Alma Mater Studiorum · Università di Bologna

Scuola di Scienze
Corso di Laurea in Fisica

Evaluation of a Cloud infrastructure for the CMS distributed data analysis in the top quark sector at the LHC

Relatore:
Prof. Daniele Bonacorsi

Correlatore:
Dott. Claudio Grandi

Presentata da:
Luca Ambroz

Sessione Autunnale II Appello
Anno Accademico 2013/2014

Abstract

In particle physics, a great amount of computational power and storage is required to carry on physics analyses. The LHC Computing Grid, a global infrastructure and set of services developed by a large community of physicists and computer scientists, has been deployed on data centres worldwide. It has demonstrated its solid capabilities in the data analysis during Run-1 at the LHC, playing a fundamental role in the Higgs boson discovery.

Nowadays, Cloud computing is emerging as a new paradigm to access large sets of shared resources for many scientific communities. Given the challenging requirements of LHC physics in Run-2 and beyond, the LHC computing community is interested in exploring Clouds and see whether they can provide a complementary approach - or even a valid alternative - to the existing technological solutions.
The purpose of this thesis is to test a Cloud infrastructure and to compare its performance to the LHC Computing Grid.

Chapter 1 presents an overview of the Standard Model. Chapter 2 describes the LHC accelerator and experiments with major focus on the CMS experiment. Chapter 3 introduces Computing in High Energy Physics: Grid and Cloud are also presented and discussed. Chapter 4 reports the original results of my work on a Grid versus Cloud comparative analysis.

Sommario

Nella fisica delle particelle, onde poter effettuare analisi dati, è necessario disporre di una grande capacità di calcolo e di storage. LHC Computing Grid è una infrastruttura di calcolo su scala globale e al tempo stesso un insieme di servizi, sviluppati da una grande comunità di fisici e informatici, distribuita in centri di calcolo sparsi in tutto il mondo. Questa infrastruttura ha dimostrato il suo valore per quanto riguarda l'analisi dei dati raccolti durante il Run-1 di LHC, svolgendo un ruolo fondamentale nella scoperta del bosone di Higgs.

Oggi il Cloud computing sta emergendo come un nuovo paradigma di calcolo per accedere a grandi quantità di risorse condivise da numerose comunità scientifiche. Date le specifiche tecniche necessarie per il Run-2 (e successivi) di LHC, la comunità scientifica è interessata a contribuire allo sviluppo di tecnologie Cloud e verificare se queste possano fornire un approccio complementare, oppure anche costituire una valida alternativa, alle soluzioni tecnologiche esistenti. Lo scopo di questa tesi è di testare un'infrastruttura Cloud e confrontare le sue prestazioni alla LHC Computing Grid.

Il Capitolo 1 contiene un resoconto generale del Modello Standard. Nel Capitolo 2 si descrive l'acceleratore LHC e gli esperimenti che operano a tale acceleratore, con particolare attenzione allesperimento CMS. Nel Capitolo 3 viene trattato il Computing nella fisica delle alte energie e vengono esaminati i paradigmi Grid e Cloud. Il Capitolo 4, ultimo del presente elaborato, riporta i risultati del mio lavoro inerente l'analisi comparata delle prestazioni di Grid e Cloud.

Contents

List of Figures

List of Tables

Chapter 1

Theory overview

1.1 The standard model

The majority of elementary particles and the interactions among them is described thanks to a theory, based on the concept of quantum fields, known as the Standard Model (SM).

In nature there are, known to men, four fundamental forces: gravity, strong interaction, weak interaction, the electromagnetic interaction. The SM has a deep understanding of only the last three. The particles of the SM can be divided into two main categories according to their spin: fermions, which have half-integer spin, and bosons, that have integer spin. Furthermore, each particles has its own antiparticle: a particle that has the same mass and spin, but opposite internal quantum numbers. There are 12 fermions which can be divided in 6 leptons and 6 quarks. Moreover quarks can have three different "colors". Color is the charge that regulates the strong interaction and there are three of them: blue, red and green. Leptons have unitary or null electric charge, where the charge is measured in units of the module of the electron charge. They can be organized in three generations:

$$\begin{pmatrix} \nu_e \\ e \end{pmatrix} \begin{pmatrix} \nu_\mu \\ \mu \end{pmatrix} \begin{pmatrix} \nu_\tau \\ \tau \end{pmatrix}$$

In each family the particles on the bottom are called, from left to right, electron, muon and tau whereas the particles on the top are the respective neutrinos. Electron, muon and tau interact through the electromagnetic and weak forces, while neutrinos only via the weak force. The strong interaction does not influence leptons. Quarks are particles that interact through the strong force, weak force and electromagnetic force.

They can be also organized in three generations:

$$\begin{pmatrix} u \\ d \end{pmatrix} \begin{pmatrix} c \\ s \end{pmatrix} \begin{pmatrix} t \\ b \end{pmatrix}$$

They are called, according to their first letter: up, down, charm, strange, top and bottom. Every quark is identified with a flavour quantum number. The following tables summarize the main properties of leptons and quarks.

	ν_e	ν_μ	ν_τ	e	μ	τ
Charge (e)	0	0	0	-1	-1	-1
Mass (MeV)	$< 2 \times 10^{-6}$	< 0.19	< 18.2	0.51	106	1770

	u	d	c	s	t	b
Charge (e)	$\frac{2}{3}$	$-\frac{1}{3}$	$\frac{2}{3}$	$-\frac{1}{3}$	$\frac{2}{3}$	$-\frac{1}{3}$
Mass (MeV)	350	350	1500	500	180000	4500

Table 1.1: Mass and charge of elementary particles.

		u	d	c	s	t	b
I, I_3	Isospin	$1, \frac{1}{2}$	0	0	0	0	0
I, I_3	Isospin	0	$1, \frac{1}{2}$	0	0	0	0
C	Charm	0	0	1	0	0	0
S	Strangeness	0	0	0	-1	0	0
T	Topness	0	0	0	0	1	0
B	Bottomness	0	0	0	0	0	-1

Table 1.2: Quantum flavour number.

Quarks have always been found in composite states called hadrons. This phenomenon is known as color confinement. There have been observed two types of hadrons: mesons, made of a quark and an anti-quark, and baryons, made of three quarks or three anti-quarks.

1.2 The interactions

The SM describes the electromagnetic force, weak force and strong force with the same theory: the Gauge Theory. The Gauge symmetric group of the SM is:

$$SU(3) \times SU(2) \times U(1)$$

where $SU(3)$ is the symmetry group of the strong interaction and $SU(2) \times U(1)$ is the symmetry group of the electroweak interaction. The interactions are mediated by bosons. Photon (γ) is the responsible for the electromagnetic interaction, the $W^{\pm}$ and Z mediate the weak interaction and 8 gluons (g) carry the strong interaction. Some of their properties are summarized in the following table.

Force	Boson	Electric charge	Spin	Mass (GeV)	Force range (fm)
Strong	$g_1, .., g_8$	0	1	0	1
Weak	$W^{\pm}, Z$	$\pm 1, 0$	1	80.4 , 91.2	10^{-3}
Electromagnetic	γ	0	1	0	∞

Table 1.3: Properties of the force-carrying bosons of the SM.

The last fundamental constituent of the SM is the Higgs boson. It is a neutral particle with zero spin which is the base of the Higgs mechanism. The Higgs mechanism consists of the spontaneous breaking of the gauge symmetry $SU(2) \times U(1)$. Particles interacting with the Higgs field gain their mass.

The Feynman diagrams are powerful tools which help visualise and calculate the probability of quantum processes.

1.2.1 The electromagnetic interaction

The electromagnetic interaction is described by Quantum ElectroDynamics, also known as QED, which is a relativistic quantum field theory. This theory is based on the abelian group $U(1)$ which implies that interaction, among charged particles, is carried through massless bosons known as photons.

The fundamental Feynmann vertex of the QED is:

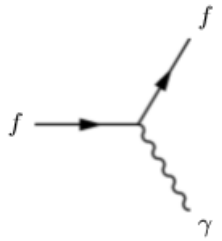


Figure 1.1: Fundamental vertex of the QED.

The probability of the interaction is proportional to the fine-structure constant:

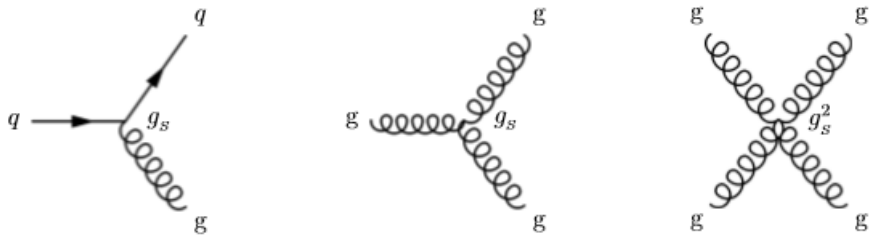
$$\alpha_{EM} = \frac{e^2}{4\pi\epsilon_0\hbar c} \approx \frac{1}{137}$$

1.2.2 The strong interaction

The theory which describes the strong interaction is Quantum ChromoDynamics, also known as QCD. This theory has been developed in analogy to the QED where the abelian group $U(1)$ of the QED has been substituted with the non-abelian group $SU(3)$ which corresponds to the 3 color charges. The force carriers are the gluons which can carry color and anticolor in 8 possible combinations:

$$r\bar{b}, b\bar{r}, r\bar{g}, g\bar{r}, b\bar{g}, g\bar{b}, (r\bar{r} - b\bar{b}), (r\bar{r} + b\bar{b} - 2g\bar{g})$$

The "white" color is not admitted. Since the QCD is not abelian, the interaction among glouns is admitted. Hence the fundamental vertexes of the QCD are:



(a) Interaction quark-quark-gluon.

(b) Interaction among three glouns.

(c) Interaction among four glouns.

Figure 1.2: Fundamental vertexes of the QCD.

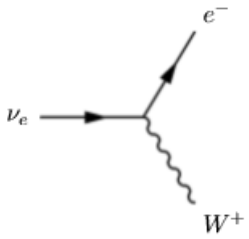
where g_s is the coupling constant for the strong force. The energy potential between two quarks can be written as:

$$U_s = -\frac{4\alpha_S \hbar c}{3r} + kr$$

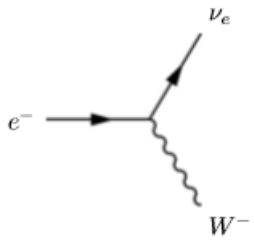
This indicates that the interaction for small distances is repulsive while for big distances is attractive. When two quarks are too far apart, the bond between them breaks, and the energy stored in the bond is used to create a new couple quark-antiquark which combine with the previous quarks to form two new mesons.

1.2.3 The weak interaction

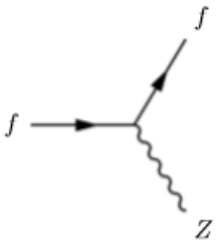
The gauge theory, which describes the weak interaction is based on the symmery group $SU(2)$. The force carrier of the weak interaction are the massive bosons: W^+, W^-, Z^0. All the fermions of the Standard Model are subjected to the weak interaction. The fundamental vertexes of the weak interaction are:



(a) Interaction among a lepton, a neutrino and a W^+.



(b) Interaction among a neutrino, a lepton and a W^-.



(c) Interaction among two fermions and a Z.

Figure 1.3: Fundamental vertexes of the weak interation.

1.2.4 The Higgs in the SM

The Higgs mechanism is responsible for generating the masses of the $W^{\pm}$ and Z bosons in the SM through the spontaneous symmetry breaking of the gauge symmetry $SU(2) \times U(1)$. Particles interact with the Higgs boson proportionally to their masses. Thus the processes which include the quark top, which is the heaviest of all quarks, are very valuable for investigating the nature of the Higgs boson. An example of these processes is the fusion of two gluons that generates two pairs of top and anti-top, one of which fuses to form a Higgs.

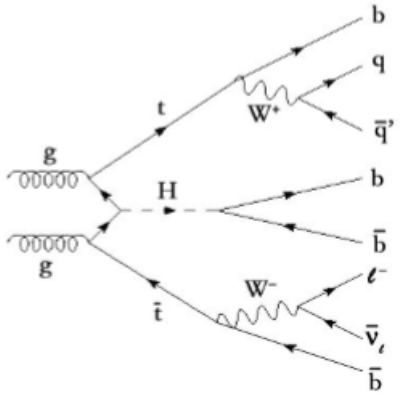


Figure 1.4: An example of production of the Higgs boson: gg Fusion.

Chapter 2

High Energy Physics at the LHC

2.1 The LHC accelerator at CERN

The Large Hadron Collider (LHC) [1, 2] is a two-ring particle accelerator and collider (see Figure 2.1) built by the European Organization for Nuclear Research (CERN) and located beneath the Franco-Swiss border near Geneva in Switzerland where the previous Large Electron-Positron collider (LEP) previously existed [3].

The purpose of the LHC is to give scientists an experimental apparatus that would enable them to test theories in high energy physics, such as the existence of the Higgs boson and supersymmetries. As an example of one of its first results, the discovery of a new particle was publicly announced on July 4th, 2012: said particle fitted very well with the Higgs boson predicted by the Standard Model [4, 5].

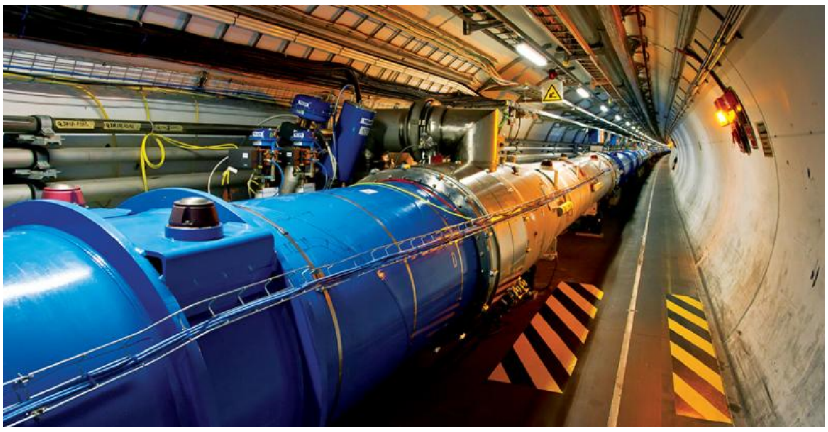


Figure 2.1: The LHC accelerator inside the underground tunnel (copyright CERN).

Protons and heavy ions are accelerated at the LHC . The acceleration process for protons is done in five steps (see Figure 2.2). Initially, hydrogen atoms are ionized in

order to produce protons and then they are injected in the LINAC 2, a linear accelerator. When protons reach the end of LINAC 2, they have reached an energy of $50MeV$ and subsequently enter the Booster where their energy goes up to $1.4GeV$. After that, they enter the Proton Synchrotron (PS) where 277 conventional electromagnets push the protons to 99.9% the speed of light. At this point, each proton has an energy of $25GeV$. Then, proton bunches are accelerated in the Super Proton Synchrotron (SPS), a circular particle accelerator with a circumference of $7km$. After protons have reached an energy of $450GeV$, they are injected into the LHC in two separate pipes in which they move in opposite directions. Here, via magnets, the particles can be accelerated up to their maximum designed energy of $7TeV$. The two pipes of the LHC intersect in the four caverns (where the four detector are installed). Here protons can collide and the products of the collision can be measured. A vacuum system is necessary so the particles do not lose energy in the acceleration process due to impacts with the molecules that constitute air. The LHC vacuum system is made up of three individual vacuum systems: the insulation vacuum for cryomagnets, the insulation vacuum for helium distribution, and the beam vacuum.

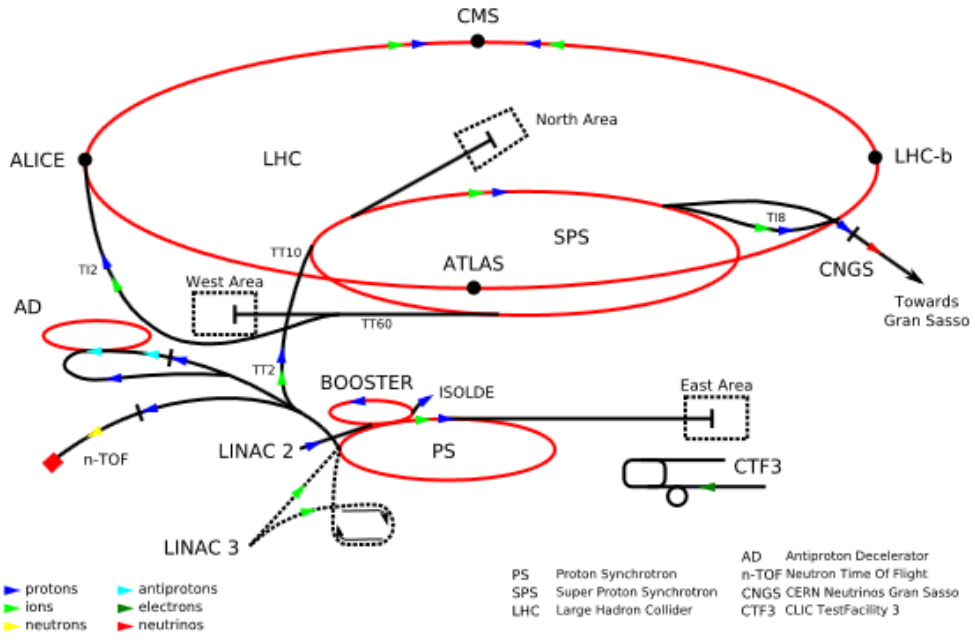


Figure 2.2: Scheme of the acceleration complex (copyright CERN).

