the parallelization of Geant4-based LINAC Application

J. EL Bakkali*[1], T. El Bardouni[1], H. Boukhal[1]

[1]ERSN-LMR, Faculty of Sciences, University Abdelmalek Essaadi, Tetouan, Morocco

Abstract

We discuss in this paper our experience in the parallelization of MC Geant4 code using Rocks cluster software, Geant4.9.4 and Geant4 MPI Interface to modeling a typical Linac head called Saturne 43. We get some problems with shared libraries as soon as we try to run our Geant4 simulation in parallel mode. We corrected these bugs by doing some instructions to make our simulation work properly. Our system consists of a homogeneous cluster of 8 CPU connected via an Ethernet network. The outlines of our simulation called ParaSaturne43Writer are a collection of eight phase-space files in IAEA format, each file produced by a specific slave machine and contains data describing particles those reaching the scoring plane located below jaws components. In the aim of auto combining this data and without user intervention, we made an utility called Geant4PhspMerger that can be used to detect automatically all phase-space files located in the working directory and finally merge them. We remarked a nearly linear speedup in running time as the number of slave machines increases; in the fact we can reduce the overall time for writing phase-space file by a factor nearly equal to the number of slave machines.

1. Introduction

Geant4 toolkit [1] is a simulation toolkit for the simulation of the passage of particles through matter. Its areas of application include medical, space sciences, high energy and accelerator physics. The main players in its development are in the discipline of high-energy physics, combining the efforts of more than 100 workers from facilities such as CERN in Europe, KEK in Japan and SLAC in the US. The Geant4 code calculates a physical evolution of each particle step-by-step by Monte-Carlo method. Geant4 has components to model the geometry, the involved materials, the fundamental particles of interest, the generation of primary particles for new events, the tracking of particles through materials and external electromagnetic fields. It treats the physics processes governing particle interactions, the response of sensitive detector components, the generation of event data, the storage of events and tracks, and the visualization of the detector and particle trajectories. The truth is; Geant4 is indeed very powerful, but also very complex. The learning curve is both steep and long. A superficial knowledge of C++ is insufficient to optimally use the toolkit. User shouldn't expect to be up and running in a few days; just the installation process can be slow and leads to strings of incomprehensible errors that only experts can understand. Geant4 simulations are painfully slow. It is acknowledged that efforts are being made to improve calculation speed by the developers, but currently it can take up to thousands of hours on your fastest computer to accurately simulate problems such as patient-dose calculations in radiotherapy. So a cluster of CPUs can be useful and a good solution to enhance the calculation speed.

A cluster is a type of parallel or distributed processing system, which consists of a collection of interconnected stand-alone computers working in collaboration as a single integrated computing resource. A computer node can be a single or multiprocessor system (PCs, workstations, or SMPs) with memory, I/O facilities, and an operating system. A cluster generally refers to two or more computers nodes connected together. The nodes can exist in a single cabinet or be physically separated and connected via a LAN. The cluster nodes can work collectively as an integrated computing resource, or they can operate as individual computers. The cluster middleware is responsible for offering an illusion of unified system image. The network interface hardware acts as a communication processor and is responsible for transmitting and receiving packets of data between cluster nodes via a network.

The task-farming paradigm consists of two entities: master and multiple slaves. The master is responsible of decomposing the problem into small tasks and distributes these tasks among a farm of slave processes, as well as gathering the partial responses in order to produce the final result of the computation. The slave processes execute in a very simple cycle: get a message with the task, process the task, and send the result to the master. Usually, the communication takes place only between the master and the slaves.

Building cluster is straightforward but managing its software cannot be easy. Rocks cluster [2] is software built on top RedHat Linux releases that supports all the hardware components that RedHat supports. In this work we used Rocks Cluster package to build and maintain our cluster. This software provides mechanisms to control the complexity of the cluster installation and the expansion process.

2. Materials and methods

2.1. Physical setup of our cluster

The physical setup of our Cluster contains the following node types:

- **Master**

PC with 2.4 GHZ CPU is where users log in, submit jobs, compile code, etc. This node can also act as a router for other cluster nodes by using network address translation (NAT). The Master node has two Ethernet interfaces one is public called eth1 and the other is private called eth0 used for private network.

