Benchmark of medical dosimetry simulation using the Grid

Stéphane Chauvie, Patricia Mendez Lorenzo, Anton Lechner, Jakub Moscicki and Maria Grazia Pia

Abstract– **The practical capability of using Grid resources to perform high-precision dosimetry simulation in radiation oncology is evaluated, taking into account the peculiar demands of different treatment modalities. For this purpose extensive benchmark tests on the LHC Computing Grid (LCG) are performed, involving the calculation of dose distributions as required for clinical practice. The software architecture of the test is based on Geant4 for simulation, on AIDA for data analysis, on the LCG middleware for grid computing and on DIANE as an intermediate layer between the application software and the computing environment.**

***Index Terms*—Monte Carlo, Geant4, parallel computing, grid, dosimetry.**

I. INTRODUCTION

MONTE Carlo simulation is widely used in many experimental physics domains to study the effects of radiation.

Simulation plays a fundamental role in detector design, in the evaluation of the physics reach of experiments, in the development and optimization of data reconstruction algorithms and data analysis methods. Monte Carlo simulation is also used for precision studies in oncological radiotherapy and nuclear medicine, and for mission critical studies in space science.

Monte Carlo codes allow the evaluation of radiation effects with great accuracy; however, the precision of the simulation is usually achieved at the price of a long computational time. Lengthy execution makes Monte Carlo simulation unsuitable to some experimental applications that would greatly profit of its precision, but would require a quick calculation response: a typical use case is the clinical exploitation in oncological radiotherapy, but a fast response is often desirable in engineering applications too, like the optimization of detector or electronics design parameters.

Various solutions have been developed in the past decades to speed up Monte Carlo simulations, such as variance reduction techniques, inverse Monte Carlo methods, and the parameterization of detector response to particle exposure (also known in high energy physics experiments as "fast simulation"). These methods allow reducing the execution time of a simulation; nevertheless, they affect the precision of the results by introducing approximations in processing the physical interactions which particles undergo in the experimental set-up.

Parallel execution has also been widely explored as a suitable technique for performing a Monte Carlo simulation in a shorter time. This solution does not affect the accuracy of the results; nevertheless, it is the source of other issues, like the availability of extensive computing resources and the need of manipulating the Monte Carlo kernel or the user application code for execution in parallel mode.

The emphasis on component-based architecture due to the relatively recent spread of the object oriented paradigm in particle physics experiments, the wide availability of computing resources at affordable price, and new emerging concepts in parallel computing, like the grid technology [1], suggest to revisit the problem of parallel Monte Carlo simulation under a novel perspective.

This paper presents the application of grid technology to the Monte Carlo simulation in dosimetry use cases relevant to oncological radiotherapy. Benchmarks for execution in a grid system are documented in set of test cases.

II. TEST CASES

The simulations in the grid environment were carried out using three test case applications publicly distributed with the Geant4 [2]-[3] toolkit in its advanced example package. In this chapter we will describe the requirements in terms of dose accuracy of the different radiotherapy technique compared to the overall simulation time. In particular we will highlight, for each test case, the computational endeavor relatively to the nature of radiation source, the physics processes involved and the complexity of the geometry.

Table I shows a summary of the execution time required to provide statistically significant results performing the three simulations in sequential on an average PC. In the application context the significance of the results is determined by the requirement that the statistical fluctuation in a volume be comparable to the error commonly acceptable in radiotherapy dose evaluation (3% point dose difference and 3 mm dose to agreement distance)[3].

Manuscript received November 14, 2007.

S. Chauvie is with Santa Croce e Carle Hospital, Via Coppino 26, I-12100, Cuneo, - Italy (phone: +39 0171 641558, fax: +39 0171 641554, e-mail: chauvie@to.infn.it)

A. Lechner is with Atomic Institute of the Austrian Universities, Vienna, Univ. of Technology, Vienna, Austria (email: Anton.Lechner@cern.ch).

P. Mendez Lorenzo and J. Moscicki are with CERN, CH-1211 Geneva 23, Switzerland(e-mail: Patricia.Mendez.Lorenzo@cern.ch and Jakub.Moscicki@cern.ch).