In order to keep the path of the subatomic particles stable, the LHC uses over 1600 superconducting magnets made of an alloy based of NbTi. There are 1232 magnetic dipoles whose purpose is to curve the beam along the circumference (see Figure 2.3), 392

magnetic quadrupoles whose duty is to focus the beam when it approaches the detectors, and several smaller correcting magnets. The operational temperature for the magnets is $1.9K$: this allows the magnets to generate a megnetic field up to $8.4T$. A powerful cryogenic system exploits the properties of the superfluid helium and is used to maintain a stable temperature.

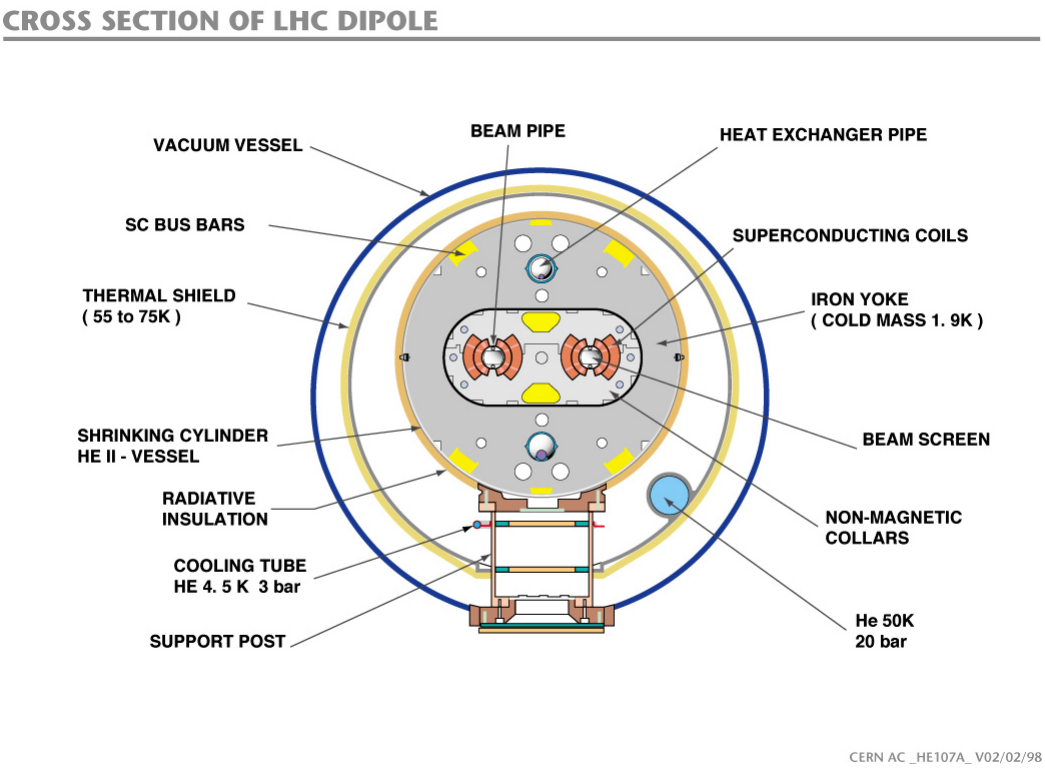


Figure 2.3: Cross section of LHC dipole (copyright CERN).

An important parameter which characterizes a particle accelerator is the machine luminosity ($\mathcal{L}$) defined as:

$$\mathcal{L} = \frac{f_{rev} n_b N_b^2 \gamma_r}{4\pi \epsilon_n \beta^*} F$$

where f_{rev} is the revolution frequency, n_b is the number of bunches per beam, N_b is the number of particles in each colliding beam, ϵ_n is the normalized transverse beam emittance, β^* is the beta function at the collision point, γ_r is a relativistic factor and F the geometric luminosity reduction factor. The number of events that occur each second is:

$$N_{event} = \mathcal{L}\sigma_{event}$$

Some of the main features and parameters of the LHC are summarised in Table 2.1.

Particles	Protons and heavy ions (Lead 82+)
Circumference	26659 m
Injected beam energy	$450GeV$ (protons)
Nominal beam energy for physics	$7TeV$ (protons)
Magnetic field at $7TeV$	$8.4T$
Operating temperature	$1.9K$
Number of magnets	1232
Number of quadrupoles	858
Number of correcting magnets	6208
Maximum Luminosity	$\mathcal{L} = 10^{34}cm^{-2}s^{-1}$
Power consumption	$\sim 180MW$

Table 2.1: Some of the LHC main parameters.

2.2 The experiments at the LHC

There are four main experiments at the LHC, each one located in its own cavern where beams collide (see Figure 2.4).

In the following paragraphs, a few introductory details on each experiment are given. A full description of each detector and its purpose and design features are out of the scope of this work, and references for this can be found elsewhere [6, 7, 8, 9, 10].

ALICE A Large Ion Collider Experiment [6] is a general-purpose, heavy-ion detector which studies the strong interaction, in particular quark-gluon plasma at extreme values of energy density and temperature during the collision of heavy nuclei (Pb). This detector has been designed to identify a great number of single events which happens during each collision of heavy nuclei. The detector weights $10'000$ tonnes and consists of a barrel part, which measures hadrons, electrons, and photons, and a muon spectrometer in the forward region.

ATLAS A Toroidal LHC ApparatuS [7] is an experiment whose main purpose is to investigate new physics beyond the Standard Model exploiting the extremely high energy at the LHC. It also searches the existence of dark matter and extra dimensions. The detector is made of four main layers: the magnet system that bends the trajectories of charged particles; the Inner Detector which measures the path of charged particles; the calorimeters which identify photons, electrons, and jets; and the Muon Spectrometer which recognises the presence of muons. The apparatus is $46m$ long with a diameter of roughly $25m$ and weights approximately $7'000$ tonnes.

CMS The Compact Muon Solenoid [8, 9] is a multi-purpose detector designed to observe a wide variety of phenomena in proton-proton and heavy ion collisions. Its first goal is to investigate the nature of electroweak symmetry breaking that is explained in the Standard Model thanks to the Higgs mechanism. This experiment will be covered in greater detail in the following section.

LHCb The Large Hadron Collider beauty [10] is a detector specialized in the study of the B meson. In particular, in this experiment several aspects of Heavy Flavor, Electroweak and QCD physics are studied. LHCb is a single-arm spectrometer with a forward angular covarage; this design is due to the fact that b and $\bar{b}$ hadrons are produced in the same forward or backward cone. The LHCb detector is made of two types of detectors: the tracking system and the particle identification system that together reconstruct the event.

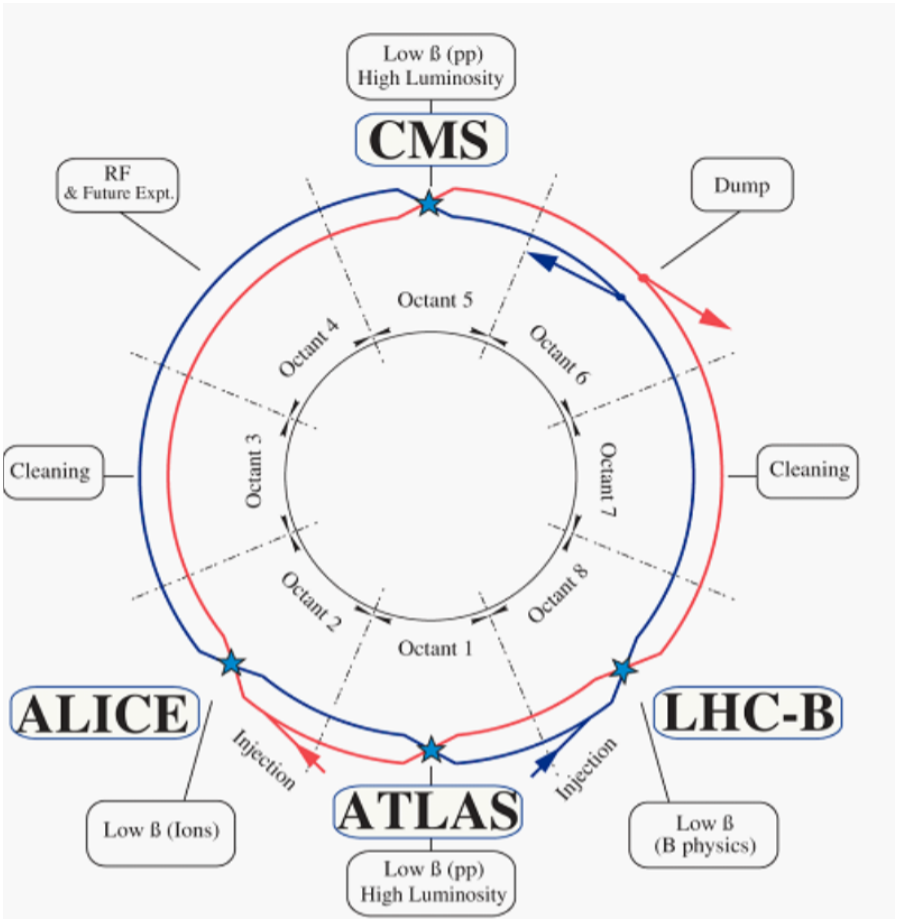


Figure 2.4: Main experiments at the LHC (copyright CERN).

2.2.1 The CMS detector

The CMS (Figure 2.5) [8, 9] is a multi-purpose detector designed to study the Standard Model and explore new physics beyond the SM limits. The discovery of a particle that fits the signature of the Higgs boson, together with ATLAS, represents one of the greatest successes of the CMS collaboration. When protons collide at their maximum designed energy ($\sqrt{s} = 14TeV$), 10^9 collisions/s will occur. The on line selection process has to trigger only 100 events/s to be saved. This high flux of particles needs specialised electronics capable of enduring a high flux of radiation while being able to make extremely challenging selections.

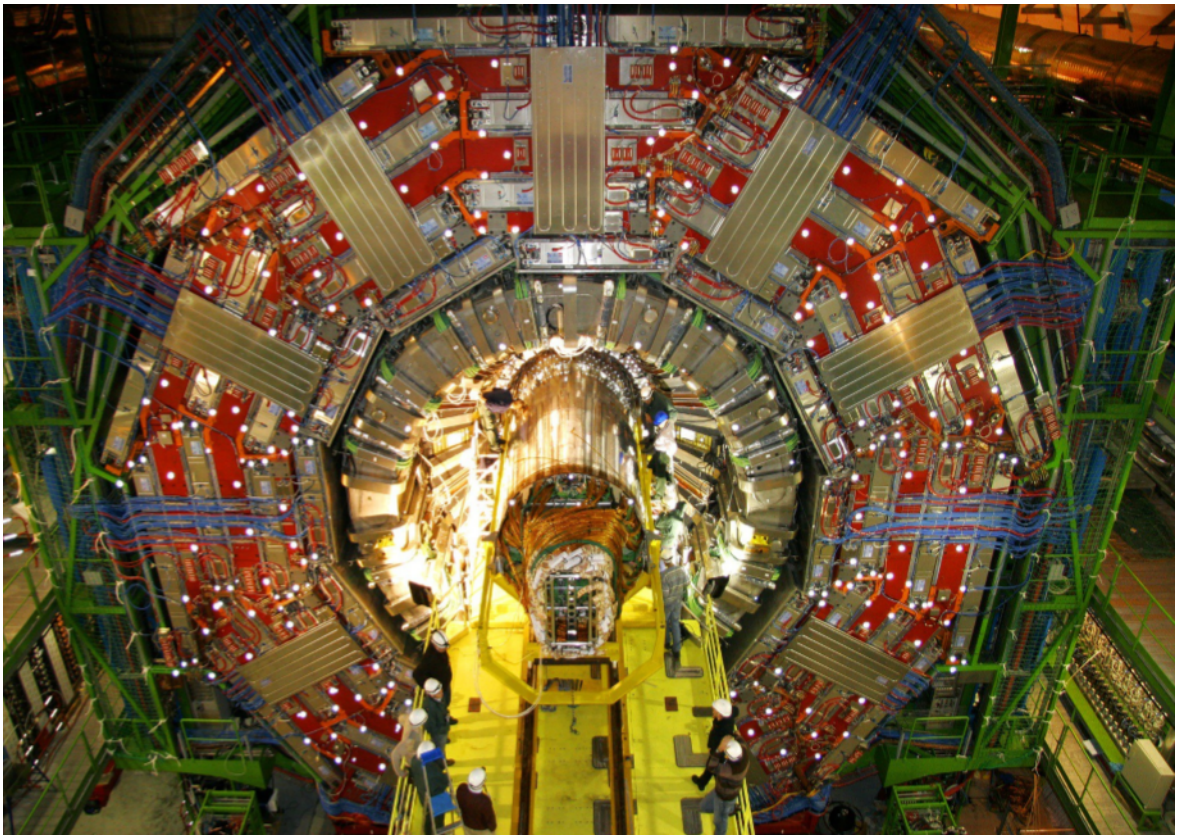


Figure 2.5: Picture of the CMS detector while open (copyright CERN).

When the detector was designed, it had to meet the following general requirements:

- identify and track muons with high precision;
- have a high precision electromagnetic calorimeter for the measurements of electrons and photons;

- have an effective tracking system for the measurement of particles' momenta;
- cover almost the whole solid angle, in order to be able to detect all the particles produced during the collisions.

In order to fulfil these criteria, the detector exploits a powerful solenoid that bends the trajectory of charged particles. Each different type of particle is detected by a specific part of the detector. Combining the knowledge of the particles' path and momenta, it is then possible to trace back the particle involved in the event and determine other information such as their masses. The detector is made up of five concentric layers (see Figure 2.6): the tracker, the electromagnetic calorimeter (ECAL), the magnet, the hadronic calorimeter (HCAL), and the muon detector. A few details on each are described in the following:

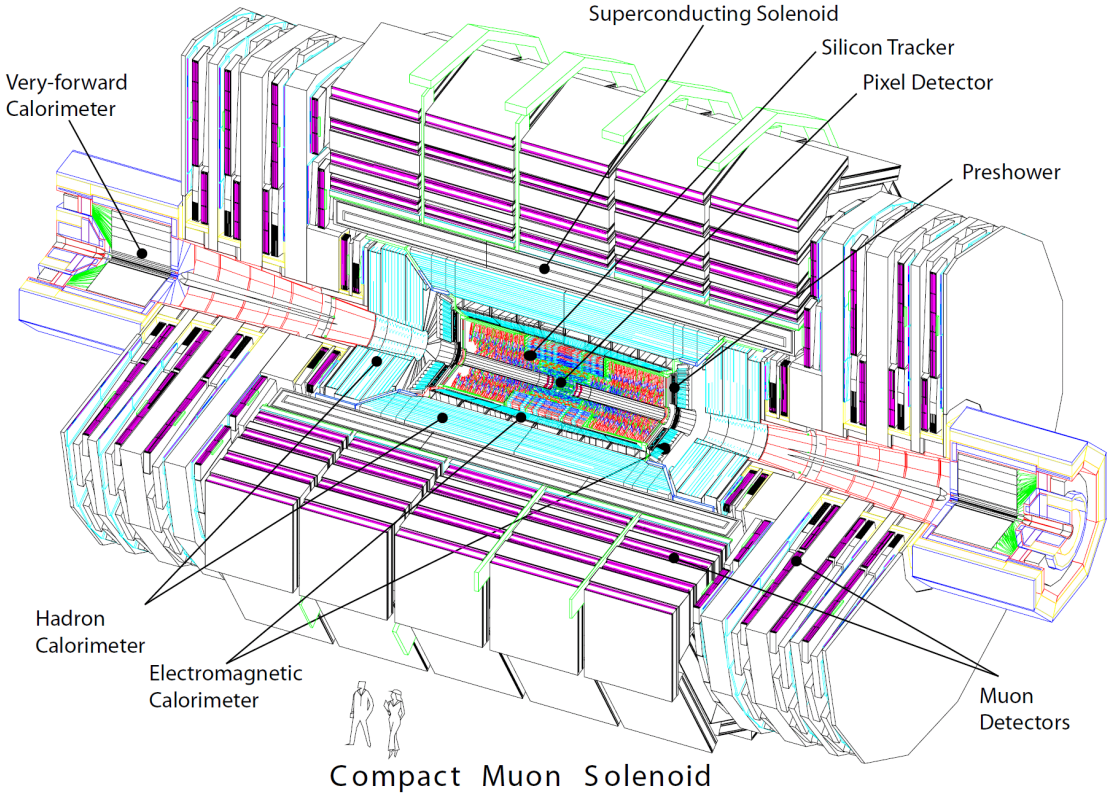


Figure 2.6: Section of the CMS detector (copyright CERN).