- **Compute**

Four dual core PCs with 2 .4 GHZ CPU are the workhorse nodes. These nodes are not seen on the public network.

- **Ethernet Network**

All compute nodes are connected with Ethernet on the private network. This network is used for administration, monitoring, and basic file sharing.

Our cluster architecture dictates these nodes types are connected as shown in **Figure 1**.

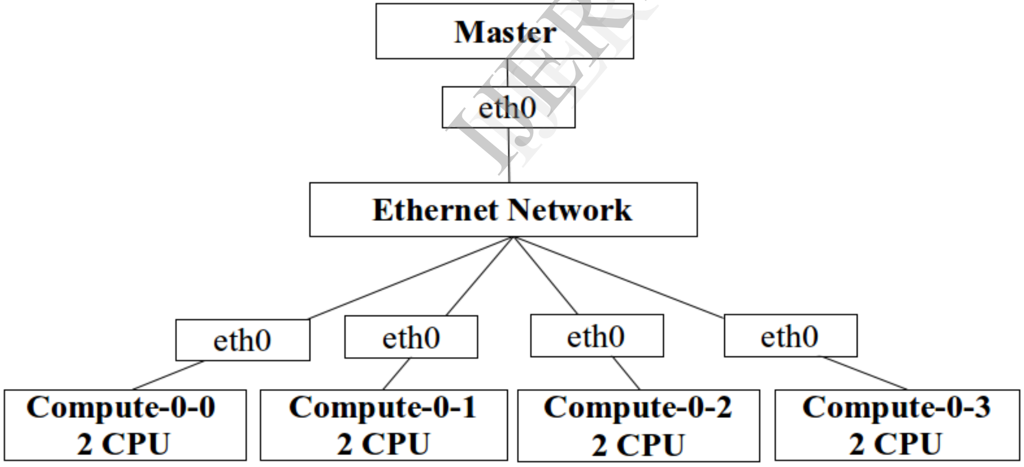


Fig.1 illustrating our cluster Architecture

Running simulation on many processors to reduce the overall time is the best solution when doing a large Geant4 simulation. Thus it is essential to parallelize Geant4-based application to reduce the running time then shorten the cycle of experiments. There are several methods of successful use of Parallel Computing in Geant4 such as ExDiane, ParGeant4 and Geant4 MPI interface. In the next section we discuss the parallelization of Geant4 using Geant4 MPI interface.

2.2. Parallelized application with Geant4 MPI Interface

MPI is a Message Passing Interface Standard. It was the first attempt to create a “standard by consensus” for message passing libraries. MPI is available on a wide variety of platforms, ranging from massively parallel systems to networks of workstations. The main design goals for MPI were to establish a practical, portable, efficient, and flexible standard for message-passing. MPI is a language-independent communications protocol used to program parallel computers.

Message passing libraries allow efficient parallel programs to be written for distributed memory systems. These libraries provide routines to initiate and configure the messaging environment as well as sending and receiving packets of data. Currently, the two most popular high-level message-passing systems for scientific and engineering application are the PVM (Parallel Virtual Machine) and MPI (Message Passing Interface). Message-passing systems are still stigmatized as low-level because most tasks of the parallelization are still left to the application programmer. When writing parallel applications using message passing, the programmer still has to develop a significant amount of software to manage some of the tasks of the parallelization, such as: the communication and synchronization between processes, partitioning and distributing data, mapping of processes onto processors, and input output of data structures.

Geant4 MPI Interface is a native interface with MPI libraries. Within this interface, Geant4 simulation can be parallelized with different MPI compliant libraries, such as LAM/MPI, OpenMPI and MPICH2. The last one is adopted in our cluster.

