Scalable data processing model of the ALICE experiment in the cloud

Lončar, Petra

Doctoral thesis / Disertacija

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture / Sveučilište u Splitu, Fakultet elektrotehnike, strojarstva i brodogradnje**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:179:516445

Rights / Prava: In copyright/Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-05-11**

UNIVERSITY OF SPLIT

FACULTY OF ELECTRICAL ENGINEERING, MECHANICAL ENGINEERING
AND NAVAL ARCHITECTURE

Petra Lončar

SCALABLE DATA PROCESSING MODEL OF THE ALICE EXPERIMENT IN THE CLOUD

DOCTORAL THESIS

Split, 2023

UNIVERSITY OF SPLIT

FACULTY OF ELECTRICAL ENGINEERING, MECHANICAL ENGINEERING
AND NAVAL ARCHITECTURE

Petra Lončar

SCALABLE DATA PROCESSING MODEL OF THE ALICE EXPERIMENT IN THE CLOUD

DOCTORAL THESIS

Split, 2023

The research reported in this thesis was carried out at Department of Electronics and Computing, University of Split, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture.

Supervisor: Sven Gotovac, PhD, Full Professor, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, University of Split, Croatia

Dissertation number: 184

Committee for assessment of doctoral dissertation:

1. Eugen Mudnić, PhD, Associate Professor, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Split

2. Josip Knezović, PhD, Full Professor, Faculty of Electrical Engineering and Computing, Zagreb

3. Goran Martinović, PhD, Full Professor, Faculty of Electrical Engineering, Computer Science and Information Technology, Osijek

4. Linda Vicković, PhD, Associate Professor, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Split

5. Dunja Božić-Štulić, PhD, Assistant Professor, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Split

Committee for defence of doctoral dissertation:

1. Eugen Mudnić, PhD, Associate Professor, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Split

2. Josip Knezović, PhD, Full Professor, Faculty of Electrical Engineering and Computing, Zagreb

3. Goran Martinović, PhD, Full Professor, Faculty of Electrical Engineering, Computer Science and Information Technology, Osijek

4. Linda Vicković, PhD, Full Professor, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Split

5. Dunja Božić-Štulić, PhD, Assistant Professor, Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture, Split

Dissertation defended on: 9 June 2023

Scalable Data Processing Model of the ALICE Experiment in the Cloud

Abstract

This thesis proposes an optimisation strategy for scalable Big Data processing in a heterogeneous Cloud. The resource needs of A Large Ion Collider Experiment (ALICE) at the European Organization for Nuclear Research (CERN) are reviewed as a motivating example. The thesis examines how to efficiently process and optimise the processing of resource-intensive tasks on a heterogeneous Cloud infrastructure distributed in five data centres to meet the needs of the ALICE experiment at the Tier 2 level. The objective was to perform research on a much larger number of tasks and resources of a significantly larger capacity than prior studies, which focused on a smaller number of tasks and resources with a lower capacity. The proposed and developed processing model for ALICE Monte Carlo production is based on a centralised software-defined management approach for the use of heterogeneous resources. Algorithms for assigning tasks to heterogeneous virtual resources have been analysed and proposed. The proposed algorithms are based on the selected Evolution Strategies meta-heuristic that has not yet been used in this domain, namely Evolution Strategies algorithm, Evolution Strategies algorithm with Longest Job First broker policy, and Evolution Strategies algorithm with Shortest Job First broker policy. The Cloud system model is implemented using the open-source CloudSim simulator. ALICE Monte Carlo production job requirements are imported into the simulation model as a workload created in Standard Workload Format (SWF) adapted for the Cloud simulator. The results of the simulation performance of the reference implementation under different loads were analysed and compared with the Genetic Algorithm from the same group of algorithms. The obtained results show multiple improvements. The proposed data processing model enables centralised software management of heterogeneous Cloud infrastructure, optimises measured metrics, improves resource usage, and achieves the system's scalability.

Keywords
ALICE, Big Data, Cloud computing, data processing, distributed management, Evolution Strategies, heterogeneity, resource management, scalability, software-defined, system simulation, task scheduling

Model skalabilne obrade podataka ALICE eksperimenta u oblaku

Sažetak

Ova disertacija predlaže strategiju optimizacije za skalabilnu obradu velikih podataka u heterogenom oblaku (engl. *Cloud*). U radu se kao motivirajući primjer razmatraju potrebe za resursima Eksperimenta na velikom ionskom sudaraču (engl. *A Large Ion Collider Experiment*, ALICE) na Europskoj organizaciji za nuklearna istraživanja (engl. *European Organization for Nuclear Research*, CERN). Rad istražuje kako učinkovito obraditi i optimizirati obradu resursno-intenzivnih zadataka na heterogenoj infrastrukturi u oblaku raspoređenoj u pet podatkovnih centara kako bi se zadovoljile potrebe na Tier 2 razini ALICE eksperimenta. Cilj je bio provesti istraživanje za višestruko veći broj zadataka i resurse znatno većeg kapaciteta u odnosu na dosadašnja istraživanja koja su provođena na manjem broju zadataka i resursima manjih kapaciteta. Predloženi i razvijeni model obrade za ALICE Monte Carlo produkciju temelji se na centraliziranom softverski definiranom pristupu upravljanja korištenjem heterogenih resursa. Analizirani su i predloženi algoritmi za dodjelu zadataka heterogenim virtualnim resursima. Predloženi algoritmi temelje se na odabranoj metaheuristici evolucijskih strategija (engl. *Evolution Strategies*) koja dosad nije korištena u ovoj domeni, a to su algoritam evolucijskih strategija, algoritam evolucijskih strategija s broker politikom kod koje prioritet izvođenja imaju najdulji zadaci (engl. *Longest Job First*) i algoritam evolucijskih strategija s broker politikom koja prioritet izvođenja daje najkraćim zadacima (engl. *Shortest Job First*). Model računalnog oblaka implementiran je pomoću CloudSim simulatora otvorenog koda. Zahtjevi ALICE Monte Carlo produkcijskih poslova uneseni su u simulaciju modela u obliku kreiranog radnog opterećenja u standardnom formatu radnog opterećenja (engl. *Standard Workload Format*, SWF) prilagođenom za rad u odabranom simulatoru. Analizirani su rezultati izvedbe simulacije referentne implementacije pod različitim opterećenjima i uspoređeni su s genetskim algoritmom (engl. *Genetic Algorithm*) iz iste skupine algoritama. Dobiveni rezultati pokazuju višestruka poboljšanja. Ovdje predložen model obrade podataka omogućava centralizirano softversko upravljanje heterogenom infrastrukturom u oblaku, optimizira mjerene metrike, poboljšava korištenje resursa i postiže skalabilnost sustava.

Ključne riječi
ALICE, veliki podaci, računarstvo u oblaku, obrada podataka, distribuirano upravljanje, evolucijske strategije, heterogenost, upravljanje resursima, skalabilnost, softverski definirano, simulacija sustava, raspoređivanje zadataka

Acknowledgments

Contents

List of Tables

List of Figures

xi

Abbreviations

ACORDE   ALICE Cosmic Ray Detector

AD   ALICE Diffractive

AF   Analysis Facility

AI   Artificial Intelligence

ALICE   A Large Ion Collider Experiment

AliEn   ALICE Environment

AMORE   Automatic MOnitoRing Environment

AOD   Analysis Object Data

ATLAS   A Torodial LHC ApparatuS

B   Bytes

BDA   Big Data Analytics

CAF   CERN Analysis Facility

CASTOR   CERN Advanced STORage manager

CCDB   Condition and Calibration Data Base

CE   Computing Element

CERN   European Organization for Nuclear Research (in French - Conseil Européen pour la Recherche Nucléaire)

CMS   Compact Muon Solenoid

CPU   Central Processing Unit

CPV   Charged Particle Veto

CRSG   Computing Resources Scrutiny Group

CRU   Common Readout Unit

CSV   Comma-Separated Values

CTF   Compressed Time Frame

CTP	Central Trigger Processor
CVMFS	CernVM File System
DAQ	Data Acquisition
DCal	Di-jet Calorimeter
DCS	Detector Control System
DDL	Detector Data Link
DQM	Data Quality Monitoring
ECS	Experiment Control System
EMCal	Electromagnetic Calorimeter
EOS	CERN storage technology
EPN	Event Processing Node
ES	Evolution Strategies
ESD	Event Summary Data
eV	Electronvolt
FIT	Fast Interaction Trigger
FLOPS	Floating-Point Operations Per Second
FLP	First Level Processor
FMD	Forward Multiplicity Detector
GDC	Global Data Collector
GPU	Graphics Processing Unit
GUI	Graphical User Interface
HEP	High Energy Physics
HL-LHC	High Luminosity LHC
HLT	High Level Trigger
HMPID	High Momentum Particle Identification Detector
HPC	High Performance Computing

HR-ZOO	Croatian Scientific and Educational Cloud (in Croatia - Hrvatski znanstveni i obrazovni oblak)
HS06	HEP-SPEC06
HSC	High Scalability Computing
HTC	High Throughput Computing
IaaS	Infrastructure as a Service
IP	Interaction Point
IT	Information Technology
ITS	Inner Tracking System
JAliEn	Java ALICE Environment
L0	Level 0 Trigger
L1	Level 1 Trigger
LCG	LHC Computing Grid
LDC	Local Data Concentrator
LEIR	Low Energy Ion Ring
LHC	Large Hadron Collider
LHCb	Large Hadron Collider beauty
LHCONE	LHC Open Network Environment
LHCOPN	LHC Optical Private Network
LINAC	Linear Accelerator
LJF	Longest Job First
LS	Long Shutdown
LTU	Local Trigger Unit
MC	Monte Carlo
MCH	Muon Chamber System
MFT	Muon Forward Tracker

MID	Muon Identifier
MIPS	Millions of Instructions Per Second
MonALISA	Monitoring Agents using a Large Integrated Services Architecture
MTR	Muon Trigger
NIST	National Institute of Standards and Technology
NoSQL	Non-Structured Query Language
NP	Non-deterministic Polynominal-time
O^2	Online-Offline system
OCDB	Offline Condition DataBase
p-p	proton-proton
p-Pb	proton-lead
PaaS	Platform as a Service
Pb-Pb	lead-lead
PDP	Physics Data Processing
PHOS	Photon Spectrometer
PID	Particle Identification
PMD	Photon Multiplicity Detector
PS	Proton Synchrotron
PSB	Proton Synchrotron Booster
PSO	Particle Swarm Optimisation
PWG	Physics Working Group
QC	Quality Control
QGP	Quark-Gluon Plasma
QoS	Quality of Service
RAM	Random Access Memory
ROOT	CERN's Object-Oriented Technology

SaaS	Software as a Service
SDN	Software-Defined Networking
SE	Storage Element
SJF	Shortest Job First
SLIMOS	Shift Leader in Matters of Safety
SPEC	Standard Performance Evaluation Corporation
SPS	Super Proton Synchrotron
SQL	Structured Query Language
STF	Sub-Time Frame
SWF	Standard Workload Format
T	Tesla
T0	TZERO
TF	Time Frame
TOF	Time Of Flight
TPC	Time Projection Chamber
TRD	Transition Radiation Detector
V0	VZERO
VM	Virtual Machine
VO	Virtual Organisation
WAN	Wide Area Network
WLCG	Worldwide LHC Computing Grid
WN	Worker Node
XaaS	Everything as a Service
ZDC	Zero Degree Calorimeter
ZN	Neutron ZDC
ZP	Proton ZDC

1 INTRODUCTION

The European Organization for Nuclear Research (in French: *Conseil Européen pour la Recherche Nucléaire*, CERN) is an international organisation established in 1954 whose main area of research is particle physics. It operates the Large Hadron Collider (LHC), the largest particle physics laboratory and most powerful particle accelerator in the world. The LHC is located in Greater Geneva, the French-Swiss cross-border agglomeration.

The most sophisticated scientific detectors and systems are used to detect, track, and identify fundamental particles at particle physics experiments at CERN. High Energy Physics (HEP) is a data-intensive field of research that encompasses the acquisition, processing, storage, access, and proper interpretation of data. The expected quantity of produced data places increasing requirements on processing, networking, and storage resources. For reliable and efficient processing and storage of large amounts of data, it is necessary to simultaneously develop and apply new technologies and invent new solutions and concepts. Data and resource management are challenging and relevant problems in optimising the entire system. This doctoral thesis deals with scalability as a property of exceptional importance in scientific data management. This challenge and the motivation to overcome it are outlined in this chapter.

1.1 Motivation and challenges

The motivation for the research of this doctoral thesis finds its origin in the fact that the huge data growth requires an infrastructure that must keep pace with the growth, must adapt to the new requirements, expand in the shortest possible time, and in addition, maintain the lowest possible costs of any modifications. The number of information to be analysed increases exponentially while the speed of processing them decreases. The increasing computing power and development of new and more advanced processors can accelerate the real-time analysis of large datasets and enable concrete, timely and valid information obtained from the dataset. In recent years, the Cloud has emerged as an interesting infrastructure option for implementing scientific workflows for modelling experiments and outsourcing data storage and application execution. Building a Cloud infrastructure is complex and challenging on many levels, and the heterogeneity of the systems included in this type of infrastructure is particularly important. As heterogeneous resources are rapidly integrated into the Cloud, adaptable strategies and abstraction techniques are required for efficient resource management.

The Information Technology (IT) sector faces many challenges that need to be addressed promptly. Strong data growth and data traffic require investment in Cloud solutions, software applications, and Big Data analytics. Such dynamics demand the IT infrastructure to be agile and easy to use while at the same time being reliable and available. The data centre needs to be adapted and improved to ensure flexibility, reliability, secure integration of solutions into existing IT infrastructure, and fast configuration to new requirements. New solutions should meet the criteria of scalability in size, performance, safety, and reliability. All system components are interconnected and interdependent as they use common hardware resources, making scaling difficult and creating performance problems with a large data flow and a large number of users.

This thesis aims to explore approaches to support the scalable processing of data at massive scales. The goal is to find a computation model that is flexible enough to encompass commonly distributed architectures and the specificities of the hardware architecture. For this purpose, systematic research on data management possibilities will be carried out to identify algorithms that effectively distribute workloads to several geographically distant sites in the Cloud. The system model will be leveraged by representative Big Data real-life applications from particle physics that have challenging resource requirements.

The LHC experiments are the source of large amounts of physics data. According to CERN data from 2017 [1], the LHC experiments surpassed 200 PB of archived data. After the detectors upgrades, a significant increase in the generation rate and the amount of important data is expected, in order of exabytes. The scheduled LHC upgrade [2] will result in increased demands for resources that will be challenging to complete with currently available computing and storage capacity. The goals of the A Large Ion Collider Experiment (ALICE) upgrade [3] from 2018 to 2022 were to upgrade subdetectors and expand computing infrastructure. The goal for the future is to minimise the volume of data obtained in the collision of different particles. The data volume reduction will be achieved through several processing phases parallel with data collection. The Worldwide LHC Computing Grid (WLCG) [4] is continuously upgraded and will participate in data processing and storage with its Tiers. The LHC ALICE Raw and Monte Carlo (MC) use cases are multidisciplinary in nature and challenging in content. Long-running Monte Carlo simulations generate enormous volumes of data at high velocities and require resources with high performance and data sharing on Cloud infrastructure. Therefore, additional computing resources are especially needed for simulations of physics events.

Scalability is a property of great importance in the growing field of Big Data management. The joint work of many researchers worldwide is made possible by scalable distributed infrastructures and Big Data technologies. Technologies for storage and processing must meet the experiments' requirements for achieving the system's scalability under new and changing conditions. Scaling data collection systems is a multidisciplinary problem that requires the development of models, structures, and technological solutions. CERN research and

development activities are focused on the application and exploitation of the potential of new technologies for data analysis and system control and architectures that will bring heterogeneity to data centres in the components of processing, storage, and networking. Heterogeneity and parallelism are inevitable concepts to meet stringent performance requirements.

Moreover, High Performance Computing (HPC) is likely to be part of the future HEP computing infrastructure since LHC's needs today are equivalent to $\sim$ 30 PFLOPS.

Personal experience of working in data-intensive science and direct research work on monitoring and analysing the data quality collected by the ALICE detector on Data Quality Monitoring and Offline critical real-time operation tasks during Run 2 (2015-2018) at CERN has led to an interest to further research scalability and improve its correlation with other parameters in the realm of Big Data in new and dynamic conditions, primarily in the Cloud. Analysing the ALICE productions, scalability properties in physics data processing are recognised as of exceptional importance.

The motivation for this research is complementary to the strategic goals for achieving digital sovereignty and competitiveness of the European Union through the construction of supercomputers and the participation of the Republic of Croatia in them. In 2019, the Republic of Croatia received the status of an associate member of CERN. This is a path and effort to contribute to developing solutions and managing large amounts of data from LHC experiments at the level of the national e-infrastructure for high performance and Cloud computing called Croatian Scientific and Educational Cloud (HR-ZOO) [5].

This thesis will consider the use of a heterogeneous Cloud ecosystem for ALICE data processing at Tier 2 of the WLCG infrastructure for subsequent data collection periods. Tier 2 consists of resources provided by well-networked universities or scientific institutions organised either as separate data centres or federations of data centres. Tier 2 participates in asynchronous, offline ALICE Monte Carlo production data processing.

Such an ecosystem defined by heterogeneity requires the development of software tools to manage applications and resource utilisation. Increasing the utilisation efficiency of current heterogeneous Cloud platforms can be achieved in many ways. One is in the resource management aspect attained by software-defined support for scalable, dynamic, and flexible data processing.

1.2   Hypothesis

This research is based on the fact that Big Data technologies have a great potential and the ability to facilitate and advance data-driven discovery in science and industry. Scalability is a property with great potential and is extremely important for Big Data. Scalability is the primary focus of Cloud computing and one of the crucial issues in the research field. It is needed to constantly explore scaling capabilities and find scalability improvements in the relevant dynamic research area.

Resource scalability in correlation with other computing paradigms can optimise overall performance and improve efficiency and reliability of a modern data system. Scalability is the issue in large-scale computing dominantly influenced by the selection of resource management algorithm that involves the dynamic allocation of heterogeneous resources. The selected Evolution Strategies metaheuristic has the potential to optimise resource utilisation. A software-defined approach would enable agility and the ability to adapt to the dynamic needs of data-intensive and computing-intensive workflow in the Cloud environment. Scalability testing is a crucial phase of Cloud system development. Relevant scalability performance metrics need to be collected and measured, and their impact interpreted to test scalability and achieve better performance for increasing workload demands.

1.3   Research methodology

The scientific research in this thesis is conducted in a manner that the modelled and simulated system reflects the realistic Cloud computing system in a scientific research environment incorporating dynamic heterogeneous resources.

The model includes a central controller that has information about the state of used resources and manages the allocation of resources through the dynamic task scheduling algorithm. The chosen Evolution Strategies algorithm inspired by the theory of evolution and natural selection represents an intelligent way to achieve scalability and increase resource utilisation by matching the tasks with the best corresponding virtual machine (VM) configuration. Through this approach, the jobs are distributed over several data centres using optimised allocation.

Simulation abstracts functionalities and behaviours of physical and virtual resources. The scalable processing model in this thesis focuses on the technique for allocating tasks to VMs considering job requirements focusing on the CPU and RAM components and QoS aspects. Differently configured VMs are multi-core resources fixed on distributed data centres' hosts.

The approach focuses on the heterogeneous Cloud environment for scientific data from the HEP domain, especially on resource-intensive ALICE Monte Carlo production jobs. Monte Carlo production jobs are asynchronous processes taking more than 60% of the total ALICE CPU wall time. For this research, simulation jobs are configured for a single-core execution and execution on 8-core resources. Monte Carlo jobs are executed as batch jobs with no fixed execution deadline. These jobs are independent and do not have enormous storage or network requirements. Their performance depends primarily on the CPU and memory capabilities. The cost for simulating one event in Run 2 was 24000 HEP-SPEC06 (HS06) [6]. It is estimated to simulate 1×10^9 events in Run 3. As input, Monte Carlo simulation jobs use a limited file size that is the output generated by event generation (~ 200 MB). MC Event Summary Data (ESD) and MC Analysis Object Data (AOD) are the output of the simulation. Network requirements for uploading simulation output are not high (~ 350 MB). Monte Carlo

jobs of a typical duration of 6 hours require at least 2 GB of RAM per core and at least the same amount of swap memory and disk space [7].

The data from Monte Carlo production are collected from MonALISA, which stands for Monitoring Agents using a Large Integrated Services Architecture, and adapted for multi-core processing. The workload is created from the specified data. Scalability testing is performed under multiple load levels, from 1000 to 20000 jobs dynamically arriving in varying intervals. Performance metrics representing the system qualities of interest for conducting the system and scalability analysis are collected, analysed, and compared.

1.4   Thesis outline

The outline of this thesis is shown in Figure 1.1. Chapter 1 explains the motivation and challenges in data processing of the ALICE data. It includes the thesis's objectives, set research hypothesis, and research methodology. Chapter 2 describes the ALICE detector components and discusses the recent upgrade of the experiment carried out in parallel with the work on this research. These upgrades will result in challenges in managing a large volume of scientific data that must be considered for optimising processing performance. This chapter discusses the challenges in the data processing. Chapter 3 identifies the necessary components and methodology for scalability analysis of ALICE Monte Carlo processing on a heterogeneous Cloud using simulation. The scalability property is analysed and the characteristics of the software-defined concept are presented. A review of Cloud computing simulators is given and the appropriate simulator is selected. Chapter 3 also provides a literature review on this research issue. Chapter 4 describes the heterogeneous Cloud infrastructure and all data centre components that support the Cloud system. The proposed data processing model of ALICE offline production is presented. Chapter 5 presents and applies a software-based Evolution Strategies metaheuristic algorithm to optimise Cloud resource allocation. Finally, Chapter 6 summarises the contributions and provides an outlook for future research on scalable Cloud data processing.
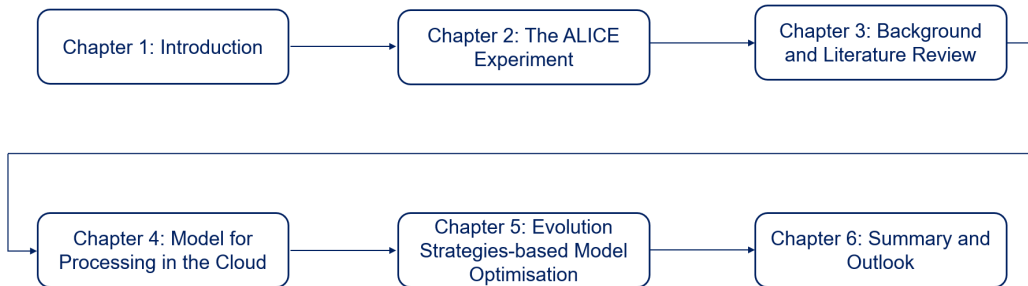
Figure 1.1: The thesis outline.

2 THE ALICE EXPERIMENT

The ALICE experiment [8] is one of CERN's four large particle physics experiments. The ALICE physics programme and detector are introduced in this chapter, and the data processing flow is discussed. A recent upgrade of the ALICE experiment carried out in preparation for data taking in the following LHC runs is described here.

2.1 The Large Hadron Collider

The LHC [9] is the last element of the accelerator complex at CERN. It helps to answer unsolved questions of the Standard Model of particle physics [10], which embodies the current understanding of fundamental particles and forces. The LHC uses the 27 km circumference tunnel built at a mean depth of 100 m, where counter-circulating beams collide. This circular accelerator accelerates two beams of particles called hadrons to obtain the highest energy collisions in the volume of about one million particle collisions per second.

The accelerator complex accelerates protons and heavy lead ions. The protons obtained from hydrogen atoms in Linear Accelerator (LINAC) 4 are injected into the Proton Synchrotron Booster (PSB) at 50 MeV. LINAC 4 replaced LINAC 2 in 2020. With the LINAC 4, hydrogen ions are accelerated to 160 MeV and stripped to protons. Passing through the Proton Synchrotron (PS) and Super Proton Synchrotron (SPS), protons are accelerated to 450 GeV. They are finally injected into the LHC, where they are accelerated for 20 minutes to 6.5 TeV. Particle beams circulate through a vacuum inside the LHC pipes in opposite directions for many hours under normal operating conditions. The particles of the LHC beam, formed in bunches, are manipulated using various magnets. Dipole magnets hold the particles in almost circular orbits, quadrupole magnets direct the beam down to the smallest possible size at the collision points, and accelerating radiofrequency cavities accelerate charged particles injected in the electromagnetic field to achieve the maximum number of collisions with high luminosity at the collision points. Heavy lead ions are produced from a highly purified lead sample heated to around 800 °C. Lead ions enter LINAC 3 before being collected and accelerated in the Low Energy Ion Ring (LEIR) from 4.2 MeV/u (energy per nucleon) to 72 MeV/u. Following that, they take the same path as protons to achieve maximum energy of 2.56 TeV/u.

A Large Ion Collider Experiment (ALICE), A Toroidal LHC ApparatuS (ATLAS) [11],

Compact Muon Solenoid (CMS) [12], and Large Hadron Collider beauty (LHCb) [13] are the four largest experiments [14] or particle detectors installed in underground caverns built around the four Interaction Points (IPs) of the LHC beams, as shown in Figure 2.1.
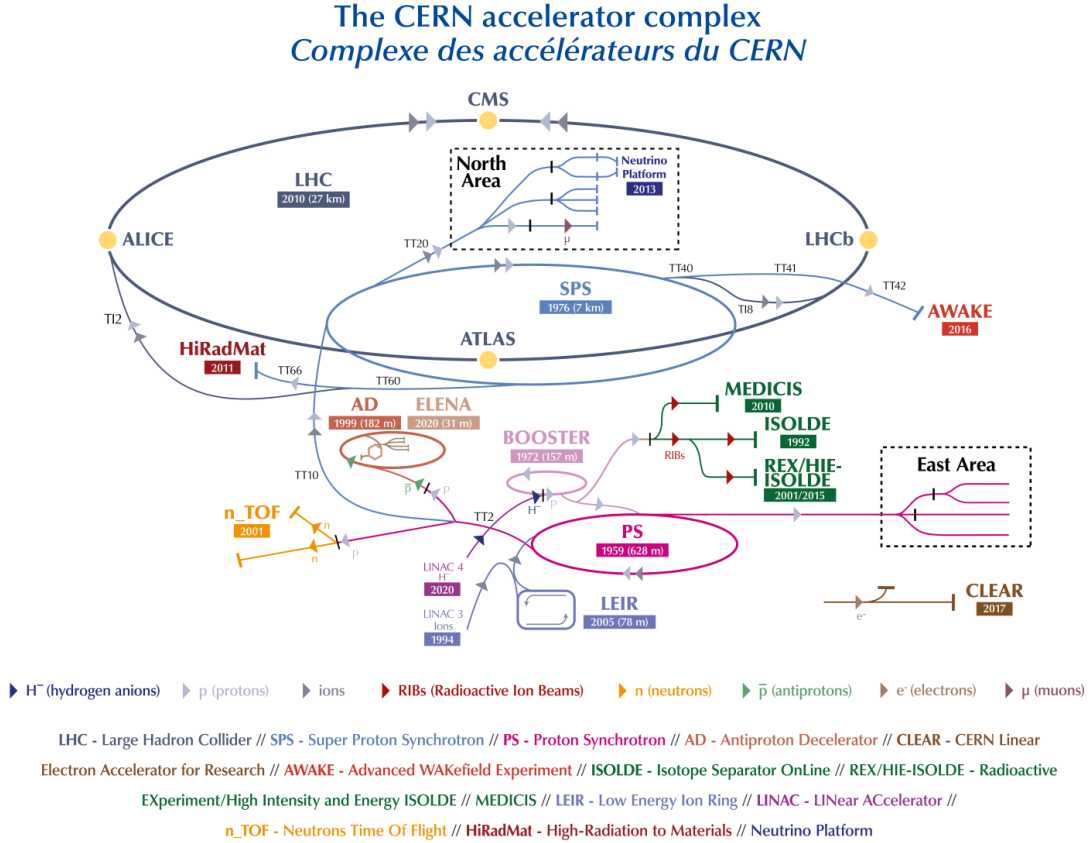
Figure 2.1: The CERN accelerator complex [15].

The ALICE detector studies heavy-ion physics and Quark-Gluon Plasma (QGP), a state of matter assumed to have filled the Universe just after the Big Bang. Quarks and gluons are the elementary particles that form protons and neutrons.

The ATLAS is a general-purpose detector designed to test the Standard Model's predictions, from precision measurements of the Higgs boson and dark matter to searches for new physics beyond the Standard Model.

The CMS is a second multi-purpose detector with scientific goals similar to the ATLAS experiment but with a different technical design. It is a cylindrical coil of superconducting cable that generates a magnetic field of 4 T and is built around a massive superconducting solenoid.

LHCb experiment studies the slight asymmetry between matter that dominates the Universe today and antimatter present in interactions of B-particles (particles containing the b quark, common in the aftermath of the Big Bang). It measures and studies the decay of

particles produced in one of the beam directions. Observing its dimensions, LHCb is the smallest of the four detectors.

The LHC experiments share the same schedule. The schedule defines periods of detectors taking data, maintenance, and upgrades. Technical Stops are periods intended for technology maintenance, while Long Shutdowns (LSs) are periods for experimental upgrades and changes when there is no physics and when major operations take place in the underground areas. Runs are periods of active data taking. The long-term LHC schedule is classified and planned as follows (Figure 2.2):

- Run 1 (2009 - 2013)

- LS1 (2013 - 2015)

- Run 2 (2015 - 2018)

- LS2 (2018 - 2022)

- Run 3 (2022 - 2025)

- LS3 (2026 - 2028)

- Run 4 (2029 - 2032)

- LS4 (2033 - 2034)

- Run 5 (2035 - 2038).

Figure 2.2: Plan for the LHC upgrade [16].

After the LS3, an upgraded version of the LHC called High Luminosity LHC (HL-LHC) [2] will be introduced. HL-LHC will operate at a higher luminosity. Luminosity is an important measure of an accelerator's performance that measures the number of potential collisions per surface unit over a given time (second). Luminosity and the volume of collected data are linearly related. Greater luminosity means more collisions and, consequently, more data.