- The tracker is able to detect muons, electrons, hadrons and tracks coming from the decay of short-lived particles (e.g. b quarks). Since it is the innermost element of

the detector it has to interfere the least with the particles produced. Most measurements are accurate to the $10\mu m$ level. The tracker has been built exclusively using silicon-based technologies; this choice allowed it to meet requirements such as radiation hardness and speed of acquisition. The CMS tracker is made of three layers of pixels, 4, 7 and $11cm$ away from the beam, surrounded by a silicon microstrips detector. In the pixels and the microstrips, when a particle passes through either one of them, an electric signal is measured, similarly to the functioning of a digital camera when a photon hits one of its pixels.

- The electromagnetic calorimeter is designed to detect particles that interact according to the QED such as photons and electrons. The main components of the ECAL are lead tungstate ($PbWO_4$) crystals that cover the entire solid angle: when the crystals are hit by a particle, scintillation occurs. With the purpose of recording the light emitted by the crystals, Avalanche PhotoDiodes (APDs) are placed around the calorimeter and Vacuum PhotoTriodes (VPTs) are placed in the endcaps. A preshower detector is placed at the end of the ECAL to distinguish single high-energy photons from pairs of low-energy photons. The preshower detector consists of two planes of lead and several silicon sensors. A photon that hits the lead generates an electromagnetic shower, made of electron-positron pairs, that is detected accurately by the sensors. It is then possible to trace back the initial energy of the photon.

- The magnet, which contains the tracker and the electronic calorimeter, is a superconducting solenoid in which a current flows that generates a uniform magnetic field up to $4T$. It is $12.5m$ long and the inner diameter is $6m$. The magnets also provides mechanical stability to the detector.

- The hadron calorimeter measures mainly hadron jets, neutrinos and exotic particles. The HCAL is a sampling calorimeter which uses "absorbers" to measure parameters such as position and momentum of the particles, and it is endowed with fluorescent scintillator materials which light up when a particle passes through them. All the light measured by the sensors is then added up to estimate the energy of the particles.

- The muon detector is placed on the outer layer of the detector, as muons are relatively non-interacting particles; in fact, they are able to pass through several meters of iron without loosing much energy. Needless to say, since they give their name to the detector, they are extremely important in several process, such as the decay of the Higgs boson in four muons. There are three types of gaseous particle detectors for muons' identification. The paths of the muons are obtained by interpolating a curve through the points of the detector hit by the particles (see Figure 2.7). There are 1400 muon chambers; 250 drift tubes (DT) and 540 cathode

strip chambers (CSC) to identify particles' position and give a trigger. 610 resistive plate chambers (RPC) form a second trigger. DTs and RPCs are displaced around the beam line whereas CSCs and RPCs complete the endcaps disks at both ends of the barrel.

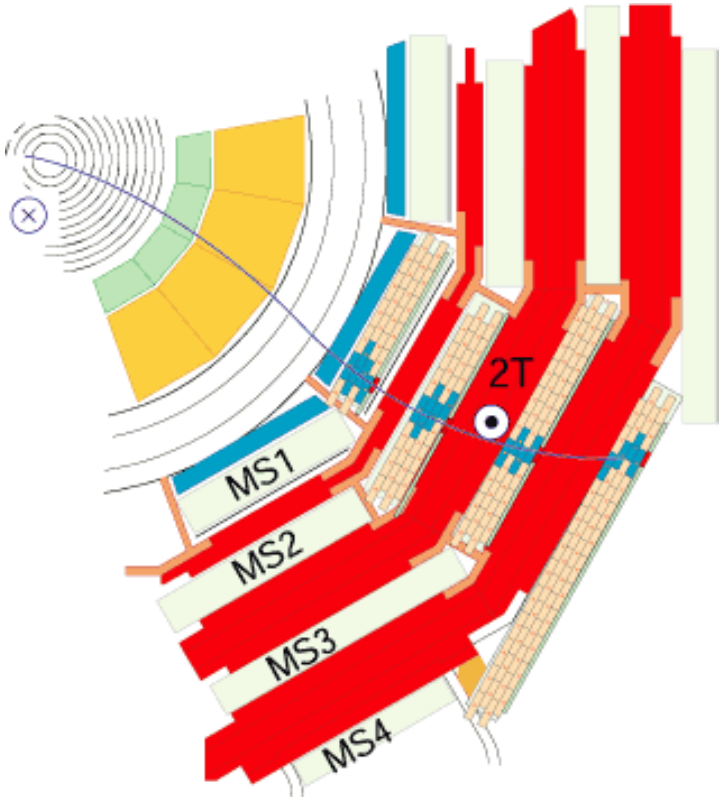


Figure 2.7: A schematic view of a muon trajectory inside the detector (copyright CERN).

Chapter 3

Computing in High Energy Physics

3.1 Introduction

During LHC operation, the accelerator produces a huge amount of data that has to be stored and later analysed by scientists. In each collision swarms of particles are produced and the signals leaving the detector are recorded. Roughly 30 Petabytes (PB) of data are produced at the LHC every year. To deal with all this information, a complex computing infrastructure has been designed and deployed that takes advantage of different computing centers distributed worldwide known as the Grid.

3.2 Grid technologies and WLCG

The Worldwide LHC Computing Grid (WLCG) project [11, 12] is a global collaboration responsible for building and maintaining a data storage and analysis infrastructure required by the experiments at the LHC. The main purpose of this infrastructure is to provide computing resources to store, distribute and analyse the data produced by the LHC to all the users of the collaboration regardless of where they might be. This idea of a shared computing infrastructure is at the basis of the concept of the Grid. The WLCG cooperates with several Grid projects such as the European Grid Infrastructure (EGI) [13] and Open Science Grid (OSG) [14]. The middleware projects provide the software layers on top of which the experiments add their own (different) application layer. At the middleware layer, the main building blocks that make up the infrastructure are the logical elements of a Grid site, namely:

- the Computing Element (CE) service, that manages the user's requests for computational power at a Grid site;
- the Worker Node (WN), where the computation actually happens on a site farm;

- the Storage Element (SE), that gives access to storage and data at a site. Data are stored on tapes and disks. Tapes are used as long-term secure storage media whereas disks are used for quick data access for analysis;
- the User Interface (UI), the machine on which a user interacts with the Grid;
- central services, that help users access computing resources. Some examples are data catalogues, information systems, workload management systems, and data transfer solutions;

The Storage Federation, which is a relatively new infrastructure developed in parallel to the site SEs provides read access to the same data, but does not rely on a catalogue to locate the files but on a set of "redirectories". When the user looks for a file, the redirector looks in the local storage. However, if the redirector does not find the data locally, it can ask the SE in its federation whether it has the file. In case the file is not found, the redirector can ask a higher level redirector if it can find the file. This process continues until either the file is found or the highest redirector does not find anything.

Grid security is based on X.509 certificates which provide authentication for both the user and the services. The user is endowed with a Grid certificate that is used to access the services he desires. A private key is assigned to the holder of the certificate and a public key is used to make requests to a service. The authorization is based on the Virtual Organization Management System (VOMS) [15]. VOMS contains all the users of the Grid and the tasks which they can perform on the Grid itself.

The computing centres around the world are organized into four types of "Tiers" [16] depending on the kind of services they provide:

Tier-0 There are two physical locations for a unique logical Tier-0 function: one is the CERN Data Centre in Geneva (Switzerland) and the other is located at the Wigner Research Centre for Physics in Budapest (Hungary). The Tier-0 is responsible for keeping the RAW data, for the first data reconstruction, for the distribution of the RAW data and the reconstruction output to the Tier-1s, and for the reprocessing of data when the LHC is not acquiring data.

Tier-1 There are 13 LHC Tier-1 sites, of which 7 were available to CMS in Run-1. They are used for large-scale, centrally-organized activities and can exchange data among them and to/from all Tier-2 sites. They are responsible for storing RAW and RECO data, for large-scale reprocessing, and for safe-keeping of the corresponding output, plus a share of the simulated data produced at the Tier-2s. Recently they have been commissioned as sites where users can perform their analysis.

Tier-2 There are now about 160 Tier-2s in LHC, distributed around the world (about 50 available to CMS in Run-1). They are usually Universities or scientific Institutes, and they often have significant CPU resources for user analysis. Tier-2 do not have tape archiving, so they have limited capabilities of storage with respect to the Tier-1s. They also handle tasks such as data generation and simulation.

Tier-3 A Tier-3 can be, for example, a cluster of relatively small size, which is connected to the Grid. There is no formal agreement among WLCG and Tier3-s, which makes such Grid-enable site the most flexible Tier level.

3.3 The CMS Computing model

In order to carry out CMS physics analysis, scientists have to be able to access the huge amount of data collected by the detectors. Furthermore, they need to be granted a lot of computational power in order to run their analysis or generate Monte Carlo simulations. To these requests, one has to add the difficulties originating from the fact that CMS is a project with collaborators from many nations. To cope with these challenges CMS uses the Grid. More specifically, the Tiers in the CMS Computing model [17, 18] have the roles outlined in the following.

Tier-0 The tasks of a CMS Tier-0 are as follows:

1. it accepts data from the DAS (Data Acquisition System) [19];
2. it stores RAW data on tapes;
3. it performs a first reconstruction;
4. it distributes the RAW data to the Tier-1s so there are two copies of all RAW data.

Tier-1 The main functions of a Tier-1 for CMS are:

1. providing a great amount of CPU for data reprocessing and data analysis;
2. data storage of RAW and RECO data (see below);
3. data distributions to/from Tier-2s;

Tier-2 The Tiers-2 of CMS provide:

1. data transfer from/to Tier-1s;
2. services for data analysis;
3. productions of Monte Carlo events;

Tier-3 CMS Tier-3s are not formally bound to WLCG even though they can offer flexible computing capabilities.

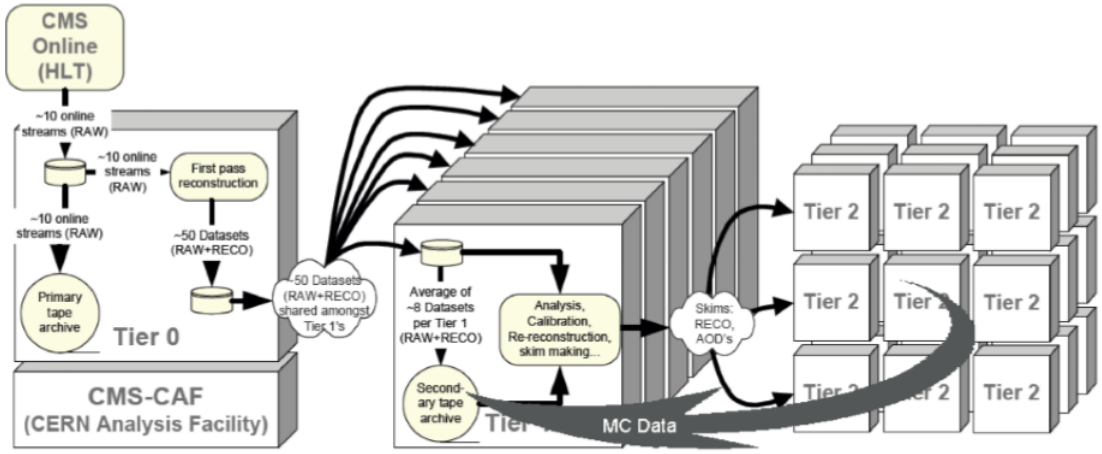


Figure 3.1: Data flow in the CMS computing model.

There are various types of data that flow through the Grid for the CMS experiment. Some of the most important (see Figure 3.1) are:

RAW The data as they are recorded by the detector.

RECO An elaborated (reconstructed) form of data that later could be used for analysis. It can contain tracks, vertices, jets and etc.

AOD Analysis Object Data. This data type is a subset of RECO. It contains information suitable for most analyses.

3.4 Grid vs Cloud? Usage of Cloud technologies in CMS

3.4.1 Cloud Computing

Cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction [20].

Essentially, Cloud computing represents a way in which an institution, either private or public, can overcome the problem of managing its servers. In fact, in the past, each institution had only one option: to buy its servers, to buy and install software, and then to keep everything operational and updated throughout the years. But now, thanks to Cloud computing, it is not necessary for an institution to worry about the hardware. Instead it can buy virtual servers from a private company and run its software on them.

Virtual servers are servers that a company can loan to a user and the user pays according to the usage of the computing resources such as CPU, memory and storage. Right now several private companies can provide Clouds. Furthermore, Cloud computing has turned out to profitable both for the company that provides the service and for user that buys it. Thus some advantages of Cloud computing from an user point of view are:

- Cost. It can be very economically beneficial since it does not require considerable investment in buying hardware and so there is no up-front cost.
- Backup and Recovery. Usually these necessities are handled by the Cloud service provider.
- Software Updates. Very often the software is updated by the Cloud provider.
- Environmentally Friendly. A Cloud provider can utilize its servers at 100% of their potential whereas a private server is utilized only according to the needs of its company.
- Flexibility. Access to computing resources can be scaled quickly according to the needs of a company.
- Access to resources from wherever it is needed. In fact, as long as there is an internet connection it is possible to access data.

Some main disadvantages are:

- Loss of data ownership. A company physically does not own its data as the server could be located on the other side of the world.
- Security problems. An institution is not more responsible of taking care of the security of its own data. Thus it is necessary to verify carefully the security of the Cloud service provider.
- Technical problems. Since technical problems can occur from time to time, a company is at the mercy of its Cloud service provider.

3.4.2 Grid vs Cloud

What are the differences between Grid and Cloud computing? The idea behind the GRID is about sharing computing resources among a partnership of institutions. It has been implemented successfully by several scientific communities. Conversely, Cloud computing is about purchasing computing resources according to the needs. It has been implemented successfully by a lot of private companies.

3.4.3 Usage of Cloud technology in CMS

Recently, Cloud resources have been made available to the CMS experiment. In order to exploit these resources it has been necessary to support a Cloud interface in addition to the standard Grid interface (CE). Actually a possible scenario for the future is a slow transition between the current Grid infrastructure, used only by the scientific community, to a Cloud infrastructure, that is becoming an industrial standard. Today the challenge for the HEP community is to get resources allocated dynamically in the same way used for allocating job slots on the Grid, instead of accessing computing resources through the static allocation of (virtual) machines, as it happens in a normal Cloud.

The most significant examples of Cloud implementation used by CMS are the online farm where the High Level Trigger (HLT) [21] runs and the Agile Infrastructure (AI) [22] in the CERN Computing Centre. In addition to these, some tests have already been done on private (HEP institutes) and public (commercial) clouds, such as Amazon [23].

When the LHC is acquiring data, the HLT works as a second level trigger of CMS. However, when the acquisition stops, the HLT can be exploited for offline calculations since its computational power is comparable to the sum of the Tier-1's of CMS. For this reason a Cloud infrastructure has been ovrlaid to the HLT farm and CMS can use it when not needed by the data acquisition. This environment has been essential to develop the CMS interface to clouds and test it at a scale. Furthermore, recent studies have tackled the possibility of checkpointing CMS jobs. Checkpointing would provide the possibility to quickliy release the HLT resources used during the short moments in which LHC is running but not acquiring data (e.g. beam dump) without losing the work done so far. The Agile Infrastructure (AI) is the name of the CERN computing infrastructure managed with Cloud tools. This will be the standard resource allocation system for CERN in Run-2 and both the CMS Tier-0 and CERN Analysis Facility (CAF) will be provided on the AI.

On both HLT and AI OpenStack [24] is used as software for Cloud management. CMS prepares virtual machine images using the tool OZ [25] and the software is brought on the machines through CVMFS [26] via a set of http proxies.

In CMS resources are accessed on Cloud using the same tools used on Grid, through a serviced called GlideinWMS [27]. In this way the framework allows the user to submit jobs either on Cloud or on Grid with the simple change of few parameters in the job description.

3.5 CMS Distributed Analysis with CRAB

The analysis, both of data and of simulated samples, is done in CMS on the Grid using a toolkit called CRAB (CMS Remote Analysis Builder) [28, 29]. CRAB provides an interface for the user to interact with the distributed computing infrastructure. Generally

speaking, an analysis usually includes two main steps: first the user develops his analysis code locally and tests it on a small scale; second, the user can prepare and submit a large set of jobs to run on an actual large dataset using CRAB. Usually, an analysis is made up of hundreds of jobs which are created and managed by CRAB. Throughout this whole work, the second version of CRAB, known as CRAB 2, will be used, i.e. the same that has been successfully used in the first run ("Run-1") of the LHC.

In order to perform an analysis, the user has to write a configuration file in a specific meta-language, which has the default name *crab.cfg*. The *crab.cfg* is divided in various sections such as CRAB, USER, CMSSW and GRID. The following parameters for the *crab.cfg* are mandatory:

- **jobtype**: the type of job which has to be executed;
- **scheduler**: the name of the scheduler that has to be used;
- **datasetpath**: the complete path and name of the dataset which has to be analysed;
- **pset**: the name of the CMSSW configuration file;
- **out_file**: the output file name generated by the *pset* file;
- **return_data**: allows to retrieve the output in the local working area. The options are 0 or 1;
- **copy_data**: allows to copy the output to a remote SE. The options are 0 or 1.