M. G. Pia is with INFN Sezione di Genova, Via Dodecaneso 33, I-16146 Genova - Italy (phone: +39 010 3536328, fax: +39 010 313358, e-mail: MariaGrazia.Pia@ge.infn.it)

TABLE I
OVERALL SIMULATION TIME IN SEQUENTIAL MODE

Test case	Execution time (CPU hours)
Brachytherapy	7
Hadrontherapy	16(calibration)-150(full)
Medical Linear Accelerator	>240

A. Brachytherapy

Brachytherapy is a radiotherapy technique that consists of inserting several sealed radioactive sources directly inside or in close contact with the tumor. The mostly used ones are Ir^{192}, I^{125} and Pd^{103}. Consequently the energy deposition will be concentrated in the proximity of the source and hence primarily in the tumor. Treatment planning systems are employed in brachytherapy for optimizing the position of the different sources and computing the time the source must be kept in the location thus determined to deliver the prescribed dose. Nevertheless, for the effectiveness of the treatment it is also essential to calculate the dose to the surrounding healthy tissue. Traditionally, treatment planning systems calculated the dose deposition inside the patient assuming that the zone were the dose is delivered was homogeneous. This approximation is not realistic if the tumor is located near bones, lungs or cavities. In this case Monte Carlo methods could provide an improved estimation of the dose deposition in a realistic model of the real patient.

The Geant4 *brachytherapy* Advanced Example [4] defines different sources in terms of geometry structure, material composition and energy spectrum of the photons emitted by the source. The simulation calculates the dose delivered to a voxel phantom, which can be represented in terms of isodose curves.

The time to achieve statistically significant results as seen in Table 1 is relatively low. The Monte Carlo simulation is not too computationally demanding due to the quite simple geometry of the source and the small target volume, with heterogeneities displaced out of the volume of irradiation. Therefore this example is suitable as a test case for fast radiotherapy simulation. A reasonable expectation would be performing the simulation in a time compatible with the clinical environment, e.g. to compare analytical dose distributions with Monte Carlo ones in less than an hour.

B. Hadrontherapy

Recent progresses in the technology of particle acceleration are pulling cyclotrons and synchrotrons, previously confined to research centers, into the clinical practice. Several general hospitals, spread around the world, are nowadays equipped with hadrontherapy facilities and use particles like protons and ions to treat radioresistant and inoperable tumors. Heavy charged particles offer various advantages with respect to conventional external γ-ray therapy: there is a good ratio between entrance and Bragg peak dose, one can bend particle in whatever axial position with active and passive scanning systems and one could optimize the depth position of Bragg peak in tissue varying the energy of the incoming particles. Moreover heavy charged particles, such as protons and carbon ions, have demonstrated a higher radiobiological efficiency, which means a higher potential of destroying tumor cells with respect to photons. The Geant4 *hadrontherapy* Advanced Example [6] gives the possibility to simulate a typical hadrontherapy treatment beam line (including all its elements) and to calculate the proton dose distribution curves. The example provides the user the possibility to design and optimize the transport beam line, check the dose distributions and, finally, to verify analytically based treatment planning systems.

While one could envisage as final goal of the simulation the comparison of analytical and Monte Carlo dose distributions inside real patient, the preclinical phase brings up two different scenarios. When Monte Carlo is used to calibrate the beam line, one needs merely to find the depth position of the Bragg peak; this could be achieved with relatively low statistics and an overall simulation time of the order of 16 CPU hours. A higher statistics, requiring about 150 CPU hours, is otherwise desirable is if one needs to perform a full simulation, e.g. for optimizing or designing the beam line or calculating dose deposition inside patients.

C. Medical Linear Accelerator

Intensity Modulated Radiation Therapy (IMRT) is a radiation therapy whose principle is to treat a patient with small beams of non-fixed fluence. IMRT represented a breakthrough in last ten years of conformal radiotherapy, giving the clinicians the outstanding possibility to deliver a higher dose to the target volume and acceptably lower dose to the surrounding healthy tissues. The Geant4 *medical linac* Advanced Example [7] models all the components of a commercial linear accelerator: the point source of electrons (and its distribution in energy and fluence), the primary collimator, the target, the vacuum window, the flattening filter, the ion chamber, the mirror, the jaws and the light field reticle. The result of the medical linac simulation is the dose distribution in a phantom filled with water.