2.2 The ALICE detector

The ALICE is a general-purpose heavy-ion experiment at the LHC. It is designed to address the physics of the strongly interacting QGP created in heavy-ion collisions. ALICE started its first data taking in 2008 and the second one in 2015. In parallel with the research process on this thesis, the ALICE LS2 upgrade took place. During LS2, the ALICE Collaboration has significantly prepared to start collecting data with an upgraded detector and higher integrated luminosity than in Run 2. The expected interaction rate for the lead-lead (Pb-Pb) collisions is 50 kHz and for the proton-proton (p-p) and proton-lead (p-Pb) sampling up to 200 kHz. The replacement of the beam pipe used in Run 2 with a smaller-diameter beam pipe was needed. Reducing the beam pipe diameter improves measurements of determining the interaction point positions. From Run 3, the ALICE experiment runs in continuous readout mode (trigger mode was used in previous Runs). Detectors have upgraded the electronics to improve the readout performance. The ALICE upgrade will significantly increase the data volume transferred from the detector electronics to the readout system.

Schematic diagrams of the detector in Run 2 and Run 3 are given in Figure 2.3 and Figure 2.4, respectively.
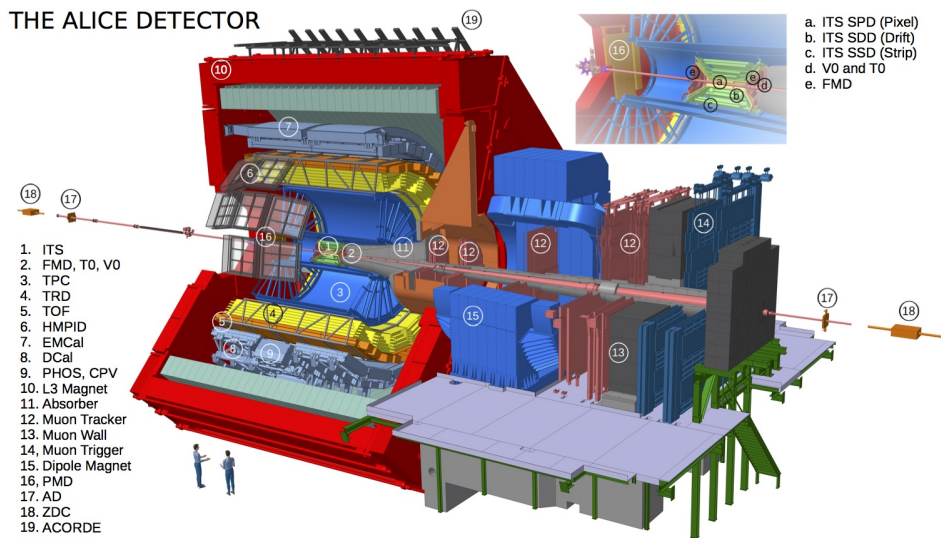
Figure 2.3: The ALICE detector in Run 2 [17].

1	**ACORDE** | ALICE Cosmic Rays Detector
2	**AD** | ALICE Diffractive Detector
3	**DCal** | Di-jet Calorimeter
4	**EMCal** | Electromagnetic Calorimeter
5	**HMPID** | High Momentum Particle Identification Detector
6	**ITS-IB** | Inner Tracking System - Inner Barrel
7	**ITS-OB** | Inner Tracking System - Outer Barre
8	**MCH** | Muon Tracking Chambers
9	**MFT** | Muon Forward Tracker
10	**MID** | Muon Identifier
11	**PHOS / CPV** | Photon Spectrometer
12	**TOF** | Time Of Flight
13	**T0+A** | Tzero + A
14	**T0+C** | Tzero + C
15	**TPC** | Time Projection Chamber
16	**TRD** | Transition Radiation Detector
17	**V0+** | Vzero + Detector
18	**ZDC** | Zero Degree Calorimeter

Figure 2.4: The ALICE detector in Run 3 [18].

The subdetectors have to ensure the detector's maximum performance during data taking. The ALICE apparatus comprises three main parts: the central barrel, the muon arm, and the forward detectors. The ALICE subdetectors are organised as individual projects with their own organisation.

Central detectors

All the detectors in the central barrel are embedded in the ALICE solenoid L3 magnet, providing a relatively low magnetic field (< 0.5 T). The detectors of the central barrel are: the Inner Tracking System (ITS), the Time Projection Chamber (TPC), the Transition Radiation Detector (TRD), the Time Of Flight (TOF), the High Momentum Particle Identification Detector (HMPID), the Photon Spectrometer (PHOS), the Charged Particle Veto detector (CPV), the Electromagnetic Calorimeter (EMCal), and the Di-jet Calorimeter (DCal). These detectors are used to track and identify particles produced at mid-rapidity. In addition, in the same rapidity region of the central barrel but on top of the L3 magnet is ALICE Cosmic Ray Detector (ACORDE), an array of large scintillators used to trigger cosmic rays for calibration and alignment purposes.

The ITS is the cylindrical detector placed close to the beamline in the central barrel. ITS's main purposes are to reconstruct primary and secondary vertices to improve the ALICE barrel tracking capabilities in the vicinity of the interaction point. In Run 1 and Run 2, ITS consisted of six cylindrical layers where different silicon technologies were used for the ITS (pixel, drift, and strip). After the LS2 upgrade, the ITS consists of seven concentric layers of monolithic pixel detectors [19] to improve the tracking performance and allow detailed study of certain

Quantum Chromodynamics theories and detection of low-momentum particles.

The TRD is one of the particle identification (PID) detectors. It identifies electrons to study production rates of heavy quarks triggering on electrons and jets with fast selection capability ($< 7\,\mu$s). TRD readout electronic has been upgraded in LS2. This detector is one of the most complex LHC detector systems of 18 large modules surrounding TPC. The TPC is the main tracking detector in the ALICE central barrel focused on hadronic physics. It is important for measuring high transverse momentum electrons produced in central Pb–Pb collisions. TPC has replaced the frontend and data concentrator electronics to improve the readout performance in Run 3 and Run 4 [20]. That will allow ALICE to record the information of all tracks produced in Pb-Pb collisions at rates of 50 kHz.

One of the detector systems dedicated to PID is the HMPID. The HMPID task is to detect the Cherenkov light, light emitted when a charged particle passes through a medium. By measuring the light's velocity and direction, it is possible to determine the mass and type of particle.

The TOF detector precisely measures the flight time of particles from the collision point out to the detector. That requires great time resolution performance.

Two calorimeters in ALICE are EMCal and PHOS. The EMCal is a Pb-scintillator sampling calorimeter optimised to measure jet production rates and jet characteristics in conjunction with the charged particle tracking in the other barrel detectors. The addendum of EMCal, the DCal detector, improves the acceptance and statistics of the measurement. The PHOS has greater granularity and energy resolution than EMCal. This crystal-based calorimeter measures photons and neutral mesons. It can sustain high particle densities. A set of proportionate CPV chambers in front of PHOS aids the separation of charged particles from photons. PHOS, EMCal, and DCal also deliver the hardware L0 and L1-level triggers to the ALICE central trigger processor to select events with high-energy photons and electrons in real-time to reduce data for later analysis.

Muon Spectometer

The Muon Spectrometer arm detectors are used to study quarkonia production at forward rapidity through their decay into muons. The main components of the spectrometer are an absorber used to filter muons from all particles coming from the background, a set of tracking chambers (Muon Tracker, MCH), and a set of trigger chambers (Muon Trigger, MTR). The MTR system is designed to select interesting events, the heavy quark resonance decays, in a decision time of about 300 ns above the configurable threshold.

From Run 3, the Trigger system has two modes of detector operation, continuous and triggered. Data loss must be reduced to the minimum to allow efficient and coordinated data taking in a continuous readout system. The MTR has been upgraded to the Muon Identifier (MID). The Muon Forward Tracker (MFT) [21] is a new tracking detector for Run 3 mounted

into the TPC. It is going to detect and tackle the physics challenges provided by significantly larger luminosity. The MFT supplements the muon spectrometer, distinguishing muon pairs coming from charm hadron or bottom hadron decays and thus extends the physics programme and enables new measurements of charm and bottom quarks.

Forward detectors

The forward detectors are located at a small radial distance from the beamline. The Zero Degree Calorimeter (ZDC), the Photon Multiplicity Detector (PMD), the Forward Multiplicity Detector (FMD), the VZERO detector (V0), the TZERO (T0) and the ALICE Diffractive (AD) are included in this group. Two identical sets of calorimeters consisting of a neutron (ZN) and a proton (ZP) ZDC are located on both sides of the ALICE detector, 112.5 m away from the interaction point. The ZDC measures the centrality and luminosity in Pb-Pb collisions and provides offline event selection to differentiate background events produced beyond the interaction zone from beam-beam collisions. The PMD measures the multiplicity and spatial distribution of photons in each nucleus collision (Figure 2.3). It is designed of two planes with cellular honeycomb chambers, two lead converter plates, and a support assembly. The PMD is made of non-magnetic materials and uses gas as the sensitive medium.

The FMD provides precise charged particle multiplicity information for all collision types in the defined pseudorapidity range. It consists of more than 50000 silicon strip channels distributed over five rings placed at different positions along the beam pipe.

The trigger detector T0 measures the interaction time. T0 has several functions: supplying main signals to the L0 trigger, waking up to TRD, and giving a precise start signal for TOF particle identification.

The main tasks of the V0 are to select interactions and to reject beam-related background events. It estimates and detects charged particles by measuring their charge and arrival time. The AD forward detector system optimises trigger efficiencies in selecting diffractive events, monitors beam-gas background, and can serve as a luminometer. It comprises two modules made of scintillator pads, one module on each side of the interaction point.

The Run 2 assembly of the V0/T0/FMD detector system is replaced by a single detector system named Fast Interaction Trigger (FIT) [22], located in the ALICE detector's forward region at positions close to the present V0/T0 location. The FIT is designed to provide the functionality of the existing forward detectors. The FIT system has to cope with requirements for low latency and perform the following tasks: fast triggering, monitoring LHC background conditions and luminosity, measuring diffractive cross sections, monitoring beam quality, and beam-gas events rejection. The FIT delivers the produced trigger signals to the Central Trigger System.

2.3   Data taking system

The ALICE data processing is divided into an online part in real-time and an offline part, as seen in Figure 2.5. Operating procedures are handled from the ALICE Run Control Centre, the central workplace for data acquisition activities in the ALICE experimental area at LHC Point 2.

The data taking procedure used in Run 1 and Run 2 in trigger readout mode will be briefly described here, as well as the recently undergone upgrade of the ALICE computing model. The ALICE Central Trigger Processor (CTP) system has been selecting events from p-p, p-Pb, and Pb-Pb collisions in time intervals generating three levels of hardware triggers in accordance with the LHC clock. In Run 2, the CTP was used in triggered mode receiving and distributing trigger signals to eliminate background signals, calibration, and commissioning at a higher interaction rate using one Local Trigger Unit (LTU) for each detector. Continuous readout of new and upgraded detectors is going to be supported by Common Readout Unit (CRU). CRU interface enables the connection of the readout systems to the new computing system and the control system. EMCal, PHOS, CPV, DCal, TRD, and HMPID are used as triggered-only detectors.

The data-driven ALICE Data Acquisition System (DAQ) [23] was handling the stream of received physics data and transfers of data over optical Detector Data Links (DDLs) to the Local Data Concentrator (LDC) computer resources, followed by processing on Global Data Collector (GDC) where the complete events were formed and formatted for storage by LDC machines. The LDCs stored the data in their memory, waiting for the High Level Trigger (HLT) decision. The HLT reconstructed data to reduce the volume of physics data by data selection and compression while keeping relevant events. Accepted data were then transferred to the GDCs, where the whole events were built. These events were stored on local disks in files encoded using the AliRoot [24] format suitable for Offline data processing. The data files were finally transferred to the CERN Data Centre, where they were archived and ready to be published on the Grid (described in Section 2.4). ALICE experimental area was interconnected with the CERN Data Centre using Ethernet links.

Conditions data produced during data taking relevant to the calibration of individual detector signals and the offline reconstruction were stored in the Offline Condition DataBase (OCDB). The SHUTTLE framework collected and handled the gathering, processing, and publication of the conditions data needed for reconstruction. After data processing, the SHUTTLE registered the produced condition files in ALICE Environment (AliEn) [25] and stored the data in the CERN Advanced STORage manager (CASTOR) tape system. AliEn processing software was used to store and analyse the experiment's data. It managed distributed storage and CPU resources for the ALICE experiment.

Stored raw and conditions data were available for offline processing and analysis on the computing Grid worker nodes (WN) using AliRoot. The AliRoot was the offline framework

for simulation and reconstruction used in Run 1 and Run 2. This ROOT [26]-based framework was also used for analysing reconstructed data and preparing physics publications.

Several more systems were necessary for the proper functioning of the experiment.

The Data Quality Monitoring (DQM) system was an essential operational part of the experiment that was giving real-time feedback on the quality of the recorded data. It used ROOT-based Automatic MOnitoRing Environment (AMORE) [27] software to identify potential issues in advance. DQM involved the online data gathering, analysis by user-defined algorithms, storage, and visualisation of the produced monitoring information gathered by processes - agents that were publishing results in a pool later visualised through a dedicated user interface.

The Detector Control System (DCS) has been a complex information and control system organised in a hierarchical way and connected with the DAQ, Trigger and Offline systems, and LHC Machine. It is responsible for the safety and accurate operation of the ALICE, remote control, acquisition, processing, and archiving of data for control, monitoring, and configuration purposes.

The Experiment Control System (ECS) has been the main part of the ALICE control system realised as a software layer on top of CTP, DAQ, HLT, and DCS that has been providing an interface to the online systems and control of data processing activities on the experiment.

Central shifts during Run 2 data taking were formed around the systems of ALICE data flow (DCS, ECS/DAQ+CTP+HLT, and DQM) to ensure high efficiency and quality assurance during runs, especially important during Pb-Pb collisions, which are of the utmost importance for ALICE experiment.
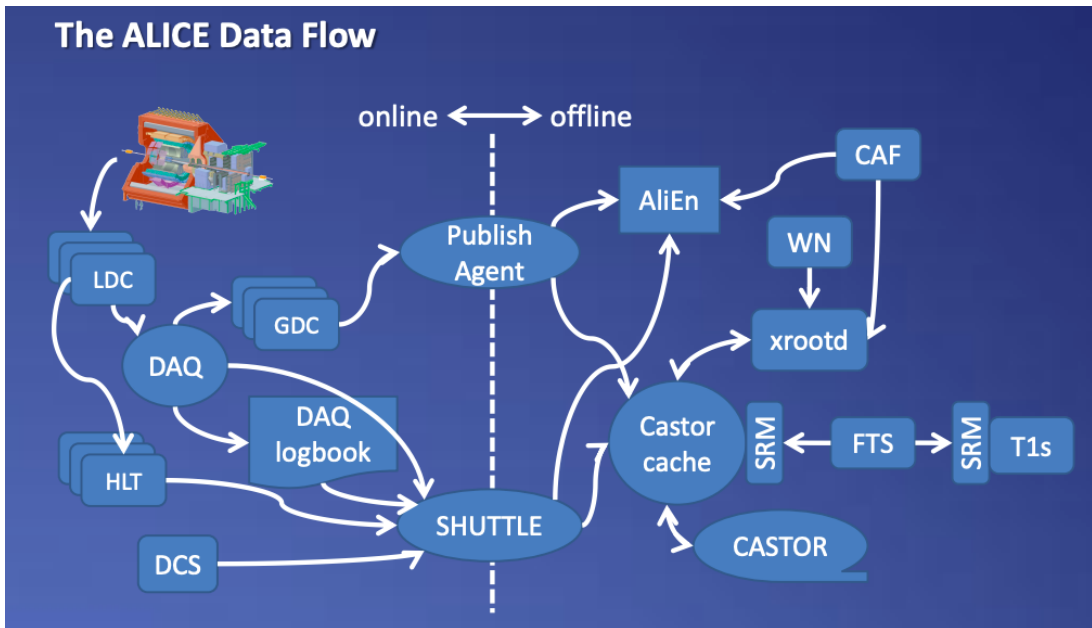
Figure 2.5: The ALICE data flow in Run 1 and Run 2 [28].

Apart from the detector system upgrade during LS2, the computing system is upgraded with the Online-Offline (O^2) computing system for future runs [3]. The new O^2 system is located in the ALICE experimental part. The goal of the computing model is to maximally reduce the data volume to minimise requirements for storage and processing resources caused by higher luminosity and the interaction rate, meaning significantly increased data amount in Run 3 and Run 4.

The O^2 system participates in the online and offline data processing. The online part takes place synchronously with the data taking, while the offline part of the data processing takes place asynchronously after a few weeks or even months on the available part of the system. The O^2 system is designed to participate in all computer tasks. ALICE processing tasks are detector calibration and reconstruction of real data, Monte Carlo simulation, reconstruction of simulated data, and organised and user analysis. In addition, it provides sufficient capacity to store the approximate amount of one-year production data.

The O^2 system consists of a network of 200 First Level Processor (FLP) and 250 Event Processing Node (EPN) computer nodes. During the different phases of data reconstruction, data of different formats and sizes based on time frames are generated. By switching to continuous readout from trigger readout mode in ALICE Run 3 data processing, a basic processing unit is no longer an event but a time frame (~ 20 ms).

The online processing on the O^2 system includes the calibration and reconstruction of raw data from the ALICE detector. As seen in Figure 2.6, local raw data compression starts on FLP nodes using lossless algorithms and continues on EPN nodes. FLPs will collect detector data at speeds of about 3.5 TB/s for the period of Pb-Pb collisions via optical readout links. The reduction and compression on each FLP result in the production of a Sub-Time Frame (STF). The partially compressed STFs are then forwarded to the EPN nodes, where the STFs bound are aggregated into TFs for the same time period. Then, the EPNs perform data reconstruction for each detector and further reduce the data. Thus, compressed data in Compressed Time Frame (CTF) format are stored on EOS disks at a total peak throughput of 90 GB/s and archived in WLCG Tier 0 and Tier 1 centres. Creating and storing CTFs completes the synchronous processing.

Data in Event Summary Data (ESD) format are produced during asynchronous calibration and reconstruction on O^2 and WLCG Tier 0 and Tier 1. Two reconstruction steps (called pass 1 and pass 2) are planned to obtain Analysis Object Data (AOD) data of a certain quality for analysis containing the final path parameters at a given point of a physical event.

In Run 3, central shifts are formed around Data Acquisition (ECS/FLP/CTP/EPN), Detector Control System (DCS), and Quality Control (QC/PDP/EOS/GRID) systems.
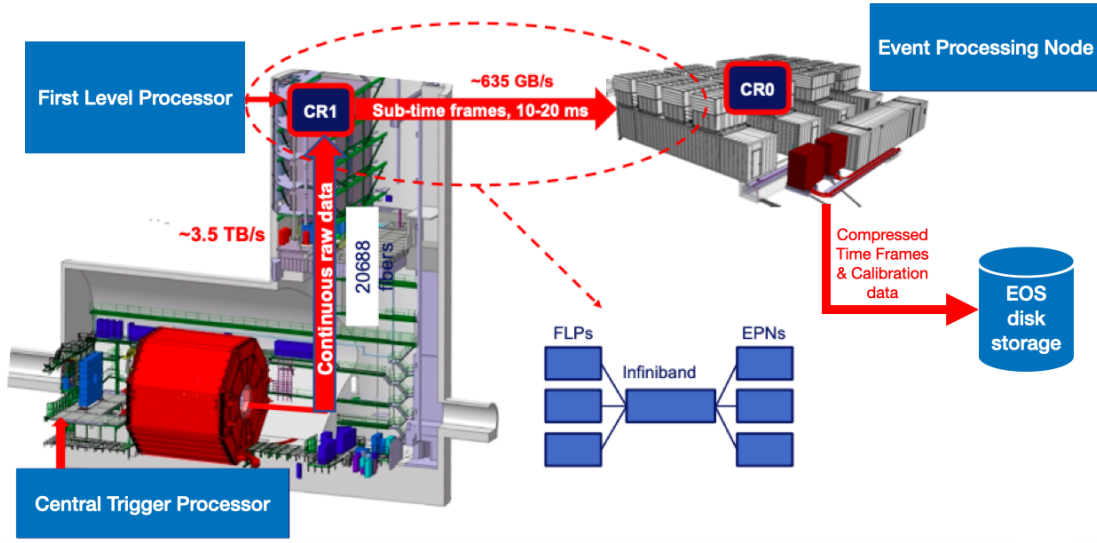
Figure 2.6: The O^2 data taking and synchronous processing. Adapted from [29].

2.4   The Worldwide LHC Computing Grid

The new O^2 system is part of the overall ALICE computing model, including Grid resources, as shown in Figure 2.8. Grid computing is one of the key distributed computing paradigms for supporting scientific research. The challenging management of LHC data is based on a distributed Grid infrastructure of the WLCG [30] that provides resources and supports the smooth functioning of the LHC experiments. The WLCG is arranged in layers termed "tiers" and includes more than 170 computing centres distributed in 42 countries worldwide, as shown in Figure 2.7. Currently, it has a capacity of 1.5 EB of storage, 1.4 million CPU cores, and peak transfer rates of 60 GB/s [31]. More than 2 million tasks run daily on the WLCG. Each WLCG Tier is mainly specialised for the given role.

The CERN Data Centre, located in Meyrin, Switzerland, is the centre of WLCG and represents Tier 0. Until recently, Tier 0 also provided the Wigner Research Centre for Physics in Budapest. CERN Data Centre has more than 10000 servers, more than 400000 processor cores, about 100000 disks, and more than 40000 tape cartridges [31]. Due to increased computing needs, it is planned to build an additional energy-efficient CERN data centre in Prévessin, France, to provide computing resources for the HL-LHC. Tier 0 takes part in the asynchronous reconstruction. Selected raw data are transferred to the CERN Tier 0 Data Centre for processing and archival storage. The network connection between ALICE and CERN Data Centre was 160 Gbps and after the upgrade, it is 1200 Gbps. Tier 0 delivers raw and reconstructed LHC data to Tier 1 and reprocesses data when LHC is not operating. Tier 1

consists of 14 computing centres. These centres primarily offer resources to store data from Tier 0 and for further data processing. Tier 0 and Tier 1 are connected with LHC Optical Private Network (LHCOPN) [32]. High bandwidth network connectivity facilitates storage on both disk and tape media.
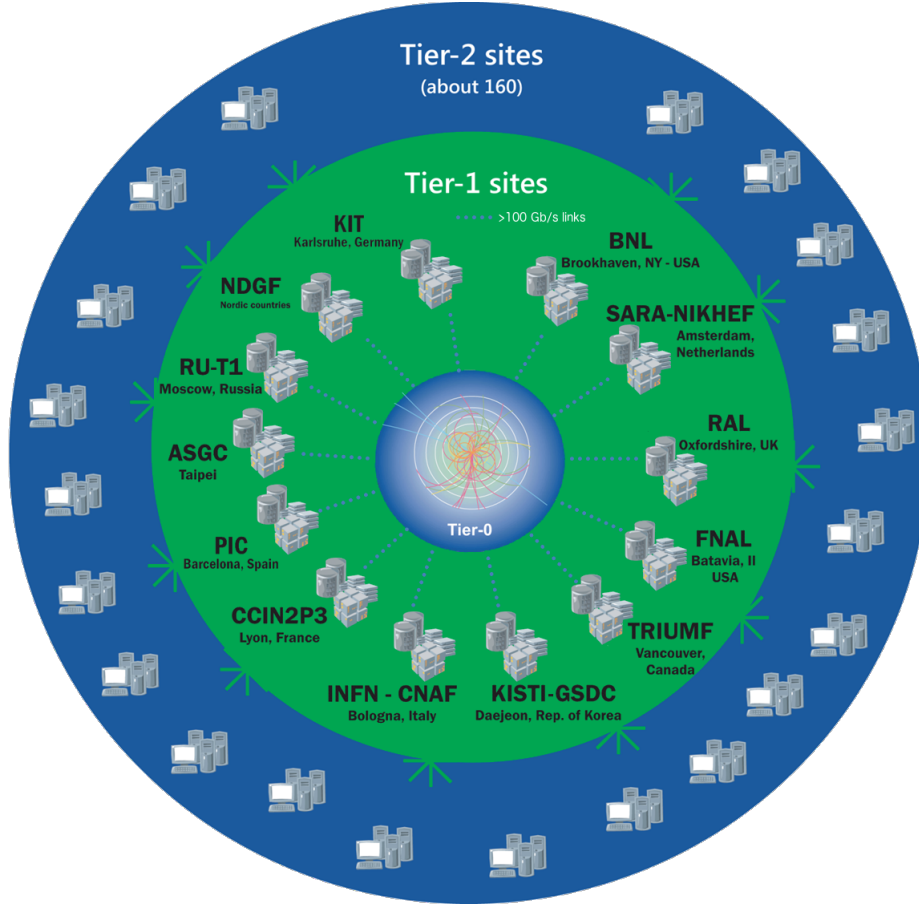
Figure 2.7: The WLCG tiers organisation in 2021 [4].

The Tier 2 level consists of resources provided by well-networked universities or scientific institutions organised either as separate data centres or federations of data centres. The basic requirements set for Tier 2 level centres are availability and reliability. Compute-intensive Monte Carlo simulation, associated reconstruction, and production of AODs are executed on Tier 2 sites. It is planned that Tier 0 and Tier 1 contribute to simulation when there is no activity. The LHC Open Network Environment (LHCONE) [32] supports job and data transfers between Tier 1 and Tier 2 sites. Dedicated Analysis Facility (AF) sites are responsible for storing AODs produced in asynchronous processing on other Tiers and running organised analysis jobs with high efficiency. The final AODs from Monte Carlo are sent to AF and associated Tier 1 for archiving. All processed CTF and AOD data are going to be removed from O^2 and Tier 1 disks to prepare space on resources for the following data taking period.
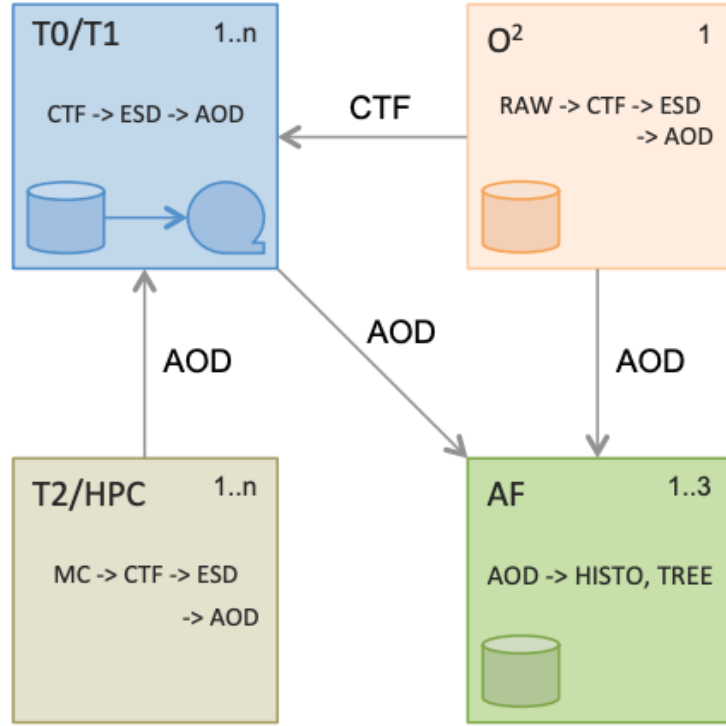
Figure 2.8: The ALICE computing model in Run 3 [3].

The LHC experiments have developed workflows and data management software to handle the allocated WLCG resources. CernVM File System (CVMFS) is used for the experiment software distribution to the WLCG nodes. The ALICE experiment takes approximately 20% of the total WLCG resources. Computing Resources Scrutiny Group (CRSG) [33] data on the Grid resources allocated to ALICE in Run 2 are shown in Table 2.1. ALICE uses resources from 1 site on Tier 0, 8 sites on Tier 1, and 22 federation sites on Tier 2. The data storage resources (disk and tape) are expressed in terabytes (TB), while the CPU is expressed in HEP-SPEC06, a standard HEP benchmark [34]. HEP-SPEC06 (HS06) benchmark is used to evaluate the CPU performance of WLCG sites tuned for the HEP domain but has also been adopted by other communities. This reproducible benchmark adopted by WLCG in 2009 is based on the industry-standardised Standard Performance Evaluation Corporation (SPEC) CPU2006 suite to compute resources. CPU consumption is calculated in HEP-SPEC06 seconds (HS06s) for all LHC experiments. The average CPU core power is 10 HS06. HS06 benchmark is going to be replaced by a benchmark for heterogeneous resources (HPC and non-x86 resources) that better correlates with today's LHC HEP workloads. Replacing HS06 with HepScore is being considered. As seen from the table, a significant increase in used resources is present.

Data sizes for different data types expressed in kilobytes (kB) for collisions of p-p, p-Pb, Pb-Pb, or Monte Carlo processing in Run 3 and Run 4 are shown in Table 2.2.

Table 2.1: Types and quantities of resources delegated to ALICE in Run 2 (disk and tape in terabytes, CPU in HEP-SPEC06). Derived from the ALICE reports.

TIER	RESOURCE	2015	2016	2017	2018
TIER 0	**Disk**	9400	13300	19300	28000
	Tape	18400	25500	29700	41400
	CPU	127000	218000	389000	541000
TIER 1	**Disk**	10100	17400	18245	27400
	Tape	11300	18500	22300	35800
	CPU	190000	253000	295000	340000
TIER 2	**Disk**	11500	14000	20060	25600
	CPU	200000	255000	299000	311000

Table 2.2: Data sizes for data types produced in collisions (in kilobytes). Adapted from [3].

	p-p	**p-Pb**	**Pb-Pb**
CTF	50	100	1600
ESD	7.5	15	240
AOD	5	10	160
MC	50	100	1600
MCAOD	15	30	480

Table 2.3 highlights the data of pledged resources on Tier 2 for 2022, the first year of the Run 3 period, as this research considers processing on Tier 2.

ALICE Collaboration has developed systems and frameworks to manage and monitor the distributed environment around the world needed to perform the ALICE physics programme. Each WLCG site member of the ALICE Collaboration provides a machine dedicated to running Virtual Organisation (VO)-specific services called VoBox. MonALISA [35], [36] is a monitoring system for ALICE distributed computing environment, including computing facilities, storage systems, and data transfer applications, that collects and aggregates telemetry

events and monitoring information in near real-time from distributed computing data centres. Each VoBox has MonALISA services installed, which are used to gather monitoring data from each running process, transfer, or service. MonALISA is an agent-based framework that automates Monte Carlo production jobs and analysis jobs by submitting them to AliEn. Agents registered as dynamic MonALISA services are running at sites around the world. Each site runs the MonALISA services that collect data and information from the local AliEn services that provide Computing Element (CE) or/and Storage Element (SE). Access to data on SE is enabled through the XRootD protocol. The AliEn workload and data management systems have been used for the distributed Monte Carlo event production, reconstruction, and analysis. The AliEn production environment consists of several middleware tools and services built around a central task queue. It makes use of the resources that have already been deployed in the WLCG infrastructures and services. The AliEn job brokering model based on pilot jobs enables flexible fair share distribution of jobs.