Furthermore, it is necessary to specify if the jobs are going to be split according to the luminosity, which is mandatory for real data, or by the number of event, which is possible only for Monte Carlo events. In both cases, two parameters have to be specified out of a list of three. For job splitting by luminosity:

- **total_number_of_lumis**: defines the number of luminosity blocks to be analysed;
- **lumis_per_job**: specifies the number of luminosity blocks that a job can access;
- **number_of_jobs**: establishes the total number of jobs that are going to run.

For job splitting by event:

- **total_number_of_events**: the total number of events to be analysed;
- **events_per_job**: assigns to each job the number of events it can access;
- **number_of_jobs**: specifies the total number of jobs that are going to run.

After the proper working environment has been set-up and both the *crab.cfg* and the CMSSW configuration have been written, it is possible to create the jobs via the command:

crab -create

This command creates the jobs according to the specifications in the CRAB configuration file. The next step is to submit the jobs to the Grid, which is done via the command:

crab -submit

At this point, it is possible to check the status of the jobs via the command:

crab -status

A standard tool used for monitoring the status of the jobs is the Dashboard. The Dashboard provides a large set of monitoring metrics and useful information to the users who submitted jobs to the Grid. In particular, a task monitoring service is in place, which offers monitoring specifically targeted to help a user track the status of his/her jobs over time, including successes versus failures, etc [30]. Moreover, further information is provided by the log files that can be retrieved with the command:

crab -getoutput

The above command gets only log files of the jobs which are in the status "Done". For the other jobs, either they have not completed yet (so the user must wait to recover the output), or they have failed (and the user can debug their failure reasons, and possibly resubmit them via CRAB again).

Chapter 4

Study of the performance of jobs execution on Grids and Clouds

4.1 Introduction

Comparing the functionalities and performances of Grid infrastructure versus a set of resources accessible via Cloud interfaces is not a trivial task. In the context of this work, by "Grid" we mean the WLCG Computing infrastructure used by CMS in production [11, 12], while for "Cloud" we refer to the set-up presented and discussed in section 3.4.3. A specific set of workflows relevant to the CMS Computing activity was defined and run onto the two different computing infrastructures. A definition of some interesting metrics to compare the two sets of submissions was made a-priori. These observables were subsequently measured, and the outcome of the measurements is discussed. Despite the conclusions of these tests cannot claim to be of general relevance in a Grid vs Cloud context, they are a contribution to the ongoing activities aiming to increase the knowledge and experience in evaluating and using different technologies available to CMS Computing in preparation for Run-2 (and beyond).

4.2 Description of workflows

A total number of 3 different workflows have been defined and submitted to test Grid and Cloud computing environments. These workflows are presented and briefly discussed below, together with the metrics associated to each.

- "Light" Workflow. A very simple test workflow aimed at demonstrating only the functionality of the workload management infrastructure accessible via Grid or Cloud interfaces, on the basis of a set of simple observables like the job success rate, the time needed for job submissions, the time needed for job completion, etc. This

workflows is comprised of very quick jobs (seconds per event) accessing the dataset: `/GenericTTbar/HC-CMSSW_5_3_1_START53_V5-v1/GEN-SIM-RECO`

- "Heavy" Workflow. Despite the name, this test workflow is far from being actually "heavy on the computing resources, but it was tuned on purpose to be as simple as the previous one although heavier in using CPU cycles on the WN (Worker Node) on which the jobs land, which is hence exploited for a longer time, approximately a couple of minutes per job. This will be explained in more detail later on, in a dedicated paragraph. Despite being simple, this workflow allows to perform some effective investigations over a set of time-related observables, including a comparison of the CPU efficiency (defined as CPT/WCT, i.e. CPU time divided by the Wallclock time) in the Grid and Cloud infrastructures used for the tests.

- "Real" Workflow. This is a "real" workflow used in the official CMS data analysis in the hadronic top sector. It will be explained in more detail later on, in a dedicated paragraph. Running this on resources accessible via Grid and Cloud interfaces, a comparison of the full set of metrics recorded in both scenarios is hence possible under a realistic CMS workload.

In the following paragraphs, the submissions outcome of each workflow is presented and discussed.

4.3 Using a "light" workflow to test basic functionalities

The aim of this first study is to evaluate two workflows which use the same dataset for analysis:

`/GenericTTbar/HC-CMSSW_5_3_1_START53_V5-v1/GEN-SIM-RECO`

and are relatively light in terms of CPU occupancy, duration and load on the overall system. The workflows are characterized by one *crab.cfg* file and one *pset* file that contain the main information concerning the jobs' submission. In both workflows the storage element, for storing the output files, was `T2_IT_Legnaro`. The jobs have been configured to run on 10000 events in total from the aforementioned dataset, divided in 100 jobs running on 100 events each. There are two main differences in the two workflows. The first one uses the standard CMS glidein-WMS while the other uses a glidein-WMS specifically configured for Cloud testing. This difference is encoded in the line of the cloud *crab.cfg*:

```
submit_host = lnl_submit-6
```

The second difference is that in the jobs are specifically sent to the Agile Infrastructure (AI) at CERN (see more details in section 3.4.3). This option is encoded in the lines of the Cloud *crab.cfg*:

```
se_white_list = T2_CH_CERN_AI

max_rss = 1900
```

The complete *crab.cfg* is in Appendix A.1. For each workflow, independent submission were performed, and data were collected. The first and most important parameter that was measured is the rate of success of the jobs. This is reported in Table 4.1.

(a) Grid

Task	Sucesses	Failures	Unknown
1	95	0	5
2	98	0	1
3	97	3	0
4	96	1	3
5	100	0	0
6	100	0	0
7	97	2	1
8	99	0	1
9	99	1	0
10	98	2	0

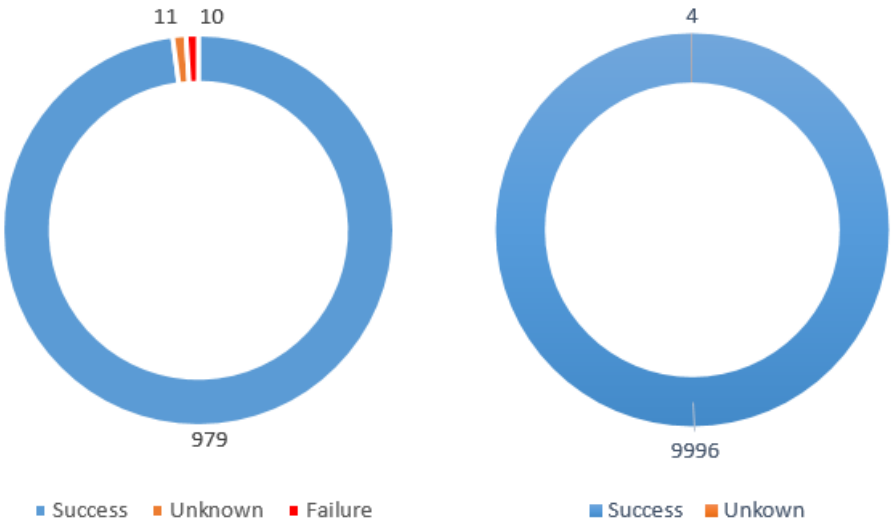
(b) Cloud

Task	Successes	Failures	Unknown
1	96	0	4
2	100	0	0
3	100	0	0
4	100	0	0
5	100	0	0
6	100	0	0
7	100	0	0
8	100	0	0
9	100	0	0
10	100	0	0

Table 4.1: Comparison of success, failure, and unknown outcomes as from the submission of the "light" workflow on (a) Grid resources and (b) Cloud resources.

The Table 4.1 can be summarized by the graphs in Figure 4.1.

From the graph it can be observed that 14 jobs out of 2000 finished in an *"unknown"* status on the monitoring. The monitoring of CMS jobs in general is performed via several tools, and the most generic and omnicomprehensive is the Dashboard [30] tool. It is a known feature (being worked on by the experts from the CERN IT division) that this tool may intermittently loose track of a fraction of the jobs submitted by a user, and this may happen for a variety of reasons, which are mostly related to the fact that its architecture elaborates the job information collected from several independent sources, thus yielding a quite complex system. On the other hand, it is always true that a submitter can rely on his/her own log files from the application itself. To by-pass the



(a) GRID (b) Cloud

Figure 4.1: Same as in Tabel 4.1 in a pictorial representation. The total number of jobs is also indicated.

"unknown" statuses, it was checked via the CRAB2 command line interface (i.d. *crab -status*) that all the jobs belonging to this task actually ended up in a *"DONE"* status. It can hence be concluded that the *"unknown"* status is most probably caused by a monitoring glitch, or an otherwise unwanted behaviour at the monitoring level, and it should not be categorised as a CRAB2 issue: the Dashboard lost track of these 14 jobs, despite they actually terminated successfully.

Checking the reasons for the job failures, the Dashboard reports a total of only two different causes of errors in the jobs:

- **Error 60307**, which corresponds to "Failed to copy an output file to the SE (sometimes caused by timeout issue)".
- **Error 8001**, which corresponds to "Other CMS Exception".

Thus the job success rates are (for Grid and Cloud respectively):

$$R_{Grid} = 98.9\%$$

$$R_{Cloud} = 100\%$$

This may lead to conclude - despite under a limited test with a "light" workflow that checks only the functionalities of the two technologies - that the Cloud infrastructure

under test is potentially as reliable as the Grid one. Once its reliability has been shown, it is possible to investigate its performance. The time needed to complete each job has been evaluated both for Grid and Cloud.

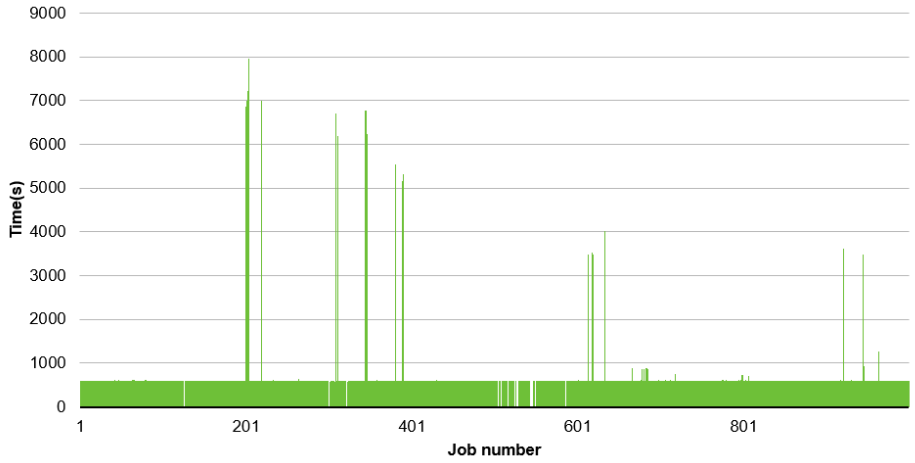


Figure 4.2: Time required for the execution of each test job in the Grid infrastructure.

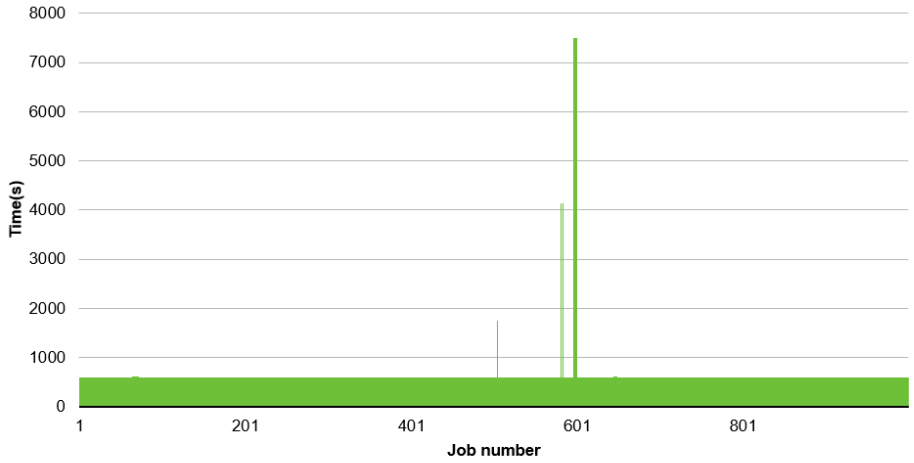


Figure 4.3: Time required for the execution of each test job in the Cloud infrastructure under test.

As can be seen on Figure 4.2 for submissions to Grid and on Figure 4.3 for submissions to Cloud, a large majority of the jobs have a duration that is approximately 600 seconds. Only a very small fraction of jobs last longer than this: this is very visible on the plots as they requires several hours to complete. Their number is larger for Grid submissions, i.e. they correspond to 2% of the submitted Grid jobs, and to 0.7% of the submitted Cloud jobs. If we calculate the average time that is necessary to a job to be finished, both for Grid and Cloud we have:

$$t_{GRID} = (60 \pm 9) \times 10s$$

$$t_{Cloud} = (60 \pm 4) \times 10s$$

Another interesting feature to check in comparing the two kind of submissions is the submission pattern itself, i.e. the distribution of the time when each job started to run after the initial bulk submission of the task as a whole. For example, in the submission of jobs from the "light" workflow to the Cloud infrastructure, the submission times are shown in Figure 4.4.

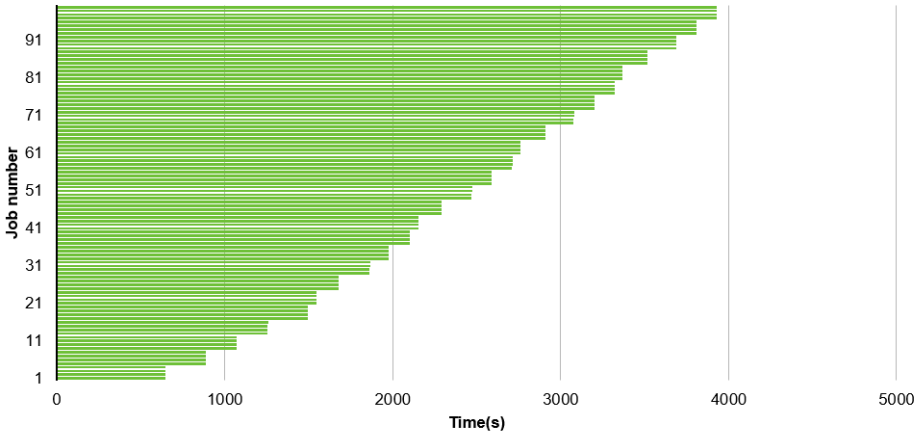


Figure 4.4: Distribution of the starting times of the jobs for Cloud submissions (see text).

It is possible to observe that not all 100 jobs start together, instead they start in group of four. After a group is started there is a delay of several seconds due to the fact that there are some protocols internal to the Cloud which do not allow to a single user to book too many computing resources in a short amount of time. It has been noticed that the last jobs can wait up to several hours before being submitted. Conversely, when the "light" workflow is submitted to the Grid, all jobs start either immediately or within few minutes.

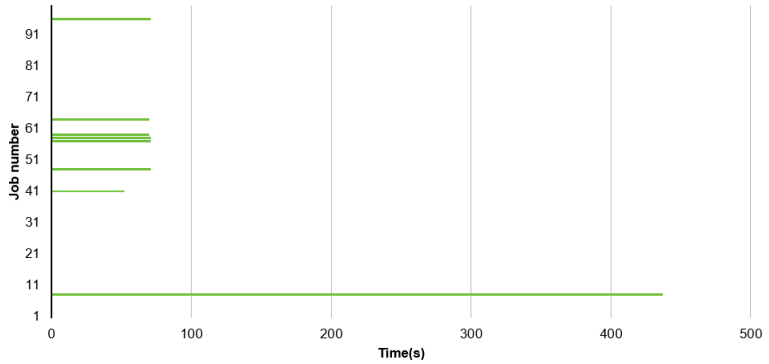


Figure 4.5: Distribution of the starting times of the jobs for Grid submissions (see text).

As it can be seen from Figure 4.5, the vast majority of the jobs are submitted in less than a second.

Furthermore, the Dashboard provides the cumulative plots of the number of events processed over time and the average rate of the event analysed in a second, defined as:

$$R = \frac{\text{total number of events}}{\text{global time for analysis}}$$

Such plots from the Dashboard, for submission to WLCG and to Cloud respectively, are shown in Figure 4.6 and Figure 4.7.