This example is the more demanding in terms of computational power (see Table I). The complete simulation of the accelerator head requires a lot of time before the emerging photons interact with the phantom. The simulation application does not use the "phase space data" approach to avoid massive data transfers trough the network and slow I/O on disk. In future, to spare time, one could further exploit the "cuts per region" Geant4 functionality to avoid time consuming physics interactions description in the collimation components. Moreover to achieve a high accuracy in the dose calculation the energy deposition ought to be calculated in small volumes, of size comparable to the computed tomography voxels used in the geometrical parameterization of the patient.

III. Software Environment

This paper exploits an original system design for the execution of Geant4 based simulation in a distributed computing environment. The solution proposed is of general applicability and is based on freely available, open source software tools: therefore, it is suitable to a variety of Geant4 applications that could profit of parallel execution, not necessarily limited to the test cases discussed in this paper.

A. Architectural design

The idea of Geant4 parallelization used in this work as already been explained elsewhere [8]; a brief summary is included here.

The solution presented addresses the problem in a general way, by providing a design solution suitable to any Geant4 application and different distributed environments: both the execution in a conventional local computer farm or in a geographically spread grid environment.

The simulation application uses three software systems: the Geant4 toolkit, an AIDA [9] compliant analysis system, and the DIANE [10] distributed analysis environment based on the Master-Worker architectural pattern.

B. Geant4-based applications in the DIANE environment

The distributed execution of a simulation requires splitting the processing into smaller chunks. Monte Carlo simulation exhibits a natural subdivision into independent units at the level of events, which is sufficient to address the user requirements of execution speed in most use cases. The main technical requirements in this case involve the management of random number generation to preserve the reproducibility of the simulation and of external files which may be needed for the simulation execution.

A generic Geant4 application design has been conceived, which is suitable to the transparent execution of the simulation either in sequential mode or in distributed mode via DIANE. The *IG4Simulation* abstract class is responsible of steering the simulation execution in the DIANE framework. Its member functions *initialize, executeMacro* and *finish* encapsulate the essential actions needed for the simulation execution. The *setSeed* member function allows the simulation user to define the initial seed for the generation of random numbers throughout the simulation. A concrete class derived from *IG4Simulation* must be provided by the simulation developer. Its *initialize* implementation instantiates the classes required to interact with the Geant4 kernel. User configuration actions are encapsulated in a macro file, which is supplied to the job through the *executeMacro* member function. A macro file may contain directions for the configuration of the Geant4 simulation and of its execution via DIANE. The interaction with DIANE requires the specification of the total number of simulation events to be generated, the number of tasks into which the job is to be divided, and the initial seed for random number generation. The generation of random numbers in the tasks is handled transparently to the user; the specification of the initial seed is sufficient to ensure the reproducibility of a simulation.

Analysis objects, such as histograms or tuples, may be generated in the course of a simulation execution; they must be available to the user in a persistent storage resulting from the execution of the whole simulation. In a parallel execution mode each task would produce the analysis objects corresponding to its own portion of the simulation; the final analysis objects are assembled by the Master at the end of the execution, resulting from the sum of the partial ones. The user can retrieve both the final and the partial analysis objects.

The overhead introduced at the initialization stage by running a Geant4 application through DIANE was estimated by comparing the execution of one event on a single machine in sequential mode as a standalone application and via DIANE. This test was performed on an Intel Pentium IV PC, equipped with a 3 GHz processor and 1 GB RAM.

TABLE II
OVERHEAD TIME INTRODUCED BY DIANE

Standalone application	4.6 ± 0.2 s
Application via DIANE, simulation only	8.8 ± 0.8 s
Application via DIANE, with analysis integration	9.5 ± 0.5 s

The time elapsed in the execution includes the time spent in the initialization phase and the time to process the single event; since this is negligible (<1 ms) in comparison to the time needed for initialization, the measurement can be ascribed to the initialization phase entirely with a safe approximation. In the case of execution through DIANE, the effects of managing the simulation execution and of integrating the analysis results at the end of the simulation were evaluated. These measurements were performed by running the application in identical conditions several times. The results are summarized in Table II

IV. Test environment

The choice of the grid sites where to perform the simulation must be carried out carefully. The complexity of grid computing does not permit, at now, to “plug and play” users’ applications; specific expertise and know-how is required to set up the execution environment.