2.3. Instructions to make Geant4 working under Rocks clustering software

The Rocks cluster installation is simple; we installed Rocks cluster software in all nodes (Master and four compute nodes) by following the installation guide of Rocks 5.4 distribution from the Rocks web site [2]. After that we compiled and installed geant4.9.4 in the Master node, the Rocks cluster software permits all the compute nodes to obtain the same image of geant4 code and thus the same simulation program. We got some errors as soon as we attempted to compile our program in parallel mode:

- The first one indicates the absence of MPI headers files; in order to get our simulation working we corrected this problem by copying the needed headers files from mpich2 package to /usr/include:

```
cp -r /opt/mpich2/gnu/include /usr/include
```

Then we copied the hosts file from master node to all compute nodes, by doing the following instructions:

```
scp-r /etc/hosts compute-0-0:/etc/hosts
scp-r /etc/hosts compute-0-1:/etc/hosts
scp-r /etc/hosts compute-0-2:/etc/hosts
scp-r /etc/hosts compute-0-3:/etc/hosts
```

After that we just need to type this command:

```
mpdboot -n 5
```

- The second one occurred when we compiled our program: we get an error in loading shared library libGLU.so.1. This problem was resolved by copying this library from master node to all compute nodes :

```
scp /usr/lib/libGLU .so.1 compute-0-0:/usr/lib
scp /usr/lib/libGLU .so.1 compute-0-1:/usr/lib
scp /usr/lib/libGLU .so.1 compute-0-2:/usr/lib
scp /usr/lib/libGLU .so.1 compute-0-3:/usr/lib
```

For the execution of our program we make a shell script that contains this code:

```
mpiexec  -machinefile file -n 8  Parasaturne43writer writer.mac
```

Where file contains the following nodes:

```
compute-0-0:2
compute-0-1:2
compute-0-2:2
compute-0-3:2
```

2.4. A simplest solution for auto-combining phase-space data files

Geant4.9.4 does not include the capability to read or write phase space files. These files are useful for dividing the simulation into parts. In other words the state of the simulation can be preserved at various planes in the simulation geometry in files known as phase-space files. The information of particles reaching a scoring plane can be stored in a phase-space file. Each entry in the file contains data such as the charge, energy, position, direction, and weight associated with a particle. The phase- space data can be used to generate new particle histories, or to obtain quantities such as the planar fluence, angular distribution and mean energy of the particles crossing the scoring plane.

The use of the phase-space is a technique to reduce the computing time without affecting the computing accuracy. The phase-space approach is based on the idea to divide the computation into two steps. During the first step, the data of all the particles hitting the scoring plane are recorded in some files from which they are recalled when the second step of the computation is performed. This approach reduces considerably the computing time because the same data can be used in different simulations. When the corresponding information of particles reaching the scoring plane is recorded, these routines create two files with extensions .IAEAphsp and .IAEAheader, respectively. The first one is a binary file in which the previously calculated data of all the particles passing through the defined plane are stored. It contains information about energy E, statistical weight w, the three components of the position (x, y, z) and the direction cosines (u, v, w). Presently, the IAEA phsp format supports the storage of the following particles: photons, electrons, positrons, neutrons and protons. In addition, in the .IAEAheader, which is an ASCII file, the user can find information related to the phase-space data stored in the file, such as a statistical summary, the number of original histories and the number of particles of each kind that crossed the phase-space plane. The default file name would be "PSF" followed by underscore and the value of coordinate z for that scoring plane with corresponding extensions. This technique can save a lot of time in cases where several calculations share the some accelerator head parts and in the case where the accelerator simulation is very slow compared to the dose calculation.

Auto-merging phase-space files is needed when such simulation is running in parallel mode and gives as outlines more than one phase-space file. We propose in this work an utility that answers to this demand. The functionality of this tool called Geant4PhspMerger is carefully explained in the following section. Geant4PhspMerger is an utility written in C++ language. It can be used to merge phase-space files collected from different slave machines in order to make a single phase-space file named phsp (IAEAphsp, IAEAheader). Geant4PhspMerger detects automatically all phase-space files located in the current directory and merge them in a single phase-space file. When the auto-merging operation is done this utility removes all phase-space files.

Geant4PhspMerger provides the following features:

- Shows the number and the names of phase-space files to be combined.
- Combine automatically all phase-space files presents in the working directory.
- Shows elapsed time occupied by combination phase.