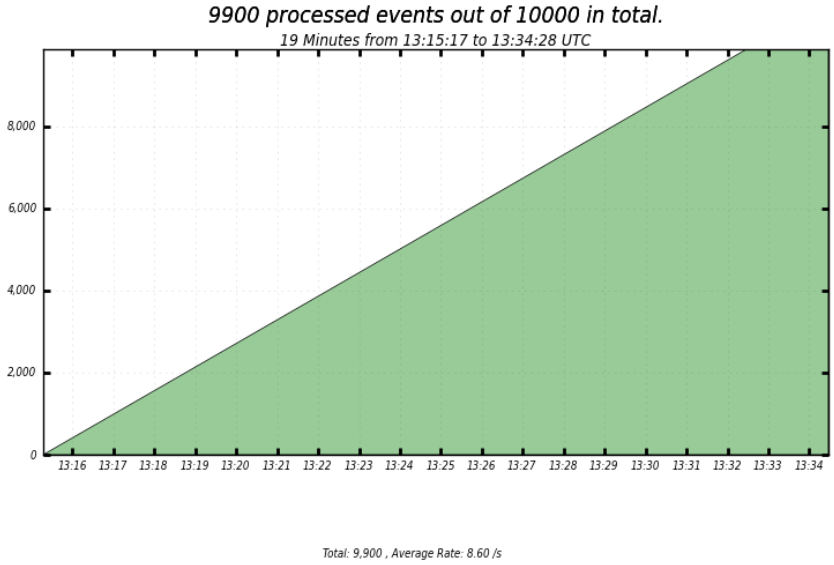


Figure 4.6: Cumulative plot of the number of events processed over time for a sample "light" workflow submitted to the Grid (source: Dashboard).

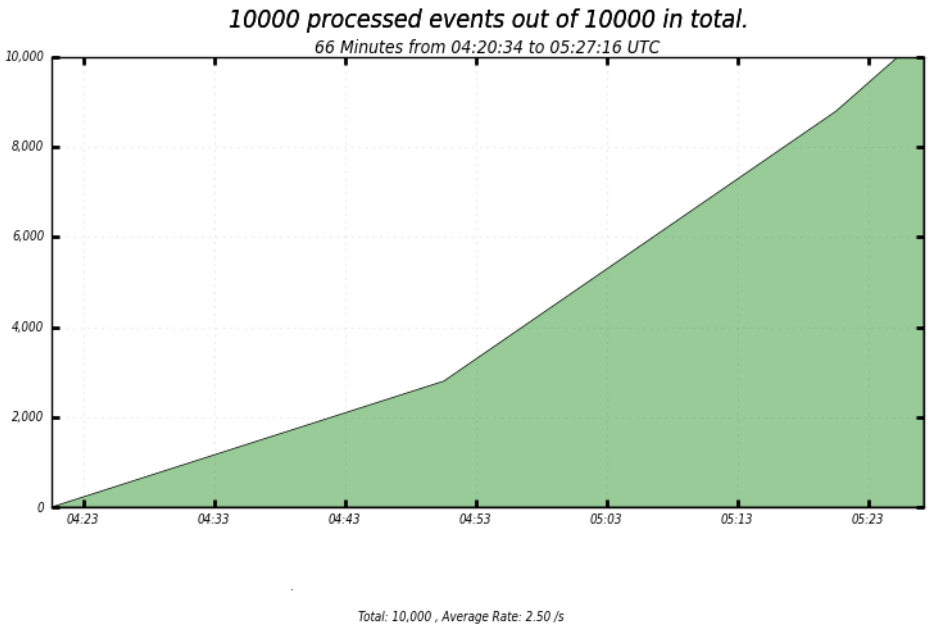


Figure 4.7: Cumulative plot of the number of events processed over time for a sample "light" workflow submitted to the Cloud (source: Dashboard).

As it can be seen at the bottom of each plot, the average rate of processed events is 8.6 events/s for the "light" workflow submitted to WLCG , while it is 2.5 events/s for the "light" workflow submitted to Cloud. This time accounts for the time needed for submitting the jobs added up to the time needed to process all events inside each job. In this sense, these two plots provide a valuable information about the overall time spent in each of the two different workflows.

4.4 Building a "heavier" workflow to investigate CPU efficiencies

The "Heavy" Workflow is a CMSSW analyzer. The analyzer in CMS jargon, is a tool for data elaboration and is implemented as a generic C++ class from which all concrete analyzers inherit. In the present case, the analyzer produces plots of the basic kinematic variables and of the events which satisfy a particular trigger condition. Furthermore, it calculates the invariant masses of dijet and trijet events.

This workflow was created in CRAB2 as a set of 10 tasks of 10 jobs each. These jobs require the availability of a given dataset `/RelValTTbar/CMSSW_5_3_14-START53_LV4_Feb7-v2/GEN-SIM-RECO`. As this dataset was good for the test but it was not adequately spread on the CMS storage systems on the Tier-2 level, an ad-hoc transfer request was made on purpose for this test using PhEDEx, the official CMS data transfer system [31, 33, 33]. The transfer request was placed for 30 Tier-2 sites, and after a few hours from the transfer request, the needed dataset became available on >50% of the selected CMS Tier-2 sites, and the submissions could start. As from the "light" workflow, the "heavy" workflow was also submitted to the Cloud infrastructure, and the results are compared among the two scenario for this workflow, as it was done for the previous one. It must be noted, though, than the relevant observables in this case differ. While in the former case ("light" workflow) we focussed on duration, starting time, etc in this latter case ("heavy" workflow) we will focus on a new set of metrics, as follows:

- **CrabUserCpuTime**: time used by the CPU to perform computation on the user's application submitted with CRAB2.
- **CrabSysCpuTime**: time spent by the CPU to perform system operations
- **ExeTime**: overall job execution time
- **CrabCpuPercentage**: parameter which evaluates the CPU's efficiency as follows:

$$\text{CrabCpuPercentage} = \frac{\text{CrabUserCpuTime} + \text{CrabSysCpuTime}}{\text{ExeTime}}$$

For each of the metrics outlined above, the mean value and the standard deviation have been calculated. The correlation between the ExeTime and the CrabCpuPercentage has been evaluated using the population correlation coefficient:

$$\rho = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{x})^2}}$$

where x and y are the ExeTime and the CrabCpuPercentage in this case.

4.4.1 Analysis of Grid submissions

In this paragraph, the measured values for the selected metrics from the Grid submissions are presented and discussed. For maximum reliability, data are extracted directly from the CRAB2 logs of each jobs. The complete *crab.cfg* for this workflow is in Appendix A.2. The value measured for the CRABCpuUserTime and the CRABSysCpuTime are shown in Figure 4.8 and 4.9 respectively.

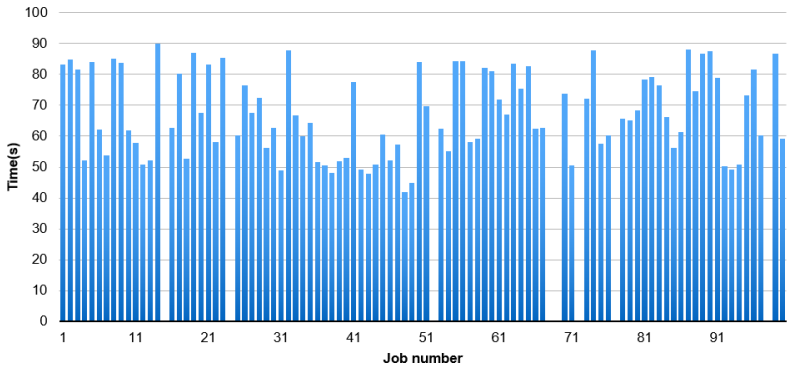


Figure 4.8: CrabUserCpuTime as a function of the job number for the submissions to the Grid of the "heavy" workflow.

The average value for CrabCpuUserTime is:

$$t_{CrabUserCpuTime} = (67 \pm 13)s$$

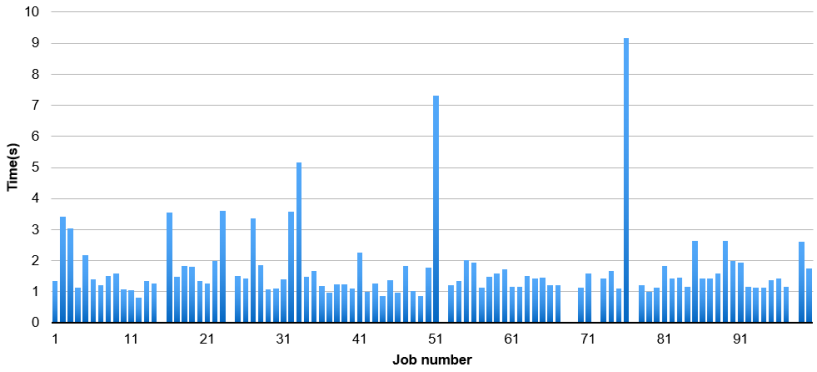


Figure 4.9: CrabSysCpuTime as a function of the job number for the submissions to the Grid of the "heavy" workflow.

The average value for CrabCpuSysTime is:

$$t_{CrabSysCpuTime} = (1.7 \pm 1.2)s$$

The CrabUserCpuTime average value is one order of magnitude greater than the CrabSysCpu time and so it accounts as the major contributor of the CrabCpuPercentage. The value measured for ExeTime is shown in Figure 4.10.

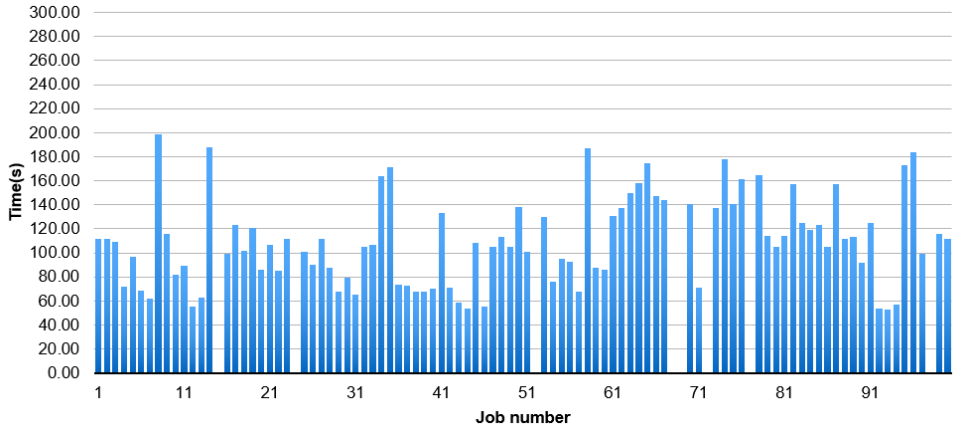


Figure 4.10: ExeTime as a function of the job number for the submissions to the Grid of the "heavy" workflow.

The average value for ExeTime is:

$$t_{ExeTime} = (110 \pm 40)s$$

According to the variables discussed above, the CrabCpuPercentage can be computed, and it is shown in Figure 4.11.

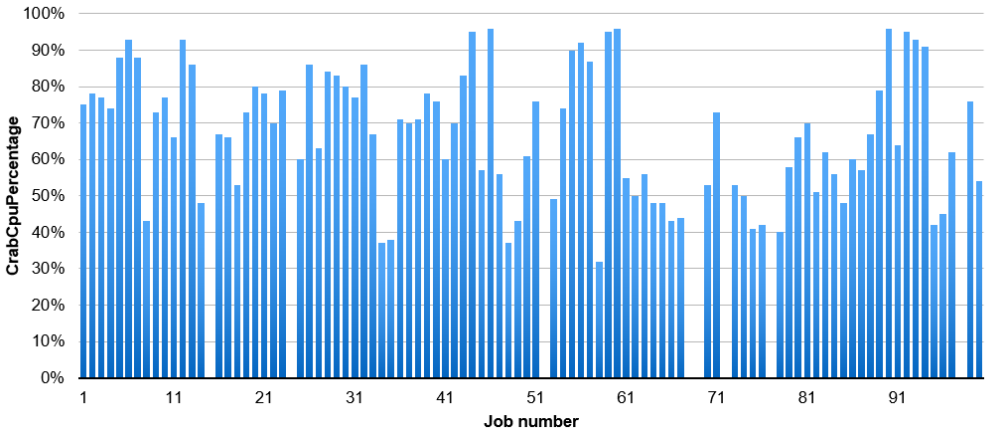


Figure 4.11: CrabCpuPercentage as a function of the job number for the submissions to the Grid of the "heavy" workflow.

The average value for CrabCpuPercentage is:

$$CrabCpuPercentage = (67 \pm 17)\%$$

In the attempt to group jobs exploiting similar CPU efficiency and seek for trends, the collected data have been grouped in CPU efficiency intervals and plotted in Figure 4.12 (each bin corresponds to a 10% CPU efficiency window).

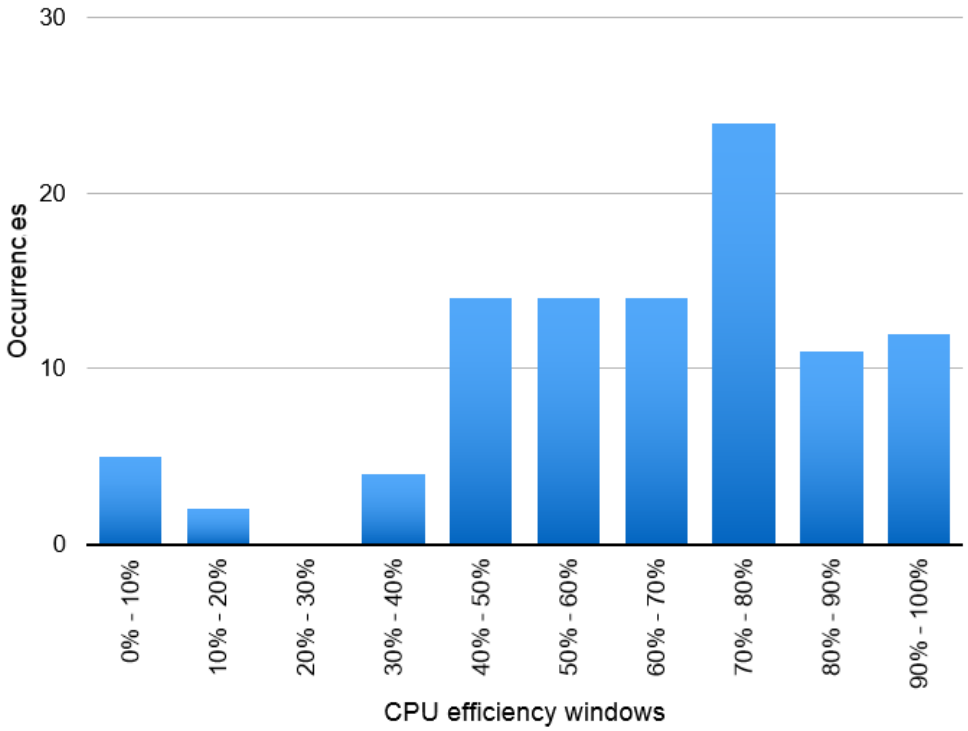


Figure 4.12: Occurrences of CrabCpuPercentage grouped in intervals (each bin corresponds to a 10% CPU efficiency window).

The distribution shows that the majority of the jobs submitted to WCG sites show a CPU efficiency that is greater than 40%, with a distribution of values quite spread from 40% to 100%, and a peak around $70 - 80\%$ which is visible despite the relatively low statistics.

The CrabCpuPercentage values can also be plotted as a function of the ExeTime for each job. This is shown in Fig 4.13.

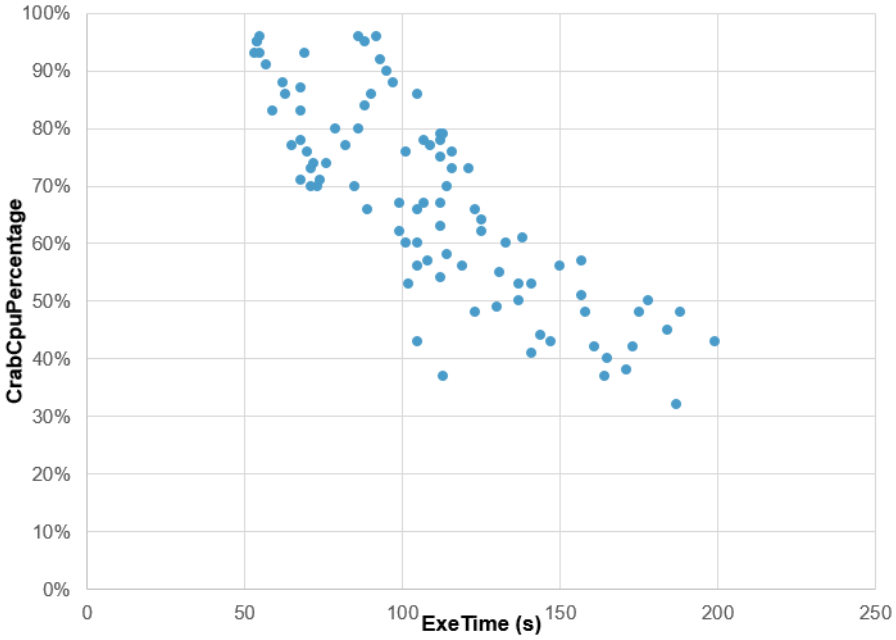


Figure 4.13: Graph which shows the ExeTime in function of the CrabCpuPercentage for the Grid infrastructure.

The population correlation coefficient has been calculated:

$$\rho = -0.81471$$

4.4.2 Analysis of Cloud submissions

The complete *crab.cfg* for this workflow is in Appendix A.2. The value measured for the CrabUserCpuTime and the CrabSysCpuTime are shown in Figure 4.14 and Figure 4.15 respectively.