The AliEn job management services compose a three-layer lightweight system [25]:

- AliEn Central Services - for managing the whole system and distributing the workload

- AliEn Site Services - for managing the interfacing to local resources and Grid Services

- JobAgents - for operating on WNs and taking payload from the main Task Queue, a central database of all submitted jobs.

Task Queue is scanned by the number of Job Optimisers, which split the jobs into sub-jobs based on the location of the required input data or user-defined criteria. This optimisation has to ensure fair share and priority policies. The system then submits generic pilots to the computing centres' batch gateways. Job is assigned only when the pilot wakes up on the worker node. The job agents are started on the worker node to download and execute the payload from the central Task Queue.

The site running AliEn services sends monitoring data regularly to the local MonALISA service running on the site through the ApMon library [36], as shown in Figure 2.9.

Data from all the services, jobs, and nodes are aggregated and displayed in the MonALISA GUI Client every few minutes. The aggregated data are collected for long-term histories.
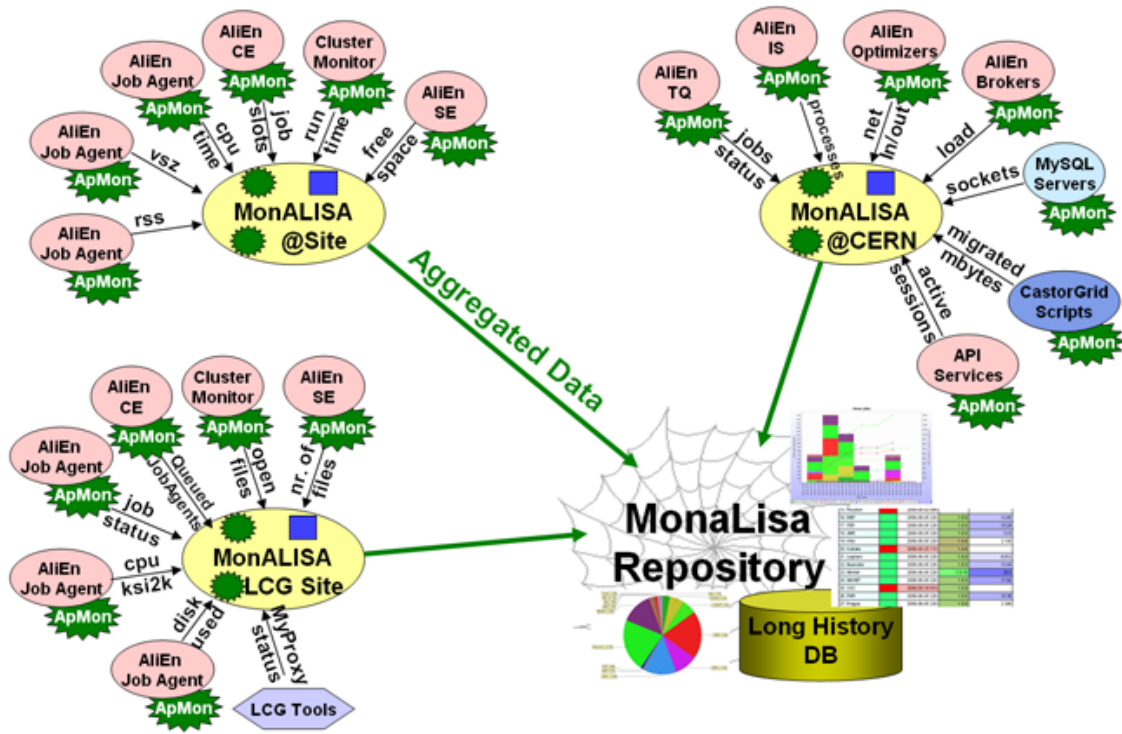
Figure 2.9: The ALICE Grid monitoring [37].

It was necessary to adapt the AliEn data catalogue to new requirements and improve its horizontal scalability, ensuring the consistency of data reading and a high replication factor. The JAliEn (Java ALICE Environment) middleware is developed for Run 3 to run more jobs.

The AliEn services layout is being reimplemented in Java as the new JAliEn framework [38] expands the AliEn interaction model with the following:

- JCentral - central component taking care of the data management and job queue

- JSite - site service for connection multiplexing and caching running on trusted site machines

- JBox - end-user (or job) service handling security matters, client-side authentication, and upstream connection running on worker nodes.

AliEn's MySQL-based file catalogue is replaced with JAliEn's in-memory NoSQL-based file catalogue designed to handle the growing processing load.

2.5   ALICE physics productions

The physics production of the ALICE experiment can be divided into the online and offline parts or raw and Monte Carlo data production, as shown in Figure 2.10. ALICE production is implemented according to a predefined physics programme that defines energy, luminosity, number of events, and collision types.
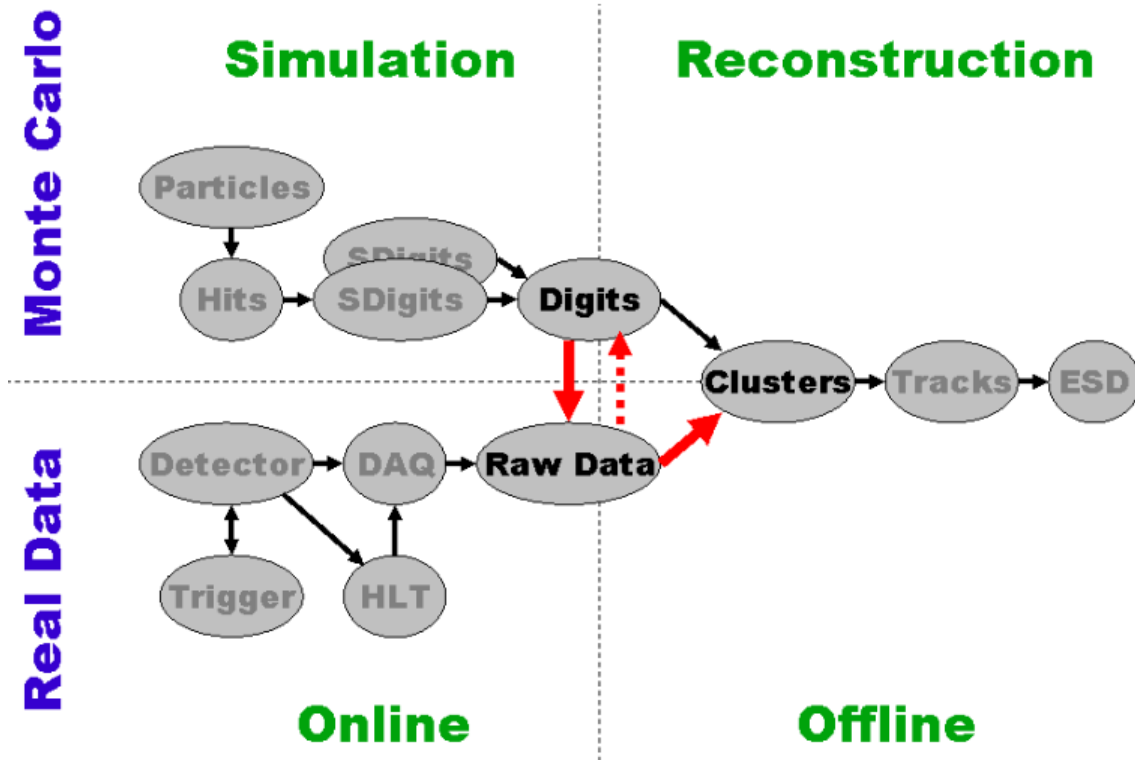
Figure 2.10: The ALICE data production [39].

2.5.1   RAW data production

In the online part, raw data are generated as a result of the activity of the detector and the collision of particles in it. Groups of subdetectors (presented in Section 2.2) and other complex computer systems (described in Section 2.3) participate in the raw production. Different colliding systems, p-p, p-Pb, and Pb-Pb, are part of the ALICE physics programme. Among them, for ALICE, Pb-Pb collisions are of particular interest. In these collisions, an enormous amount of particle multiplicity occurs.

ALICE raw production in Run 2 is shown in Figure 2.11. In Figure 2.11, it can be seen that the processing of collected data continued on the WLCG during the LS2 period. Visualised data are extracted from the MonALISA framework [35].
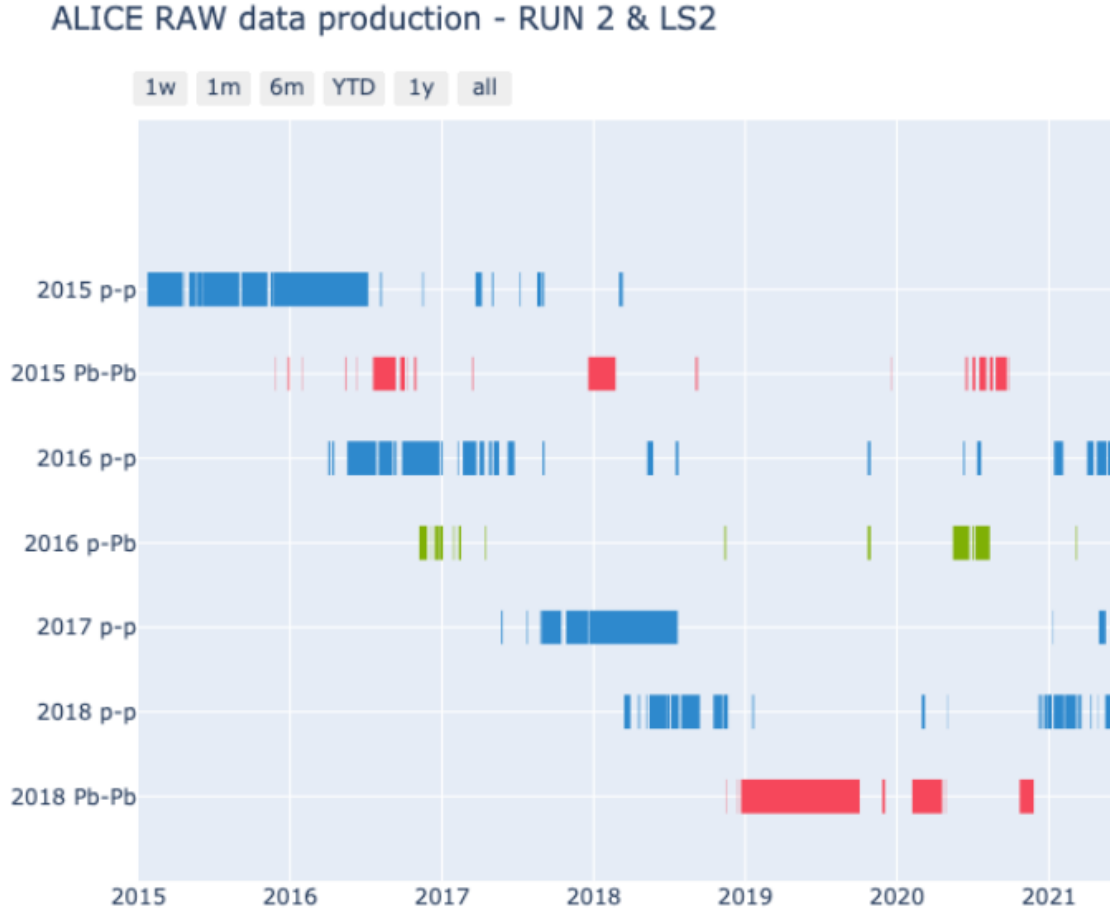
Figure 2.11: Raw data production in Run 2 (2015 – 2018) and processing steps during LS2 (2018 – until June 2021). Data derived from the MonALISA.

2.5.2   Monte Carlo data production

Numerous studies at ALICE rely heavily on Monte Carlo simulations [40]. The objective of Monte Carlo simulations is to provide and predict the outcome and properties of the event after collisions as accurately as possible. Monte Carlo simulations are crucial for performing data analysis during which data are examined, cleaned, and transformed using highly specialised tools and algorithms. Monte Carlo simulations are employed to simulate primary collision and collision geometries ranging from central to peripheral collisions to estimate the detector efficiencies, optimise detector design, and verify analysis performance. In their research, physicists involved in the ALICE experiment compare experimental results with theoretical forecasts. ALICE produces almost the same number of Monte Carlo events as the raw data. The information included in Monte Carlo events can be used in physics analyses. The Monte Carlo production workflow consists of several stages, as shown in Figure 2.10. The AliRoot

framework has supported Monte Carlo simulation at event generation, transport, digitisation, and event reconstruction.

Event generation is the first stage of Monte Carlo production. The type of particles and their decays need to be defined to generate events of particular interest. The main event generators used in production by ALICE are PYTHIA [41] (for p-p events), DPMJET [42], and HIJING [43] (for Pb-Pb events). Event generators simulate particle collisions as seen by a detector, with theoretically predicted distributions to resemble real collisions. Generated particles are described by the information needed for particle identification, particle type, particle momentum, charge, mass, and vertex position. The particles produced by the generator phase are then tracked through the ALICE experimental setup. This process is delegated to the transport libraries such as Geant3 [44], Geant4 [45], and Fluka [46]. The output of the transport phase are the hits produced in the detectors. Hits give precise details on a particle's path through a detector, including its position and energy deposition. The summable digits signal from the detector, which correlates to the raw data, is then created from these hits. The phase of creating summable digits is followed by the digitisation process, during which the digits are produced and stored. Digits use real thresholds and contain information similar to one obtained in real data taking.

The reconstruction chain of the experimental data and Monte Carlo samples is the same. It consists of local reconstruction, followed by the primary vertex, then tracking, and finishes with the secondary vertex. ALICE computing model envisages three reconstruction passes, depending on the computing budget. The reconstruction uses the digits as input in a special ROOT format or as raw data. During local reconstruction, each detector performs clusterisation. The single cluster consists of particle signals crossing the sensitive area detected by neighbouring detector elements. Then, the primary vertex is reconstructed using silicon pixel detectors in the two ITS inner layers, and the track candidates, called seeds, are found. To achieve precise primary vertex reconstruction, tracking for the ALICE central tracking detectors is done using the Kalman Filter [47]. The track finding in the TPC starts at the outer part, where the track density is minimal. Propagation of the state vector and the covariance matrix go towards the smaller TPC radii. Tracks prolongate to the ITS and then outward direction to the outer radius of the TPC, followed by propagation to the TRD, TOF, HMPID, EMCal, and PHOS, where they acquire the PID information. Finally, reconstructed tracks are refitted backward to the primary vertex with the Kalman filter to obtain the track parameters' values at or near the primary vertex. For the secondary vertex reconstruction, the tracks that made it through the final refit toward the primary vertex are merged and utilised. They are candidates for a secondary decay vertex if their closest approach distance is less than a certain threshold and the closest approach point is placed before the initial measured points on both tracks. Reconstructed tracks can be compared with the particles produced by Monte Carlo simulation. Reconstruction output comes in the ESD format, which contains all the information about the event or a list of reconstructed tracks/particles and global event properties.

The final stage of event processing is the analysis. ESDs are further reduced to the AODs by applying some standard physics cuts to remove unnecessary information for the physics analysis. Most of the physics analyses are performed on the AODs. Analyses tasks that process the same set of input data are chained and organised as analysis trains. ALICE analysis framework reads each event only once, and different algorithms can be applied to it according to the needs of the ALICE user. Dedicated Physics Working Groups (PWGs) coordinate scheduled analysis and propose large-scale productions that need to be approved by the ALICE Collaboration.

The new O^2 software framework based on the ALFA and FairRoot [3] contains new simulation and reconstruction code and algorithms for Run 3 with main changes in detector geometries and digitisation.

The visualisation of performed analysis of ALICE Monte Carlo production, mainly running on Tier 2 sites during Run 2 and LS2, is shown in Figure 2.12.
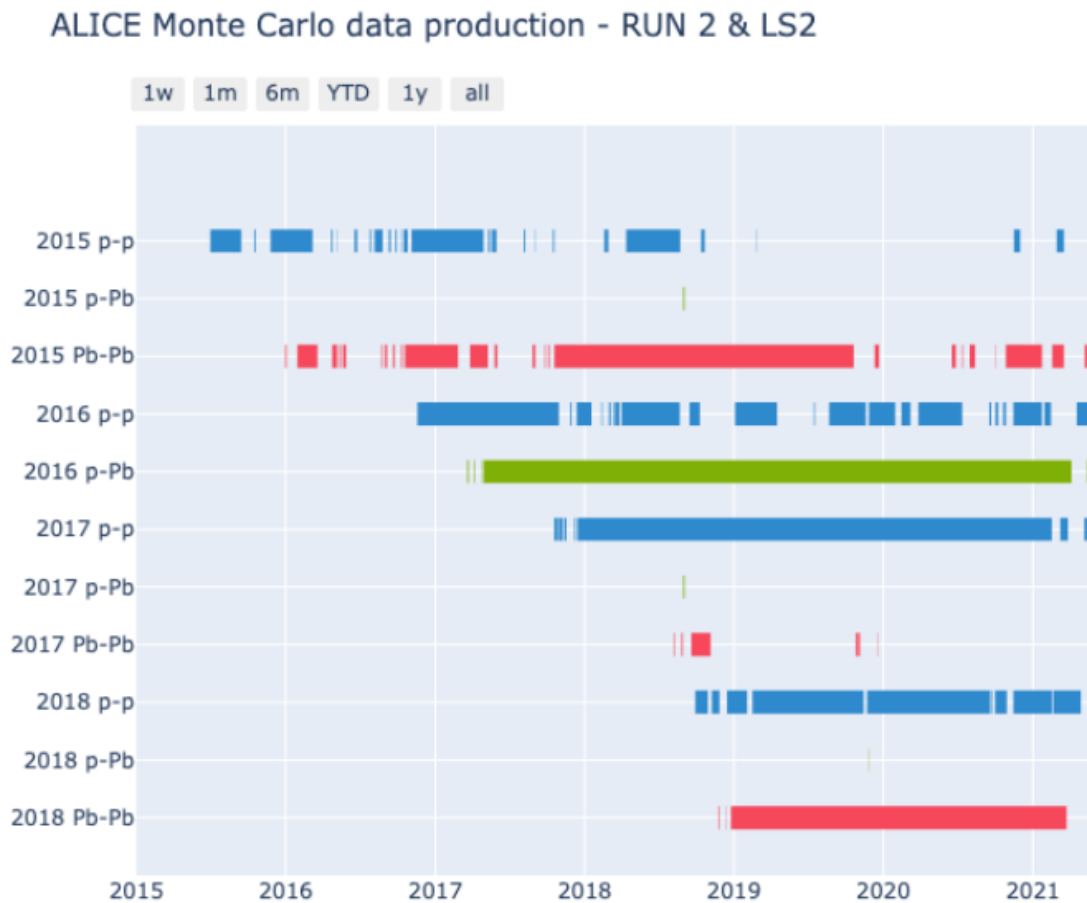
Figure 2.12: Monte Carlo production during Run 2 (2015 – 2018) and LS2 (2018 – until June 2021). Data derived from the MonALISA.

Monte Carlo simulations are one of the most computationally demanding jobs. In Run 1 and Run 2, Monte Carlo simulation jobs were submitted as batch jobs to the AliEn system and distributed for execution mainly on WLCG Tier 2 sites according to their resource availability. The WLCG performs many tasks in parallel. Reconstruction tasks, Monte Carlo production and data filtering follow an ordered schedule, while single-user Monte Carlo productions and data analysis are unpredictable. The CPU usage per job type during RUN 2 and LS2 can be seen in Figure 2.13. The Monte Carlo production jobs take more than 60% of the total CPU usage on the WLCG.
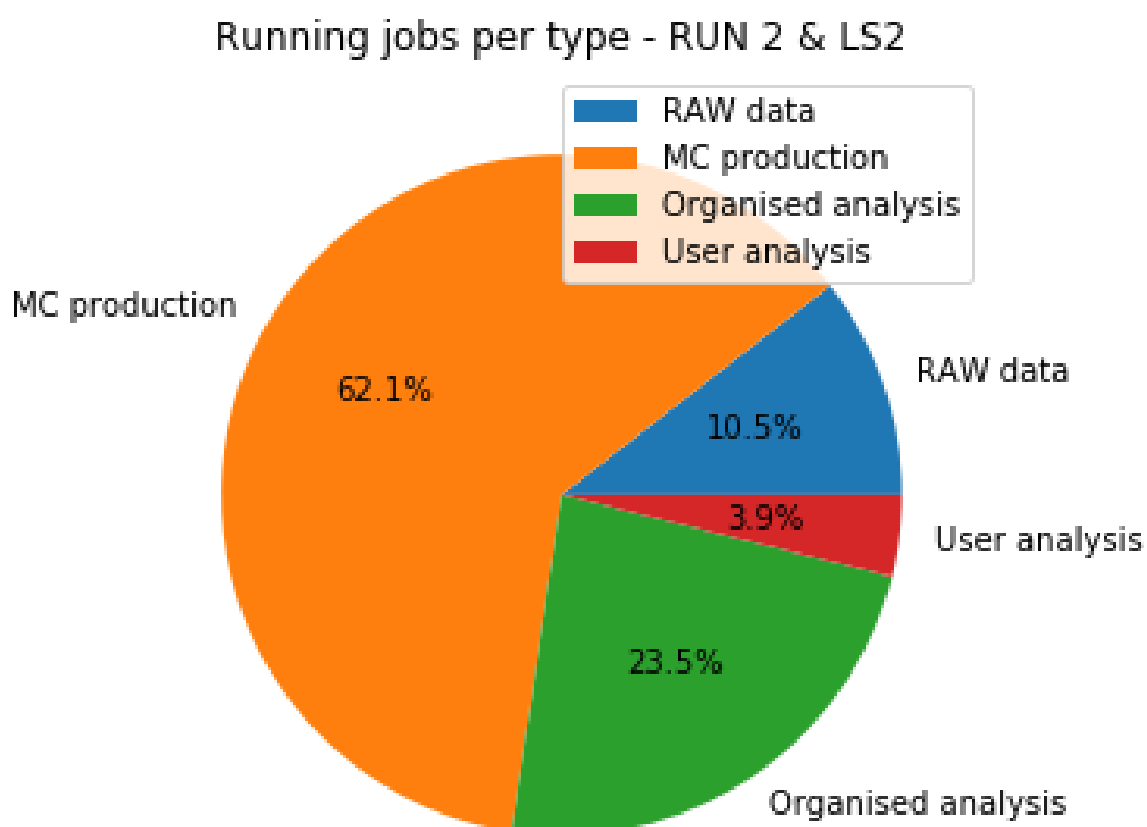
Figure 2.13: Running ALICE jobs during Run 2 and LS2. Data derived from the MonALISA.

Table 2.3: Pledges provided by the ALICE Tier 2 federations in 2022 (disk pledges in terabytes, CPU pledges in HEP-SPEC06). Derived from the ALICE data.

Federation	Country	Type	Pledge	Federation	Country	Type	Pledge
AT-HEPHY-VIENNA-UIBK	Austria	Disk	550	PL-POLISH-WLCG	Poland	Disk	1900
		CPU	4800			CPU	22000
CZ-PRAGUE-T2	Czechia	Disk	2200	RO-LCG	Romania	Disk	10500
		CPU	20000			CPU	39000
DE-GSI	Germany	Disk	5200	RU-RDIG	Russian Federation	Disk	3143
		CPU	49700			CPU	33660
FR-GRIF	France	Disk	2023	SE-SNIC-T2	Sweden	Disk	/
		CPU	21270			CPU	2820
FR-IN2P3-IPHC	France	Disk	480	SK-TIER2-FEDERATION	Slovakia	Disk	1200
		CPU	7000			CPU	7800
FR-IN2P3-LPC	France	Disk	419	T2-LATINAMERICA	Latin America	Disk	600
		CPU	7000			CPU	16300
FR-IN2P3-LPSC	France	Disk	/	T2_UNAM	Mexico	Disk	500
		CPU	3972			CPU	4900
FR-IN2P3-SUBATECH	France	Disk	2500	UA-TIER2-FEDERATION	Ukraine	Disk	200
		CPU	13500			CPU	3000
HU-HGCC-T2	Hungary	Disk	1290	UK-SOUTHGRID	United Kingdom	Disk	1010
		CPU	12570			CPU	10619
IN-DAE-KOLKATA-TIER2	India	Disk	3000	US-LBNL-ALICE	United States	Disk	8400
		CPU	50000			CPU	82000
IT-INFN-T2	Italy	Disk	8820	ZA-CHPC-T2	South Africa	Disk	1250
		CPU	92700			CPU	12000

3 BACKGROUND AND LITERATURE REVIEW

This chapter deals with current trends and essential aspects of processing in the Cloud. Challenges in this area are directing the modelling of a heterogeneous Cloud infrastructure platform adaptable to HEP data requirements for an operationally efficient geo-dispersed heterogeneous system. This chapter identifies the fundamental concepts, components, and methods for simulating ALICE Monte Carlo data production on a heterogeneous Cloud. Then, the Cloud computing simulators are analysed, and the most convenient Cloud simulator for this research is explored. The literature review and conclusions on scheduling strategies for achieving scalability in the heterogeneous Cloud are provided.

3.1 Heterogeneous Cloud computing

The distributed computing paradigms have marked computer science and enabled the successful conduct of scientific research and the work of a large organisation. Within the research of this thesis, an analysis of the development of distributed computing paradigms was conducted [48], as shown in Figure 3.1. These distributed computing paradigms are: Cluster computing, Grid computing, Cloud computing, Fog computing, and Dew computing. The physical infrastructure of distributed computing paradigms is based on a distributed computing model that involves storing data in various networked locations, distributed database systems, and various analytical tools and applications. The physical infrastructure must meet the criteria of scalability, openness, transparency, and QoS. Large scientific experiments have organised and used their resources as one of the computing paradigms. An example is CERN that uses computer and network resources through the WLCG.

Cloud computing has revolutionised computer science with optimal and flexible use of the pool of configurable computing Infrastructure as a Service (IaaS), Platform as a Service (PaaS), or Software as a Service (SaaS) over the Internet. Cloud computing is based on virtualised IT infrastructure pooled and divided regardless of the limits of physical hardware using special software to realise maximum utilisation and cost savings of the data centre resources. The United States National Institute of Standards and Technology (NIST) defines Cloud computing [49] as:

"...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. This Cloud model comprises five essential characteristics, three service models, and four deployment models." Cloud deployment models are public, private, hybrid, and community Cloud. Essential Cloud services characteristics provisioned by virtualisation and automation enablers and described by NIST are:

- on-demand self-service

- broad network access

- resource pooling

- rapid elasticity

- measured service.

Cloud services have significantly evolved from the NIST definition of Cloud computing. In addition to the core services defined by NIST, IaaS, SaaS, and PaaS, a wide range of services is getting developed. For marketing purposes, services add the suffix "as a Service (aaS)" and are collectively named Everything as a Service (XaaS) [50]. In addition, Cloud computing serves as a platform for developing and collaborating with new paradigms such as Fog, Edge, and Dew computing that play an important role in IoT and the realisation of fifth-generation mobile networks and services [51], [52], [53].
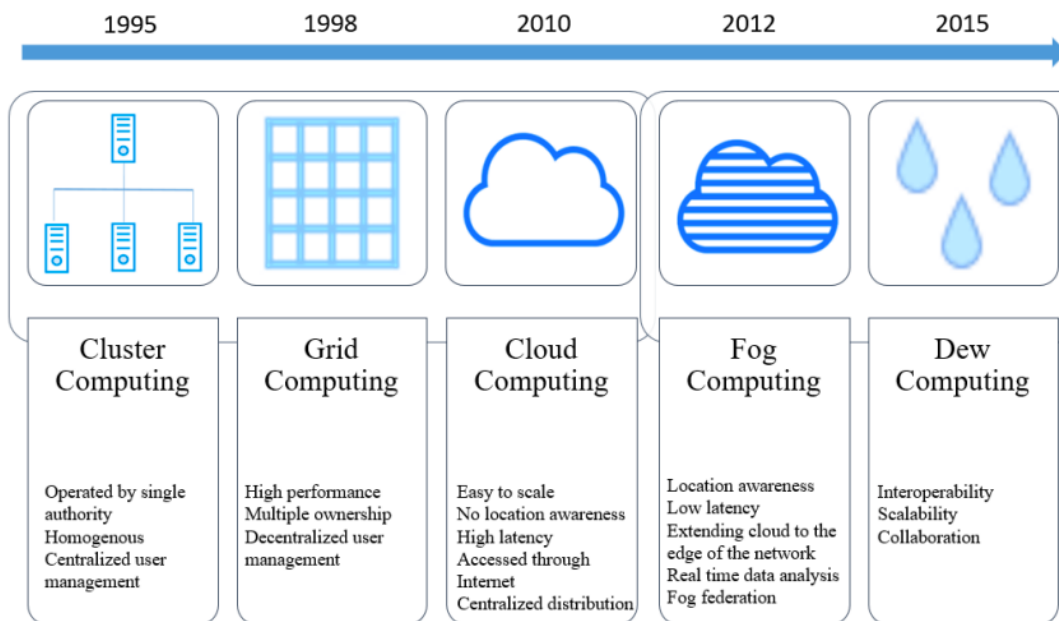
Figure 3.1: Distributed computing paradigms timeline [48].

Cloud computing has been a ubiquitous and accepted choice for data storage and processing. Cloud is a cost-effective platform for highly scalable database searches, distributed storage, and processing over the Internet. The shift towards using the potential and flexibility of Cloud computing to process large amounts of data in different disciplines is increasing and tends to increase further. Cloud computing and modern techniques such as artificial intelligence, machine learning, and HPC platforms are integral elements of digital transformation and infrastructure modernisation projects. The use of Cloud services for workload migration, data analytics, and improving data centre efficiency is continuing to expand as a result of digital transformation. The Cloud services market is also growing, dominated by Amazon Web Services, Microsoft, Salesforce, Google, and Oracle. Current research and statistics [54] show the growth and rapid progress of Cloud technologies.