For understanding how to compile and install Geant4PhspMerger on Linux machines, user should follow these steps:

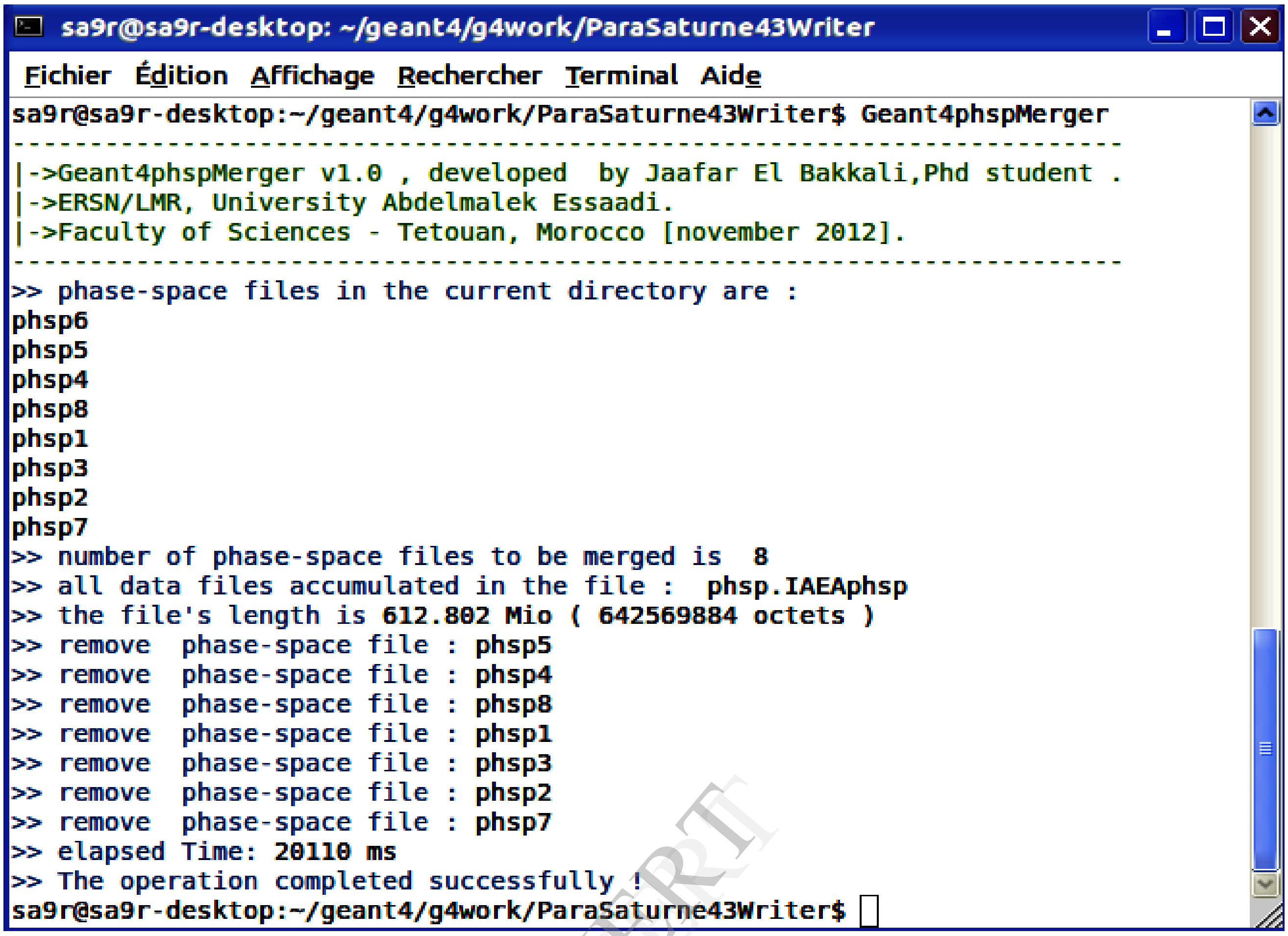
1 - Download the IAEA routines from its web site [3].