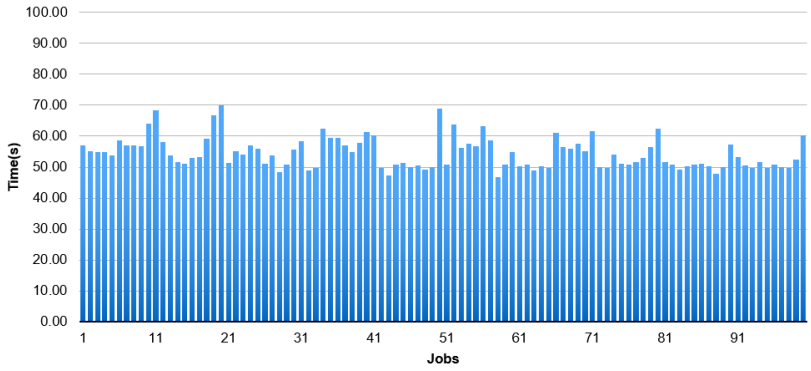


Figure 4.14: CrabUserCpuTime as a function of the job number for the submissions to the WLCG of the "heavy" workflow.

The average value for CrabUserCpuTime is:

$$t_{CrabUserCpuTime} = (54 \pm 5)s$$

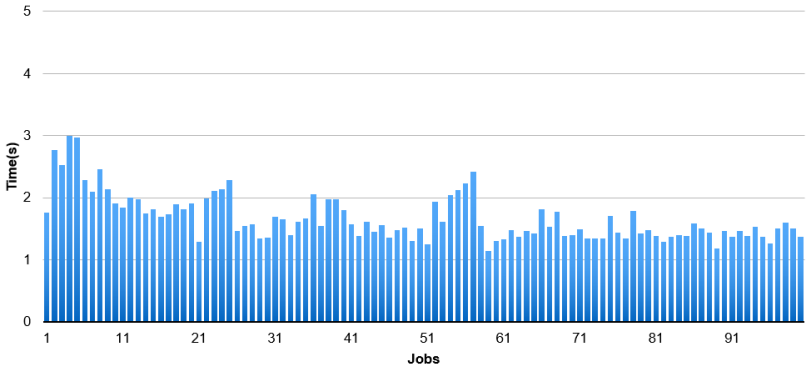


Figure 4.15: CrabSysCpuTime as a function of the job number for the submissions to the WLCG of the "heavy" workflow.

The average value for CrabSysCpuTime is:

$$t_{CrabSysCpuTime} = (1.6 \pm 0.3)s$$

The value measured for ExeTime is shown in Figure 4.16.

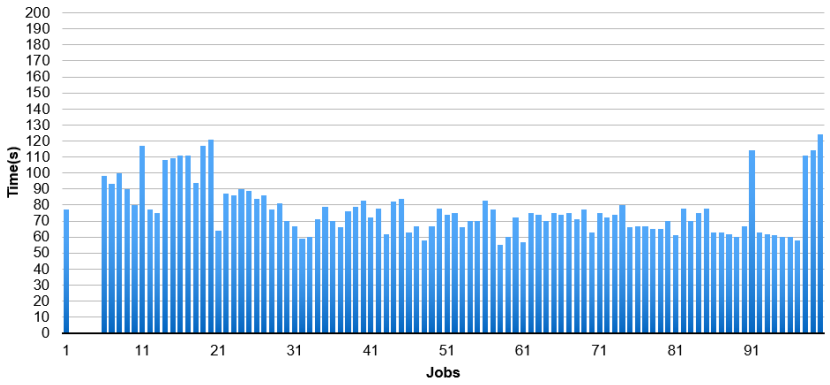


Figure 4.16: ExeTime as a function of the job number for the submissions to the WLCG of the "heavy" workflow.

The average value for ExeTime is:

$$t_{ExeTime} = (77 \pm 16)s$$

According to the variables discussed above, the CrabCpuPercentage can be computed, and it is shown in Figure 4.17.

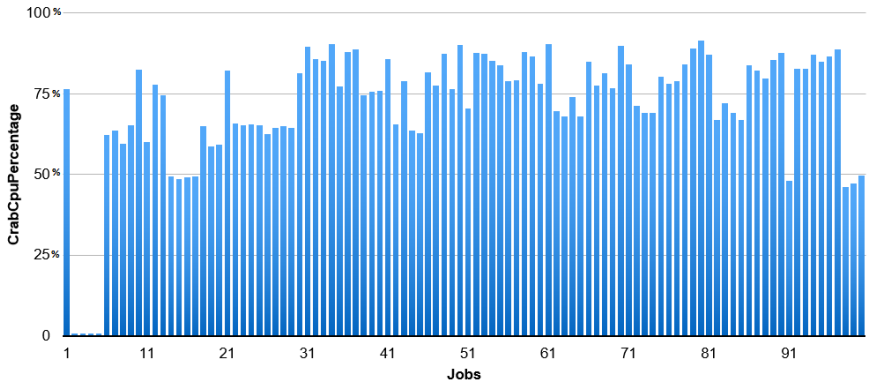


Figure 4.17: CrabCpuPercentage as a function of the job number for the submissions to the WLCG of the "heavy" workflow.

The average value for CrabCpuPercentage is:

$$CrabCpuPercentage = (75 \pm 12)\%$$

In the attempt to group jobs exploiting similar CPU efficiency and seek for trends, the collected data have been grouped in CPU efficiency intervals and plotted in Figure 4.18 (each bin corresponds to a 10% CPU efficiency window).

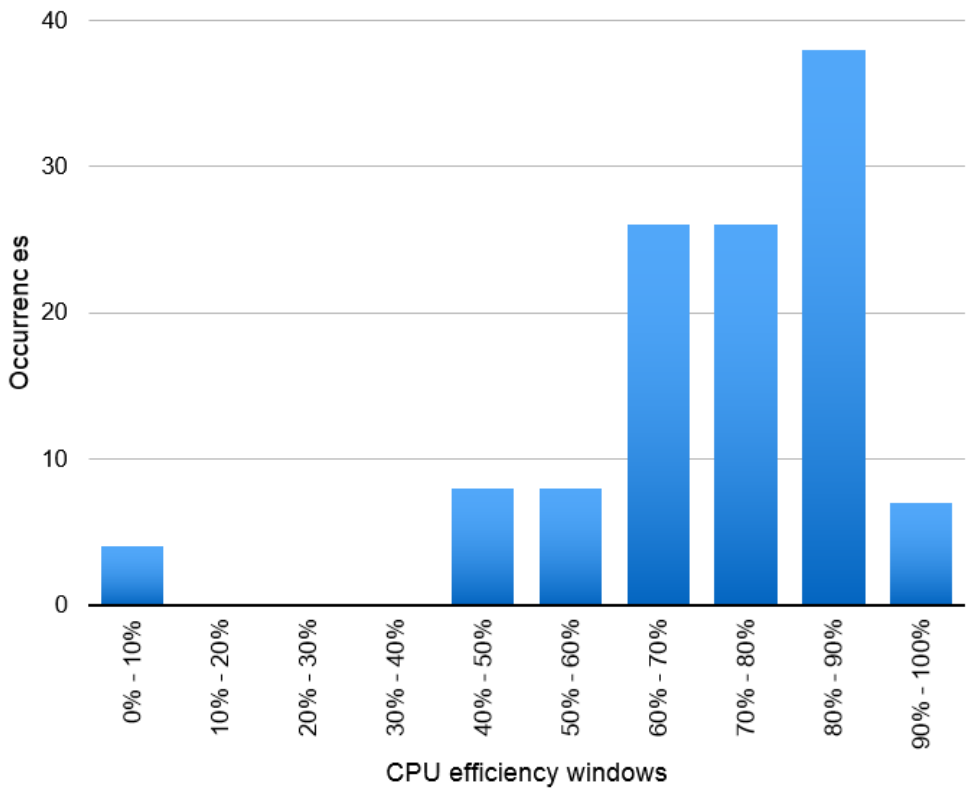


Figure 4.18: Occurrences of CrabCpuPercentage grouped in intervals (each bin corresponds to a 10% CPU efficiency window).

The CrabCpuPercentage values can also be plotted as a function of the ExeTime for each job. This is shown in Fig 4.19.

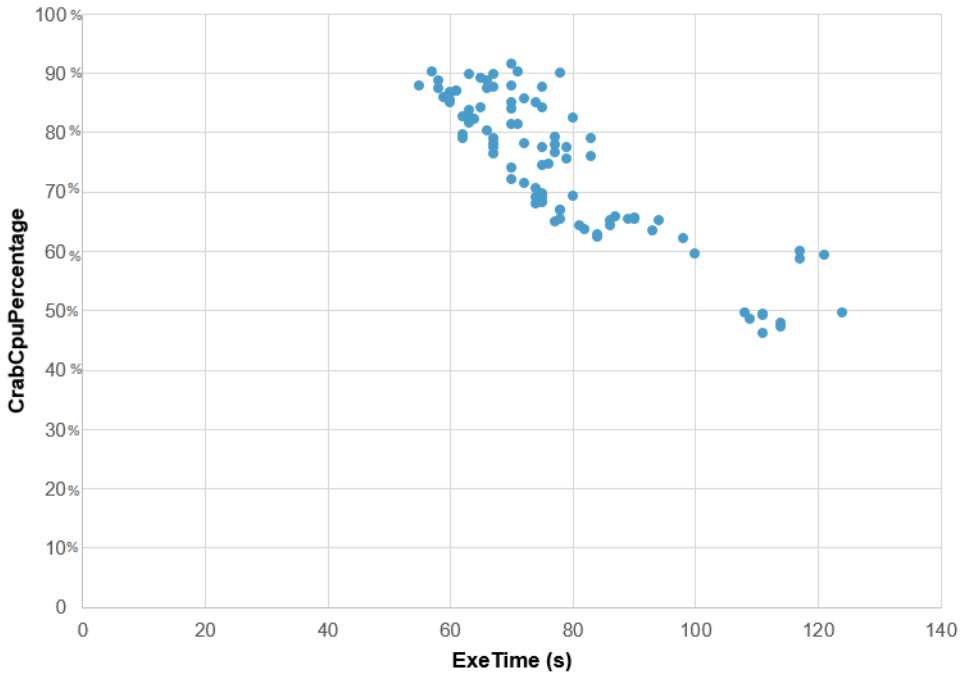


Figure 4.19: Graph which shows the ExeTime in function of the CrabCpuPercentage for the Cloud infrastructure.

The population correlation coefficient has been calculated:

$$\rho = -0.87477$$

4.4.3 Grid versus Cloud performance comparison

In this paragraph, a comparison of the performances measured in the Grid versus the Cloud submissions is presented and discussed. Table 4.2 summarizes the average values of the metrics evaluated.

	Grid	Cloud
CrabUserCpuTime	$(67 \pm 13)s$	$(54 \pm 5)s$
CrabSysCpuTime	$(1.7 \pm 1.2)s$	$(1.6 \pm 0.3)s$
ExeTime	$(110 \pm 40)s$	$(77 \pm 16)s$
CrabCpuPercentage	$(67 \pm 17)\%$	$(75 \pm 12)\%$

Table 4.2: Comparison of the performances of Cloud and Grid for the workflow "heavy"

In a nutshell, no major deviation from a generally good behaviour and acceptable performance figures is observed in any of the two scenario under study. In terms of the metrics chosen to explore and compare them - the submissions through Cloud interfaces look comparable in performances to the submissions done to a general WLCG Grid infrastructure. On average, higher values of the variable CrabCpuPercentage can be observed on the Cloud. Within the limited scope of this test it is not straightforward to draw solid conclusions; on the other hand, a possible explanation for this may be that the Cloud infrastructure under study consists to a very limited and "clean" environment i.e. the AI resources at CERN. These, in general, may be more reliable as a computing resource than a full and open set of Grid sites worldwide accessible on the production infrastructure, where the performance of each site may vary from one to the other. This fact is also attested by the large standard deviation of the ExeTime variable for the submissions to the Grid, which is almost a factor of 3 marker than the submissions to the Cloud infrastructure. It is remarkable, though, that - despite the limited scale of the test - it was anyway observed and measured that accessing the AI infrastructure through Cloud interfaces via a non-production glideInWMS system is not causing a decrease of performance in any of the metrics chose to quantitatively analyse the system performances.

4.5 Running a "real" workflow and compare Grid versus Cloud

As outlined in the introduction of this Chapter, it was considered of relevance to prepare and submit a third workflow in addition to the "light" and "heavy" workflows, which stands as an example of a "real" workflow of relevance for the CMS physics program. In some more detail, this "real" workflow executes two mains tasks:

- it skims the data of a multijet sample to a small fraction of event of physical interest, for a complete study on the channel of the fully hadronic top quark;
- it converts the format from the CMS data format in a format which is easier to be accessed and analysed by the user.

Overall, it is an example of a typical analysis workflow in particle physics. Firstly, the Physics Analysis Toolkit (PAT) is used, which exploits the CMSSW tools to gather information in collections that will exemplify the analysis. Secondly, the collections which are not necessary to the user are removed and the events are filtered according to the same criteria. In this way, PAT also reduces the size of the data that the user elaborates in his final analysis steps. Thirdly, the data are converted in a format which can be easily analysed, getting rid of more unnecessary information. At this point the user can perform his analysis through ROOT (the data analysis framework) in his local resources.

In conclusion, this workflow reduces significantly the storage space required for the final analysis. This process is so efficient that very often the user can make his final analysis and his plots directly on his laptop.

This workflow was created in CRAB2 as a unique task of 1271 jobs. These jobs requires the availability of a given dataset `/TTJets_MSDecays_central_TuneZ2star_8TeV-madgraph-tauola/Summer12_DR53X-PU_S10_START53_V19-v1/AODSIM`, which - due to its overall size (approximately 25TB) - was available in a few Grid sites. After checking which sites they were, and their good records of site availability over the past few months, it was concluded that they were enough in number and reliable enough to proceed with the jobs preparation and submission, without further data replication elsewhere. On the contrary, for the Cloud submission round, two different choices were evaluated. The first choice was to configure CRAB2 to ignore data locality, send jobs to AI and let the system deal with remote data access. The second choice was to transfer the input data to the CERN Tier-2 site (namely: T2_CH_CERN in CMS jargon), whose storage is co-accessible also by the T2_CH_CERN_AI node. As the first choice had technical difficulties, and would have anyway added a latency due to the remote data access, the second option was chosen. An ad-hoc transfer request was placed on purpose for this test using PhEDEx [31, 33, 33].

After approximately 28 hours from the transfer request (see Figure 4.20), the dataset became available on T2_CH_CERN in almost its entirety, hence the submissions could start.

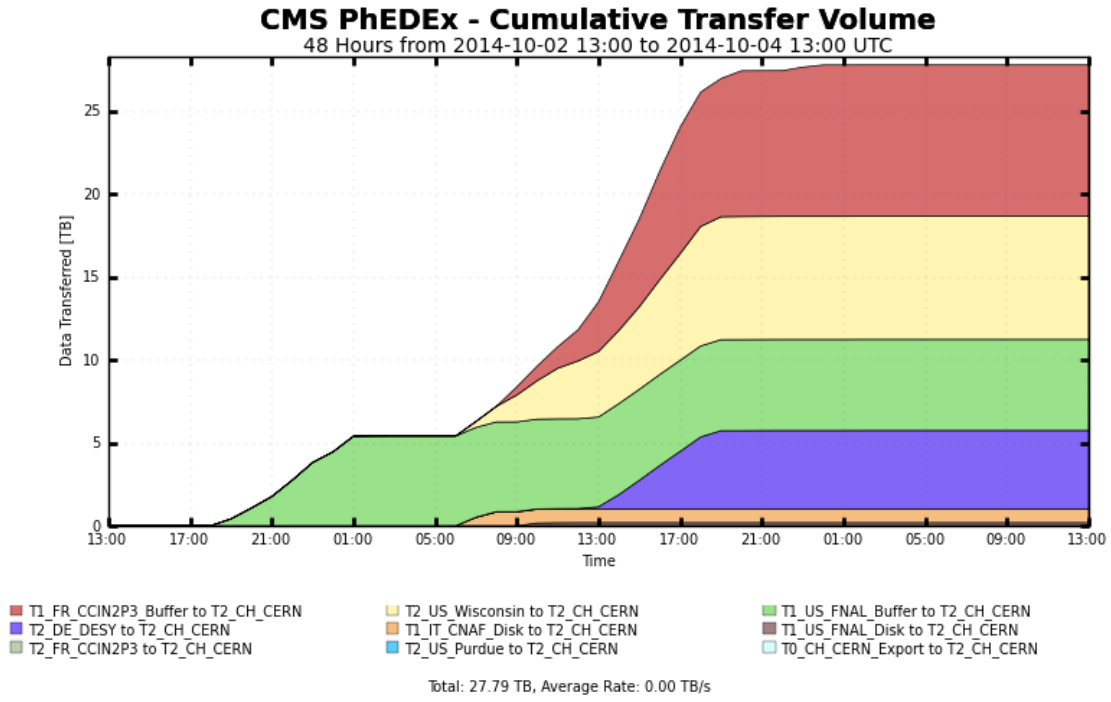


Figure 4.20: Cumulative transfer volume to the CERN Tier-2 site in response to a transfer request submitted for this thesis. Each color corresponds to a different source site (Source PhEDEx).