As technology advances, the traditional Cloud environment based on homogeneous infrastructure now evolves into the modern heterogeneous Cloud data centres [54], [55], [56]. Heterogeneity is a broad concept. The heterogeneity concept is manifested in the diversity of resources, whether it is hardware platforms produced by various manufacturers or different servers having different processing, memory, storage, and networking capacities. The term "heterogeneity" in this context refers to the variance in the computing power, storage capacity, and networking capabilities of various servers. The study [57] shows that with the rapid growth of Big Data, the need to exploit the potential of HPC is growing. Therefore, HPC platforms are moving to Cloud-based architectures [58]. The efficient exploitation of the heterogeneous Cloud infrastructure has become a topic of great importance in recent years. The availability of heterogeneous Clouds makes possible the deployment of a wide range of multiple specific applications and workloads, both data and compute-intensive. As a result, more and more applications and services are moving to the Cloud [58], [59].

These heterogeneous features and diversity support a wide range of applications but have challenges related to the complexity of managing such an environment. Clouds based on heterogeneous computing infrastructures may have specific requirements on the network. Combining several interconnected topological architectures can simultaneously ensure scalability and high performance. The complexity introduced by heterogeneous Clouds cannot be solved with conventional approaches to resource management. That requires a combination of multiple service abstraction modes to do appropriate mapping between application requirements and the resource characteristics done at both hardware and software levels. Improving performance while preserving cost-effectiveness can be achieved through scaling. This research is focused on the execution of ALICE simulation jobs and the processing of scientific data in heterogeneous Cloud data centres. A proper task scheduling algorithm should improve performance and ensure scalability when processing ALICE simulation jobs in heterogeneous data centres in the Cloud.

3.2   Scalability

Scalability is the primary focus of system design and a core concern in large-scale computing. Over the past decades, there have been several attempts to define scalability uniformly. The etymology of "scalability" derives from the Latin word *scala, scalae*, meaning stairs or ladder. This Latinism is accustomed in all European languages. In computing, it means variable in size, the ability to significantly and relatively effortlessly increase or decrease capacity according to a change in requirements. Scalability issues are related to resource demands and managerial issues.

Scalable use of resources is a complex process that should enable the optimal use of all available resources. Resources should be allocated where and when needed, with decision-making based on data. Data volume puts scalability at the centre of Big Data processing and storage. As a multidimensional problem, it requires the development of models, structures, and technological solutions. Scalability relates to other system qualities, like throughput, latency, efficiency, and cost, that have to be optimised simultaneously. Adaptability to dynamic changes is one of the main factors in the design of distributed systems and parallel algorithms. Scalability testing is used to evaluate scientific computing environments and technologies developed to handle Big Data. Therefore, it is necessary to further investigate and consider the correlation of scalability with other factors in the field of Big Data and to improve and optimise the scalability property in new modern technologies. Hill [60] analysed scalability as an attribute for describing multiprocessor systems and decision-making on architectural design. He concluded that there was no generally accepted definition and that scalability as a term should be used with caution.

In [61], an attempt was made to identify attributes important for designing a scalable system. Structural scalability, load scalability, space scalability, and space-time scalability are considered interdependent general types of scalability. Therefore, these aspects should be distinguished to recognise whether they limit growth or affect performance.

Then, the research project on scalability [62] agreed that there was no uniform definition for a scalable real-world system. The authors started with the definition of scalability as the ability to handle an increased workload. They studied two aspects of scalability, the aspects of hardware use and software use, all without adding resources to a system and by repeatedly applying a cost-effective strategy for extending a system's capacity. The authors analysed the causes of scalability failure related to the resource limits, such as network bandwidth or memory. Achieving scalability is inseparable from attributes like performance, usability, or cost. The scalability audit concept comprises potential resource bottlenecks, incorrect scaling assumptions, general scaling strategies, and scalability assurance methods. These factors are to be applied to evaluate whether the system's architectural design is scalable. The relationship between the maximum demand that the system can meet (D) and the cost-effectiveness (K) of an idealised and more realistic scalable system is shown in Figure 3.2. An idealised

scalable system that is entirely cost-effective is extreme. A system that gradually improves cost-effectiveness as demand grows is more realistic. The region where the system is highly scalable is its "sweet spot."
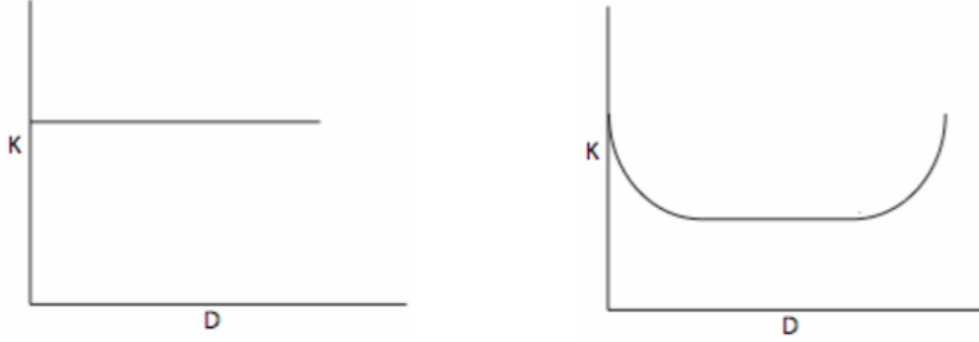
Figure 3.2: System demand (D) vs. cost-effectiveness (K) of the idealised (on the left) and more realistic (on the right) scalable system [62].

Scalability testing is an essential phase of system or service development. It is required to collect and measure relevant scalability performance metrics and interpret metrics influence to test scalability and achieve better performance for increasing workload demands.

The majority of the scalability metric proposals for homogeneous systems are based on two scalability metrics: isospeed and isoefficiency. These metrics are derived from Amdahl's law tied to efficiency and speedup.

Isospeed [63] depends on the number of processors needed to keep the efficiency constant while increasing the problem size. The isospeed scalability metric defines an algorithm-machine combination as scalable if the achieved average unit speed remains constant with increasing numbers of processors, provided the problem size is increased proportionally. The isospeed scalability function is defined as $\psi(N,N') = \dfrac{N'W}{NW'}$ where N and N′ are the initial and scaled number of employed processors, respectively, W and W′ are the initial and scaled problem size (work), respectively. The isoefficiency scalability function expresses the system's ability to keep the parallel efficiency constant when the system and problem size increase [64]. Highly scalable systems have small isoefficiency function. The parallel efficiency (E) is limited by speedup (S) and the number of processors used (p), $E = \dfrac{S}{p}$. The speedup of a parallel system is defined as the ratio of sequential execution time and parallel execution time of an algorithm. Sun et al. [65] proposed an isospeed-efficiency metric to measure the scalability of the general computing environment, both homogenous and heterogeneous computing systems. Isospeed-efficiency combines the isospeed and isoefficiency through a concept called "marked speed" to describe the computing power for a single computing

node and a computing system as a constant for a study derived from the benchmarked speed of the nodes. Scalability Testing and Analysis System - STAS [66] implements an isospeed-efficiency scalability metric to provide the system's scalability analysis through four components: system characterisation, algorithm pre-analysis, scalability tester, and scalability analyser. It can describe the scalability of algorithms and systems and evaluate their performance.

Cloud-based applications must satisfy three non-functional requirements: scalability, elasticity, and efficiency [67]. In [68], the authors define scalability, elasticity, and efficiency. Workload and resources are the concepts most used in literature to define scalability. Cloud performance metrics in terms of scalability, elasticity, and efficiency are reviewed. Scalability should be distinguished from elasticity [69], [70]. Scalability and elasticity are significant non-functional Cloud properties. Scalability is the system's ability to support workload by increasing or decreasing available resources over a period of time, while elasticity is the system property manifested as a dynamical adaptation by provisioning and de-provisioning resources to changing workload demands at each point in time. According to [71] and [72], scalability is a prerequisite for elasticity. Weber et al. [72] highlight the difference between the definitions of scalability and elasticity. They state that scalability is a prerequisite for elasticity and that high elasticity implies appropriate resource allocation and usage. Varying load intensity is the input variable to which a Cloud system is subjected when analysing scalability. Service level objectives and the number of provisioned resources are used to evaluate the system's adaptability to varying load intensity. Inspired by elasticity measures, authors in [70] introduced two technical scalability metrics that address the volume of software instances and quality scaling performance regarding average response time. The proposed metrics were tested to assess Cloud-based software systems' scalability depending on demand scenarios. Metrics are quantitatively expressed as a ratio between actual and ideal values. Figure 3.3 shows the main concepts used to measure elasticity (resource demand/supply and under/over-provisioning). Steady increase and stepped increase are demand patterns used to test and achieve scalability [70], as shown in Figure 3.4.

Real-world Cloud-based systems are unlikely to exhibit ideal scalability. More realistic scenarios of Cloud systems are the stepwise changes of demand [70]. Analysing the scalability of a heterogeneous Cloud system requires an analysis of system design characteristics, workload characteristics, and a clear definition of scalability metrics for heterogeneous systems. The scalability of a system is achievable through horizontal or vertical scaling. Horizontal scaling or scaling out means adding more nodes or machines to the system to increase performance rather than increasing resources such as CPU, network, and storage by scaling vertically.

Most of the analysed scientific papers use workload, bandwidth, latency, resource allocation (CPU, memory, disk), time, robustness, speed, reliability, availability, efficiency, energy use, and price as parameters for evaluation and defining scalabil-
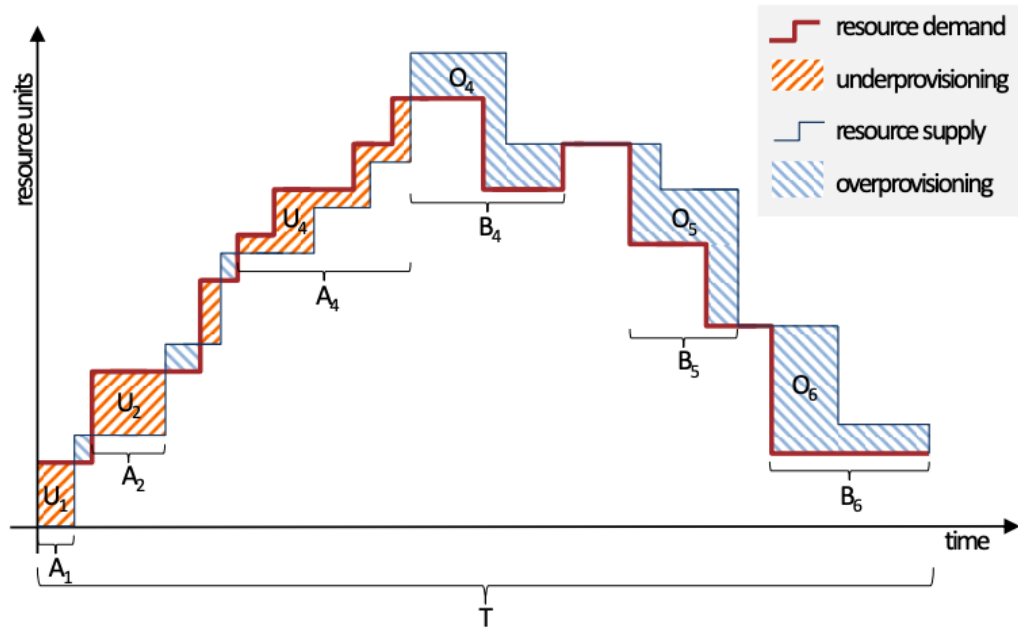
Figure 3.3: Elasticity metrics [71].
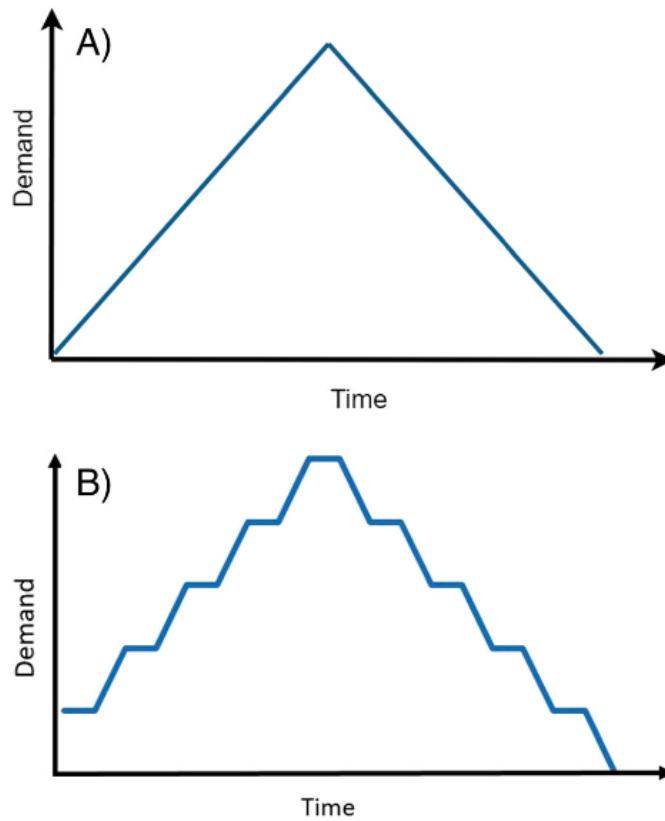
Figure 3.4: Scalability demand patterns: A) steady rise and fall of demand; B) stepped rise and fall of demand [70].

ity [68], [72], [73], [74], [75], [76]. The Cloud scalability measure is driven by QoS and productivity, to which scalability is directly proportional [77]. Productivity is also directly related to the problem size.

A framework [78] analyses the scalability of a software system as a multi-criteria optimisation problem, and scalability is defined as "a quality of software systems characterised by the causal impact that scaling aspects of the system environment and design have on certain measured system qualities as these aspects are varied over expected operational ranges." Therefore, scalable factors and dependent variables influencing the system's scalability must be identified, as well as objectives and metrics for unique scalability questions.

Introducing software-defined technology in data centres opens scalability challenges on both control and data planes [79], [80]. Scalability is a crucial property in the Software-Defined Networking (SDN) control plane characterised by the throughput and flow setup latency [81]. An overview of SDN scalability challenges and approaches for achieving higher performance at the control plane in different environments such as data centres, enterprise networks, campus networks, Cloud networks, and Wide Area Network (WAN) is given in [81].

Achieving scalability between several networked data centre locations in the Cloud depends heavily on applied resource management and scheduling approaches. The management of resources is based on different scheduling algorithms and metaheuristics for controlling the use of shared resources and achieving effective use of all sites' capabilities when placing tasks on processing. Resource provisioning and resource monitoring as scalable resource management components should consider and analyse aspects of heterogeneity, speed, and volume of the generated data over time.

3.3   Cloud system simulators

Modelling and simulation are approaches used in research and engineering to assess, study, and improve Cloud environments. These tools are considered essential for reproducing or verifying forecasts and analysing complex systems and various processes. Management of VMs and job allocation in heterogeneous data centres in the Cloud represent significant problems that require extensive usage of modelling and simulation. Today's computers and algorithms allow complex simulations of different systems and simulations of real-world and natural phenomena. Simulation techniques enable researching solutions for resource allocation problems and allow building large realistic models of organised systems without using a production system. Simulation as a research method has been used extensively for researching Cloud environments, resource utilisation, allocation policies, energy consumption, cost performance, and other aspects related to Cloud data centres. System performance can be evaluated concerning different parameters such as total time, cost, or network distance.

Several Cloud simulation frameworks have been developed to examine Cloud environment behaviour. These simulators have been built on different platforms focusing on various

features. Simulators relevant to this research area have been studied. It is often required to adjust a simulator for application to a specific research model. The characteristics of the simulators are presented in Table 3.1.

CloudSim [85] and OMNeT++ [92] are the platform bases for various research related to Cloud computing. These discrete event simulators have been adapted to different research needs.

After analysing the Cloud simulation frameworks, the Discrete-Event Simulator CloudSim was selected. CloudSim is a frequently used Java-based simulator suitable for research studies related to Cloud computing. This simulator enables modelling Cloud infrastructure, including policies for provisioning VMs to hosts, scheduling jobs and machines, modelling costs, and managing on-demand resource provisioning. CloudSim supports real application traces usage, containing tasks called cloudlets to compute the application's resource utilisation. Another reason for choosing the CloudSim simulator is the option of simulating a heterogeneous Cloud computing environment. CloudSim allows the simulation of different Cloud IaaS usage scenarios. This Cloud simulator is flexible enough to encompass common specificities of the hardware architecture through creating data centres, setting the data centre characteristics, creating data centre brokers, and creating VMs with different configurations for processing cloudlet workload. The CloudSim entities for modelling Cloud infrastructure and policies are shown in Figure 3.5.
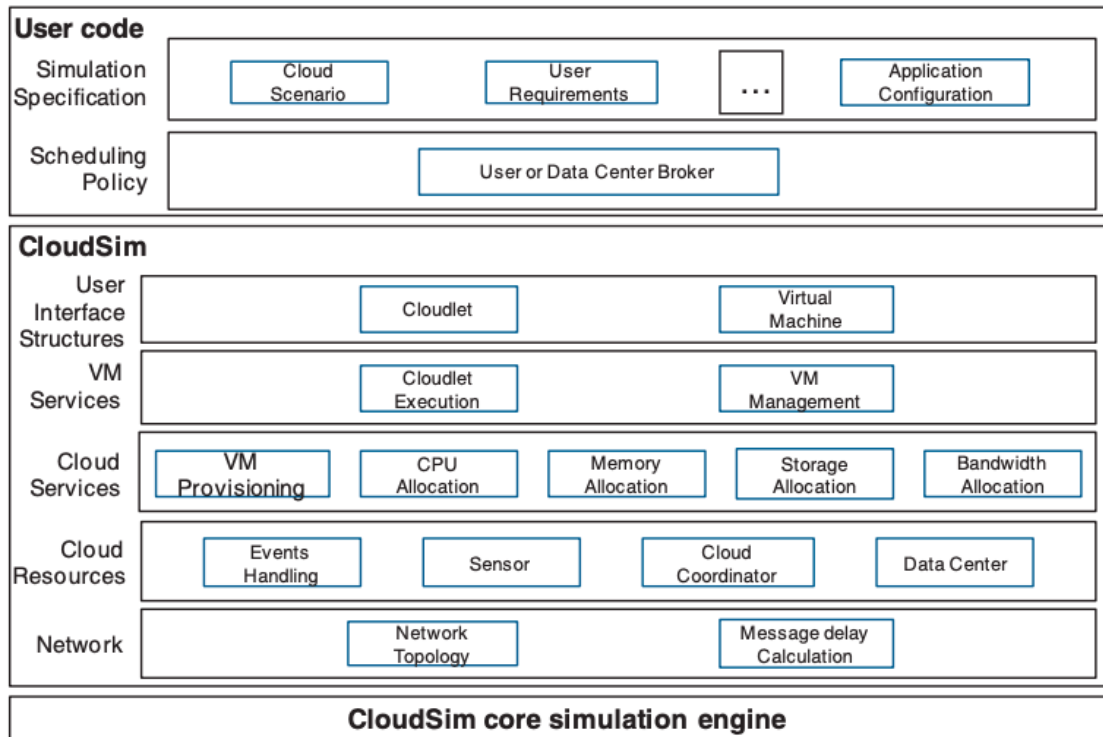
Figure 3.5: The CloudSim architecture [85].

Table 3.1: Characteristics of Cloud simulators.

SIMULATOR	UNDERLYING PLATFORM	PROGRAMMING LANGUAGE	OPEN-SOURCE SOFTWARE	SIMULATION PRINCIPLE	SCHEDULING POLICIES	NETWORK MODELLING	ENERGY MODELLING	COST MODELLING	GRAPHICAL USER INTERFACE
BigDataSDNSim [82]	CloudSim	Java	+	Event-based	+	+	+	+	-
CloudAnalyst [83]	CloudSim, SimJava	Java	+	Event-based	+	+	+	+	+
CloudNetSim++ [84]	OMNeT++	C++	+	Packet-based	+	+	+	+	+
CloudSim [85]	SimJava	Java	+	Event-based	+	+	+	+	-
CloudSimSDN [86]	CloudSim	Java	+	Event-based	+	+	+	+	+
DCSim [87]	-	Java	+	Event-based	+	+	+	-	-
GDCSim [88]	BlueSim	XML, C++	+	Event-based	+	-	+	-	-
GreenCloud [89]	NS-2	C++, OTcL	+	Packet-based	+	+	+	-	+
iCanCloud [90]	OMNeT++, MPI	C++	+	Packet-based	+	+	-	+	+
Mininet [91]	-	Python, C	+	Packet-based	+	+	+	-	+
OMNeT++ [92]	-	C++, NED	+	Event-based	+	+	+	-	+

3.4   Management and scheduling algorithms

The performance, reliability, and cost of Cloud systems are impacted by resource management. Effective Cloud resource management includes complex policies for optimising resource provisioning and resource scheduling that can be further categorised on several techniques [93]. A detailed survey on Cloud resource management was carried out in [94]. Resource management problems belong to the class NP-hard [95]. Thus, innovative, efficient, and adaptable solutions are required to organise sharing of geo-distributed Cloud resources among multiple users with different requirements. Scalability, horizontal as well as vertical, is affected by resource management policies. These policies can be categorised into several classes: admission control, capacity allocation, load balancing, energy optimisation, and QoS guarantees. Mechanisms to implement these policies are control theory, machine learning, utility-based, and market-oriented mechanisms [96].

Numerous load balancing strategies were analysed, with special emphasis on task scheduling. Task scheduling often addresses both the allocation and scheduling of tasks on VM resources. Different factors should be considered, from the heterogeneity of hardware and software systems to workflow requirements.

Various heuristic, metaheuristic, and hybrid algorithms [97] are implemented to solve task scheduling problems. Task scheduling is a process of selecting the site or Cloud for placing and running tasks and mapping the submitted tasks to available virtual resources based on task properties. Scheduling algorithms can be categorised as static and dynamic, offline and online, and preemptive and non-preemptive scheduling algorithms [93]. Static scheduling algorithms need information in advance and do not meet the needs of fluctuating Cloud computing workloads. Cloud resource management requires dynamic algorithms to optimise in real-time and adapt decisions to the current requirements and state of the resources. Offline or batch processing does not require user intervention, and resource-intensive jobs organised in batches are usually allocated on resources to run when the system is less loaded. Preemptive task scheduling can remove a running task from the allocated resource and assign it to another Cloud resource. Non-preemptive scheduling does not interrupt the running task on the allocated resource until the task is finished. Workflow and task scheduling algorithms are classified according to the Cloud environment, VM type, dataset and application type, execution time, cost, and power [98]. Certain prescribed objective functions must be satisfied by task scheduling when processing tasks on different Cloud sites, like minimising the makespan and total completion time while achieving load balancing at each site and consequently improving scalability and efficiency. There are many stochastic optimisation approaches to this problem coping with variables whose values vary with time.

Inspiration for solutions to problems from the discrete or continuous domain can be found in natural, evolutionary, biological, physical, or social systems. The most common task scheduling implementations are based on: First Come, First Served, Round-Robin,

Shortest Job First, Minimum Execution Time, Max-Min, Min-Min, Heterogeneous Earliest Finish Time, Earliest Deadline First, Tabu Search, Genetic Algorithm, and Particle Swarm Optimisation (PSO). Many proposed algorithms are based on and compared with these reference algorithms.

Dynamic algorithmic techniques from the Artificial Intelligence (AI) field are being introduced in Cloud task scheduling. AI algorithms are viewed as capable of adapting to the system's actual behaviour to meet some specified goals in Cloud environments. The most common AI task scheduling implementations are based on algorithms belonging to the fields of Swarm Intelligence and Evolutionary Algorithms. The social behaviour of biological systems is modelled in swarm-based optimisation algorithms. Social interactions in a randomly initialised swarm play an important role in updating the current position of each individual. Such systems are characterised by the individual's constant search for a better position to adapt to the environment. The foundation of Evolutionary Algorithms is natural evolution. Through generations of a randomly initialised population, Evolutionary Algorithms apply evolutionary operators, including selection, crossover, and mutation, to find an ideal solution. The Particle Swarm Optimisation, Ant Colony, Tabu Search, Harmony Search, and Bee Colony algorithms belong to the group of Swarm Intelligence algorithms. Genetic Programming, Genetic Algorithm, Evolutionary Programming, and Evolution Strategies are well-known algorithms from the group of Evolutionary Algorithms.

Among analysed metaheuristics, the Evolution Strategies algorithm is chosen for implementation on task scheduling in the heterogeneous Cloud computing environment. The Evolution Strategies approach has not been applied to the resource and task scheduling domain problems. It is flexible and adaptable to the requirements of modern large-scale Cloud environments under different loads as an intelligent resource management optimisation strategy. The concepts of evolution and natural selection served as a foundation for the self-adaptable Evolution Strategies algorithm. This algorithm employs random mutation, recombination, and selection depending on the fitness function value. During the iterative process, these processes are applied to a population of individuals with potential solutions to generate better solutions. Chapter 5 explains the fundamental ideas behind the Evolution Strategies algorithm.

3.5   Literature review

This section examines pertinent research on heterogeneous Cloud computing, as well as resource management and scheduling techniques. The thorough analysis of the relevant literature reveals that there is no systematic classification or survey on the issue of task scheduling in heterogeneous Cloud computing.

The survey on Cloud resource management and its components presented related definitions and classification taxonomy in the field [94]. Addressing dynamic heterogeneous environments with data-intensive workflows was the focus of the analysis. Identified are gaps

and future challenges to be addressed by research and development. The gaps are identified in addressing the data-intensive loads, hybrid and multi-Cloud scenarios concerning resource heterogeneity and distribution, rescheduling and performance fluctuations, and reliability.

A job scheduling policy based on the stochastic Tabu Search algorithm was proposed and examined in [99] to solve the resource allocation and task scheduling problem in Grid/Cloud networks. A hybrid job scheduling approach merged Tabu and Harmony Search algorithms [100] and achieved satisfactory results in terms of makespan and cost.

Algorithms that simulate natural processes will have importance as intelligent optimisation and resource management methods, primarily in AI and reinforcement learning.

Task scheduling approaches based on an adapted PSO algorithm were proposed in several studies to better balance local and global search. Modified PSO (M-PSO) [101] used a non-linear function for the inertia weight to facilitate search and reduced energy consumption required to accomplish a given workload by mapping all tasks to the system's computing resources. The authors in [102] proposed an adaptive inertia weight strategy for Cloud-based task scheduling to better balance local and global search. The hybrid task scheduling strategy based on the PSO technique and fuzzy theory was proposed to minimise makespan and maximise resource utilisation [103]. It used a fuzzy system for fitness calculations with input factors and later applied four modified velocity updating methods to explore search space. More existing PSO-based scheduling schemes in a Cloud computing environment were analysed and classified in [104]. The PSO-based task scheduling algorithm was integrated with the Ant Colony algorithm [105]. The experiment carried out in a laboratory environment showed improvements in fitness, cost, and running time when comparing results with PSO and Ant Colony algorithms. The Ant Colony algorithm is a population-based metaheuristic inspired by the foraging behaviour of ants and their pheromone communication to find a path between their colony and source of food. The enhanced Ant Colony task optimisation algorithm was used to balance the VM load and utilisation rate [106]. The function for searching for the optimal task scheduling solution combined three objectives: minimum waiting time, resource load balancing degree, and task completion cost. The load weight coefficient of VMs was introduced in the update process of the local pheromone. Moon et al. proposed the slave ants-based ant colony optimisation - SACO for allocating tasks to VMs and enhancing the performance of the task scheduler in the Cloud [107]. The proposed algorithm was focused on solving the global optimisation problem with slave ants by avoiding long paths whose pheromones are wrongly accumulated by the leading ant. Diversification and reinforcement strategies were applied to slave ants. This algorithm does not consider resource heterogeneity and the type of computing instances.

An in-depth literature review showed that the Genetic Algorithm metaheuristic is a dominant optimisation algorithm and the most common Evolutionary Algorithm in the Cloud scheduling domain. Duan et al. [108] proposed an Adaptive Incremental Genetic Algorithm (AIGA) and modelled task scheduling as an objective optimisation problem. The algorithm

used the different mutation and crossover rates to achieve minimum makespan and progressed in some measured parameters compared with other relevant algorithms. Multi-objective Balancer Genetic Algorithm (BGA) [109] optimisation analysed makespan and average resource utilisation ratio for tasks arriving in batches. It used the balancer to balance the workload among VMs. The fitness function combined the makespan and load balancing. The BGA algorithm is evaluated based on makespan and average resource utilisation ratio metrics. A hybrid Genetic Algorithm, the Modified Genetic Algorithm combined with Greedy Strategy (MGGS) [110], aimed to improve makespan and load balancing considering the expected total time for the VM to execute all assigned tasks as the fitness criteria in a fewer number of iterations. The performance of the MGGS algorithm was compared with several existing algorithms based on the total completion time, average response time, and QoS parameters.

The existing research on software-defined approaches in the Cloud based on heterogeneous resources is analysed. Intelligent adaptive algorithms play a key role in Cloud environments that use software-defined procedures for central, configurable, and dynamic software management and optimisation of the use of remote hardware resources related to network, storage, and CPU for different computer architectures. Concerning the network as a core system resource, SDN is a widely adopted agile approach for dynamic environments that require continuous adaptation to achieve greater efficiency. SDN provisions and manages network traffic and balances load automatically. These SDN functions are significant in Cloud data centre environments. SDN's main feature is the separation of control and forwarding functions from the infrastructure level. Control-level software services facilitate programmable, automated, centralised, and remote network infrastructure management that enables the introduction of new technologies, features, and additional network elements. Energy efficiency, security, virtualisation, and performance (network throughput, latency, availability, and QoS) were highlighted as common challenges for large-scale Cloud data centres in the study on employing a software-defined approach for Cloud computing [111]. According to the given taxonomy, this research aims to improve performance in heterogeneously configured inter-Cloud data centre architecture focusing on joint optimisation of resources for batch processing. An adaptive Genetic Algorithm strategy was applied to SDN-based Cloud data centres [112] for resource allocation and VM placement. Energy, VMs, and intra-data centre communication costs were considered for optimisation. Related taxonomy and challenges in performance optimisation, usability, and viability of metrics, resources, infrastructure, and software were highlighted in a survey on HPC Cloud [58]. It is highlighted that HPC Cloud will become indispensable for supporting research efforts in Big Data and AI. Proper resource allocation algorithms and resource virtualisation technologies in the hybrid Cloud environment can balance resource requirements and improve the expected QoS cost-effective model.

A literature survey has shown that the Evolution Strategies approach has not yet been used in resource and task scheduling, but it has the ability and potential to optimise resource utilisation. Thus, it was selected for application in this research.