The first step consists of the identification of the resources of the grid to be used for the simulation. The user submitting the job must be a member of a virtual organization (VO) recognized by the grid system to be used. The simulations presented in this paper were executed as member of the Geant4 VO in the LCG grid. The selection of the resources to use is done prior to the job submission depending on their availability and the priority granted to Geant4 VO. Automated software tools exist for this purpose, but a certain degree of manual intervention is necessary in the course of a large scale production. GANGA was utilized to set-up a list of computing element and to select them according to their state (production, draining mode). Other elements to be considered

are the number of CPU a VO is allowed to use at a given site, the estimated waiting time in a batch queue and the estimated workload on a given node due to other concurrent jobs.

A constraint in the selection of suitable grid resources for execution is that they ought to have an operating system and architecture compatible with Geant4 supported platforms.

V. Results

The benchmarks described in the following sections document the performance of the system for distributed simulation in the three use cases selected; they present results in response to the requirements of the medical physics community for fast and precise Monte Carlo simulation, suitable to the clinical practice.

A. Splitting into tasks

The total number of events to be simulated depends on the desired statistical accuracy of the simulation. The task splitting for parallel execution should balance the task size (hence the task execution time) and the number of workers to be allocated. A higher granularity of tasks allows more effective use of the CPU time, with the drawback of dealing with an overall output of larger size resulting from the multiple simulation streams.

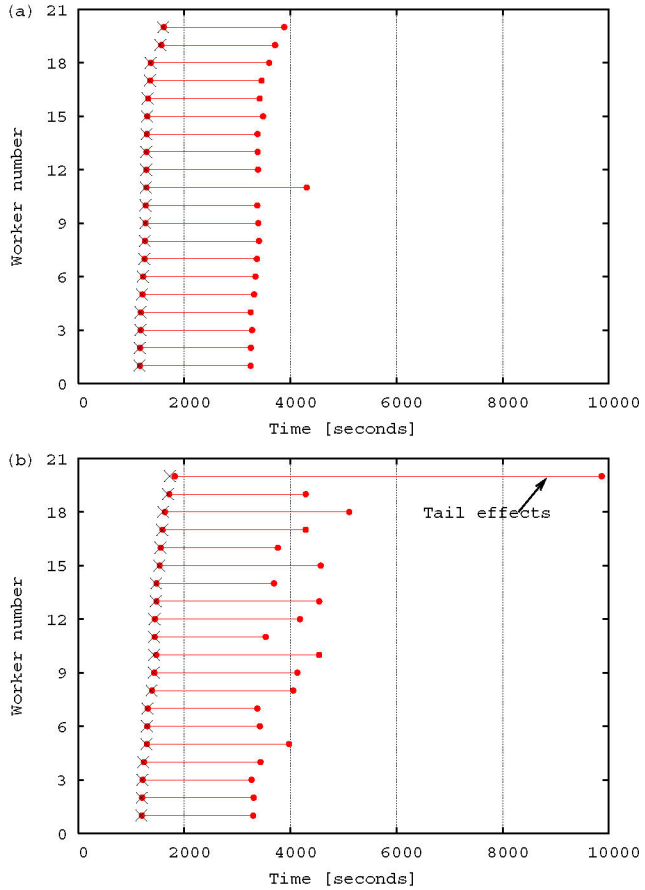

Fig. 1 The effect of a slow node is particularly relevant with a fast use case, i.e. the brachytherapy. In the bottom one can observe that the overall simulation time doubled because of a slow node in the grid.

As a general rule, the task optimization had to be done case by case in relation to the different features of the use cases. In this work two different categories of use case could be identified: fast and intensive simulation. To study the effect task splitting approach in real-life cases the brachytherapy simulation can be considered as an example of fast simulation and the hadrontherapy as an intensive one.