Not all the blocks of that dataset were actually 100% available, though, so only a fraction of the jobs could be created with CRAB2 (roughly, only 85% of the jobs) and submitted to the Grid. Subsequently, when also the rest of the dataset transfer was complete, all jobs were finally created and an additional submission took place. Given the amount of jobs in the task, as this is the only workflow for which it is not viable to opt for a high multiplicity of submissions, in order to make sure that the current analysis is not biased on a specific submission round, we at least tried it twice: we report results only for one, but the results for the other one are comparable in figures, and equivalent in conclusions.

Concerning the "Cloud" submission, a technical limitation arose. The same workflow which was submitted to Grid, was also created and submitted on AI resources at CERN via the ad-hoc glideIn-WMS instance. Due to overlapping CMS activities on the AI

resources, the system administrators had to reduce the amount of available machines, allowing only a maximum of 20 concurrent jobs at any given time, thus leaving all the other pending jobs in a long waiting queue. As the completion time for the complete workflow (more than 1200 jobs) would have been too long, it was decided to stop the running workflow when a total number of "terminated" jobs reached a couple of hundreds jobs (namely, submissions were stopped at 200 jobs). The subsequent analysis was hence done on the outcome of these jobs only, but their number is still reasonably high to allow comparisons that are statistically correct.

4.5.1 Analysis of Grid submissions

The complete *crab.cfg* for this workflow is in Appendix A.3. The value measured for the CrabUserCpuTime and the CrabSysCpuTime are shown in Figure 4.21 and Figure 4.22 respectively.

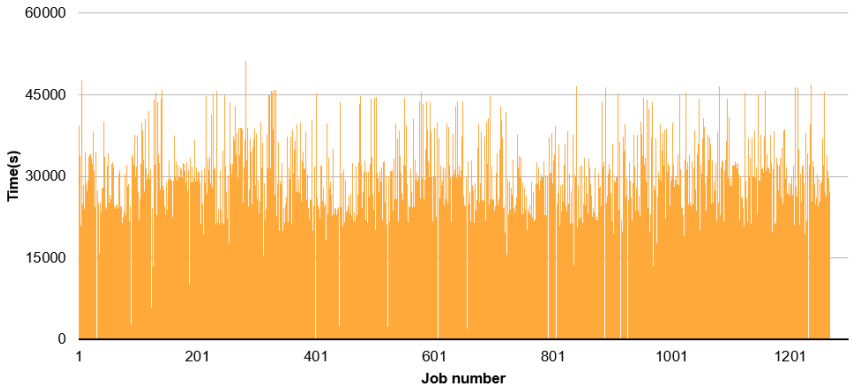


Figure 4.21: CrabUserCpuTime as a function of the job number for the submissions to the Grid of the "real" workflow.

The average value for CrabUserCpuTime is:

$$t_{CrabUserCpuTime} = (2.8 \pm 0.8) \times 10^4 s$$

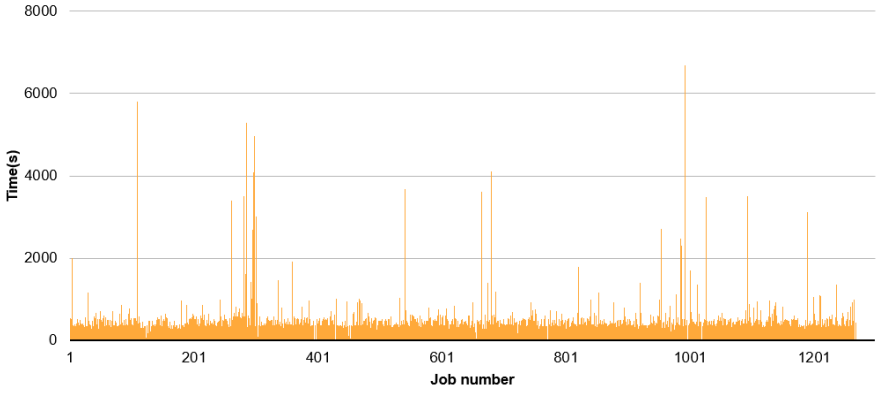


Figure 4.22: CrabSysCpuTime as a function of the job number for the submissions to the Grid of the "real" workflow.

The average value for CrabSysCpuTime is:

$$t_{CrabSysCpuTime} = (5 \pm 2) \times 10^2 s$$

The value measured for ExeTime is shown in Figure 4.23.

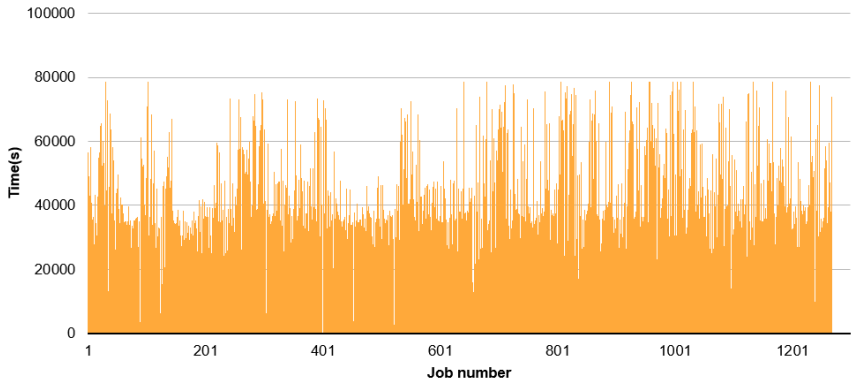


Figure 4.23: ExeTime as a function of the job number for the submissions to the Grid of the "real" workflow.

The average value for ExeTime is:

$$t_{ExeTime} = (4.2 \pm 1.4) \times 10^4 s$$

According to the variables discussed above, the CrabCpuPercentage can be computed, and it is shown in Figure 4.24.

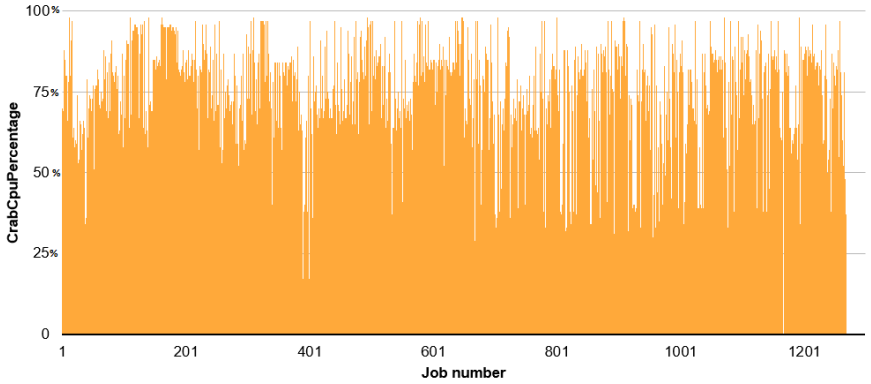


Figure 4.24: CrabCpuPercentage as a function of the job number for the submissions to the Grid of the "real" workflow.

The average value for CrabCpuPercentage is:

$$CrabCpuPercentage = (72 \pm 17)\%$$

In the attempt to group jobs exploiting similar CPU efficiency and seek for trends, the collected data have been grouped in CPU efficiency intervals and plotted in Figure 4.25 (each bin corresponds to a 10% CPU efficiency window).

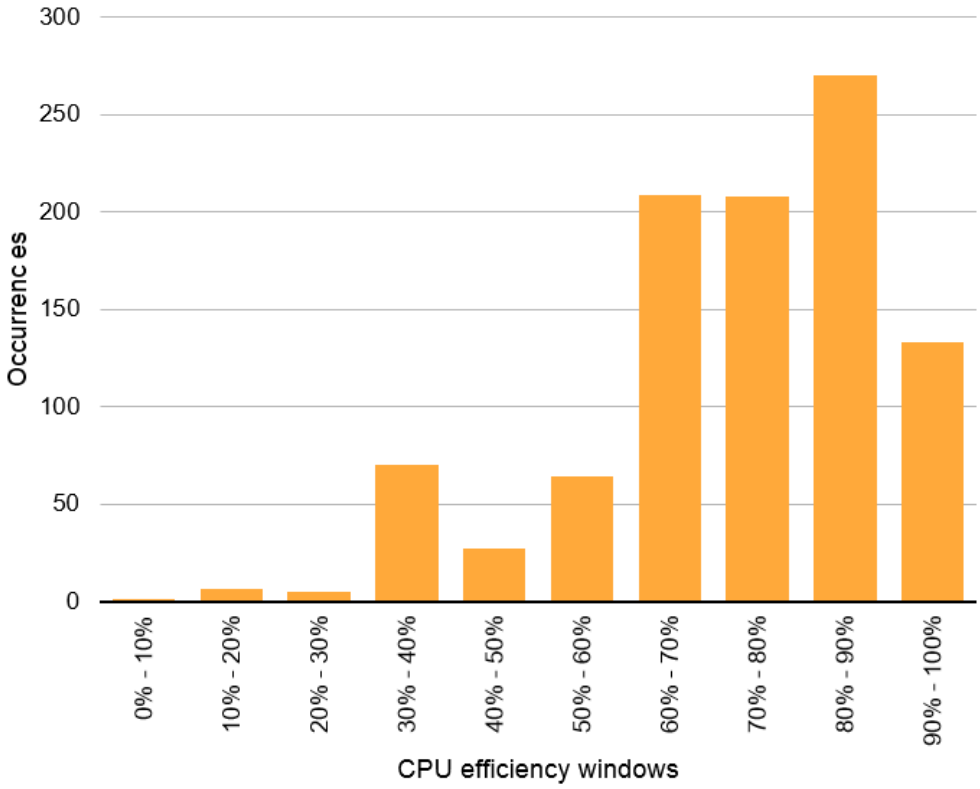


Figure 4.25: Occurrences of CrabCpuPercentage grouped in intervals (each bin corresponds to a 10% CPU efficiency window).

The CrabCpuPercentage values can also be plotted as a function of the ExeTime for each job. This is shown in Fig 4.26.

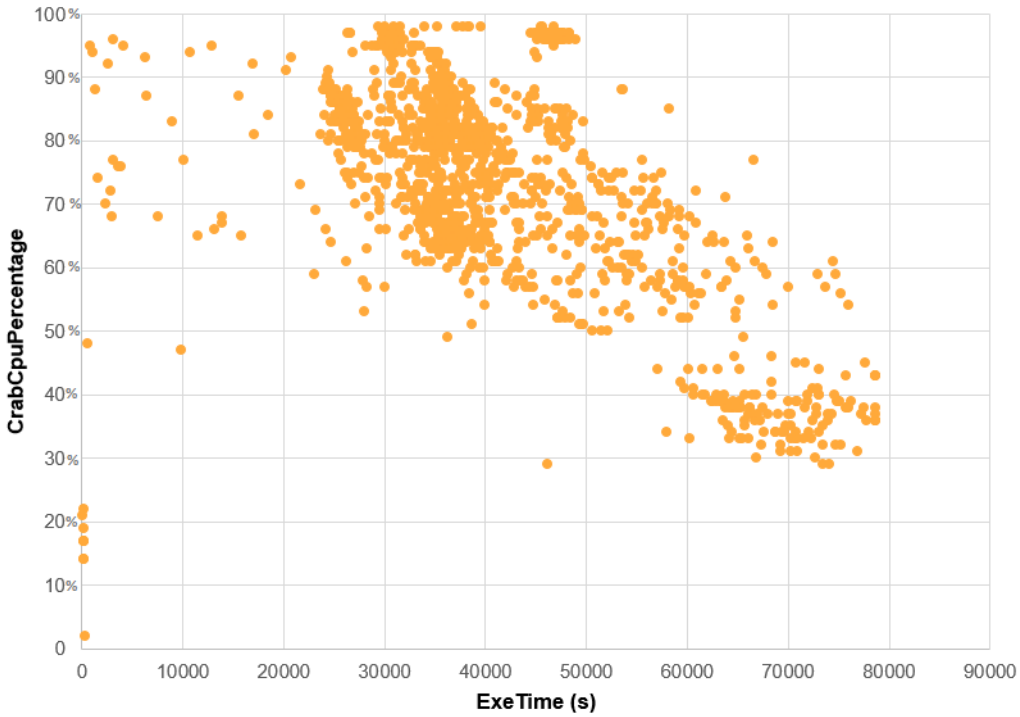


Figure 4.26: Graph which shows the ExeTime in function of the CrabCpuPercentage for the Grid infrastructure.

The population correlation coefficient has been calculated:

$$\rho = -0.56222$$

4.5.2 Analysis of Cloud submissions

The complete *crab.cfg* for this workflow is in Appendix A.3. The value measured for the CrabUserCpuTime and the CrabSysCpuTime are shown in Figure 4.27 and Figure 4.28 respectively.

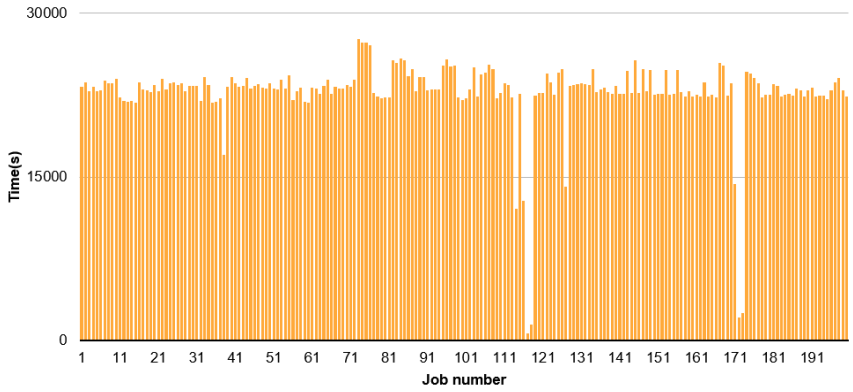


Figure 4.27: CrabUserCpuTime as a function of the job number for the submissions to WLCG of the "real" workflow.

The average value for CrabUserCpuTime is:

$$t_{CrabUserCpuTime} = (2.2 \pm 0.4) \times 10^4 s$$

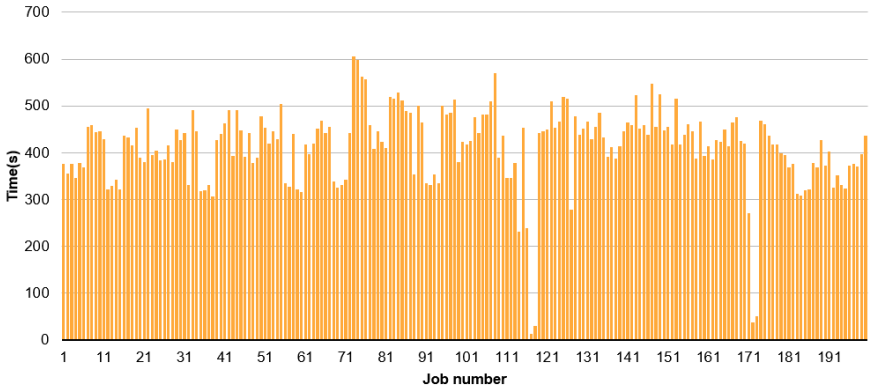


Figure 4.28: CrabSysCpuTime as a function of the job number for the submissions to WLCG of the "real" workflow.

The average value for CrabSysCpuTime is:

$$t_{CrabSysCpuTime} = (4.1 \pm 0.8) \times 10^2 s$$

The value measured for ExeTime is shown in Figure 4.29.

Figure 4.29: ExeTime as a function of the job number for the submissions to WLCG of the "real" workflow.

The average value for ExeTime is:

$$t_{ExeTime} = (2.4 \pm 0.4) \times 10^4 s$$

According to the variables discussed above, the CrabCpuPercentage can be computed, and it is shown in Figure 4.30.

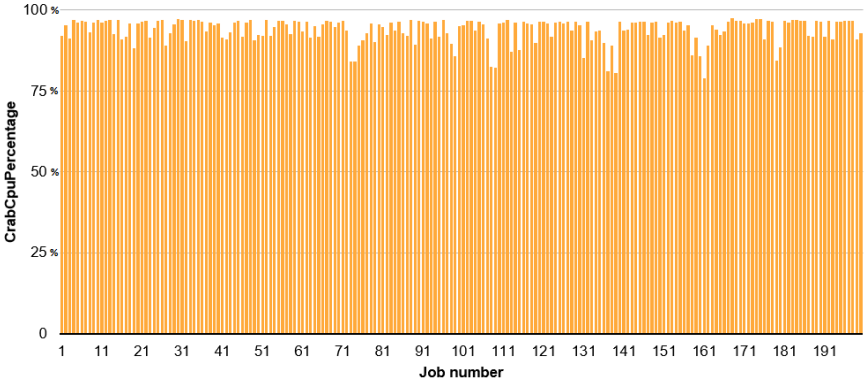


Figure 4.30: CrabCpuPercentage as a function of the job number for the submissions to WLCG of the "real" workflow.