It has also been shown that the number of research projects and approaches in centralised Cloud resource management and task scheduling in geographically distributed heterogeneous data centres is quite limited. The simulation was the dominant approach for evaluating the system performances of the models proposed in the surveyed papers. The number of tasks in the workload in existing research was also considered. A review of relevant literature has shown that prior tests and studies were carried out on workloads of several hundred tasks. In this research, the intention was to increase the number of tasks significantly. Thus, a workload of several tens of thousands of tasks was created. A limited number of reviewed approaches considered load balancing, which is significant for VM-task allocation when dealing with the heterogeneity of tasks and resources. Most existing resource management and task scheduling models are evaluated using a smaller number of metrics to prove the scalability of these approaches, which could be extended by introducing additional metrics to assess scalability. To increase scalability in Cloud systems, further improvements in resource management could still be made.

Several studies and projects are tackling the challenges of HEP data processing in the Cloud.

CERNBox [113] is the Cloud storage system dedicated to CERN users and is based on the EOS, a highly distributed disk system. CERNBox servers are placed in the CERN Data Centre. CERNBox enables storing, accessing, and sharing data and files stored in the CERN EOS infrastructure from any web browser.

The ATLAS experiment has significantly used Cloud computing services during Run 2 and continues improving them. The Big Production and Data Analysis (BigPanDA) project [114] extends the previously used data-intensive workload management system to exa-scalable Cloud computing platforms. The BigPanDA introduces new types of computing resources into ATLAS computing infrastructure for processing during periods of high system load. In addition to expanding the system beyond the Grid computing, the goal is to achieve the system application outside HEP.

Helix Nebula's strategic initiative [115] for European scientific Cloud computing set the priorities for the widespread adoption and availability of Cloud services across different scientific domains. The initiative has documented an Information as a Service business model through which fast implementation, adoption, scaling, and competition are applied. These steps tend to provide a roadmap for scientific data lifecycle management. The scientific Cloud called Helix Nebula Science Cloud (HNSciCloud) is a hybrid platform that supports the deployment of Cloud services, HPC, and Big Data capabilities. Jobs from the ATLAS experiment were used as a use case to test the applicability of Cloud computing to LHC computing [116]. Tests have shown that the Cloud is an appropriate infrastructure for simulation workloads executed on the WLCG.

In [117], four CMS workflows of different complexity were analysed on the dynamic resources of Fermilab HEPCloud. The results showed the successful use of dynamically

projected resources for highly scalable CMS workflows. In addition, they showed that the cost of commercial Clouds for the experiment scale is higher but comparable to the cost of on-site resources.

The HPC resources were integrated with the High Throughput Computing (HTC) CERN data centre infrastructure [118]. Tools used for CERN's internal Cloud were managing the HPC cluster. Experiences in applying Cloud computing technologies in the HPC environment are described, along with job scheduling and resource management challenges.

Furthermore, an HPC cluster was integrated with one centre of the ATLAS experiment at the Tier 2/Tier 3 level using the virtualisation technique [119]. The aim was to achieve an efficient and scalable environment. Different configurations and performance tests have been conducted to evaluate results.

Researchers from Karlsruhe Institute of Technology have developed Transparent Adaptive Resource Dynamic Integration System (TARDIS) [120], a resource manager for dynamic integration and allocation of diverse resources (Grid sites, HPC centres, Cloud providers) by focusing on the requirements of individual HEP workflows. As a result, a specialised software environment can be provided on all the opportunistic resources integrated into one overlay batch system using the DRONE concept for transparently and dynamically providing resources. Allocation, utilisation, demand, and supply are the metrics used in the decision process of providing the resources to fit the demand. However, network limitations and CPU inefficiency are present in the case of I/O intensive jobs since the jobs run on integrated resources and read the files from remote Grid sites due to the lack of permanent HEP storage.

4 MODEL FOR PROCESSING IN THE CLOUD

This research aims to improve performance within a heterogeneously configured Cloud architecture of geographically distributed data centres by optimising resource utilisation for batch processing. An adjustable and scalable simulation environment is required to assess the resource management and task allocation approaches within a Cloud while avoiding challenges and costs incurred in a physical Cloud environment.

4.1 ALICE data preparation from processing on Tier 2

Applications from the science and business domains benefit from the scalability and performance of Cloud services. Workloads from these domains consist of multiple computational tasks that have to be scheduled and evaluated on parallel systems in the required time. Workloads are based on independent tasks (Bag of Tasks) or tasks with dependencies (Directed Acyclic Graphs). Tasks in a workload can be processed as transactions or batches and can have stringent requirements on CPU, storage, or memory. The significant challenges for efficient resource management in Cloud for scientific research containing heterogeneous resources are good resource utilisation and task scheduling. These challenges could be addressed through the task execution on resources appropriate to the task requirements but with constant resource usage monitoring.

During the research, the existing workloads from resource-intensive scientific applications were analysed. Significant emphasis was on workloads that simulate real-world activities in HEP. The LHC experiments at CERN create large collections of data measurable in petabytes (PB). The amount of HEP data is being created and growing at high speed. Multiple copies of data are distributed and stored in WLCG's numerous data centres. Raw data processing, simulation, reconstruction, and analysis processes are executed on WLCG resources.

The workload has to be open-source and should allow modelling on a selected CloudSim simulator for reproducible evaluation of the heterogeneous multi-site Cloud model and the applied method detailed in Chapter 5. Workload serves as input data to simulate intra-Cloud behaviour focusing on task allocation and joint optimisation of resources for batch processing between geographically distributed data centres. CloudSim supports workloads in Standard Workload Format (SWF) [121]. SWF format is easy to parse and uses integer data type in standard units. Independent jobs in the ASCII log are represented as a single line. Jobs are

described by the 18 characteristics shown in Figure 4.1.

The Data Fields in the SWF file	
Job Number	Requested Memory
Submit Time	Status
Wait Time	User ID
Run Time	Group ID
Number of Allocated Processors	Executable (Application) Number
Average CPU Time Used	Queue Number
Used Memory	Partition Number
Requested Number of Processors	Preceding Job Number
Requested Time	Think Time from Preceding Job

Figure 4.1: Job characteristics in the SWF file. Derived from the [121].

Several real-life workload logs were collected from the supercomputers in production and converted into SWF format. An example is the LHC Computing Grid (LCG) workload log from the parallel workload archive [122]. The log provided by the e-Science Group of HEP at Imperial College London contains jobs of 11 days of activity in 2005 from multiple nodes and has 188041 jobs. The jobs were collected from the LCG testbed. The exponentially distributed jobs required one CPU to process a certain amount of data.

Apart from the capacity and complexity of simulated resources, the number of jobs in the used workload affects the scalability of the system. Most researchers in the field have based their research on a smaller number of tasks and resources of lower capacity than is the objective of this research, especially in the Big Data era. A significant increase in the amount of data and their complexity is expected compared with previous data taking periods. Therefore, a workload to simulate batch processing of ALICE Monte Carlo production jobs on Tier 2 has been created. The workload is in CSV (comma-separated values) format, compliant with job fields defined by SWF requirements. The resource needs of the ALICE experiment before and after the significant modification in the experiment settings were analysed. That served as a motivation to develop a scientific workload with computationally demanding tasks associated with the ALICE Monte Carlo simulation. All Monte Carlo production jobs are submitted to the central ALICE GRID task queue. After the user submits the job called the master job, AliEn separates the master job into the different subtasks specified as sub-jobs.

Sub-jobs are then submitted to Tier 2 site resources. The Monte Carlo jobs are CPU-intensive, with a small set of input data, and without large Tier 2 output data storage requirements.

The synthetic workload is created by utilising the Monte Carlo log data about jobs executed on more than WLCG 60 locations taken in February 2022. It consists of 49026 batch sub-jobs (tasks) of one proton-lead (p-Pb) Monte Carlo production master job. The data have been extracted from the ALICE Grid monitoring system. Certain properties are adapted to SWF format using most of the data from the Monte Carlo data logs. The sub-jobs are submitted dynamically for concurrent processing. In Run 1 and Run 2, ALICE sub-jobs were assigned to a VM configured in advance for processing with 1 CPU core and 2 GB of RAM per core. ALICE processing in Run 3 and Run 4 will be based on timeframes requiring larger memory and multi-core processing. Execution will move from 1-core to 8-core to achieve high CPU efficiency. Hence, the ALICE Monte Carlo sub-jobs in the workload are modelled to require either 1 or 8 cores. In the simulation framework, sub-jobs are represented as tasks (cloudlets). Tasks have varying times of execution and use different amounts of resources. The execution length of the cloudlet varies from 2340 to 60780 (in Million Instructions).

Created workload follows the stepped increase and decrease demand pattern shown in Figure 4.2.
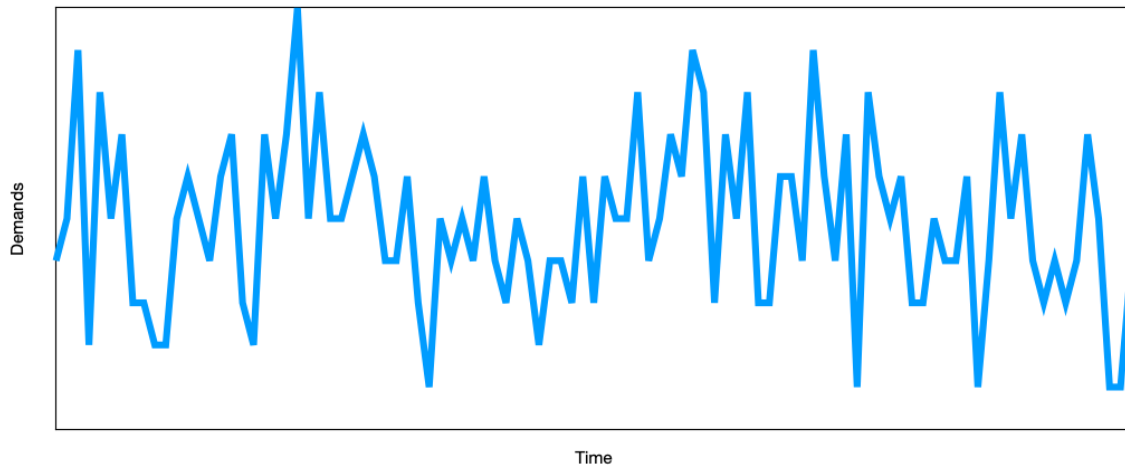
Figure 4.2: Demand pattern of created workload.

VM placement strategy is needed to assure greater accessibility and usability of incorporated HPC and Cloud infrastructure based on multi-core VMs used for testing multi-core task processing.

4.2 Conceptual modelling of heterogeneous multi-site Cloud architecture

This section describes the infrastructure of a heterogeneous Cloud environment, its instances, and its requirements. The distributed heterogeneous Cloud system is modelled to represent its key configuration characteristics. Conceptual modelling is an important task to be carried out before developing computer code in the simulation modelling phase. The real-world system to be modelled needs to be understood to determine the content of the simulation model and objectively model the system's purpose. The foundation of conceptual modelling is the abstraction of the real system based on its description. Simpler conceptual models are preferred over complex ones since they are more adaptable during simulation research. Simple models are easier to validate, more flexible, and faster to develop and execute. They require less data and the obtained results are easier to interpret [123].

The following are examples of Cloud systems of different infrastructure scales and capacities that share similar features and resource management challenges with the system simulated in this research.

INFN Cloud infrastructure [124] supports scientific research based on services built on state-of-the-art, open-source, vendor-neutral architecture for computing and data. The core backbone connects the large data centres and several federated sites of the INFN multi-site Cloud infrastructure. WLCG experiments have access to INFN infrastructure that also serves the research needs in other scientific domains.

OVH [125] is the largest European provider of Cloud services and has 17 data centres in France and 32 data centres in the world. The given structure of OVH will be the backbone of the future Cloud project of the European Union named Gaia-X [126].

Google Cloud [127] is one of the largest providers of Cloud computing services and infrastructure. Google Cloud operates data centres on 5 continents, currently in 34 independent geographic areas, to meet users' latency, availability, and durability requirements. The users can choose from a wide range of products and APIs available from Google computing, storage, analytics, and networking products and services. The concept is developed according to the model and needs of such distributed heterogeneous Cloud systems.

The goal of the HR-ZOO [5] is to build a computing and data Cloud that will serve as a basis for the national research and innovation e-infrastructure. Five interconnected data centres located in four cities in the same area form the HR-ZOO Cloud:

- Zagreb - HR-ZOO ZG1 (DC 1)
        - HR-ZOO ZG2 (DC 2)
- Osijek - HR-ZOO OS (DC 3)
- Rijeka - HR-ZOO RI (DC 4)
- Split - HR-ZOO ST (DC 5).

Solving advanced problems from different research domains is based on parallel processing on high performance clusters that generate and consume increasing amounts of data. HPC or supercomputers are becoming ubiquitous and will have increasing applications in many fields. They can process a large amount of data in an extremely short time and are necessary for contributing to scientific and social challenges. There are numerous initiatives and plans for investments in the development of supercomputer infrastructure, which is crucial for competitiveness and independence in the data economy. In the last updated list from June 2022 of the TOP500 [128] most powerful world supercomputers, 118 supercomputers are located in Europe, with 2 of them in the top 10 of the list. HPC applications focus on large computational loads, while Big Data Analytics (BDA) applications are demanding in terms of storage. These paradigms result in very different sets of key requirements with constraints on timing and precision, availability, software, service-level performance, memory, storage, communication, and scheduling. Cloud facilitates the dynamical access to large-scale resources (storage, compute, network) requested by these applications. Convergence of HPC, BDA, and Cloud results in new data processing paradigms and leads to challenges related to the extreme scale of data management.

Heterogeneity is a factor that can affect resource management at the architecture, configuration, or load level. HR-ZOO provides resources for High Performance Computing (HPC), High Throughput Computing (HTC), High Scalability Computing (HSC), and resources for storing large data collections provided by the Cloud, as shown in Figure 4.3. The HR-ZOO HPC system will have a performance of around 1 PFLOPS. The supercomputing infrastructure is going to be used for the specific needs of science and higher education, industry, and the public sector. The University Computing Centre of the University of Zagreb (Srce) will manage the national centre. The HR-ZOO multi-site Cloud infrastructure of data centres could contribute to data processing of LHC experiments' productions at WLCG Tier 2. The data centres are interconnected with high bandwidth capacity links of 100 Gbit/s. HR-ZOO infrastructure will be linked to the European research and educational community via the GEANT network.

According to the HR-ZOO documentation, data centres have general purposes x86 architecture and Linux OS. Heterogeneous data centres consist of hosts with differently configured instances of VMs. The system has three groups of hosts. These are hosts with processor resources, hosts with high memory capacity, and hosts with HPC processors. The world's highest-performing server CPU for general-purpose computing and providing great performance in solving scientific and compute-intensive problems [129] is used to simulate HPC hosts with performance expressed in TFLOPS. These HPC hosts are designed with an adjusted number of 64-core processors with a processor base clock of 2.45 GHz.

Additional heterogeneity is introduced at the level of VMs. The system is configured with a total of 2100 VMs distributed to data centres of different resource capacities.

Resources: HSC, HTC, HPC

HSC
15 hosts
    2x32 core CPU@2.9 GHz
    0.7 TB DDR, 20 TB all-flash NVMe SSD
HPC
1 host
    2 GB RAM/CPU core, 400 TFLOPS, 3.8 TB NVMe SSD
2 hosts
    112 CPU cores@2.45 GHz, 4 TB RAM, 3.8 TB NVMe SSD
1 host
    16 CPU cores/GPU, 96 GB RAM/GPU, 40 GB RAM/GPU, 1000 TFLOPS, 3.8 TB NVMe SSD
1 host
    2 GB RAM/CPU core, 400 TFLOPS, 2x7.6 TB NVMe SSD
1 host
    16 CPU cores/GPU, 96 GB RAM/GPU, 40 GB RAM/GPU,1000 TFLOPS, 2x7.6 TB NVMe SSD

HTC
7 hosts
    1500 CPU cores@2.2 GHz, 4 GB RAM/CPU core
    3.8 TB NVMe SSD
2 hosts
    112 CPU cores@2.2 GHz
    2 TB RAM, 3.8 TB NVMe SSD
1 host
    16 GPU, 16 CPU cores/GPU, 96 GB RAM/GPU, 40 GB RAM/GPU
    3.8 TB NVMe SSD
3 host
    4 CPU cores@2.2 GHz, 16 GB RAM, 133 TB NVMe SSD, (10 CPU cores, 10 GB RAM)/NVMe
    SSD
3 host
    4 CPU cores@2.2 GHz, 16 GB RAM, 2 PB HDD, 80 TB NVMe SSD, (1 CPU cores, 5GB RAM)/
    HDD
3 host
    64 CPU cores@2.2 GHz, 4 GB RAM/CPU core, 2x7.6 TB NVMe SSD

Resources: HSC, HTC

HSC
4 hosts
    32 core CPU@2.9 GHz
    0.5 TB DDR4, 18.75 TB all-flash NVMe SSD
HTC
1 host
    1500 CPU cores@2.2 GHz
    4 GB RAM/CPU core
    3.8 TB NVMe SSD

Resources: HSC

HSC
15 hosts
    2x32 core CPU@2.9 GHz
    0.7 TB DDR4, 20 TB all-flash NVMe SSD

Resources: HSC, HTC

HSC
4 hosts
    32 core CPU@2.9 GHz
    0.5 TB DDR4, 18.75 TB all-flash NVMe SSD
HTC
1 host
    1500 CPU cores@2.2 GHz
    4 GB RAM/CPU core
    3.8 TB NVMe SSD

Resources: HSC, HTC

HSC
4 hosts
    32 core CPU@2.9 GHz
    0.5 TB DDR4, 18.75 TB all-flash NVMe SSD
HTC
1 host
    1500 CPU cores@2.2 GHz
    4 GB RAM/CPU core
    3.8 TB NVMe SSD

DC 2

HR-ZOO

DC 3   DC 1

DC 4   DC 5

Figure 4.3: The HR-ZOO global system description. Derived from the HR-ZOO data.

VMs are comprised of varied CPU, memory, storage, and network capacities and are used with a space-shared policy. There are three main different-sized VM instance types. The third type of VMs is modelled after the features of the latest generation of general-purpose Amazon EC2 m5a.2xlarge instances [130]. Resource utilisation can be improved by scaling the number of available VMs. Time-shared provisioning policy is used for allocating tasks to VMs' cores, enabling dynamic context switching. Figure 4.4 illustrates used provisioning policies on two CPU cores hosting two VMs and each VM hosting four tasks.
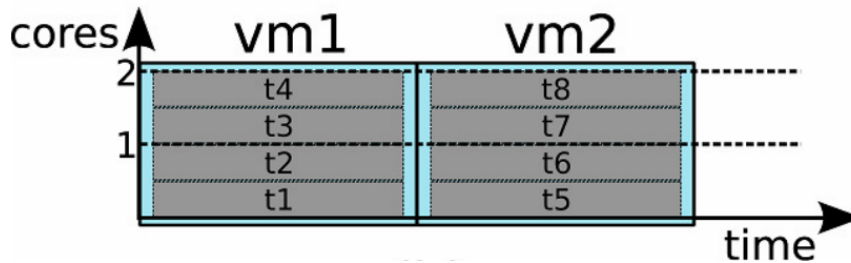
Figure 4.4: Used provisioning policies - space-shared for VMs and time-shared for tasks [85].

Created workload serves as an input to the model. The one data centre (DC 2) governs the distributed heterogeneous resources through the central broker. Based on task specifications, the central broker manages the distribution of tasks on the VMs and enforces a scheduling policy to select a data centre to run the submitted tasks. The central broker acts as a task scheduler and manager that determines the host, performs the selection, schedules, and implements computations during the simulation on the appropriate resources.

4.3 Simulation modelling of heterogeneous multi-site Cloud architecture

A simulation study is an analysis and design technique used in research and management to evaluate and estimate the characteristics of the models that represent diverse and complex real-world systems. Analysing simulated system models, it can be concluded that the most simulated models are stochastic, dynamic, and discrete. Experimenting with the actual system often requires vast computing resources. Simulation is done before implementation to gain knowledge of input variables or modifications in the system environment and their effect on a system's performance improvement. This section describes the simulation model of the distributed heterogeneous Cloud system based on the conceptual model in more detail. The Cloud infrastructure model based on the settings of the HR-ZOO is simulated in the CloudSim software framework version 4.0 integrated with the Eclipse environment. CloudSim provides many features needed to simulate a Cloud system. It has a layered architecture for implementing allocation and scheduling algorithms. CloudSim has fundamental entity classes for modelling data centres, hosts, VMs, brokers, and application services (cloudlets) [85]. Different provisioning policies for allocating cloudlets to VM and VMs to hosts can be applied in a large-scale simulation environment to meet task cloudlet requirements or VM deployment requirements. The characteristics of different VM instance types are shown in Table 4.1. The simulation environment consists of data centres modelled as in Table 4.2. Millions of Instructions Per Second (MIPS) characterise the processing power, while the total size capacity for storage and memory is specified in MB. Cloud entities can communicate through a message passing mechanism. Simulation implies the process of creating and maintaining data centre characteristics, determining the central broker, creating and setting the configuration of VMs, and the process of creating a workload with tasks that are sent to the central broker and executed on VMs.

The Cloudlet and DatacenterBroker classes are noteworthy. The Cloudlet class contains attributes that describe the execution of tasks. These properties are also important for execution monitoring. Cloudlets are mapped for execution on a specific VM according to the proposed algorithm. The DatacenterBroker class receives a list of cloudlets for its execution on defined VMs. Tasks are read from the created workload in CSV format using a specialised function.

Output performance measures are observed and gathered during and after the simulation run. The number of cloudlets in the experiments varied from 1000 to 20000. Each experiment was repeated 10 times. After implementing the metaheuristic algorithm for task scheduling in a simulated heterogeneous Cloud computing environment, the mean values of the obtained results have been analysed and will be presented in Chapter 5.

Table 4.1: Capacity of heterogeneous VM instance types.

VM INSTANCE TYPES	1	2	3
CPU CORES	2	4	8
CPU [MIPS]	1000-1500	1000-1500	2000-2350
RAM [GB]	8	8	35
STORAGE [GB]	10	10	10
BANDWIDTH [Gbps]	1	1	1
DATA CENTRE	DC 1, DC 2, DC 3/4/5	DC 1, DC 2	DC 2

Table 4.2: Hardware capacity of data centres.

Data centre	DC 1	DC 2	DC 3/4/5
Paradigms	HSC	HPC, HSC, HTC	HSC, HTC
Number of hosts	15	40	5
Number of CPU cores	960	62333	1628
CPU [MIPS]	2900	2200/2450/2900	2200/2900
Total RAM [GB]	10000	171389	8000
Total storage	300000	7068200	78800

4.4   Metrics for evaluating the processing model in the Cloud

Evaluation of task scheduling and resource allocation in large-scale systems is challenging for data centres' administrators and users since scale and cost are unpredictable. Cloud system modelling and simulation require consistent metrics with a high level of accuracy to assess performance. Scalability is affected by the following metrics: makespan, throughput, average resource utilisation, average execution time, degree of imbalance, and the number of VM instances over time. These metrics are used in this research to demonstrate the effectiveness of the proposed metaheuristic and the scalability of the simulation model.

- **Makespan** is the commonly used performance metric [131] for efficient scheduling concerning the time required for the concurrent execution of submitted tasks on assigned VMs. The optimisation goal is to minimise the makespan. Makespan is a metric that indicates the maximum time required to execute assigned tasks on each VM:

$$ET_{VM_i} = \sum_{j=1}^{m} ET_i(task_j), \text{ where i} \in \text{VMs} \tag{4.1}$$

$$MS = Max(ET_{VM_1}, \dots, ET_{VM_n}) \tag{4.2}$$

MS – makespan
ET – maximum execution time
j – task
i – VM.

- **Throughput** is an important metric to consider when analysing scalability and CPU efficiency. Throughput describes the operating rate as the number of tasks processed per second [132]. The optimisation goal is to achieve the highest possible throughput.

$$T = \frac{N_{tasks}}{MS} \tag{4.3}$$

T – throughput
N_{tasks} – number of tasks
MS – makespan.

- **Average Resource Utilisation** is an important quantitative metric that depicts the effectiveness of resource usage in the Cloud system. Optimising the average utilisation of resources results in load balancing in Cloud data centres. Average Resource Utilisation is calculated using the following equation [131], [133]:

$$RU_{avg} = \frac{\sum_{i=1}^{n} ET_{VM_i}}{MS \times n} \tag{4.4}$$

RU_{avg} – average resource utilisation

MS – total makespan

ET – total execution time of VM resources that process tasks

n – number of VM resources.

- **Average Execution Time** is the time elapsed in the execution of the task on the VM as a Cloud resource. The goal is to minimise this performance metric that is calculated by the following equation [133]:

$$ET_{avg} = \frac{\sum_{j=1}^{m} ET(task_j)}{m} \tag{4.5}$$

ET_{avg} – average execution time

ET – execution time of task

j – task

m – the number of tasks.

- **Degree of Imbalance** indicates the load balancing performance between virtual resources in different data centres. The degree of imbalance is based on execution times on all VMs and is calculated using the equation below [134]:

$$DI = \frac{T_{max} - T_{min}}{T_{avg}} \tag{4.6}$$

DI – degree of imbalance

T_{max} – maximum execution time

T_{min} – minimum execution time

T_{avg} – average execution time.

- **Number of VM instances over time** is employed for observing proportional growth and reduction of capacity under system load changes to match the costs of operating the system with traffic. The number of active VM instances per individual data centre (scale-in) and across data centres (scale-out) is observed in intervals.

The primary goals of the task scheduling strategy in the considered Cloud environment are to achieve a reduced makespan, better throughput, increased average resource utilisation, shortened average execution time, a smaller degree of imbalance, and scalability of allocated VMs. The proposed metaheuristic strategy for task scheduling and load balancing will be evaluated in the context of these metrics under different loads and described in Chapter 5.

5   EVOLUTION STRATEGIES-BASED MODEL OPTIMISATION

The migration of resource-intensive applications to the Cloud requires new approaches to solve existing limitations. These approaches have to be dynamic, adaptive, and often automated. One of the solutions to address mentioned challenges and objectives is to use software-based concepts when designing and using Cloud infrastructure.

This chapter presents an optimisation strategy based on the Evolution Strategies metaheuristic. Applied task scheduling algorithms address both the allocation and scheduling of tasks across heterogeneous VMs hosted on Cloud data centres for scientific research. This chapter is derived from a paper entitled "Scalable Management of Heterogeneous Cloud Resources Based on Evolution Strategies Algorithm," published by the gold open access journal IEEE Access in June 2022 [135].

5.1   Software management concept

Cloud relies on virtualised resources and its key advantages are dynamic scalability and high availability of services. Adding or removing capacity is easy as demands evolve and grow over time. Efficient resource management must be employed in Cloud data centres to fulfil QoS requirements related to resources shared by various tenants and applications. Cloud deployment is based on software and a good network connection.

Software-based system management has to improve the usage and adjust the capacity and performance of distributed storage, computing, and networking platforms. An essential feature of the software-defined architecture is separating the core control layer from the infrastructure layer. The control layer provides software services for a dynamic, monitored, and program-driven management of the underlying heterogeneous computing, networking, and storage components. There is a trend to base physical infrastructure on industry-standard x86 architecture and open-source hardware. Every data centre component can be centrally managed, and functions can be automated through intelligent software. With the software-defined approach, the time to provision new resources can be reduced, infrastructure performance can be improved, managing virtualised resources can be flexible, and the system can be easily adapted to dynamically changing workloads.

The dynamically adaptive intelligent algorithms applied via the software-based controller

have the potential to improve the efficiency of Cloud task scheduling and system's scalability. These intelligent algorithms belong to branches of evolving Computational Intelligence computing paradigm, a subfield of AI. Neural Networks, Fuzzy Systems, Evolutionary Computation, Swarm Intelligence, and Artificial Immune Systems are branches of the Computational Intelligence paradigm [136]. These models follow the intelligence and behaviour of natural systems, from biological and evolutional to social optimisation processes, and are adaptable to new conditions.
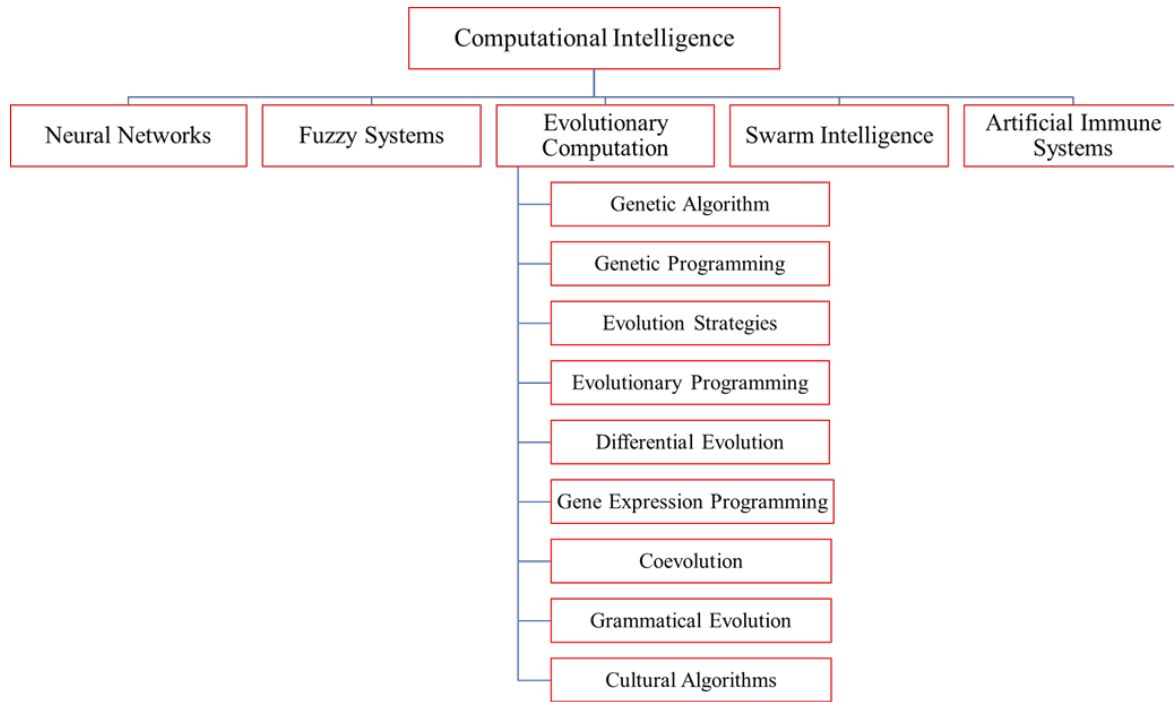
Figure 5.1: Classification of intelligence algorithms.