2 - Create a new directory “Geant4PhspMerger-install” and copy into it the following files: iaea_header (.cc && .hh), iaea_phsp (.cc && .hh), iaea_record (.cc && .hh) and utilities (.cc && .hh).

3 - Add the two files Geant4PhspMerger.cc and install.sh into Geant4PhspMerger-install directory.

4 - Run the bash script ‘install.sh’, thus the compilation and installation will be done automatically. To call this utility one should open new terminal, change directory to the application one, and type the following command:

```
Geant4PhspMerger
```


sa9r@sa9r-desktop: ~/geant4/g4work/ParaSaturne43Writer
Fichier Édition Affichage Rechercher Terminal Aide

```
sa9r@sa9r-desktop:~/geant4/g4work/ParaSaturne43Writer$ Geant4phspMerger
-----------------------------------------------------------------------
|->Geant4phspMerger v1.0 , developed  by Jaafar El Bakkali,Phd student .
|->ERSN/LMR, University Abdelmalek Essaadi.
|->Faculty of Sciences - Tetouan, Morocco [november 2012].
-----------------------------------------------------------------------
>> phase-space files in the current directory are :
phsp6
phsp5
phsp4
phsp8
phsp1
phsp3
phsp2
phsp7
>> number of phase-space files to be merged is  8
>> all data files accumulated in the file :  phsp.IAEAphsp
>> the file's length is 612.802 Mio ( 642569884 octets )
>> remove  phase-space file : phsp5
>> remove  phase-space file : phsp4
>> remove  phase-space file : phsp8
>> remove  phase-space file : phsp1
>> remove  phase-space file : phsp3
>> remove  phase-space file : phsp2
>> remove  phase-space file : phsp7
>> elapsed Time: 20110 ms
>> The operation completed successfully !
sa9r@sa9r-desktop:~/geant4/g4work/ParaSaturne43Writer$
```

Fig.2 Screensaver from Geant4PhspMerger utility execution

The source and binary files of Geant4PhspMerger could be downloaded from our web site [4]. User should probably not have to care about compilation; he can just copy the binary file to /usr/lib folder. Geant4PhspMerger has been tested under Centos and Ubuntu distributions.

2.5. Simulation of linac head

In this work we build our own simulation program called ParaSaturn43Writer to simulate a 12MV photon beam delivered by a typical linear accelerator Saturne 43 that is widely used in radiation therapy, in particular IMRT (Intensity Modulated Radiation Therapy). The term IMRT refers to a particular radiation therapy whose principle is to treat a patient from a number of different directions with beams of non uniform fluences, which have been optimized to deliver a high dose to the target volume and acceptably low dose to the surrounding healthy tissues.

Our code was used to simulate the Saturne 43 LINAC treatment head and generates a phase-space file at a plane situated at 50 cm from the W target. In the case of the square field 10 x 10 cm^2. Our program ParaSaturne43Writer was called to write IAEA formatted phase-space files (PSF) by using an interface to these files. This interface is composed of the following files:

- The IAEA routines published on its web site [3].
- The files defining the Geant4 writer class: G4IAEAphspWriter (.hh and .cc).

In this study, we simulated a linac head dedicated to generate 12 MV photon beam. The components of the linear accelerator head are shown in **Figure 3** with HepRepp visualization system. These components include Titanium window, W target, primary collimator, flattening filter, ionization chamber and secondary collimator jaws. In **Figure 4**, using RayTracer visualization system, we show our generated 3D view of the modeled linac head associated to a water phantom with dimension of 40 × 40 × 40 cm^3 that was placed at a source to surface distance (SSD) of 90 cm. The upper and lower jaws were set to create a field size of 10 × 10 cm^2 on the phantom surface. The depth in water is expressed from the external side of the entrance window of the phantom. Thus, a measurement of 10 cm depth means 4 mm of PMMA plus 9.6 cm of water.