The average value for CrabCpuPercentage is:

$$CrabCpuPercentage = (94 \pm 4)\%$$

In the attempt to group jobs exploiting similar CPU efficiency and seek for trends, the collected data have been grouped in CPU efficiency intervals and plotted in Figure 4.31 (each bin corresponds to a 10% CPU efficiency window).

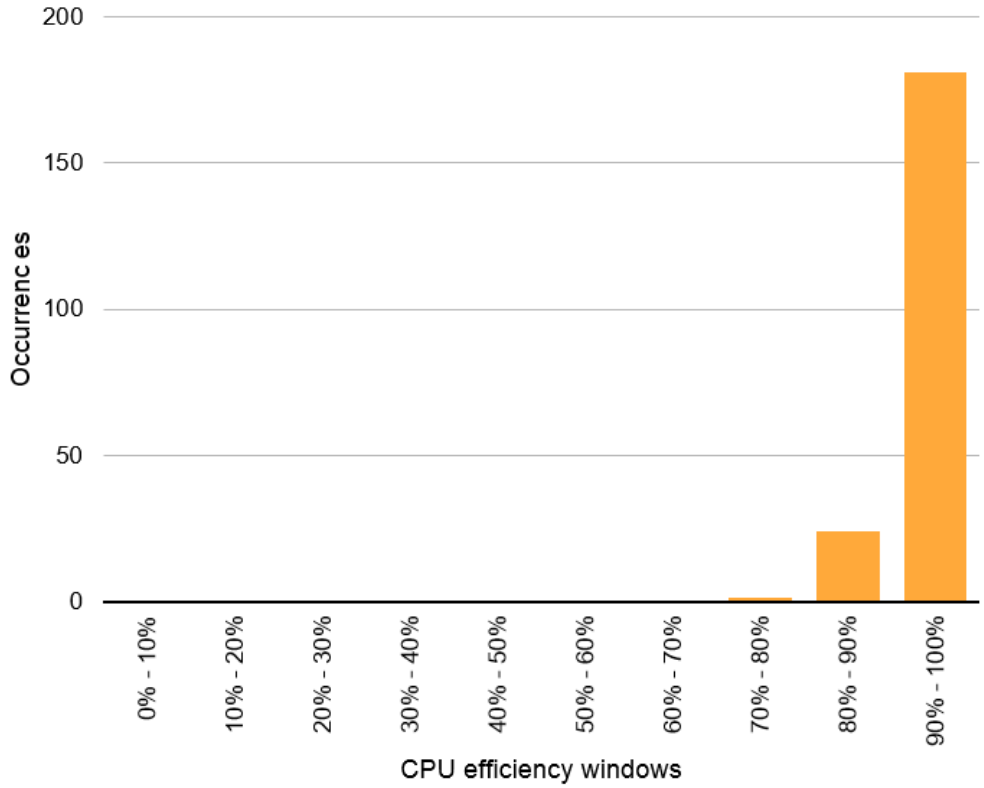


Figure 4.31: Occurrences of CrabCpuPercentage grouped in intervals (each bin corresponds to a 10% CPU efficiency window).

The CrabCpuPercentage values can also be plotted as a function of the ExeTime for each job. This is shown in Fig 4.32.

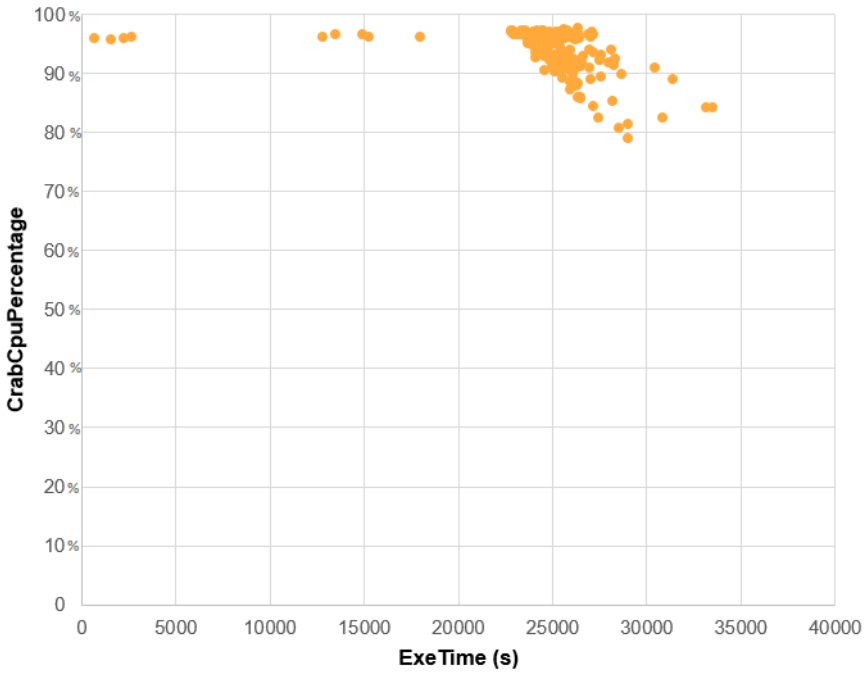


Figure 4.32: Graph which shows the ExeTime in function of the CrabCpuPercentage for the Cloud infrastructure.

The population correlation coefficient has been calculated:

$$\rho = -0.38174$$

4.5.3 Grid versus Cloud performance comparison

The "real" workflow offers a sample of jobs which runs for a considerably larger amount of time compared to the "heavy" workflow. Hence, the majority of the CPU time is spent in doing actual calculations on the events for the analysis. This implies that the CrabCpuPercentage should be higher than that of the "heavy" workflow. This is exactly what we observe. Table 4.3 summarises the measured average values of the chosen metrics for the "real" workflow.

	Grid	Cloud
CrabUserCpuTime	$(2.8 \pm 0.8) \times 10^4 s$	$(2.2 \pm 0.4) \times 10^4 s$
CrabSysCpuTime	$(5 \pm 5) \times 10^2 s$	$(4.1 \pm 0.8) \times 10^2 s$
ExeTime	$(4.2 \pm 1.4) \times 10^4 s$	$(2.4 \pm 0.4) \times 10^4 s$
CrabCpuPercentage	$(72 \pm 17)\%$	$(94 \pm 4)\%$

Table 4.3: Comparison of the performance of Cloud and Grid for the workflow "real".

The relative fractions of jobs which ended up in running on different WLCG sites for the "real" workflow are shown in Figure 4.33.

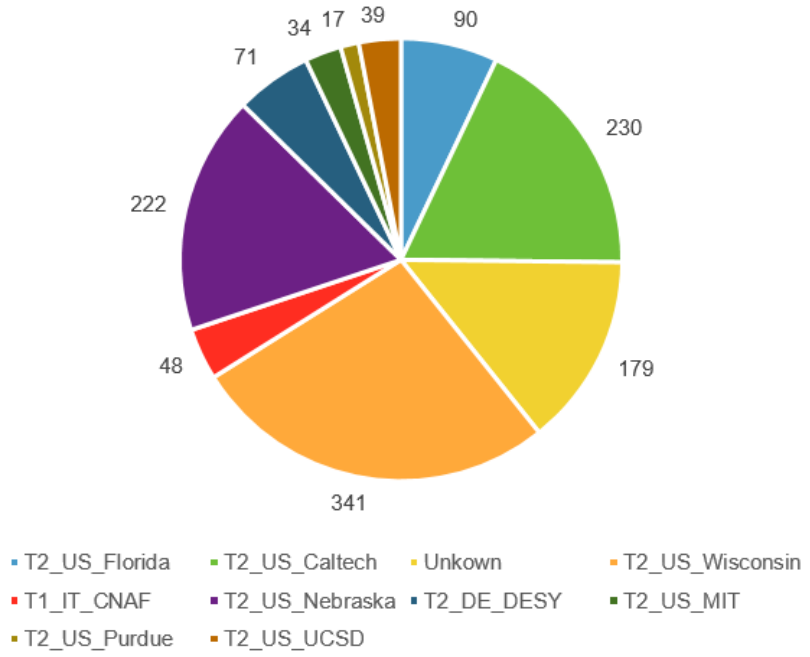


Figure 4.33: Breakdown of the job submission of the "real" workflow into different WLCG sites. A total of 9 sites (at least) were used. The jobs tagged with "unknown" are jobs whose metadata are lost by the Dashboard monitoring (see text for details).

The jobs ran on at least 9 different Grid sites. A fraction of the jobs ended up in running on a Grid site whose name was not properly reported back to the Dashboard infrastructure and exposed in the task monitoring tool: for these jobs, we may assume that they ran also on potentially different sites, thus increasing the total number of sites reached by this submission, but it has no net effect on the analysis and so we did not investigate this feature deeper.

Chapter 5

Conclusions

Cloud computing is emerging as a new paradigm to access large sets of shared resources for many scientific communities. In this thesis I report the original results of my Grid versus Cloud comparative analysis for some representative workflows of the CMS experiment at the LHC.

Three distinct workflows have been identified and investigated, with different goals. In the first one, the goal was to test the basic functionalities. It was observed that both the Cloud interface and the Grid production environment expose similar interfaces with a comparable ease of use and similar performances. Furthermore, the measured rates of successful jobs on Grid and Cloud are $R_{Grid} = 98.9\%$ and $R_{Cloud} = 100\%$ respectively, thus showing comparable reliability levels. The second workflow was designed to perform a heavier use of CPU cycles in order to evaluate and compare other metrics, e.g. the CPU efficiencies on Grid and Cloud. In the clean environment and controlled set-up used in this work, the Cloud did not show any drawbacks with respect to the production LHC Grid, whereas the overall Cloud performances, at least within the collected statistics, seems to be even better. The third workflow consists of a real analysis task in the context of the fully hadronic top physics. It is remarkable that, despite the complexity of this real analysis task, we have obtained the same results as from the test workflows, thus indicating that Cloud resources can be used for real CMS analysis.

Despite no general conclusions should be drawn on tests with the current statistics of submitted jobs, and although additional work would be needed and tests at higher scale would be beneficial, we observe that a Cloud infrastructure may offer a computing environment with functionalities and performance figures comparable to those offered by the LHC Computing Grid services in production since many years.

Appendix A

A.1 CRAB configuration file for the "light" workflow

The CRAB configuration file for the "light" workflow is reported below. In the submission to WLCG, the last lines of the cfg file must be commented (see below).

```
[CMSSW]
allow_NonProductionCMSSW  = 1
total_number_of_events    = 10000
number_of_jobs            = 100
pset                      = WorkflowLIGHT_configuration.py
datasetpath               = /GenericTTbar/HC-CMSSW_5_3_1_START53_V
                            5-v1/GEN-SIM-RECO
output_file               = outfile.root

[USER]
return_data               = 0
copy_data                 = 1
storage_element           = T2_IT_Legnaro
user_remote_dir           = LucaAmbroz_LIGHT

[CRAB]
scheduler                 = remoteGlidein
jobtype                   = cmssw
submit_host               = lnl_submit-6 # only for Cloud

[GRID]
se_white_list             = T2_CH_CERN_AI # only for Cloud
max_rss                   = 1900 # only for Cloud
```

A.2 CRAB configuration file for the "heavy" workflow

The CRAB configuration file for the "heavy" workflow is reported below. In the submission to WLCG, the last lines of the cfg file must be commented (see below).

```
[CMSSW]
allow_NonProductionCMSSW  = 1
total_number_of_events    = 10000
number_of_jobs            = 10
pset                      = WorkflowHEAVIER_configuration.py
datasetpath               = /RelValTTbar/CMSSW_5_3_14-START
                            53_LV4_Feb7-v2/GEN-SIM-RECO
output_file               = outfile.root

[USER]
return_data               = 0
copy_data                 = 1
storage_element           = T2_IT_Legnaro
user_remote_dir           = LucaAmbroz_HEAVIER

[CRAB]
scheduler                 = remoteGlidein
jobtype                   = cmssw
submit_host               = lnl_submit-6 # only for Cloud

[GRID]
se_white_list             = T2_CH_CERN_AI # only for Cloud
max_rss                   = 1900 # only for Cloud
```

A.3 CRAB configuration file for the "real" workflow

The CRAB configuration file for the "real" workflow is reported below. In the submission to WLCG, the last lines of the cfg file must be commented (see below).

```
[CMSSW]
allow_NonProductionCMSSW  = 1
pset                      = WorkflowTOP_configuration.py
datasetpath               = /TTJets_MSDecays_central_TuneZ2star_
                            8TeV-madgraph-tauola/Summer12_DR53X-
                            PU_S10_START53_V19-v1/AODSIM
total_number_of_events    = -1
events_per_job            = 50000

[USER]
user_remote_dir           = LucaAmbroz_TOP
copy_data                 = 1
return_data               = 0
publish_data              = 0
storage_element           = T3_IT_Bologna

[CRAB]
jobtype                   = cmssw
scheduler                 = remoteGlidein
use_server                = 0
submit_host               = lnl_submit-6 # only for Cloud

[GRID]
se_white_list             = T2_CH_CERN_AI # only for Cloud
max_rss                   = 1900 # only for Cloud
```

Bibliography

[1] The Large Hadron Collider, http://home.web.cern.ch/topics/large-hadron-collider

[2] "*The CERN Large Hadron Collider*", http://jinst.sissa.it/LHC/, IOP/Sissa

[3] R. Aßmann, M. Lamont, S. Myers, "*A Brief History of the LEP Collider*", Nucl. Phys. B, Proc. Suppl. 109 (2002) 17-31

[4] S. Chatrchyan *et al.* [CMS Collaboration], "*Observation of a new boson at a mass of 125 GeV with the CMS experiment at the LHC*", Phys. Lett. B **716** (2012) 30

[5] S. Chatrchyan *et al.* [CMS Collaboration], "*A New Boson with a Mass of 125 GeV Observed with the CMS Experiment at the Large Hadron Collider*", Science **338** (2012) 1569.

[6] ALICE experiment web page, http://aliceinfo.cern.ch/Public/Welcome.html

[7] ATLAS experiment web page, http://www.atlas.ch/

[8] CMS Collaboration, "*The CMS experiment at the CERN LHC*", JINST **3** S08004 (2008)

[9] CMS experiment web page, http://cms.web.cern.ch/

[10] LHCb experiment web page, http://lhcb-public.web.cern.ch/lhcb-public/

[11] J. D. Shiers, "*The Worldwide LHC Computing Grid (worldwide LCG)*", Computer Physics Communications 177 (2007) 219–223

[12] WLCG, http://lcg.web.cern.ch/lcg/

[13] European Grid Infrastructure, http://www.egi.eu/

[14] Open Science Grid, http://www.opensciencegrid.org

[15] Virtual Organization Membership Service, http://toolkit.globus.org/grid_software/security/voms.php

[16] D. Bonacorsi, "WLCG Service Challenges and Tiered architecture in the LHC era", IFAE, Pavia, April 2006

[17] CMS Collaboration, "The CMS Computing Model", CERN LHCC 2004-035

[18] CMS Collaboration, "The CMS Computing Project Technical Design Report", CERN-LHCC-2005-023

[19] G Bauer et al., "*The data-acquisition system of the CMS experiment at the LHC*", Journal of Physics, Conference Series 331 (2011) 02202

[20] P. Mell, T. Grance, National Institute of Standards and Technology U.S. Department of Commerce "*The NIST Definition of Cloud Computing*" Special Publication 800-145

[21] High Level Trigger, http://lhcb-trig.web.cern.ch/lhcb-trig/HLT/

[22] Ramn Medrano Llamas et al, "*Commissioning the CERN IT Agile Infrastructure with experiment workloads*", 2014 J. Phys.: Conf. Ser. 513 032066

[23] Amazon Elastic Compute Cloud, http://aws.amazon.com/ec2/

[24] OpenStack, http://www.openstack.org/

[25] OZ, https://github.com/clalancette/oz/wiki

[26] CVMFS, http://cernvm.cern.ch/portal/filesystem

[27] GlideinWMS, http://www.uscms.org/SoftwareComputing/Grid/WMS/glideinWMS/doc.prd/index.html

[28] CRAB online manual http://cmsdoc.cern.ch/cms/ccs/wm/www/Crab/Docs/crab-online-manual.html

[29] D. Spiga et al., "*The CMS Remote Analysis Builder (CRAB)*", Lect. Notes Comput. Sci. 4873 580-586 (2007)

[30] The Dashboard project, http://dashboard.cern.ch

[31] T. Barrass et al, "*Software agents in data and workflow management*", Proc. CHEP04, Interlaken, 2004. See also http://www.pa.org

[32] D. Bonacorsi, T. Barrass, J. Hernandez, J. Rehn, L. Tuura, J. Wu, I. Semeniouk, "*PhEDEx high-throughput data transfer management system*", CHEP06, Computing in High Energy and Nuclear Physics, T.I.F.R. Bombay, India, February 2006

[33] L. Tuura et al., "*Scaling CMS data transfer system for LHC start-up*", J. Phys.: Conf. Ser. 119 072030 (2008)