The fast use case has been divided into many small tasks of 20000 events each with a task execution time of about 25 s. 40 workers accomplished the simulation; a higher number of nodes would not be more effective, since the time for the last workers to be registered and become active might affect negatively the total job duration time.

For the intensive use case a different approach was used, in which every workers accomplished a single task. This is not the fastest option, rather is a compromise to deal with the large disk space required in this test case by the simulation output produced by each task.

Two effects contribute significantly to the total simulation time in a grid environment: the presence of a very slow node and the worker registration time.

The effect of a slow node is particularly relevant in the fast use case scenario. In Fig. 1 one can see the results obtained by executing the brachytherapy simulation in the same grid environment on identical computing elements in two different occurrences. Each one of the 20 workers executed one single task. While in the top picture one can see that the simulation time is more or less identical on each node, the bottom picture illustrates the occurrence where a worker was very slow, probably because the workload changed unpredictably. The performance of this slow worker influenced strongly the overall efficiency by approximately doubling the total simulation time. Indeed the overall simulation time is, clearly, determined by the slower node rather than by the average of simulation time over each node. Dividing up the simulation into la large number of small tasks when a fast response is desired contributes to minimize the effects due to slower nodes.

The second effect highlighted by this study is the long time the registration of the nodes on the grid could take in particular conditions. Fig. 2a shows a case of fast registration: even the last registered worker accomplishes a good number of tasks, thus contributing to the parallelization efficiency. Fig. 2b illustrates an example of slow registration: in this case some workers registered late, hence they could perform a relatively small number of tasks, while the fast starting nodes do the major part of the work. An even worse occurrence is shown in Fig. 2c: the effects of delayed registration are clearly visible, moreover one of the workers crashed. Unfortunately, the behavior of the nodes where the simulation is run is unpredictable at the time of launching the simulation.