The group of Evolutionary Computation algorithms has a significant application in this research domain. Genetic Algorithms, Genetic Programming, Evolution Strategies, and Evolutionary Programming are the most known Evolutionary Computation algorithms. Evolutionary processes, such as natural selection, reproduction, recombination, mutation, and survival of the fittest, are the elements used to improve the ability of an individual to survive in dynamically changing and competitive environments. Solving computer-based problems using Evolutionary Computation paradigms implies applying these elements.

The Evolutionary Strategies metaheuristic has often been applied to solve real problems in all fields of activity in today's world. However, an extensive review showed that the (μ, λ)-Evolution Strategies algorithm did not have its application in managing resources in large-scale Cloud computing. Its properties and potential for adaptation to the needs of modern systems and various loads were key for selecting the Evolution Strategies algorithm in this research.

5.2 Evolution Strategies algorithm

The Evolution Strategies (ES) algorithm is a global optimisation search algorithm developed by Rechenberg and Schwefel [137]. This metaheuristic is a population-based technique that mimics the biological evolution optimisation process and adaptive behaviour of living organisms. It is based on the concepts of natural evolution, which through mutation, natural selection, and reproduction, generates the individuals that survive the evolution process. In other words, suitable candidates concerning the objective function of the chosen domain are selected. The Evolution Strategies algorithm applies the self-adaptation of mutation parameters, which is not a principle used by other evolutionary algorithms. Evolution Strategies algorithms employ the self-adaptation principle provisioned through a software approach. Self-adaptive systems apply operators and conditions to modify the system behaviour during runtime. These models seize the knowledge about the system that is necessary to perform adaptation actions in meeting set goals.

The basic form of the Evolution Strategies algorithm is a two-membered (1+1)-Evolution Strategy with one parent and one descendant per generation. Multi-membered (μ, λ)-Evolution Strategy (comma strategy) and $(\mu+\lambda)$-Evolution Strategy (plus strategy) schemes were introduced for black-box optimisation in more complex settings. The population in multi-membered strategies consists of μ parents and λ offspring, where $\lambda \geq \mu$. What distinguishes the two multi-membered schemes employing stochastic and deterministic operators is the selection process determining the next generation. In the case of a (μ, λ)-Evolution Strategies, the next generation is formed of the μ best individuals selected from λ offspring, while in the case of a $(\mu+\lambda)$-Evolution Strategies, the next generation is formed of the μ best individuals selected from parents and offspring $(\mu+\lambda)$ individuals. Application of the (μ, λ) scheme of the Evolution Strategies algorithm in complex, dynamic problem spaces prevails over $(\mu+\lambda)$ as it removes outdated solutions from the next generation, leaves local optima, and tracks moving optima in a changing environment. The implementation steps of the (μ, λ)-Evolution Strategies task scheduling model are described in the following text.

5.2.1 (μ, λ)-Evolution Strategies algorithm

The methodology implemented in CloudSim uses (μ, λ)-Evolution Strategies metaheuristic to dynamically allocate tasks to resources needed to process them. The Evolution Strategies algorithm is applied as a non-preemptive scheduling algorithm. The optimal solution is pursued through an iterative process, where each generation represents an iteration step.

The selection of suitable VMs from a given set of all created machines for each task and execution order on the assigned VMs are based on the task properties, VM characteristics, and the capacities of the physical infrastructure.

In Evolution Strategies, each element from the population is a potential solution to

the problem being solved. The parent population of μ individuals evolves to λ offspring individuals, where the condition for the (μ, λ) strategy is $\lambda > \mu$. The next generation of potential solutions is generated using the mutation operator that modifies the randomly selected parent from the previous generation. A mutation operator is applied to explore new possible solutions and introduce a variation to the population. From the observed λ offspring candidates, the μ of the individuals are selected for the next generation. Fitness values guide the parallel search of the large search space in a reasonable computation time and are used to evaluate the individuals as potential solutions. Offspring individuals are compared by fitness value and survive only one generation. Additionally, the Hall of Fame strategy is added to hold the optimal solution in the pool of the best individuals from each generation. The stopping criterion of the iterative process is predefined, and that criterion is a fixed number of generations.

Implemented are the following steps of the (μ, λ)-Evolution Strategies–based approach:

1. Initialisation of the population

   The initial population of individuals is randomly generated. An individual is of the size of the workload. The individual is represented by pairs of tasks, called cloudlets, and assigned VM. Population individual is randomly initialised vector containing n parameters X = $(x_1, x_2, x_3, \ldots, x_n)$, where x_i= (cloudlet, VM), $x_i \in X$. Figure 5.2 shows an example of the individual in the initial population.

$cloudlet_1$	$cloudlet_2$	$cloudlet_3$	$cloudlet_4$	$cloudlet_5$	...	$cloudlet_n$
VM_{100}	VM_{21}	VM_{12}	VM_{17}	VM_9	...	VM_1

Figure 5.2: Example of the individual in the initial population.

2. Evaluation of population

   The fitness function evaluates individuals in the population. The fitness function determines the real-valued candidate solution, the fittest in the struggle for a limited amount of existing resources. It helps in metrics optimisation. As the task scheduling on heterogeneous resources is a multi-objective optimisation problem, the applied fitness function considers the processing speed and RAM capacity of VMs, and the length of cloudlets and RAM used by cloudlets. It is defined as:

$$FitnessFunction = \frac{CloudletLength}{VMProcessingSpeed} + \frac{CloudletRAM}{VMCapacityRAM} \qquad (5.1)$$

The individual with a lower fitness value is given more preference and is considered fitter.

3. Selection

The selection operator simulates the principle of natural selection that allows some offspring individuals to survive and others not. In (μ, λ)-Evolution Strategies–based approach, μ individuals at the end of each generation are selected for creating λ offspring individuals using mutation operator. Previous studies [138], [139], preliminary studies, and conducted tests for this research with different combinations of parameters in the search for the Evolution Strategies algorithm parameters confirmed that the parent-offspring ratio of $\frac{1}{7}$ provides the best selective pressure in the systematic process of searching for optimal solutions.

4. Mutation

A mutation is an essential and fundamental part of the evolution process. In nature, mutations happen randomly. Here, the random resetting mutation operator has been used to create λ offspring as new candidate scheduling solutions. VMs from the set of machines present at the parent are assigned randomly to cloudlets. The mutation operator randomly chooses 10% of the individual population unit and changes the previously assigned VMs with the ones present in the parent. Among that, the VM at the selected position is replaced with a randomly selected VM from the list of all created VMs to maintain the diversity in the population, as shown in Figure 5.3.

Self-adaptation is employed to control and adapt the mutation distribution algorithmically. The mutation strategy also applies the Gaussian distribution function. Gaussian distribution evaluates the values with mean value and standard deviation. The Gaussian distribution function is calculated as $f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(x-\mu)^2}{2\sigma^2}}$, where μ is the mean calculated as $\mu = \frac{\Sigma x}{N}$, $\sigma = \sqrt{\frac{\Sigma(x-\mu)^2}{N}}$ is the standard deviation of the Gaussian distribution, N is the number of observations, and x is the value in the distribution.

VM change is conditioned with the random value from obtained Gaussian distribution based on MIPS values present in the parent VMs. If the set condition is met, the randomly assigned VM remains in the cloudlet-VM pair. Otherwise, a new VM meeting the condition is assigned to the cloudlet.
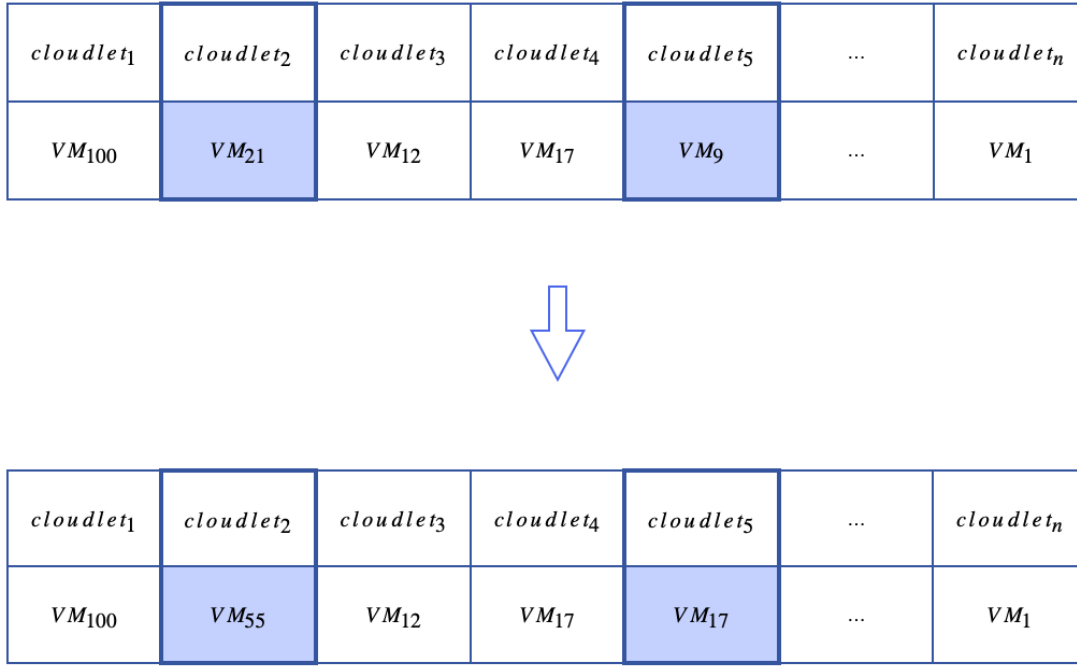
$cloudlet_1$	$cloudlet_2$	$cloudlet_3$	$cloudlet_4$	$cloudlet_5$	...	$cloudlet_n$
VM_{100}	VM_{21}	VM_{12}	VM_{17}	VM_9	...	VM_1

$cloudlet_1$	$cloudlet_2$	$cloudlet_3$	$cloudlet_4$	$cloudlet_5$	...	$cloudlet_n$
VM_{100}	VM_{55}	VM_{12}	VM_{17}	VM_{17}	...	VM_1

Figure 5.3: Performing mutation on offspring individuals.

The resulting offspring generated through mutation represents a new task allocation solution. The mutation is being applied across generations iteratively.

5. Hall of Fame

The Hall of Fame feature is applied to the Evolution Strategies algorithm to preserve the optimal candidate solution from each generation. The Hall of Fame principle enables saving the fittest individual from each population in the pool. After preserving the fittest in the generation, the implemented Evolution Strategies algorithm selects next-generation parent with uniform probability from child individuals. At the end of the process of finding the best solution, the fittest solution is selected among the best individuals from the Hall of Fame. Every cloudlet is assigned to the chosen VM.

The list of cloudlet-VM pairs is submitted to a central broker for managing cloudlet execution dynamically based on cloudlet submission time.

Pseudocode of implemented (μ, λ)-Evolution Strategies algorithm is shown in Figure 5.4.

Algorithm Pseudocode of Evolution Strategies algorithm

Input: $(\mu, \lambda, \text{GenerationSize}, \text{CloudletList}, \text{VirtualMachineList})$

Output: S_{best}(Mapping of cloudlets to appropriate VMs)

```
 1: begin
 2: Population = InitializePopulation(Random, pairs < Cloudlet, VM >)
 3: EvaluatePopulation(Population)
 4: S_best = FindBest(Fitness)
 5: add S_best to BestSolution
 6: while ¬StopCondition(GenerationSize) do
 7:     for i = 0 to λ do
 8:         Parent = SelectParent(Population, μ)
 9:         Child_i = Mutate
10:         add Child_i to Children
11:     end
12:     EvaluatePopulation(Children)
13:     S_best = FindBest(Fitness)
14:     add S_best to BestSolution
15:     Population = Children
16: end
17: return best S_best from BestSolution
```

Figure 5.4: Pseudocode of the proposed (μ, λ)-Evolution Strategies algorithm [135].

5.2.2   Longest Job First data centre broker policy

The execution length of tasks is known in advance from their description. After the Evolution Strategies task scheduling procedure is executed, the list of tasks (cloudlets) with assigned VMs is submitted to the central broker. Tasks are distributed to assigned VMs bound to different data centres. For the optimisation process of load balancing and management of the resource utilisation for the processing of compute-intensive tasks, the Longest Job First broker policy is added to the Evolution Strategies algorithm. The goal is to choose and process longer and more complex tasks first. The Longest Job First broker policy allowed sorting tasks in descending order of their pre-assigned instruction length and enabled their submission to VMs specified by the Evolution Strategies procedure. Memory and CPU resources are primarily assigned to tasks having a greater length. Evolution Strategies scheduling algorithm with Longest Job First broker policy is used in non-preemptive mode.

The steps of the algorithm for achieving better system utilisation for increased system loads based on the Evolution Strategies algorithm for task allocation with the implemented Longest Job First broker policy are shown as a flowchart in Figure 5.5.
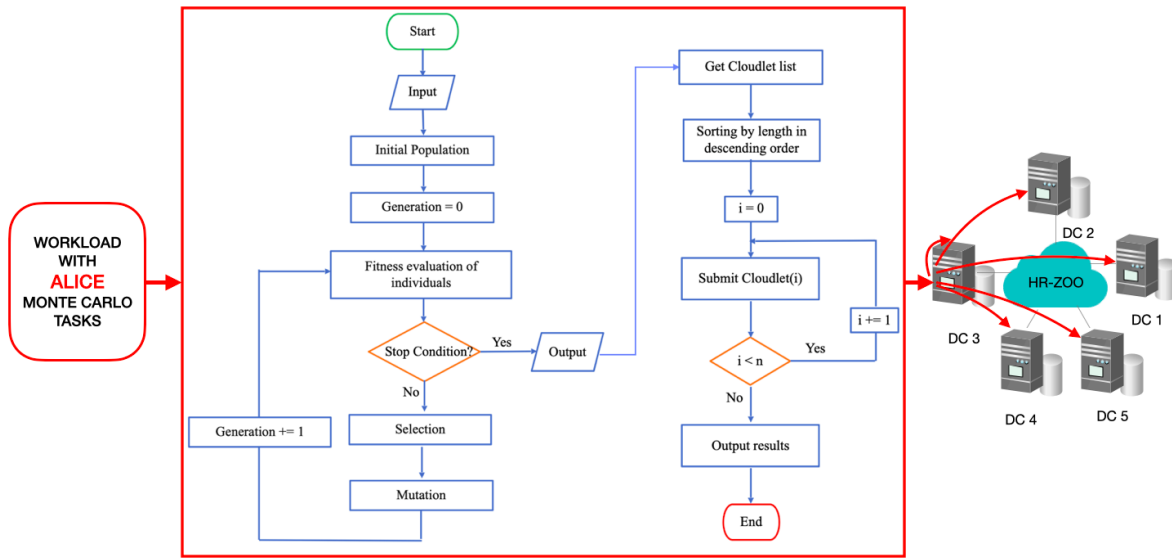
Figure 5.5: Applied optimisation approach based on (µ, λ)-Evolution Strategies algorithm with Longest Job First broker policy.

5.2.3 Shortest Job First data centre broker policy

The Evolution Strategies with Shortest Job First scheduling algorithm is applied to batch-type processing of computationally intensive tasks. The Evolution Strategies algorithm puts tasks into a queue according to their arrival time. With the addition of the fast-scheduling Shortest Job First algorithm, the tasks with the shortest execution time are given priority to be executed first.

The Evolution Strategies approach with Shortest Job First can minimise the waiting time for the execution. With the Evolution Strategies with Shortest Job First, the waiting time for the longer tasks is shorter than the waiting time for shorter tasks in the case of Evolution Strategies with Longest Job First.

The flowchart of the Evolution Strategies with Shortest Job First algorithm is given in Figure 5.6.
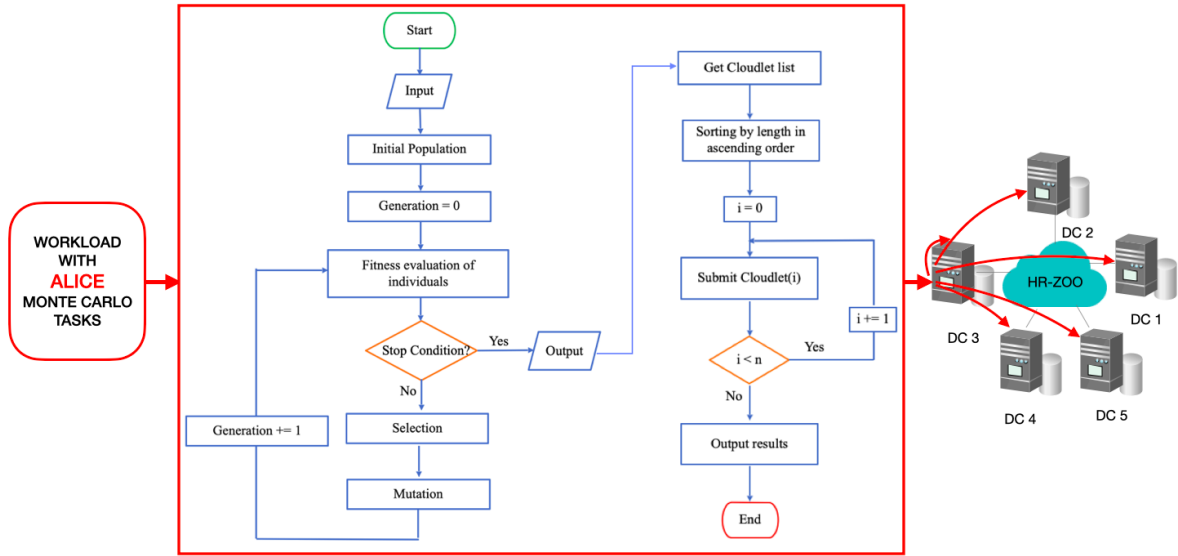
Figure 5.6: Applied optimisation approach based on (μ, λ)-Evolution Strategies algorithm with Shortest Job First broker policy.

5.3 Experimental evaluation

This section presents the results of the experimental implementation of algorithms based on the principles of (μ, λ)-Evolution Strategies metaheuristic. Three principles are used and discussed for task scheduling and balancing in a heterogeneous Cloud environment. These are the Evolution Strategies (ES), Evolution Strategies with Longest Job First (ES - LJF), and Evolution Strategies with Shortest Job First (ES - SJF) principles. Experimental results of selected metrics obtained by Evolution Strategies, Evolution Strategies with Longest Job First, and Evolution Strategies with Shortest Job First simulation are compared with results of the same metrics obtained by Genetic Algorithm simulation.

Observed performance evaluation metrics in simulation studies that have been considered for improvement are:

- Makespan

- Average Resource Utilisation

- Throughput

- Average Execution Time

- Degree of Imbalance

- Load distribution analysis

- Scalability analysis.

The optimisation goals of adaptive Evolution Strategies-based algorithms are to minimise the makespan, maximise resource utilisation, increase throughput, reduce average execution time and imbalance, and execute all tasks assigned to a heterogeneous Cloud environment with respect to VMs. An important goal is to achieve resource scalability. An increase in complexity is achieved by testing the performance for different loads. Simulation tests employ different numbers of tasks (1000, 2000, 5000, 10000, 15000, and 20000 tasks) from the dataset described in Chapter 4. Each algorithm is evaluated 10 times. Average results are analysed. The Evolution Strategies algorithm considers the task characteristics and matches task characteristics to the most appropriate VM instance. The validation is performed for each algorithm and the resulting plots are shown in the following figures.

The graph in Figure 5.7 shows the makespan values of the algorithms. The goal is to reduce makespan as that suggests better performance. The graph reveals a significant reduction of makespan, a time difference between the start and finish times of the task list, when applying Evolutionary Strategies-based algorithms compared with the Genetic Algorithm. The Evolution Strategies with Shortest Job First approach outperforms other policies and shows the lowest makespan value. The values show the anticipated increase in the overall makespan as the workload ratio increases.
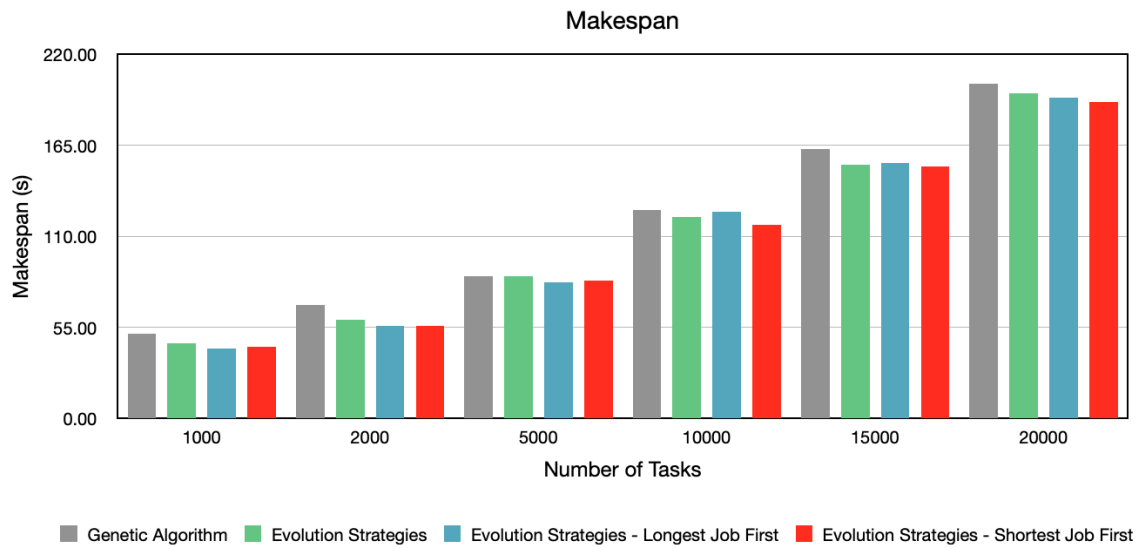

Figure 5.7: Makespan.

Resource utilisation is a key performance indicator of optimal management across complex Cloud infrastructure intended for supporting scientific research. Figure 5.8 reveals that the value of resource utilisation increases with the increase in the number of tasks. Evolution Strategies-based scheduling algorithms have a higher resource utilisation rate. Evolution

Strategies with Shortest Job First approach tends to outperform other approaches in utilising available resources. It is assumed that the processed tasks would be just a part of the planning workload running in the data centres. Thus, the resource utilisation rate shown in the graph corresponds to the application resource usage.
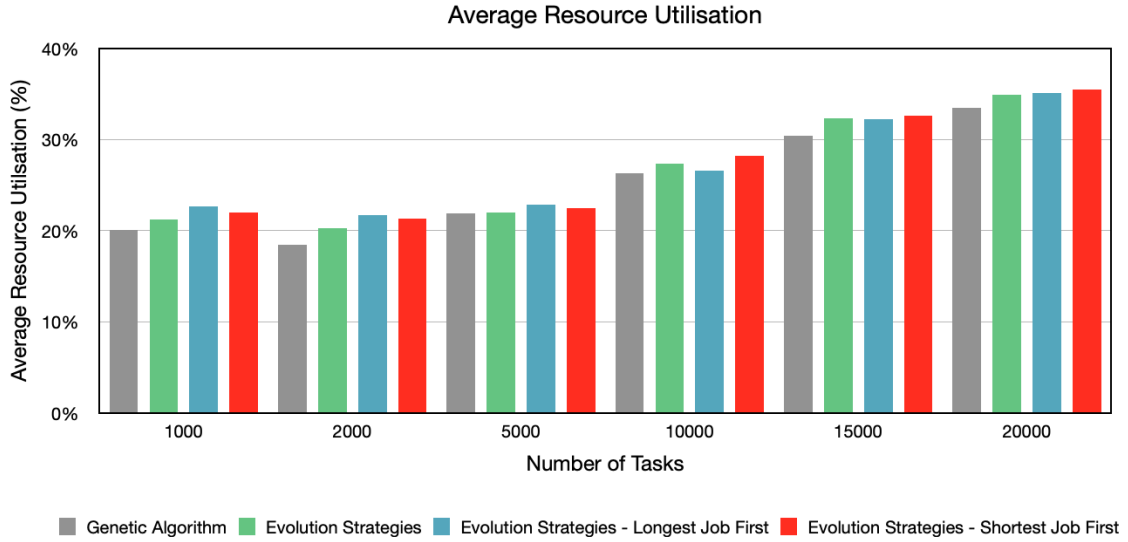
Figure 5.8: Average Resource Utilisation.

Throughput is a vital metric for assessing the productivity of task processing that consequently affects QoS. Throughput is an important factor for analysing scalability properties under different loads. The results of all compared policies are shown in Figure 5.9. The results show that the Evolution Strategies approaches, especially the Evolution Strategies - Shortest Job First approach, improved throughput in all test scenarios. The applied approach results in increased throughput as more tasks are processed in less time and consequently has a greater average resource utilisation.

The task is a single execution unit. The task scheduling algorithms aim to reduce the total execution time on the remote compute nodes in non-preemptive mode. Tasks are executed simultaneously on VM resources decided by the scheduler. Figure 5.10 shows the measurement results, which demonstrate no significant difference in the average cloudlet execution time between the Genetic Algorithm and the Evolutionary Strategies approaches. This performance characteristic is related to data from the workload. In the case of Evolution Strategies - Longest Job First, the results are slightly higher owing to the waiting time for the allocated VM resource.
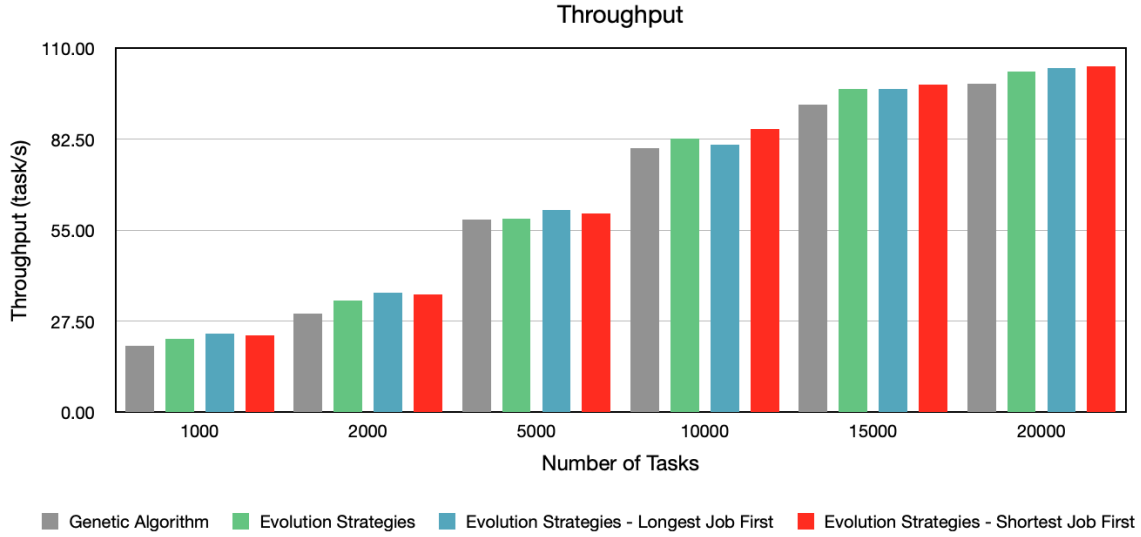
Figure 5.9: Throughput.

Figure 5.10: Average Execution Time.