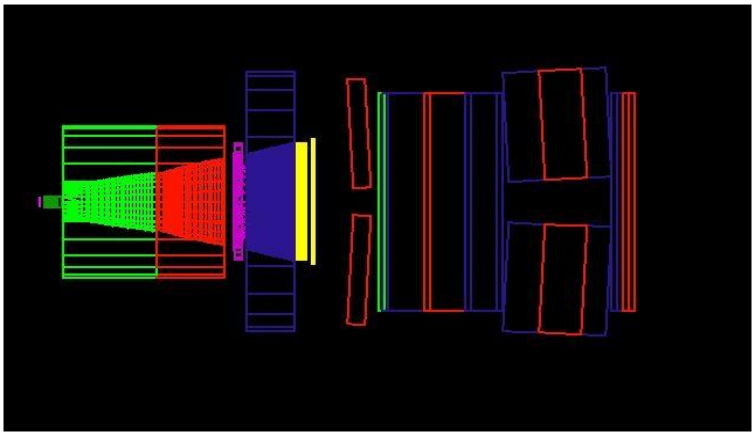


Fig.3 Illustrating Saturne 43 head Geometry using HeppRep

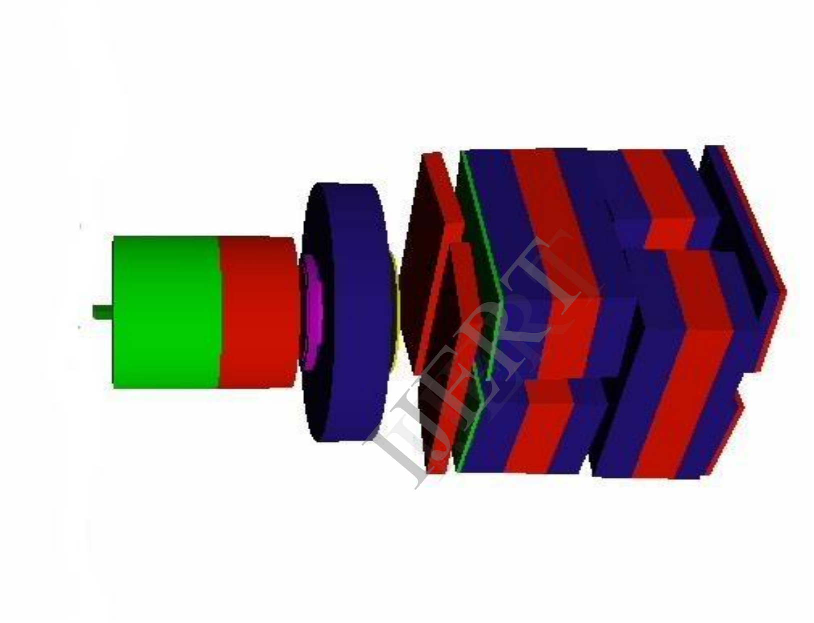


Fig.4 Illustrating Saturne 43 head Geometry using RayTracer

3. Results and discussion

3.1. Cluster linear speedup checking

In the aim to test the efficiency of parallelization of our application using rocks cluster software, we run our code on different number of workers. The **Table 1** describes the evolution of the Average CPUs time per number of CPUs:

Number of CPUs	1	2	4	8
Number of histories	10^4	2.10^4	4.10^4	8.10^4
Average CPU Time (s)	9.09	17.31	35.99	70.12
Average CPU Time / Number of CPUs	9.09	8.655	8.9975	8.765

Table.1 Average CPUs time per number of CPUs for scoring phase-space data at 50 cm from target for 12MV clinical photon beams.

As a conclusion we can say that the nearly linear speedup is achieved when the number of slave machines increases; in fact we can reduce the overall time for writing phase-space file by a factor nearly equal to the number of slave machines.

3.2. Comparison between simulated data and measured ones

With rocks cluster approach, we have running multiple simulations dedicated to model the LINAC treated head. Instead of simulate 24 million of histories in single CPU that can take about two days to deliver a phase-space file; we can reduce the overall time to six hours when running eight simulations at same time. Thus, Tree million histories per node has been considered and the split number associated to bremsstrahlung photons was set to 60. Each simulation produces at the end of each run a phase-space file with IAEA format. The Geant4PHspMerger utility take action to auto-combining all phase-space files in single one. Then this file was considered to be a 12 MV photon generator in a program called ParaSaturn43Reader dedicates to simulate dose distributions in water phantom.