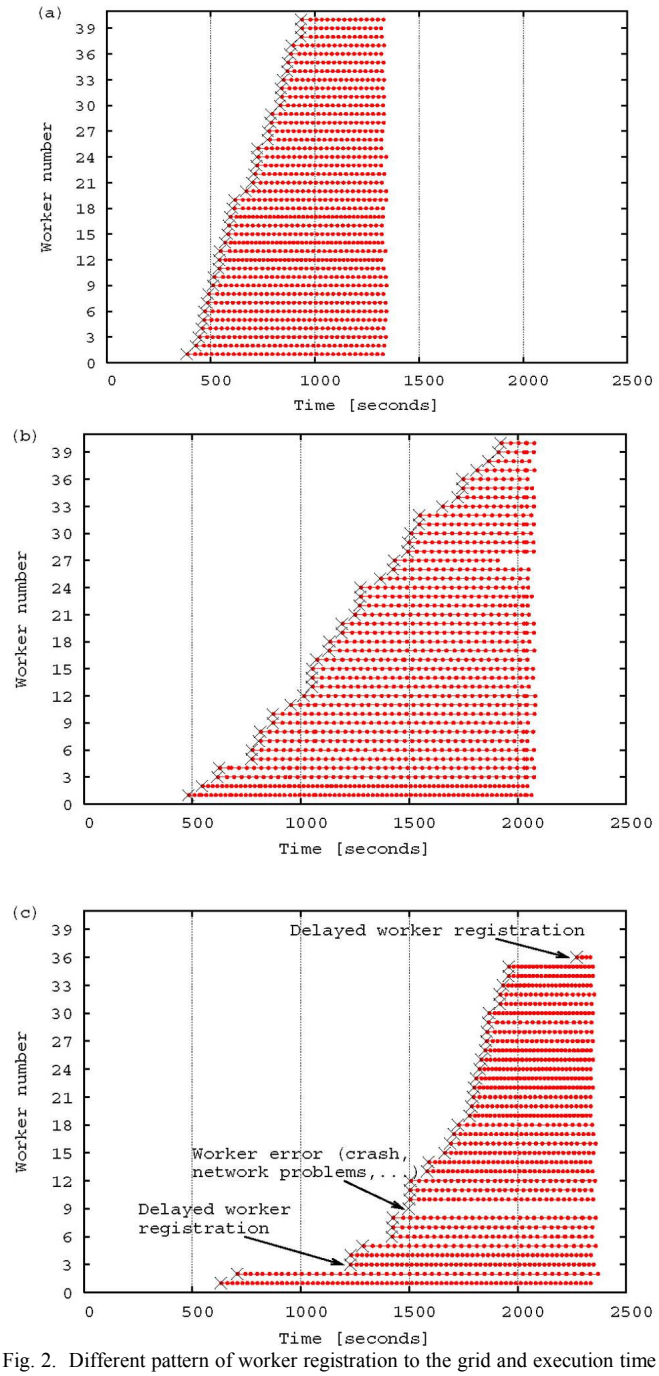

Fig. 2. Different pattern of worker registration to the grid and execution time for the brachytherapy example: fast (a), slow (b) and very slow registration (c).

B. Benchmarks on a computing grid

The benchmarks obtained in this study are summarized in Table III. For both the test cases considered (fast and intensive) 50 simulation runs were performed over a period of about 2 months. In the brachytherapy example a good dose calculation accuracy is achieved in sequential simulation with $2 \cdot 10^7$ events, requiring an overall simulation time of about 40 minutes. For hadrontherapy adequate accuracy for the calibration case is achieved with 10^5 events, resulting in an overall simulation time of about 16 hours in sequential mode.

TABLE III
DETAILS OF SIMULATION RUN ON THE GRID

Test case	Brachytherapy	Hadrontherapy (calibration)
Period of testing	3 weeks	5 weeks
Number of runs performed	50	50
Number of events simulated (per run)	$2 \cdot 10^7$	10^5
Number of DIANE workers applied (per run)	40	20
Number of tasks (per run)	10^3	20
Number of events per task	$2 \cdot 10^4$	$5 \cdot 10^3$
Time per task	25±0.5 CPU s	3000±50 CPU s
Sequential overall simulation time	417±8 CPU minutes	16.7±0.3 CPU hours

Based on the arguments discussed in the previous session, a relatively high number of workers (40), with a high number of tasks per run, represents the optimal configuration for a fast use case such as the brachytherapy simulation. Small tasks of $2 \cdot 10^4$ events takes in average 25 s each to be completed. Fig. 3 shows the histogram of task duration and overall simulation duration for this case. Ideal parallelization efficiency would correspond to an overall simulation time of about 10 minutes; the real execution time is affected by various parameters, first of all the fact that the resources used are not solely exploited for this simulation execution.

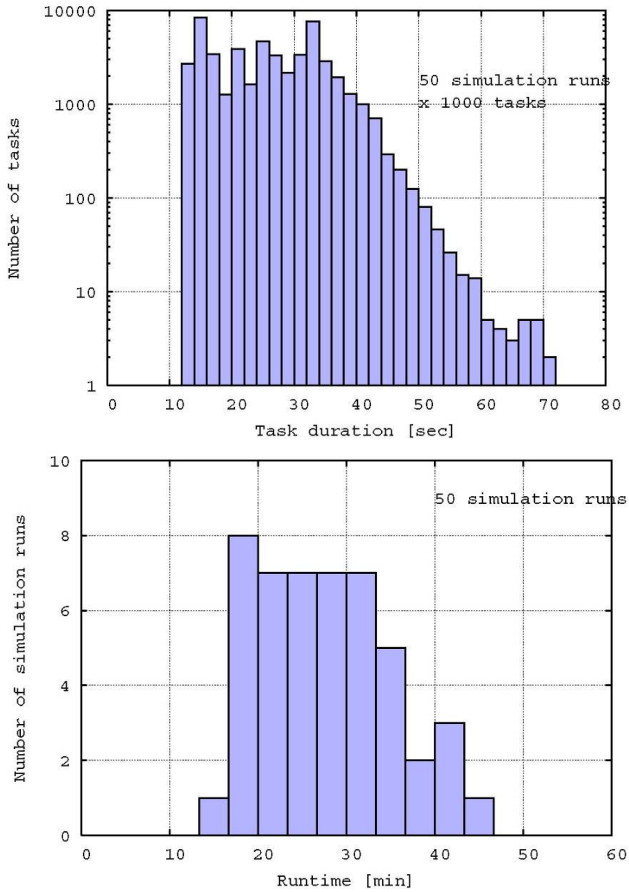

Fig. 3. Histogram of task duration (top) and overall simulation time (bottom) in the brachytherapy example.