Proposed algorithms reduce the degree of imbalance, as presented in Figure 5.11. Dynamic Cloud load balancing is based on a software approach. Task distribution is done on five highly scalable data centres according to their capacities. The dynamic process of task distribution is shown in Figure 5.12, Figure 5.13, and Figure 5.14. The DC 1 and DC 2 receive the most tasks for processing. Tasks are distributed based on RAM and CPU requirements. For most

test cases, Evolution Strategies with Longest Job First is more efficient in attaining the desired lower degree of imbalance. This metaheuristic prioritises the longest tasks without affecting the number and choice of VMs to be used.
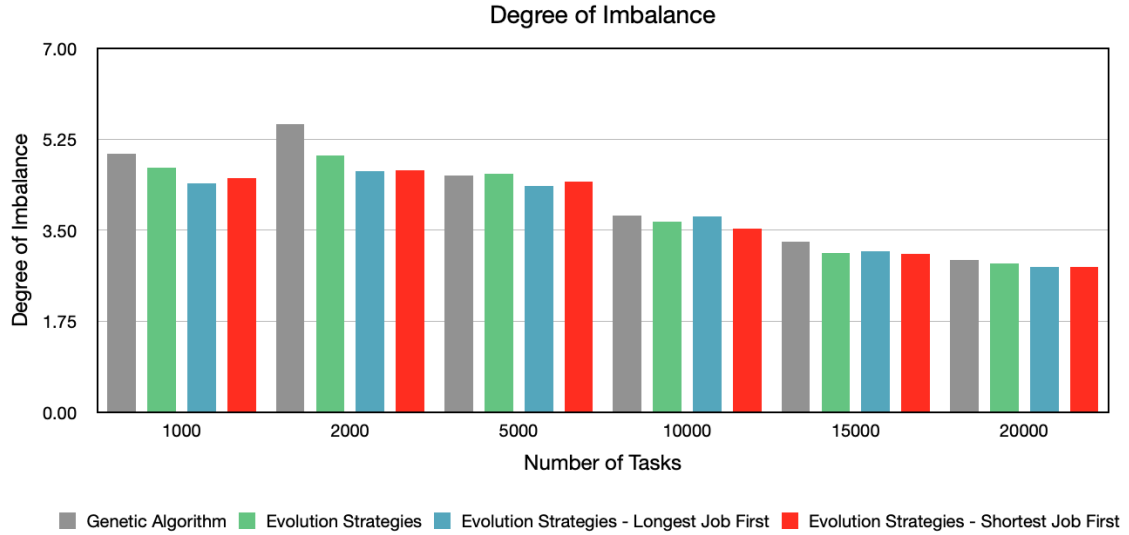
Figure 5.11: Degree of Imbalance.
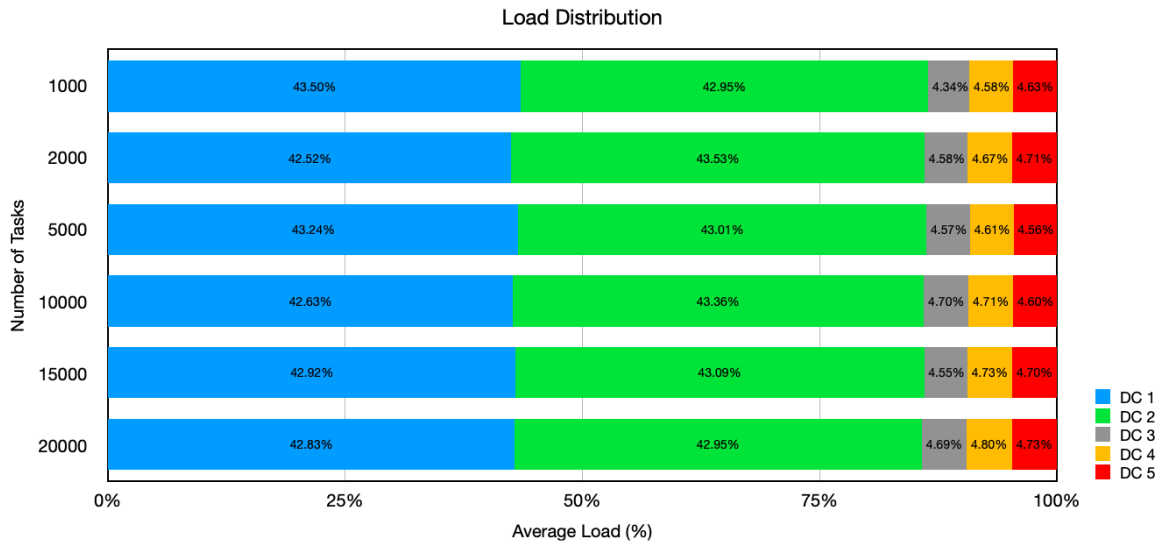
Figure 5.12: Load Distribution for (µ, λ)-Evolution Strategies algorithm.
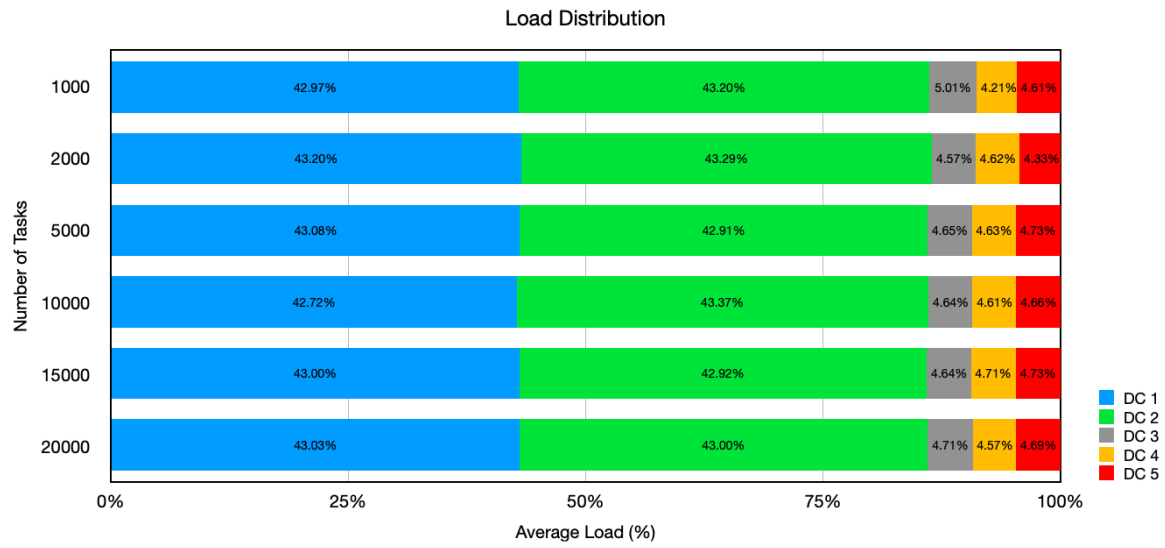
Figure 5.13: Load Distribution for (μ, λ)-Evolution Strategies algorithm with Longest Job First broker policy.

Figure 5.14: Load Distribution for (μ, λ)-Evolution Strategies algorithm with Shortest Job First broker policy.

Simulated infrastructure can easily handle the tasks with the features observed here. As a result, all submitted tasks are successfully processed. The system scales vertically and horizontally for the changed volume of tasks. Evolution Strategies approaches dynamically balance task allocation to Cloud resources and exhibit similar scalability performance, as seen in Figure 5.15, Figure 5.16, and Figure 5.17. These software-based task allocation policies realised through a central broker achieve dynamic provisioning and periodic scaling of VM instances.

Table 5.1 highlights and shows the number of active VM instances over time in the case of a workload with 1000 tasks and the case of a workload with 10000 tasks. Applied task scheduling algorithms based on the Evolution Strategies metaheuristic tend to scale at the VM level. Scalability is analysed from the aspect of the number of VMs required over time. Applied scheduling policies effectively scale resources on-demand in response to workload changes and with respect to the predefined metrics. As a result, the number of virtual resources grows and reduces over time and tracks the progress of tasks during the time for various workloads.
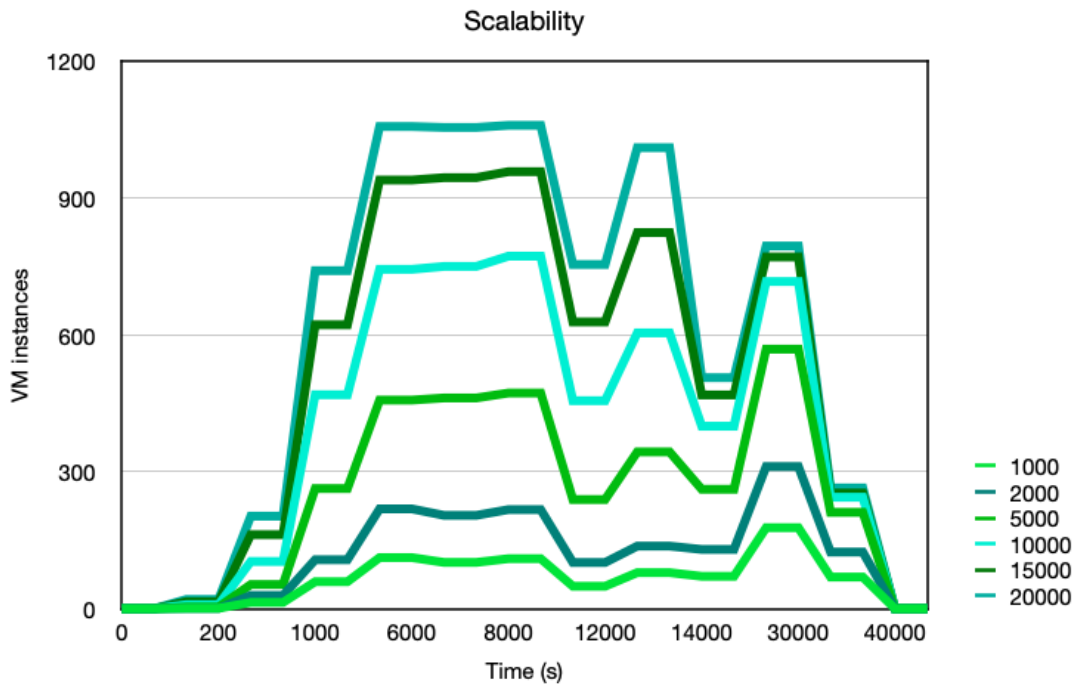
Figure 5.15: Scalability of (μ, λ)-Evolution Strategies algorithm.

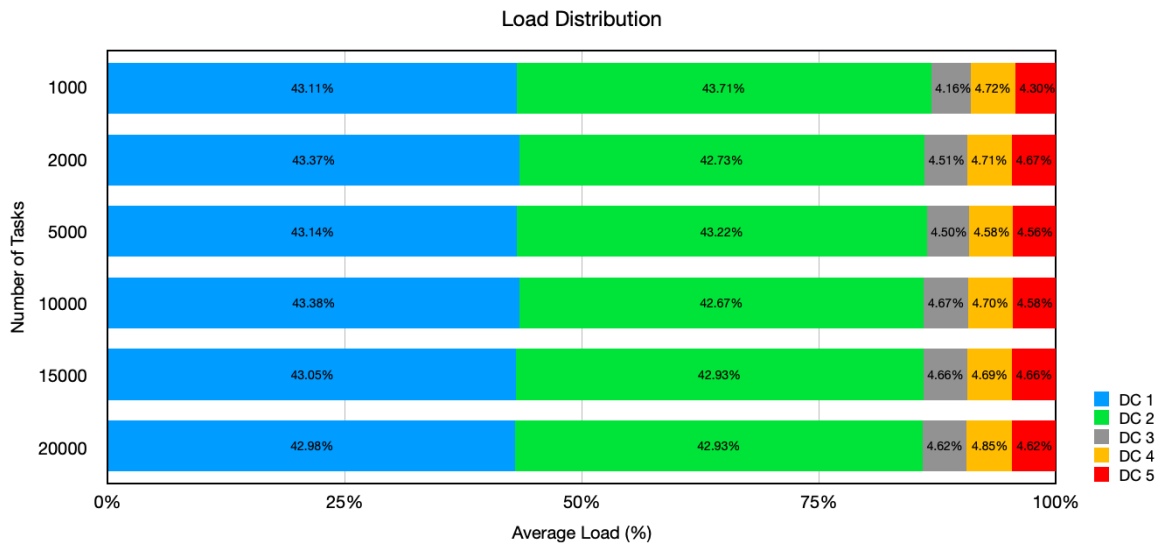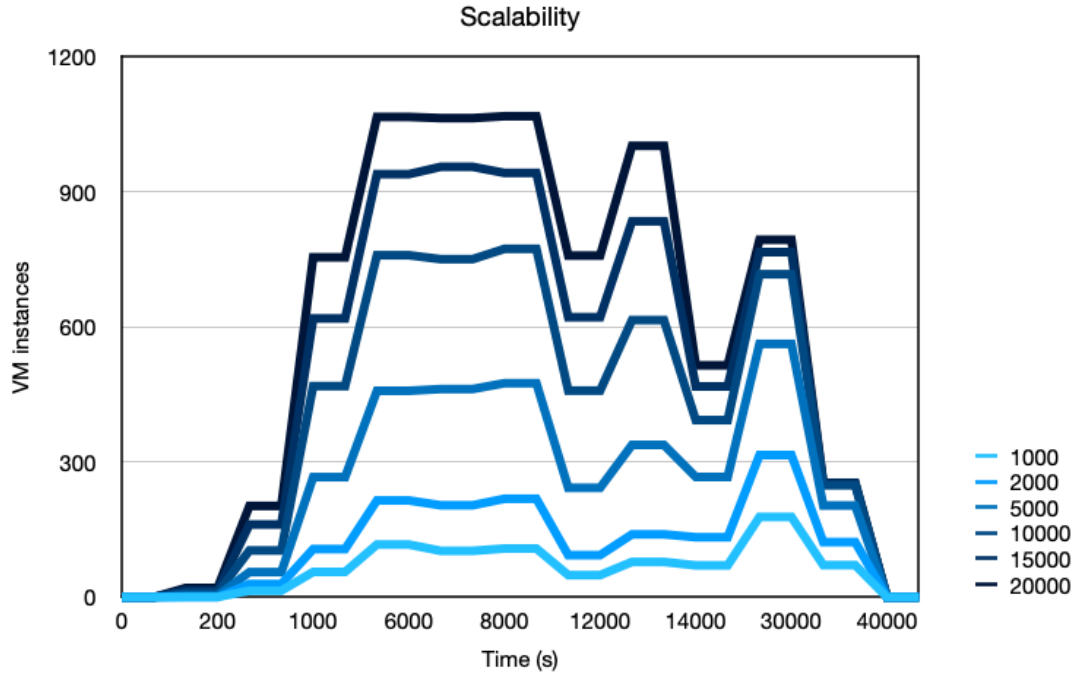*Figure 5.16: Scalability of (μ, λ)-Evolution Strategies algorithm with Longest Job First broker policy.*
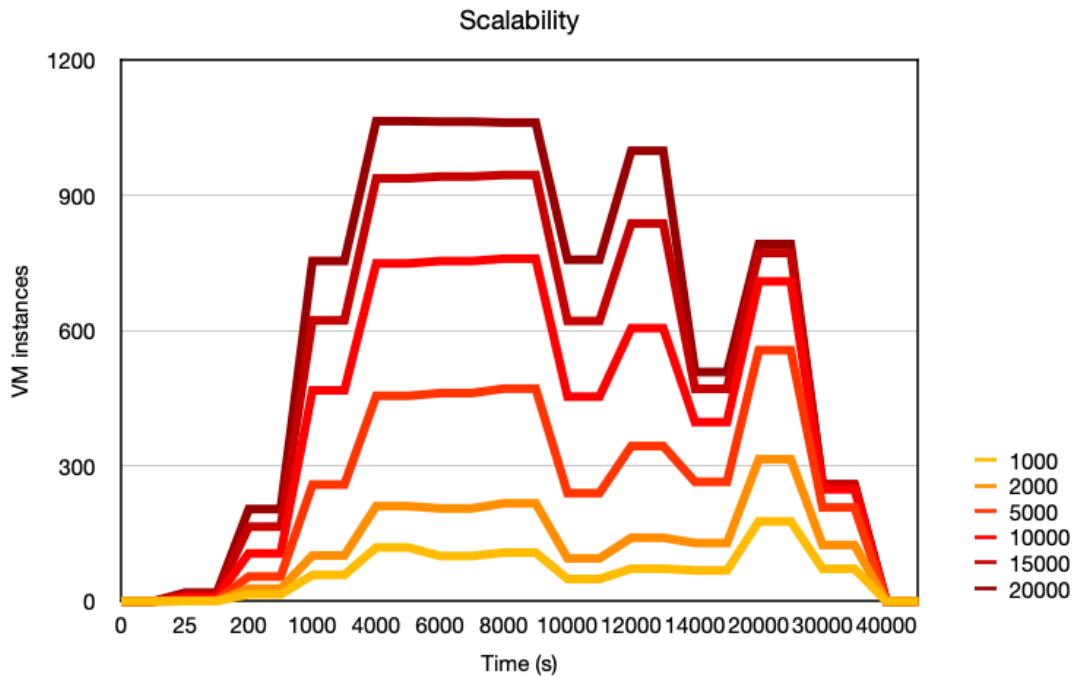
Figure 5.17: Scalability of (μ, λ)-Evolution Strategies algorithm with Shortest Job First broker policy.

Table 5.1: Number of active VM instances over time for 1000 and 10000 tasks.

Number of tasks Time [s]	1000			10000		
	ES	ES - LJF	ES - SJF	ES	ES - LJF	ES - SJF
0	0	0	0	0	0	0
25	1	1	1	9	10	9
200	14	15	16	104	104	106
1000	59	56	59	469	469	468
4000	112	117	120	744	759	749
6000	102	103	101	751	750	754
8000	110	108	108	773	773	760
10000	49	49	50	456	459	454
12000	79	78	73	605	615	606
14000	71	71	69	400	393	397
20000	178	178	177	718	717	709
30000	69	71	72	244	249	248
40000	0	0	0	0	0	0

71

5.4   Discussion

The proposed metaheuristic approach to solve the optimisation problem in the Cloud environment is based on natural selection. In this simulation experiment, the three (μ, λ)-Evolution Strategies–based algorithms were tested against a Genetic Algorithm. A simulated Cloud system is characterised by heterogeneity. A metaheuristic for the dynamic allocation of resources needed to process tasks has to find a near-optimal solution in a reasonable computation time. Resource-intensive Monte Carlo tasks are the main focus of the experiments. These tasks are CPU and RAM-bound and long-running.

The optimisation goals of adaptive algorithms have been achieved. The results obtained using Evolutionary Strategies-based approaches are better compared with results obtained using the Genetic Algorithm under the same conditions. The Evolutionary Strategies-based approaches achieved a smaller makespan, higher resource utilisation, increased throughput, similar average execution time, and reduced resource utilisation imbalances. In the Evolution Strategies with Longest Job First algorithm, the waiting time is higher as longer tasks have execution priority. Meanwhile, tasks with smaller instruction lengths are waiting for free resources. The Evolution Strategies with Shortest Job First algorithm minimises the makespan, increases throughput, and efficiently uses VM resources in most cases. It can be deduced from an analysis of resource utilisation that the data centre load and the number of VM instances dynamically follow the number of tasks to be executed.

The Evolution Strategies metaheuristic enables an orchestrated and adaptive task scheduling. The distribution of tasks in the workload affects the overall performance of task scheduling algorithms. By analysing the size distribution of tasks in the workload, it can be concluded that the characteristics and number of tasks, along with the type of VMs, affect the success of the task scheduling algorithms and their differences for specific metrics. The observed evaluation metrics showed that Evolution Strategies with Longest Job First performed better than Evolution Strategies and Evolution Strategies with Shortest Job First for fewer tasks in the workload. In the case of a larger number of tasks in the workload, it is evident that the performance is better for the proposed Evolution Strategies with the Shortest Job First algorithm. It happens due to the workload containing a greater proportion of longer tasks.

The evaluation of the performance data presented in this study leads to the conclusion that task scheduling based on the Evolution Strategies metaheuristic exhibits scalability characteristics. Thus, the data support the premise that the choice of task scheduling algorithm is of great importance for achieving scalability in a Cloud system based on heterogeneous resources. The demand pattern and fluctuations in created workload are met. As a result, load balance and scalability of using system resources are ensured.

6 SUMMARY AND OUTLOOK

This chapter concludes the presented study and highlights the key contributions. It also discusses and indicates opportunities for future research in this area, planned as a continuation of work on this topic.

6.1 Summary and conclusions

This thesis presented the study of scalability and processing data on the scientific Cloud motivated by observation from various perspectives, reviewing relevant literature, and experience working at CERN. The research is conducted in the context of the ALICE experiment and processing its Monte Carlo data at the Tier 2 level. These topics are becoming more relevant due to the rising volume of generated data, growing requirements for reliable and fast processing, and rapid technology development leading to the heterogeneity of system resources. The conducted research was preceded by an analysis of the scientific research published so far in this area. In general, there is a clear constant growing interest in the Cloud infrastructure, and a massive application of this technology is to be expected. Numerous organisations in a wide range of research activities want to make use of the efficiency and flexibility that Cloud computing provides to improve their productivity. It is essential to choose the best adapted models and specific solutions for following larger workloads with a higher efficiency level when scaling the heterogeneous infrastructure and different available technologies in the Cloud. Cloud computing enables the advanced automation and application of intelligent algorithms in processing collected data. Some of the research contributions are the overview and analysis of Cloud simulators as well as the overview and analysis of the concept of scalability in the thematic research domain. Scalability is a fundamental concern for a multi-site Cloud system. It has been examined how scalability is affected by resource management approaches and, in particular, by task scheduling algorithms. The newly proposed model enables central software management of heterogeneous Cloud infrastructure distributed in five data centres aimed at scalable resource utilisation. For this research, after a detailed analysis of the algorithms, the metaheuristic algorithm Evolution Strategies from the Computational Intelligence group was selected, that has not been used so far for dynamically assigning tasks to the resources of such a system. Besides Evolution Strategies, Evolution Strategies with Longest Job First approach and Evolution Strategies with Shortest Job First approach were developed to optimise the

scheduling of Monte Carlo jobs. The implemented Evolution Strategies algorithm uses the information it has about all available virtual resources and tries to allocate tasks with different resource demands to those VMs that best meet those demands. For the evaluation of task-VM pairs, the algorithm uses the proposed fitness function and selects the best solution that will result in optimal values of the monitored metrics. These metrics affect the scalable use of resources, both vertically and horizontally, depending on the number of incoming tasks arriving in the system to maintain a constant task processing time and the efficiency of the services provided by the Cloud. For the evaluation of the proposed model, it was necessary to prepare an appropriate workload for the simulation. As the emphasis was on computationally intensive tasks of the Monte Carlo production of the ALICE experiment, such a workload has been created and adapted to work in the selected Cloud simulator. A workload in SWF format with data from ALICE p-Pb production job has been created. The proposed methodology has been compared with the commonly used Genetic Algorithm. The obtained results confirmed a correlation between analysed variables and scalability. From the analysis of the research findings, it emerges that the (μ, λ)-Evolution Strategies–based approaches achieved greater overall processing effectiveness and resource utilisation and maintained a relatively constant task processing time while increasing the throughput in proportion to the increase in incoming task processing requests. Among Evolution Strategies-based approaches, Evolution Strategies with implemented broker policies stand out in achieving set goals. Finally, research results confirm that applying the proposed model based on the Evolution Strategies metaheuristic makes it possible to improve resource utilisation and achieve scalability of the heterogeneous multi-site Cloud system and satisfy the identified requirements, thus achieving the main planned contributions of the thesis.

Scientific contributions to this topic are woven and realised in papers published in scientific and international peer-reviewed publications:

Journal paper

- Scalable Management of Heterogeneous Cloud Resources Based on Evolution Strategies Algorithm

Conference papers

- Modeling and Simulation of Heterogeneous Resources in the Cloud: (Work in Progress)

- Software-Defined Storage Optimization of Distributed ALICE Resources

- Data-Intensive Computing Paradigms for Big Data

Poster

- Big Data Storage in High Energy Physics.

6.2 Outlook

This section gives an outlook on potential avenues of research that could extend the efforts outlined in this thesis to optimise scalable resource utilisation. Resource management in distributed paradigms is a contemporary problem. There is a growing trend of requests for the use of Cloud infrastructure, on which other emerging distributed paradigms also rely. Software-defined management is an aspect to be further considered in future research work in the context of data and resource sharing within multiple distributed environments. It is challenging to make a scalable architectural solution, consolidate the data centres, and control the load. Therefore, the controllers should gather all resource information and exchange relevant information.

Evolution Strategies has shown efficiency in achieving scalability in a distributed environment. It is planned to employ an Evolution Strategies-based approach as a Reinforcement Learning algorithm to find an optimal and proactive task scheduling policy in such an environment.

BIBLIOGRAPHY

[1] CERN Data Centre passes the 200-petabyte milestone, `https://home.cern/news/news/computing/cern-data-centre-passes-200-petabyte-milestone`, [Accessed: July 2019].

[2] O. Aberle et al., *High Luminosity Large Hadron Collider (HL-LHC): Technical design report*, CERN Yellow Reports: Monographs, CERN-2020-010, Geneva, 2020.

[3] P. Buncic, M. Krzewicki, and P. Vande Vyvre for the ALICE Collaboration, *Technical Design Report for the Upgrade of the Online-Offline Computing System*, Technical Design Report, CERN-LHCC-2015-006, ALICE-TDR-019, Geneva, 2015.

[4] Worldwide LHC Computing Grid, `https://wlcg-public.web.cern.ch/`, [Accessed: July 2022].

[5] Croatian Scientific and Educational Cloud (HR-ZOO), `https://www.srce.unizg.hr/hr-zoo/en`, [Accessed: Mar. 2022].

[6] S. Wenzel for the ALICE Collaboration, ALICE MC in Run3/4, *HL-LHC Detector Simulation Mini-Workshop*, Apr. 2021, `https://indico.cern.ch/event/1028379/contributions/4335577/attachments/2236252/3790340/HL-LHC_ALICE-April2021-N%20(2).pdf`, [Accessed: Apr. 2021].

[7] HNSciCloud for CERN, *Total Cost of Ownership (TCO) Study from T-Systems and RHEA*, Geneva, 2019.

[8] The ALICE Collaboration, The ALICE experiment at the CERN LHC, *Journal od instrumentation*, 3, S08002, Aug. 2008.

[9] L. Evans and P. Bryant, LHC machine, *Journal of Instrumentation*, 3, S08001, Aug. 2008.

[10] M. K. Gaillard, P. D. Grannis, and F. J. Sciulli, The standard model of particle physics, *Reviews of Modern Physics*, 71, S96, 1-25, Mar. 1999.

[11] The ATLAS collaboration, The ATLAS Experiment at the CERN Large Hadron Collider, *Journal od instrumentation*, 3, S08003, Aug. 2008.

[12] The CMS Collaboration, The CMS experiment at the CERN LHC, *Journal od instrumentation*, 3, S08004, Aug. 2008.

[13] The LHCb Collaboration, The LHCb detector at the LHC, *Journal od instrumentation*, 3, S08005, Aug. 2008.

[14] CERN experiments, `https://www.home.cern/science/experiments`, [Accessed: May 2022].

[15] Facts and figures about LHC, `https://home.cern/resources/faqs/facts-and-figures-about-lhc`, [Accessed: May 2022].

[16] High Luminosity LHC project, `https://hilumilhc.web.cern.ch/content/hl-lhc-project`, [Accessed: May 2022].

[17] ALICE detector in Run 2, `https://cds.cern.ch/record/2282027/files/ALICE_RUN2_labels_HR.png`, [Accessed: Dec. 2021].

[18] ALICE detector in Run 3, `https://cds.cern.ch/record/2263642/files/ALICE_RUN3_labels.jpg`, [Accessed: Dec. 2021].

[19] The ALICE Collaboration, Technical Design Report for the Upgrade of the ALICE Inner Tracking System, CERN-LHCC-2013-024, ALICE-TDR-017, *Journal of Physics G*, 41, 8, 087002, July 2014.

[20] The ALICE Collaboration, *Technical Design Report for the Upgrade of the ALICE Time Projection Chamber*, CERN-LHCC-2013-020, ALICE-TDR-016, Geneva, 2014.

[21] The ALICE Collaboration, *Technical Design Report to the Muon Forward Tracker*, CERN-LHCC-2015-001, ALICE-TDR-018, Geneva, 2015.

[22] The ALICE Collaboration, *Upgrade of the Readout & Trigger System*, CERN-LHCC-2013-019, ALICE-TDR-015, Geneva, 2015.

[23] T. Anticic et al. for the ALICE Collaboration, Commissioning of the ALICE data acquisition system, *Journal of Physics: Conference Series*, 119, 022006, 1-9, July 2008.

[24] R. Brun, P. Buncic, F. Carminati, A. Morsch, F. Rademakers, and K. Safarik, Computing in ALICE, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 502, 2-3, 339–346, Apr. 2003.

[25] S. Bagnasco, L. Betev, P. Buncic, F. Carminati, F. Furano, A. Grigoras, C. Grigoras P. Mendez-Lorenzo, A.J. Peters, and P. Saiz, The ALICE workload management system: Status before the real data taking, *Journal of Physics: Conference Series*, 219, 062004, 1-6, May 2010.

[26] R. Brun and F. Rademakers, ROOT — An object oriented data analysis framework, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 389, 1–2, 81-86, Apr. 1997.

[27] B. von Haller et al. for the ALICE Collaboration, The ALICE Data Quality Monitoring, *Journal of Physics: Conference Series*, 219, 022023, 1-9, May 2010.

[28] F. Carminati, A. Colla, C. Zampolli for the ALICE Collaboration,The SHUTTLE: the ALICE framework for the extraction of the conditions data, *13th International Workshop on Advanced Computing and Analysis Techniques in Physics Research (ACAT)*, Jaipur, India, Feb. 2010, `https://indico.cern.ch/event/59397/contributions/2050111/attachments/996353/1416927/CZampolli_5.pdf`, [Accessed: May 2021].

[29] D. Rohr, Overview of the ALICE O2 System, *Workshop IX on Streaming Readout 2021*, Dec. 2021, `https://indico.phy.ornl.gov/event/112/contributions/469/attachments/496/1348/2021-12-09_Streaming_Readout_Workshop_O2.pdf`, [Accessed: Dec. 2021].

[30] I. Bird et al., *LHC computing Grid: Technical Design Report*, CERN-LHCC-2005-024, LCG-TDR-001, Geneva, 2005.

[31] CERN Data Centre - Key Facts and Figures, April 2022, `https://information-technology.web.cern.ch/sites/default/files/CERNDataCentre_KeyInformation_April2022V1.pdf`, [Accessed: Apr. 2022].

[32] E. Martelli and S. Stancu, LHCOPN and LHCONE: Status and future evolution, *Journal of Physics: Conference Series*, 664, 052025, 1-7, Dec. 2015.

[33] Computing Resource Scrutiny Group reporting, `http://wlcg-rebus.cern.ch/core/pledge/list/`, [Accessed: Sept. 2022].

[34] M. Alef and I. Gable, HEP specific benchmarks of virtual machines on multi-core CPU architectures, *Journal of Physics: Conference Series*, 219, 052015, 1-8, May 2010.

[35] MonALISA framework, `http://alimonitor.cern.ch`, [Accessed: Mar. 2022].

[36] I. Legrand, H. Newman, R. Voicu, C. Grigoras, C. Cirstoiu, and C. Dobre, MonALISA: A distributed service system for monitoring, control and global optimization, *Proceedings of Science - XII Advanced Computing and Analysis Techniques in Physics Research (ACAT08)*, 070, 020, 1-16, Erice, Italy, Nov. 2008.

[37] ALICE Grid monitoring with MonALISA,`https://alien.web.cern.ch/content/alice-grid-monitoring-monalisa`, [Accessed: Jan. 2022].

[38] A. G. Grigoras, C. Grigoras, M. M. Pedreira, P. Saiz, and S. Schreiner, JAliEn – A new interface between the AliEn jobs and the central services, *Journal of Physics: Conference Series*, 523, 1, 012010, 1-7, June 2014.

[39] The ALICE data production, `https://alice-offline.web.cern.ch/Activities/Raw-Data.html`, [Accessed: Jan. 2022].

[40] F. Carminati and A. Morsch, Simulation in ALICE, *Proceedings of 13th International Conference on Computing in High Energy and Nuclear Physics (CHEP 2003)*, TUMT004, 1-6, La Jolla, California, USA, Mar. 2003.

[41] T. Sjostrand, S. Mrenna, and P. Skands, PYTHIA 6.4: Physics and manual, *Journal of High Energy Physics*, 06, 05, 026, 1-576, May 2006.

[42] J. Ranft, Dual parton model at cosmic ray energies, *Physical Review D*, 51, 1, 64-84, Jan. 1995.

[43] X.-N. Wang and M. Gyulassy, HIJING: A Monte Carlo model for multiple jet production in pp, pA and AA collisions, *Physical Review D*, 44, 11, 3501-3516, Dec. 1991.

[44] R. Brun, R. Hagelberg, M. Hansroul, and J. C. Lassalle, *Simulation program for particle physics experiments, GEANT: user guide and reference manual*, CERN-DD-78-2, Geneva, 1978.