In order to validate the results of our MC simulation, the depth doses and profile curves for a homogeneous water phantom of 40 × 40 × 40 cm^3 were compared to experimental ones obtained at the French National Metrological Laboratory for ionizing radiation (LNHB) [5].

The results shows that the simulated data agreed well with measured data, except data points who located in the penumbra region, where the dose profile has a high gradient, after all, 91.1% of the calculated data points seems agree with experience within 1.5%. The percent difference in this region was about 6%. The ambiguities may possibly come from inaccuracies in the simulation geometry, the approximation of the initial source configuration or uncertainties in the measured data. The **Figure 5** shows the cross beam profiles for measured and calculated data points.

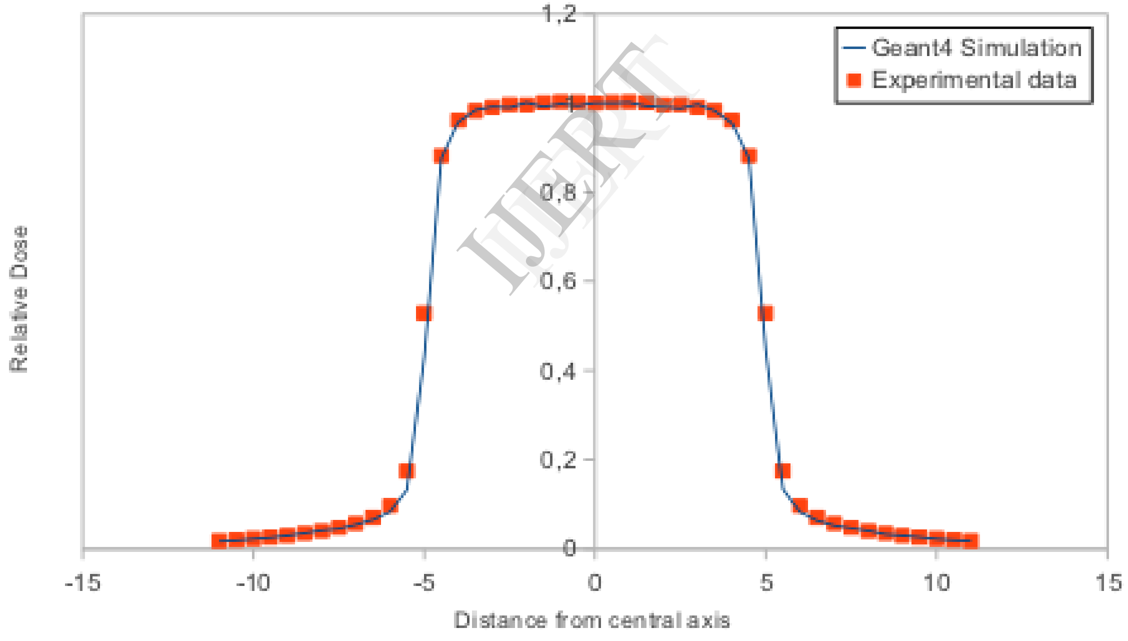


Fig.5 Comparison of calculated and experimental relative depth dose due to 12 MV photon beam in homogeneous water phantom, for a 10 × 10 cm2 field size. Results are normalized to the dose at the depth of 10 cm.

For the depth dose curve, it's seems that 97.6% of the calculated data points agree within 1.5% with the experimental measurements for depth 10 cm, so except the first data point all others ones were accepted, The ambiguities may possibly caused by inaccuracies in the approximation of the initial source configuration or uncertainties in the measured data. The **Figure 6** shows the depth doses curves for measured and calculated data points.