Nevertheless the results are quite good: 64% of the runs terminated within 30 minutes and 96% terminated within 40 minutes. Only 4 runs completed in more than 40 minute, while 9 runs terminated in less than 20 minutes. The distribution of task time is quite wide (the maximum to minimum execution time ratio is about 6), with an average value of 25 s per task. A limited amount of tasks completed in more than twice the average time: this result is related to the choice of splitting the execution into small tasks to limit the delay due to slow nodes. In the hadrontherapy case a lower number of workers (20) was used; we adopted the single worker single task approach with tasks of 5·103 events, each one lasting 50 minutes in average. Fig. 4 shows the histogram of task duration and overall simulation duration for this case. An ideal parallelization efficiency would result in an overall simulation time of about 50 minutes; in the real-life test 84% of the runs terminated in 100 minutes. Also in this case the distribution of task duration is quite large, with an average value of 50 minutes and maximum to minimum execution time ratio of about 4.

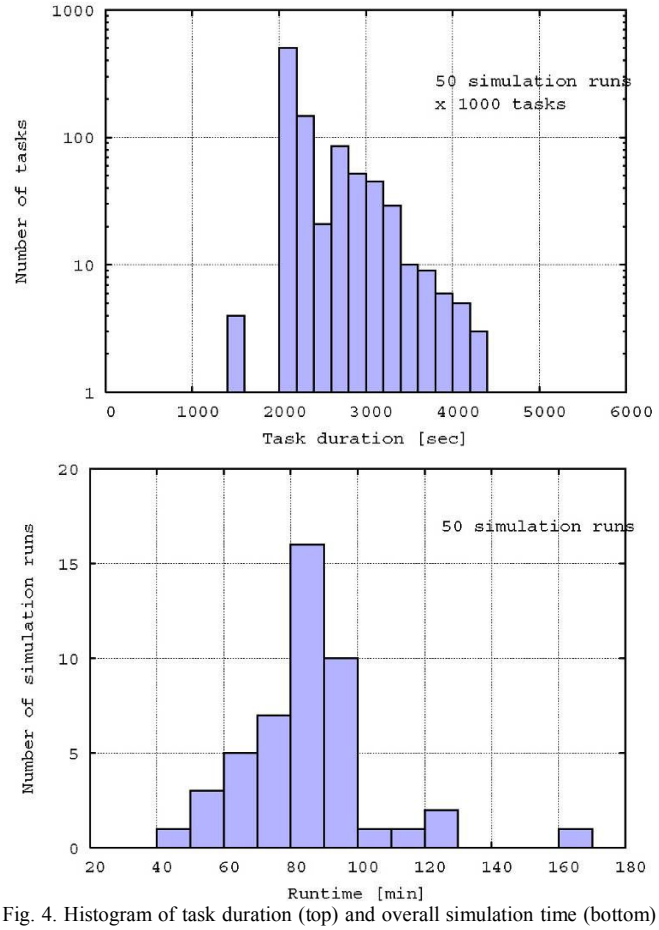

Fig. 4. Histogram of task duration (top) and overall simulation time (bottom) in the hadrontherapy calibration example.

C. Execution of very CPU-intensive use cases

Some additional tests were performed with the hadrontherapy example in the case of a very intensive use case, corresponding to a simulation aimed at obtaining high precision dose estimates. Such a use case requires increasing the statistics of events to be simulated by 2 orders of magnitude: the overall simulation time is expected to be 2-3 days in the same configuration of number of workers and task splitting. Three test runs were performed for this configuration: two did not achieve completion for reasons not intrinsic to the simulation itself, rather ascribable to the computing environment, while one completed successfully. The partial completion rate was about 80% in the two incomplete cases.