[45] S. Agostinelli et al., Geant4 Collaboration, GEANT4 - a simulation toolkit, *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, 506, 3, 250–303, July 2003.

[46] A. Ferrari, P. R. Sala, A.Fassò, and J. Ranft, *FLUKA: a multiparticle transport code (Program version 2005)*, CERN Yellow Reports: Monographs, CERN-2005-010, SLAC-R-773, INFN-TC-05-11, Geneva 2005.

[47] P. Billoir, Track fitting with multiple scattering: A new method, *Nuclear Instruments and Methods in Physics Research*, 225, 2, 352-366, Aug. 1984.

[48] P. Loncar, Data-intensive computing paradigms for big data, *Proceedings of the 29th DAAAM International Symposium*, 1010-1018, Zadar, Croatia, Oct. 2018.

[49] P. Mell and T. Grance, *The NIST Definition of Cloud Computing*, NIST SP 800-145, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 2011.

[50] E. Simmon, *Evaluation of Cloud Computing Services Based on NIST SP 800-145*, NIST Special Publication 500-322, National Institute of Standards and Technology, Gaithersburg, Maryland, USA, 2018.

[51] M. De Donno, K. Tange, and N. Dragoni, Foundations and evolution of modern computing paradigms: cloud, IoT, edge, and fog, *IEEE Access*, 7, 150936-150948, Oct. 2019.

[52] C. Stergiou, K. E. Psannis, B.-G. Kim, and B. Gupta, Secure integration of IoT and cloud computing, *Future Generation Computer Systems*, 78, 3, 964-975, Jan. 2018.

[53] R. Chaudhary, N. Kumar, and S. Zeadally, Network service chaining in fog and cloud computing for the 5G environment: Data management and security challenges, *IEEE Communications Magazine*, 55, 11, 114-122, Nov. 2017.

[54] B. Varghese and R. Buyya, Next generation cloud computing: New trends and research directions, *Future Generation Computer Systems*, 79, 3, 849-861, Feb. 2018.

[55] S. Crago, K. Dunn, P. Eads, L. Hochstein, D.-I. Kang, M. Kang, D. Modium, K. Singh, J. Suh, and J. P. Walters, Heterogeneous cloud computing, *IEEE International Conference on Cluster Computing*, 378–385, Austin, Texas, USA, Sept. 2011.

[56] S. P. Crago and J. P. Walters, Heterogeneous cloud computing: The way forward, *Computer*, 48, 1, 59–61, Jan. 2015.

[57] B. S. Gigler, A. Casorati, and A. Verbeek, *Financing the future of supercomputing: How to increase investments in high performance computing in Europe*, European Investment Bank, June 2018.

[58] M. A. S. Netto, R. N. Calheiros, E. R. Rodrigues, R. L. F. Cunha, and R. Buyya, HPC Cloud for scientific and business applications: Taxonomy, vision, and research challenges, *ACM Computing Surveys*, 51, 1, 8, 1-29, Jan. 2019.

[59] G. Guidi, M. Ellis, A. Buluç, K. Yelick, and D. Culler, 10 years later: Cloud computing is closing the performance gap, *Companion of the ACM/SPEC International Conference on Performance Engineering (ICPE '21)*, 41–48, Virtual Event, France, Apr. 2021.

[60] M. D. Hill, What is scalability?, *ACM SIGARCH Computer Architecture News*, 18, 4, 18–21, Dec. 1990.

[61] A. B. Bondi, Characteristics of scalability and their impact on performance, *Proceedings of the 2nd international workshop on Software and performance (WOSP '00)*, 195–203, Ottawa, Ontario, Canada, Sept. 2000.

[62] C. Weinstock and J. Goodenough, *On system scalability*, Technical Note CMU/SEI-2006-TN-012, Software Engineering Institute, Carnegie Mellon University, Pittsburgh, Pennsylvania, USA, 2006.

[63] X.-H. Sun and D. T. Rover, Scalability of parallel algorithm-machine combinations, *IEEE Transactions on Parallel and Distributed Systems*, 5, 6, 599-613, June 1994.

[64] A. Y. Grama, A. Gupta, and V. Kumar, Isoefficiency: measuring the scalability of parallel algorithms and architectures, *IEEE Parallel & Distributed Technology: Systems & Applications*, 1, 3, 12-21, Aug. 1993.

[65] X. -H. Sun, Y. Chen, and M. Wu, Scalability of heterogeneous computing, *International Conference on Parallel Processing (ICPP'05)*, 557-564, Oslo, Norway, June 2005.

[66] Y. Chen and X. -h. Sun, STAS: A scalability testing and analysis system, *IEEE International Conference on Cluster Computing*, 1-10, Barcelona, Spain, Sept. 2006.

[67] M. Becker, S. Lehrig, and S. Becker, Systematically deriving quality metrics for cloud computing systems, *Proceedings of the 6th ACM/SPEC International Conference on Performance Engineering (ICPE '15)*, 169–174, Austin, Texas, USA, Jan. 2015.

[68] S. Lehrig, H. Eikerling, and S. Becker, Scalability, elasticity, and efficiency in cloud computing: A systematic literature review of definitions and metrics, *Proceedings of the 11th International ACM SIGSOFT Conference on Quality of Software Architectures (QoSA)*, Montreal, Canada, 83–92, May 2015.

[69] VMware, Cloud scalability, *https://www.vmware.com/topics/glossary/content/cloud-scalability.html*, [Accessed: Jan. 2022].

[70] A. Al-Said Ahmad and P. Andras, Scalability analysis comparisons of cloud-based software services, *Journal of Cloud Computing*, 8, 10, 1-17, July 2019.

[71] N. R. Herbst, S. Kounev, and R. Reussner, Elasticity in cloud computing: What it is, and what it is not, *10th International Conference on Autonomic Computing*, 23-27, San Jose, California, USA, June 2013.

[72] A. Weber, N. Herbst, H. Groenda, and S. Kounev, Towards a resource elasticity benchmark for cloud environments, *Proceedings of the 2nd International Workshop on Hot Topics in Cloud service Scalability (HotTopiCS '14)*, 5, 1-8, Dublin, Ireland, Mar. 2014.

[73] R. Han, L. K. John, and J. Zhan, Benchmarking big data systems: A review, *IEEE Transactions on Services Computing*, 11, 3, 580-597, 1, May-June 2018.

[74] P. Jogalekar and M. Woodside, Evaluating the scalability of distributed systems, *IEEE Transactions on Parallel and Distributed Systems*, 11, 6, 589-603, June 2000.

[75] R. W. Watson, High performance storage system scalability: architecture, implementation and experience, *22nd IEEE / 13th NASA Goddard Conference on Mass Storage Systems and Technologies (MSST'05)*, 145-159, Monterey, SAD, Apr. 2005.

[76] D. Gudu, M.Hardt, and A. Streit, Evaluating the performance and scalability of the Ceph distributed storage system, *IEEE International Conference on Big Data*, 177-182, Washington, District of Columbia, USA, Oct. 2014.

[77] K. Hwang, Y. Shi, and X. Bai, Scale-out vs. Scale-up techniques for cloud performance and productivity, *IEEE 6th International Conference on Cloud Computing Technology and Science*, 763-768, Singapore, Dec. 2014.

[78] L. Duboc, D. Rosenblum, and T. Wicks, A framework for characterization and analysis of software system scalability, *Proceedings of the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on the foundations of software engineering (ESEC-FSE '07)*, 375–384, Dubrovnik, Croatia, Sept. 2007.

[79] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, On scalability of software-defined networking, *IEEE Communications Magazine*, 51, 2, 136-141, Feb. 2013.

[80] Y. A. Bangash, H. Abbas, W. Iqbal, M. M. Z. M. Khan, B. Rauf, and H. Afzal, Delay reduction through optimal controller placement to boost scalability in an SDDC, *IEEE Systems Journal*, 14, 3, 4489-4499, Sept. 2020.

[81] M. Karakus and A. Durresi, A survey: Control plane scalability issues and approaches in software-defined networking (SDN), *Computer Networks*, 112, 279-293, Jan. 2017.

[82] K. Alwasel, R. N. Calheiros, S. Garg, R. Buyya, M. Pathan, D. Georgakopoulos, and R. Ranjan, BigDataSDNSim: A simulator for analyzing big data applications in software-defined cloud data centers, *Journal of Software: Practices and Experience*, 51, 5, 893–920, Oct. 2021.

[83] B. Wickremasinghe, R. N. Calheiros, and R. Buyya, CloudAnalyst: A CloudSim-based visual modeller for analysing cloud computing environments and applications, *24th IEEE International Conference on Advanced Information Networking and Applications*, 446-452, Perth, Australia, Apr. 2010

[84] A. W. Malik, K. Bilal, K. Aziz, D. Kliazovich, N. Ghani, S. U. Khan, and R. Buyya, CloudNetSim++: A toolkit for data center simulations in OMNET++, *11th Annual High Capacity Optical Networks and Emerging/Enabling Technologies (Photonics for Energy)*, 104-108, Charlotte, North Carolina, USA, Dec. 2014.

[85] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms, *Journal of Software Practices and Experience*, 41, 1, 23–50, Jan. 2011.

[86] J. Son, A. V. Dastjerdi, R. N. Calheiros, X. Ji, Y. Yoon, and R. Buyya, CloudSimSDN: Modeling and simulation of software-defined cloud data centers, *15th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, 475-484, Shenzhen, China, May 2015.

[87] M. Tighe, G. Keller, M. Bauer, and H. Lutfiyya, DCSim: A data centre simulation tool for evaluating dynamic virtualized resource management, *8th international conference on network and service management (cnsm) and 2012 workshop on systems virtualiztion management (svm)*, 385-392, Las Vegas, Nevada, USA, Oct. 2012.

[88] S. K. S. Gupta, Rose Robin Gilbert, A. Banerjee, Z. Abbasi, T. Mukherjee, and G. Varsamopoulos, GDCSim: A tool for analyzing green data center design and resource management techniques, *International Green Computing Conference and Workshops*, 1-8, Orlando, Florida, USA, July 2011.

[89] D. Kliazovich, P. Bouvry, and S. U. Khan, GreenCloud: a packet-level simulator of energy-aware cloud computing data centers, *Journal of Supercomputing*, 62, 1263–1283, Dec. 2012.

[90] A. Núñez, J. L. Vázquez-Poletti, A. C. Caminero, G. G. Castane, J. Carretero, and I. M. Llorente, iCanCloud: A flexible and scalable cloud Infrastructure simulator, *Journal of Grid Computing*, 10, 185–209, Apr. 2012.

[91] Mininet, `http://mininet.org/`, [Accessed: Dec. 2021].

[92] Omnet++, `https://omnetpp.org/`, [Accessed: Dec. 2021].

[93] M. Kumar, S. C. Sharma, A. Goel, and S. P. Singh, A comprehensive survey for scheduling techniques in cloud computing, *Journal of Network and Computer Applications*,143, 1–33, Oct. 2019.

[94] N. Gonzalez, T. Carvalho, and C. Miers, Cloud resource management: towards efficient execution of large-scale scientific applications and workflows on complex infrastructures, *Journal of Cloud Computing*, 6, 13, June 2017.

[95] M. R. Garey and D. S. Johnson, *Computers and Intractability: A guide to the theory of NP–completeness (Series of Books in the Mathematical Sciences)*, New York, New York, USA: Freeman Company, 1979.

[96] D. C. Marinescu, *Cloud resource management and scheduling,cloud computing: Theory practice*, Ch.9, 2nd ed. Cambridge, Morgan Kaufmann, Massachusetts, USA, 2018.

[97] D. A. Shafiq, N. Z. Jhanjhi, and A. Abdullah, Load balancing techniques in cloud computing environment: A review, *Journal of King Saud University - Computer and Information Sciences*, 34, 7, 3910-3933, July 2022.

[98] M. Masdari and M. Zangakani, Efficient task and workflow scheduling in inter-cloud environments: Challenges and opportunities, *Journal of Supercomputing*, 76, 1, 499–535, Jan. 2020.

[99] P. Yi, H. Ding, and B. Ramamurthy, A tabu search based heuristic for optimized joint resource allocation and task scheduling in grid/clouds, *IEEE International Conference on Advanced Networks and Telecommuncations Systems (ANTS)*, 1–3, Kattankulathur, India, Dec. 2013.

[100] H. Alazzam, E. Alhenawi, and R. Al-Sayyed, A hybrid job scheduling algorithm based on tabu and harmony search algorithms, *Journal of Supercomputing*, 75, 12, 7994–8011, Dec. 2019.

[101] Z. Zhou, J. Chang, Z. Hu, J. Yu, and F. Li, A modified PSO algorithm for task scheduling optimization in cloud computing, *Concurrency and Computation Practice and Experience*, 30, 24, e4970, 1-11, Dec. 2018.

[102] S. Nabi, M. Ahmad, M. Ibrahim, and H. Hamam, AdPSO: Adaptive PSO-based task scheduling approach for cloud computing, *Sensors*, 22, 3, 920, 1-22, Jan. 2022.

[103] N. Mansouri, B. M. H. Zade, and M. M. Javidi, Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory, *Computers and Industrial Engineering*, 130, 597–633, Apr. 2019.

[104] M. Masdari, F. Salehi, M. Jalali, and M. Bidaki, A Survey of PSO-based scheduling algorithms in cloud computing, *Journal of Network and Systems Management*, 25, 122–158, Jan. 2017.

[105] X. Chen and D. Long, Task scheduling of cloud computing using integrated particle swarm algorithm and ant colony algorithm, *Cluster Computing*, 22, 2761–2769, Mar. 2019.

[106] X. Wei, Task scheduling optimization strategy using improved ant colony optimization algorithm in cloud computing, *Journal of Ambient Intelligence and Humanized Computing*, doi: org/10.1007/s12652-020-02614-7, 1-12, Oct. 2020.

[107] Y. Moon, H. Yu, J.-M. Gil, and J. Lim, A slave ants based ant colony optimization algorithm for task scheduling in cloud computing environments, *Human-centric Computing and Information Sciences*, 7, 28, 1-10, Oct. 2017.

[108] K. Duan, S. Fong, S. Siu, W. Song, and S. Guan, Adaptive incremental genetic algorithm for task scheduling in cloud environments, *Symmetry*, 10, 5, 168, 1-13, May 2018.

[109] R. Gulbaz, A. B. Siddiqui, N. Anjum, A. A. Alotaibi, T. Althobaiti, and N. Ramzan, Balancer genetic algorithm—A novel task scheduling optimization approach in cloud computing, *Applied Sciences*, 11, 14, 6244, 1-24, July 2021.

[110] Z. Zhou, F. Li, H. Zhu, H. Xie, J. H. Abawajy, and M. U. Chowdhury, An improved genetic algorithm using greedy strategy toward task scheduling optimization in cloud environments, *Neural Computing and Applications*, 32, 1531–1541, Mar. 2020.

[111] J. Son and R. Buyya, A taxonomy of software-defined networking (SDN)-enabled cloud computing, *ACM Computing Surveys*, 51, 3, 59, 1-36, May 2019.

[112] S. Rawas, Energy, network, and application-aware virtual machine placement model in SDN-enabled large scale cloud data centers, *Multimedia Tools and Applications*, 80, 15541–15562, Apr. 2021.

[113] H. Gonzalez Labrador, G. Alexandropoulos, E. Bocchi, D. Castro, B. Chan, C. Contescu, M. Lamanna, G. Lo Presti, L. Mascetti, J. Moscicki, P. Musset, E. Karavakis, R. Pelletier, and R. Valverde, CERNBox: the CERN cloud storage hub, *EPJ Web of Conferences*, 214, 04038, 1-8, Sept. 2019.

[114] A. Klimentov, P. Buncic, K. De, S. Jha, T. Maeno, R. Mount, P. Nilsson, D. Oleynik, S. Panitkin, A. Petrosyan, R. J. Porter, K. F. Read, A. Vaniachine, J. C. Wells, and T. Wenaus, Next generation workload management system for big data on heterogeneous distributed computing, *Journal of Physics: Conference Series*, 608, 1, 012040, 1-8, May 2015.

[115] HelixNebula, `https://www.helix-nebula.eu`, [Accessed: Jan. 2022].

[116] C. Cordeiro, A. De Salvo, A. Di Girolamo, L. Field, D. Giordano, R. Jones, and L. Villazon, Accessing commercial cloud resources within the European Helix Nebula cloud marketplace, *Journal of Physics: Conference Series*, 664, 022019, 1-8, Dec. 2015.

[117] L. Bauerdick, B. Bockelman, D. Dykstra, S. Fuess, G. Garzoglio, M. Girone, O. Gutsche, B. Holzman, D. Hufnagel, H. Kim, R. Kennedy, D. Mason, P. Spentzouris, S. Timm, A. Tiradani, E. Vaandering on behalf of the CMS Collaboration, Experience in using commercial clouds in CMS, *Journal of Physics: Conference Series*, 898, 052019, 1-8, Nov. 2017.

[118] P. Llopis, C. Lindqvist, N. Høimyr, D. van der Ster, and P. Ganz, Integrating HPC into an agile and cloud-focused environment at CERN, *EPJ Web of Conferences*, 214, 07025, 1-8, Sept. 2019.

[119] F. Bührer, A. Gamel, B. Roland, U. Schnoor, and M. Schumacher on behalf of the ATLAS Collaboration, Integration of a heterogeneous compute resource in the ATLAS workflow, *EPJ Web of Conferences*, 214, 07014, 1-9, Sept. 2019.

[120] M. J. Schnepf, R. F. von Cube, M. Fischer, M. Giffels, C. Heidecker, A. Heiss, E. Kuehn, A. Petzold, G. Quast, and M. Sauter, Dynamic integration and management of opportunistic resources for HEP, *EPJ Web of Conferences*, 214, 08009, 1-8, Sept. 2019.

[121] S. J. Chapin, W. Cirne, D. G. Feitelson, J. Patton Jones, S. T. Leutenegger, U. Schwiegelshohn, W. Smith, and D. Talby, Benchmarks and standards for the evaluation of parallel job schedulers, *in: Feitelson, D.G., Rudolph, L. (eds) Job Scheduling Strategies for Parallel Processing, JSSPP 1999, Lecture Notes in Computer Science,* 1659, 67-90, Springer, Berlin, Heidelberg, July 2000.

[122] The LCG Grid log, `https://www.cs.huji.ac.il/labs/parallel/workload/l_lcg/index.html`, [Accessed: Mar. 2022].

[123] S. Robinson, Conceptual modeling for simulation, *Proceedings of the 2013 Winter Simulation Conference: Simulation: Making Decisions in a Complex World (WSC '13)*, 377–388, Washington, District of Columbia, USA, Dec. 2013.

[124] INFN Cloud, `https://www.cloud.infn.it/`, [Accessed: Aug. 2022].

[125] OVH Cloud, `https://us.ovhcloud.com/about/company/data-centers`, [Accessed: Aug. 2022].

[126] Gaia-X, `https://gaia-x.eu/`, [Accessed: Aug. 2022].

[127] Google Cloud, `https://cloud.google.com/`, [Accessed: Aug. 2022].

[128] TOP500, `https://www.top500.org/`, [Accessed: Aug. 2022].

[129] 3rd Generation AMD EPYC Processors, `https://www.amd.com/en/processors/epyc-7003-series`, [Accessed: Mar. 2022].

[130] Amazon EC2 M5 Instances, `https://aws.amazon.com/ec2/instance-types/m5/`, [Accessed: Mar. 2022].

[131] M. Kalra and S. Singh, A review of metaheuristic scheduling techniques in cloud computing, *Egyptian Informatics Journal*, 16, 3, 275–295, Nov. 2015.

[132] S. Nabi, M. Ibrahim, and J. M. Jimenez, DRALBA: Dynamic and resource aware load balanced scheduling approach for cloud computing, *IEEE Access*, 9, 61283–61297, Apr. 2021.

[133] D. A. Shafiq, N. Z. Jhanjhi, A. Abdullah, and M. A. Alzain, A load balancing algorithm for the data centres to optimize cloud computing applications, *IEEE Access*, 9, 41731–41744, Mar. 2021.

[134] M. A. Elaziz, S. Xiong, K. P. N. Jayasena, and L. Li, Task scheduling in cloud computing based on hybrid moth search algorithm and differential evolution, *Knowledge-Based Systems*, 169, 39–52, Apr. 2019.

[135] P. Loncar and P. Loncar, Scalable management of heterogeneous cloud resources based on evolution strategies algorithm, *IEEE Access*, 10, 68778-68791, June 2022.

[136] A. P. Engelbrecht, *Computational intelligence: An Introduction, 2nd Edition*, Wiley, The Atrium, Southern Gate, Chichester, West Sussex PO19 8SQ, England, 2007.

[137] H. G. Beyer and H. P. Schwefel, Evolution strategies - A comprehensive introduction, *Natural Computing*, 1, 3–52, Mar. 2002.

[138] A. E. Eiben and J. E. Smith, Popular evolutionary algorithm variants, in *Introduction to Evolutionary Computing (Natural Computing Series), 2nd ed.*, Berlin, Germany: Springer, 99–116, 2015.

[139] Z. Michalewicz and M. Schoenauer, Evolutionary algorithms, in *Encyclopedia of Information Systems, vol. 2*, Elsevier Science, USA, 259–267, 2003.

APPENDIX

Below are tables with the raw data from the Section 5.3.

Table A.1: Makespan values.

Number of tasks	Genetic Algorithm	Evolution Strategies	Evolution Strategies - Longest Job First	Evolution Strategies - Shortest Job First
1000	51.13	45.63	42.49	43.44
2000	68.50	59.76	55.97	56.16
5000	86.24	86.24	82.13	83.54
10000	125.75	121.67	124.96	117.25
15000	163.03	153.52	154.26	152.31
20000	202.15	196.40	193.64	191.36

Table A.2: Throughput values.

Number of tasks	Genetic Algorithm	Evolution Strategies	Evolution Strategies - Longest Job First	Evolution Strategies - Shortest Job First
1000	20.12	22.08	23.71	23.20
2000	29.77	33.77	36.03	35.69
5000	58.29	58.41	61.10	60.11
10000	79.93	82.59	80.78	85.68
15000	93.00	97.81	97.78	99.08
20000	99.22	102.94	103.90	104.67

Table A.3: Average Resource Utilisation values.

Number of tasks	Genetic Algorithm	Evolution Strategies	Evolution Strategies - Longest Job First	Evolution Strategies - Shortest Job First
1000	20.10%	21.20%	22.70%	22.00%
2000	18.40%	20.25%	21.70%	21.30%
5000	21.90%	22.00%	22.80%	22.50%
10000	26.30%	27.30%	26.60%	28.20%
15000	30.40%	32.30%	32.20%	32.60%
20000	33.50%	34.90%	35.10%	35.50%

Table A.4: Average Execution Time values.

Number of tasks	Genetic Algorithm	Evolution Strategies	Evolution Strategies - Longest Job First	Evolution Strategies - Shortest Job First
1000	7.56	7.57	7.60	7.59
2000	7.71	7.74	7.75	7.75
5000	7.12	7.12	7.14	7.13
10000	6.85	6.85	6.86	6.85
15000	6.89	6.91	6.91	6.91
20000	7.12	7.09	7.12	7.10

Table A.5: Degree of Imbalance values.

Number of tasks	Genetic Algorithm	Evolution Strategies	Evolution Strategies - Longest Job First	Evolution Strategies - Shortest Job First
1000	4.96	4.70	4.40	4.50
2000	5.54	4.94	4.63	4.64
5000	4.55	4.59	4.36	4.44
10000	3.79	3.66	3.76	3.53
15000	3.28	3.07	3.09	3.05
20000	2.93	2.86	2.79	2.79

Table A.6: Number of active VM instances over time for the Evolution Strategies algorithm.

Number of tasks Time [s]	1000	2000	5000	10000	15000	20000
0	0	0	0	0	0	0
25	1	1	4	9	15	20
200	14	29	53	104	162	203
1000	59	107	264	469	623	741
4000	112	219	458	744	940	1058
6000	102	205	462	751	945	1056
8000	110	217	473	773	959	1060
10000	49	101	239	456	629	755
12000	79	137	344	605	825	1011
14000	71	130	262	400	469	507
20000	178	312	569	718	771	795
30000	69	124	211	244	254	265
40000	0	0	0	0	0	0

Table A.7: Number of active VM instances over time for the Evolution Strategies algorithm with Longest Job First broker policy.

Number of tasks Time [s]	1000	2000	5000	10000	15000	20000
0	0	0	0	0	0	0
25	1	0	4	10	15	21
200	15	29	56	104	161	203
1000	56	107	267	469	619	754
4000	117	215	458	759	939	1066
6000	103	204	462	750	955	1063
8000	108	219	475	773	942	1068
10000	49	93	243	459	622	758
12000	78	140	338	615	834	1002
14000	71	133	267	393	468	515
20000	178	316	562	717	766	793
30000	71	122	204	249	253	254
40000	0	0	0	0	0	0

Table A.8: Number of active VM instances over time for the Evolution Strategies algorithm with Shortest Job First broker policy.

Number of tasks / Time [s]	1000	2000	5000	10000	15000	20000
0	0	0	0	0	0	0
25	1	1	3	9	15	19
200	16	28	55	106	166	205
1000	59	102	259	468	623	755
4000	120	211	456	749	938	1064
6000	101	206	462	754	942	1063
8000	108	217	472	760	945	1061
10000	50	95	240	454	622	757
12000	73	141	344	606	838	999
14000	69	129	265	397	471	508
20000	177	315	557	709	772	792
30000	72	125	208	248	260	258
40000	0	0	0	0	0	0

Curriculum Vitae

Petra Lončar was born in Split, Croatia, where she finished elementary school and I Gymnasium Split. She received her bachelor's degree in electrical engineering and information technology and her master's degree in electronics and computer engineering from the Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture (FESB), University of Split, Croatia, in 2012 and 2014, respectively. During her graduate studies, she resided in Clermont-Ferrand, France, where she completed her master's thesis at Blaise Pascal University. After her studies, from 2014 to 2015, she was employed by Ericsson Nikola Tesla, Zagreb, Croatia as an IP Multimedia Subsystem (IMS) Solution Integrator. She worked on several big projects abroad, in Germany, England, and Croatia. Since 2015, she has been working as a Research and Teaching Assistant with the FESB, University of Split, at the Department of Electronics and Computing, Chair of Computer Architecture and Operating Systems. In the same year, she enrolled in the Postgraduate Study of Electrical Engineering and Information Technology at FESB. Since 2016, she has been a member of the ALICE Collaboration at CERN. During ALICE Run 2 data taking period, she participated in research tasks on Data Quality Monitoring (DQM) and Shift Leader in Matters of Safety (SLIMOS) at CERN in Switzerland. She completed the Service Task at CERN as a doctoral student in the ALICE Collaboration. In 2018, as a DQM expert, she participated in shifts during the Pb-Pb period, particularly important for the ALICE experiment. She trained as the ALICE exhibition guide. Petra participated in the international conference "2018 LHC Days in Split" as a member of the organising committee. From 2022, Petra is ALICE Junior Ambassador of Croatia. She is a member of the IEEE organisation.

Petra Lončar's research interests include Big Data, data science, Cloud computing, and scalable resource management. She actively speaks English, French, and Italian. She specialised in numerous scientific and professional workshops in the country and abroad. To date, she has authored scientific papers published in a top-tier peer-reviewed journal and international conferences. A complete list of her publications is available via the link: `https://www.bib.irb.hr/pretraga?operators=and|32984|text|profile`. She was awarded the City of Split scholarship for academic excellence. She received the Dean's Commendation and the Rector's Award for Excellence during her graduate studies. She received the University of Split award for contributions to the development and recognition of the University of Split for cooperation with CERN.

Životopis

Petra Lončar rođena je u Splitu u Hrvatskoj gdje je završila osnovnu školu i I. gimnaziju Split. Na Fakultetu elektrotehnike, strojarstva i brodogradnje (FESB) Sveučilišta u Splitu 2012. godine stječe akademski naziv prvostupnice inženjerke elektrotehnike i informacijske tehnologije, a 2014. godine akademski naziv magistre inženjerke elektronike i računalnog inženjerstva. Diplomski rad izradila je na Sveučilištu Blaise Pascal tijekom studijskog boravka u Clermont-Ferrandu u Francuskoj. Nakon studija, od 2014. do 2015. bila je zaposlena u kompaniji Ericsson Nikola Tesla u Zagrebu kao IP Multimedia Subsystem (IMS) solution integrator. Radila je na više velikih projekata u Njemačkoj, Engleskoj i Hrvatskoj. Od 2015. godine radi kao asistent u istraživanju i nastavi na Zavodu za elekroniku i računarstvo na FESB-u, na Katedri za arhitekturu računala i operativne sustave kada upisuje Poslijediplomski studij elektrotehnike i informacijske tehnologije. Od 2016. godine članica je ALICE kolaboracije na CERN-u. Sudjelovala je na znanstveno-istraživačkim zadacima Data Quality Monitoring (DQM) i Shift Leader in Matters of Safety (SLIMOS) tijekom ALICE Run 2 perioda prikupljanja podataka na CERN-u u Švicarskoj. Uradila je CERN Service work zadatak kao doktorski student u ALICE kolaboraciji. U 2018. godini u ulozi DQM eksperta sudjeluje na smjenama tijekom Pb-Pb perioda koji je od posebne važnosti za ALICE eksperiment. Educirala se za vodiča i prezentatora ALICE izložbenog postava. Sudjelovala je na međunarodnoj konferenciji „2018 LHC Days in Split" kao članica organizacijskog odbora. Od 2022. godine, Petra je ALICE Junior ambasador Hrvatske. Članica je IEEE organizacije.

Znanstveno-istraživačko područje interesa i rada Petre Lončar uključuje velike podatake, podatkovnu znanost, računalstvo u oblaku i skalabilno upravljanje resursima. Aktivno govori engleski, francuski i talijanski jezik. Usavršavala se na brojnim znanstvenim i stručnim radionicama u zemlji i inozemstvu. Autorica je znanstvenih radova objavljenih u međunarodno recenziranom časopisu i na međunarodnim konferencijama. Potpuna lista njenih publikacija dostupna je putem poveznice: `https://www.bib.irb.hr/pretraga?operators=and|32984|text|profile`. Tijekom studija bila je stipendistica grada Splita zbog osobitog uspjeha u školovanju. Dobitnica je Pohvalnice dekana i Rektorove nagrade za izvrsnost na diplomskom studiju. Za suradnju s CERN-om primila je Zahvalnicu Sveučilišta u Splitu za osobiti doprinos razvoju i prepoznatljivosti Sveučilišta u Splitu.