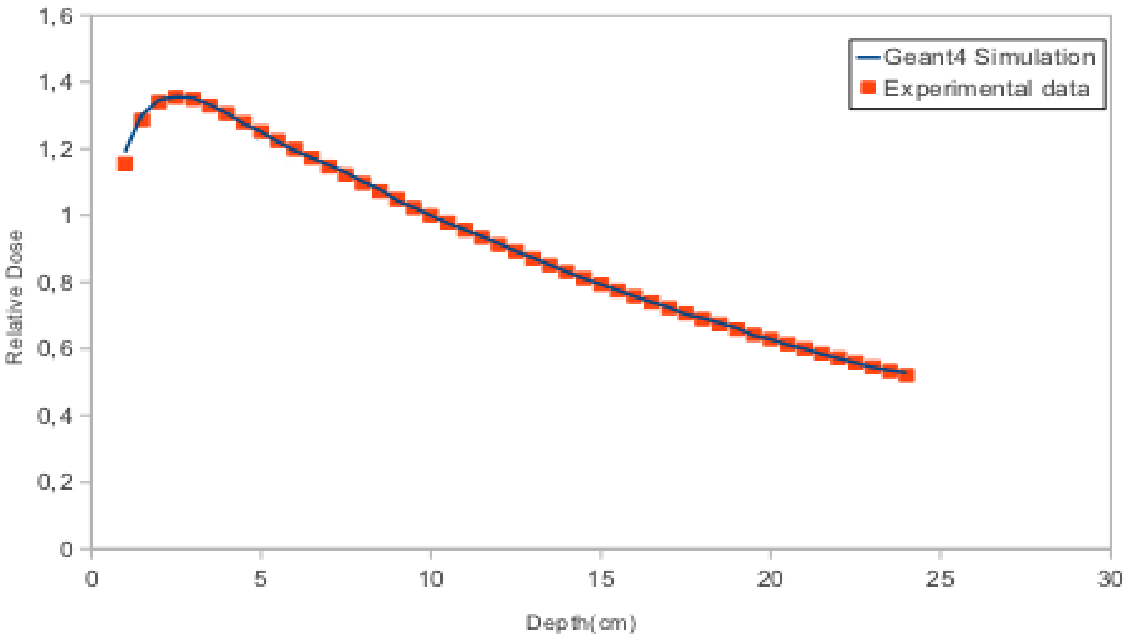


Fig. 6
Comparison of calculated and measured dose profile at the depth of 10 cm due to 12 MV photon beam in homogeneous water phantom, for a 10 × 10 cm2 field size.

4. Conclusion

From this paper it is pointed out that the parallel computing is useful when a Geant4-based applications demand a lot of time for making desired job. In our case we found that the parallelization of the simulation using Rocks cluster software and Geant4 MPI interface could make our simulation faster. Thus, within the parallelization we will be able to divide a large simulations into others shorts and to combine their output in a single file using our solution called Geant4PhspMerger. A comparison between simulated data and measured ones using gamma criterion, shows that is possible to use Monte Carlo Geant4 code to validate such LINAC used in radiotherapy with accuracy with 1.5%.

5. References

[1]_ J. Allison et al., IEEE TNS 53 (2006) 270. [Online] Available: http://geant4.cern.ch

[2] « www.rocksclusters.org | Rocks Website ». [Online] Available: http://www.rocksclusters.org/

[3] R. Capote and I. Kawrakow, « Read/write routines implementing the IAEA phsp format », version of December 2009. [Online] Available: http://www.nds.iaea.org/phsp/software/iaea phsp Dec2009.zip

[4] J.EL Bakkali, « Geant4PhspMerger , a c++ tool for auto combing Phsp files » . [Online] Available: http://sourceforge.net/projects/geant4phspmerge

[5] L. Blazy, D. Baltes, J. M. Bordy, D. Cutarella, F. Delaunay and J. Gouriou, "Comparison of PENELOPE Monte Carlo Dose Calculations with Fricke Dosimeter and Ionization Chamber Measurements in Inhomogeneous Phantoms (18 MeV Electron and 12 MV Photon Beams)," Physics in Medicine and Biology, Vol. 51, No. 22, 2006, pp. 5951-5965.