The medical linac example exhibits a behavior similar to the other two test cases, when run in equivalent configurations with a low statistics. Nevertheless, its demanding CPU requirements, corresponding to the high statistics needed for adequate accuracy, place it in a position similar to the very intensive use case of hadrontherapy, i.e. requiring some days to produce statistically relevant results. The grid computing environment evaluated in this study does not appear mature yet for the regular production of such challenging simulation.

CONCLUSION

Long simulation time is the limiting factor for the introduction of Monte Carlo dose calculation into clinical practice. Since Monte Carlo methods are intrinsically parallel, a solution to this problem consists in running the simulation in parallel onto a cluster of PCs.

This paper explored the possible solution of running the simulation on a geographically distributed grid rather than a local farm. The documented benchmarks demonstrate that this computing technique is promising for application to medical dosimetry problems: in relatively fast use cases, like brachytherapy simulation, calculation times close to the requirements in clinical practice can be achieved. Computationally intensive use cases, like the simulation of a whole treatment head for IMRT, do not look achievable in the current grid computing environment; nevertheless, due to the fast evolution of this technique, it is realistic to expect that the requirements for such challenging simulation use cases would be satisfied by grid computing systems in the future.

ACKNOWLEDGMENT

The authors are grateful to J. Knobloch for his precious and continuous support. A special thanks goes to A. Mantero for being the first to move his step through the Geant4-DIANE world. Additional thanks goes to the Geant4 collaborators who contributed to the development of the advanced examples used in this work: S. Agostinelli, G. A. P. Cirrone, G. Cuttone, F. Di Rosa, F. Foppiano, S. Garelli, S. Guatelli, M. Piergentili, and G. Russo.

REFERENCES

[1] I. Foster and C. Kesselman, The Grid, Morgan Kaufmann, 2003.
[2] S. Agostinelli et al., “Geant4 - a Simulation Toolkit”, *Nucl. Instrum. Methods A*, vol. 506 pp. 250-303, Jul. 2003.
[3] J. Allison, K. Amako, J. Apostolakis, H. Araujo, P. Arce Dubois, M. Asai, et al., “Geant4 developments and applications”, *IEEE Trans. Nucl. Sci.,* vol. 53, no. 1, pp. 270-278, Feb. 2006.

[4] Low DA, Dempsey JF "Evaluation of the gamma dose distribution comparison method" *Med Physics 30:2455-63, 2003*

[5] F. Foppiano, S. Guatelli, M.G. Pia "**A** general-purpose dosimetric system for brachytherapy" *The Monte Carlo Method: Versatility Unbounded in a Dynamic Computing World*Chattanooga, Tennessee, April 17-21, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL, 2005

[6] G. A. P. Cirrone et al., "Implementation of a New Monte Carlo - GEANT4 Simulation Tool for the Development of a Proton Therapy Beam Line and Verification of the Related Dose Distributions" *IEEE Trans. Nucl. Sci.,* vol. 52, no. 1, pp. 262-265, Feb. 2005.

[7] F. Foppiano, B. Mascialino, M.G. Pia, M. Piergentili "Geant4 Simulation of an Accelerator Head for Intensity Modulated RadioTherapy" in *The Monte Carlo Method: Versatility Unbounded in a Dynamic Computing World,* Chattanooga, Tennessee, April 17-21, 2005, on CD-ROM, American Nuclear Society, LaGrange Park, IL, 2005.

[8] F. Foppiano, S. Guatelli, J. Moscicki, M.G. Pia, M. Piergentili "From DICOM to GRID: a Dosimetric System for Brachytherapy born from HEP", in *Proceedings of IEEE-NSS*, Portland, 2003 and INFN/AE-04/6.

[9] G. Barrand, P. Binko, M. Donszelmann, A. Johnson, and A. Pfeiffer, "Abstract interfaces for data analysis: component architecture for data analysis tools", Proc. of CHEP 2001 Int. Conf. on Computing in High Energy and Nuclear Physics, pp. 215-218, Science Press, Beijing, 2001.

[10] J. T. Moscicki, "DIANE -distributed analysis environment for GRID-enabled simulation and analysis of physics data", in IEEE NSS Conference Record 2003.